ISSN 2658-5014 (Print)
2021. Т. 3, № 1 (5). С. 39–49

СИИТ
СИСТЕМНАЯ ИНЖЕНЕРИЯ И ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

ISSN 2686-7044 (Online)
http://siit.ugatu.su

УДК 004.65

MONITORING YOUTUBE VIDEO VIEWS IN THE EDUCATIONAL ENVIRONMENT BASED ON SITUATION-ORIENTED DATABASE AND RESTFUL WEB SERVICES

V. V. MIRONOV[1], A. S. GUSARENKO[2], N. I. YUSUPOVA[3]

[1] mironov@list.ru, [2]gusarenko@ugatu.su, [3]yusupova@ugatu.ac.ru

Уфимский государственный авиационный технический университет

Поступила в редакцию 22 января 2021 г.

Аннотация. The monitoring problem the student's views of educational video posted on YouTube is considered. Monitoring of comments posted by students when viewing is proposed for this problem solution. The functionality of the video viewing monitoring subsystem based on the collection and analysis of student comments as part of the university educational system is discussed. The structure of the relational database for the accumulation of information about video views is considered. An example of an analytical report on video views is given. Situational-oriented database (SODB) is used both to populate the relational data warehouse (ETL process) and to generate analytical reports on video views. The capabilities of SODB in organizing web microservices are demonstrated by the example of managing heterogeneous data that is retrieved from the YouTube API and the educational database, and then placed into a relational database. The concept of virtual documents mapped on heterogeneous data sources used in SODB. The implementation of this concept when mapped on web services such as the YouTube API is explained. A set of RESTful web services developed based on the SODB to solve the reviewed problem. The versatility and simplicity of the hierarchical situational model, both when defining and managing web services, is noted. The practical implementation of the subsystem monitoring views of educational videos on the PHP platform is described. Examples of analytical reports generated by the subsystem for practical use in the educational process are discussed.

Ключевые слова: educational video; monitoring video views; YouTube API; ETL-process; analytical reports; heterogeneous data integration; situational-oriented database; hierarchical situational model; web services; RESTful services; microservices.

ВВЕДЕНИЕ

Video and other multimedia educational content [1] is actively used to improve student learning. Videos posted on the web enable students to master academic disciplines at a convenient pace at a convenient time. The most popular platform for hosting educational content [2] is the YouTube video hosting site, which gives anyone the opportunity to freely (free of charge) post their videos from various areas, as well as view, rate, comment [3], add to favorites and share these or other videos [4]. This opportunity is actively used by teachers in organizing the educational process and students in studying academic disciplines. Independent development of educational material [5] through the Internet raises the problem of monitoring this process by

This work is supported by Russian Foundation for Basic Research (RFBR), grant 19-07-00682

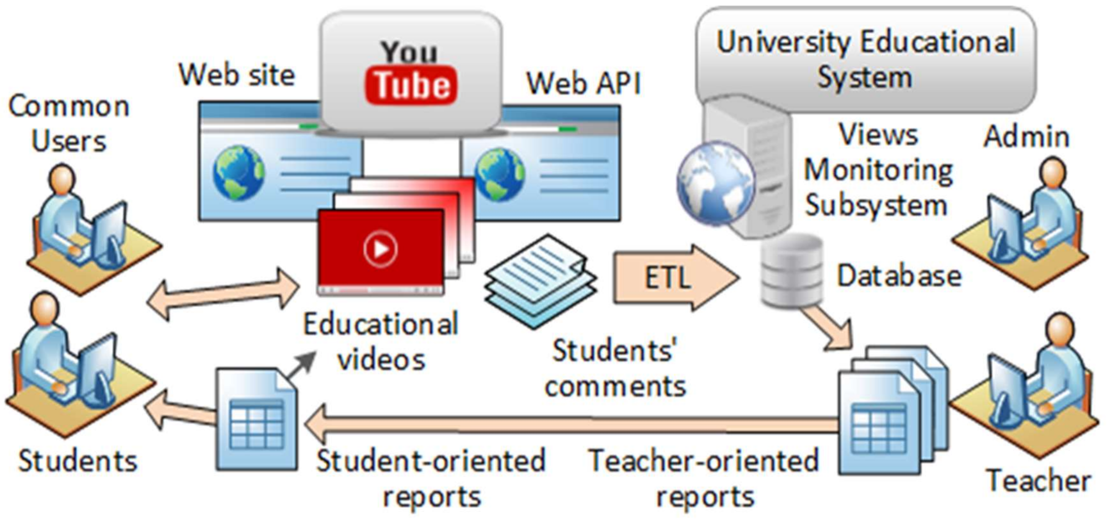


Fig. 1. Monitoring video views in the educational process

teachers. The first step here is to monitor the fact of the student watching a video. The pedagogical experience of the authors shows that in a tense educational process, quite many students try to evade it by performing tasks and final tests without previewing the video [6]. Therefore, it is natural for a teacher's desire to first ensure that the training video content was viewed by a student [7]. Thus, the problem of monitoring by teachers of the views recommended educational videos take place. For obvious reasons YouTube does not provide information about specific views of specific individuals, so the solution to this problem is not trivial.

This article has a twofold purpose:

1. show how you can monitor YouTube video views based on information technology, which provides for the collection of comments to videos posted by students and their accumulation in a database in the university educational environment for the subsequent generation of analytical reports for teachers and students [8];
2. demonstrate the capabilities of situational-oriented databases for information technology support of this process based on service-oriented architecture and RESTful web services [9, 10].

MONITORING VIEWS BASED ON COMMENTS

The idea of monitoring video views is to collect and analyze comments on videos posted by students. To this end a subsystem for monitoring the viewing of educational videos is created within the educational environment of the educational institution. The student is required to pre-create his channel on YouTube, through which comments will be posted. Next, the student must register with the monitoring subsystem and give her the ID of his channel. When viewing a video that is in the list of mandatory the student must leave a comment at the beginning and at the end of the viewing. Comments are uploaded from YouTube to the monitoring subsystem's database where the student's duration of the video is rated.

