

УДК 004.65

DOI: [10.26102/2310-6018/2022.39.4.003](https://doi.org/10.26102/2310-6018/2022.39.4.003)

Situation-oriented databases: processing heterogeneous documents of microservices in a document-based storage

A.S. Gusarenko 

Ufa State Aviation Technical University, Ufa, Russian Federation
gusarenko.as@ugatu.su 

Abstract. The research is focused on a situation-oriented approach to the processing of heterogeneous data obtained from microservices that are widespread due to the implementation of the microservice architecture underlying many information systems. Such information systems are sources of heterogeneous data provided to the user upon request via the Internet. Data in the form of documents is provided by services included in the information system. The volume of such data can be large, and its processing requires specialized technologies available in document-oriented big data storages (SODB). As part of a situationally oriented database, a microservice is implemented that provides data in JSON format through its programming interface. There is a problem of loading and processing large amounts of data in the storage where specialized statistical functions of Map-Reduce are implemented. The manual method of loading and obtaining results for SODB is laborious because it requires the implementation of routine operations for loading data, applying functions to the loaded data, creating functions inside the storage and obtaining results. This task was not considered within the scope of the project on creating situation-oriented databases, and the possibilities for developing specialized elements and methods for processing large-scale heterogeneous data in a hierarchical situational model with the required equipment were not studied. The developed models for processing documents make processing heterogeneous data less laborious and help to create data-driven applications by means of situation-oriented databases in the framework of the introduced data processing model as part of a hierarchical situational model with the involvement of big data processing technologies of specialized document-oriented storages. The proposed tools are examined by the example of the SODB application for solving the problems of course design in the educational process using the developed microservice saturated with heterogeneous data collected while designing a course remotely.


Keywords: situation-oriented database, built-in dynamic model, heterogeneous data sources, JSON, document storage, microservices, RESTful-services.

Acknowledgements. This research is supported by RFBR (grant 19-07-00682). The results of the study, reflecting the structure of the developed software solution, have been obtained as part of the state task No. FEUE-2020-0007.

For citation: Gusarenko A.S. Situation-oriented databases: processing heterogeneous documents of microservices in a document-based storage. *Modeling, Optimization and Information Technology*. 2022;10(4). Available from: <https://moitvvt.ru/ru/journal/pdf?id=1247> DOI: 10.26102/2310-6018/2022.39.4.003

Ситуационно-ориентированные базы данных: обработка гетерогенных документов микросервисов в документо-ориентированном хранилище

А.С. Гусаренко 

Уфимский государственный авиационный технический университет, Уфа, Российская Федерация
gusarenko.as@ugatu.su 

Резюме. Работа сосредоточена на ситуационно-ориентированном подходе к обработке гетерогенных данных, получаемых из микросервисов получивших распространение благодаря реализации микросервисной архитектуры, положенной в основу многих информационных систем. Такие информационные системы являются источниками гетерогенных данных, предоставляемых пользователю по запросу через сеть интернет. Данные в виде документов предоставляются сервисами, входящими в состав информационной системы. Объемы таких данных могут быть велики, а для обработки требуются специализированные технологии, имеющиеся в документо-ориентированных хранилищах больших данных (СОБД). В составе ситуационно-ориентированной базы данных реализован микросервис, предоставляющий через свой программный интерфейс данные в формате JSON. Возникает задача загрузки и обработки больших объемов данных в хранилище, где реализованы специализированные статистические функции Map-Reduce. Ручной способ загрузки и получения результатов для СОБД является трудоемким, так как требуется реализация рутинных операций по загрузке данных, применению функций к загруженным данным, созданию функций внутри хранилища и получению результатов. Данная задача не рассматривалась в рамках проекта создания ситуационно-ориентированных баз данных, а возможности по разработке специализированных элементов и методов обработки гетерогенных данных большого объема в иерархической ситуационной модели с требующимся оснащением не исследовались. Разработанные модели обработки документов делают процесс обработки гетерогенных данных менее трудоемким и позволяют создавать собственные приложения на базе ситуационно-ориентированных баз данных, опираясь на введенную модель обработки данных в составе иерархической ситуационной модели с привлечением технологий обработки больших данных специализированных документо-ориентированных хранилищ. Предложенные средства рассматриваются в примере приложения СОБД для решения задач курсового проектирования в учебном процессе с задействованием разработанного микросервиса, насыщенного гетерогенными данными, собранными в процессе дистанционного курсового проектирования.

Ключевые слова: ситуационно-ориентированная база данных, встроенная динамическая модель, гетерогенные источники документов, JSON, хранилище документов, микросервисы, RESTful-сервисы.

Благодарности. Работа выполнена при поддержке РФФИ (грант 19-07-00682). Результаты исследования, отражающие структуру разрабатываемого программного решения, получены в рамках государственного задания № ДВУЭ-2020-0007.

Для цитирования: Гусаренко А.С. Ситуационно-ориентированные базы данных: обработка гетерогенных документов микросервисов в документо-ориентированном хранилище. *Моделирование, оптимизация и информационные технологии*. 2022;10(4). Доступно по: <https://moitvvt.ru/ru/journal/pdf?id=1247> DOI: 10.26102/2310-6018/2022.39.4.003 (на англ.).

Introduction

In the current context, there is an increase in the number of information systems that focus in their development on technologies providing remote access to user functions via the Internet. This model of software distribution ensures the growth of many web services operating on the basis of RESTful protocol [1]. Web services for using RESTful include implemented APIs (API – Application Programming Interfaces) – the interfaces that provide third-party access to automated functions and data through a single-entry point. The current stage in the creation of web systems is to build services based on the architecture of "microservices" [2], where a microservice is presented as a module that has its own limited set of automated functions, its own database separated from other databases in the architecture of the information system and its microservices.

Microservices are considered here as rich data single entry points through which persistent responses are requested with interest fragments of its heterogeneous data [3]. Responses to the user can consist of data portions or include large documents that need to be

processed programmatically using special tools for working with big data [4]. This task cannot be solved by conventional means; therefore, big data and data warehouse technologies are used such as Map-Reduce [5] – a technology for mapping and reducing big data and Bulk-loading [6] – a technology for loading entire data collections into a document repository.

For situationally-oriented databases (SODB) [7], the task of processing documents received from microservices is relevant, on the one hand, it is required to receive data in model states, and on the other hand, load them into a third-party storage along with the Map-Reduce functions created for this data, thus shaping the data representation as "views" containing these two functions. Another topical aspect of this task is the methods of creating collections in the storage, loading data in whole (Bulk-loading) or portionwise. At the same time, it is important that these steps are performed programmatically and not through the existing storage and microservice user experiences so that it would be possible to embed the tools in a hierarchical situational model, where you can specify in the states the microservice data with a request using a single entry point, data format which it is required to obtain data as well as data from the storage involved in data processing. The storage data includes the specification of the Map-Reduce functions in the view and the method of loading data received from the microservice. For the final stage, it is required to receive a response from the storage with the results of processing documents in the collection created by the Map-Reduce functions, and, in addition to this, error handling is required when receiving and loading data. In this paper, the possibility of equipping a hierarchical situational model [8] of SODB with similar tools for processing heterogeneous data with the introduction of new elements and parameters into the model to ensure the functionality of processing big data is explored.

