A Time-Series Database Analysis Based on a Multi-attribute Maturity Model

Ionut PETRE[1,2]*, Radu BONCEA[1,3], Constanta Zoie RADULESCU[1], Alin ZAMFIROIU[1,4], Ionut SANDU[1,2]

[1] National Institute for Research and Development in Informatics, 8-10 Maresal Averescu Avenue, Bucharest, 01145, Romania

[2] "Lucian Blaga" University of Sibiu, 10 Victoriei Blvd., Sibiu, 550024 Romania

[3] Politehnica University of Bucharest, 313 Independence Av., Bucharest, 060042, Romania

[4] Bucharest University of Economic Studies, 6 Romana Square, Bucharest 010374, Romania

ionut.petre@ici.ro (**Corresponding author*), radu@rotld.ro, zoie.radulescu@ici.ro, alin.zamfiroiu@ici.ro, ionut@rotld.ro

Abstract: Time Series Database (TSDB), is a particular type of data repository. TSDBs are capable of diverse functionalities regarding operations on time series data and are developed using different technologies. A large number of TSDB solutions, available both as open source and commercial software, has emerged in the last years. Selecting the proper TSDB is a challenging endeavour for a potential enterprise client. In this paper, a set of open-source TSDBs which includes InfluxDB, Graphite, RRDTool, Prometheus, OpenTSDB and TimescaleDB - has been selected for analysis, evaluation and ranking. A comparison of TSDBs implies the establishment of a set of attributes. A set of quantitative and qualitative attributes which have different scales and units of measure has been selected. The problem of selecting or ranking TSDBs evaluated in this paper with a set of attributes is a Multi-Attribute Decision Making problem (MADM). For solving the TSDBs evaluation, analysis and selection a multi-attribute TSDBs maturity model has been proposed. The model for the selected TSDBs set and for the 18 attributes, 10 quantitative and 8 qualitative, has been validated. The model for all the attributes, both quantitative and qualitative, has been solved and, finally, a comparison between the obtained rankings has been made. The comparison of the ranks obtained shows that the leading TSDB is InfluxDB.

Keywords: Time Series database, Quantitative and qualitative attributes, Maturity models, Multi-attribute method, Time Series database comparison.

1. Introduction

Recent years have witnessed an exponential increase in data, mainly due to consolidated markets for IoT and machine learning applications. As (DOMO, 2018) estimates, for each living person on the planet, by 2020, approximately 1.7 MB of data will be generated each minute. Most of these data are stored in data lakes or data warehouses, mainly depending on processing context. A large portion of these data comes as time series data, generated by IoT, monitoring or mobile applications and devices (Vevera & Onofrei-Riza, 2019) and used in machine learning through means of data mining to generate data intelligence (Last et al., 2004).

Time Series Databases (TSDBs), considered in this paper, are a particular type of data repositories, revolving around *time* – mainly TSDB is a database storing sequences of data values for each point in time. Time series are present in all areas of applied science and engineering involving temporal measurements. Therefore, as (Namiot, 2015) states, data persistence mechanisms for time series are among oldest tasks for databases.

The basis behind time series consists in repeated measurements of the parameters over time. Most times, the intervals are regular, but this is not a requirement. The use of the measurements is employed when the data is not updated but rather accumulated over time, with new data being added for each measured item at a time point. The main purpose of a time series database is to store or log sensor or other data over periods of time. A time series database must handle a large number of database entries and should provide means for filtering and analysing data.

The continuous developments in the information society have turned big data analytics into an appealing approach for all types of organizations. The combination of simplified models for development, commoditization, a wider palette of data management tools, and low-cost utility computing has effectively lowered the barrier to entry (Loshin, 2013).

In this paper (Boncea et.al., 2017) is presented a maturity model for big data technologies. This model considers several important characteristics divided into two classes: Technical and Business. In (Dhanuka, 2016) there are five areas to measure the maturity of Big Data solutions within organizations. These areas are: Sponsorship, Data and Analytics practices, Technology and Infrastructure, Organization and Skills and Process Management.

https://doi.org/10.24846/v28i2y201906

This paper is organized as follows.

In section 2, the TSDBs that will be analysed, evaluated and compared are described. The software maturity models for evaluating a specific software product or process to produce a corresponding maturity level are presented in section 3. A Multi-attribute maturity model for Time Series database is proposed in section 4. This TSDB model has four phases: maturity attributes definition, analysis and evaluation, general score calculation and ranking and selection. The quantitative and qualitative attributes evaluation is detailed in subsection 4.2. For TSDBs the general score calculation and ranking on a MADM method is presented in subsection 4.3. The MADM method is applied for all attributes, both quantitative and qualitative and the solutions obtained are discussed in subsection 4.4. Finally, the conclusions are presented in section 5.

2. Analysed TSDB solutions

The technologies for collecting and storing large scale time series are used by more and more companies and people. In recent years, the volume of time series data has greatly expanded and the tools for handling time series data, at this scale, are more demanded.

The selection of the solutions to be analysed is based on DB-Engines Ranking of Time Series DBMS (https://db-engines.com/en/ranking/time+series+dbms). The most popular TSDB, according to their ranking from February 2019, have been chosen.

