DOSSIER-Cloud

DEVOPS-BASED SOFTWARE ENGINEERING FOR THE CLOUD

http://www.dossier-cloud.eu

Deliverable D3.3

Survey Paper / Technical Report on DevOps Metrics and Measurements

Document details:

Editor :	Panayiotis Christodoulou
Contributors :	Andreas Andreou, Dimitris Christodoulou, Kyriaki Antoniou, Constantinos Stylianou, Luciano Baresi, Clement Quinton, Srdjan Krstic, Mike Papazoglou, Willem-Jan van den Heuvel
Date:	22 June 2017
Version:	9.0

Document history:

Version	Date	Contributor	Comments
1.0	03/6/17	Dimitris Christodoulou	Initial document, structure and content
2.0	04/6/17	Panayiotis Christodoulou	First review
3.0	12/6/17	Mike Papazoglou, Willem-Jan van den Heuvel, Kyriaki Antoniou, Dimitris Christodoulou	Addition of new papers
4.0	15/6/17	Panayiotis Christodoulou	Corrections. Second review
5.0	16/6/17	Constantinos Stylianou	Corrections. Third review
6.0	18/6/17	Andreas Andreou, Luciano Baresi, Clement Quinton, Srdjan Krstic	Corrections. Fourth review
7.0	19/6/17	Panayiotis Christodoulou	Corrections. Fifth Review
8.0	20/6/17	Andreas Andreou	Corrections. Final review for approval
9.0	22/6/17	Luciano Baresi, Mike Papazoglou	Approved final version after minor revision

Contents

1. Introduction

Many companies that design, develop and utilize software systems make a clear separation of their information systems departments, and in most cases divide the development and operations into two different departments. Some professionals have addressed the fact that this division has had a negative impact, due to gaps in communication processes and lack of smooth collaboration between these departments, thus they introduced the DevOps concept.

Recently, there has been a shift towards the DevOps approach to software development, which focuses on a framework where software development and operation processes are continuous and inseparable. DevOps is a framework that provides a solution to the aforementioned problem by integrating developments and operations of software engineering (Erich et al., 2014). It is a new term emerging from the collision of two major trends: agile processes and the collaboration between the development and operations staff. Operations have become very important in the service-oriented world, therefore DevOps is used throughout the development lifecycle when designing, developing or operating a service. DevOps is not a process or methodology that can fit in all Software Engineering problems so it should be considered as a tool that meets the sole environment of an organization.

DevOps affects software development in the way of building software and automating the process of deployment. The software development lifecycle has now been upgraded for a new era of developing software and deployment is now conducted in a more automated fashion, thus saving time and money.

Due to the recent technological and software management advancements of this decade in many sectors of software engineering, it is significant to address the topic of DevOps. From the perspective of technological improvements, we see the advances in recent years on the hardware visualization ability to deploy new virtual machines (VMs) every second using only the hardware needed to run a specific application, without losing money or time on redundant resources. From the perspective of software management, we see that people and team management has dramatically changed. DevOps adopts an agile-like culture of software engineering; the main variation is on the fact that developers and operators are now working together in all phases of software development. One may argue that it is important to adopt DevOps as soon as possible and enjoy its benefits due to the market competition that is high and tough. If a company relies on traditional ways of software development, deployment will continue to be time-consuming resulting in the company losing money and resources.

The adoption of a new approach such as DevOps, as well as its customization to address the needs of the organization in which it is placed, require deep understanding of its main concepts, methods and tools. This understanding may be developed by monitoring this process. In this context, DevOps and its practical implications must be measured. This deliverable presents an

empirical study that outlines the most significant concepts of the scientific knowledge formed on DevOps metrics and measurements. The following research questions motivated this study:

RQ1: What DevOps metrics are reported in the current literature?

RQ2: What metrics and measurements of DevOps can be extracted from the literature of cloud-based systems?

RQ3: What are the main outcomes that can lead to better DevOps metrics?

In seeking answers to these questions, this study presents the methodology used for identifying the metrics and measurements utilized in DevOps, as well as an overview of the findings for further future research.

The remainder of this study is structured as follows: Section 2 describes the methodology used for the identification of the relevant material to identify metrics and measurements, while section 3 lists the main DevOps metrics. Section 4 reports the main outcomes of recent studies and, finally, section 5 closes the deliverable with some conclusions.

2. Methodology

According to the literature there are a limited number of studies that present DevOps metrics, with most of them simply based on the specific nature of companies/organizations. This section is divided into two main parts and presents the methodology used to identify the DevOps metrics and measurements from current literature and using the literature conducted on cloud-based systems.

In order to answer the aforementioned research questions, we applied a methodology to gather enough material to justify the positions and statements made; this methodology is described in this section.

