Multi-Database Monitoring Tool for the E-Health Services

Igor Kotsiuba
Computer Science and Engineering department, PIMEE NAS of Ukraine
Kyiv, Ukraine
i.kotsiuba@gmail.com

Inna Skarga-Bandurova
Computer Science and Engineering department, Volodymyr Dahl East Ukrainian National University
Severodonetsk, Ukraine
skarga-bandurova@snu.edu.ua

Maksym Nesterov
Computer Science and Engineering department, Volodymyr Dahl East Ukrainian National University
Severodonetsk, Ukraine
m.nesterov@snu.edu.ua

Tetiana Biloborodova
Computer Science and Engineering department, Volodymyr Dahl East Ukrainian National University
Severodonetsk, Ukraine
beloborodova.t@gmail.com

Yury Yanovich
Bitfury
Amsterdam, Netherlands
yury.yanovich@bitfury.com

Viacheslav Zhygulin
Bitfury
Amsterdam, Netherlands
viacheslav.zhygulin@bitfury.com

Abstract—The paper outlines some aspects of developing an information system for e-Health database administrators (DBA). This system is equipped with advanced database performance monitoring and prediction features and based on different captured metrics as CPUs, RAMs, HDDs, and other values as well as historical events from host cluster/nodes. The data are aggregated, processed and transformed to provide the developers, production and host DBA teams with timely, proper, and valuable information about existing issues. The graphical representation of critical parameters to the working instance is provided. The approaches to forecasting database performance troubles related to the degradation of its characteristics due to different cumulative effects are discussed.

Keywords—database, administration, performance, monitoring, dashboard.

I. INTRODUCTION

The digital transformation of health care has made medical information and services globally accessible. Healthcare networks include not only clinics and, doctor's offices, but also multiuser medical environments and digital tools such as mobile health applications or personal devices. Accurate and efficient database monitoring of e-Health services is crucial to uninterrupted business operations, system intelligence and delivering life-critical services.

The complexity of modern e-Health services has led to more complicated database operation tests and complex collection of operational data using traditional embedded tools and strategies. Large, heterogeneous environments can have serious problems regarding their management. Additional challenges arise when considering multiple database platforms or/and multi-database environments. Traditionally, e-Health database platforms are comprised of a variety of database types, the most popular of them being Microsoft SQLServer, MySQL, Oracle, and DB2. These platforms currently support thousands of databases. Therefore, the general goal is to design and develop a multi-database monitoring system for consolidating information into a single comprehensive view and ensuring consistently high database performance and availability.

The paper is organized as follows: Section 2 is comprised of the related works and motivation. Section 3 describes the proposed system and its key capabilities as well as an information panel for performance monitoring and visualization. The short discussion of mathematical models for DB performance prediction is given in Section 4 followed by the conclusions in Section 5.

II. MOTIVATION

There are different monitoring tools, e.g., SQLSentry [1], System Center Operations Manager [2], Idera [3], SolarWinds [4], Quest Software [5], etc. At the same time, few of them work with multiple database platforms or/and multi-database environment and can supply cross-platform database administration. Thus, SolarWinds has an Oracle-specific and SQL Server-specific monitoring tool [4]. A cross-platform database infrastructure monitoring dashboard for the Hanover was represented in [6]. All the same, E. Sarmiento [7] is noted that for both database platforms Oracle Enterprise Manager (OEM) with the SQL Server plug-in is a pain to manage.

Moreover, as it mentioned in [8], multiple separate tools cannot give the global view on performance problems and next steps in troubleshooting.

At the moment, most Ukrainian healthcare information systems utilize the Oracle database monitoring tools in a limited mode. For health care purposes, data must meet requirements of high availability and safety. In this context, Oracle proposes a Real Application Cluster (RAC) which enables a quick view of one database containing multiple instances. In its turn, OEM allows the management of single instances of an Oracle Database and is available only as a licensed version, which reasonably excludes usage of complementary functions of diagnostics and tuning. Automatic workload repository (AWR) automatically writes all values to AWR Data snapshots. When OEM starts, it massively begins to read data values from AWR snapshots; thus it loads a central processing unit (CPU), HDD, RAM. Besides that, OEM contains different advisers for tuning DB to get maximum performance but they are not always doing it properly, for example, SQL executive plan can be changed after every call of SQL query, it depends on the amount of data, table structure (primary keys, foreign keys, and others), but these changes do not always give good performance.

