УДК 004.055
doi:10.18720/SPBPU/2/id24-482

Щербаков Николай Васильевич [1],
доцент, канд. техн. наук;
Резединова Евгения Юрьевна [2],
ст. преподаватель;
Щукин Александр Валентинович [3],
доцент, канд. техн. наук

АДАПТИВНАЯ АРХИТЕКТУРА ДЛЯ СОВРЕМЕННОЙ СИСТЕМЫ УПРАВЛЕНИЯ ОБУЧЕНИЕМ

[1] Австрия, Грац, Технический университет города Грац, wbtmaster2@gmail.com;
[2,3] Россия, Санкт-Петербург, Санкт-Петербургский политехнический университет Петра Великого, Высшая школа программной инженерии,
[2] rezedinova_eyu@spbstu.ru, [3] Alexander.Schukin@spbstu.ru

Аннотация. Система управления обучением (LMS) является важным компонентом общей инфраструктуры электронного обучения в современном университете. С

одной стороны, LMS представляет собой структурированное хранилище учебных материалов, снабженное дополнительными средствами коммуникации, такими как форум, аннотации, чаты и другие. С другой стороны, современная LMS реализует и более сложные сценарии обучения, такие как дистанционное проведение лекций, загрузка студенческих проектов, онлайн-анкетирование, наставничество, выполнение онлайн-упражнений. В данной статье авторы представляют архитектуру современной LMS под названием WBT (Web Based Training) Master. Такая архитектура позволяет обеспечить адаптивность системы управления обучения к требованиям онлайн-курсов и особенностям различных устройств, используемых пользователями. В статье также описывается архитектура системы и процесс адаптации функциональной части и пользовательских интерфейсов.

Ключевые слова: электронное обучение, система управления обучением, программная архитектура, облачный сервис, социальные сети, LMS.

Nikolai V. Scerbakov [1],
Associate Professor, Candidate of Technical Sciences;
Eugenia U. Rezedinova [2],
Senior Lecturer;
Alexander V. Schukin [3],
Associate Professor, Candidate of Technical Sciences

ADAPTIVE ARCHITECTURE FOR A MODERN LEARNING MANAGEMENT SYSTEM

[1] Graz University of Technology, Graz, Austria, wbtmaster2@gmail.com;
[2,3] Peter the Great St.Petersburg Polytechnic University, St. Petersburg, Russia,
[2] rezedinova_eyu@spbstu.ru, [3] Alexander.Schukin@avalon.ru

Abstract. Learning Management System (LMS) is an important component of the overall e-learning infrastructure in a modern university. On the one hand, LMS is a structured repository of learning materials, equipped with additional communication tools such as forums, annotations, chats, and others. On the other hand, a modern LMS implements more complex learning scenarios, such as remote lecturing, uploading of student projects, online polling, mentoring, and performing online exercises. In this article, the authors present the architecture of a modern LMS called WBT (Web Based Training) Master. This architecture allows the learning management system to adapt to the requirements of online courses and the features of various devices used by users. The article also describes the system architecture and the process of adapting the functional part and user interfaces.

Keywords: E-learning, learning management system, software architecture, cloud service, social networks, LMS.

Введение

WBT-Master – это современная и инновационная система управления обучением (LMS), которая разрабатывалась и успешно использовалась в техническом университете Граца (Австрия), в течение нескольких лет [1–3, 5, 6]. Существует целый ряд проблем, решение которых оправдывают разработку адаптивной LMS и отличают новую систему от существующих

реализаций [1]. Помимо таких общих функций системы, как анонсы курсов, файловые репозитории, описание учебных программ, обмен файлами, форумы, чаты, обмен электронной почтой, онлайн-викторины и опросы общественного мнения, средства аннотирования и оценки документов [7, 8] и т. д. разработанная система использует функциональность ряда облачных сервисов, таких как DropBox, Google Doc, MS One. Система поддерживает несколько инновационных сценариев электронного обучения, которые позволяют реализовать такие передовые парадигмы, как обучение на практике, перевернутое обучение, совместное решение проблем [9–12] и другие.

Цель разрабатываемой системы – преодоление существующих недостатков обучающих веб-приложений, таких как отсутствие адаптивности к разным требованиям курса и к различным пользовательским устройствам. Можно рассматривать курс электронного обучения как комбинацию учебного контента и нескольких обучающих приложений. Если рассматривать продвинутые системы управления обучением, то таких средств обучения должны быть десятки, начиная от примитивных средств обсуждения (форумы, чаты, социальные фреймворки) и заканчивая сложнейшей автоматической системой выставления оценок, основанной на анализе активности пользователей во время прохождения курса (например, на основе искусственного интеллекта) [13–16].

