Моделирование, оптимизация и информационные технологии.
Научный журнал, Том 7, № 3 http://moit.vivt.ru/

2019

УДК 004.65 DOI: 10.26102/2310-6018/2019.26.3.031

В.В. Миронов, А.С. Гусаренко, Н.И. Юсупова

ПРИМЕНЕНИЕ ВЕБ-СЕРВИСОВ НА ОСНОВЕ СИТУАЦИОННО-ОРИЕНТИРОВАННОЙ БАЗЫ ДАННЫХ ДЛЯ МОНИТОРИНГА ПРОСМОТРА УЧЕБНОГО ВИДЕОКОНТЕНТА

ФГБОУ ВО «Уфимский государственный авиационный технический университет», Уфа, Россия

В данной статье рассматривается задача мониторинга просмотров студентами образовательных видеороликов, размещенных на видеохостинге YouTube. Предложено решение этой задачи на основе контроля и анализа комментариев, размещенных студентами при просмотре. Обсуждаются вопросы организации этого процесса, а также функциональность подсистемы мониторинга просмотров видео, обеспечивающей сбор и анализ студенческих комментариев, в составе университетской образовательной системы. Рассматривается структура реляционной базы данных для накопления сведений о просмотрах видео. Приводятся примеры аналитических отчетов о просмотрах видео, ориентированных на студентов и на преподавателей. Для заполнения реляционного хранилища данных (процесс ETL), а также для формирования аналитических отчетов о просмотрах видео применяется ситуационно-ориентированная база данных (СОБД). Демонстрируются возможности СОБД по организации микро-сервисов на примере управления разнородными данными, которые извлекаются из YouTube API и базы данных хранилища образовательной системы, а затем помещаются в реляционную базу данных на основе концепции виртуальных документов, отображаемых на разнородные источники данных. Поясняется реализация этой концепции при отображении виртуальных документов на веб-сервисы, такие как YouTube API. Обозревается совокупность REST -сервисов, разработанных на основе СОБД для решения задачи мониторинга просмотров. Отмечаются универсальность и простота иерархической ситуационной модели СОБД как при задании веб-сервисов, так и при управлении ими. Описывается практическая реализация подсистемы мониторинга просмотров образовательных видео на платформе PHP.

Ключевые слова: образовательные видео, мониторинг просмотров, YouTube API, ETL-процесс, аналитические отчеты, интеграция разнородных данных, ситуационно-ориентированные базы данных, REST-сервисы, микросервисы.

ВВЕДЕНИЕ

Масштабные задачи по созданию «цифровых университетов», поставленные в рамках Национального проекта «Образование», делают актуальными исследования и разработки в области управления и реализации образовательного процесса с использованием современных обучающих технологий. Видео и другой мультимедийный образовательный контент активно применяется для повышения эффективности обучения

студентов. Видеоролики, размещенные в веб, дают возможность студентам осваивать учебные дисциплины в удобное время в удобном темпе. Популярной платформой для размещения учебного контента является видеохостинговый сайт YouTube, предоставляющий любому желающему возможность свободно разместить свои видеозаписи из различных предметных областей, а также просматривать, оценивать, комментировать, добавлять в избранное и делиться теми или иными видеозаписями. Такую возможность активно используют преподаватели при организации учебного процесса и студенты при изучении учебных дисциплин [1–6].

Самостоятельное освоение учебного материала через интернет порождает задачу мониторинга этого процесса со стороны преподавателей. Первым шагом здесь является контроль самого факта просмотра студентом того или иного видео. Педагогический опыт авторов свидетельствует, что в условиях напряженного учебного процесса достаточно большое число студентов пытаются уклониться от этого, выполняя задания и итоговые тесты без предварительного просмотра видео. Поэтому естественно желание педагога прежде всего убедиться, что учебный видеоконтент был просмотрен студентом. Таким образом, возникает задача мониторинга со стороны преподавателей просмотров студентами рекомендованных образовательных видео. По понятным причинам YouTube не сообщает сведения о конкретных просмотрах конкретными лицами, поэтому решение этой задачи не является тривиальным.

Данная статья преследует двоякую цель:

– показать, как можно организовать мониторинг просмотров видео в YouTube на основе информационной технологии, которая предусматривает сбор комментариев к видео, размещенных студентами, и накопление их в базе данных в университетской образовательной среде для последующего формирования аналитических отчетов для преподавателей и студентов;

– продемонстрировать возможности ситуационно-ориентированных баз данных для информационно-технологической поддержки этого процесса на основе сервис-ориентированной архитектуры и REST-сервисов [7–14].

1. ЗАДАЧА МОНИТОРИНГА ПРОСМОТРОВ ВИДЕО

Идея решения задачи мониторинга просмотров учебных видеороликов состоит в сборе и анализе комментариев к видео [1–6], размещенных студентами. Для этого в составе образовательной среды учебного заведения создается подсистема мониторинга просмотров учебных видео. Студент обязан предварительно создать свой канал на

YouTube, через который будут размещаться комментарии. Далее, студент должен зарегистрироваться в подсистеме мониторинга и сообщить ей идентификатор своего канала. При просмотре видео, находящегося в списке обязательных, студент обязан оставить комментарий в начале и по окончании просмотра. Комментарии загружаются из YouTube в базу данных подсистемы мониторинга, где оцениваются сам факт и длительность просмотра студентом данного видеоролика.

1.1. Подсистема мониторинга просмотров видео

Процесс мониторинга просмотров по предлагаемой схеме иллюстрируется на рисунке 1.