The implementation of these features helps to satisfy the needs of processing documents from microservices in the popular JSON format [9], reduce the complexity of processing such type of data and cover data from many similar microservices while processing data more efficiently by using Map-Reduce technology. The proposed implementation is based on a situation-oriented database platform with a built-in hierarchical situational model, where data from heterogeneous sources, including microservices, is processed in its states.

Literature review

According to published data on situationally oriented databases, research on heterogeneous data sources was considered; documents in XML, JSON, Visio, Word formats, archives and data from web services [10] were processed as heterogeneous data. It was possible to map data onto relational database tables data [11]. In the design of a situationally oriented database, the polyglot persistence approach [12, 13] was used. Under its influence, the architecture of the SODB continues to change. Capabilities for processing heterogeneous data sources are added based on the development of additional functionality. In addition to covering sources by diversity, various volumes of documents were processed by means of the stream processing method [14, 15] and the cached method.

As of today, a large volume of scientific and technical publications has been published, where issues from the field of big data are analyzed in detail [16, 17] as well as the trends prevailing in this area and the use of these technologies in the educational process [18, 19, 20]. In this case, this is important because the SODB is currently implementing a research prototype with data processing in the course design web system from the Department of Automated Control Systems of Ufa State Aviation Technical University.

After analyzing publications on "big data" technologies [18, 19, 20], it can be stated with certainty that now, in relation to SODB, there are no means and methods of data processing using Map-Reduce functions [21, 22, 23], although these functions appear in publications, which is not surprising since big data technologies have been actively developing recently. In

SODB, there were tasks that required solving the problem of processing large documents [15], but this is only the first part of data processing, and here we are talking about processing using Map-Reduce functions for JSON documents received from the SODB microservice with subsequent loading into the NoSQL data warehouse and not for XML documents in the stream. Previously, this problem has not been solved in relation to SODB, which underscores the research interest and relevance of using "big data" technologies in processing heterogeneous data from DBMS sources using the example of any storage with implemented "big data" functions such as CouchDB [24, 25, 26].

Processing heterogeneous documents from SODB microservices in a document-oriented data warehouse

Turning to the modern architecture of situationally oriented databases, it is important to present and consider it with due regard for the tasks of processing heterogeneous documents obtained from SODB microservices in a document-oriented storage. The proposed instance of the architecture is shown in Figure 1 where the data of the SODB microservice remained as a source of heterogeneous data, and other sources are not required in the form in which they were considered in the paper on processing office documents [29].

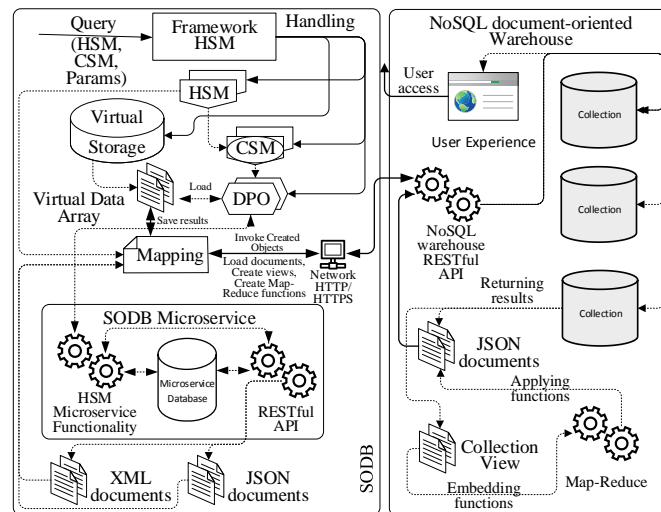


Figure 1 – Architecture for processing heterogeneous documents from SODB microservices in a document-oriented data warehouse

Рисунок 1 – Архитектура обработки гетерогенных документов из микросервисов СОБД в документо-ориентированном хранилище данных

Here, its own developed microservice SODB is used as a source which produces data on students from its database separated from other databases of the course design support information system. The microservice contains functionality for processing data in JSON/XML formats, a database, and an implemented RESTful API, where there is an entry point through which GET requests are sent via the HTTP/HTTPS protocols in the form of a URL with access tokens and parameters of the desired result. In this case, a JSON document with information about students is noteworthy. The documents received from the service must be uploaded in JSON format for processing in a document-oriented storage, regardless of whether it is own microservice or an external call to it is transferred via the RESTful API on the Internet. Another third-party service could be in its place. An example of SODB proprietary microservice is used here. Furthermore, processing in the model state is activated by calling methods in the model state for loading JSON data into a document-oriented data store. It is proposed to load data in

two ways. These are the "Bulk-Load" loading method and the batch loading method. For a successful upload, it is required to implement in the model not only methods for managing data, but also the RESTful API of the used big data store. Since big data processing is achieved using big data technologies, it will also be necessary to create model support with parameters or elements entered into the model. In the model, it will be necessary to provide the parameters for connecting to the microservice and specify the settings for obtaining data from it. The previously used `doc` element with new parameters focused on the nature of the sources (the microservice and NoSQL document-oriented data storage) may well be suitable for this. Elements that have the specifics of processing big data could have parameters for calling "Bulk-Load" methods or a batch method. Additionally, the task of creating a view that serves as a container for the Map-Reduce functions arises in the course of the initiated processing. It is assigned to some data set, i.e. a collection. Inside the view, the text of the Map-Reduce functions is specified. For their deployment via the RESTful API of the storage, the SODB model must also contain either their text in JavaScript or a link to a file with the source text of the functions. This can be implemented as element parameters or new model elements can be introduced. After the data, views and functions are loaded into the storage, this view is called and the processing result is obtained. For this, the `src` data receiver is provided in the model which is responsible for unloading the result with error handling in the `err` element.

Retrieving data from the SODB microservice. In our example, a data processing model is required (see Figure 2, *a, b*) to connect with a virtual machine on which document-oriented data store is deployed. CouchDB storage chosen to demonstrate processing capabilities [29]. A model with a virtual array of documents has been created for this data warehouse (see Figure 2, *a*) Previously, the authors developed the concept of virtual multi-documents and their structuring through the use of entry-elements in a similar style and taking into account the input elements [30–31]. The external document storage Data was specified in the `doc` element with the parameter `host="http://gusar.servepics.com"` meaning that the store is on the server and is accessible via a RESTful single-entry point.

