Leveraging Software Quality Attributes in NoSQL Database Systems

Tonata Nakasholoolo
Information Technology Department, Cape peninsula University of Technology, Cape Town, South Africa
Email: tonata93@gmail.com

Tiko Iyamu
Information Technology Department, Cape peninsula University of Technology Cape Town, South Africa
Email: iyamut@cput.ac.za

2022 5th International Conference on Information and Computer Technologies (ICICT) | 978-1-6654-6960-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICICT55905.2022.00048

***Abstract*—The increasing interest in Not Only Structured Query Language (NoSQL) database systems has pruned debates among technical and business personnel in many organisations, about its quality and value to their businesses. Also, existing work has been silent on how software engineers can gain more benefits from the diverse nature of NoSQL technologies, from software quality attributes perspective. Thus, the objectives of this study were to: (1) understand the factors that influence leveraging of software attributes to business activities; and (2) examine how software quality attributes can be leveraged with NoSQL database systems. In achieving the objectives, the qualitative methods were employed. The interpretivist approach was followed in the analysis of the data. Based on the analysis, seven factors were found to influences the quality attributes in the leveraging of the NoSQL database systems in organisations. Based on the interpretation of the findings, a leveraging model was proposed. The model is intended to provide a guide on how software quality attributes can be leveraged with NoSQL database systems in organisation. This includes to facilitate NoSQL technology selection.**

Keywords—Database, NoSQL, Qualitative methods

I Introduction

Existing databases, such as MySQL, JAVADB and OracleDB are characteristically differentiated by attributes, which include size, compatibility and integration [1]. The differentiation implies that there are gaps in attempt to deploy or make use of the databases. This paves opportunity for emergence of innovation, such as Not Only SQL, which is often refers to as "NOSQL" [2]. The NoSQL databases support horizontal scalability, high availability and elasticity, which the traditional database management systems (RDBMS) does not support. The NoSQL databases provide horizontal scalability which allows the database management systems to expand with data volumes without affecting ongoing operations [3]. Also, the NoSQL offer characteristics and cooperation between processing requirements of big data applications and the availability including scalability of computational resources [4].

The classical approach to databases is the data model [2]. NoSQL database supports four kinds of data models, namely, key-value, document-oriented, column family and graph data models [3]. Each of which provide a different approach in the way that the data is stored and retrieved in an environment [5]. Column family stores data by grouping of datasets into column families associated with a key, whilst the graph data model stores data in the form of nodes (entity) and edges (relationships between entities), the graph data model is especially suitable for applications that are focused on modelling extensive relationships between entities [2]. This is achieved through the construction of a model that will not only show these complexities but also, facilitate the process of leveraging software quality attributes in NoSQL database systems.

NoSQL is an emergent technology that requires constant development, evaluations and assessments of its deployment and use within different environments. There exists an assortment of studies and evaluations of NoSQL, but not enough to verify how suitable each database is in a specific scenario [6]. This gap continues to have an impact on how organisations deploy, use and realize their investments on NoSQL, due to lack of suitability or appropriateness. This includes leveraging software quality attributes to organizations suitability.

This research therefore focuses on bridging the gap, which will enable leveraging of software quality attributes with organizations' suitability. Thus, the objectives of the study were to: (i) examine how software quality attributes can be leveraged in NoSQL database systems; (ii) describe the quality attributes of NoSQL databases, which can fulfil organisations' suitability; and (iii) determine the factors that influence leveraging software quality attributes in NoSQL database systems.

II NOSQL As Software And Database

Software is an artefact of information and communication technology (ICT) that is often developed to act as unified system to support the needs of users. Ssoftware seeks to simplify and integrate operations, processes and information flow. The reliance on software makes it critical to leverage its use with individual and organization's purposes. However, it is never easy to achieve and articulate software leverage. This could be attributed to the variety of attributes which different software possesses [7]. There are challenges with integration mostly because of the integral difficulties of linking diverse systems and the lack of a lucid data structure connecting different systems [8]. These challenges include poor planning and design decisions from the ICT team and various applications being obtained from different vendors that make

978-1-6654-6960-9/22/$31.00 ©2022 IEEE
DOI 10.1109/ICICT55905.2022.00048

use of dissimilar operating systems and data formats which could hinder interoperability.