For the first step, we followed the guidelines for a systematic literature review proposed by Kitchenham (2007). Although a systematic literature review is outside the scope of this work, this helped us to organize the process of finding and classifying relevant works. We searched for articles indexed in Scopus, Science Direct, IEEE Xplore, ACM Digital Library, SpringerLink, Google Scholar and Wiley Online. The search strings used were “DevOps” and “developers and operations”, “development and operations”, “cloud”, “cloud computing” and “cloud infrastructure”, combined with “metrics”, “parameters”, “measurement” in all possible combinations. The search comprised articles published up to and including 2016. As DevOps is a very recent topic, we considered both journal and conference articles. Finally, we excluded duplicate papers from the results generated by search engines and databases. After these steps, the initial collection consisted of 87 potentially relevant works. Then, we performed a detailed,

qualitative analysis by reading these papers in order to identify and suppress both different papers of the same authors/group incrementally reporting their results, and works that used the term “DevOps" with a different meaning. From this analysis we used a filtering process based on a set of four different types of criteria on the initial list of papers. This set of criteria consisted of the type of the paper, publication year, publication venue and number of citations. We accepted only scientific papers published in recognized venues with a significant number of citations. The final set of papers was organized into several categories and these are presented in the section followed.

2.1 DevOps Metrics from Literature

The first part of the study concentrates more on the limited DevOps metrics literature and presents a selection of works conducted from various companies/organizations. Most of these studies outline a list of DevOps metrics based on individual experiences.

We first identified a list of only five possible papers as there is lack of extensive scientific literature on the topic. We filtered them based on various factors, such as top scientific venues/journals, and number of citations, to narrow down this list and leave only three relevant papers. The main DevOps metrics identified in these papers are described in details in section 3.

2.2 DevOps Metrics from Cloud-based Systems Literature

In the second part we used the recent literature conducted on cloud computing. Specifically, we examined the factors that influence a cloud-based environment and tried to match those factors with DevOps metrics. We first identified a list of 87 possible papers and then we filtered them as previously based on the scientific venues published, the publication year and the number of citations, to construct a final list of 39 papers.

Figure 1 presents the number of cloud-based papers per year (2010-2016) that were examined in this part of our study. The most significant findings are also described in the next section.

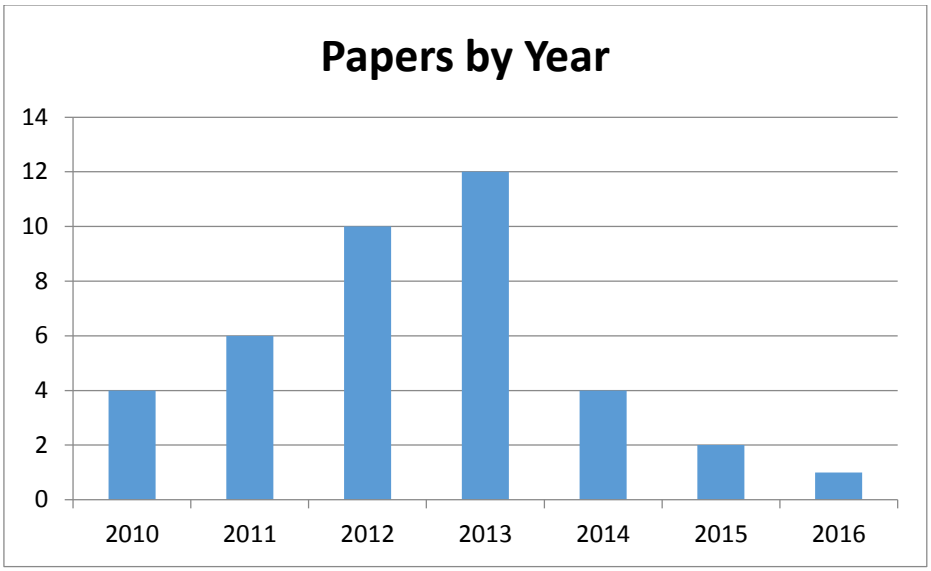

Figure 1. Number of papers examined by year of publication for the cloud-based systems literature.

3. Findings

3.1 DevOps Metrics from Literature

The work presented by Chakravarty (2014) outlines a number of metrics that are used to track DevOps success. The metrics mentioned below were identified from IBM teams upon transition of their development process from agile methodologies to DevOps. These metrics will be used throughout this deliverable to aid the quantification of DevOps properties that need to be measured.

1. **Deployment frequency**: How often is the code being deployed and new code reaches customers? This metric should trend up or remain stable from week to week.
   Example: Twice a week, 50 times a day.
2. **Change volume**: For each deployment, how many user stories and new lines of code were shipped?
   Example: Three new features per day, average 500 lines of new code per week. Another parameter to consider in addition to this volume is **complexity of change** to give a measure of the difficulty introduced by a change.
3. **Lead Time (from development to deployment):** How long does it take on average to get the code from the development team completed, subjected to a cycle of Alpha and Beta testing, and reaching to 100% deploy and upgrade on production? Lead time should be reduced as the team gets a better hold of the lifecycle.
4. **Percentage of failed deployments:** What percentage of deployments failed causing an outage or a negative user reaction? This metric should decrease over time.

