

УДК 378; 519.711.2

Кафедральная система управления обучением в ИТ для МГТУ им. Н.Э. Баумана

Бекасов Д. Е.^{1,2,*}

[*bekas@bmstu.ru](mailto:bekas@bmstu.ru)

¹МГТУ им. Н.Э. Баумана, Москва, Россия

²ООО «ПолисСофт», Москва, Россия

В рамках международного научного конгресса "Наука и инженерное образование. SEE-2016", II международная научно-методическая конференция «Управление качеством инженерного образования. Возможности вузов и потребности промышленности» (23-25 июня 2016 г., МГТУ им. Н.Э. Баумана, Москва, Россия).

В статье рассмотрен опыт формулирования требований, проектирования и разработки кафедральной системы управления обучением для ИТ-образования в МГТУ им. Н. Э. Баумана. Система решает задачи автоматизации специфичных для ИТ-образования процессов на основе профессиональных инструментов, таких как системы контроля версий. Такой подход не только позволяет решить задачу автоматизации, но и подготавливает студентов к будущей профессиональной деятельности.

Ключевые слова: МГТУ, сервис-ориентированная архитектура (COA), ИТ-образование, LMS, Git, GitLab

Введение

Сегодня в высшем образовании преобладают две образовательных модели – традиционная и дистанционная. Традиционная, вузовская модель предполагает очное общение студентов и преподавателей в реальном времени. При дистанционном образовании участники могут не находиться в одном месте в одно и то же время.

Всё большую популярность набирает смешанный вариант – традиционная модель с автоматизированными бизнес-процессами взаимодействия студентов и преподавателей [1]. Это делает высшее образование более доступным, удобным, масштабируемым и персонализированным. А значит, повышает качество образовательной среды. Основные образовательные процессы, которые можно выделить для автоматизации, - это работа с учебными материалами, общение между студентами и преподавателями, работа с расписаниями, уведомления о сроках заданий и разных кафедральных и университетских событиях, выдача и сдача учебных заданий, учет успеваемости. Это типовые процессы, присутствующие в любом образовательном учреждении. Однако каждый ВУЗ, каждый факультет и каждая кафедра имеют свои особенности. Дальнейшие рассуждения проведены с точки зрения автоматизации образовательных процессов кафедры «Программное обес-

печение ЭВМ и ИТ» факультета «Информатика и системы управления» МГТУ им. Н. Э. Баумана.

1. Анализ предметной области

Образовательные процессы в ИТ во многом схожи с классическими образовательными процессами, но имеют специфику. В соответствии с «русским методом обучения ремеслам» центральное место в подготовке инженера занимает практика, максимально приближенная к будущей профессиональной деятельности [2]. В ИТ-образовании это место занимает большой прикладной блок по разработке программного обеспечения. Поэтому под учебным заданием в общем случае можно понимать разработку программного обеспечения. В связи с этим можно выделить три специфичные для ИТ-образования активности.

1. Хранение исходного кода и его изменений. Как в краткосрочной перспективе для целей выполнения и сдачи учебных заданий, так и в долгосрочной – для организации архива студенческих работ.
2. Обсуждение исходного кода студентом и преподавателем. Это основной элемент обучения программированию и обязательный этап сдачи любого задания.
3. Проверка работы (тестирование) программного обеспечения. Второй обязательный этап сдачи любого учебного задания. Разработанное программное обеспечение должно решать поставленную задачу. Преподаватель убеждается в этом, проверяя задачу на заготовленных тестах.

Перечисленные активности не являются специфическими для образовательной среды и входят, пусть и в ином виде, в коммерческий процесс разработки программного обеспечения. Для организации хранения кода и его изменений и обеспечения командной работы над ним, используются системы контроля версий (git, svn, mercurial) [3]. В этих системах программный код хранится в репозиториях, история его изменений – в последовательности фиксаций, а версионность организуется с помощью набора веток – параллельных последовательностей таких фиксаций. Для организации совместной работы над кодовыми репозиториями и удаленного просмотра и обсуждения кода используют веб-интерфейсы систем контроля версий (GitLab, TFVC) [4]. Контроль качества автоматизируется благодаря подходу «Continuous Integration», предполагающему автоматическую компиляцию и сборку программ из репозитория и их тестирование на заранее заготовленных сценариях (GitLab CI, Jenkins) [5].

Использование рассмотренных профессиональных инструментов, с одной стороны, позволяет решить проблему автоматизации рассматриваемых процессов, а с другой – готовит студентов к непосредственной профессиональной деятельности.

Так же необходимо отметить, что в МГТУ уже существует информационная система, автоматизирующая все административные и часть учебных процессов, - это «Электронный Университет» [6]. Помимо неё, в МГТУ внедрена единая система аутентификации (CAS) [7] и единая служба каталогов (LDAP). Так же существует официальный сайт МГТУ, портал библиотеки МГТУ, проект «Персональные страницы МГТУ», и «Элек-

тронная образовательная система» на основе Moodle, внедренная на нескольких кафедрах. Поэтому информационная система, автоматизирующая рассмотренные выше процессы должна быть интегрирована в существующую информационную инфраструктуру.