VIDEO VIEWING MONITORING SUBSYSTEM

The process of monitoring views on the proposed scheme is illustrated in Fig. 1. Educational content in the form of a collection of videos posted on YouTube and available to students for viewing and commenting through the website YouTube. It is also available to regular Internet users in case of posting educational content with open access (which makes it necessary to filter comments). YouTube provides programmatic access to data [11] in the form of an API (Application Program Interface). Using the YouTube API, the viewing monitoring subsystem within the university educational environment can read comments that were left by students on educational videos. This process (ETL – Extract, Transform, Load) provides for the selection of comments that meet certain requirements. In particular, the comments should belong to students who have been previously registered in the monitoring subsystem. Students' comments should be paired, that is, indicate both the beginning and the end of viewing. Thus, the student must at least twice make comments while watching a video. The following trick is proposed to reduce the number of comments posted on YouTube. The fact is used that for each comment YouTube records two points in time: the moment of posting and the moment of last change. Therefore, upon completion of the review, instead of leaving a new comment, the student can simply change the comment that corresponds to the beginning of the review. As a result, each viewing will correspond to one comment on YouTube.

Another feature is related to the permissibility of viewing a single video after several attempts. Videos can be time consuming and it will be difficult for a student to master it in one view. In this case, the possibility of several attempts is provided, with a separate comment for each attempt. By analyzing the attempts, you can estimate the total viewing time of the video.

The results of the ETL process are recorded in the monitoring subsystem database [12] in the form of viewing facts. Each fact corresponds to one attempt to view a video by a certain student. The fact is identified by the three attributes: video identifier; user channel identifier; time identifier [13]. These attributes can be obtained from the YouTube API as part of the comment information. The set of numerical indicators associated with the fact identifier includes the moments when the comment was published and its changes, as well as the duration of viewing as a percentage of the total length of the video. The first two indicators are provided by the YouTube API, and the third indicator is calculated when the fact is entered into the database.

RELATIONAL DATABASE FOR MONITORING VIDEO VIEWS

The relational database [13] is intended for information support for monitoring video views (Fig. 2). It includes 7 tables, rows of which are populated from various data sources [14] (squares mark attributes, which are components of the primary key, unique attributes are marked with a diamond).

1. The videos table contains video information. The data source for this table is the YouTube Video API web service [15]. The videoId attribute is an identifier that YouTube assigns to each posted video. For ease of use, a numeric durSec attribute has been added to the table, which corresponds to the symbolic attribute duration.

2. The playlists table contains information about playlists containing thematic video sets. The playlistId attribute is an identifier that YouTube assigns to each created playlist, and the title attribute is the name of the playlist. Thus, this table is a directory of playlist names. The data source for this table is the YouTube API web service Playlists.

3. The playlist_video table contains information about the videos included in the playlist. This table contains attribute pairs (playlistId,

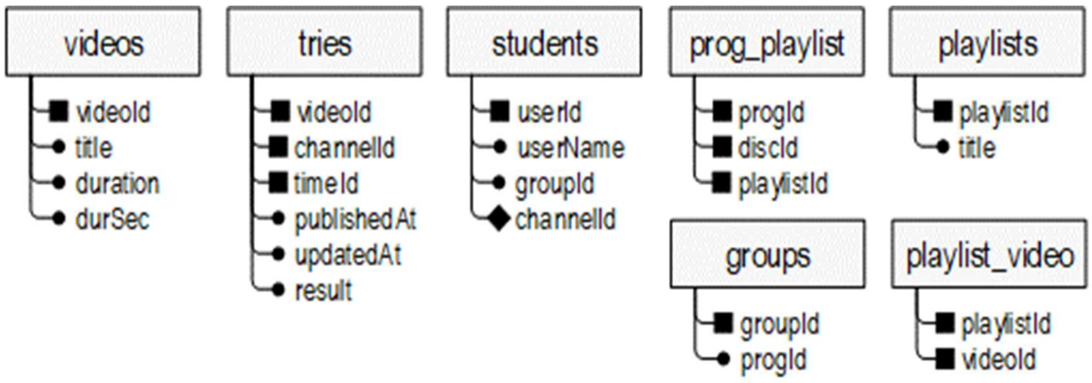


Fig. 2. Relational database for monitoring video views

videoId). The data source for this table is the YouTube API web service [16] Playlist Items.

4. The table tries contains information about attempts to view videos by a student. As already noted, each attempt corresponds to a comment. The data source [17] for this table is the YouTube API Content Threads web service, which provides information about video comments. Table rows are identified by a triple of attributes: videoId, channelId and timeId. The videoId attribute is a video identifier, as in the videos table. The channelId attribute is the channel ID of the user who posted this comment. When filling in the table, only comments belonging to a student registered in the system, that is, a user whose channel ID is in the students table, are considered. The timeId attribute is a timestamp that identifies various attempts by the same student to the same video.

5. The students table contains information about students. The table is updated during the online registration of students at the initial stage of the development of the academic discipline. The userId attribute is a common student identifier. It is used to identify table rows. The groupId attribute is the ID of the student group to which the student is attached. The channelId attribute is an identifier for a student's YouTube channel through which the student will post comments on the video. This attribute must be unique, i.e., there cannot be two students with the same channel ID.

6. The groups table contains information about student groups. The data source for this table is the dean's database. The groupId attribute is a group identifier, and the progId attribute is a group program of study identifier.

7. The prog_playlists table contains information about playlists that students studying in a certain program within a certain academic discipline should review. The data source for this table is a work program that establishes the requirements for mastering the discipline. The attribute progId is the identifier of the program of study, the attribute discId is the identifier of the academic discipline, and the attribute playlistId is the identifier of the playlist.