Example: 9% deployments failed this month as opposed to 15% last month. This metric should be reviewed in combination with the change in volume. If the changed volume is low or remained stable but the percent of failed deployments is increased, then there may be a dysfunction somewhere.

5. **Mean-time-to-recovery:** When was a fail observed and how long did it take to recover? This is a true indicator of how well a changes is handled and should ideally reduce over time. One may expect some spikes in this number due to complex issues not encountered before.
   Example: On average it took the team 15 minutes to resolve each failure last week and 14 minutes this week. One failure was observed each day for last week and each two days for this week.
6. **Customer Ticket Volume**: Number of alerts generated by customers to indicate issues in the service. This is a basic indicator of customer satisfaction.
   Example: 54 tickets were generated this week as opposed to 38 last week while the user volume remained steady is not a good thing.
7. **Percentage Change in User Volume**: Number of new users signing up, interacting with a service and generating traffic. As new users sign up, is the infrastructure able to handle the demand?
   Example: This week the number of customers spiked by 30% due to an external event causing volume of requests to go up.
8. **Availability**: What is the overall uptime for a service and has any Service Level Agreements (SLAs) been violated?
   Example: 99.9% uptime consistently for the last three months even with change in user volume.
9. **Performance (Response Time):** Is a service performing within the predetermined thresholds? This metric should remain stable irrespective of the percentage change in user volume or new deployments.
   Example: Sub five second response time from all geographic locations and devices.

In addition, there are also similar or additional metrics offered by other organizations/companies based on their own experience. The study described by Hewlett Packard in 2016 (Hewlett Packard, 2016) presents a new list of metrics divided into four dimensions:

1. **Velocity:** A measure of the rate at which an organization can deliver software changes, described by:
   i. Frequency of deployment: How often a new version of a specific product or service is released
   ii. Speed of deployment: How long a single deployment takes for a specific product or service, expressed in number of hours from deployment approval to running in the next environment
   iii. Speed of build verification (QA): Total time it takes for a build to go through the QA cycle
   iv. Frequency of build verification (QA): Number of builds a QA team can consume or handle in a certain period

2. **Quality:** A measure of how well the software change being delivered meets business requirements and performs to satisfaction, described by:
    i. Deployment success rate: Percentage of deployments that are considered successful after validation
    ii. Incident/defect volumes: Number of incidents and detects reposted in a specific release of a product or service
    iii. Requirements coverage ratio: Percentage of traceability between requirements and tests
3. **Productivity:** A measure of an organization's capacity and efficiency in delivering software changes, described by:
    i. Feature usage: Number or percentage of unused features in a specific product or service once in production
    ii. Mean time to restore service (MTTRS): Time to restore a service or a function when the disruption is due to defects that require a fix to be developed. In other words, once an error is found, how long does it take to develop a fix and release it to production?
4. **Security:** A measure of the vulnerabilities introduced when delivering software changes, described by:
    i. Security test pass rate: Ratio of failed versus passed static security source code scans after a successful build in a certain period of time
    ii. Code scanning detection rate: Number of security scans that come back with a problem in a certain timeframe or given a process phase, as well as the number of problems encountered.

Finally, Menzel et al. (2016) present an additional work on DevOps metrics. This work is based on the DevOps transformation of the Capgemini group that offers consulting, technology and outsourcing services. According to the authors, the DevOps framework has four main value propositions:

1. **Increased agility**: Speed to market or better-maximized agility through a constant and fully integrated deployment capability.
2. **Increased Quality**: Leads to increasing also end-user satisfaction.
3. **Improved Innovation**: Leads to increasing innovation cycles.
4. **Reduced Outages**: Up to 80% outages are change-related.

3.2 DevOps Metrics from Cloud-based Systems Literature

Unlike in the early days of computing, where there was a single computer located at a remote data center, shared by many users (companies), cloud computing offers nowadays the ability to share computing resources among many different users. Cloud computing is an Internet-based model that enables convenient, on demand and pay-per-use access to a pool of shared resources.

It is a new technology that satisfies a user's requirement for computing resources (like networks, storage, servers, services and applications), without physically acquiring them. At the same time, it reduces the overhead of the organization of marinating the large system but it has also associated risks and threats including security, data leakage, insecure interface, sharing of resources, and inside attacks (Choubey et al., 2011). By studying cloud-based systems literature, a number of works have touched upon issues related to DevOps metrics.