Important Time Series solutions that are analysed in this paper are: *InfluxDB, Graphite, RRDTool, Prometheus, OpenTSDB and TimescaleDB*.

Apache Druid was not considered in this analysis due to the fact that the solution is still in incubation stage. *Kdb+* was not considered in this paper due to the lack of available information for measuring the quantitative indicators. Other time-series databases have a low market-share and were not considered in the present analysis.

2.1 InfluxDB

InfluxDB, released in 2013, is a high-performance data store, NoSQL, non-relational database, developed by InfluxData, dedicated for time series data, allowing high throughput ingest, compression and real-time querying of that same data. The data points can have one of the fields on a measurement, all of the fields on a measurement, or any number in-between, so the schemas are not mandatory to be defined before.

InfluxData provides a comprehensive platform to collect, store, analyse and visualize data, known as TICK stack. InfluxDB supports a wide range of client libraries and provides binaries that can run across several operating systems. The development language behind InfluxDB is Go. For writing in and querying from, the database has command line interface, HTTP API, client libraries and plugins for common data. In order to ensure the interaction with the data, InfluxDB offers InfluxQL, an SQL-like query language.

The lack of external dependencies makes InfluxDB very attractive from the practical point of view (Namiot, 2015).

Since the time-series databases can receive a large amount of data each second, InfluxDB automatically compacts the data to minimize the storage space. The data can be down sampled, and you can specify the time frame for which high precision raw data should be stored, then storing only the lower precision data. InfluxDB has two important features - Continuous Queries (CQ) and Retention Policies (RP) - for automating the processes of downsampling data and expiring old data. (https://www.influxdata.com/products/influxdb-overview).

2.2 Graphite

Released in 2006, Graphite consists of three components. The first is a twisted daemon called Carbon—which passively listens for time series data. Data is stored in a simple library called Whisper. Finally, graphs can be rendered on-demand via a simple Django web app (Berman, 2018).

Graphite-web – *considered in the stats in this paper*. This is a Django-based web application that renders graphs and dashboards.

Graphite-carbon - metric processing daemons.

Graphite-whisper - time-series database library. A fixed-size database designed to provide a reliable and fast storage of numeric data. It allows higher resolution of recent data to degrade into lower resolutions for long-term retention.

2.3 RRDTool – Round Robin Database Tool

There are currently 2 major versions. Only v1.x is considered in the current paper, as it is more widely used than v2.x.

Released in 1999, it is a high-performance data logging and graphing solution that can be integrated in shell scripts, Perl, Python, Ruby, Lua or TCL apps. The data analysis section of RRDtool can generate graphical representations of the data values collected in a time frame. (https://oss.oetiker.ch/rrdtool/doc/rrdtool.en.html).

RRDTool stores data in a compact manner, in a circular buffer that keeps the system storage footprint constant over time. Therefore, the database has the same amount of data points throughout its lifetime by removing old data sets when new data is inserted. When setting up a Round Robin Database, the interval when consolidation should occur can be specified, together with the consolidation function that should be applied to build the consolidated values - average, minimum, maximum, last.

2.4 Prometheus

Written in Go and released in 2015, it is designed for recording real-time metrics in a time series database (allowing for high dimensionality) built using a HTTP pull model, with flexible queries and real-time alerting. It was developed by SoundCloud and it implements a highly dimensional data model where time series are identified by a metric name and a set of key-value pairs. It has a query language for slicing the dimensional data for detailed exploration, graphing or alerting.

SoundCloud began the development of Prometheus driven by the need of a single solution to integrate a multi-dimensional data model, operational simplicity, scalable data collection, with a powerful query language.

The data is stored as metrics, each metric being referenced and queried by its name. Labels can be specified for the metrics and queries can also be made using those labels. Prometheus servers scrape metrics from instrumented jobs and store the scraped samples locally and apply rules over this data to either record new timeseries from existing data or generate alerts. (Volz & Rabenstein, 2015).

2.5 OpenTSDB

It was released in 2011. Distributed, scalable, designed to collect, store and serve billions of data points without affecting precision. Its architecture supports very high-performance data recording. It is built on top of Hadoop and HBase (http://opentsdb.net).

OpenTSDB is made up of a time series daemon (TSD) as well as a command line tool set. In order to use OpenTSDB, one or more TSDs must be run. There is no master or shared status to run as many TSDs as possible to handle the load. Each TSD uses the HBase data base or the Google Bigtable service to store and process data from time series.

Using an optimized data scheme, it is possible to quickly aggregate the time series to minimize storage space. TSD users do not need to directly access the base store, but can communicate with the TSD through a telnet protocol, an HTTP API, or a built-in GUI. All communications with the TSD are made on the same port.

To extract data from different sources and send them to TSDs, there are several tools available or write scripts that collect data and then can be periodically pushed to TSDs.

OpenTSDB offers a simple built-in interface that can generate graphs as an image by selecting one or more values and labels.

2.6 TimescaleDB

TimescaleDB is a relatively new solution, released in 2017, and was initially designed to make SQL scalable for time-series. It uses hypertables which are an abstraction or a virtual view of many individual tables holding the data, called chunks. A hypertable is defined by a standard schema with column names and types, with at least one column specifying a time value, and one optional column for specifying an additional partitioning key.

