УДК 004.4

МОДИФИКАЦИЯ ИНСТРУМЕНТАЛЬНОГО КОМПЛЕКСА «ПОСТРОИТЕЛЬ ТЬЮТОРОВ», БАЗИРУЮЩЕГОСЯ НА МИКРОСЕРВИСНОЙ АРХИТЕКТУРЕ

Бирюкова А.А., Красновский А.М., Салахов М.Р.

«МИРЭА - Российский технологический университет», 119454, Россия, г. Москва, проспект Вернадского, 78, e-mail: biryukova@mirea.ru, *akrasnovskii@yandex.ru, muratsalakhov@gmail.com*

В статье рассмотрена задача повышения эффективности работы инструментального комплекса «Построитель тьюторов» путём модификации подсистем «Редактор» и «Проигрыватель». Анализируется процесс создания обучающих программ с поддержкой ветвлений при помощи редактора инструментального комплекса. Подробно разобран вариант модификации редактора и описаны преимущества его модифицированной версии. Рассматривается изменение архитектуры серверной части подсистемы «Проигрыватель» и модификация базы данных для повышения её эффективности.

Ключевые слова: электронное обучение, инструментальный комплекс, построитель тьюторов, обучающие программы, ветвление в обучающей программе, микросервис.

MODIFICATION OF THE «TUTOR BUILDER» TOOLKIT, BASED ON THE MICROSERVICE ARCHITECTURE

Biryukova A.A., Krasnovskii A.M., Salakhov M.R.

«MIREA - Russian Technological University", 119454, Moscow, 78 Vernadskogo Avenue, Russia, e-mail: biryukova@mirea.ru, akrasnovskii@yandex.ru, muratsalakhov@gmail.com

The report discusses the problem of increasing the efficiency of the instrumental complex «Builder of tutors» by modifying the subsystems «Editor» and «Player». The process of creating training programs with branching support using the editor of the instrumental complex is analyzed. The variant of the editor modification is analyzed in detail and the advantages of its modified version are described. Changes in the architecture of the server part of the «Player» subsystem and modification of the database to improve its efficiency are considered.

Keywords: e-learning, toolkit, tutor builder, tutorials, branching in the tutorial, microservice.

Введение

Усовершенствование методов и инструментов электронного обучения в последние десятилетия представляло собой одно из актуальнейших направлений развития информационных технологий. Особо высокая значимость электронных средств обучения обусловлена настоящей ситуацией этого года – COVID-19, который вынудил огромные массы людей в различных областях деятельности перейти на удаленную работу, и при этом множество людей были вынуждены обучиться новым программным средствам удалённой работы. В этой связи создание инструментов, обеспечивающих и поддерживающих технологии опережающего обучения массовых профессиональных пользователей программных продуктов, несомненно, является актуальной задачей [1]. Предлагаемая статья относится к области построения обучающих программ. Одним из инструментов построения обучающих программ является инструментальный комплекс «Построитель тьюторов» [3].

Текущая версия инструментального комплекса «Построитель тьюторов» состоит из следующих основных подсистем: перехватчик, редактор, администратор и проигрыватель [6]. При этом серверная часть инструментального комплекса является единым монолитным приложением и используется всеми подсистемами инструментального комплекса. Главной особенностью такого приложения является необходимость использования определенного набора языков программирования и фреймворков для решения всех задач, что усложняет сопровождение инструментального комплекса. Эти недостатки монолитной модели сервера определяют необходимость обновлять и поддерживать инструментальный комплекс большой группой программистов. Поэтому было принято решение перейти к архитектуре на основе микросервисов. Микросервисная архитектура позволит увеличить эффективность инструментального комплекса, поскольку для каждого микросервиса могут использоваться технологии, наиболее эффективно решающие задачу микросервиса. Также при использовании микросервисной архитектуры можно выделить набор микросервисов и баз данных для

каждой подсистемы, что позволит сделать реализацию более приспособленной для решения поставленной задачи. Редактор является ядром системы «Построитель тьюторов», которое преобразует «заготовку», построенную перехватчиком, в полноценную обучающую программу. Проигрыватель поддерживает процесс обучения большого количества массовых профессиональных пользователей и должен обеспечивать эффективное взаимодействие с большим количеством клиентов. Рассмотрим более подробно модификацию этих двух основных подсистем.