First, the work presented by Garg et al. (2013) proposes a framework (Service Measurement Index –SMICloud) for ranking cloud services based on a set of performance and quality of services (QoS) attributes. The authors propose an Analytical Hierarchical Process (AHP) based ranking mechanism to evaluate the cloud services based on different applications in order to create a healthy competition among cloud providers to satisfy their SLA and improve their QoS. In this paper, a list of DevOps metrics has been identified, such as lead-time (from development to deployment), percentage of failed deployments and performance (response time). In addition, Bardsiri et al. (2014) propose many QoS metrics for service vendors. Those metrics are used to facilitate the future practice and research in cloud services evaluation, as well as to spot a series of metrics that can be used in a DevOps framework, such as percentage change in user volume, percentage of failed deployments and lead-time (from development to deployment). Finally, Al-Shammari et al. (2014) investigate the measurement of quality of experience (QoE) for Software as a Service (SaaS) applications in cloud computing. The target of this work is to specify the main key performance Indicators (KPIs) that can be used for defining a metric for the QoE in cloud computing. The proposed metric is defined as a function for both QoS and SLA parameters and can also be used to identify various DevOps metrics like performance (response time), percentage change in user volume, customer ticket volume and mean-time-to-recovery.

Rajan and Jeyakrishnan (2013) describe various kinds of load balancing mechanisms and approaches in different cloud environments. A detailed list of load balancing algorithms is investigated and classified according to the type of the algorithm (either static or dynamic). In the static technique, there is usually prior knowledge and assumptions about the global status of the system, such as a job resource requirements, communication time and processing power of system nodes. This knowledge can aid in the identification of the mean-time-to-recovery and availability in DevOps environments. Moreover, the work of Chaczko et al. (2011) analyzes the load balancing in cloud computing and the applicability of using load balancing techniques to obtain measurable improvements in resource utilization. The main focus of this work is to discuss possible ways to improve the performance of cloud networks by introducing resource load balancing techniques that use the message-oriented middleware within the web service oriented model of software architecture. Based on the findings, percentage change in user volume, availability and performance (response time) DevOps metrics are all identified.

The study by Saripalli and Pingali (2011) presents a framework and a methodology to facilitate cloud adoption decision making. The goal of this work is the development of a flexible and fully quantitative decision support framework based on proven multi-attribute decision methods,

which enables decision makers to comparatively assess the relative robustness of alternative cloud adoption decisions in a defensible manner. This procedure can be used to measure deployment frequency in DevOps.

Additionally, Al-Roomi et al. (2013) compare various proposed pricing model techniques that are used in cloud computing and report the results. The main outcome of this work is that the best pricing approach includes attributes regarding the end-user, such as user satisfaction level, QoS and end-user utility. These attributes can be used to recognize various DevOps metrics such as deployment frequency, performance (response time) and percentage of failed deployments.

The work by Beloglazov et al. (2012) presents the vision, challenges, and architectural elements for energy-efficient management of cloud-computing environments. The authors conducted a research on energy-efficient computing and proposed an architectural framework and principles for energy-efficient cloud computing. Experimental results showed that this approach can lead to a substantial reduction of energy consumption in cloud data centres in comparison to static resource allocation techniques. Finally, this method can be used to compute the performance (response time) metric in a DevOps environment.

Almeida et al. (2013) outline an approach for measuring the elasticity of database systems in the cloud. The major contributions of this paper are the definition of a model with a set of metrics to measure database system elasticity from different perspectives and the evaluation of the proposed model through various scenarios. The proposed metrics can also be used in DevOps for measuring performance (response time) and percentage change in user volume. Moreover, the study by Alhamad et al.(2011) describes a trust evaluation model for Infrastructure as a Service (IaaS) that utilizes fuzzy-set theory. The proposed scheme enables cloud users to evaluate the trustworthiness of cloud services providers when they are interested in shifting their current systems to cloud data centres. The trust factors introduced in this work demonstrate the effectiveness and efficiency of the proposed system and can also be used in a DevOps environment to identify the performance (response time) rate.

Becker et al. (2015) conducted a systematic literature review to recommend scalability, elasticity, and efficiency metrics for cloud computing systems. The Goal Question Metric (GQM) plan used in this work identifies six new metrics that can also be utilized in DevOps to measure percentage change in user volume and availability.

Krebs et al. (2014) propose a series of metrics to quantify the performance isolation of shared systems. The first group of metrics is based on the impact of an increased workload on the QoS from one customer to other customers, while the second one reduces the workload of the customers working within their quota. The aforementioned metrics can be used to identify performance (response time) and percentage change in user volume in DevOps. Also, the work by Suakanto et al. (2012) presents a setup that measures the QoS received by cloud computing customers using an HTTP service that runs in a cloud computing infrastructure. The major goal of this research is to identify the relation between the number of users and the response time. The

final outcome shows that by increasing the number of users in a cloud system the average response time increases, and vice versa. Following the same methodology in a DevOps framework, the performance (response time), percentage change in user volume, availability and mean-time-to-recovery measures can be computed.

The study by Malgey and Chauhan (2016) carries out a review of the existing literature regarding security in cloud computing, not only to summarize the existing vulnerabilities and threats concerning this topic but also to identify and analyze the current state and the most important security issues. Data availability ensures continuous access to data even in the occurrence of a natural or man-made disaster or events, such as fires or power outages. Thus, this approach can also be used in DevOps for computing the percentage change in user volume, availability and mean-time-to-recovery.