The NoSQL incorporates a wide range of architectures and technologies that look to solve big data performance and scalability issues that RDBMS were not designed to address [3]. Contrary to misapprehensions, NoSQL does not forbid the use of Structured Query Language (SQL). Although it is true that some NoSQL databases are completely non-relational, others merely avoid relational functionality such as join queries and fixed schemas [9]. This is evident in the different data models available amongst NoSQL systems i.e., key-value, document, column-family and graph data models [2]. Also, a vast number of NoSQL databases support flexible schemas where data may differ from instance to instance. Ppossibly the paramount strength of NoSQL databases is the ability to support horizontally scalable, distributed, fault tolerant architecture in meeting the needs of big data and big data analytics [10].

The traditional relational database (relational database management system) fell short in this regard because even though they could handle large volumes of data, it could only accommodate for limited formats. It is also vertically scalable, which proved to be a bottleneck in the management of big data [2]. NoSQL provides four kinds of data models, namely: document-oriented data stores that focus on storing data in a denormalized format such as JSON or XML to facilitate for faster retrievals; key-value data stores that make use of key value pairs to store data; graph data stores use nodes and edges to store data and the relationships between them; column family data stores group related columns into families and store them with a related key [3].

Software engineers are presented with a plethora of technologies to choose from when designing and developing applications. These decisions are primarily based on the functional and non-functional requirements of proposed applications [11]. An important consideration is how application data will be stored and managed because this ultimately facilitates information exchange in a software application to meet stakeholder needs [12]. Subsequently, software engineers are required to make data modelling and management decisions based on application requirements. In the context of data storage and management, non-functional requirements are important in choosing the right database technology [13]. The NoSQL is a technology that presents novel approaches to data storage and management. The question arises, how best software engineers can gain maximum advantage of non-functional requirements when opting to use NoSQL technology.

III Research Methodology

The objectives of this study as stated in the introduction section, were to gain better understanding of how software quality attributes can be leveraged in NoSQL database systems. Thus, we employ the qualitative methods [14], within which documentation technique was applied [15]. As shown in Table 1, the existing documents, mostly peer reviewed articles were gathered as data. Primarily, the use of documentation enables researchers to fully share knowledge and gain an understanding of what exists. Using the documentation technique, literature published between the year 2010 and 2016 will be collected. There will be four areas of focus, using the following criteria: (i) Software Quality; (ii) Software Integration; (iii) Leveraging; and (iv) NoSQL.

The interpretivist approach was followed in the analysis of the data. This means that the analysis was based on subjectivism. The interpretivism is a research paradigm that dictates that research cannot be objectively observed but by subjectively gaining "understanding, explaining and demystifying social reality" [16]. The interpretivist approach affirms that only through subjective interpretation of reality, which makes reality difficult to understand [17].

TABLE I: Frequency of Special Characters

Focus	Purpose	Source of data/Literature
Software Quality	To understand the software quality attributes in NoSQL database systems.	Distribution, Data, Deployment [18].
		Design Assistant for NoSQL Technology Selection [18].
Leverage	To understand the factors that influence leveraging of software quality.	A framework for ranking of cloud computing services [19].
		Exploiting non-functional preferences in architectural adaptation for self-managed systems [20].
		Leveraging software architectures to guide and verify the development of sense/compute/ control applications [21].
Software Integration	To examine the factors, which influence software integration in NoSQL database systems.	Integration of Heterogeneous Databases [22].
		Systems integration and collaboration in architecture, engineering, construction, and facilities management: A review [23].
		Integration and Virtualization of Relational SQL and NoSQL Systems including MySQL and MongoDB [24].
		Factors influencing efficient integration of safety systems [25].
		Quality Attribute-Guided Evaluation of NoSQL Databases: An Experience Report [26].
NoSQL	To understand the characteristics of NoSQL database systems in relation to software integration.	Architecture Knowledge for Evaluating Scalable Databases [27].
		On the potential integration of an ontology-based data access approach in NoSQL stores [28].
		Integration and virtualization of relational SQL and NoSQL systems including MySQL and MongoDB [29].