There are two powerful tools available to administrators: First, OEM and their deep knowledge of a work inginstance. However, when dealing with RAC and more than one instance, it's hard to get all instances working without proper and accurate information about DB, especially if there is the set of configuration parameters. In addition, there are different customized tools for database monitoring and scanning. One of the most significant challenges with all these tools is understanding of the massive amounts of the collected data. In many cases, unless the business users personally send the information to the DBA, problems with the application can go unnoticed up to the moment the system goes to restart after the critical state. Often this leads to a situation where the DBA draws in the database when there has already been a denial of service, reloaded or shut-down. Investigation begins with obtaining an inquiry (ticket) or SQL report from users. In general case, available OEM database monitoring tools withdraw current state and permit to look over historical data without comprehensive predictive analysis. In case of large, heterogeneous environments (as medical databases actually are) these tools are not highly-efficient so long as DBA can obtain notices (alerts) only after outputting tickets. Moreover, it is necessary to launch OEM on every launched instance to see statistics. Practically any embedded monitoring service can extract indicators and notify about issues with data, but the main challenge here is to provide information at the proper time and with the context explaining what indices lead to problems with a database. Users need a solutions allowing them to understand the real causes of bottle-necks in performance to find the source of the specific problem and solve it taking into account all system parameters (not only indexes of the database). These problems require new comprehensive methodologies enabling users them to gain a complete understanding of what is going on with the system and estimate the achievable performance with a given set of resources.

III. DESCRIPTION OF SYSTEM

A. System overview

Taking into account the complexity of the E-Health system, we propose an extended methodology for troubleshooting and performance that involve additional procedures for monitoring historical events, special prediction and optimization tools and techniques on the side of the database administrator [9]. A general scheme of our approach for database performance monitoring is shown in Fig. 1. The system is designed to gather reports of potential dangers and identify possible shutdown (denial of service, performance problems) and display information for users to support decisions in case of problems with the database, when the database performance is close to their limits or the database has already been a denial of service, reloaded or shut-down.

The main functions of the system are:

- collecting operational data such as CPU, memory and disk space utilization,
- capturing the trend on the size of the database files;
- monitoring disk space utilization and do proper capacity planning;
- database performance monitoring;
- forecasting database disk capacity based on historical events day/month/hours.

Proposed infrastructure for database performance monitoring consists of (1) a Database Performance Monitoring system including performance index database and (2) a Web/Mobile application composed of dashboard and notification tools for informing about critical changes in indexes and/or data values.

For gathering backend statistics, we use PL/SQL scripts and bash scripts oh host OS. A graphical interface is designed with DHTMLX and HTML5-based JavaScript interactive graphs; data are uploaded using AJAX technology. A developed database performance monitoring tool is suitablefor both SQL Server and Oracle.

The proposed system is useful in two scenarios that imply different processes:

- Monitoring DB current status and DB activity.
- Notification about the alerts for CPU workload, TOP SQL, capacity problem, tablespace, etc.
- Notification about the possible troubles from historical data occurred over the same period day/month/hours.

The system collects vital information from the network environment and stores it. Customized reports with historical database metrics make it possible to analyze trends, determine whether critical resources are available and are over- or underutilized, and help to decide which capacity planning initiatives make sense.

The gathering of database performance statistics is executed by native database tools. Existing Oracle features for diagnostic, optimization, performance, and workload stabilization in the database include (see Table 1) [9]:

- Automatic Workload Repository (AWR).
- Automatic Database Diagnostic Monitor (ADDM).
- Automatic SQL Tuning.
- Automatic Performance Diagnostics and Tuning Options.

Table 1. Automatic performance diagnostics and tuning options

Options	Overview	Limitation
AWR	The AWR automatically captures performance metrics in a snapshot and allows to make a report based on the differences of snapshots [10]	Oracle Diagnostics Pack and Tuning Pack are required (addition licensing). Only aggregated information is accessed. Some data can be 'omitted' [11]. Performance tuning with ADDM can often involve multiple iterations
ADDM	The ADDM tracks changes in database performance by leveraging the data in the AWR. It reports top problems in a snapshot and provides a holistic tuning solution [10]	
Automatic SQL Tuning	The query optimizer generates the execution plans of the SQL statements [12]	Accessible from Enterprise Manager. Do not guarantee the best execution plan