Subashini et al. (2011) focus on security issues of SaaS that are based on the following key security elements: Data and network security, data locality, data integrity, data segregation, data access, authentication and authorization, data confidentiality issue, web application security, data breaches, availability, vulnerability in virtualization, backup and identity management and sign-on process. In this work, multiple users can store their data using the applications provided by SaaS and data of various users resides at the same location making them vulnerable. Metrics are proposed to deal with security issues that can also be used to measure any percentage change in user volume in a DevOps environment.

Zissis and Lekkas (2012) first evaluate cloud security by identifying unique security requirements and then try to present a solution to eliminate the potential threats. Their proposed solution, which relies upon cryptography specifically based on a Public Key Infrastructure to ensure the authentication, integrity and confidentiality of involved data and communications, can be used as a means to capture availability in DevOps environments. Moreover, the work by Ali et al. (2015) presents security issues that arise on cloud computing and outline recent solutions found in recent literature to counter security issues. Furthermore, a brief overview of security vulnerabilities in mobile cloud computing is also presented and can be used to identify availability and mean-time-to-recovery in DevOps.

Vaquero et al. (2015) analyze the security risks that multitenancy induces to established cloud systems and then review state-of-the-art solutions that address the associated risks, in order to argue that systems employing access control and encryption techniques can secure the different elements present in a virtualized (multitenant) data centre. Following the solutions presented in this work, the availability and mean-time-to-recovery in a DevOps environment can be identified. Moreover, the work by Dastjerd et al. (2012) shows how monitoring services can be described, deployed, and then executed to enforce accurate penalties by eliminating SLA failure; this approach can be used to measure the performance (response time) in a DevOps context.

The study conducted by Whaiduzzaman et al. (2014) presents a state-of-the-art survey on vehicular cloud computing and taxonomy for vehicular cloud, in which special attention has been

given to the extensive applications, cloud formations, key management, inter-cloud communication systems, and broad aspects of privacy and security issues. Additionally, the work carried out by Yan et al. (2013) also promotes the vision of vehicular clouds (VCs). The main contribution of this work is the identification and analysis of a number of security challenges and potential privacy threats in VCs. Although security issues have received attention in cloud computing and vehicular networks, authors identified security challenges that are specific to VCs. Those challenges can be determined using various metrics that can be used also in the DevOps approach, such as percentage change in user volume and performance (response time).

Tariq (2012) discusses security issues of cloud computing, while at the same time proposes basic building blocks of information security metrics for cloud computing. The proposed framework that helps cloud users to create information security metrics and analyze cloud threats can be used to identify availability in DevOps. Moreover, the work performed by Hashizume et al. (2013) aims to elaborate and analyze the numerous unresolved issues threatening cloud computing adoption and diffusion by identifying various cloud-based measures that can also be utilized in a DevOps environment, such as deployment frequency, lead time (from development to deployment) and percentage change in user volume.

The study by Mircea (2012) focuses on the assessment of data security in the cloud by presenting the perspective of the data security in the cloud from the point of view of risks. The proposed approach is oriented towards the importance of security insurance through the lifecycle of data and can be used for availability and performance (response time) purposes in DevOps. In addition, the work by Saripalli and Walters (2010) presents a quantitative risk and impact assessment framework (QUIRC), to assess the security risks associated with cloud-computing platforms. The proposed framework defines risk as a combination of the probability of a security threat event and its severity, measured with its impact. QUIRC's key advantage is its fully quantitative and iterative convergence approach, which enables stakeholders to comparatively assess the relative robustness of different cloud vendor offerings and approaches in a defensible manner. The proposed approach can measure availability, mean-time-to-recovery and performance (response time) in DevOps environments.

Tordsson et al. (2012) explore the heterogeneity that exists in cloud providers as each one offers a different infrastructure and pricing policy. The proposed approach is evaluated in a high-throughput computing cluster case study where experimental results confirm that multi-cloud deployment provides better performance and lower costs compared to the usage of a single cloud system. This methodology can be followed to compute availability and performance (response time) in DevOps. Additionally, Abuhussein et al. (2012) attempt to identify and categorize a list of attributes that reflect the various aspects of cloud security and privacy. These attributes can be used to assess and compare cloud computing services so that consumers can make well educated choices. Cloud service providers can use those attributes to build and/or offer better cloud solutions. Consequently, these attributes can be used to compute various

DevOps metrics, such as availability, performance (response time) and percentage change in user volume.

Jouini et al. (2012) illustrate the use of a cyber security metrics to define an economic security model for cloud computing systems. The authors also then suggest two cyber security measures in order to better understand system threats and, thus, propose appropriate counter measure to mitigate them. The proposed metrics can be used for the computation of the availability, performance (response time) and percentage change in user volume in a DevOps environment. Additionally, the study presented by Almorsy et al. (2011) introduces a new cloud security management framework based on improving the collaboration between cloud providers, service providers and service consumers for managing the security of the cloud platform and the hosted services. This framework is built on top of a number of security standards that assist in automating the security management process and is evaluated on a multitenant SaaS application. Metrics from the study can be used to calculate the percentage of failed deployments and performance (response time) in DevOps.