Thus, database tables are filled/replenished at different times from different data sources [18, 19]. Different combinations of tables are used to generate various reports provided by teachers and students.

For example, to generate a report (with all attributes) about videos that students should view when they master academic disciplines, it is necessary to join tables of students, groups, prog_playlists, playlist_video, videos, playlists. The model of the corresponding virtual table mustView is presented on Fig. 3, a.

Here the symbol "butterfly" means the relational operation of the inner equi-join of tables. The rows of this virtual table are identified by four identifier attributes. (userId, playlistId, discId, videoId).

The data relating to a program of study, to a particular discipline, to a separate student group, etc., are obtained from the mustView virtual table by filtering rows by the values of the corresponding attributes. In Fig. 3, b shows a model illustrating the filtering of a table for a student name = "Миронов В. В.".

To get information for a view report, you must join the mustView virtual table with the tries table. (the virtual table views, Fig. 3, c). Note that the relational operation of the left outer join is used to form the virtual table views. This allows you to reflect the required views that have not yet been completed by the student (in this case, the attributes of the views are assigned null values). The views virtual table contains the lowest granularity information. To obtain aggregated (summary) indicators, you need to expose this table relational grouping operation according to the corresponding criteria. In Fig. 3, d illustrates the acquisition of a composite indicator qty (number of views) at the level of student groups. For this, the virtual table views is grouped by a pair of attributes (progId, groupId). As a result, the set of source lines of the views table is divided into subsets corresponding to one student group. The aggregate function COUNT (*) (counting the number of lines in a group) is applied to each subset, which results in the total number of view attempts for each student group.

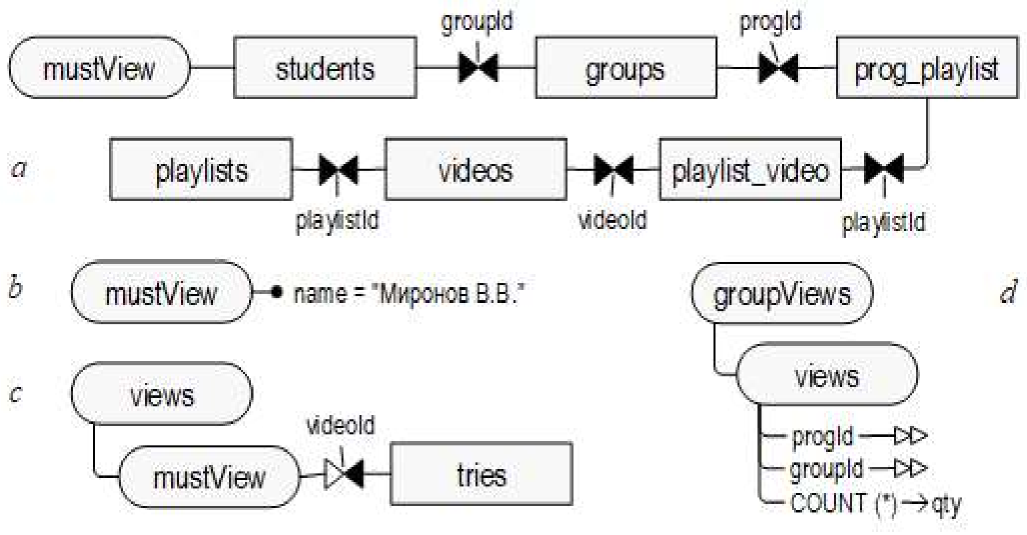


Fig. 3. Connection of tables when generating reports for monitoring video views: *a – information about the videos that students should view when mastering academic disciplines; b – filter strings by attribute value; c – information about the mandatory videos; d – getting aggregates by grouping a table*

ANALYTICAL REPORTS ON VIDEO VIEWS

Analytical reports on the views are based on the facts stored in the database, as well as other background information collected [20] during the registration of students in the educational system. Reports [21] are intended for both teachers and students. The reports contain both indicators of facts, and derivatives of indicators calculated based on indicators of facts (attempts, credits, time credits, time remaining, etc.). Indicators are formed with varying degrees of integration (video, playlist, discipline, etc.). An example of such a report is illustrated in Fig. 4. This report is a summary table and is aimed at students, that is, it contains information about video views that are required for students when they master an academic discipline. Videos are grouped into themed playlists. For playlists, relevant summary indicators are provided. The table also shows the overall results for this student in this discipline. The indicators presented in the report reflect the read (completed) and remaining (indebted) views in quantitative, temporal and percentage terms with varying degrees of aggregation. So, from the report it is clear

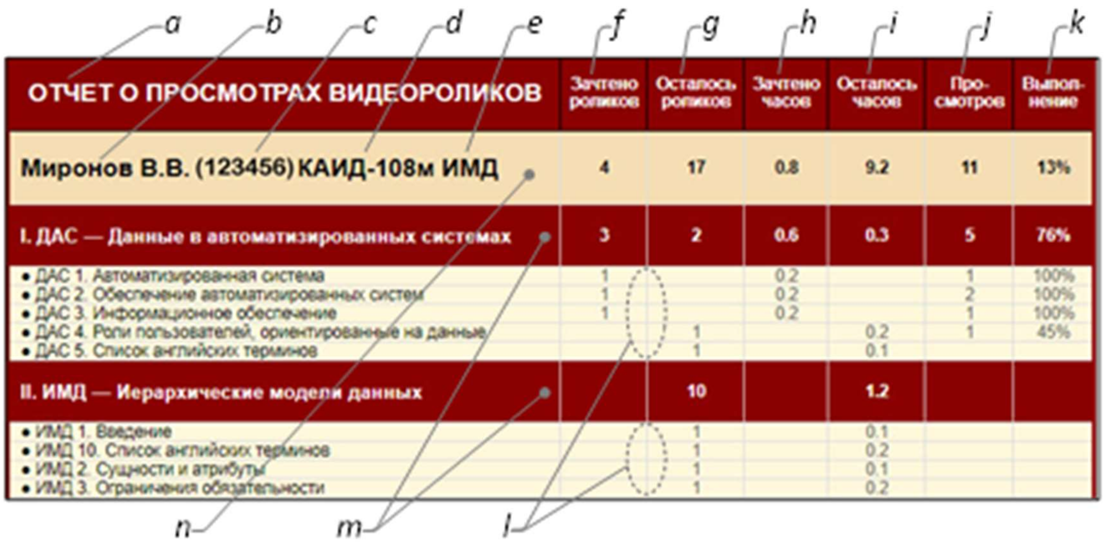
that, in general, the student "Миронов В.В." viewed 4 videos of the total duration of 0.8