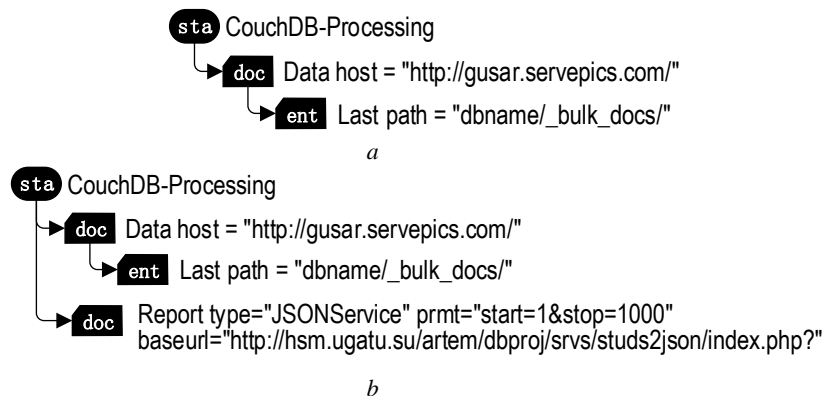


Figure 2 – Specifying a virtual document in a model with a mapping onto a document-oriented data store (*a*), extracting data from the SODB microservice and mapping it onto a document-oriented data store (*b*)

Рисунок 2 – Задание виртуального документа в модели с отображением на документо-ориентированное хранилище данных (*a*), извлечение данных из микросервиса СОБД и последующее отображение на документо-ориентированное хранилище данных (*b*)

Next is the `ent` element denoting the structuring part of the heterogeneous source and the part of the URL that helps to map data with the "Bulk-Load" method. To solve the problem of loading data, it is necessary to continue working from the stage of obtaining data from the SODB microservice. In this case, the SODB service for course design of students is suitable

where the completion of the stages of the course project is recorded. The functionality of the microservice allows the user to request data in XML or JSON formats. The model with the specification of a data request from the SODB microservice is shown in the `doc` element in Figure 2, *b*. The name of this microservice is `Report` and its type is specified in the `type="JSONService"` parameter. The single entry point for requests is specified in the `baseurl` where the address of the microservice is set `baseurl=http://hsm.ugatu.su/artem/dbproj/srvs/studs2json/index.php?`. Additionally, the parameters `prmt="start=1&stop=1000"` are sent with the request which is the designation of the start position `start=1` and the end position `stop=1000`. In this way the volume of the number of nodes loaded in the JSON file is regulated.

Structuring the SODB model with elements and parameters for processing large documents in a document-oriented storage

Extending the model with new structural parameters and elements to create a view in a document-oriented data warehouse. Situation-oriented databases have not currently used technologies for processing large documents from an external document-oriented storage in contrast to the technologies for streaming processing large documents [4]. At this stage, it is required to extend the model capabilities with the functionality of processing large documents by introducing new `map` and `rdc` elements into the multidocument, thus structuring it according to the functions in Figure 3, *a, b*.

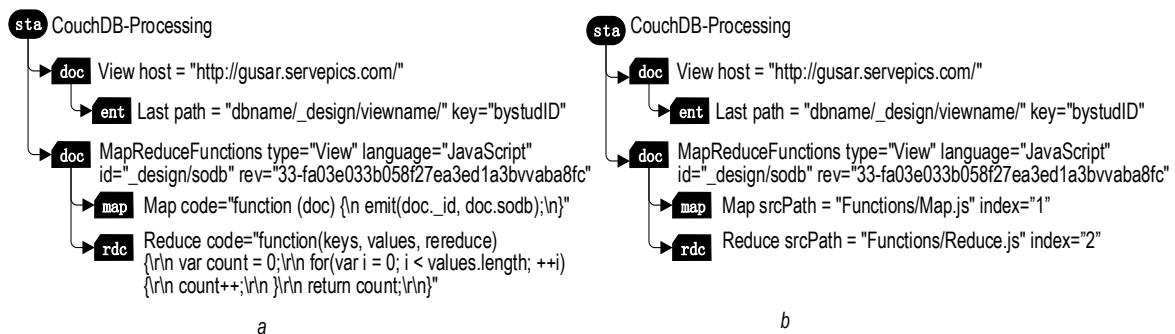


Figure 3 – A fragment of a model with a multi-document structured by elements of processing large documents in a document-oriented data warehouse, with involvement in the processing of the source code of functions in model (*a*), with involvement in the processing of the source code library of functions (*b*)

Рисунок 3 – Фрагмент модели с мультидокументом, структурированным элементами обработки больших документов в документо-ориентированном хранилище данных, с привлечением к обработке исходного кода функций в модели (*a*), с привлечением к обработке библиотеки исходных текстов функций (*b*)

At the same time, the `entry` element here has a new `key="bystudID"` parameter since such parameters can be sent to the storage to return documents that match the given key. The `doc` element takes on additional `id` and `rev` parameters as required by the repository to specify the `id` and version of the document. The `doc` element must specify the `language` parameter which gives information to the interpreter about the nature of the data contained in the `map` and `rdc` structural elements. The base storage language is JavaScript using the model fragment from Figure 3, *a* as an example and the `Map` element contains the code in the input code parameter in JavaScript. This is the source code for the `Map` function loaded into the store.

Next comes the `rdc` element which also contains the JavaScript code to deploy the Reduce function to the store. Figure 3, *a, b* shows two options for placing functions. It remains possible to write the function code into the model directly, i.e. when it is not required to process data often and the Map_Reduce functions are standard and do not form a library. Consequently, the `code` parameter from Figure 3, *a* can be used. With the growth in the number of functions and the need to reduce the text of the model, an option is provided when the functions are placed in the library. This is shown in Figure 3, *b*. The `srcPath` parameter is already used here which sets the address of the `Functions/Map.js` and `Functions/Reduce.js` libraries. This is due to the fact that it is required to process a document using various functions. Pairs of functions can follow each other, and the elements `doc`, `map`, `rdc` form the so-called representation in a document-oriented storage environment with technology for processing large documents. In order to apply different pairs of functions, the `map` and `rdc` elements contain indexes in the `index` parameters which can be specified in the model states to select the desired function from the library.

The state of the submodel for loading data from the SODB microservice into a document-oriented storage. The SODB data processing model consists not only of the description of documents and data arrays but also of the states in which the data processing task arises. Therefore, it is possible to consider a fragment of the model where it is clearly required to process data. Figure 4, *a, b* shows such a fragment of the model, where there is a sub-model named `proc` and the state of the same name containing a `dpo` data processing object named `DPO_Load_Data.`