TimescaleDB is implemented as an extension on PostgreSQL, implying that the time series database is running within an overall PostgreSQL instance. TimescaleDB uses SQL, supporting all SQL operations and queries that are expected to work in PostgreSQL.

The extension model in TimescaleDB allows it to use the advantages of PostgreSQL such as reliability, security, connectivity to a wide

range of third-party tools. It contains standard database objects like tables, indexes, and triggers. TimescaleDB goes beyond PostgreSQL features when it comes to handling time-series data. These advantages are most easily seen when interacting with hypertables, which behave like normal tables yet maintain high performance even while scaling storage to normally prohibitive amounts of data. Hypertables can engage in normal table operations, including JOINs with standard tables (https://timescale.com).

3. Software Maturity Models

The concept of software process maturity introduced by Humphrey (Humphrey,1989) and implemented as the Capability Maturity Model for Software - CMM, (Paulk et al., 1993), is predicated on the premise that, while some differences exist, the software development process is similar to a manufacturing process, and, as a consequence, the concepts of statistical process control are directly applicable.

For evaluating a specific software product or process to produce a corresponding maturity level, in the scientific literature, there are many capability and maturity models (Rafa & Abran, 2011). Some of them are:

- Capability maturity model integration for software engineering - CMMi (SEI, 2002),
- Software Maintenance Maturity Model- S3M (April et al, 2004; April et al, 2005),
- Testing Maturity Model- TMM (Burnstein et al, 1996a; Burnstein et al, 1996b),
- Open Source Maturity Model-OSMM (Golden, 2004) and
- Software Product Maturity Model (Nastro, 1997).

Each of these maturity models consists of a number of sub models based on ISO 9126 quality characteristics and software products quality measures.

Besides these models intended to be applicable in any context, there is a trend towards the customization of such models so as to target target contexts more specifically. Such orientation is motivated by specific quality needs of certain contexts or specific standards. An example is open source software. There are more than 20 different OSS evaluation methods (Stol & Ali Babar, 2010), *(Adnan et al., 2017).*

4. A multi-attribute maturity model for Time Series databases

As previously mentioned, time series data are a special type of data. They are stored in a special type of databases: Time Series Databases. TSDBs are specialized in storing and querying time series data.

When a user selects a certain TSDB, it is best to be aware what TSDBs have to offer and how they can meet the user's requirements. The TSDBs are characterized by attributes. The problem of selecting a TSDB from a set of TSDBs with multiple attributes is a multi-attribute decision problem (Multi Attribute Decision Making-MADM). The aim of MADM is to find the most desirable TSDB or rank the feasible TSDBs for supporting decision makings.

For TSDBs evaluation, analysis and selection, a multi-attribute TSDBs maturity model is proposed.

The phases of the multi-attribute TSDBs maturity model proposed in this paper are the following:

1. *TSDBs maturity attributes definition*: used to define the attributes with respect to TSDBs families and attribute selection.
2. *TSDBs Analysis and Evaluation*: used to evaluate the TSDBs in relation to the attributes (and sub-attributes) selected.
3. *TSDBs general score calculation* for each TSDB based on a multi-attribute method (MADM method).
4. *TSDBs ranking and TSDB selection*.

4.1. Maturity Attributes Definition

In the present multi-attribute TSDBs maturity model a set of quantitative and qualitative attributes has been considered. The present analysis is not limited to a purely quantitative approach that could induce a limited vision on the solution, is less flexible and may omit a number of very important technical and business wise issues. The qualitative approach is difficult to quantify, but it is useful in the case of complex software solutions. In Figure 1 the attributes and sub-

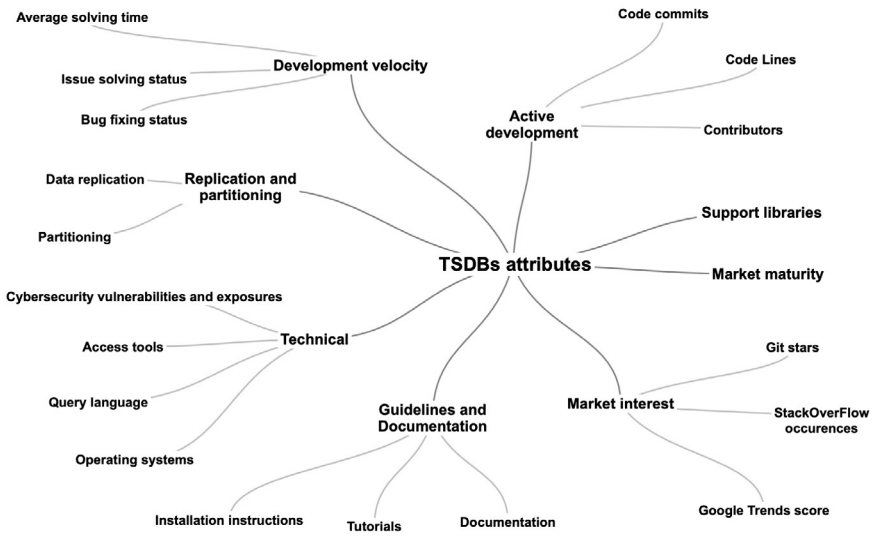

Figure 1. Attributes and sub attributes considered in the multi-attribute TSDB model

attributes considered in the multi-attribute TSDB model are displayed.