IVData Analysis

The data as contained in Table 1 was interpretively analysed. The analysis of the data was carried out in accordance with the research objectives to: (1) understand software quality attributes in NoSQL database systems; (2) understand the factors that influence leveraging of software quality; (3) examine the factors, which influence software integration in NoSQL database systems; and (4) understand the characteristics of NoSQL database systems in relation to software integration:

A. *Software quality attributes in NoSQL database systems*

From the data that is tabulated in Table 1, our analysis reveals five factors as the main attributes of software quality within NoSQL database systems. This includes availability, high performance, horizontal scalability, non-functional requirements and architecture flexibility:

Availability: Availability as the capability of a software product to perform an essential function in a certain period. Similarly, availability as the measure of how often data and applications are readily accessible when needed and the ability of the software product to fulfil requests [30]. Availability is a critical aspect of the computing environment, from both software and hardware perspectives. High availability guards an organization from lost revenues when access to data and critical business applications are disrupted [31]. However, there are levels of availability, different from one environment to another. Thus, NoSQL's level of availability in organizations needs to be defined within context and relevance. This is to ensure its contribution to the business, which defines its quality. In the context of NoSQL, high availability means that the system is tolerant of node failures and can also remain available in the event of hardware and software upgrades [32]. Thus, NoSQL's compatibility with other software and hardware is critically important for availability. Majority of problems can be categorized as unplanned outages; planned outages; backup window reduction; load balancing; and disaster recovery [31]. In achieving high availability, replication is an important technique used by NoSQL databases.

High Performance: The essence of performance has to do with the measure of how efficient and fast a system can complete computing tasks. The performance of software quantifies the effectiveness and speed of the application [33]. Performance is defined as the use of computers, networks, algorithms and parallel processing for running of software reliably, efficiently and quickly [34]. Performance varies from low to high. Arguably, every organization will prefer high performance. Software including NoSQL performances are influenced by different factors, from one environment to another. The high performance in NoSQL databases is influenced by six factors: workload; system resources; throughput; contention; latency; and response times [35].

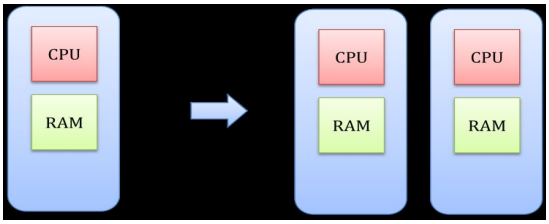

Figure 1: Horizontal scaling

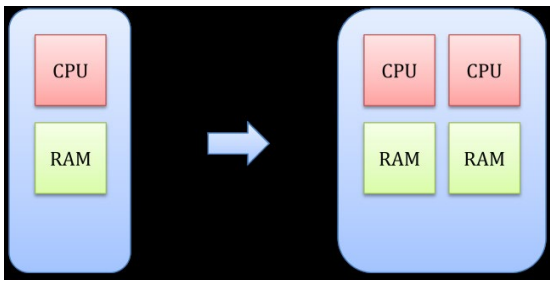

Figure 2: Vertical scaling

Horizontal Scalability: Scalability is defined as the simplicity in which a system can be adapted to fit a specification [36]. A scalable system including software is one that can accommodate increased data set and increased usage. In addition, scalability is one capabilities of a system to provide satisfactory performance under growing demands such as increased traffic and data volumes, whilst reducing the need to redesign the operational facets of the system under challenges [37]. Also, scalability could be horizontal or vertical depending on the capability of the software, as shown in Figures 1 and 2.

In achieving scalability, NoSQL database conform to horizontal scalability. This is because the NoSQL databases are horizontally scalable, while relational databases are vertically scalable. Horizontal scalability is achieved through the addition of multiple units of a small capacity to a system instead of adding a single unit of a large capacity. For example, a NoSQL database would add smaller machines (CPU & Hard disk) to accommodate a scalable system rather than increasing the size of the machine (CPU & Hard disk).

Non-functional requirements: Non-functional requirements (NFRs) do not describe what a software will do but rather how the software will do it and how the system should behave. The behaviour of software, such as NoSQL helps to define its quality within an environment. This is primarily because the business or users depend on the software behaviour in using it to carry out their processes and activities. The NFRs may include design constraints, external interface requirements and software quality attributes [38]. In essence, NFRs specify the software's quality attributes. Some worth noting in the context of software quality is the Dimensions of Software Quality – which categorizes NFRs into five and goes further to list sub characteristics in each category, as shown in Figure 3. It is evident that the use of NFRs is vital in classifying software quality in organizations.