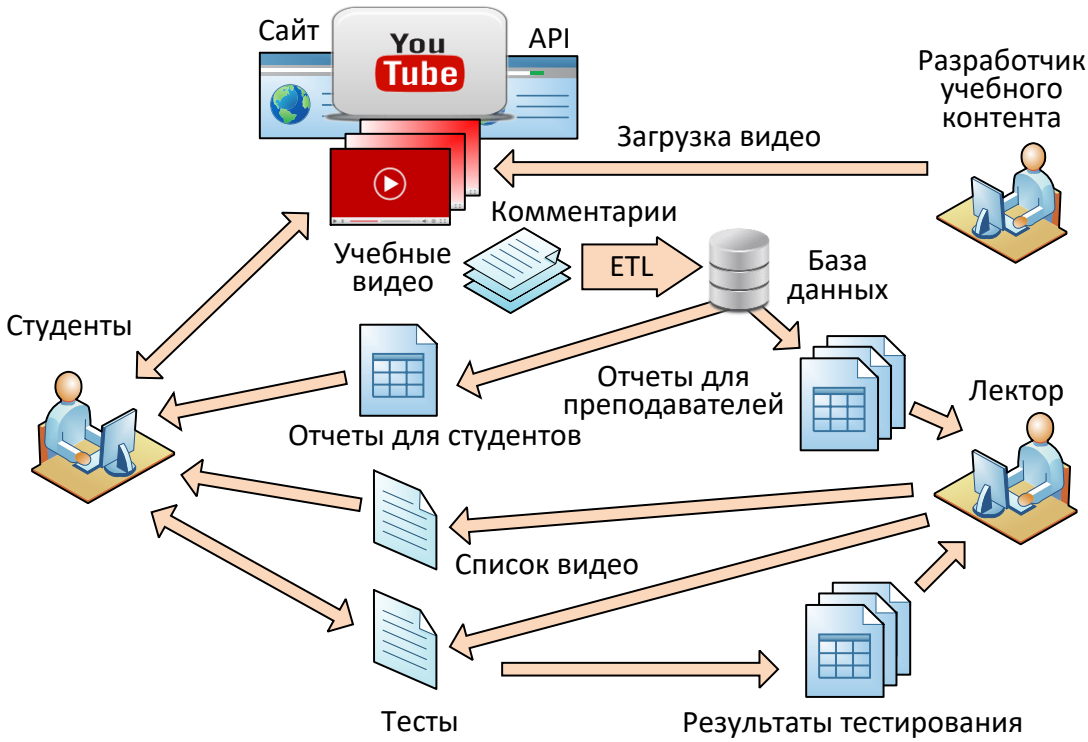


Рисунок 1 — Мониторинг просмотров видео в образовательном процессе

Образовательный контент в виде совокупности видеороликов размещен на YouTube и доступен студентам для просмотра и комментирования через сайт YouTube. (Он может быть доступен и обычным пользователям интернета в случае размещения образовательного контента с открытым доступом, что порождает необходимость фильтрации комментариев). YouTube предоставляет возможность программного доступа к данным в виде API (Application Program Interface). Используя

YouTube API, подсистема мониторинга просмотров в составе университетской образовательной среды может считывать комментарии, которые были оставлены студентами к образовательным видео. Этот процесс (ETL — Extract, Transform, Load) предусматривает отбор комментариев, удовлетворяющих определенным требованиям. В частности, комментарии должны принадлежать студентам, которые были предварительно зарегистрированы в подсистеме мониторинга.

Комментарии студентов должны быть парными, т. е. указывать как начало, так и окончание просмотра. Таким образом, студент должен по крайней мере дважды вносить комментарии при просмотре видео. Для сокращения числа комментариев, размещенных на YouTube, предложена следующее: поскольку для каждого комментария YouTube фиксирует два момента времени: момент размещения и момент последнего изменения, то по завершении просмотра вместо того, чтобы оставлять новый комментарий, студент может просто изменить комментарий, который соответствует началу просмотра. В результате каждой попытке просмотра будет соответствовать один комментарий на YouTube.

Еще одна особенность связана с допустимостью просмотра одного видеоролика за несколько попыток. Видео могут быть продолжительными по времени и студенту будет трудно освоить его за один просмотр. На этот случай предусмотрена возможность нескольких попыток, с отдельным комментарием для каждой попытки. Анализируя совокупность попыток, можно оценить общее время просмотра видеоролика.

Результаты ETL-процесса заносятся в базу данных подсистемы мониторинга в виде фактов просмотра для последующего анализа. Каждый факт соответствует одной попытке просмотра определенным студентом определенного видео и идентифицируется тройкой атрибутов: идентификатор видео; идентификатор канала пользователя; идентификатор момента времени. Эти атрибуты можно получить из YouTube API в составе информации о комментарии. Набор числовых показателей, ассоциированных с идентификатором факта, включает моменты публикации комментария и его последнего изменения, а также длительность просмотра в процентах от общей длительности видеоролика. Первые два показателя предоставляются YouTube API, а третий показатель вычисляется при занесении факта в базу данных.

На рисунке 1 внизу показан также контур контроля усвоения учебного материала, содержащегося в видеороликах, на основе тестирования или

какого-либо иного оценочного контроля. В данной статье эти вопросы не рассматриваются.

1.2. База данных для мониторинга просмотров видео

Реляционная база данных предназначена для информационной поддержки мониторинга просмотров видео (рисунок 2). Она включает 7 таблиц, строки которых заполняются из различных источников данных (квадратами помечены атрибуты, являющиеся компоненты первичного ключа, ромбиком помечены уникальные атрибуты).