In Table1, the set of quantitative attributes is presented and in Table 2, the set of qualitative attributes for the problem of assessment analysis and selection of TSDB. Tables 1 and 2 include attributes, sub-attributes, a short presentation of each sub-attribute and the symbol corresponding to the attribute, symbols which are used in the multi-attribute TSDB model.

4.2 TSDBs Analysis and Evaluation

Choosing an appropriate TSDB for storing large amounts of time series data is a difficult task because different attributes should be considered in the analysis, evaluation and selection.

Table 1. TSDB quantitative attributes

Attribute	Sub-attribute	Symbol	Description
Active Development (AD)	Commits	TNC	The total number of code commits
	Code Lines	TCL	The total number of code lines
	Contributors	TC	The average number of contributors, as it appeared in GitHub and OpenHub.
Development velocity (DV)	Average solving time	AST	The average time for solving an issue
	Bug solving status	BSS	The ration between the number of open bugs vs closed bugs
	Issue solving status	ISS	The ration between the number of open issues vs closed issues
Market maturity (MM)	Market maturity	MM	The period of time in which the solution has been present on the market
Market interest (MI)	Git stars	GS	The number of stars received on GitHub
	StackOverFlow appearances	SOF	The number of tag occurrences in this professional forum
	Google Trends score	GTS	The average trend over the past year as available in Google Trends

Table 2. TSDB qualitative attributes

Attribute	Sub-attribute	Symbol	Description
Replication and partitioning (RP)	Data Replication	DR	Database replication is used in order to provide performance improvement, fault-tolerance and high availability
	Data Partitioning	DP	Data partitioning allows the division of data during the loading of Master Data
Support libraries	Support libraries	SL	Allow external solutions developed in various programming languages and paradigms to interact with the database by using native functions of the language
API and access	Access tools	AT	The access methods that are available for each analysed solution
	Query languages	QL	Support for standardized query languages
	Operation systems	OS	The operating systems on which the solution can be deployed
	CVEs	CVE	Cybersecurity vulnerabilities and exposures
Guidelines and Documentation (GD)	Guidelines and Documentation (GD)	GD	GD refers to documentation on installation, management and usage of available features, such as system administration or development guidelines

4.2.1 Quantitative Attributes

In this subsection the quantitative attributes considered for the present model are analysed and evaluated. For this we analysed several technical resources available online – GitHub (https://github.com), OpenHub (https://www.openhub.net/), StackOverFlow (https://stackoverflow.com) and Google Trends (https://trends.google.com).

Active development (AD)

AD refers to the state of development of a certain TSDB. The sub-attributes considered for the AD attribute are:

- *TNC - Total number of code commits*
- *TCL–Total number of code lines*
- *TC–Total number of contributors:* the average number of contributors, as it appears in GitHub and OpenHub.

For the considered TSDBs the sub-attributes evaluation is presented in Table 3.

Table 3. Active Development evaluation

TSDB	Sub-Attribute		
	TNC	*TCL*	*TC*
InfluxDB	30310	328114	420
Graphite	3963	7827	354.5
Prometheus	5748	638660	32
RRDTool	3034	100066	102
OpenTSDB	1639	115116	136.5
TimescaleDB	1283	57571	40.5

Development velocity (DV)

DV refers at the promptness for solving and fixing issues. The sub-attributes considered for attribute DV are:

- *AST - Average solving time*: the average time for solving an issue. It has been calculated as the average solving time for the latest 20 known reported issues on GitHub.
- *BSS – Bug solving status*: the ratio between the number of open bugs vs fixed bugs calculated by: $BSS_i = \frac{TB_i - FB_i}{FB_i}$

where:

- *TB* - total number of bugs;
- *FB* - number of fixed bugs;
- *ISS – Issue solving status*: the ration between the number of open issues vs fixed issues: $ISS_i = \frac{OI_i}{FI_i}$

where:

- *OI* – number of open issues;
- *FI* – number of fixed issues;

For the considered TSDBs the values have been calculated, as shown in Table 4.

Table 4. Development Velocity evaluation

TSDB	Sub-Attribute		
	AST	*BSS*	*ISS*
InfluxDB	0.2	0.472	0.256
Graphite	18.5	0.514	0.481
Prometheus	2.6	0.059	0.113
RRDTool	6.1	1.000	0.256
OpenTSDB	28.1	0.733	0.876
TimescaleDB	7.14	0.080	0.265

Market maturity (MM)

The attribute MM considers the period of time in which the solution has been present on the market.

For the TSDBs considered the evaluated solutions are presented in the Table 4.

Market interest (MI)

MI shows the level of interest in the market and consists of the following sub-attributes:

- *GS – Git stars*: the number of stars received on GitHub.
- *SOF – StackOverFlow occurrences* when doing tag searching on the largest question and answer site for IT professionals and programmers.
- *GTS – Google Trends score*: the average interest over time for the past year as available in Google Trends. In Google Trends each data point is divided by the total searches based on geographic location and time range. The resulting numbers are then scaled on a range of 0 to 100, where 100 is the maximum search interest for the time and location selected (Rogers, 2016).

For the considered TSDB solutions the sub-attributes evaluation is presented in Table 5.