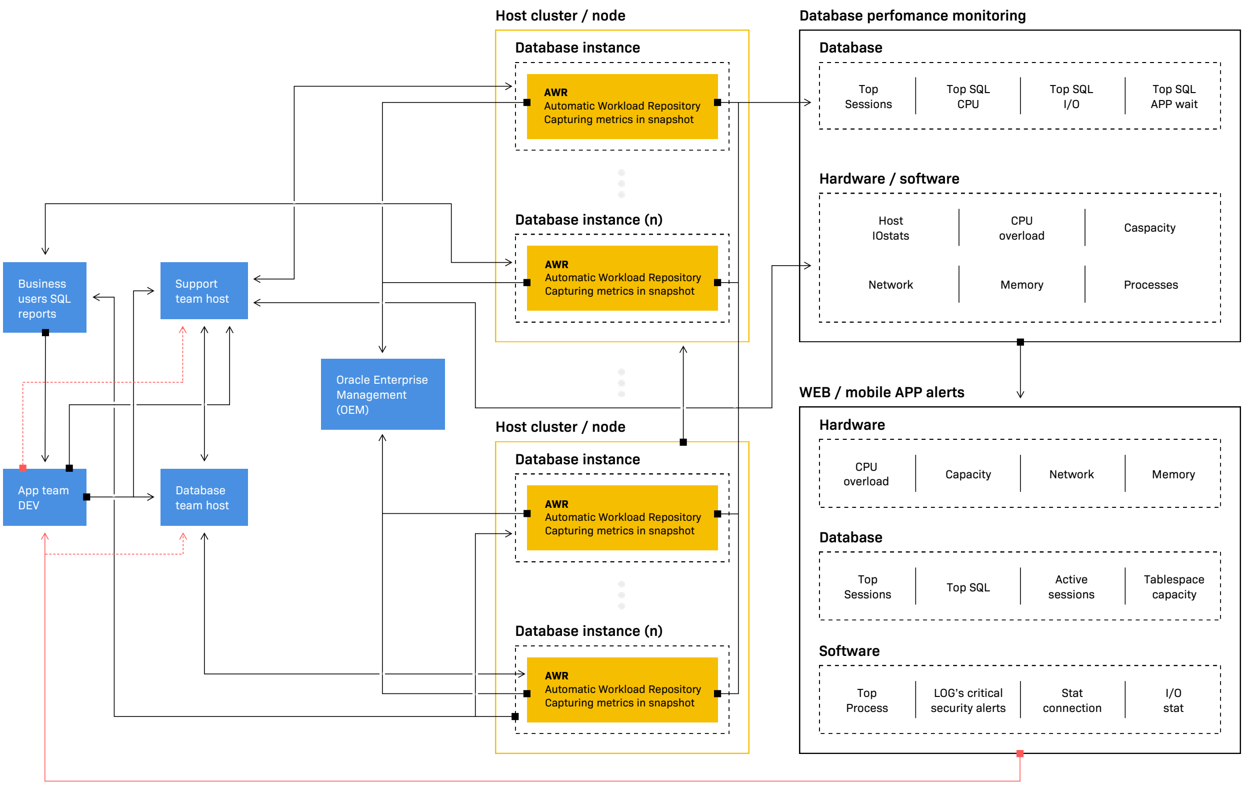

Figure 1 — A centralized monitoring approach for multi-database performance monitoring