Можно рассматривать три возможных пути для решения существующих проблем:

– курс изначально обеспечивает беспрепятственное сочетание всех потенциально доступных средств обучения;

– инструменты могут быть реализованы как независимые веб-приложения, и преподаватель предоставляет обучающимся только ссылки на такие приложения;

– должен быть разработан специальный механизм, который позволит построить уникальную комбинацию инструментов и интерфейса для каждого курса и каждого конечного пользовательского устройства (компьютер, смартфон или планшет).

Очевидные недостатки первых двух методов (такие как сложный пользовательский интерфейс и запутанная навигация по разным компонентам курса) делают третий вариант более предпочтительным в долгосрочной перспективе. Кроме того, если рассматривать учебный курс, доступный на различных устройствах пользователя (в том числе мобильных – смартфон, планшет), архитектура, обеспечивающая столь гибкую адаптацию вида курса, представляется единственно возможной [5]. В данной статье авторы коротко излагают концепцию адаптивной архитектуры в том виде, в каком она была реализована в WBT-Master.

1. Постановка задачи

1.1. Основной функционал системы

Система реализована в виде так называемого AJAX (асинхронного Java и XML) веб-приложения. AJAX – это метод доступа к веб-серверам от клиента с помощью специальных объектов JavaScript «Fetch». Базовая функциональность системы разделена между компонентами Front-End и Back-End. Компонент Front-End реализован с помощью методов программирования JavaScript и HTML DOM. Front-End может запрашивать данные с сервера с помощью специального JavaScript-объекта; получать ответы и обрабатывать данные на стороне клиента. Компоненты Front-End и Back-End используют JSON (нотация объектов JavaScript) в качестве коммуникационного протокола.

Отмеченные выше преимущества имеют большое значение для современных LMS, поскольку они способствуют удовлетворению запросов пользователей в части функциональности и производительности системы, особенно в ситуации, когда сотни или даже тысячи студентов работают одновременно в ситуациях ограниченного времени, например, онлайн-экзамен, приближающиеся сроки дедлайна и т. п. При этом фактическая установка LMS Back-End не требует мощного оборудования и/или специальных аппаратных решений, таких как кластеры серверов, балансировщики нагрузки, реверсивные прокси и т. д.

Еще одним существенным аспектом этого архитектурного решения является возможность использования нескольких Front-End-приложений для связи с одним и тем же набором WEB-сервисов. Другими словами, могут быть разные пользовательские клиенты, которые можно использовать в разных обстоятельствах (настольный клиент, планшет, смартфон и т. д.).

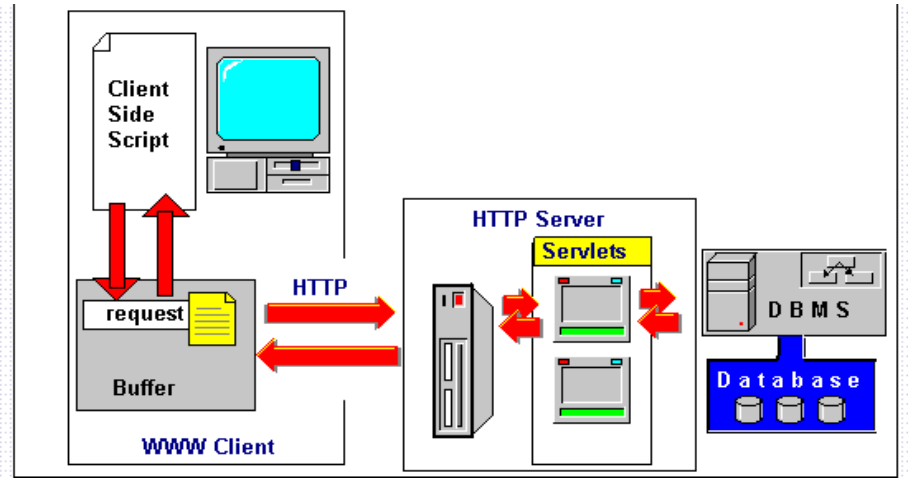

Рис. 1. Базовая функциональность системы

В предлагаемой нами архитектуре определены два уровня внешнего интерфейса. На первом уровне клиент получает доступ к стандартному HTML-документу и скрипту, который может интерпретировать специальный язык – язык описания курса. Кроме того, каждый курс определяется

как специальный файл описания курса в формате JSON. Описание курса загружается и динамически интерпретируется базовым компонентом первого слоя.