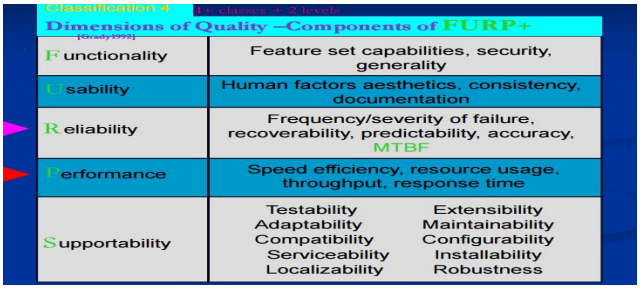

Figure 3: Dimensions of quality

Architecture flexibility: Flexibility is the ease of which software can be modified or co-exist in its use in an environment, other than those for which it was specifically designed. Flexibility plays a vital role for software quality as it supports ease of use [21]. The architecture flexibility is defined as the capability of software to rearrange its building blocks to model different architecture [39]. In a NoSQL database context, this can be seen as the ability of a specific database product to support a flexible, ease of use and implicit data modelling practices. Schema-less modelling and a range of data models to leverage contribute to the flexible NoSQL database architecture. NoSQL provides are more relaxed, implicit approach to data modelling which is commonly known as "schema-less" [40]. The semantics of the data are rooted within the "flexible connection topology" and matching data model, thus making provision for greater flexibility in large data set management [41].

B. Understanding the factors that influence software quality leverage

Leveraging in this study refers to taking maximum advantage of a phenomenon, by outlining influences and characteristics. Three factors, which include standardization, measurement metrics and implementation primarily influence the leveraging of software quality from the perspective of NoSQL:

Standardization of processes: Standardization is the extent to which recurrent processes and established standards are adhered to [42]. It is concerned with the use of common processes to satisfy heterogeneous needs, such as software within an environment. It includes accomplishment by means of associated rules derived from customer requirements for software. This can improve efficiency by leveraging economy of scale and simplifying the development and data modelling processes [19]. Furthermore, there are different levels of standardization, namely the use of common components in different solutions; the design of standardized solutions for comparable requirements; and the standardization of processes for similar solutions.

Measurement metrics: Measurement can be defined as the process of assigning a number or category to an object to describe an attribute of that object [43]. An attribute is a measurable property of an object, such as NoSQL. Software quality metrics are the measurement functions with software data as inputs and numerical value as outputs which can be interpreted as the extent to which software possesses a given attribute that affects the quality [19]. It is necessary to think critically before selection of metrics, as it involves "multiple criteria and an interdependent relationship between them" [19:2]. For example, programming of software is measured primarily to ensure that the functionalities are incorporated. Otherwise, it will be a challenge in assessing the benefits to the user and the organizations at large. Functionalities, such as the handling of synchronization of interactions and processes are critical to software implementation [21].

Implementation: Implementation has impact on how software is or can be leveraged in an organization. This is mainly because it is through implementation the objective of software can be assessed and achieved. This depends on whether the implementation is decentralized or centralized, as they have repercussions on whether or how a user makes use of the software. For instance, a decentralized approach can remove the performance bottleneck of centralization by employing the otherwise unused processing power of peers to drive solutions and can additionally make the system more tolerant of faults during adaptation. However, decentralization introduces issues of consistency when hosts have only a limited view of the global system state. Resolving these issues is central to our ongoing research [20]. This is mainly because it is virtually impossible to completely understand how the developer will distribute the software [21].

C. Examining the factors, which influence software integration in NoSQL database systems

Software integration is the process of connecting different computing systems and software applications to act in a synchronized manner [44]. The analysis reveals that there are four factors, which primarily influence software integration within the NoSQL database environment. This includes processes, data structure, heterogeneity and interoperability:

Processes: Processes are the series of actions taken, to achieve a specific goal. When integrating software with NoSQL databases, processes of data retrieval and manipulation factor into this integration practice [25]. A common approach that facilitates the process of data retrieval and manipulation is the use of a query language [24]. The integration of software is influenced by process efficiency [25]. As previously mentioned, NoSQL database systems have varied data models, each with different approaches for retrieval and manipulation of data. Query language can be used to facilitate easier data aggregation, to improve software engineering productivity [24]. The query language affects the performance and scalability of a software application [27]. In addition, if the process of data retrieval and manipulation within NoSQL database is not efficient it could have negative implications on software integration [22].

Data Structure: Data can be presented in many formats, such as JSON, XML, media files and PDF [22]. A selected number of NoSQL databases can store data in these varied formats. Subsequently, the software integration process with NoSQL databases can be influenced by the data structure needs of a given software application [22]. The implications of not taking the structure into account of data used in an application could render it inefficient and problematic [23]. Furthermore, data scope characterization and growth rate of data are factors that influence the integration of software with NoSQL databases [26].