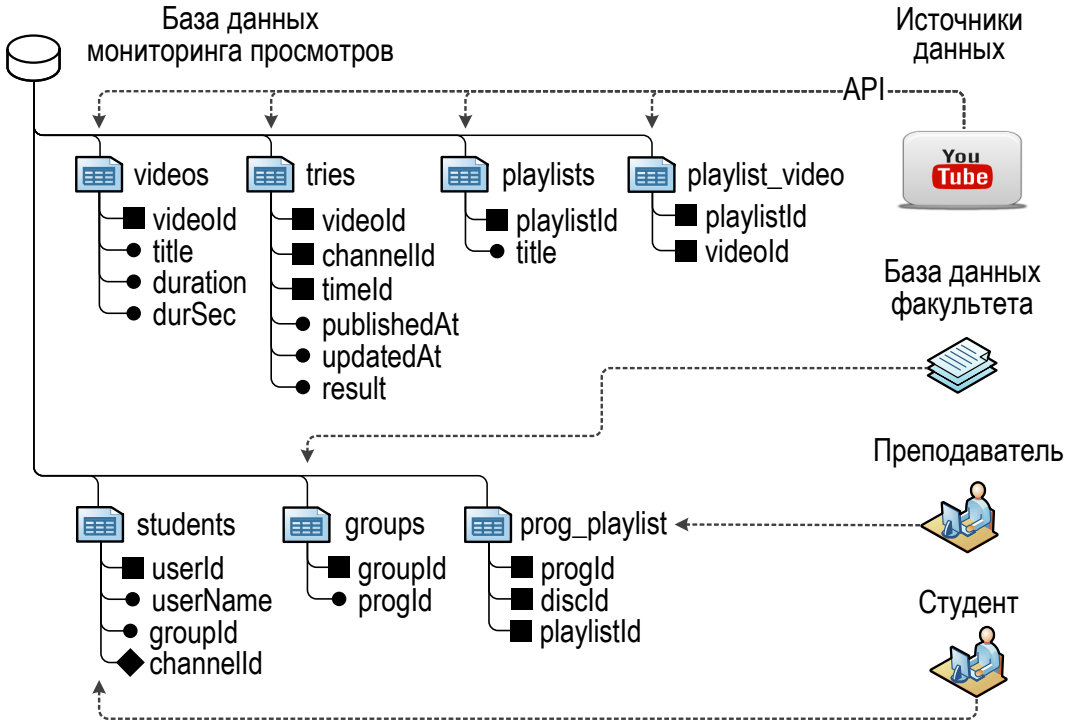


Рисунок 2 — Реляционная база данных мониторинга просмотров видео

Таблица `videos` содержит сведения о видео. Источником данных для этой таблицы служит веб-сервис YouTube API Video [14–24]. Атрибут `videoId` — это идентификатор, который YouTube назначает каждому размещенному видео. Для удобства использования в таблицу добавлен числовой атрибут `durSec`, который соответствует символьному атрибуту `duration` (продолжительность).

Таблица `playlists` содержит сведения о плейлистах, содержащих тематическое множество видео. Атрибут `playlistId` — это идентификатор, который YouTube назначает каждому созданному плейлисту, а атрибут `title` — это название плейлиста. Таким образом, эта таблица является справочником названий плейлистов. Источником данных для этой таблицы служит веб-сервис YouTube API Playlists.

Таблица `playlist_video` содержит сведения о видео, включенных в плейлист. Эта таблица содержит пары атрибутов (`playlistId`, `videoId`). Источником данных для этой таблицы служит веб-сервис YouTube API Playlist Items.

Таблица `tries` содержит сведения о попытках просмотра видео студентом. Как уже отмечалось, каждой попытке соответствует свой комментарий. Источником данных для этой таблицы служит веб-сервис YouTube API Comment Threads, предоставляющий информацию о комментариях к видео. Строки таблицы идентифицируются тройкой атрибутов: `videoId`, `channelId` и `timeId`. Атрибут `videoId` — это идентификатор видео, как в таблице `videos`. Атрибут `channelId` — это идентификатор канала пользователя, разместившего данный комментарий. При заполнении таблицы учитываются только комментарии, принадлежащие студенту, зарегистрированному в системе, т. е. пользователю, чей идентификатор канала присутствует в таблице `students`. Атрибут `timeId` — это метка времени, которая идентифицирует различные попытки одного и того же студента к одному и тому же видео.

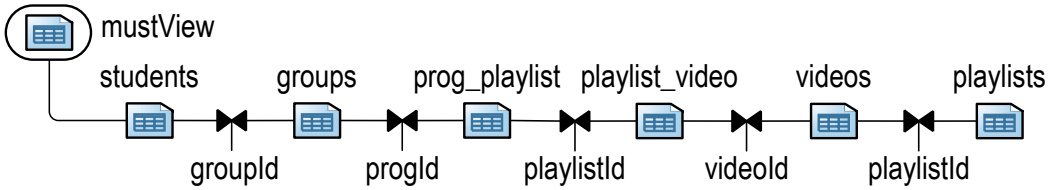
Таблица `students` содержит сведения о студентах и пополняется в ходе онлайн-регистрации студентов на начальном этапе освоения учебной дисциплины. Атрибут `userId` — это общий идентификатор студента. Он используется для идентификации строк таблицы. Атрибут `groupId` — это идентификатор студенческой группы, к которой прикреплен студент. Атрибут `channelId` — это идентификатор YouTube-канала, принадлежащего студенту, через который студент будет оставлять комментарии к видео. Этот атрибут должен быть уникальным, т. е. не может быть двух студентов с одинаковым идентификатором канала.

Таблица `groups` содержит сведения о студенческих группах. Источником данных для этой таблицы служит база данных деканата. Атрибут `groupId` — это идентификатор группы, а атрибут `progId` — это идентификатор направления обучения группы.