hours on the "ИМД" discord, and it remained to view 17 videos with a total duration of 9 for this discipline 2 hours, i.e., 13% of the required volume. These totals are detailed in a report at the level of playlists and at the level of individual videos. So, the report shows that the playlist "I. ДАС" viewed by 76%.

At the same time, videos 1–4 are viewed completely, and video 5 is only 45%; video 2 was viewed in two attempts, the rest were viewed in one attempt.

The interactive features of the report are that the title of the video is simultaneously a hyperlink to this video on the web.

By clicking on the title, the student goes to the appropriate page on YouTube to watch the video. Teacher-oriented reports additionally contain levels of greater integration: the level of the student group, the level of the field of study, the level of the academic discipline, etc.


a
b
c
d
e
f
g
h
i
j
k
n
m
l

ОТЧЕТ О ПРОСМОТРАХ ВИДЕОРОЛИКОВ	Зачтено роликов	Осталось роликов	Зачтено часов	Осталось часов	Про-смотров	Выпол-нение
Миронов В.В. (123456) КАИД-108м ИМД	4	17	0.8	9.2	11	13%
I. ДАС — Данные в автоматизированных системах	3	2	0.6	0.3	5	76%
• ДАС 1. Автоматизированная система	1		0.2		1	100%
• ДАС 2. Обеспечение автоматизированных систем	1		0.2		2	100%
• ДАС 3. Информационное обеспечение	1		0.2		1	100%
• ДАС 4. Роли пользователей, ориентированные на данные		1		0.2	1	45%
• ДАС 5. Список английских терминов		1		0.1		
II. ИМД — Иерархические модели данных		10		1.2		
• ИМД 1. Введение		1		0.1		
• ИМД 10. Список английских терминов		1		0.2		
• ИМД 2. Сущности и атрибуты		1		0.1		
• ИМД 3. Ограничения обязательности		1		0.2		

Fig. 4. Fragment of the report on student-oriented video views (in Russian): *a – report header; b – student name; c – student code; d – student group code; e – discipline code; f – the amount credited video; g – number of remaining videos; h – credited*
duration, hours; i – remaining duration, hours; j – number of attempts; k – percent complete by time; l – video lines; m – playlist strings; n – student total for discipline

SITUATION-ORIENTED DATABASE SOLUTIONS

The above concepts can be considered as a technical specification (development specification) for the creation of a subsystem for monitoring views. Based on them, it is possible to develop a monitoring [22] subsystem using well-known traditional methods and approaches of web designing. Our goal, as already noted above, is to demonstrate the capabilities and advantages of solving this problem based on situation-oriented databases [8]. Namely, to show that in this way the task is solved faster and easier.

SITUATIONALLY-ORIENTED DATABASES

A Situation-Oriented Database (SODB) [9] is an integrator of heterogeneous data driven by an integrated situational model. The key features of SODB that are relevant to the viewing monitoring task in question are:

– built-in situational model that sets possible states of the monitoring process and possible transitions between states;

– virtual documents associated with the states of the situational model, which are displayed on heterogeneous sources of real data;

– an interpreter that processes virtual documents during the processing of a situational model. SODB [11] architecture in general is shown in Fig. 5.

The situational model is a hierarchy of submodels, each of which, in turn, is a finite state model. The submodel consists of elements: possible states and possible transitions between states. The hierarchy of submodels is formed due to the nesting of submodels. Each submodel (except for one root submodel) is a child of some state of its parent submodel. Thus, the non-root submodel sets the sub-states for its parent state.

Virtual documents are HSM state elements that define the mapping to external data. Thus, virtual documents allow for uniform processing of heterogeneous (heterogeneous) real data located in physical data stores [23]. SODB can currently map virtual documents to the following types of real data [24]:

– to local or remote files in XML or JSON format;

– to web services on the Internet;

– to local ZIP archives;

– to relational databases.

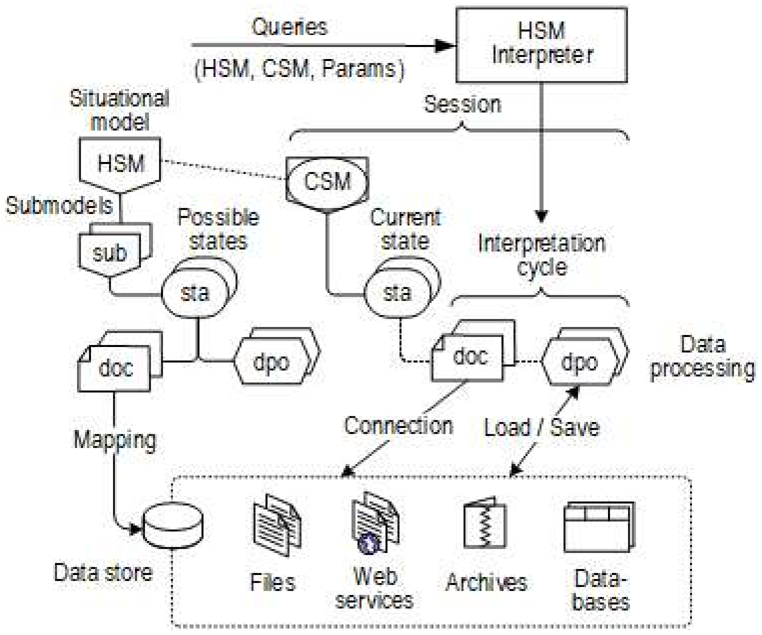


Fig. 5. SODB Architecture: *HSM – hierarchical situational model; CSM – current state model; sub –submodel element; sta – state element; doc –virtual document element; dpo – document processing object element*