Heterogeneity: The diversity that exists in semantics, systems and the NoSQL data models all factor into the software integration with NoSQL database systems [22]. This includes (1) Semantic Heterogeneity**,** which refers to datasets for the one application developed by independent parties which leads to dissimilar meanings and interpretations of data [22]. For example, one may represent gender as male and female. Another possible way of such a representation is to use letters "M" or "F". This influences the software integration with NoSQL databases, as software is developed by teams of programmers who all have different interpretations of the real

world. Furthermore, these perceptions can influence the way data is represented in a NoSQL database. Thus, semantic heterogeneity has an influence on software integration with NoSQL databases. (2) System Heterogeneity is the existence of diverse platforms like Windows, Linux, Unix that present a problem situation [22]. With software applications using NoSQL databases, the platform support a given database has will influence the choice in the integration of software. For instance, given a NoSQL database that only has platform support for one system, the software integrating with it would only be able to support that given platform, subsequently placing limitations on the software application [23]. (3) Data model Heterogeneity: NoSQL comprises of four different data models, which have their own query language and query syntax, which present advantages and disadvantages [22].

Interoperability: The notion of software integration is about interoperability, which is the ability of software, hardware and data management system to manage and communicate data effortlessly [23]. The essential idea of integrating a software application with database management systems is to enable them to communicate and then to interoperate to reach achieve a common objective as set out by the software specifications. In the context of NoSQL databases, interoperability plays a role in software integration to allow the specifications set out to be realized.

D. Software integration: Characteristics of NoSQL database systems

Four factors, which include data model, data distribution and replication support, Multi-version Concurrency control and security were found to be the main characteristics of NoSQL database systems in relation to software integration:

Data model: In distributed database computing much like NoSQL, the data model commands how application data can be organized and queried. This has a direct impact on the data and software architecture of an application [29]. In essence, the data model of a given NoSQL database will have an influence on the way in which data is modelled for a specific software application. Data model can be organized into three groups, namely: *Data Organization* – how a database platform allows data to be modelled; *Keys and Indexes* – the flexibility in data key definition can influence application performance, scalability and modifiability; and *Query Approaches* – which includes the options available for querying a database [27].

Data Distribution and Replication support: Data Distribution describes how a database coordinates access to data that is distributed over deployment configurations. The mechanism employed to locate data to return results to client affects performance, availability and scalability of data-intensive software applications [27. The presence of various software architecture presents a few alternatives that a NoSQL database can adopt to achieve data distribution, these can greatly affect the quality attributes of the software application [28]. In additional, NoSQL databases also support data replication, this facility is necessary to achieve high availability in data-intensive software applications [26].

Multi-version Concurrency control: In distributed database competing, the use of a central lock manager can prove to be a bottleneck, because all nodes in a database cluster would need to report to the central lock manager to carry out operations [29]. This is not an ideal situation because it degrades the performance of the database cluster and presents issue in fault tolerance. NoSQL database systems provide mechanisms to combat the bottleneck of central lock in distributed databases by means of optimistic replication with multi-version concurrency control (MVCC) [28]. In essence, this architectural trait means that when querying a database each transaction sees a snapshot of the data which is a database version, regardless of the current state of the underlying data [29].

Security: Security is necessary in the data tier layer of a software application because it is way of preserving data integrity and preventing unauthorized access [28]. There are different approaches to database security which are presented in Table 2 below, these approaches present key factors in determining how a data platform can be integrated into a security domain [27].

TABLE II: SECURITY [27]

Feature	Allowed Values
Client authentication	Custom user/password; X509; LDAP; Kerberos; HTTPS
Server authentication	Shared keyfile; server credentials
Credential store	In database; external file
Role-based security	Supported; not supported
Security role options	Multiple roles per user; role inheritance; default roles; custom roles; not supported
Scope of rules	Cluster; database; collection; object; field
Database encryption	Supported; not supported
Logging	Configurable event logging; configurable log flush conditions; default logging only

V FINDINGS AND DISCUSSION

From the analysis of the data as presented above, some factors were found to be of influence on the Leveraging of Software Quality Attributes in NoSQL Database Systems. This includes: (i) Requirements – as defined by both business and technical units; (i) Compatibility – with other software and hardware; (iii) Governance – which guides standards of performance, scalability and availability; (iv) Measurement metrics – assessing the quality through set criteria (v) Context of use – complexity of data model; data growth; operational environment; test environment; and user access patterns; and (vi) Architecture - data model; query language; API; data distribution and replication; security; concurrency control.