Описание курса определяет:

– иерархию так называемых виртуальных экранов (прямоугольные области на экране браузера), их положение и свойства видимости (экраны могут быть видимыми или скрытыми, перемещаемыми или постоянными, перекрывающимися и т. д.);

– драйверы (например, классы данных JavaScript), обеспечивающие функциональность обучающего приложения на виртуальном экране;

– коммуникаторы (например, классы JavaScript), обеспечивающие обмен данными между драйверами.

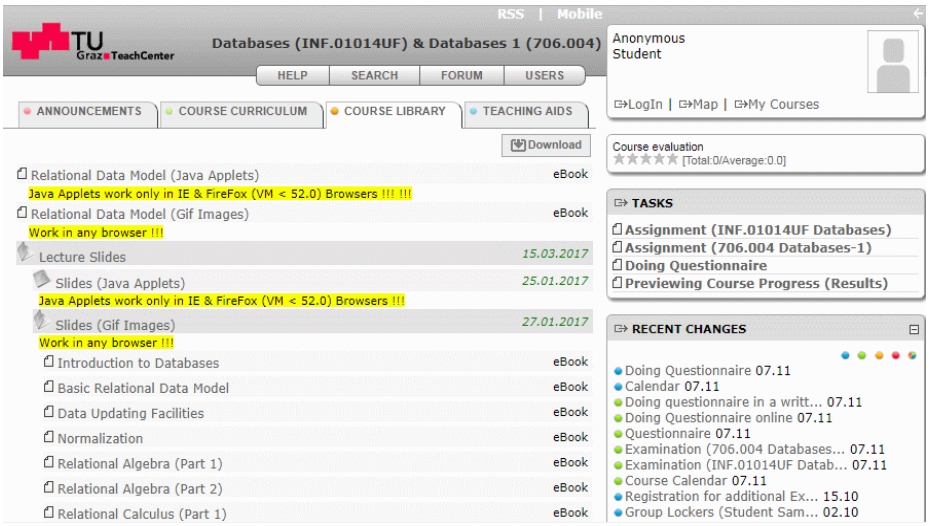

Рис. 2. WBT-Master электронного обучающего курса

Предлагаемая архитектура создает возможность для реализации пользовательского интерфейса в LMS. Традиционно пользовательский интерфейс основан на метафоре «Прыжок», когда пользователи нажимают на активные фрагменты HTML – так называемые «Якоря», и другое приложение активируется и визуализируется на экране клиента. Архитектура виртуального экрана позволяет использовать такие современные решения пользовательского интерфейса, как модель событий документа, динамические подсказки, плавающие окна и виртуальные всплывающие окна.

Таким образом, можно считать, что первый слой функционала системы разбирает файл описания курса и размечает экран пользователя на несколько виртуальных экранов. Виртуальные экраны представляют собой обычные HTML-объекты типа «DIV» с параметрами, обеспечивающими требуемое положение и свойства объектов.

Второй уровень базовой функциональности обеспечивается, когда драйверы и коммуникаторы (функции JavaScript) извлекаются с сервера, загружаются в виртуальную машину JavaScript и инициализируются на клиенте.

1.2. Язык описания курса

Учебный курс описывается в виде иерархии виртуальных экранов различных типов. Каждый экран представляет собой типизированный объект JSON:

```
Object Virtual Screen: объекта
{ Type: String;
VS_Name: String,
Driver: String,
Communicator: String,
Optional Offset_X: String,
Optional Offset_Y: String,
Optional Width: String
Optional Height: String,
Optional Members: [Array-of-Screens] },
```

Тип виртуального экрана определяет обучающее приложение, которое управляет экраном. Рассмотрим следующие основные типы:

Тип «Объявления» представляет собой список участников; обычно членами являются обычные HTML или текстовые документы. Все объявления отображаются в виде прокручиваемого списка.

Тип «Учебный план курса» представляет собой набор текстовых или других документов, которые визуализируются в виде раскрывающегося меню элементов. Обычно в этих документах описывается расписание, цели, предпосылки и требования к курсу.

Тип «Библиотека курса» представляет собой набор учебных документов, которые визуализируются в виде древовидной структуры, состоящей из папок и подпапок (см. рис. 2).

Тип «Учебные пособия» представляет собой экран, аналогичный экрану «Библиотека курса», но элементами такого контейнера «Учебные пособия» являются другие приложения. Например, «Викторина», «Загрузки пользователей» и т. д. являются объектами данных. Объекты, относящиеся к типам «Учебные пособия», инициируются как всплывающие экраны в верхней части основного экрана «Учебные пособия».