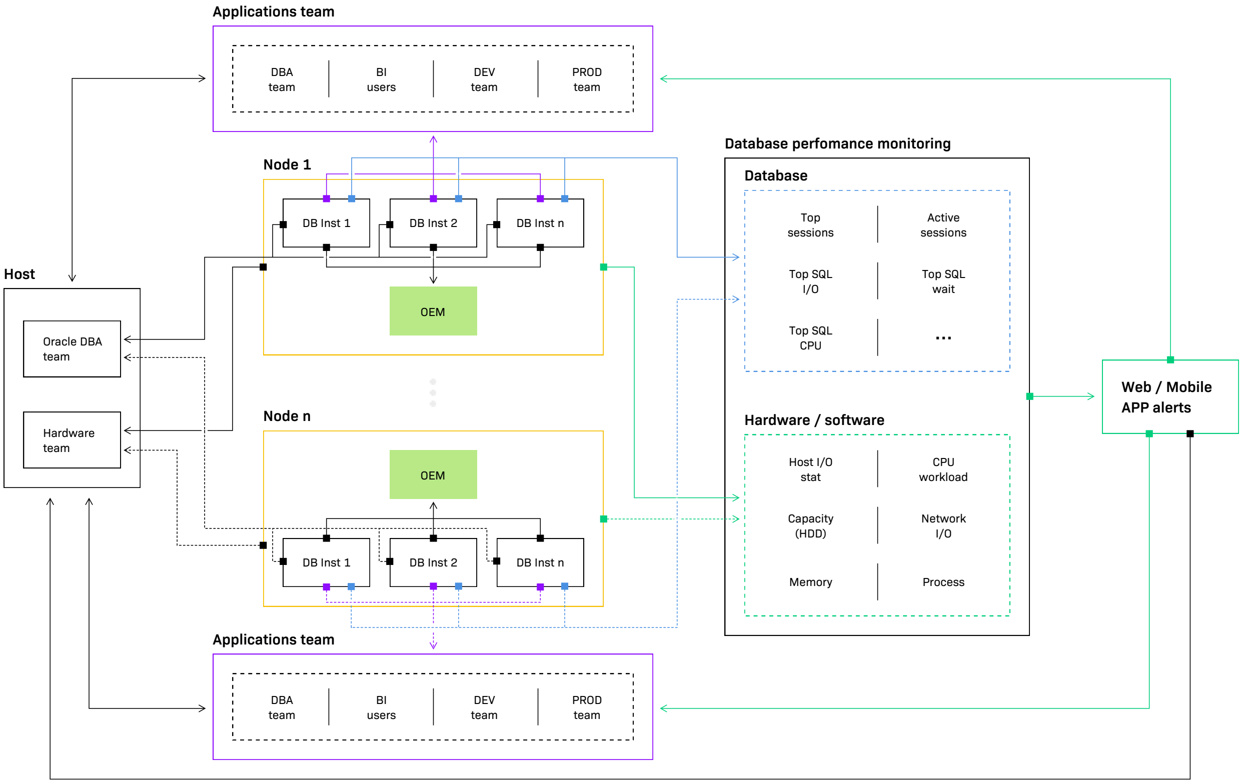

Figure 2 — The consolidated decision for monitoring of medical databases

The AWR is a repository of historical performance data where statistical data is collected and processed for problem detection and self-tuning purposes. The gathered data is stored both in memory and in the database and can be displayed in reports and views [11]. By default, Oracle Database automatically generates sets of historical data for specific time periods (snapshots) once every hour and retains the statistics in AWR for eight days.

The ADDM automatically analyzes pairs of AWR snapshots and provides assistance in resolving performance problems. The ADDM works in three modes: analyze, diagnose, and recommend.

The Automatic Tuning Optimizer performs four types of tuning analysis:

- Statistics Analysis;
- SQL Profiling;
- Access Path Analysis;
- SQL Structure Analysis.

When the Oracle database executes SQL statements, the query optimizer operates in two modes: a normal mode and a tuning mode [12]. SQL Tuning Advisor uses the Automatic Tuning Optimizer to perform an analysis. Optimizer statistics describe more details about the databaseand the objects in the database. These statistics are used by the query optimizer to choose the best execution plan foreach SQL statement [13] and include:

- Table statistics;
- Column statistics;
- Index statistics;
- System statistics.

In real applications, the sizes of some tables may increase much faster than others while some table sizes remain constant. It is well known that queries get slower with additional data in the database [14]. In some cases, automatic SQL tuning does not guarantee the best execution plan. This means that troubleshooting and/or performance tuning by database administrators is necessary. Besides that, there are many other reasons for why the response time of a database could slow down, and expert knowledge is required to recognize and handle the problem. Therefore, when dealing with multi-database management system, the parameters of existed databases, hardware and software should be taken into account during the monitoring process to gain a complete understanding of what is going on with the system and estimate the achievable performance with a given set of resources.