Модификация подсистемы “Редактор”

Редактор инструментального комплекса «Построитель тьюторов» предназначен для создания обучающих программ для обучения массового профессионального пользователя. Инструментальный комплекс «Построитель тьюторов», как и каждое реальное программное средство, находится в непрерывном развитии. В последнем обновлении комплекса в редактор была добавлена поддержка ветвлений в обучающих программах, благодаря чему стало возможным учитывать предпочтения различных пользователей [5]. Ветвление в обучающей программе – это несколько альтернативных путей, приводящих к одному результату. Разным пользователям будет предпочтительнее проходить один и тот же сценарий разными способами для решения поставленной задачи за минимальное время.

Рассмотрим существующего механизма создания обучающих программ, проведем его анализ и определим недостатки. В текущей версии редактора обучающая программа состоит из нескольких связанных между собой разделов, а каждый раздел, в свою очередь, состоит из шагов. Чтобы создать обучающую программу, сначала необходимо создать её разделы. В процессе создания раздела можно менять разнообразные параметры: текст задания, текст подсказки, действие для перехода к следующему кадру, порядок кадров и другие. После того, как необходимые разделы созданы, создается сама обучающая программы: выбираются разделы, которые будут включены в обучающую программу, задается их порядок.

Для обучающих программ, которые содержат ветвление, необходимо выбрать по тексту подсказки кадр начала ветвления первого раздела, кадр окончания ветвления первого раздела, кадр второго раздела, присоединённый к кадру начала ветвления, кадр второго раздела, присоединённый к кадру окончания ветвления. Аналогичным образом можно добавить ветвление к уже присоединённому разделу. После создания обучающая программа не поддается никаким изменениям.

Существующий механизм создания обучающих программ содержит три существенных недостатка. Первый недостаток связан с тем, что выпущенную обучающую программу нельзя никаким образом изменить. Можно только внести необходимые изменения в разделы, из которых создавалась обучающая программа, и создать её заново. В таком случае придется затратить время на повторное определение точек ветвления. Второй недостаток связан с тем, что выбор кадров начала и окончания ветвления ограничен только одним разделом (основным или предыдущим присоединённым). Третий недостаток связан с невозможностью просматривать пути обучающей программы после её создания. Для того, чтобы проверить, что в точках ветвления переключение на альтернативные пути происходит правильно, необходимо многократно проходить обучающий сценарий в проигрывателе, делая разный выбор в точках ветвления.

Для устранения данных недостатков был разработан новый механизм создания обучающих программ с ветвлением. Основой нового механизма послужил тот факт, что обучающая программа с ветвлением представляет собой ориентированный ациклический граф, вершинами которого являются чередующиеся кадры и действия. Данный граф содержит одну начальную и одну конечную вершину. В новой версии редактора был введён новый функционал, позволяющий переключаться в точках ветвления между альтернативными путями обучающей программы, редактировать кадры и действия любого пути и добавлять альтернативный путь между произвольными кадрами любого уже существующего пути. Для переключения между альтернативными путями на кадрах начала ветвления был добавлен специальный индикатор, обозначающий, какой именно путь выбран в данный момент. Вследствие данных изменений пропала необходимость в разделении обучающей программы на разделы. Кадры и действия для альтернативных путей берутся из цепочек кадров и действий (фрагментов). После добавления альтернативного пути кадры и действия фрагмента (их копии) становятся частью самой обучающей программы, и к ним можно добавлять другие альтернативные пути.