Ramgovind et al. (2010) provides an overall security perspective of cloud computing with the aim to highlight the security concerns that should be properly addressed and managed to realize the full potential of cloud computing. Likewise, the study by Xiao and Xiao (2013) provides a comprehensive review of the existing security and privacy issues in cloud environments. In this work the five most representative security and privacy attributes (i.e., confidentiality, integrity, availability, accountability, and privacy-preservability) are identified. Security measurements can lead us to compute the percentage change in user volume in DevOps.

The work by Venters and Whitley (2012) identifies a range of research questions that arise from the analysis of current literature on cloud computing and also provides a framework that consists of a technological dimension (the desire for equivalence, variety, abstraction and scalability) and a service dimension (the desire for efficiency, creativity and simplicity). By using the proposed framework, the percentage change in user volume and thus the mean-time-to-recovery in a DevOps system, can be identified. Moreover, Montes et al. (2013) provide a deep insight into cloud monitoring by detailing its characteristics and highlighting the differences between them. The main target of this work was to present a general-purpose cloud monitoring framework to cover all different cloud monitoring scenarios and to propose a generic cloud monitoring architecture. The proposed architecture can also be used to measure performance (response time) in DevOps environments.

Jennings and Stadler (2015) outline a framework for cloud resource management that is based on the available literature and then classify it into eight sub-domains related to global scheduling, local scheduling, demand profiling, utilization estimation, pricing, application scaling, workload management, cloud management, and measurement studies. Various metrics used in this study can be used to identify availability, performance (response time) and percentage change in user volume for DevOps environments.

Singh and Kumar (2012) address the problem of measuring the quality of cloud systems. Five metrics based on the availability of web applications provided by a cloud environment and their diverse facets are defined in this work. Those metrics are applied to six different cloud environments and are then compared for their quality. A cloud's worst, average and best performance is taken into consideration so that the clients can select the cloud that they really want. Finally, using several of the aforementioned metrics, the percentage of failed deployments, mean-time-to-recovery and lead-time (from development to deployment) in DevOps environments can be identified.

The work of Ward and Barker (2014) performs a survey for monitoring the tools developed before cloud computing and contemporary tools, which were not designed specifically for cloud monitoring but have related goals. In addition, it examines the socio-technical aspects of monitoring, and investigates the engineering challenges and practices behind implementing monitoring strategies for cloud computing. The main outcome of this study is that older tools, which originate from previous domains, cannot fit in cloud computing because they rely on centralized strategies, pull models, manual configuration and other anachronisms. Newer tools attempt to avoid the aforementioned issues and can be used to measure availability, mean-time-to-recovery and percentage change in user volume in a DevOps framework.

The work by Mach and Schikuta (2013) develops a comprehensive analytical model, which includes variable cost in the development and evaluation of business strategies for cloud environments that can be used for both cloud providers and cloud consumers. The major goal of the proposed model is to include all important economic fundamentals and methods. On the basis of this model, the performance (response time) in DevOps can be evaluated according to the chosen business models.

Finally, Kim et al. (2010) present a trust model for efficient reconfiguration and allocation of computing resources to satisfy various user requests. The proposed approach first collects and analyzes reliability based on historical information of servers in a cloud data centre and then prepares the best available resources for each service. Furthermore, the authors carried out additional experiments for reliability analysis using four data types. Experimental results showed that by using the proposed trust model, cloud providers can utilize their resources more efficiently to provide highly trusted services to users, as well as to identify various DevOps metrics like performance (response time) and mean-time-to-recovery.

4. Conclusions

This deliverable presents an empirical study that outlines the most significant concepts of the scientific knowledge formed on DevOps metrics and measurements. The methodology followed in this study provided the means to answer the research questions posed at the beginning of this

study: RQ1- What DevOps metrics are reported in the current literature? RQ2 - What metrics and measurements of DevOps can be extracted from the literature of cloud-based systems? RQ3 - What are the main outcomes that can lead to better DevOps metrics?

The process for the identification of relevant scientific papers on DevOps metrics was proved difficult as there are a very limited number of relevant studies published. This is aligned with Deliverable 2.3 - Survey paper / Technical report on DevOps oriented software engineering of the DOSSIER project which concludes that the relevant literature is not clear or consistent as regards the definition of the DevOps phenomenon, while the scientific or research sources are scattered and limited.