Table 5. Market Maturity and Market Interest evaluation

TSDB	Sub-Attribute			
	Age	GS	SOF	GTS
InfluxDB				
Graphite	11	4434	935	16.44
Prometheus	4	21628	1381	40.18
RRDTool	20	518	541	20.96
OpenTSDB	8	3463	226	8
TimescaleDB	2	7112	49	26.05

4.2.2 Qualitative Attributes

These attributes refer to aspects that are difficult to quantify and often times require expert review and feedback for appropriate evaluation.

Replication and partitioning (RP)

Table 6 presents the features available for data partitioning and replication.

Table 6. Data partitioning and replication

TSDB	Sub-attribute	
	Data Partitioning	Data Replication
InfluxDB	Sharding	Selectable replication factor
Graphite	None	None
Prometheus	Sharding	Yes
RRDtool	None	None
OpenTSDB	Sharding	Selectable replication factor
TimescaleDB	Yes, across time and space (hash partitioning) attributes	Master-slave replication with hot standby and reads on slaves

Data partitioning (DP) is a method that allows the division of data during the loading of master data. The tables are split into smaller chunks by rules that are set by the user. This is particularly convenient when large data sets are stored, increasing overall system speed and maintainability.

Database replication (DR) is used as means to improve performance, fault-tolerance and high availability, as each replica can provide access without requiring any coordination with the rest of replicas (Muñoz-Escoí et al., 2007).

Sharding is a database architecture pattern related to horizontal partitioning — the practice of separating the table rows into multiple different tables, known as partitions. Each of them maintains the same schema and columns, but entirely different rows so the data held in each partition is unique and independent of the data held in other partitions (Drake, 2019).

Support libraries (SL)

SL refers to the 3rd party libraries that allow external solutions developed in various programming languages and paradigms to seemingly integrate the TSDB capabilities. Table 7 presents the support libraries for the selected TSDBs.

Table 7. Support libraries

TSDB	Support libraries
InfluxDB	.Net; Clojure; Erlang; Go; Haskell; Java; JavaScript; JavaScript (Node.js); Lisp; Perl; PHP; Python; R; Ruby; Rust; Scala
Graphite	JavaScript (Node.js); Python
Prometheus	.Net; C++; Go; Haskell; Java; JavaScript (Node.js); Python; Ruby
RRDtool	C; C#; Java; JavaScript (Node.js); Lua; Perl; PHP; Python; Ruby
OpenTSDB	Erlang; Go; Java; Python; R; Ruby
TimescaleDB	.Net; C; C++; Delphi; Java; JavaScript; Perl; PHP; Python; R; Ruby; Scheme; Tcl

API and access tools (AA)

In Table 8 the access tools or the interfaces that are available for each analysed solution and the operating systems on which they can be deployed are presented. The number of existing reports has been included in Common Vulnerabilities

Table 8. API and access methods

TSDB	Sub-attributes			
	Access tools	SQL	OS	CVEs
InfluxDB	HTTP API, JSON over UDP	SQL-like query language	Linux, OS X	1
Graphite	HTTP API, Sockets	No	Linux, Unix	1
Prometheus	RESTful HTTP/JSON API	No	Linux, Win	4
RRDtool	in-process shared library, Pipes	No	HP-UX, Linux	2
OpenTSDB	HTTP API, Telnet API	No	Linux, Win	3
TimescaleDB	native C library, streaming API for large objects, ADO.NET, JDBC, ODBC	Yes	Linux, OS X, Win	0

and Exposures – CVE - a dictionary of publicly disclosed cybersecurity vulnerabilities and exposures (https://cve.mitre.org).

Guidelines and Documentation (GD)

GD refers to documentation on installation, management and usage of available features. All the analysed solutions provide documentation, with installation instructions for different operating systems, concepts and insights for administration and configuration features. The most modern documentation is provided for TimescaleDB, with tutorials and installation instructions on servers or Cloud, development guidelines, migration instructions from PostgreSQL to TimescaleDB etc. The study has also considered whether the TSDB can provide documentation for the developers which might contribute to the TSDB development.

InfluxDB offers a tutorial for InfluxDB Line Protocol. Prometheus offers several basic tutorials. RRDTool provides tutorials, but the documentation needs improvement in terms of structure and user interface. Graphite and OpenTSDB do not provide complete tutorials.

The analysed technologies can be used in Cloud. Cloud Computing is a new step in the Internet development and in the industrialization of computing power; its resources can be accessed through Internet-based communication systems (Dumitrache, 2014) and (Banciu, 2016). Many companies that build IoT applications want to use the cloud capabilities.

4.2.3. Qualitative Attributes Evaluation

For the qualitative TSDB attributes evaluation a questionnaire is defined. The questionnaire includes the evaluation of the selected qualitative TSDB attributes. The expert's evaluation is based on a measure scale defined for the attributes. The experts give a score to each TSDB in relation to each attribute by using this measure scale.

For qualitative attributes the measure scale for the TSDB evaluation is considered across five levels. The scores 1, 2, 3, 4, and 5 represent 'no importance' 'low importance' 'medium importance' 'high importance' and 'very high importance' respectively.

In Table 9 the evaluations of experts for TSDBs and qualitative attributes are presented.