Situation Model Interpreter (HSMI) is a document-processing program that complies with the built-in situational model. The interpreter cyclically or by request bypasses the hierarchy of submodels, performing two main functions:

– monitors the current state of the situational model;

– processes virtual documents associated with current states.

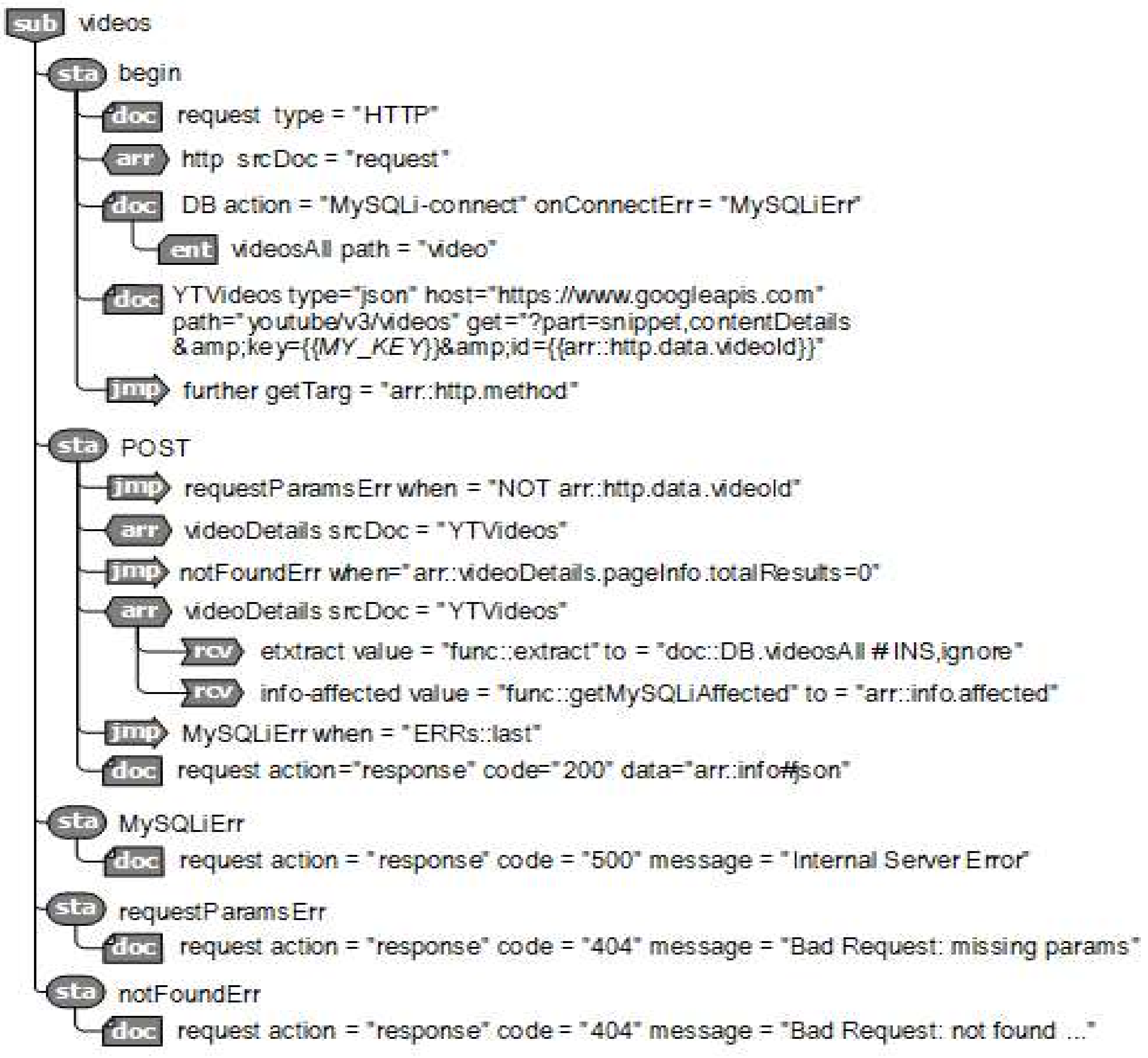
Having processed the situational model on some interpretation cycle, the interpreter stores information about the current states of submodels in external memory in the form of a current state model (CSM). The interpreter uses this information to continue processing correctly in the next interpretation cycle.

The processing of virtual documents is specified in the situational model using DPO-elements (data processing objects). During the interpretation, the interpreter creates the corresponding processing objects based on DPO-elements. Real documents are loaded into processing objects based on the corresponding virtual documents [13]. The processing results of downloaded documents are saved in an external environment.