Наглядное сравнение обучающей программы с ветвлением в текущей и новой версии редактора приведено на рис. 1. Для обозначения возможности редактирования кадров и действий используется символ «шестерёнки». В текущей версии редактора кадр начала и окончания ветвления ограничивается одним разделом, поэтому, в случае начала ветвления на кадре 2.2, закончить ветвление можно только на кадре 3.2.1 или 4.2. Как видно из рис. 1, в новой версии редактора можно редактировать кадры и действия любого существующего пути и добавлять альтернативные пути между двумя произвольными кадрами существующего пути. Например, путь «кадр 2.2 → кадр 3.2.2 → кадр 5» можно добавить только в новой версии редактора.

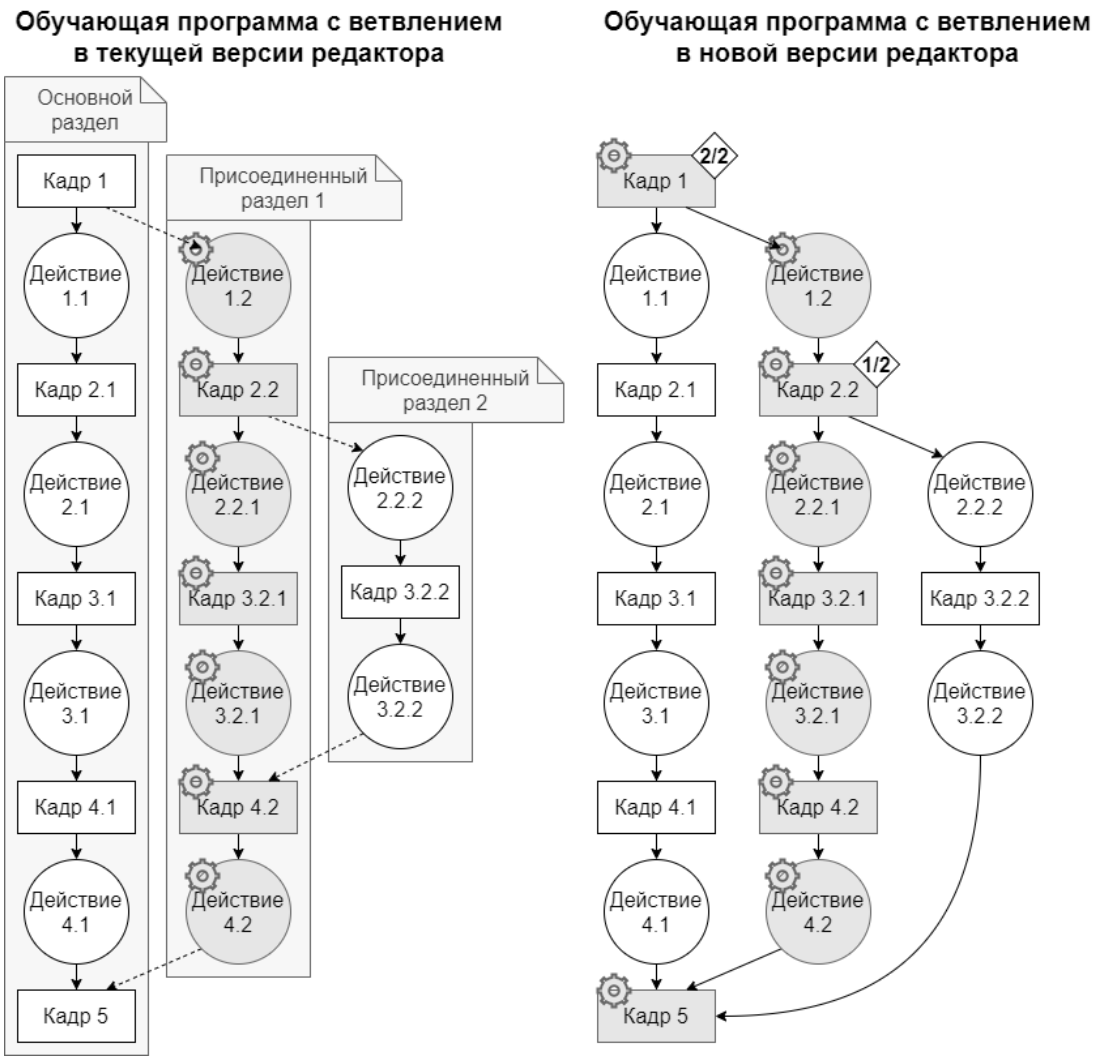

Рис. 1. Сравнение обучающей программы с ветвлением в текущей и новой версии редактора