Based on the findings, a model (Figure 4) was developed. The primary aim of the model is to gain better understanding of the influencing factors, and how they interconnect with each other. Without an understanding of the interconnection, it is near impossible to assessing what constitutes software quality in a NoSQL environment. The discussion below should be read with Figure 4 to gain a better understanding.

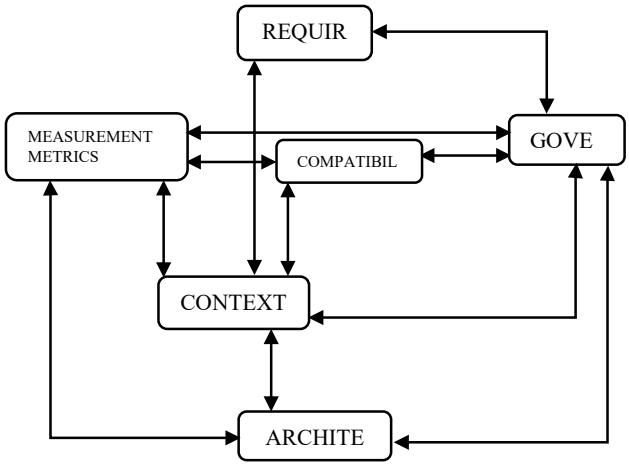

Figure 4: Software leverage

Requirements - The development of software is guided by both business and technical requirements. This is primarily to gain leverage from the software. Business requirements are derived from business processes that serve the purpose of reaching business objectives. Technical requirements are derived from the technical characteristics that a given software application is required to fulfil. These can include characteristics pertaining to performance, efficiency, dependability, availability etc. Requirements gathered are dependent on the context of use and governance and vice versa as depicted by the model above.

Context of use – this is composed of different aspects; it includes elements such as user access patterns; the operational environment in which an application will run; the testing environment; complexities involved in modelling of data needs for a specific application; and rate at which data will grow all of which have a direct influence on the requirements.

Governance – in the context of this study, governance refers to the standards that guide the performance, scalability and availability quality attributes that the NoSQL database systems exhibits within an operational environment. The standards formulated by organizations to guide quality attributes are often derived from requirements- be it technical or business. Also, the standards that guide performance, availability and scalability of NoSQL systems are directly impacted by the context of use.

Measurement metrics – these are measurement criteria, which are formulated based on the business and technical requirements, to gain leverage from the software. It is the extent upon which a software application possesses some property. This is the criteria upon which governance standards for performance, scalability and availability and context of use will be measured to reach organizational suitability. Most importantly, thus, the set of criteria is required to be compatible with organizational objectives.

Compatibility – to gain leverage, which manifest to return on investment, software must be compatible within an environment. To ensure compatibility, the software should be selected based on standards, principles as defined by governance. Also, the context within which the software will be used guides compatibility. Thus, compatibility of software contributes to suitability within an organization.

Architecture – the architecture is the underlying mechanisms of NoSQL systems, which is the primary carrier of software quality attribute. The architecture is based on the analysis artefact that ensures the design phase produces a suitable software application. It is evident that aspects that make up the architecture of a NoSQL database system will have repercussions on the software application integrating it. The NoSQL architecture is made of up two categories: Data Architecture and Software Architecture. Data Architecture is made up of the data model; query language; propriety API's; and consistency. The software architecture is made up of data distribution and replication support; security; and concurrency control. The architectural elements of NoSQL database systems can be measured using a set of criteria to understand their suitability for organizations. Additionally, the context in which a system is to be used leverages certain architectural traits to achieve suitability in each scenario. The governance standards that guide performance, availability and scalability are largely influenced by the architectural elements of a NoSQL database system and how they function in a harmonized manner to accomplish these quality attributes.

VI Conclusions

This research is important, and of interest to both academic and practitioners, as: it uncovers the complexities involved in leveraging NoSQL database systems to create awareness amongst software engineers; it determines the factors that influence the leveraging of software quality attributes in NoSQL database systems; and subsequently presents a software leveraging model that consolidates the complexity and factors in order to show how software quality attributes can be leveraged in NoSQL database systems to help software engineers and organizations gain maximum advantage of NoSQL technologies. Furthermore, the research contributes a novel model of software leveraging into the body of knowledge. As the research took an interpretivists philosophy, findings were subjectively analysed and to further analyse suitability future work could include implementation and suitability testing of the model in an organizational context, also an evaluation of NoSQL products with the software leveraging model may be another interesting avenue of research.