SITUATIONAL MODEL OF MONITORING VIEWS

A fragment of one of the HSM submodels in graphical representation is shown in Fig. 6. The sub:videos submodel acts as the RESTful service [25] videos, which receives video information from YouTube and uploads the required data to the videos table of the viewing monitoring subsystem database. A submodel contains several states corresponding to various functioning situations:

– sta: begin – this is the initial processing state, in which the virtual documents used by the service are declared, and the situation is selected to continue processing;



```
sub videos
  sta begin
    doc request type = "HTTP"
    arr http srcDoc = "request"
    doc DB action = "MySQLi-connect" onConnectErr = "MySQLiErr"
      ent videosAll path = "video"
    doc YTVideos type="json" host="https://www.googleapis.com"
        path="youtube/v3/videos" get="?part=snippet,contentDetails
        &amp;key={{MY_KEY}}&amp;id={{arr::http.data.videoId}}"
    jmp further getTarg = "arr::http.method"
  sta POST
    jmp requestParamsErr when = "NOT arr::http.data.videoId"
    arr videoDetails srcDoc = "YTVideos"
    jmp notFoundErr when="arr::videoDetails.pageInfo.totalResults=0"
    arr videoDetails srcDoc = "YTVideos"
      rcv etxtract value = "func::extract" to = "doc::DB.videosAll#INS,ignore"
      rcv info-affected value = "func::getMySQLiAffected" to = "arr::info.affected"
    jmp MySQLiErr when = "ERRs::last"
    doc request action="response" code="200" data="arr::info#json"
  sta MySQLiErr
    doc request action = "response" code = "500" message = "Internal Server Error"
  sta requestParamsErr
    doc request action = "response" code = "404" message = "Bad Request: missing params"
  sta notFoundErr
    doc request action = "response" code = "404" message = "Bad Request: not found ..."
```

Fig. 6. Fragment of a situational model of microservice

– sta: POST, sta:GET, sta:PUT, и sta:DELETE – these are states that specify the continuation of processing in accordance with the HTTP Request Method (only the first of these states is shown in Fig. 6 so as not to clutter up the model);

– sta: MySQLiErr, sta:requestParamsErr, и sta:notFoundErr — these are conditions for completing processing in exceptional situations when a particular error is detected.

Virtual documents used by the service [26] determine the mapping to external data:

– doc: request – this is a mapping to the HTTP-request that this service requested. The request method and the parameters passed with the request become available in JSON format using this virtual document loaded into the dpo-object arr:http;

– doc: YTVideos – this is a mapping to the YouTube API Videos service, which provides JSON information about a specific video by a given identifier;

– doc: DB – this is a mapping to the relational database MySQL subsystem for monitoring views. The display establishes a connection to the database (in the event of an error, a transition to the state sta: MySQLiErr). Entry-element ent: videosAll points to the video table in the connected database;

The choice of further processing is determined by the jmp: further transition element,

which provides a transition to the state corresponding to the HTTP-request method. So, if the method is POST, then the transition to the state sta: POST. Processing in fig. 6 disclosed only for the case of the POST method.

This method is commonly used in RESTful services [24] to perform web resource update operations. The table videos is updated by this method in the microservice in question. To do this, an associative array arr:videoDetails is created and the virtual JSON document doc: YTVideos is loaded into it. Data of interest is extracted from the loaded array using the rcv: extract element, formatted and transferred to the doc: DB.videosAll virtual document for uploading to the database videos table. Loading into the table is performed by the INSERT IGNORE SQL statement, which ensures that only information about new videos is inserted using the POST method.

RESTFUL SERVICES FOR MONITORING VIEWS

The implementation of the working version of the viewing monitoring subsystem is based on a service-oriented architecture [22], namely, based on RESTful microservices [26]. This architecture provides high independence and modifiability of the components of the subsystem, which is important in the conditions of heterogeneity and variability of the used data sources. As part of the implementation of this approach, a set of microservices in a situation-oriented database environment has been developed. Microservices [23] of the lower level are focused on servicing individual tables in the database of the viewing monitoring subsystem. These microservices allow to add, retrieve, update, and delete data:

– The videos microservice processes the videos table, including the information received from the YouTube API Video web service;

– The playlists microservice processes the playlists table based on information received from the YouTube Playlists API web service;

– The playlist_video microservice processes the playlist_video table based on data from the YouTube service API Playlist Items;

– The tries microservice processes the table tries, including recording information about comments from the YouTube API Web service Comment Threads;

– The students microservice processes the students table, including entering information about students during the online registration of students at the initial stage of mastering the discipline;

– The groups microservice processes the groups table based on the dean's database;

– The prog_playlist microservice processes the prog_playlist table containing information about playlists that students should view.

Top-level microservices process data from several tables using lower level microservices [24]. So, the reports microservice generates data for analytical [15, 21] reports for both teachers and students. An example of one of these reports was considered above (see Fig. 4).

CONCLUSION

Thus, an analysis of student comments is proposed to solve the problem of monitoring student views of educational videos posted on YouTube. For this, a viewing monitoring subsystem is provided as part of the university educational system. Student comments are retrieved using the YouTube API and entered into the relational database of the monitoring subsystem. Based on this and other information, analytical reports on views are generated for both teachers and students. The monitoring subsystem is implemented based on situation-oriented databases, which are used both to populate the database and to generate analytical reports on video views. The monitoring subsystem has a service-oriented architecture based on the principles of RESTful organization and microservices. The universality and simplicity of the hierarchical situational model is achieved both when defining web services and when managing them, when using situation-oriented databases to solve this problem.

ACKNOWLEDGMENTS

The paper was performed as part of a situational-oriented database development project supported by the Russian Foundation for Basic Research (RFBR), grant 19-07-00682.

СПИСОК ЛИТЕРАТУРЫ

1. J.-B. Alayrac, et al., Learning from Narrated Instruction Videos // *Proc. IEEE Transactions on Pattern Analysis and Machine Intelligence.* 2018. Vol. 40. Pp. 2194–2208. DOI: 10.1109/TPAMI.2017.2749223.

2. **Zhou L., Xu C., Corso J.J.** Towards automatic learning of procedures from web instructional videos // *Proc. 32nd AAAI Conference on Artificial Intelligence (AAAI'2018).* 2018. Pp. 7590–7598.