Помимо основных типов, существует множество дополнительных типов, таких как форум, чат, опрос мнения, отправка заданий, разработка проекта, викторина и др. Следует особо отметить, что список доступных типов постоянно расширяется по мере реализации новых алгоритмов. Так, в последнее время все современные облачные сервисы, такие как DropBox, Google Doc, MS One и т. д., были реализованы в виде новых типов для системы. Основные социальные сети также были включены в качестве особых типов виртуальных экранов.

Параметр «Драйвер» относится к определенной реализации выбранного сценария обучения. Следует обратить внимание, что может существовать несколько разных реализаций одного и того же сценария, например, несколько разных библиотечных реализаций.

Помимо драйверов алгоритмы связи (параметр «Коммуникатор») также должны быть реализованы в виде классов JavaScript, имеющих методы «SendMessage», «GetMessage», «Notify» и т. д.

Например, все драйверы могут запрашивать идентификатор курса на верхнем экране, запрашивать прогресс курса у некоторых драйверов, уведомлять другие драйверы о происходящих событиях и т. д.

2. Разработка драйверов

2.1. Обоснование выбора языка моделирования

Все драйверы реализованы в виде специальных классов данных. Все классы наследуют свойства (т. е. расширяют) абстрактный класс «Драйвер». Например,

```
abstract class Driver{
public readonly course_title: string;
private readonly course_id: string;
private readonly year: number;
public abstract render(): void;
}
class [driver_name]  extends Driver {
public render():void
{// main functionality of the driver goes here}}
```

Определение класса драйвера вместе со всеми остальными классами, которые необходимы для корректной работы, упакованы в текстовый файл [driver].js. Файлы HTML и JavaScript, представляющие курс на первом уровне архитектуры, создают все необходимые виртуальные экраны в виде объектов «DIV» и динамически загружают [driver].js с сервера.

```
var driver = document.createElement('script');
    driver.src = [driver_name] + '.js';
    driver.onload =  callBack;
    document.body.appendChild(driver);
```

Как только определение всех классов загружено, драйвер активируется с помощью метода render().

```
var callBack = function(){
oDriver = new [driver_name](courseId);
oDriver.render();
...}
```

Исходный контент и все элементы управления визуализируются внутри выделенного виртуального экрана. Вся дальнейшая функциональность реализуется посредством загружаемого вместе с классом драйвера программного обеспечения.

Коммуникатор разрабатывается и активируется схожим образом.

```
abstract class Communicator{
private readonly course_id: string;
public abstract render(): void;
public abstract sendMessage(string):void;
public abstract getMessage():string;
public abstract setEvent(string):string; }
```

Основное отличие состоит в том, что один и тот же коммуникатор может использоваться (и обычно используется) разными виртуальными экранами для обеспечения связи между драйверами. Коммуникация осуществляется следующими двумя способами:

– обмен текстовыми сообщениями;

– обработка внешних событий, которые запускаются на одном экране и обрабатываются на другом.

Поскольку драйверы загружаются динамически, то функциональность драйверов может быть даже изменена во время выполнения или если предстоит работать со специальными пакетами, распространяемыми через различные репозитории (например, Play Store для мобильных приложений). Такие возможности значительно способствуют эволюционному развитию системы.

2.2. Разработка курсов электронного обучения

Построение онлайн-курса обучения обычно следует нисходящему подходу:

Во-первых, инициируется конкретный курс, который получает идентификатор, название и общие свойства (дата открытия, дата закрытия, доступа и т. д.). Таким образом, курс можно рассматривать как «пустой» экран, на котором размещается лишь несколько экранов функциональных компонентов.

Во-вторых, преподаватель определяет набор виртуальных экранов, необходимых для реализации желаемого сценария обучения. В простейшем случае достаточно подэкранов «Объявления», «Учебный план курса» и «Библиотека курса». Все экраны, необходимые для курса, описываются как виртуальные экраны определенного типа и вставляются в определение экрана «Курс» как новые элементы.

Наконец, некоторые обучающие материалы добавляются в курс с помощью виртуальных экранов, определенных выше. После создания учебного курса виртуальные экраны можно вставлять, изменять или удалять гибким способом и от пользователя не требуется каких-либо специальных знаний об Интернете и программировании. Следует обратить внимание, что навигационная структура органично встраивается в отдельные виртуальные экраны. Это важно, поскольку экран или приложение можно гибко повторно использовать в другом курсе электронного обучения.

Если курс предназначен для работы с помощью различных устройств конечного пользователя, то необходимо реализовать несколько описаний курса, чтобы определить разные топологии виртуальных экранов и, возможно, разные драйверы для разных клиентов.