Table 9. The expert's evaluations for qualitative attributes

Qualitative attributes	Experts TSDB evaluations					
	InfluxDB	Graphite	Prometheus	RRDtool	OpenTSDB	TimescaleDB
Data Partitioning	4	1	4	1	4	4
Data Replication	4	1	3	1	5	3
Support libraries	5	3	5	4	5	4
Access means	5	4	5	2	4	3
Query Language	5	1	4	1	1	5
Operation Systems	4	4	5	3	5	5
CVE	4	3	3	3	2	5
Guidelines and Documentation	5	3	5	2	4	5

4.3. TSDBs General Score Calculation

The problem considered below is represented by the TSDBs general score calculation based on a multi-attribute method.

A set of TSDBs has been selected:

A = {InfluxDB, Graphite, Prometheus, RRDtool, OpenTSDB, TimescaleDB} (number of TSDB is *m=6*).

A set of quantitative and qualitative attributes and sub-attributes for the TSDB analysis, evaluation and selection has already been defined in Tables 1 and 2.

Let *C = {TNC, TCL, TC, AST, BSS, ISS, MM, GS, SOF, GTS, DR, DP, SL, AT, SQL, OS, CSV, GD}* be the set of TSDBs sub-attributes (number of TSDB attributes is *n=18)*.

The group of experts gives each attribute a weight (importance of TSDB attribute). Let $\mathbf{w} = (w_j)$ be the *n*-dimensional vector of attribute weights.

$$\sum_{j=1}^{n} w_j = 1.$$

Table 10. The TSDBs attribute and sub-attribute weights

	Attribute and weight	Sub-attribute	Weights	Total weights
Quantitative (0.6)	Active Development (AD) (0.2)	Commits	0.133	0.08
		Code Lines	0.067	0.04
		Contributors	0.133	0.08
	Development velocity (DV) (0.2)	Average solving time	0.067	0.04
		Bug solving status	0.117	0.07
		Issue solving status	0.067	0.04
	Market maturity (MM) (0.1)	Market maturity	0.083	0.05
	Market interest (MI) (0.1)	Git stars	0.133	0.08
		StackOverFlow app.	0.100	0.06
		Google Trends score	0.100	0.06
Qualitative (0.4)	Replication and partitioning (RP) (0,15)	Data Replication	0.1875	0.075
		Data Partitioning	0.1875	0.075
	Support libraries (0.15)	Support libraries	0.2	0.080
	API and access (0.05)	Access tools	0.15	0.060
		Query Language	0.05	0.020
		Operation systems	0.025	0.010
		CVEs	0.025	0.010
	Guidelines and Documentation (GD) (0.05)	Guidelines and Documentation (GD)	0.175	0.07

A group of experts with competence in the TSDB field has evaluated the selected qualitative attributes and has determined the importance of each attribute in relation to the others. Opinions from experts on qualitative attributes will be aggregated using a weighting method. A weighting method can be chosen from a set of weighting methods. Examples of weighting methods are:

- Step-Wise Weight Assessment Ratio Analysis (SWARA),
- Extended Stepwise Weight Assessment Ratio Analysis (SWARA),
- Decision-Making Trial and Evaluation Laboratory (DEMATEL),
- Simple Multi-Attribute Rating Technique (SMART),
- Analytical Hierarchy Process (AHP),
- Entropy method (Shannon's entropy concept, Shannon method),
- Mean square deviation method.

A synthesis of the objective and subjective criteria weighting methods is presented in (Radulescu & Radulescu, 2018) and (Zavadskas et al., 2018).

The attribute and sub-attribute weights are presented in Table 10.

Given the sets A (of TSDBs) and C (of the evaluation of TSDBs attributes) we build a m x n matrix $E = (e_{ij})$ called evaluation matrix has been developed.

The entry e_{ij}, $i = 1,2,...,m$, $j = 1,2,...,n$ represents the evaluation of the i-th TSDB by means of the j-th attribute. The entries of each column have the same measurement unit or scale. It is supposed that the matrix $E = (e_{ij})$ has positive entries ($e_{ij} > 0$).

The normalized matrix:
$\overline{E} = (\overline{e}_{ij})$ $i = 1,2,...,m,$ $j = 1,2,...,n$ is calculated as $\overline{e}_{ij} = e_j / \sqrt{\sum_{k=1}^{n} e_k^2}$.

Then, the weights normalized matrix $\overline{\overline{E}} = (\overline{\overline{e}}_{ij})$, $i = 1,2,...,m,$ $j = 1,2,...,n$ is calculated as:
$\overline{\overline{e}}_{ij} = \overline{e}_{ij} \times w_j$

4.4. TSDBs ranking and selection

The positive ideal and negative ideal solutions A^+ and A^- have been calculated as:

$A^+ = (a_j^+)$, $a_j^+ = \max_i \overline{\overline{e}}_j$;

$A^- = (a_j^-)$, $a_j^+ = \min_i \overline{\overline{e}}_j$ $j = 1,2,...,n.$

For each TSDB $i = 1,2,...,m$ the Euclidean distances from the positive and negative ideal solution are calculated:

$$d_i^+ = \left[\sum_{j=1}^{n} \left| \overline{\overline{e}}_j - a_j^+ \right|^2 \right]^{1/2}$$

$$d_i^- = \left[\sum_{j=1}^{n} \left| \overline{\overline{e}}_j - a_j^- \right|^2 \right]^{1/2}, \quad i = 1,2,...,m$$

Then it is computed the relative closeness to the ideal solution $s^T = (s_i^T)$,

$s_i^T = d_i^- / (d_i^+ + d_i^-), \quad i = 1,2,...,m.$ s^T is the solution. The solution and TSDG ranks are presented in Table 11. The leading TSDB in this rank is InfluxDB.