B. Consolidated key capabilities

To deal with multi-database management system, we developed a unified solution that meets the requirement of performance data and provides historical data sampling for Oracle DBA team, Application team, and Hardware team (Fig. 2). The system is intended for cross-platform database monitoring and performs the following operations:

- Real-time database monitoring in cluster/node. It enables users to display a workload of every particular instance; inform about the performance of the database, automatically gathering statistics on pre-determined time intervals; update data with near real-time interactive time series graphs that minimize time to react to the upcoming issue.
- Notification about the current state of vital DB and host metrics that can be personality configured by the problem, so it minimizes the restart database or application and loss of critical data.
- Automatic monitoring of infringements with preinstalled indicators. Various predetermined indicators are used for monitoring of critical components for problems and the collection of statistics of workload.
- Preventions of possible problems in the database (blocking, predicted and actual execution time for queries, records used, bottlenecks, low memory etc.) with opportunities of the set-up of threshold values. Notice of alarm is realized as alert and popup on screen, as well as sending of alerts by e-mail and mobile application.
- Collecting and storage of information on productivity in Database of Performance monitor database. This serves to provide generalized and detailed information on tendencies and diagnostics of productivity. Results of the comparison between data in real time and historical data can initiate mailing of notices and the notification.
- Group of key performance indicators of a DB.

With the offered scheme of information exchange (see Fig. 2), we provide a quicker response and ensure a predictionof different critical situations in DB to both the application developers teams and DBA on HOST. However, to change parameters or SQL execution plan, the coordination between the teamof developers and the BI users of the application is still necessary. Commonly, BI users of the application report system malfunctions. And only BI users of theapplication can inform accurately about when they see the result from the reports.

C. Database performance monitoring and visualization

All troubled databases and their respective monitoring values and times are displayed on the dashboard. The task involves not only offering an optimal integration into existing e-Health processes but also obtaining a maximum of the ease of use. To meet these requirements, we propose the information panel for monitoring infrastructure of the cross-platform database includes various types of displays: bar chart, time series, line charts, pie charts (e.g., Read/Write, Send/Receive, Load/Free, etc.). We incorporated internal Oracle information and other metrics such as memory usage and CPU usage into the dashboard. Thus, the system displays the following specific database's metrics as key decision support information to DB administrators and analysts:

- Disk I/O

These indicators display information about reading and recording from a disk(s) which are connected in OS. As we deal with different options of a configuration from just connected HDD, SDD, RAID, RAC there are possible different values of reading and record depending on hardware indicators. Values of this indicator depending on tablespace arrangement with tabular data can signal about the shortage of the place on a disk, or about a surplus of data what for a solution will demand to move them on more fast-effective disk with other technology, etc.

- Ethernet

Parameter describes the capacity in terms of how many users enter and how many leave data. Also from dens of an operating system, it is possible to watch the mistakes connected with sending packages to a certain port, address or page and from where there are inquiries, what POST/GET inquiries with what speed packages move. If it belongs in SSH, then it is possible to look through the logs on a brute force of passwords for users and blocking on IP.

- Memory/CPU

OEM shows DB consumption of itself (data come from AWR). For an operating system (OS) is how many resources are used by the process started under this instance. In addition, the statistics on all started processes (instance) is gathered for OS on RAM/CPU.

- Tablespace

On OS it is information about how many real engaged places are available at a disk, plus the forecast when the disk space and the limited indicators as speed, temp, a backup, etc. run out.

- Top SQL

The top processes on CPU, time execution, R/W are displayed.

- Active session

The number of active sessions (connections) to a DB which wait for SQL execution, or wait when

Figure 3 — Performance dashboard screen shot (Demo available from https://turion.info/prm/ [15])

the table for start of SQL of inquiry is released DBA can use this data to understand their current strategy and determine what adjustments need to be made in the future. This information enables DBA to perform classification reports by their significance. Potential significant events are further classified as (1) urgent situations that require immediate intervention or (2) events that warrant in-depth analysis.

D. Web/Mobile APP Alerts

The alerting system is based on three key algorithms that are used for the best understanding of the arising problems with databases:

- Comparison with historical data

This algorithm at each poll compares the current value of the variable with the predetermined threshold and reports when the value moves to their limits. Instead of generating an alarm signal every time when the value is close to threshold, the software analyzes historical data to reveal tendencies and real, constant problems.

- A deviation from normal

This algorithm uses the ability to estimate the general through the productivity of all network infrastructures. Instead of comparing the current productivity to the fixed threshold, the software uses historical data for establishment of whether this situation is typical for a particular day and time and then estimates whether the present behavior deviates accepted standards.