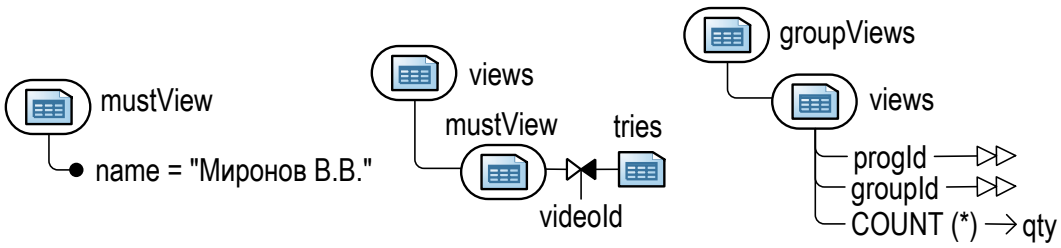
Таблица `prog_playlist` содержит сведения о плейлистах, которые должны просмотреть студенты, обучающиеся по некоторому направлению в рамках некоторой учебной дисциплины. Источником данных для этой таблицы служит рабочая программа, устанавливающая требования к освоению дисциплины. Атрибут `progId` — это идентификатор направления обучения, атрибут `discId` — это идентификатор дисциплины, а атрибут `playlistId` — это идентификатор плейлиста.

Таким образом, таблицы базы данных заполняются/пополняются в разное время из разных источников данных. Для формирования различных отчетов, предоставляемых преподавателям и студентам, используются различные сочетания таблиц. Например, для формирования отчета (со всеми атрибутами) о видеороликах, которые студенты должны просмотреть при освоении учебных дисциплин, требуется соединение таблиц `students`, `groups`, `prog_playlist`, `playlist_video`, `videos`, `playlists`. Модель соответствующей виртуальной таблицы `mustView` представлена на рисунке 3, *a*. Здесь символ «бабочка» означает реляционную операцию внутреннего эквисоединения таблиц. Строки этой виртуальной таблицы идентифицируются четверкой атрибутов-идентификаторов (`userId`, `playlistId`, `discId`, `videoId`). Данные, относящиеся к отдельному направлению подготовки, к отдельной дисциплине, к отдельной студенческой группе и т. д., получаются из виртуальной таблицы `mustView` путем фильтрации строк по значениям соответствующих атрибутов. На рисунке 3, *b* представлена модель, иллюстрирующая фильтрацию таблицы для студента `name = "Миронов В. В."`.

Чтобы получить сведения для отчета о просмотрах, нужно соединить виртуальную таблицу `mustView` с таблицей `tries` (виртуальная таблица `views`, рисунок 3, *c*). Отметим, что для формирования виртуальной таблицы `views` применяется реляционная операция внешнего левого соединения (светлый треугольник слева в символе операции соединения). Это позволяет отразить обязательные просмотры, которые еще не были выполнены студентом (в этом случае атрибутам просмотров присваиваются null-значения).



a



b c d

Рисунок 3 — Соединение таблиц при формировании отчетов мониторинга просмотров видео: a — сведения о видеороликах, которые студенты должны просмотреть при освоении учебных дисциплин; b — фильтрация строк по значению атрибута; c — сведения о просмотрах обязательных видеороликов; d — получение сводных показателей путем группирования таблицы

Виртуальная таблица `views` содержит сведения самого низкого уровня гранулярности. Чтобы получить укрупненные (сводные) показатели, нужно подвергнуть эту таблицу реляционной операции группирования по соответствующему критерию. На рисунке 3, *d* иллюстрируется получение сводного показателя `qty` (количество просмотров) на уровне студенческих групп. Для этого виртуальная таблица `views` сгруппирована по паре атрибутов (`progId`, `groupId`). В результате множество исходных строк таблицы `views` разбито на подмножества, соответствующие одной студенческой группе. К каждому подмножеству применяется агрегатная функция `COUNT (*)` (подсчет числа строк), что дает в результате общее число попыток просмотров для каждой студенческой группы.

1.3. Аналитические отчеты о просмотрах видео

Аналитические отчеты о просмотрах строятся на основании фактов, сохраненных в базе данных, а также другой справочной информации, собираемой при регистрации студентов в образовательной системе. Отчеты предназначены как для преподавателей, так и для студентов. Отчеты содержат как показатели фактов, так производные показатели, вычисляемые на основе показателей фактов (попытки, зачтенные, зачтенное время, оставшееся время и т. д.). Показатели формируются с различной степенью укрупнения (видео, плейлист, дисциплина и т. д.). Примеры таких отчетов иллюстрируются на рисунках 4 и 5.

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		

Рисунок 4 — Фрагмент отчета о просмотрах видео для студентов

Первый отчет (рисунок 4) ориентирован на студентов, т. е. содержит сведения о просмотрах видео, которые обязательны для студентов при освоении определенной учебной дисциплины. Видео сгруппированы в тематические плейлисты, для которых в отчете приведены соответствующие сводные показатели. Также в таблице приведены общие итоги для данного студента по данной дисциплине.

Показатели, представленные в отчете, отражают зачтенные (выполненные) и оставшиеся (задолженные) просмотры в количественном, временном и процентном выражении с разной степенью укрупнения. Так, из отчета видно, что в целом студент «Миронов В. В.» просмотрел по дисциплине «ИМД» 4 видеоролика общей продолжительностью 0,8 часа, а осталось просмотреть по этой дисциплине 17 видеороликов общей продолжительностью 9,2 часа, т. е. выполнено 13% требуемого объема. Эти итоговые показатели детализированы в отчете на уровне плейлистов и на уровне отдельных видео. Также из отчета видно, что плейлист «I. ДАС» просмотрен на 76%. При этом видео 1–4 просмотрены полностью, а видео 5 только на 45%; видео 2 просмотрено за две попытки, остальные просмотрены за одну попытку.