Table 11. TSDB Solution and ranks for all sub-attributes

	Positive ideal solution	Negative ideal solution	Solution	TSDB Ranks
InfluxDB	0.06040	0.12323	0.671074	1
Graphite	0.11291	0.06213	0.354954	4
Prometheus	0.09623	0.10022	0.510156	2
RRDtool	0.12701	0.06238	0.329375	5
OpenTSDB	0.11406	0.07787	0.405716	3
TimescaleDB	0.12987	0.04923	0.274889	6

Table 12. TSDB Solution and ranks for quantitative attributes

	Positive ideal solution	Negative ideal solution	Solution	TSDB Ranks
InfluxDB	0.09935	0.18637	0.652264	1
Graphite	0.16635	0.10083	0.377386	3
Prometheus	0.15705	0.14781	0.484839	2
RRDtool	0.1898	0.10321	0.352250	4
OpenTSDB	0.18871	0.09512	0.335134	5
TimescaleDB	0.21273	0.04290	0.167816	6

Table 13. TSDB Solution and ranks for qualitative attributes

	Positive ideal solution	Negative ideal solution	Solution	TSDB Ranks
InfluxDB	0.02429	0.12946	0.841979	1
Graphite	0.13202	0.03543	0.211627	5
Prometheus	0.04874	0.11669	0.705345	3
RRDtool	0.14063	0.0188	0.117927	6
OpenTSDB	0.03448	0.13244	0.793419	2
TimescaleDB	0.05997	0.10493	0.636292	4

If we consider the problem of ranking TSDBs for quantitative and qualitative attributes separately we obtain the following solutions (Table 12 and 13). A comparison of the ranks obtained for the three problems shows that the first two have 3 equal positions whereas the first and the third have only one position.

5. Conclusions

By using the time series information, the sequence of the events can be determined and also the correlations between different types of events can be established. Furthermore, the predictions based on the determined behaviour of an app. can be established as well. Patterns can be determined and recognized after making a timely behaviour analysis.

The contributions of the paper are:

- An analysis and evaluation of a set of TSDBs in rapport with quantitative and qualitative attributes.
- A TSDBs ranking and selection based on a proposed multi-attribute maturity model.
- The analysis provides important information for a TSDB potential user to be able to better understand the most important quantity and quality attributes that can influence the decision of TSDB selection to use.

Acknowledgements

Authors wish to express their gratitude to Dr. Niculescu Andrei, the Managing Editor of SIC for his helpful insights throughout the publication process.

This work was supported by project PN 19 37 04 01*"New solutions for complex problems in current ICT research fields based on modelling and optimization"*, funded by the Romanian Core Program of the Ministry of Research and Innovation (MCI), 2019-2022.

REFERENCES

1. Adnan, Z., Khalid, K. & Sarfaraz, N. (2017). Comparison of Open Source Maturity Models, *Procedia Computer Science*, *111*(C), 348-354.

2. April, A., Abran, A. & Dumke, R. R. (2004). Assessment of software maintenance capability: A model and its architecture. In *Proceedings of the IASTED Conference on Software Engineering (CSMR'04)*, Innsbruck, Austria (pp. 309-314).

3. April, A., Hayes, J. H., Abran, A. & Dumke, R. R. (2005). Software maintenance maturity model (SMmm): The software maintenance process model, *Journal of Software Maintenance and Evolution: Research and Practice*, *17*(3), 197-223.

4. Banciu, D & Dumitrache, M. (2016). Managing a cloud-based documents system. In *Proceedings of the 2nd International Scientific Conference SAMRO*, Editura Tehnică (pp. 13-19).

5. Berman, D. (2018). *Prometheus vs. Graphite: Which Should You Choose for Time Series or Monitoring?*. Online, available at: <https://logz.io/blog/prometheus-vs-graphite/>, last accessed: June 10, 2019.

6. Boncea R., Petre, I., Smada, D. M. & Zamfiroiu, A. (2017). A Maturity Analysis of Big Data Technologies, *Informatica Economică, 21*(1), 60-71.

7. Burnstein, I., Suwanassart, T. & Carlson, C. R. (1996a). Developing a testing maturity model: Part I, *CrossTalk: The Journal of Defense Software Engineering, 9*(8), 21-24.

8. Burnstein, I., Suwanassart, T. & Carlson, C. R. (1996b). Developing a testing maturity model: Part II, *CrossTalk: The Journal of Defense Software Engineering, 9*(9), 19-26.

9. Dhanuka, V. (2016). *Big Data maturity model, Hortonworks*. Online, available at: <http://hortonworks.com/wp-content/uploads/2016/04/Hortonworks-Big-Data-Maturity-Assessment.pdf>, last accessed: June 10, 2019.