- Mode of forecasting of refusals

The prediction mode is used to increase the ability of the DBA to respond to different performance issues. Grounded in history monitoring data, such as CPU, Memory, Storage I/O, Network I/O, etc. and query complexity for various data sizes the performance prediction models evaluate the query execution time and inform DBA about upcoming trouble event [16]. This makes it possible to identify risks and prepare mitigating plan.

Using these algorithms and applying drafting rules, the system can discover a loss of performance in a few hours of observation.

When critical errors are found, scripts send an e-mail to DBA with the notice of possible mismatches.

IV. DATABASE PERFORMANCE PREDICTION MODELS

At the moment, we have developed and implemented two approaches to forecast DB performance troubles closely connected with degradation of its characteristics due to different cumulative effects (congestion by the requests, exceeding time execution, etc.):

- Method of the control chart;
- Method of a hypothesis testing by calculating number of leaps in data.

A. Control charts

The control chart functions by the principle of the online monitoring which is widely used for detection of the emergence of shifts of the process with use of graphic display of characteristics of qualities which has been measured or calculated from a sample in comparison with number or time of selection. As the control chart EWMA is chosen (or the exponential weighted moving average) — this type of the chart, is used for monitoring of variables or attributes of data on the basis of all history of the controlled process [17]. While other control charts process rational subgroups of selections individually, the chart EWMA traces the exponential weighted moving average of all previous samples. As the EWMA parameter can be considered as the average of all past and current observations, it is very tolerant of the assumption of normality. Therefore EWMA is almost ideally nonparametric (without distribution) [18].

B. Sign replacement

The forecasting of the database performance pitfalls due to degradation, as well as the detection of performance issues related to urgent situations, need to answer the question about whether there is a trend in the time series or not. This is solved by statistical tests which demand to specify a modelof a trend.

From the results of previous research on the available databases, it is defined that standard statistical methods [19] do not provide a possibility of assessment of objective variability of observed properties, and in this case, traditional tests without structural features does not ensure reliable and consistent results. Tests produce good results in case of lack of correlation of counting of a time series on which the trend, and for rather large volumes of selection is imposed. Therefore as the generalizing characteristics of nonrandom variability, we use the following specifications [20]:

1) lengths of half-cycles of fluctuations of values of the studied sign (distances between adjacent minima and maxima);
2) amplitudes of fluctuations and gradients of change of values of sign (i.e. speeds of change of its values).

The number of points of sign change in the random sequences depends only on the total of elements of the sequence of N.

If $N > 10$, a statistical distribution of the number of points of the sign change is close to normal with mathematical expectation and dispersion

$$M(t) = \frac{2N - 4}{3} \quad (3)$$

$$\sigma^2(t) = \frac{16N - 29}{90} \quad (4)$$

It enables to utilize the tables of normal distribution at statistical check of a hypothesis of existence of a trend. Hypothesis testing is based on the comparison of the actual value of a number of points of the sign change of t received on the studied chart with its theoretical value $M(t)$ calculated on (3). For obtaining probabilistic criterion, the difference between the actual and theoretical number is divided on $\sqrt{\sigma^2(t)}$

$$Z = \frac{t - M(t)}{\sqrt{\sigma^2(t)}} \quad (5)$$

It is supposed that in the random sequence the value t and $M(t)$ shouldn't differ significantly; therefore, the probability of high values of the Z criterion, in the absolute value, will be small. As values of the Z criterion are distributed normally with parameters 0 and 1, in his size by means of tables of normal distribution it is possible to estimate probability received on the studied number of a deviation of the actual number of points of sign change from theoretical provided that a time series is random. If this probability is low (for example, less than 0.05), then the hypothesis about the random character of the studied time series is rejected and it is considered that it has a trend. The technique of checking the hypothesis on the existence of a trend by the number of leaps can be used in case the ordered sequence consists of two types of elements which conditionally can be divided on two groups of signs "+" and "–".

Leap is the sequence of intervals within one type of the sign. For example, the sequence

$$(+)(+)(-)(-)(-)(-)(+)(+)(-)(-)(+)(-)(-)(-)(-)(+)$$

is divided into seven leaps

$$(++)(----)(++)(--)(+)(----)(+).$$

The sequences of this type can be obtained by division of all values of the studied property into groups depending on their location relative to the median value. All values more than median will obtain the sign "+", and all values less the median will be labeled as "–". The number of leaps in the random sequences depends on quantity of elements of first (n_1) and second (n_2) of type.