References

[1] Coronel, C. and Morris, S. (2016). *Database systems: design, implementation, & management*. Cengage Learning.

[2] Pokorny, J. (2013). NoSQL databases: a step to database scalability in web environment. *International Journal of Web Information Systems,* 9(1), 69-82.

[3] Moniruzzaman, A.B.M. & Hossain, S.A. (2013). Nosql database: new era of databases for big data analytics-classification, characteristics and comparison. *International Journal of Database Theory and Application,* 6(4), 1 -14.

[4] Grolinger, K., Higashino, W.A., Tiwari, A. & Capretz, M.A. (2013). Data management in cloud environments: NoSQL and NewSQL data stores. *Journal of Cloud Computing: advances, systems and applications,* 2(1), 22 - 46.

[5] Kaur, K. & Rani, R. 2013. Modeling and querying data in NoSQL databases. *Proceedings of the Big Data 2013 IEEE International Conference on, October 2013*. IEEE: 1-7.

[6] Lourenço, J.R., Cabral, B., Carreiro, P., Vieira, M. and Bernardino, J. 2015. Choosing the right NoSQL database for the job: a quality attribute evaluation. *Journal of Big Data*, *2*(1): 1 - 26.

[7] Jansen, A. & Bosch, J. 2005. Software architecture as a set of architectural design decisions. Proceedings of *5th Working IEEE/IFIP Conference on Software Architecture (WICSA'05)* IEEE: 109-120.

[8] Jiménez, M., Piattini, M. & Vizcaíno, A. 2009. Challenges and improvements in distributed software development: a systematic review. *Advances in Software Engineering*: 3- 17.

[9] Bonnet, L., Laurent, A., Sala, M., Laurent, B. & Sicard, N. 2011. Reduce, you say: What nosql can do for data aggregation and bi in large repositories. *Proceedings of 2011 22nd International Workshop on Database and Expert Systems Applications*, *August 2011*. IEEE: 483-488.

[10] Hammes, D., Medero, H. & Mitchell, H. 2014. Comparison of NoSQL and SQL Databases in the Cloud. *Proceedings of the Southern Association for Information Systems (SAIS), Macon, GA*, March 2014. 21-22.

[11] Hofmann, H.F. & Lehner, F., 2001. Requirements engineering as a success factor in software projects. *IEEE software*, 18(4): 58 - 66.

[12] Heer, J. & Agrawala, M. 2006. Software design patterns for information visualization. *IEEE transactions on visualization and computer graphics*, *12*(5): 853-860.

[13] Giorgini, P., Rizzi, S. & Garzetti, M., 2005. November. Goal-oriented requirement analysis for data warehouse design. *Proceedings of the 8th ACM international workshop on Data warehousing and OLAP, November 2005.* ACM: 47-56.

[14] Ponterotto, J.G. 2005. Qualitative research in counseling psychology: A primer on research paradigms and philosophy of science. *Journal of counseling psychology*, *52*(2): 126 - 136.

[15] Lin, G., 2009. Higher Education Research Methodology-Literature Method. *International Education Studies*, *2*(4): 179-181.

[16] Mack, L. 2010. The philosophical underpinnings of educational research. *Polyglossia*, *19*: 1-11.

[17] Leitch, C.M., Hill, F.M. & Harrison, R.T. 2009. The philosophy and practice of interpretivist research in entrepreneurship: Quality, validation, and trust. *Organizational Research Methods*.

[18] Gorton, I. & Klein, J. 2015. Distribution, data, deployment: Software architecture convergence in big data systems. *IEEE Software*, *32*(3): 78-85.

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

[20] Sykes, D., Heaven, W., Magee, J. & Kramer, J. 2010. Exploiting non-functional preferences in architectural adaptation for self-managed systems. *Proceedings of the 2010 ACM Symposium on Applied Computing, March 2010.* ACM: 431-438.

[21] Abdullah, D., Khan, M.H. & Srivastava, R. 2015. Flexibility: A Key Factor to Testability. International Journal of Software Engineering & Applications (IJSEA), 6(1): 89 -99.

[22] Garg, B. & Kaur, K. 2015. Integration of Heterogeneous Databases. Proceedings of *2015 International Advances in Computer Engineering and Applications (ICACEA),* 2015. IEEE: 1033-1038.

[23] Shen, W., Hao, Q., Mak, H., Neelamkavil, J., Xie, H., Dickinson, J., Thomas, R., Pardasani, A. & Xue, H., 2010. Systems integration and collaboration in architecture, engineering, construction, and facilities management: A review. *Advanced Engineering Informatics*, *24*(2): 196-207.