10. Drake, M. (2019). *Understanding database sharding*. Online, available at: <https://www.digitalocean.com/community/tutorials/understanding-database-sharding>, last accessed: May 28, 2019.

11. Dumitrache, M. (2014). Cloud Computing – A new phase in Internet development, *Romanian Journal of Information Technology and Automatic Control*, *24*(4), 40-48.

12. Golden, B. (2004). *Succeeding with Open Source*. Boston, MA, USA, Addison-Wesley Professional.

13. Humphrey, W. S. (1989). *Managing the Software Process*. Addision-Wesley Publishing Company.

14. Last, M., Kandel, A. & Bunke, H. (2004). *Data Mining In Time Series Database*, 1-190. World Scientific Publishing Co. Pte. Ltd.

15. Loshin, D. (2013). *Big Data Analytics: From Strategic Planning to Enterprise Integration woth Tools, Techniques, NoSQL and Graph*. Amsterdam, Elsevier.

16. Muñoz-Escoí, F. D., Decker, D, Armendáriz-Íñigo, J. E. & González de Mendívil, J. R. (2007). *Database Replication Approaches, Technical Report ITI-ITE - 07/19*. Instituto Tecnológico de Informática, Valencia (Spain).

17. Namiot, D. (2015). Time series databases. In *XVII International Conference «Data Analytics and Management in Data Intensive Domains» (DAMDID/RCDL'2015),* Obninsk, Russia (pp. 132-137).

18. Nastro, J. (1997). A software product maturity model, *CrossTalk: The Journal of Defense Software Engineering, 10*(8).

19. Paulk, M., Curtis, B., Chrissis, N. B. & Weber, C. (1993). Capability Maturity Model for Software, Version 1.1, *IEEE Software*, *10*(4), 18-27.

20. Radulescu, C. Z. & Radulescu, M., (2018). Group Decision Support Approach for Cloud Quality of Service Criteria Weighting, *Studies in Informatics and Control*, *27*(3), 275-284. DOI: doi.org/10.24846/v27i3y201803

21. Rafa E. Al-Qutaish & Abran, A. (2011). A Maturity Model of Software Product Quality, *Journal of Research and Practice in Information Technology*, *43*(4), 307-327.

22. Rogers, S. (2016). *What is Google Trends data—and what does it mean?*. Online, available at: <https://medium.com/google-

news-lab/what-is-google-trends-data-and-what-does-it-mean-b48f07342ee8>, last accessed: May 28, 2019.

23. Stol, K. J. & Ali Babar, M. (2010). A Comparison Framework for Open Source Software Evaluation Methods. In *Proceedings of the OSS 2010, IFIP AICT - International Federation for Information Processing, vol. 319* (pp. 389-394).

24. Vevera, A. V. & Onofrei-Riza, D. B. (2019). *Investigații mobile – captură, analiză* și *stocare a datelor senzitive*, *Romanian Journal of Information Technology and Automatic Control*, *29*(1), 45-50.

25. Volz, J. & Rabenstein, B. (2015). *Prometheus: Monitoring at SoundCloud.* Online, available at: <https://developers.soundcloud.com/blog/prometheus-monitoring-at-soundcloud>, last accessed: June 10, 2019.

26. Zavadskas, E. K, Stević, Ž., Tanackov, I. & Prentkovskis, O. (2018). A Novel Multicriteria Approach – Rough Step-Wise Weight Assessment Ratio Analysis Method (R-SWARA) and Its Application in Logistics, *Studies in Informatics and Control*, *27*(1), 97-106. DOI: doi.org/10.24846/v27i1y201810

27. *** *Common Vulnerabilities and Exposures*. Online, available at: <https://cve.mitre.org>, last accessed: May 28, 2019.

28. *** *DB-Engines Ranking of Time Series DBMS*. Online, available at: <https://db-engines.com/en/ranking/time+series+dbms>, last accessed: May 28, 2019.

29. *** *DOMO – Data Never Sleeps*. Online, available at: <https://www.domo.com/assets/downloads/18_domo_data-never-sleeps-6+verticals.pdf>, last accessed: May 28, 2019.

30. *** *GitHub*. Online, available at: <https://github.com>, last accessed: May 28, 2019.

31. *** *Google Trends*. Online, available at: <https://trends.google.com>, last accessed: May 28, 2019.

32. *** *InfluxDB*. Online, available at: <https://www.influxdata.com/products/influxdb-overview/>, last accessed: May 28, 2019.

33. *** *OpenHub*. Online, available at: <https://openhub.net>, last accessed: May 28, 2019.

34. *** *OpenTSDB*. Online, available at: <http://opentsdb.net/>, last accessed: May 28, 2019.

35. *** *RRDTool*. Online, available at: <https://oss.oetiker.ch/rrdtool/doc/rrdtool.en.html>, last accessed: May 28, 2019.

36. *** *SEI (2002): CMU/SEI-2002-TR-029: Capability maturity model integration for software engineering (CMMI-SW) – Staged Representation, Version 1.1*. Pittsburgh, PA, USA, Software Engineering Institute, Carnegie Mellon University.

37. *** *StackOverFlow*. Online, available at: <https://stackoverflow.com/>, last accessed May 28, 2019.

38. *** *Timescale*. Online, available at: <https://www.timescale.com/>, last accessed: May 28, 2019.