Statistical distribution of number of leaps (u) in the random sequences asymptotically normal (which distribution aspires to normal at increase in the amount of selection) with mathematical expectation

$$M(u) = \frac{2n_1 n_2}{n_1 + n_2} + 1 \quad (6)$$

and dispersion

$$\sigma(u) = \frac{2n_1 n_2 (2n_1 n_2 - n_1 - n_2)}{(n_1 + n_2)^2 (n_1 + n_2 - 1)} \quad (7)$$

As in the previous case, the actual value of number of leaps (u) is compared with theoretical value $M(u)$ in the studied time series by criterion:

$$Z = \frac{u - M(u)}{\sqrt{\sigma^2(t)}} \quad (8)$$

Next, the probability of the Z value in the random sequence is determined from the tables of the normal distribution. If this probability is low, the hypothesis about the randomness in the studied sequence is rejected, and we considered that there is a trend.

As an example, have a look at the fragment of the time series in fig. 4.

Having drawn on a profile the line corresponding to median value ($Me = 36$), it is possible to divide observed values into two classes. In this case, the amount of values, lower median $n_1 = 9$, upper median $n_2 = 8$, and the number of leaps $u = 6$.

Then mathematical expectation and dispersion of a number of leaps for random series for $n_1 = 9$ and $n_2 = 8$: $M(u) \approx 9.5$; $\sigma(u) \approx 4$. The value of the Z criterion is $Z = -1.75$, that corresponds to probability 0.04. Thus, at confidential probability 0.95, hypothesis about the random changes of the studied parameter in this direction should be rejected. The trend in this case is established visually.

Both considered ways is applicable for identification of conformities of a certain type. The local laws evoked by smooth changes of the studied properties can be effectively detected by the sign change while the "leaps" works better for overall changes typical to whole time series. Therefore for acceptance of a hypothesis about the trend existence, it is enough to have confirmation from one of the given ways, however existence in experimental results I type errors gives the grounds for development of other methods. At complex studying of the main metrics of a DB when several characteristics for the monotonous changing phenomenon are at the same time fixed, the hidden regularities should be revealed by means of other methods.

V. CONCLUSION

The provision of ubiquitous database performance monitoring gives us the chance to learn about problems which are not connected with a specific instance. We can further explore these problems to find their cause — for example, perhaps other started instances are occupying resources and therefore it is on these instances there are unrelated problems resulting in poor performance.

An important point of this approach is the deployment of the prediction and monitoring systems (as custom scripts and developing prediction model) within DBA tools. As a new contribution, we have shown how these modes can be efficiently implemented into a traditional troubleshooting cycle. When troubleshooting is forecast or analarm is triggered, the DBA will obtain an automatic notification (via e-mail, web-page, etc.) about the recent developments in order to take appropriate action.

The proposed tool is a voluntary reporting system. However, fluctuations of reliable indicators in changes of different reports must be analyzed with deep understanding about what is running and what is happening with data when SQL script are executed.

However, we understand that if we focus only on what is happening at the moment, in most cases, we will lose time and miss the most critical items explaining the real cause of a particular problem in SQL Server. Therefore, an extended forecasting approach concerning database performance will be presented as the subject of another paper. Different cases and troubleshooting forecast strategies will be discussed in cloud database applications as well.