Заключение

В заключение следует отметить, что перечисленные ниже особенности WBT-Master заметно отличают данную систему от других существую-

щих систем управления обучением. В предложенной системе использованы новые архитектурные решения. Система может адаптироваться под требования преподавателя и под особенности персональных устройств конечного пользователя.

Разработанная система предлагает новую парадигму структурирования курса, которая выходит за рамки простого структурирования курса с помощью гиперссылок. Использование виртуальных экранов в качестве основной парадигмы разработки курса позволяет успешно применять предложенную систему преподавателям и студентам, слабо или почти не владеющим Интернетом и компьютерными технологиями. Предложенная система использует функциональные возможности нескольких облачных сервисов. Система поддерживает несколько сценариев электронного обучения, которые позволяют реализовать такие современные парадигмы, как обучение на практике, перевернутое обучение, совместное решение проблем и т. д. Все сценарии реализованы как драйверы для виртуальных экранов.

Список литературы

1. Dietinger T., Maurer H. GENTLE – General Network Training and Learning Environment // Proc. of ED-MEDIA98 / ED-TELECOM 98. – Freiburg, 1998. – Pp. 274–280.

2. Ebner M., Scerbakov N., Maurer H. New features for eLearning in higher education for civil engineering // Journal of Universal Science and Technology of Learning. – 2006. – Vol. 1, No. 1. – Pp. 93–106.

3. Ebner M., Scerbakov N., Tsang P., Holzinger A. EduPunks and learning management systems – conflict or chance? // In: Proceedings of International Conference on Hybrid Learning (IHCL 2011). Lecture Notes in Computer Sciences (LNCS 6837). – Springer, 2011. – Pp. 224 – 238.

4. Scerbakov A., Ebner M., Scerbakov N. Using cloud services in a modern learning management system // Journal of Computing and Information Technology. – 2015. – Vol. 23(1). – Pp. 75–86.

5. Scerbakov N. TU Graz Teach-Center [Electronic resource]. – URL: http://coronet.iicm.tugraz.at/wbtmaster/welcome.html (access date: 01.05.2024).

6. Nagler W., Ebner M., Scerbakov N. Reading and learning with any device university content for e-books and e-Readers // In: World Conference on Educational Multimedia, Hypermedia and Telecommunications. – 2011. – Pp. 1775–1782.

7. Nagler W., Wiesenhofer K., Ebner M., Scerbakov N. E-books and e-reader devices for e-learning in higher education: The case of Graz University of Technology // In: E-Books and E-Readers for E-Learning. – 2012. – Pp. 1–24.

8. Nagler W., Wiesenhofer K., Ebner M., Scerbakov N. E-books für die Universität // In: Handbuch E-Learning – Expertenwissen aus Wissenschaft und Praxis – Strategie, Instrumente, Fallstudien. – 2012. – Pp. 1–4.

9. Kappe F., Scerbakov N. File uploading scenarios in a modern learning management system // In: Proceedings of 9th International Conference on Education and New Learning Technologies, Barcelona, Spain, July 3–5, 2017. – Pp. 4904–4909.

10. Kappe F., Scerbakov N. Object-oriented architecture of a modern learning management system // In: Proceedings of 9th International Conference on Education and New Learning Technologies, Barcelona, Spain, July 3–5, 2017. – Pp. 4910–4916.

11. Kappe F., Scerbakov N. Quantitative evaluation of university lecturing // In: Proceedings of 20th International Conference on Interactive Collaborative Learning, Budapest, Hungary, September 27–29, 2017. – Pp. 335–340.

12. Scerbakov N., Schukin A., Sabinin O. Plagiarism detection in SQL student assignments // In: Proceedings of 20th International Conference on Interactive Collaborative Learning, Budapest, Hungary, September 27–29, 2017. – Pp. 321–326.

13. Schukin A., Pak V., Rezedinova E. E-learning course on mobile web applications: comparison of learning // INTED2024 Proceedings. – 2024. – Pp. 5609–5614.

14. Pak V., Scerbakov N., Rezedinova E. Feedback in e-learning: comparative study // INTED2024 Proceedings. – 2024. – Pp. 5615–5619.

15. Schukin A., Scerbakov N., Rezedinova E. A data model for semi-structured data // AIP Conf. Proc. February 21, 2024. – Vol. 3063(1). – Paper 070009. – DOI: https://doi.org/10.1063/5.0200874.

16. Scerbakov N., Schukin A., Rezedinova E. Architecture of modern e-learning management system // 2023 4th International Conference on Communications, Information, Electronic and Energy Systems (CIEES), Plovdiv, Bulgaria, 2023. – Pp. 1–5. – DOI:10.1109/CIEES58940.2023.10378809.