Интерактивные возможности отчета состоят в том, что названия видео одновременно является гиперссылкой на это видео в веб. Кликнув по названию, студент переходит на соответствующую страницу на сайте YouTube для просмотра видео.

Отчеты, ориентированные на преподавателя (рисунок 5), дополнительно содержат уровни большего укрупнения: уровень

студенческой группы, уровень направления обучения, уровень учебной дисциплины и др. В качестве клиента для этого отчета применена сводная таблица Excel, предоставляющая большие возможности гибкого отображения укрупненных или детальных данных, получаемых из микросервиса [14–24].

Группа–студент–плейлист–ролик	Зачтено, рол.	Осталось, рол	Зачтено, час	Осталось, час	Просмотров	% Ролика
КАИД-108м	**1**	**43**	**0,2**	**20**	**3**	
Миронов В.В.	**1**	**21**	**0,2**	**10**	**3**	
«Цветущая сложность»: обзор СУБД	1	5	0,2	7,7	3	
0. Общие замечания: видеолекция из	1		0,2		2	100
1. Дореляционные СУБД: видеолекция		1		1,6	1	6
2. Реляционные СУБД: видеолекция из		1		1,7		
3. Объектные СУБД: видеолекция из се		1		1,2		
4. Многомерные СУБД: видеолекция и		1		1,7		
5.1. NoSQL, NewSQL и др: видеолекция		1		1,5		
Данные в автоматизированных системах		5		0,9		
Иерархические модели данных		11		1,4		
Шадрина М.В.		**22**		**10**		
ЭАС-308		**94**		**26**		

Рисунок 5 — Фрагмент отчета о просмотрах видео для преподавателей

2. РЕШЕНИЯ НА ОСНОВЕ СИТУАЦИОННО ОРИЕНТИРОВАННЫХ БАЗ ДАННЫХ

Изложенные выше концепции можно рассматривать как техническое задание (development specification) на создание подсистемы мониторинга просмотров. Основываясь на них, можно разработать подсистему мониторинга, используя известные традиционные методы и подходы веб-проектирования. Наша цель, как уже отмечалось выше, состоит в том, чтобы продемонстрировать возможности и преимущества решения этой задачи на основе ситуационно-ориентированных баз данных. А именно, показать, что таким путем задача решается быстрее и проще.

2.1. Ситуационно-ориентированные базы данных

Ситуационно-ориентированная база данных (СОБД) — это интегратор гетерогенных данных, управляемый встроенной ситуационной

моделью. Ключевые особенности СОБД, существенные для рассматриваемой задачи мониторинга просмотров, это:

– встроенная ситуационная модель (HSM — Hierarchical Situation Model), которая задает возможные состояния процесса мониторинга и возможные переходы между состояниями;

– виртуальные документы, ассоциированных с состояниями ситуационной модели, которые отображаются на разнородные источники реальных данных;

– интерпретатор, который в ходе обработки ситуационной модели обрабатывает виртуальные документы.

Ситуационная модель представляет собой иерархию субмоделей, каждая из которых, в свою очередь, — это модель конечных состояний. Субмодель состоит из элементов: возможных состояний и возможных переходов между состояниями. Иерархия субмоделей образуется за счет вложенности субмоделей. Каждая субмодель (кроме одной корневой субмодели) является ребенком некоторого состояния своей родительской субмодели. Тем самым некорневая субмодель задает субсостояния для своего родительского состояния.

Виртуальные документы — это элементы состояний HSM, которые задают отображение на внешние данные. Тем самым виртуальные документы позволяют единообразно обрабатывать разнородные (гетерогенные) реальные данные, размещенные в физических хранилищах данных. В настоящее время СОБД способна отображать виртуальные документы на следующие типы реальных данных:

– на локальные или удаленные файлы в формате XML или JSON;

– на веб-сервисы в сети интернет;

– на локальные ZIP-архивы;

– на реляционные базы данных.

Интерпретатор ситуационной модели (HSMI) — это программа, обрабатывающая документы в соответствии с встроенной ситуационной моделью. Интерпретатор циклически или по запросу обходит иерархии субмоделей, выполняя две основные функции:

– отслеживает текущего состояния ситуационной модели;

– обрабатывает виртуальные документы, ассоциированных с текущими состояниями.

Обработав ситуационную модель на некотором цикле интерпретации, интерпретатор сохраняет информацию о текущих состояниях субмоделей во внешней памяти в виде модели текущего состояния (CSM). Интерпретатор использует эту информацию для корректного продолжения обработки на следующем цикле интерпретации.

Обработка виртуальных документов задается в ситуационной модели с помощью DPO-элементов (Data Processing Objects — объектов обработки данных). В ходе интерпретации на основании DPO-элементов интерпретатор создает соответствующие объекты обработки. Реальные документы загружаются в объекты обработки на основании соответствующих виртуальных документов. Результаты обработки загруженных документов сохраняются во внешней среде.

2.2. Ситуационная модель мониторинга просмотров

Фрагмент одной из субмоделей HSM в XML-нотации приведен в листинге 1. Субмодель `sub:videos` выполняет функции RESTful-сервиса `videos`, который получает из YouTube сведения о видеоролике и загружает требуемые данные в таблицу `videos` базы данных подсистемы мониторинга просмотров.