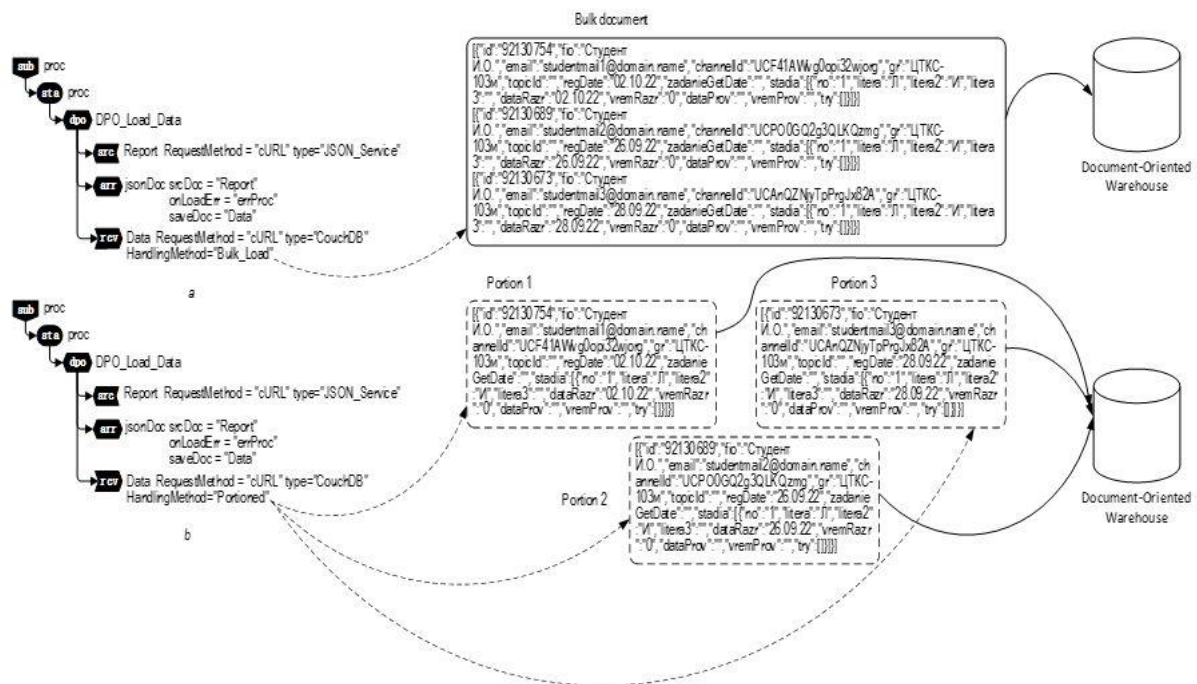


Figure 4 – Fragment of a hierarchical situational model of loading JSON data into a document-oriented storage using the "Bulk-Load" method

Рисунок 4 – Фрагмент иерархической ситуационной модели загрузки JSON-данных в документо-ориентированное хранилище методом "Bulk-Load"

This data processing object solves the problem of loading data into the storage. The model contains a `src` element named `Report`, an indication of the request method using the `cURL` extension, connection to the microservice of the typified as `JSON_Service`, which

means that this microservice produces data in JSON format. Next, the `jsonDoc` element `arr` is placed, which refers in the `srcDoc` parameter to the `Report` virtual document considered in the model fragment from Figure, 2, *b* with the result saved in a document-oriented storage; the `saveDoc` parameter refers to the data receiver specified in the model further by the `rcv` element in which there is a link to the virtual single-entry `Data` document seen previously in Figure 2, *a*.

Bulk-Load data loading method. The `rcv` receiver element also has the storage type `type="CouchDB"` and its `Bulk_Load` method. The interpreter, bypassing this model in the interpretation cycle, calls for work not only the `cURL` extension but also the functionality for working with a document-oriented data store and prepares the data of the virtual data processing object for loading JSON "in its entirety". This method helps to upload data as a common file, but the opportunity remains to upload multiple files, which will increase the time it takes to load data into the storage.

Portioned data loading method. In the receiver element, you can also set an alternative loading method `Portioned` which differs in that the document is divided into portions and then each portion is loaded in turn. It should be noted that in this case the description of the multi-document changes; the entry point is `http://gusar.servepics.com :5984/dbname/` not `http://gusar.servepics.com:5984/dbname/_bulk_docs/`. Portions can be formed from nodes. These nodes are transferred to the storage, where a queue is formed and then the storage continues creating documents in the collection after the load script completes.

Creating views in a document-oriented data store. Having dealt with the specifications in the SODB model for loading data and with the methods of sending them to the storage, the focus is shifting to such a toolkit of a document-oriented storage as a "view". This tool serves as a container for technologies such as Map-Reduce precisely in view of the detailed description of the statistics of the Map and Reduce functions. When such a view is called, it sends a function to the document collection to obtain the results. The view has the appearance of a separate document with an ID and a document modification date and is styled using JavaScript. In other words, a view in a document-oriented data store is the text of a program in a document bound to a data collection.

In SODB, it is proposed to specify the representation in the states of the submodel. In our example from Figure 4, *a, b*, there are two options for methods but both of them add an additional data processing object with a specification referring to a multi-document. An example of such a data processing object is shown in Figure 5 with the name `DPO_Create_View`. The `MapReduceFunctions` data source is specified for it with the content in JavaScript.

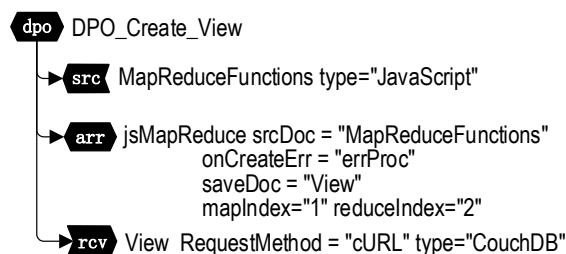


Figure 5 – Model fragment with an example of a data processing object containing specifications for creating a view in a document-oriented data store

Рисунок 5 – Фрагмент модели с примером объекта обработки данных, содержащего спецификации создания представления в документо-ориентированном хранилище данных

The way the `src` is handled in the `arr` element where there is a reference to the error-handling state since the creation of the view can be accompanied by errors. According to the results of processing, it is required to save the document. In the `arr` element named `jsMapReduce`, the `saveDoc="View"` parameter is provided which enables saving as a view. When the view is created, many variants of the Map and Reduce functions, indicated in the `MapReduceFunctions` multi-document description structure from Figure 3, *b*, can be involved in it if necessary. When creating a view, it is necessary to determine which functions will be used to place it in the view. This task is proposed to be solved by introducing indices in the parameters of the `arr` element such as `mapIndex` and `reduceIndex`. The pairs of Map and Reduce functions are defined by the numbers 1 and 2 and they must match the same indexes inside the multidocument.

When a view document is formed in the virtual data processing object, a receiver is defined for it (see the Figure 5). It shows a `rcv` receiver element with the name `View` and the `RequestMethod="cURL"` creation method. The storage type is determined by the `type="CouchDB"` parameter. The volume of view documents is small compared to the data itself, so the specialized methods for loading, discussed earlier with the data example (see the Figure 4), are not proposed.

Case study: processing student's data in a document-oriented warehouse

A hierarchical situational model of data processing from a SODB microservice is considered using an example shown in Figure 6.