Для реализации нового механизма создания обучающих программ с ветвлением потребовалось изменить формат хранения. В текущей версии редактора для хранения обучающих программ используется документоориентированная СУБД MongoDB, относящаяся к нереляционным базам данных. За счет использования микросервисной архитектуры в новой версии инструментального комплекса у редактора появилась возможность выбрать именно тот тип базы данных, который наилучшим образом подходит для хранения обучающих программ с ветвлением при условии их частого изменения, не навязывая использование той же базы данных другим компонентам инструментального комплекса, как это было сделано в текущей версии системы. Нереляционные базы данных можно разделить на документоориентированные и графовые. Документоориентированные базы данных предназначены для хранения иерархических структур данных, а графовые – для задач, в которых данные имеют большое количество связей. Поскольку графовые СУБД имеют преимущества в наглядности представления и простоте внесения изменений в схему базы данных, в новой версии редактора на базе микросервисов было решено использовать именно графовую СУБД.

Модификация подсистемы «Проигрыватель»

Подсистема «Проигрыватель» инструментального комплекса должна выполнять следующие функции: вывод списка доступных пользователю обучающих программ, воспроизведение обучающей программы, сбор статистики прохождения пользователем обучающей программы, преобразование обучающей программы [4]. Микросервисная архитектура позволит серверной части проигрывателя более эффективно выполнять свои задачи. Проигрыватель может быть оснащен отдельной базой данных и отдельными микросервисами среднего слоя, которые обслуживают только проигрыватель. База данных проигрывателя должна хранить обучающие программы, которые готовы к воспроизведению. Сама обучающая программа состоит из кадров (неделимая единица программы, содержит изображение-образ экрана программного продукта, задание и подсказку к заданию) и действий (информация о действии, которое ожидается от пользователя для перехода к следующему кадру). Инструментальный комплекс должен поддерживать неограниченное количество ветвлений,

следовательно из одного кадра пользователь может перейти к одному или более кадров. Такая структура программы является направленным графом с большим количеством отношений между вершинами, а значит, так же как и в редакторе, наиболее подходящий вид базы данных для хранения данной информации в проигрывателе– графовая. Текущая реализация инструментального комплекса хранит обучающие программы в документоориентированной базе данных, используя для этого 13 сущностей. В новой версии было принято решение отказаться от группировки кадров в разделы, которые раньше использовались для ветвлений программы, поскольку в новой версии ветвление может начинаться в любом месте программы. Также благодаря особенностям графовой базы данных отпадает необходимость в сущностях-соединителях, которые реализовывали связь между двумя другими сущностями. После оптимизации новая схема базы данных на основе графов содержит только три сущности: программа, кадр, действие. При этом сущность программа соединяется с первым кадром, а каждый кадр соединяется с другими с помощью действия. Новая схема хранения обучающей программы в базе данных проигрывателя приведена на рис. 2.