Листинг 1 — Фрагмент субмодели

```
001  <sub:videos>
002    <sta:begin>
003      <doc:request type = "HTTP"/>
004      <arr:http srcDoc = "request" />
005      <doc:request type = "HTTP"/>
006      <doc:DB action = "MySQLi-connect" onConnectErr = "MySQLiErr">
007        <ent:videosAll path = "videos"/>
008      </doc:DB >
009      <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}}"/>
010      <jmp:further getTarg = "arr::http.method"/>
011    </sta:begin>
012    <sta:POST>
013      <jmp:requestParamsErr when = "NOT arr::http.data.videoId"/>
014      <arr:videoDetails srcDoc = "YTVideos"/>
015      <jmp:notFoundErr when = "arr::videoDetails.pageInfo.totalResults=0"/>
016      <arr:videoDetails srcDoc = "YTVideos">
017        <rcv:extract value = "func::extract"
             to = "doc::DB.videosAll#INS,ignore"/>
```

```
018         <rcv:info-affected value = "func::getMySQLiAffected"
          </arr:videoDetails>
019       <jmp:MySQLiErr when = "ERRs::last"/>
020       <doc:request action = "response" code = "200"
            data = "arr::info#json"/>
021     </sta:POST>
022     <sta:MySQLiErr>
023       <doc:request action = "response" code = "500"
            message = "Internal Server Error: {{ERRs::last.message}}"/>
024     </sta:MySQLiErr>
025     <sta:requestParamsErr>
026       <doc:request action = "response" code = "404"
            message = "Bad Request: missing required params"/>
027     </sta:requestParamsErr>
028     <sta:notFoundErr>
029       <doc:request action = "response" code = "404"
            message = "Bad Request: {{arr::http.data.videoId}}"/>
030     </sta:notFoundErr>
031   </sub:videos>
```

Субмодель содержит несколько состояний, соответствующих различным ситуациям функционирования:

`sta:begin` — это начальное состояние обработки, в котором декларируются виртуальные документы, используемые сервисом, и выбирается ситуация для продолжения обработки;

`sta:POST`, `sta:GET`, `sta:PUT`, и `sta:DELETE` — это состояния, задающие продолжение обработки в соответствии с HTTP Request Method (только первое из этих состояний показано на рис. 6, чтобы не загромождать модель);

`sta:MySQLiErr`, `sta:requestParamsErr`, и `sta:notFoundErr` — это состояния для завершение обработки в исключительных ситуациях, когда обнаружена та или иная ошибка.

Виртуальные документы, используемые сервисом, задают отображение на внешние данные:

`doc:request` — это отображение на HTTP request, который запросил данный сервис. Request method и параметры, переданные с запросом, становятся доступными в формате JSON с помощью этого виртуального документа, загруженного в dpo-объект `arr:http`;

doc:YTVideos — это отображение на сервис YouTube API Videos, который предоставляет сведения в формате JSON о конкретном видеоролике по заданному идентификатору;

doc:DB — это отображение на реляционную базу данных MySQL подсистемы мониторинга просмотров. Отображение устанавливает соединение с базой данных (в случае ошибки выполняется переход в состояние sta:MySQLiErr). Entry-элемент ent:videosAll указывает на таблицу video в подсоединенной базе данных;

Выбор дальнейшей обработки задается элементом перехода jmp:further, который обеспечивает переход в состояние, соответствующее методу HTTP request. Так, если метод POST, то выполняется переход в состояние sta:POST. Обработка на рис. 6 раскрыта только для случая метода POST. Этот метод обычно используется в RESTful-сервисах для выполнения операций обновления веб-ресурсов. Таблица videos обновляется этим методом в рассматриваемом микросервисе. Для этого ассоциативный массив arr:videoDetails создается и виртуальный JSON-документ doc:YTVideos в него загружается. Интересующие данные вычленяются из загруженного массива с помощью элемента rcv:extract, форматируются и передаются в виртуальный документ doc:DB.videosAll для загрузки в таблицу videos базы данных. Загрузка в таблицу выполняется SQL-оператором INSERT IGNORE, что обеспечивает вставку сведений только о новых видеороликах при использовании метода POST.

2.3. REST-сервисы для мониторинга просмотров

Реализация рабочей версии подсистемы мониторинга просмотров выполнена на основе сервис-ориентированной архитектуры, а именно, на основе RESTful микросервисов. Данная архитектура обеспечивает высокую независимость и модифицируемость компонентов подсистемы, что важно в условиях разнородности и изменчивости используемых источников данных.

В рамках реализации этого подхода разработан комплекс микросервисов в среде ситуационно-ориентированнных баз данных. Микросервисы нижнего уровня ориентированы на обслуживание отдельных таблиц в базе данных подсистемы мониторинга просмотров. Эти микросервисы позволяют вносить, извлекать, обновлять и удалять данные:

– микросервис videos обрабатывает таблицу videos, в том числе заносит в нее сведения, получаемые из веб-сервиса YouTube API Video;

– микросервис `playlists` обрабатывает таблицу `playlists` на основе сведений, получаемых из веб-сервиса YouTube API Playlists;

– микросервис `playlist_video` обрабатывает таблицу `playlist_video`, основываясь на данных из веб-сервиса YouTube API Playlist Items;

– микросервис `tries` обрабатывает таблицу `tries`, в том числе заносит сведения о комментариях из веб-сервиса YouTube API Comment Threads;

– микросервис `students` обрабатывает таблицу `students`, в том числе заносит сведения о студентах в ходе онлайн-регистрации студентов на начальном этапе освоения учебной дисциплины;

– микросервис `groups` обрабатывает таблицу `groups` на основе базы данных деканата;

– микросервис `prog_playlist` обрабатывает таблицу `prog_playlist`, содержащую сведения о плейлистах, которые должны просмотреть студенты.

Микросервисы верхнего уровня обрабатывают данные из нескольких таблиц базы данных, используя микросервисы нижнего уровня. Так, микросервис `reports` формирует данные для аналитических отчетов как для преподавателей, так и для студентов. Примеры таких отчетов были рассмотрены выше (см. рисунки 4 и 5).