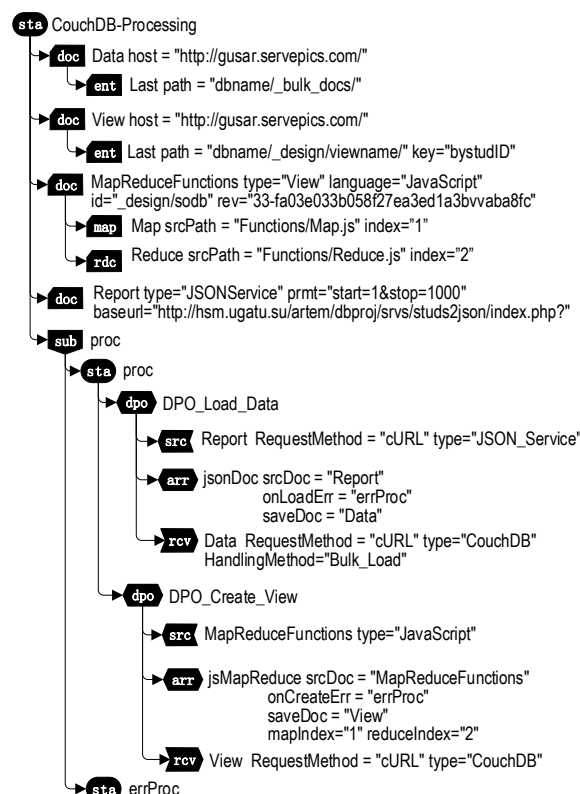


Figure 6 – General view of the hierarchical situational model of SODB for data processing in a document-oriented storage

Рисунок 6 – Общий вид иерархической ситуационной модели СОБД для обработки данных в документо-ориентированном хранилище

In this case, the studs2json microservice supplies large volume data that is loaded into a document-oriented storage based on SODB using the Bulk_Load method. The second stage is the loading of the view with the Map-Reduce functions used to obtain data on students. A multi-document View is specially created for the DPO_Create_View data processing object, where the entry point is specially since a special entry point is provided in the document-oriented storage in the API for views.

A sample microservice with up-to-date data on students is shown in Figure 7. Requests to the service are sent via HTTP. The GET request URL line indicates the number of nodes requested for processing. In Figure 7, the results of the request <https://hsm.ugatu.su/artem/dbproj/srvs/studs2json/index.php?start=1&stop=10000> are presented in the browser in the form of a JSON document.

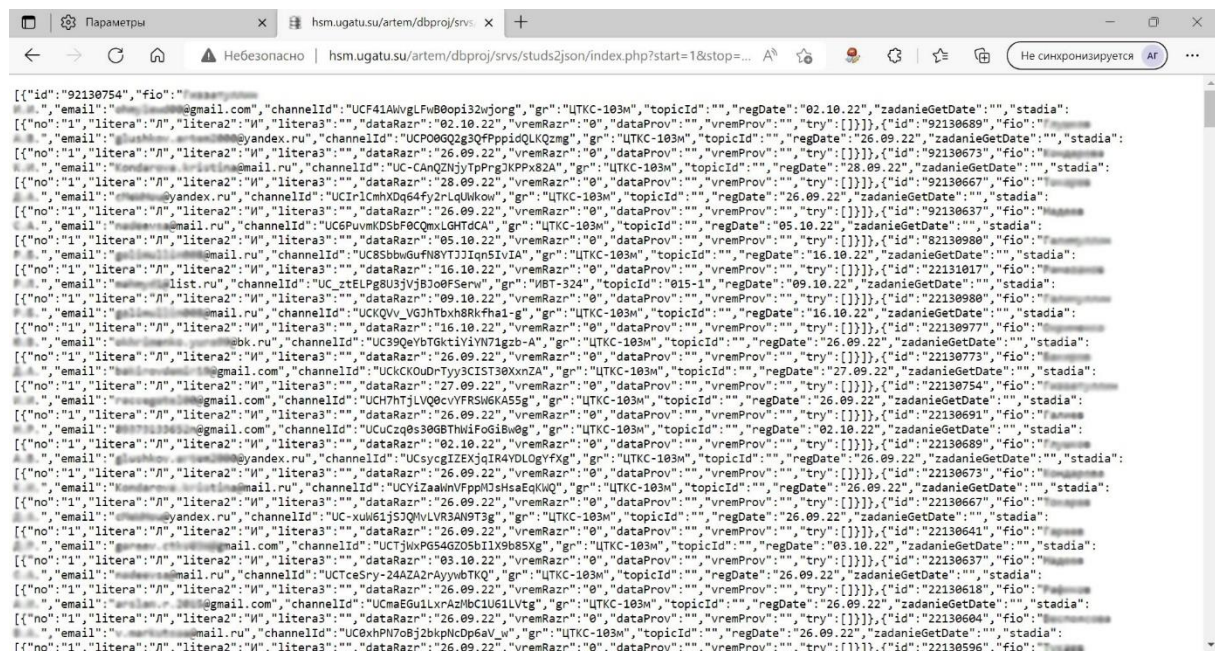


Figure 7 – Retrieving students’ data from the stud2json SODB microservice (students' data has been changed)

Рисунок 7 – Извлечение данных по студентам из микросервиса stud2json СОБД (данные студентов изменены)

The practical implementation and testing of the results was carried out using the well-known Apache CouchDB data warehouse [29], where there is an implemented Map-Reduce technology and a Bulk-Load loading method.

The processing of large documents from microservices has found its practical reflection in the SODB project. The screen with student data is shown in Figure 8 where it is possible to see the list of students by their IDs. For each student, the nodes of the document are extracted which show stage completion by each student with a certain status. Such tasks are often required to be solved, when it is necessary to involve the statistical functions prescribed in the data representations while constructing summary charts and tables. By using the model from Figure 6, it is possible to reduce the programming effort and avoid redundant data processing code in the SODB. The new specifications of multi-documents and data processing objects introduced into the model reduce the program code of the hierarchical situational model by 8-10 times [3, 7].

The screenshot displays the 'Курсовой проект «Базы данных»' (Course Design Project «Databases») interface. It features a navigation menu on the left with sections like 'ИНСТРУКЦИИ', 'ВИДЕОУРОКИ ТЕОРИЯ', 'ВИДЕОУРОКИ VISIO', and 'ИНСТРУМЕНТЫ'. The main content area shows a table titled 'Учебное полугодие 2022-2 — Зарегистрированные студенты:' (Academic semester 2022-2 — Registered students:). The table lists 24 students with their IDs, topics, groups, names, addresses, and progress status. The 'Состояние' (Status) column shows various states such as 'Ждет проверки' (Waiting for check), 'Ошибки' (Errors), and 'Зачтено' (Completed).