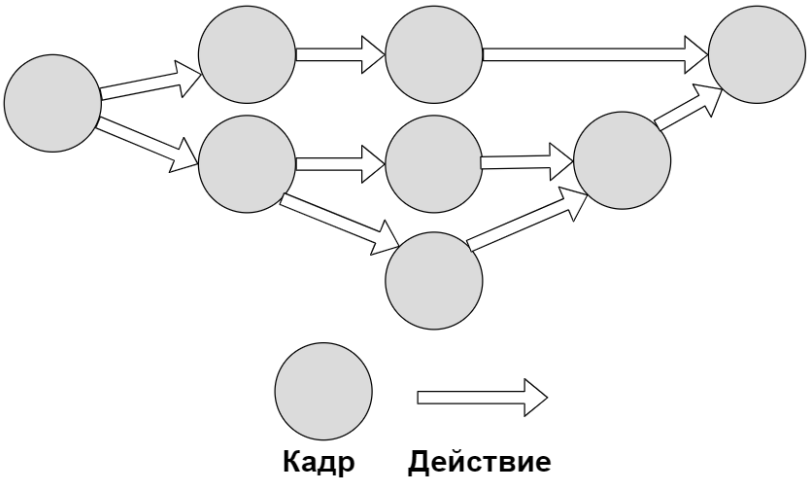

Рис. 2. Схема данных в базе данных проигрывателя

Для решения задач серверной части инструментального комплекса было выделено 2 микросервиса: микросервис API проигрывателя и микросервис-преобразователь программ. Микросервис API проигрывателя реализует программный интерфейс приложения (занимается транспортной логикой и взаимодействием с СУБД). Микросервис API проигрывателя работает только с базой данных проигрывателя и имеет интерфейсы для получения списка доступных пользователю обучающих программ, самой обучающей программы, а также для загрузки новой обучающей программы. Схема микросервисной архитектуры проигрывателя приведена на рис. 3.

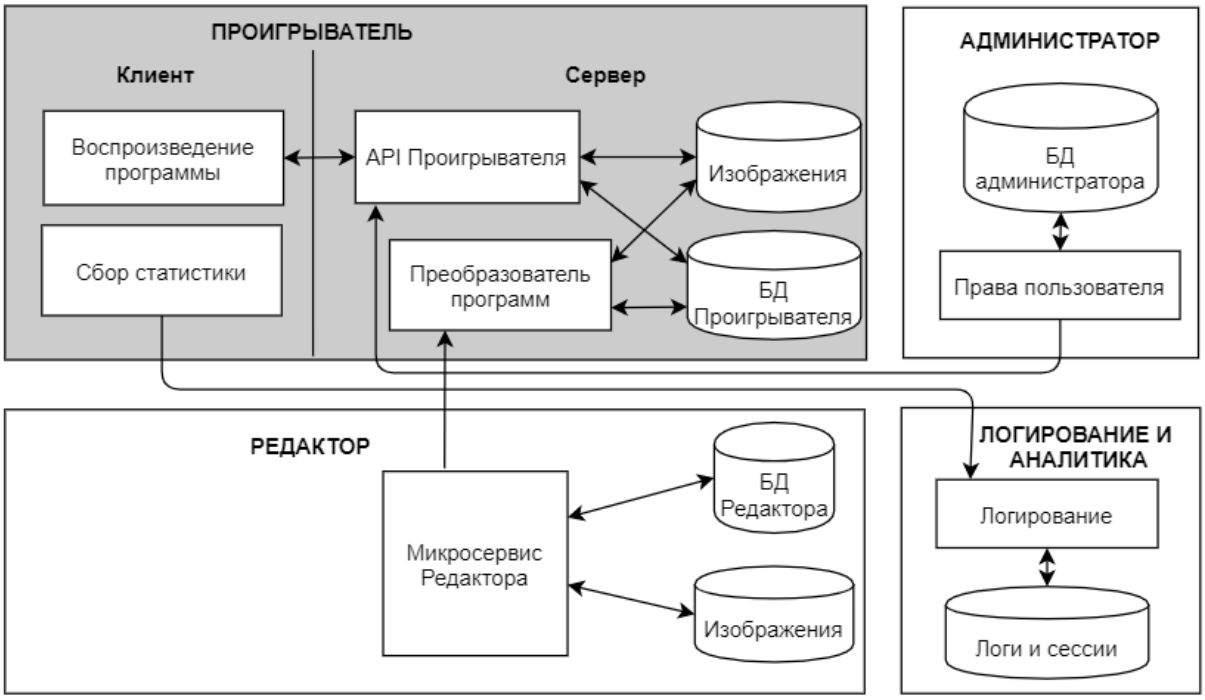

Рис. 3. Схема микросервисной архитектуры подсистемы проигрывателя.