3. E. Poche, et al., Analyzing User Comments on YouTube Coding Tutorial Videos // *Proc. IEEE International Conference on Program Comprehension*. 2017. Pp. 196-206. DOI: 10.1109/ICPC.2017.26.

4. **H. Silva, I. Azevedo** Instructional videos and others on Youtube: Similarities and differences in comments // *Proc. of the 9th International Conference on Computer Supported Education (CSEDU'2017).* 2017. Pp. 418-425.

5. C. S. Lee, et al., Making sense of comments on YouTube educational videos: A self-directed learning perspective // *Online Information Review.* no. 5 (41). 2017. Pp. 611–625. DOI: 10.1108/OIR-09-2016-0274.

6. **G. Olguin, T. Varella, A. C. Seabra** Accessibility and social issues in e-learning for engineering students // *Proc. Lecture Notes in Engineering and Computer Science.* 2016, Vol. 2225. pp. 243-247.

7. **C.W.-Y. Chen,** Analyzing online comments: a language-awareness approach to cultivating digital literacies // *Computer Assisted Language Learning*, vol. 33, no. 4. 2019. Pp. 435-454. DOI: 10.1080/09588221.2019.1569068

8. **Миронов В. В., Гусаренко А. С., Юсупова Н. И.** Встраивание отображений виртуальных мультидокументов на реальные источники данных в ситуационно-ориентированных базах // Прикладная информатика. 2018. Т. 13. № 3 (75). С. 47–60. [**V. V. Mironov, A. S. Gusarenko, N. I. Yusupova,** "Integration of Virtual Multidocument Mappings into Real Data Sources in Situational-Oriented Databases", (in Russian), in *Applied Informatics.* vol. 13, no. 3(75), 2018. Pp. 47–60.]

9. **V. V. Mironov, A. S. Gusarenko, N. I. Yusupova,** "Situation-oriented databases: document management on the base of embedded dynamic model", In *Proc. CEUR Workshop Proceedings (CEUR-WS.org): Selected Papers of the XI International Scientific-Practical Conference Modern Information Technologies and IT-Education (SITITO'2016)*, vol. 1761, 2016. pp. 238–247.

10. **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"*, vol. 3, no. 5. 2018. Pp. 240–244.

11. **Миронов В. В., Гусаренко А. С., Юсупова Н. И.** Ситуационно-ориентированные базы данных: polyglot persistence на основе REST-микросервисов // Прикладная информатика. 2019. Т. 14. № 5(83). С. 87–97. [**V. V. Mironov, A. S. Gusarenko, N. I. Yusupova,** "Situation-oriented databases: polyglot persistence based on REST microservices" // *Applied Informatics*, (in Russian), vol. 14, no. 5(83), 2018. Pp. 87–97. DOI: 10.24411/1993-8314-2019-10038]

12. **Гусаренко А. С.** Усовершенствование модели ситуационно-ориентированной базы данных для взаимодействия с MySQL // Известия высших учебных заведений. Приборостроение. 2016. Т. 59. № 5. С. 355–363. [**A. S. Gusarenko**, Improvement of Situation-Oriented Database Model for Interaction with MySQL // *Journal of Instrument Engineering*, (in Russian), vol. 59, no. 5, 2016. Pp. 355–363. DOI: 10.17586/0021-3454-2016-59-5-355-363]

13. **Миронов В. В., Гусаренко А. С., Юсупова Н. И.** Структурирование виртуальных мультидокументов в ситуационно-ориентированных базах данных с помощью entry-элементов // Труды СПИИРАН. 2017. № 53. С. 225–243. [**V. V. Mironov, A. S. Gusarenko, N. I. Yusupova** Structuring virtual multi-documents in situationally-oriented databases by means of entry-elements // *SPIIRAS Proceedings*, (in Russian), Vol. 4, no. 53, 2017. pp. 225–242. DOI: 10.15622/sp.53.11]

14. **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). С. 29–36. [**V. V. Mironov, A. S. Gusarenko, N. I. Yusupova** // The Invariance of The Virtual Data in The Situationally Oriented Database When Displayed on Heterogeneous Data Storages, (in Russian), *Herald of Computer and Information Technologies,* no. 1(151). 2017. Pp. 29–36.]

15. H. Aragon, et al., Workload characterization of a software-as-a-service web application implemented with a microservices architecture // *Proc. The Web Conference 2019 – Companion of the World Wide Web Conference (WWW'2019)*, 2019. pp. 746-750. DOI: 10.1145/3308560.3316466

16. **Huf A., Siqueira F.** Composition of heterogeneous web services: A systematic review // *Journal of Network and Computer Applications,* Vol. 143, 2019. Pp. 89–110. DOI: 10.1016/j.jnca.2019.06.008

17. **Villaça L. H. N., Azevedo L.G., Baio F.** Query strategies on polyglot persistence in microservices in *Proc. of the ACM Symposium on Applied Computing*, 2018. Pp. 1725-1732. DOI: 10.1145/3167132.3167316.