No	Код	Тема	Группа	ФИО	Адрес	Этап	Состояние	Видео	Зачтено
1	20130104	010-1	ПИ-325	Евдокимов А.А.	evdokimov.a.a@gmail.ru	9	- Ждет проверки		
2	20130491	011-1	ПИ-326	Зыкина А.А.	zykina.a.a@gmail.ru	3	- Ждет проверки		
3	20130731	017-1	ПРО-330	Сидорова С.С.	sidorova.s.s@gmail.com	2	- Ждет проверки		
4	20131137	001-3	ИВТ-324	Ильинский И.И.	ilyinskiy.i.i@gmail.com	2	- Ждет проверки		
5	20130481	001-1	ПИ-326	Федорова И.И.	fedorova.i.i@yandex.ru	2	- Ждет проверки		
6	20130555	011-1	ПРО-329	Иванова А.И.	ivanova.a.i@yandex.ru	2	- Ждет проверки		
7	20130178	002-1	ПИ-326	Ивановский С.А.	ivanovskiy.s.a@gmail.com	2	- Ждет проверки		
8	20130790	006-1	ИВТ-326	Давыдов С.С.	davydov.s.s@yandex.ru	2	- Ждет проверки		
9	20130370	002-4	МО-321	Губинский Г.С.	gubinskiy.g.s@yandex.ru	2	- Ждет проверки		
10	20130371	025-1	МО-322	Петров Г.Г.	petrov.g.g@gmail.com	2	- Ждет проверки		
11	20130088	001-1	ПИ-325	Сидорова Т.М.	sidorova.t.m@mail.ru	2	- Ждет проверки		
12	20130382	017-1	МО-321	Петрова П.А.	petrova.p.a@gmail.com	2	- Ждет проверки		
13	20130846	005-1	ПИ-326	Михайлова А.М.	mikhailova.a.m@gmail.com	2	- Ждет проверки		
14	20130329	007-1	ИВТ-326	Сидорова С.С.	sidorova.s.s@mail.ru	2	- Ждет проверки		
15	20130720	025-1	ПИ-325	Петрова А.А.	petrova.a.a@gmail.com	1	- Ждет проверки		
16	2030830	021-1	ПИ-325	Николаев А.А.	nikolaev.a.a@mail.ru	1	- Ждет проверки		
17	20130824	001-3	ПИ-325	Александров А.А.	alexandrov.a.a@gmail.com	1	- Ждет проверки		
18	20130337	010-1	ПИ-324	Сидорова С.С.	sidorova.s.s@mail.ru	1	- Ждет проверки		
19	20130421	014-1	ИВТ-324	Петрова Т.М.	petrova.t.m@gmail.com	1	- Ждет проверки		
20	20130330	016-1	ИВТ-326	Сидорова С.С.	sidorova.s.s@mail.ru	5	- Ошибки		
21	20130529	001-1	ПРО-328	Петрова С.С.	petrova.s.s@gmail.com	5	- Ошибки		
22	20130547	002-4	ПРО-327	Миронов И.И.	mironov.i.i@gmail.com	5	- Ошибки		
23	20130366	016-1	МО-321	Ивановский С.С.	ivanovskiy.s.s@mail.ru	4	- Ошибки		
24	20130322	010-1	МО-321	Иванова А.А.	ivanova.a.a@gmail.com	3	- Ошибки		

Figure 8 – The interface of the "course design" project on databases with data on students (students' data has been changed)

Рисунок 8 – Интерфейс проекта "курсового проектирования" по базам данных с данными по студентам (данные студентов изменены)

Conclusion

This article is devoted to the development of a SODB model for processing heterogeneous documents from microservices in a document-oriented data warehouse. It has been established that in the model, due to the introduction of new structured elements of multi-documents, it is possible to specify microservices and views mapped onto a real data warehouse while additional parameters for managing loading methods are also introduced into data processing objects. Owing to the specification inside the data processing objects, it becomes possible to control the Bulk_Load and Portion data loading methods proposed in the SODB. All the proposed tools fit into the modern microservice architecture, which also enables the use of external microservices as sources of heterogeneous SODB data.

To involve Map-Reduce technologies in data processing, a high-level abstraction model is used which reduces labor intensity compared to manually loading data into the storage, thus minimizing the use of the storage user interface. The research is carried out almost entirely using the SODB user experience. All proposed solutions were tested on a research prototype in a course design project for the course on databases.

REFERENCES

1. Wilde E., Pautasso C. REST: From Research to Practice. *Springer Science & Business Media*. 2011:528. DOI:10.1007/978-1-4419-8303-9.
2. Mironov V.V., Gusarenko A.S., Yusupova N.I. Situation-oriented databases: polyglot persistence based on REST microservices. *Prikladnaya informatika = Applied Informatics*. 2019;5(83):87–97. DOI:10.24411/1993-8314-2019-10038. (In Russ.).
3. Mironov V.V., Gusarenko A.S., Tuguzbaev G.A. Extracting semantic information from graphic schemes. *Informatika i avtomatizatsiya = Informatics and Automation*. 2021;20(4):940–970. DOI:10.15622/IA.20.4.7. (In Russ.).
4. Gusarenko A.S., Mironov V.V., Yusupova N.I. Stream processing of large documents in situationally oriented databases. *ITIDS'2018: Trudy 6-oy mezhdunarodnoy konferentsii*

- po Informatsionnym tekhnologiyami intellektual'noy podderzhki prinyatiya resheniy = ITIDS'2018: Proceedings of the 6-th International Conference Information Technologies for Intelligent Decision Making Support. Ufa, Russia: USATU; 2018:7–12. (In Russ.)*
5. Seera N.K., Taruna S. Analyzing Cost Parameters Affecting Map Reduce Application Performance. *International Journal of Information Technology and Computer Science*. 2016;8(8):50–58. DOI:10.5815/ijitcs.2016.08.06.
 6. Papadopoulos A., Manolopoulos Y. Parallel bulk-loading of spatial data. *Parallel Computing*. 2003;29(10):1419–1444. DOI:10.1016/j.parco.2003.05.003.
 7. Mironov V.V., Gusarenko A.S., Tuguzbaev G.A. Situation-oriented databases: the formation of personalized graphic documents for educational design support. *Modelirovaniye, optimizatsiya i informatsionnyye tekhnologii = Modeling, Optimization and Information Technology*. 2020;8(2):1–18. DOI:10.26102/2310-6018/2020.29.2.013. (In Russ.).
 8. Gusarenko A.S. Improvement of Situation-Oriented Database Model for Interaction with Mysql. *Izvestiya vysshikh uchebnykh zavedeniy Priborostroyeniye = Journal of Instrument Engineering*. 2016;59(5):355–63. DOI:10.17586/0021-3454-2016-59-5-355-363. (In Russ.).
 9. Mironov V.V., Gusarenko A.S., Yusupova N.I., Smetanin Y.G. JSON documents processing using situation-oriented databases. *Acta Polytechnica Hungarica*. 2020;17(8):29–40. DOI:10.12700/APH.17.8.2020.8.3.
 10. Mironov V.V., Gusarenko A.S., Yusupova N.I. Monitoring YouTube Video Views in the Educational Environment Based on Situation-Oriented Database and RESTful Web Services. *Sistemnaya inzheneriya i informatsionnyye tekhnologii = Systems Engineering and Information Technologies*. 2021;3(1(5)):39–49.
 11. Mironov V.V., Gusarenko A.S., Yusupova N.I. Building of Virtual Multidocuments Mapping to Real Sources of Data in Situation-Oriented Databases. *Communications in Computer and Information Science*. 1204 CCIS. 2021:167–178. DOI:10.1007/978-3-030-78273-3_17.
 12. Kolonko M., Mullenbach S., Polyglot Persistence in conceptual modeling for information analysis. *ACIT'2020: Proc. 10th Int. Conf. on Advanced Computer Information Technologies*. 2020:590–594. DOI:10.1109/ACIT49673.2020.9208928.
 13. Mironov V.V., Gusarenko A.S., Yusupova N.I. Situation-oriented databases: polyglot persistence based on REST microservices. *Applied Informatics*. 2019;14(5(83)):87–97. DOI: 10.24411/1993-8314-2019-10038. (In Russ.).
 14. Mironov V.V., Gusarenko A.S., Yusupova N.I. Stream handling large volume documents in situationally-oriented databases. *International Scientific Journal INDUSTRY 4.0 Scientific Technical Union of Mechanical Engineering "INDUSTRY 4.0."* 2018;3(5):240–4.
 15. Mironov V., Gusarenko A., Yusupova N. Stream Documents Processing Invariance in Situation-Oriented Databases. *7th Scientific Conference on Information Technologies for Intelligent Decision Making Support (ITIDS'2019)*. Atlantis Press; 2019:309–15. DOI:10.2991/itids-19.2019.55.
 16. Sakr S., Zomaya A.Y. Handbook of Big Data Technologies [Internet]. Springer International Publishing; 2017. Available from: <https://link.springer.com/book/10.1007/978-3-319-49340-4> (accessed on 12.10.2022).
 17. Sakr S., Zomaya A.Y. Encyclopedia of big data technologies. *Springer International Publishing*; 2019. DOI: 10.1007/978-3-319-77525-8.
 18. Bakshi K. Technologies for Big Data. *IGI Global*; 2014:1–22. DOI:10.4018/978-1-4666-4699-5.ch001.
 19. Singh S., Singh P., Garg R., Mishra P.K. Big Data: Technologies, Trends and