Микросервис-преобразователь программ получает информацию о выпущенной программе из редактора, преобразует ее и записывает в базу данных в пригодном для проигрывания виде. Редактор хранит изображения-образы в исходном виде, а для лучшего воспроизведения обучающей программы проигрывателем нужно уменьшить вес этих изображений, чтобы не возникало задержек при их загрузке во время прохождения программы. Поэтому этот микросервис также занимается сжатием изображений-образов экрана программы. Микросервис получает список всех использованных изображений-образов обучающей программы, подгружает их из хранилища изображений редактора, сжимает и загружает в хранилище изображений проигрывателя. Под процессом сжатия понимается вычисление разницы между двумя последовательно идущими изображениями-образами. Таким образом, мы храним неизменно только первое изображение-образ, а все последующие получает путем наслоения на предыдущий кадр изменившихся пикселей.

Заключение

В результате использования микросервисной архитектуры и модификации подсистем «проигрыватель» и «редактор» появилась возможность увеличить эффективность работы инструментального комплекса за счет выбора для каждого задания наиболее подходящего инструментария. Модификация редактора инструментального комплекса «Построитель тьюторов» позволяет существенно расширить возможности по производству обучающих программ с ветвлением и уменьшить временные затраты при их редактировании, а модификация проигрывателя – обеспечить более высокую производительность при обучении большого количества пользователей.

Список литературы

1. Григорьев В.К. Модель обучения массовых профессиональных пользователей информационно-управляющих систем // Открытое образование. – 2009. – № 1 – С. 10-14.
2. Григорьев В.К., Бирюкова А.А. Экспериментальные исследования комплекса для создания обучающих программ «Построитель тьюторов». // Информатизация и связь. – 2016. – № 4. – С. 90-96.
3. Григорьев В.К. Инструментально-моделирующий комплекс для опережающего обучения МПП ИУС // Открытое образование. 2011. №1. – С. 44-55.
4. Волк А.Ю. Модификация процессов хранения и обработки данных в проигрывателе инструментального комплекса «Построитель тьюторов» // А.Ю. Волк, А.С. Илюшечкин. - Вестник науки и образования. – 2018. – С. 37.
5. Илюшечкин А.С. Расширение возможностей инструментального комплекса «Построитель тьюторов» за счет вариативности правильных действий / А.С. Илюшечкин, А.Ю. Волк // Современные информационные технологии в образовании: Материалы XXX международной конференции. Троицк – Москва. 2019. – С. 219-221.
6. Развитие автоматизированного комплекса «Построитель тьюторов» / В.К. Григорьев, А.А. Бирюкова, А.Ю. Волк, А.С. Илюшечкин // Информатизация и связь. – 2020. – № 1. – С. 21-27.
7. Ньюмен С. Создание микросервисов. – СПб.: Питер, 2016. – 304 с.: ил. – (Серия «Бестселлеры O’Reilly»).

References

1. Grigoriev V.K. Model of training of mass professional users of information and control systems / / Open education. – 2009. – No. 1 – P.10-14.
2. Grigoriev V.K., Biryukova A.A. Experimental research of the complex for creating training programs «Tutor Builder» // Informatization and communication. – 2016. – No. 4. – P. 90-96.
3. Grigoriev V.K. Instrumental modeling complex for advanced training of IUS MPP / / Open education. 2011. No. 1. – P. 44-55.
4. Volk A.U. Modification of data storage and processing processes in the player of the toolkit «Tutor Builder» / / A.Y. Volk, A.S. Ilyushechkin. - Bulletin of science and education. - 2018. - P. 37.
5. Ilyushechkin A.S. Expanding the capabilities of the toolkit «Tutor Builder» due to the variability of correct actions / A.S. Ilyushechkin, A.Y. Volk // Modern information technologies in education: Materials of the XXX international conference. Troitsk – Moscow. 2019. – P. 219-221.
6. Development of the automated toolkit «Tutor Builder» / V.K. Grigoriev, A.A. Biryukova, A.Y. Volk, A.S. Ilyushechkin / / Informatization and communication. – 2020. – No. 1. – P. 21-27.
7. Newman S. Creation of microservices. – St. Petersburg: Peter, 2016. – 304 p.: ill. – (Series «Bestsellers O'reilly»).