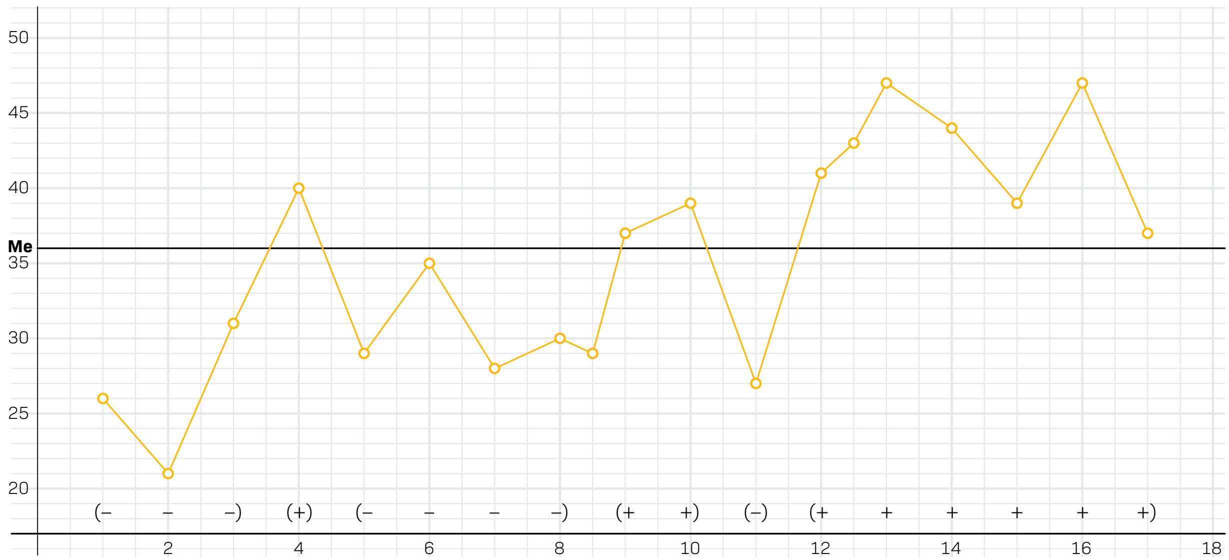

Figure 4 — Characteristics of a time series and checkpoint location relative to the median value

References

[1] SQLSentry [Online]. Available: http://www.sqlsentry.com.

[2] System Center Operations Manager [Online]. Available: https://technet.microsoft.com/en-us/library/hh205987.aspx.

[3] Idera [Online]. Available: https://www.idera.com.

[4] SolarWinds [Online]. Available: http://www.solarwinds.com/database-performance-analyzer-sql-server.

[5] Quest Software [Online]. Available: http://www.dell.com/learn/us/en/vn/secure/acq-quest/.

[6] S. Berselli, K. Peng, and H. Zhu. A Cross-Platform Database Infrastructure Monitoring Dashboard for The Hanover Insurance Group, WPI, 2012.

[7] E. Sarmiento [Online]. Available: http://www.edwinmsarmiento.com.

[8] Foglight solution for cross-platform database performance management Brochure-Foglight 4 Cross Platform DB-US-KS-27723 [Online]. Available: https://www.quest.com/documents/foglight-solutions-for-cross-platform-database-performance-management-datasheet-67983.pdf.

[9] M. Nesterov, I. Skarga-Bandurova. Troubleshooting and Performance Methodology for Business Critical Systems. The 9th IEEE International Conference on Dependable Systems, Services and Technologies, DESSERT'2018, 24–27 May, 2018.

[10] Oracle Database Online Documentation, 12c Release 1 (12.1) / Database Performance Tuning Guide: 6 Gathering Database Statistics, [Online]. Available: https://docs.oracle.com/database/121/TGDBA/gather_stats.htm#TGDBA169, 2018.

[11] AWR, ASH — use cases and 11.2 enhancements, [Online]. Available: https://nanopdf.com/download/content-russian-oracle-user-group_pdf.

[12] Oracle Database Online Documentation, 12c Release 1 (12.1) / Database Performance Tuning Guide: 17 Automatic SQL Tuning, [Online]. Available: https://docs.oracle.com/cd/B28359_01/server.111/b28274/sql_tune.htm#PFGRF028, 2018.

[13] Oracle Database Online Documentation, 12c Release 1 (12.1) / Database Performance Tuning Guide:13 Managing Optimizer Statistics, [Online]. Available: https://docs.oracle.com/cd/B28359_01/server.111/b28274/stats.htm#PFGRF003, 2018.

[14] C. Chi, Y. Zhou, Y. Xiaojun: "Performance prediction for performance-sensitive queries based on algoritmic complexity". Tsinghua Science and Technology, Vol. 18(6), 2013. pp. 618–628.

[15] R&D Lab Turion EUNU [Online]. Available: https://turion.info/prm/.

[16] C. Shallahamer, Forecasting Oracle Performance, Apress, NY, 2014.

[17] EWMA Control Charts NIST/Sematech Engineering Statistics Handbook. National Institute of Standards and Technology. Retrieved 04.2012.

[18] D. Montgomery, Statistical Quality Control. O'Relly, 7th ed., 2012. ISBN 9781118146811.

[19] A. K. Jaiswal, A. Khandelwal. A textbook of computer based numerical and statistical techniques. New Age International (P) Ltd., 2009. p. 624.

[20] A. Kazhdan, Mathematical modelling in geology and EPP, Moscow, 1989.