The definition of DevOps metrics and associated ways of measuring them was assisted by the study of the area of cloud computing. Since the origins of the DevOps paradigm is the agile approach and its application was primarily on distributed software systems and services, it was considered useful to address DevOps metrics starting from the cloud and then examining how and to what extent a specific metric can be applied to a DevOps context. In this case the relevant literature studied was considered sufficient to extract some useful conclusions, the most significant of which is that there is a list of metrics applied in the cloud environment that can prove useful for DevOps as well, and assist in monitoring properties like performance (i.e., response time), change in user volume and deployment frequency. Also, properties related to SLA conformance or violations can be measured but from the perspective of developers and operators rather than the end-user, something which may be considered extremely useful for maintaining high levels of customer satisfaction. Finally, based on the findings of this study, one may argue that the metrics required to quantify a DevOps framework are based on the needs of the organization in which it is placed.

The last research question has not been adequately addressed by the existing literature. This was expected since the sources were extremely limited in this respect. There is ample room for future work on the topic. DevOps is a new trend which has not been sufficiently researched thus far and therefore the associated metrics are still a grey area. What needs to be done is to devote more time to study the practical implications of such metrics in a real-world context and any attempt to quantify and assess them should definitely involve empirical experimentation in software related companies and organizations. The number and type of metrics may vary from organization to organization and therefore studies on this subject should be concentrated on defining a subset of metrics tailored to meet the needs of a particular property or parameter in the business or application context, such as performance, security availability and customer satisfaction. What would be extremely useful in practise is to have these subsets of metrics being accompanied by a set of guidelines to provide support as to which metrics to use for monitoring and assessing a particular property or parameter. This may indeed lead to better DevOps metrics or more useful exploitation of existing ones.

References

Abuhussein, A., Bedi, H., & Shiva, S. (2012, December). Evaluating security and privacy in Cloud computing services: A Stakeholder's perspective. In Internet Technology and Secured Transactions, 2012 International Conference for (pp. 388-395). IEEE.

Alhamad, M., Dillon, T., & Chang, E. (2011). A trust-evaluation metric for Cloud applications. International Journal of Machine Learning and Computing, 1(4), 416.

Ali, M., Khan, S. U., & Vasilakos, A. V. (2015). Security in Cloud computing: Opportunities and challenges. Information Sciences, 305, 357-383.

Almeida, R. F., Sousa, F. R., Lifschitz, S., & Machado, J. C. (2013). On defining metrics for elasticity of Cloud databases. In SBBD (Short Papers) (pp. 12-1).

Almorsy, M., Grundy, J., & Ibrahim, A. S. (2011, July). Collaboration-based Cloud computing security management framework. In Cloud Computing (CLOUD), 2011 IEEE International Conference on (pp. 364-371). IEEE.

Al-Roomi, M., Al-Ebrahim, S., Buqrais, S., & Ahmad, I. (2013). Cloud computing pricing models: a survey. International Journal of Grid and Distributed Computing, 6(5), 93-106.

Al-Shammari, S., & Al-Yasiri, A. (2014). Defining a metric for measuring QoE of SaaS Cloud computing. Proceedings of PGNET, 251-256.

Bardsiri, A. K., & Hashemi, S. M. (2014). Qos metrics for Cloud computing services evaluation. International Journal of Intelligent Systems and Applications, 6(12), 27.

Becker, M., Lehrig, S., & Becker, S. (2015, January). Systematically deriving quality metrics for Cloud computing systems. In Proceedings of the 6th ACM/SPEC International Conference on Performance Engineering (pp. 169-174). ACM.

Becker, M., Lehrig, S., & Becker, S. (2015, January). Systematically deriving quality metrics for Cloud computing systems. In Proceedings of the 6th ACM/SPEC International Conference on Performance Engineering (pp. 169-174). ACM.

Beloglazov, A., Abawajy, J., & Buyya, R. (2012). Energy-aware resource allocation heuristics for efficient management of data centers for Cloud computing. Future generation computer systems, 28(5), 755-768.

Chaczko, Z., Mahadevan, V., Aslanzadeh, S., & Mcdermid, C. (2011, September). Availability and load balancing in Cloud computing. In International Conference on Computer and Software Modeling, Singapore (Vol. 14).

Choubey, R., Dubey, R., & Bhattacharjee, J. (2011). A survey on Cloud computing security,

challenges and threats. International Journal on Computer Science and Engineering (IJCSE), 3(3), 1227-1231.

Dastjerdi, A. V., Tabatabaei, S. G. H., & Buyya, R. (2012). A dependency-aware ontology-based approach for deploying service level agreement monitoring services in Cloud. Software: Practice and Experience, 42(4), 501-518.

Erich, Floris, Chintan Amrit, and Maya Daneva. (2014). A mapping study on cooperation between information system development and operations." International Conference on Product-Focused Software Process Improvement. Springer International Publishing.

Garg, S. K., Versteeg, S., & Buyya, R. (2013). A framework for ranking of Cloud computing services. Future Generation Computer Systems, 29(4), 1012-1023.

Hashizume, K., Rosado, D. G., Fernández-Medina, E., & Fernandez, E. B. (2013). An analysis of security issues for Cloud computing. Journal of Internet Services and Applications, 4(1), 5.