ЗАКЛЮЧЕНИЕ

Таким образом, анализ комментариев студентов предложен для решения задачи мониторинга просмотров студентами образовательных видео, размещенных на YouTube. Для этого подсистема мониторинга просмотров предусмотрена в составе университетской образовательной системы. Комментарии студентов извлекаются с помощью YouTube API и заносятся в реляционную базу данных подсистемы мониторинга. На основе этой и другой информации аналитические отчеты о просмотрах формируются как для преподавателей, так и для студентов. Подсистема мониторинга реализована на основе ситуационно-ориентированных баз данных, которые используются как для заполнения базы данных, так и для формирования аналитических отчетов о просмотрах видео. Подсистема мониторинга имеет сервис-ориентированную архитектуру, базирующуюся на принципах организации REST и микросервисов. Универсальность и простота иерархической ситуационной модели достигается как при

определении веб-сервисов, так и при управлении ими, при использовании ситуационно-ориентированных баз данных для решения этой задачи.

Работа выполнена в рамках проекта развития ситуационно-ориентированных баз данных, поддержанного Российским фондом фундаментальных исследований (РФФИ), грант 19-07-00682.

ЛИТЕРАТУРА

1. Poche E. et al. Analyzing User Comments on YouTube Coding Tutorial Videos. In: IEEE International Conference on Program Comprehension. 2017. P. 196–206.
2. Shoufan A. Estimating the cognitive value of YouTube's educational videos: A learning analytics approach. Computers in Human Behavior. 2019. Vol. 92. P. 450–458.
3. Silva H., Azevedo I. Instructional videos and others on Youtube: Similarities and differences in comments. In: CSEDU 2017 - Proceedings of the 9th International Conference on Computer Supported Education. 2017. Vol. 1. P. 418–425.
4. Lee C.S. et al. Making sense of comments on YouTube educational videos: A self-directed learning perspective. Online Information Review. 2017. Vol. 41, № 5. P. 611–625.
5. Saurabh S., Gautam S. Modelling and statistical analysis of YouTube's educational videos: A channel Owner's perspective. Computers and Education. 2019. Vol. 128. P. 145–158.
6. Mitrovic A. et al. Reflective experiential learning: Using active video watching for soft skills training. In: ICCE 2016 - 24th International Conference on Computers in Education: Think Global Act Local - Main Conference Proceedings. 2016. P. 192–201.
7. Mironov V. V., Gusarenko A. S., Yusupova N. I. Stream handling large volume documents in situationally-oriented databases. In: International Scientific Journal INDUSTRY 4.0. Scientific Technical Union of Mechanical Engineering "INDUSTRY 4.0." 2018. Vol. 3, № 5. P. 240–244.
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.
9. 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.

10. Миронов В. В., Гусаренко А. С., Юсупова Н. И. Отображение виртуальных XML-документов на таблицы MySQL в ситуационно-ориентированных базах данных: «распределенный» подход // Информационные технологии и вычислительные системы. 2017. № 1. С. 77–89.
11. 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.
12. Миронов В. В., Гусаренко А. С., Юсупова Н. И. Ситуационно-ориентированные базы данных: современное состояние и перспективы исследования // Вестник УГАТУ. 2015. Т. 19, № 2 (68). С. 188–199.
13. Миронов В. В., Гусаренко А. С. Использование RESTful-сервисов в ситуационно-ориентированных базах данных // Вестник УГАТУ. 2015. Т. 19, № 1 (67). С. 232–239.
14. What's a (micro)service – part 1?, http://chrisrichardson.net/post/microservices/general/2019/02/16/whats-a-service-part-1.html, last accessed 06/05/2019.
15. What are microservices?, http://microservices.io/index.html, last accessed 06/05/2019.
16. Microservices Pattern: Microservice Architecture pattern, http://microservices.io/patterns/microservices.html, last accessed 06/05/2019.
17. Microservices Pattern: A pattern language for microservices, http://microservices.io/patterns/, last accessed accessed: 06/05/2019.
18. Dixon D. PHP Microservices — Creating A Basic Restful Crud API, https://medium.com/helium-mvc/php-microservices-creating-a-basic-restful-crud-api-dabb1a1941a5, last accessed 06/05/2019.
19. Wang F.-J., Fahmi F. Constructing a service software with microservices. In: Proceedings - 2018 IEEE World Congress on Services, SERVICES 2018. 2018. P. 33–34.
20. Villaça L.H.N., Azevedo L.G., Baio F. Query strategies on polyglot persistence in microservices. In: Proceedings of the ACM Symposium on Applied Computing. 2018. P. 1725–1732.
21. Dragoni N. et al. Microservices: How to make your application scale // Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). 2018. Vol. 10742 LNCS. P. 95–104.
22. Donham J. A domain-specific language for microservices. In: Scala 2018 - Proceedings of the 9th ACM SIGPLAN International Symposium on Scala, co-located with ICFP 2018. 2018. P. 2–12.

23. Cavallari M., Tornieri F. Information systems architecture and organization in the Era of MicroServices. Lecture Notes in Information Systems and Organisation. 2018. Vol. 24. P. 165–177.
24. Petrasch R. Model-based engineering for microservice architectures using Enterprise Integration Patterns for inter-service communication. In: Proceedings of the 2017 14th International Joint Conference on Computer Science and Software Engineering, JCSSE 2017 2017.