- Applications. *International Journal of Computer Science and Information Technologies*. 2015;6(5):4633–4639.
20. Mamedova G.A., Zeynalova L.A., Melikova R.T. Big data technologies in e-learning. *Open Education*. 2018;16;0(6):41–8. DOI: 10.21686/1818-4243-2017-6-41-48.
 21. Jiang D., Ooi B.C., Shi L., Wu S. The performance of MapReduce: an in-depth study. *Proceeding Proc VLDB Endow*. 2010;3(1–2):472–483. DOI:10.14778/1920841.1920903.
 22. Shim K. MapReduce Algorithms for Big Data Analysis. In: *Madaan A, Kikuchi S, Bhalla S, editors. Lecture Notes in Computer Science. Databases in Networked Information Systems*. Berlin, Heidelberg: Springer; 2013: 44–8. DOI:10.1007/978-3-642-37134-9_3.
 23. Tao Y., Lin W., Xiao X. Minimal MapReduce algorithms. *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*. New York, NY, USA: Association for Computing Machinery; 2013:529–40. DOI: 10.1145/2463676.2463719.
 24. Györödi C.A., Dumșe-Burescu D.V., Zmaranda D.R., Györödi R.Ş., Gabor G.A., Pecherle G.D. Performance Analysis of NoSQL and Relational Databases with CouchDB and MySQL for Application’s Data Storage. *Applied Sciences*. 2020;10(23):8524. DOI: 10.3390/app10238524.
 25. Anderson J.C., Lehnardt J., Slater N. CouchDB: The Definitive Guide: Time to Relax. *O’Reilly Media, Inc.*; 2010:274.
 26. Lennon J. *Beginning CouchDB*. *Apress*; 2010:299.
 27. Gusarenko A.S. Certificate of state registration of the computer program No. 2022617552. Microservice of a situationally oriented database for loading data and Map-Reduce functions into a document-oriented storage. 2022. (In Russ.).
 28. Gusarenko A.S. Certificate of state registration of the computer program No. 2022617505. Situational database modules for extracting large documents and archives from RESTful services of heterogeneous data stores. 2022. (In Russ.).
 29. Apache CouchDB. Available from: <https://couchdb.apache.org/> (accessed on 17.10.2022).
 30. Mironov V.V., Gusarenko A.S., Yusupova N.I. Situation-Oriented Databases: Processing Office Documents. *Modelirovaniye, optimizatsiya i informatsionnyye tekhnologii = Modeling, optimization and information technology*. 2022;28;10(2):1–16. DOI:10.26102/2310-6018/2022.37.2.021.
 31. Mironov V.V., Gusarenko A.S., Yusupova N.I. The Invariance of The Virtual Data in The Situationally Oriented Database When Displayed on Heterogeneous Data Storages. *Vestnik komp'yuternykh i informatsionnykh tekhnologiy = Herald of Computer and Information Technologies*. 2017;(1(151)):29–36. DOI: 10.14489/VKIT.2017.01.PP.029-036 (In Russ.).
 32. Course project "Databases". Available from: <http://hsm.ugatu.su/artem/dbproj/> (accessed on 20.10.2022). (In Russ.).

СПИСОК ИСТОЧНИКОВ

1. Wilde E., Pautasso C. REST: From Research to Practice. *Springer Science & Business Media*. 2011:528. DOI:10.1007/978-1-4419-8303-9.
2. Миронов В.В., Гусаренко А.С., Юсупова Н.И. Ситуационно-ориентированные базы данных: polyglot persistence на основе REST-микросервисов. *Прикладная информатика*. 2019;14(5(83)):87–97. DOI:10.24411/1993-8314-2019-10038.
3. Миронов В.В., Гусаренко А.С., Тугузбаев Г.А. Извлечение семантической информации из графических схем. *Информатика и автоматизация*. 2021;20(4):940–70. DOI:10.15622/IA.20.4.7