Hewlett Packard. (2016). Measuring DevOps success How do you know DevOps is working? Watch these KPIs.

Jennings, B., & Stadler, R. (2015). Resource management in Clouds: Survey and research challenges. Journal of Network and Systems Management, 23(3), 567-619.

Jouini, M., Aissa, A. B., Rabai, L. B. A., & Mili, A. (2012). Towards quantitative measures of Information Security: A Cloud Computing case study. International Journal of Cyber-Security and Digital Forensics (IJCSDF), 1(3), 248-262.

Kim, H., Lee, H., Kim, W., & Kim, Y. (2010). A trust evaluation model for QoS guarantee in Cloud systems. International Journal of Grid and Distributed Computing, 3(1), 1-10.

Kitchenham, B.: Guidelines for performing systematic literature reviews in software engineering. In: Technical report, Ver. 2.3 EBSE Technical Report. EBSE (2007)

Krebs, R., Momm, C., & Kounev, S. (2014). Metrics and techniques for quantifying performance isolation in Cloud environments. Science of Computer Programming, 90, 116-134.

Mach, W., & Schikuta, E. (2013). Toward an economic and energy-aware Cloud cost model. Concurrency and Computation: Practice and Experience, 25(18), 2471-2487.

Malgey S. & Chauhan P. (2016). A Review on Security Issues and their Impact on Cloud Computing Environment. International Journal of Advanced Research in Computer and Communication Engineering Vol. 5, Issue 6, June 2016

Menzel, G., & Macaulay, A. (2016). DevOps - The Future of Application Lifecycle Automation.

Mircea, M. (2012). Addressing Data Security in the Cloud. World Academy of Science, Engineering and Technology, 66, 539-546.

Montes, J., Sánchez, A., Memishi, B., Pérez, M. S., & Antoniu, G. (2013). GMonE: A complete approach to Cloud monitoring. Future Generation Computer Systems, 29(8), 2026-2040.

Payal Chakravarty. (2014). https://devops.com/devops-scorecard/

Rajan, R. G., & Jeyakrishnan, V. (2013). A survey on load balancing in Cloud computing environments. International Journal of Advanced Research in Computer and Communication Engineering, 2(12), 4726-4728.

Ramgovind, S., Eloff, M. M., & Smith, E. (2010, August). The management of security in Cloud computing. In Information Security for South Africa (ISSA), 2010 (pp. 1-7). IEEE.

Saripalli, P., & Pingali, G. (2011, July). Madmac: Multiple attribute decision methodology for adoption of Clouds. In Cloud Computing (CLOUD), 2011 IEEE International Conference on (pp. 316-323). IEEE.

Saripalli, P., & Walters, B. (2010, July). Quirc: A quantitative impact and risk assessment framework for Cloud security. In Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on (pp. 280-288). Ieee.

Singh, G., & Kumar, R. (2012). Availability metrics for Cloud vibrant behaviour with benchmarks influence on diverse facets. International Journal of Software Engineering & Applications, 3(1), 101.

Suakanto, S., Supangkat, S. H., & Saragih, R. (2012). Performance measurement of Cloud Computing services. arXiv preprint arXiv:1205.1622.

Subashini, S., & Kavitha, V. (2011). A survey on security issues in service delivery models of Cloud computing. Journal of network and computer applications, 34(1), 1-11.

Tariq, M. I. (2012). Towards information security metrics framework for Cloud computing. International Journal of Cloud Computing and Services Science, 1(4), 209.

Tordsson, J., Montero, R. S., Moreno-Vozmediano, R., & Llorente, I. M. (2012). Cloud brokering mechanisms for optimized placement of virtual machines across multiple providers. Future Generation Computer Systems, 28(2), 358-367.

Vaquero, L., Rodero-Merino, L. & Moran, D. (2015). Looking the Sky: a survey on IaaS Cloud security

Venters, W., & Whitley, E. A. (2012). A critical review of Cloud computing: researching desires and realities. Journal of Information Technology, 27(3), 179-197.

Ward, J. S., & Barker, A. (2014). Observing the Clouds: a survey and taxonomy of Cloud monitoring. Journal of Cloud Computing, 3(1), 24.

Whaiduzzaman, M., Sookhak, M., Gani, A., & Buyya, R. (2014). A survey on vehicular Cloud

computing. Journal of Network and Computer Applications, 40, 325-344.

Xiao, Z., & Xiao, Y. (2013). Security and privacy in Cloud computing. IEEE Communications Surveys & Tutorials, 15(2), 843-859.

Yan, G., Wen, D., Olariu, S., & Weigle, M. C. (2013). Security challenges in vehicular Cloud computing. IEEE Transactions on Intelligent Transportation Systems, 14(1), 284-294.

Zissis, D., & Lekkas, D. (2012). Addressing Cloud computing security issues. Future Generation computer systems, 28(3), 583-592.