18. **Tserpes K.** stream-MSA: A microservices' methodology for the creation of short, fast-paced, stream processing pipelines // *ICT Express,* vol. 5, no. 2, 2019. pp. 146–149. DOI: 10.1016/j.icte.2019.04.001

19. **Radchenko G., Alaasam A., Tchernykh A.** Microworkflows: Kafka and Kepler fusion to support digital twins of industrial processes // *Proc. of 11th IEEE/ACM International Conference on Utility and Cloud Computing Companion, UCC Companion 2018,* 2019. Pp. 83-88. DOI: 10.1109/UCC-Companion.2018.00039

20. **Katsifodimos A., Fragkoulis M.** Operational stream processing: Towards scalable and consistent event-driven applications // *Proc. of Advances in Database Technology – EDBT,* 2019. Pp. 682-685. DOI: 10.5441/ 002/edbt.2019.86

21. **Hoque S., Miranskyy A.** Architecture for analysis of streaming data", in *Proc. of IEEE International Conference on Cloud Engineering (IC2E'2018),* 2018. Pp. 263-269. DOI: 10.1109/IC2E.2018.00053

22. Borodulin K., et al. Towards digital twins cloud platform: Microservices and computational workflows to rule a smart factory // *Proc. of the10th International Conference on Utility and Cloud Computing (UCC'2017),* 2017. Pp. 205-206. DOI: org/10.1145/3147213.3149234

23. **Marquez G., Astudillo H.** Actual Use of Architectural Patterns in Microservices-Based Open Source Projects // *Proc. of Asia-Pacific Software Engineering Conference (APSEC' 2019),* Pp. 31-40. DOI: 10.1109/APSEC.2018.00017

24. **Taibi D., Lenarduzzi V., Pahl C.** Architectural patterns for microservices: A systematic mapping study // *Proc. of the*

8th International Conference on Cloud Computing and Services Science (CLOSER'2018), 2018. Pp. 221-232.

25. **Quenum J. G., Aknine S.** Towards executable specifications for microservices // Proc. *IEEE International Conference on Services Computing, (SCC'2018) – Part of the 2018 IEEE World Congress on Services,* 2018. Pp. 41-48. DOI: 10.1109/SCC.2018.00013

26. **Hamzehloui M. S., Sahibuddin S., Salah K.** A systematic mapping study on microservices // Advances in Intelligent Systems and Computing, vol. 843, 2019. Pp. 1079–1090. DOI: 10.1007/978-3-319-99007-1_100

ABOUT AUTHORS

MIRONOV, Valeriy Viktorovich, prof. dep. Control Automation System. Dipl. radiophysicist (Voronezh state university, 1975). Dr. Tech. sciences on control in Tech. systems (USATU, 1995). Research in Hierarchical models and situation control.

GUSARENKO, Artem Sergeevich, Docent at the dep. Control Automation System. Dipl. Informatic-economist (USATU, 2010). Phd. In Computer Science on Mathematical and software of computers, complexes, and computer networks (USATU, 2013). Research in Hierarchical models and situation control.

YUSUPOVA, Nafisa Islamovna, prof. dep. computational mathematics and cybernetics. Dr. Tech. sciences. Research in Hierarchical models and situation control, Artificial Intelligence.

МЕТАДАННЫЕ

Название: Мониторинг просмотров видео YouTube в образовательной среде на основе ситуационно-ориентированной базы данных и веб-сервисов RESTful.

Авторы: V. V. Mironov[1], A. S. Gusarenko[2], N. I. Yusupova[3]

Принадлежность: Уфимский государственный авиационный технический университет, Россия.

Почта: [1]mironov@list.ru, [2]gusarenko@ugatu.su, [3]yusupova@ugatu.ac.ru

Язык: Английский.

Источник: СИИТ (научный журнал Уфимского государственного авиационного технического университета). Т. 3, № 1 (5), стр. 39-49, 2021, ISSN 2686-7044 (онлайн), ISSN 2658-5014 (печатный вариант).

Аннотация. Рассмотрена проблема мониторинга просмотров обучающимися обучающими видеороликами, размещенными на YouTube. Для решения этой проблемы предлагается мониторинг комментариев, оставляемых студентами при просмотре. Обсуждается функционал подсистемы видеонаблюдения на основе сбора и анализа отзывов студентов в рамках образовательной системы вуза. Рассмотрена структура реляционной базы данных для накопления информации о просмотрах видео. Приведен пример аналитического отчета по видеопросмотрам. Ситуационно-ориентированная база данных (SODB) используется как для заполнения реляционного хранилища данных (процесс ETL), так и для создания аналитических отчетов по просмотрам видео. Возможности SODB в организации веб-микросервисов демонстрируются на примере управления разнородными данными, которые извлекаются из API YouTube и образовательной базы данных, а затем помещаются в реляционную базу данных. Концепция виртуальных документов, отображаемых на разнородные источники данных, используемые в SODB. Объясняется реализация этой концепции при отображении на веб-сервисах, таких как YouTube API. Набор веб-сервисов RESTful, разработанный на основе SODB для решения рассматриваемой проблемы. Отмечается универсальность и простота иерархической ситуационной модели как при определении веб-сервисов, так и при управлении ими. Описана практическая реализация подсистемы мониторинга просмотров учебных видеороликов на платформе PHP. Обсуждаются примеры аналитических отчетов, формируемых подсистемой для практического использования в учебном процессе.

Ключевые слова: обучающее видео; мониторинг просмотров видео; YouTube API; ETL-процесс; аналитические отчеты; интеграция разнородных данных; ситуационно-ориентированная база данных; иерархическая ситуационная модель; веб-сервисы; RESTful сервисы; микросервисы.

ОБ АВТОРАХ

МИРОНОВ Валерий Викторович, проф. деп. Система автоматизации управления. Дипл. радиофизик (Воронежский государственный университет, 1975). Д-р техн. наук об управлении в Тех. системы (УГАТУ, 1995). Иссл. в области иерархических моделей и управления ситуацией.

ГУСАРЕНКО Артем Сергеевич, доц. зам. Система автоматизации управления. Дипл. информатик-экономист (УГАТУ, 2010). Канд. наук. В области компьютерных наук по математике и программному обеспечению компьютеров, комплексов и компьютерных сетей (УГАТУ, 2013). Исследование иерархических моделей и управления ситуацией.

ЮСУПОВА Нафиса Исламовна, проф. деп. вычислительная математика и кибернетика. Д-р техн. наук. Иссл. в обл. иерархических моделей и управления ситуацией, искусственный интеллект.