4. Гусаренко А.С., Миронов В.В., Юсупова Н.И. Поточковая обработка больших документов в ситуационно-ориентированных базах данных. *ITIDS'2018: Труды 6-ой международной конференции по Информационным технологиями интеллектуальной поддержки принятия решений*. Уфа, Россия: УГАТУ; 2018:7–12.
5. Hou X, Li N, Yang H, Liang Q. Comparison of Wordprocessing Document Format in OOXML and ODF. *2010 Sixth International Conference on Semantics, Knowledge and Grids*. 2010:297–300. DOI:10.1109/SKG.2010.44.
6. Papadopoulos A., Manolopoulos Y. Parallel bulk-loading of spatial data. *Parallel Computing*. 2003;29(10):1419–1444. DOI:10.1016/j.parco.2003.05.003.
7. Миронов В.В., Гусаренко А.С., Тугузбаев Г.А. Ситуационно-ориентированные базы данных: формирование персонализированных графических документов для поддержки учебного проектирования. *Моделирование, оптимизация и информационные технологии*. 22;8(2):1–18. DOI:10.26102/2310-6018/2020.29.2.013.
8. Гусаренко А.С. Усовершенствование модели ситуационно-ориентированной базы данных для взаимодействия с MySQL. *Известия высших учебных заведений Приборостроение*. 2016;59(5):355–63. DOI:10.17586/0021-3454-2016-59-5-355-363.
9. Mironov V.V., Gusarenko A.S., Yusupova N.I., Smetanin Y.G. JSON documents processing using situation-oriented databases. *Acta Polytechnica Hungarica*. 2020;17(8):29–40. DOI:10.12700/APH.17.8.2020.8.3.
10. Миронов В.В., Гусаренко А.С., Юсупова Н.И. Мониторинг просмотров видео YouTube в образовательной среде на основе ситуационно-ориентированной базы данных и веб-сервисов RESTful. *Системная инженерия и информационные технологии*. 2021;3(1(5)):39–49.
11. Mironov V.V., Gusarenko A.S., Yusupova N.I. Building of Virtual Multidocuments Mapping to Real Sources of Data in Situation-Oriented Databases. *Communications in Computer and Information Science*. 1204 CCIS. 2021:167–178. DOI:10.1007/978-3-030-78273-3_17.
12. Kolonko M., Mullenbach S., Polyglot Persistence in conceptual modeling for information analysis. *ACIT'2020: Proc. 10th Int. Conf. on Advanced Computer Information Technologies*. 2020:590–594. DOI:10.1109/ACIT49673.2020.9208928.
13. Миронов В.В., Гусаренко А.С., Юсупова Н.И. Ситуационно-ориентированные базы данных: polyglot persistence на основе REST-микросервисов. *Прикладная информатика*. 2019;14(5(83)):87–97. DOI: 10.24411/1993-8314-2019-10038.
14. Mironov V.V., Gusarenko A.S., Yusupova N.I. Stream handling large volume documents in situationally-oriented databases. *International Scientific Journal INDUSTRY 4.0 Scientific Technical Union of Mechanical Engineering "INDUSTRY 4.0."* 2018;3(5):240–4.
15. Mironov V., Gusarenko A., Yusupova N. Stream Documents Processing Invariance in Situation-Oriented Databases. *7th Scientific Conference on Information Technologies for Intelligent Decision Making Support (ITIDS'2019)*. Atlantis Press; 2019:309–15. DOI:10.2991/itids-19.2019.55.
16. Sakr S., Zomaya A.Y. Handbook of Big Data Technologies. Springer International Publishing; 2017. Available from: <https://link.springer.com/book/10.1007/978-3-319-49340-4> (accessed on 12.10.2022).
17. Sakr S., Zomaya A.Y. Encyclopedia of big data technologies. Springer International Publishing; 2019. DOI: 10.1007/978-3-319-77525-8.
18. Bakshi K. Technologies for Big Data. *IGI Global*; 2014:1–22. DOI: 10.4018/978-1-4666-4699-5.ch001.
19. Singh S., Singh P., Garg R., Mishra P.K. Big Data: Technologies, Trends and

- Applications. *International Journal of Computer Science and Information Technologies*. 2015;6(5):4633–4639.
20. Mamedova G.A., Zeynalova L.A., Melikova R.T. Big data technologies in e-learning. *Open Education*. 2018;16;0(6):41–8. DOI: 10.21686/1818-4243-2017-6-41-48.
 21. Jiang D., Ooi B.C., Shi L., Wu S. The performance of MapReduce: an in-depth study. *In proceeding Proc VLDB Endow*. 2010;3(1–2):472–483. DOI:10.14778/1920841.1920903.
 22. Shim K. MapReduce Algorithms for Big Data Analysis. In: *Madaan A, Kikuchi S, Bhalla S, editors. Lecture Notes in Computer Science. Databases in Networked Information Systems*. Berlin, Heidelberg: Springer; 2013:44–8. DOI: 10.1007/978-3-642-37134-9.
 23. Tao Y., Lin W., Xiao X. Minimal MapReduce algorithms. *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*. New York, NY, USA: Association for Computing Machinery; 2013:529–40. DOI:10.1145/2463676.2463719.
 24. Györödi C.A., Dumșe-Burescu D.V., Zmaranda D.R., Györödi R.Ş., Gabor G.A., Pecherle G.D. Performance Analysis of NoSQL and Relational Databases with CouchDB and MySQL for Application’s Data Storage. *Applied Sciences*. 2020;10(23):8524. DOI: 10.3390/app10238524.
 25. Anderson J.C., Lehnardt J., Slater N. CouchDB: The Definitive Guide: Time to Relax. *O’Reilly Media, Inc.*; 2010:274.
 26. Lennon J. *Beginning CouchDB*. Apress; 2010:299.
 27. Гусаренко А.С. Свидетельство о государственной регистрации программы для ЭВМ № 2022617552. Микросервис ситуационно-ориентированной базы данных для загрузки данных и функций Map-Reduce в документо-ориентированное хранилище. 2022.
 28. Гусаренко А.С. Свидетельство о государственной регистрации программы для ЭВМ № 2022617505. Модули ситуационно-ориентированной базы данных для извлечения больших документов и архивов из RESTful-сервисов гетерогенных хранилищ данных. 2022.
 29. Apache CouchDB. Доступно по: <https://couchdb.apache.org/> (дата обращения 17.10.2022).
 30. Миронов В.В., Гусаренко А.С., Юсупова Н.И. Ситуационно-ориентированные базы данных: обработка офисных документов. *Моделирование, оптимизация и информационные технологии*. 2022;28;10(2):1–16. DOI: 10.26102/2310-6018/2022.37.2.021.
 31. Миронов В.В., Гусаренко А.С., Юсупова Н.И. Инвариантность виртуальных данных в ситуационно-ориентированной базе данных при отображении на разнородные хранилища. *Вестник компьютерных и информационных технологий*. 2017;(1(151)):29–36. DOI: 10.14489/VKIT.2017.01.PP.029-036.
 32. Курсовой проект «Базы данных». Доступно по: <http://hsm.ugatu.su/artem/dbproj/> (дата обращения 20.10.2022).

ИНФОРМАЦИЯ ОБ АВТОРАХ / INFORMATION ABOUT THE AUTHORS

Гусаренко Артем Сергеевич, кандидат технических наук, доцент кафедры Автоматизированных систем управления, Уфимского государственного авиационного технического университета, Уфа, Российская Федерация.

e-mail: gusarenko.as@ugatu.su

ORCID: [0000-0003-4132-6106](https://orcid.org/0000-0003-4132-6106)

Artem Sergeevich Gusarenko, Candidate of Technical Sciences, Associate Professor at the Department of Automated Control Systems, Ufa State Aviation Technical University, Ufa, Russian Federation.

*Статья поступила в редакцию 21.10.2022; одобрена после рецензирования 03.11.2022;
принята к публикации 18.11.2022.*

*The article was submitted 21.10.2022; approved after reviewing 03.11.2022;
accepted for publication 18.11.2022.*