V.V. Mironov, A.S. Gusarenko, N.I. Yusupova

APPLICATION OF WEB SERVICES BASED ON SITUATION-ORIENTED DATABASE FOR MONITORING THE VIEWING OF THE EDUCATIONAL VIDEO-CONTENT

Ufa State Aviation Technical University,
Ufa, Russia

This article discusses the task of monitoring student views of educational videos hosted on YouTube video hosting. A solution to this problem is proposed based on the control and analysis of comments posted by students during viewing. The organization of this process is discussed, as well as the functionality of the video viewing monitoring subsystem, which provides for the collection and analysis of student comments, as part of the university educational system. The structure of a relational database for the accumulation of information about video views is considered. Examples of analytical reports on watching videos aimed at students and teachers are given. To fill the relational data warehouse (ETL process), as well as to generate analytical reports on video viewing, a situation-oriented database (SODB) is used. The SODB capabilities for organizing micro-services using the example of heterogeneous data management, which are extracted from the YouTube API and the database of the educational system storage, are demonstrated and then placed in a relational database based on the concept of virtual documents displayed on heterogeneous data sources. This implementation is explained when mapping virtual documents to web services such as the YouTube API. A set of REST-services is developed that is developed on the base of SODB to solve the task of monitoring views. The universality and simplicity of the hierarchical situational model of the SODB is noted both when defining web services and when managing them. The practical implementation of the subsystem for monitoring the views of educational videos on the PHP platform is described.

Keywords: educational videos, monitoring views, YouTube API, ETL-process, analytical reports, integration heterogeneous data, situation-oriented databases, REST-services, microservices.

REFERENCES

1. Poche E. et al. Analyzing User Comments on YouTube Coding Tutorial Videos. In: IEEE International Conference on Program Comprehension. 2017. P. 196–206.

2. Shoufan A. Estimating the cognitive value of YouTube's educational videos: A learning analytics approach. Computers in Human Behavior. 2019. Vol. 92. P. 450–458.
3. Silva H., Azevedo I. Instructional videos and others on Youtube: Similarities and differences in comments. In: CSEDU 2017 - Proceedings of the 9th International Conference on Computer Supported Education. 2017. Vol. 1. P. 418–425.
4. Lee C.S. et al. Making sense of comments on YouTube educational videos: A self-directed learning perspective. Online Information Review. 2017. Vol. 41, № 5. P. 611–625.
5. Saurabh S., Gautam S. Modelling and statistical analysis of YouTube's educational videos: A channel Owner's perspective. Computers and Education. 2019. Vol. 128. P. 145–158.
6. Mitrovic A. et al. Reflective experiential learning: Using active video watching for soft skills training. In: ICCE 2016 - 24th International Conference on Computers in Education: Think Global Act Local - Main Conference Proceedings. 2016. P. 192–201.
7. 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. Vol. 3, № 5. P. 240–244.
8. Mironov V.V., Gusarenko A.S., Yusupova N.I. Integration of Virtual Multidocument Mappings into Real Data Sources in Situational-Oriented Databases. Applied Informatics. 2018. Vol. 13, № 3(75). P. 47–60.
9. Mironov V.V., Gusarenko A.S., Yusupova N.I. Structuring virtual multi-documents in situationally-oriented databases by means of entry-elements. SPIIRAS Proceedings. 2017. Vol. 4, № 53. P. 225–242.
10. Mironov V.V., Gusarenko A.S., Yusupova N.I. Displaying virtual XML-documents on MySQL tables in the situation-oriented databases, "distributed approach". Journal of Information Technologies and Computing Systems. 2017. № 1. P. 77–89.
11. 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. Herald of Computer and Information Technologies. 2017. № 1(151). P. 29–36.
12. Mironov V.V., Gusarenko A.S., Yusupova N.I. Situation-Oriented Databases: Current State and Prospects for Research. Vestnik UGATU. 2015. Vol. 19, № 2 (68). P. 188–199.
13. Mironov V.V., Gusarenko A.S. Using of RESTful-Services in Situationally-Oriented Databases. Vestnik UGATU. 2015. Vol. 19, № 1 (67). P. 232–239.

14. What's a (micro)service - part 1?, http://chrisrichardson.net/post/microservices/general/2019/02/16/whats-a-service-part-1.html, last accessed 06/05/2019.
15. What are microservices?, http://microservices.io/index.html, last accessed 06/05/2019.
16. Microservices Pattern: Microservice Architecture pattern, http://microservices.io/patterns/microservices.html, last accessed 06/05/2019.
17. Microservices Pattern: A pattern language for microservices, http://microservices.io/patterns/, last accessed accessed: 06/05/2019.
18. Dixon D. PHP Microservices — Creating A Basic Restful Crud API, https://medium.com/helium-mvc/php-microservices-creating-a-basic-restful-crud-api-dabb1a1941a5, last accessed 06/05/2019.
19. Wang F.-J., Fahmi F. Constructing a service software with microservices. In: Proceedings - 2018 IEEE World Congress on Services, SERVICES 2018. 2018. P. 33–34.
20. Villaça L.H.N., Azevedo L.G., Baio F. Query strategies on polyglot persistence in microservices. In: Proceedings of the ACM Symposium on Applied Computing. 2018. P. 1725–1732.
21. Dragoni N. et al. Microservices: How to make your application scale // Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). 2018. Vol. 10742 LNCS. P. 95–104.
22. Donham J. A domain-specific language for microservices. In: Scala 2018 - Proceedings of the 9th ACM SIGPLAN International Symposium on Scala, co-located with ICFP 2018. 2018. P. 2–12.
23. Cavallari M., Tornieri F. Information systems architecture and organization in the Era of MicroServices. Lecture Notes in Information Systems and Organisation. 2018. Vol. 24. P. 165–177.
24. Petrasch R. Model-based engineering for microservice architectures using Enterprise Integration Patterns for inter-service communication. In: Proceedings of the 2017 14th International Joint Conference on Computer Science and Software Engineering, JCSSE 2017 2017.