[24] Lawrence, R. 2014. Integration and Virtualization of Relational SQL and NoSQL Systems including MySQL and MongoDB. *Proceedings of the International Conference on Computational Science and Computational Intelligence*, 2014. IEEE: 285-290.

[25] Janaćković, G. 2013. Factors influencing efficient integration of safety systems. *Proceedings of the Common Conference on Human and Social Sciences,* November 2013. HASSACC*:* 13 - 16.

[26] Klein, J. Gorton, I. Ernst, N. Donohoe, P. Pham, K & Matser, C. 2014. *Quality Attribute-Guided Evaluation of NoSQL Databases: An Experience Report.* Carnegie-Melon University, Pittsburgh.

[27] Gorton, I., Klein, J. & Nurgaliev, A., 2015. Architecture Knowledge for Evaluating Scalable Databases. *Proceedings of the 2015 12th Working IEEE/IFIP Conference on Software Architecture (WICSA), May 2015.* IEEE: 95 – 104.

[28] Curé, O., Kerdjoudj, F., Faye, D., Le Duc, C. & Lamolle, M., 2013. On the potential integration of an ontology-based data access approach in NoSQL stores. *International Journal of Distributed Systems and Technologies (IJDST)*, *4*(3): 17-30.

[29] Mackin, H. Perez, G & Tappert, C. C. 2016. Integration and virtualization of relational SQL and NoSQL systems including MySQL and MongoDB. *Proceedings of Student-Faculty Research Day. CSIS, Pace University,* May 2016. 1-8.

[30] Leavitt, N. 2009. Is cloud computing really ready for prime time? *IEEE Computer Society*, *27*(5): 15-20.

[31] Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I. & Zaharia, M. 2010. A view of cloud computing. *Communications of the ACM*. *53*(4): 50-58.

[32] Thant, P.T. & Naing, T.T. 2011. Improving the availability of NoSQL databases for Cloud Storage. University of Computer Studies, Yangon.

[33] Hall, T., Beecham, S., Bowes, D., Gray, D. & Counsell, S. 2012. A systematic literature review on fault prediction performance in software engineering. *IEEE Transactions on Software Engineering*, *38*(6): 1276-1304

[34] Mauch, V., Kunze, M. & Hillenbrand, M. 2013. High performance cloud computing. *Future Generation Computer Systems*, *29*(6): 1408-1416

[35] Abramova, V., Bernardino, J. & Furtado, P. 2014. Experimental evaluation of NoSQL databases. International Journal of Database Management Systems, 6(3): 1-16.

[36] Rana, O.F. & Stout, K. 2000. What is scalability in multi-agent systems? *Proceedings of the fourth international conference on Autonomous agents*, June 2000, 56-63.

[37] Herbst, N.R., Kounev, S. & Reussner, R. 2013. Elasticity in cloud computing: What it is, and what it is not. *Proceedings of the 10th International Conference on Autonomic Computing (ICAC 13)*: 23-27.

[38] Chung, L., Nixon, B.A., Yu, E. & Mylopoulos, J. 2012. *Non-functional requirements in software engineering*. Springer Science & Business Media

[39] Amaya, N., Zervas, G. & Simeonidou, D. 2013. Introducing node architecture flexibility for elastic optical networks. *Journal of Optical Communications and Networking*, *5*(6): 593-608

[40] Han, J., Haihong, E., Le, G. & Du, J. 2011. Survey on NoSQL database. Proceedings of *Pervasive computing and applications (ICPCA), 2011 6th international conference, October 2011:* 363-366.

[41] Bakshi, K. 2012. Considerations for big data: Architecture and approach. *Proceedings from 2012 Aerospace Conference,* March 2012. IEEE: 1-7.

[42] Liu, J.Y.C., Chen, V.J., Chan, C.L. & Lie, T. 2008. The impact of software process standardization on software flexibility and project management performance: Control theory perspective. *Information and Software Technology*, 50(9): 889-896.

[43] Kaner, C., 2004. Software engineering metrics: What do they measure and how do we know. *Proceedings of the 10th International Software Metrics Symposium.*

[44] Abts, C., Boehm, B.W. and Clark, E.B., 2000. COCOTS: A COTS software integration lifecycle cost model-model overview and preliminary data collection findings. *Proceedings of the ESCOM-SCOPE Conference*: 18-20.