2. Существующие решения

Рассматриваемую задачу решает такой класс информационных систем, как системы управления обучением (LMS). Рынок LMS стремительно и неуклонно растёт. Согласно отчёту о рынке eLearning специализированной на образовательных технологиях компании Docebo, мировой рынок LMS вырастет на 23.17% в период между 2017 и 2018 годами, а к 2016 году оценивается в 38,3 миллиардов долларов США [8]

Первой стандартной моделью, регламентирующей работу LMS, оказалась SCORM (Sharable Content Object Reference Model, «образцовая модель объекта содержимого для совместного использования») — сборник спецификаций и стандартов, разработанный для LMS в 1999 году. В июне 2006 года SCORM стал стандартом в США для всех разработок в области электронного обучением. Стандарт регламентирует структуру учебных блоков и взаимодействие с системами управления обучением через программный интерфейс [9]. В настоящее время стандарт устарел и на его смену пришел Tin Can API [10].

Одной из самых распространенных LMS в мире является Moodle - среда дистанционного обучения с открытым исходным кодом [11]. Основными возможностями системы являются организация дистанционных курсов в различных форматах (в том числе SCORM), инструменты для взаимодействия пользователей, расширение функциональных возможностей с помощью дополнительных модулей. К недостаткам системы можно отнести отсутствие технической документации на русском языке, каких-либо инструкций и рекомендаций по пользованию, невозможность использовать систему частично, высокая универсальность системы, что снижает ее способность настройки под специфические процессы.

Blackboard – так же очень популярная LMS. Полностью платная, обладает большим количеством возможностей, постоянно обновляется, имеет проработанный модуль аналитики. Из недостатков – закрытость, сложность освоения и нестабильность работы [12].

Еще одной LMS, достойной упоминания, является eFront. Даже несмотря на то, что она больше подходит для бизнес-структур или небольших учебных центров, нежели для образовательных организаций и ориентирована на чисто дистанционное, а не смешанное обучение. Система обладает возможностями гибкой настройки функционала и инструментарием для индивидуальной подстройки образовательных траекторий. Из недостатков – небольшое количество модулей и инструментов, отсутствие возможности подстроится под специфический процесс [13].

Из всех рассмотренных LMS (см. табл. 1) для решения рассмотренных задач могла бы подойти только Moodle при условии ее значительной доработки. Но стоимость ее адаптации под озвученные образовательные процессы в ИТ так высока, что было принято реше-

ние о разработке собственной расширяемой LMS, изначально поддерживающей кафедральные бизнес-процессы. Центральная идея проекта – использование системы контроля версий для реализации процесса создания, выдачи, выполнения, проверки и приема учебных заданий. Другая важная составляющая – полная интеграция с информационными системами МГТУ («Электронным университетом», единой системой аутентификации, Moodle) и поддержка стандартов LMS (SCORM, Tin Cat API).

Таблица 1. Сравнение популярных LMS

	Moodle	Blackboard	eFront
Бесплатность	Да	Нет	Да
Быстрая настройка	Нет	Да	Да
Автоматизированный процесс выдачи/сдачи заданий	Нет	Нет	Нет
Система коммуникации между участниками	Да	Да	Нет
Система уведомлений	Да	Да	Нет
Расширение функциональности	Да	Нет	Да
Поддержка специфичных для IT образовательных процессов	Нет	Нет	Нет

3. Жизненный цикл учебного задания

Был предложен следующий вариант формализации жизненного цикла задания в терминах систем контроля версий. Преподаватель выдает вариант учебного задания студенту. При этом создается отдельный репозиторий для выполнения выданного варианта задания. В этом репозитории создается одна ветка (ветка преподавателя или master-ветка в терминологии git-flow), которая содержит задание, дополнительные учебные материалы и примеры. Когда студент приступает к выполнению задания, для него из ветки преподавателя создается отдельная ветка (ветка студента или develop-ветка в терминологии git-flow). Студент работает в своей ветке до тех пор, пока не посчитает задание выполненным. При этом, он может пользоваться системой контроля версий в полном объеме: делать фиксации изменений и создавать другие рабочие ветки. После того, как задание будет выполнено, оно может быть отправлено на проверку с помощью так называемого «запроса на слияние» (merge-request). Преподаватель получит уведомление об этом запросе и сможет просмотреть и прокомментировать все выполненные студентом изменения, а также просмотреть результаты автоматического тестирования. После этого, запрос может быть отклонен или принят. В первом случае задание отправляется на доработку до следующего «запроса на слияние». Во втором – задание считается выполненным и фиксируется в ветке преподавателя. Студент всегда сможет вернуться к нему и просмотреть все свои изменения и обсуждения с преподавателем. Кроме того, у преподавателя появляется удобный инструмент оценки схожести сдаваемых работ (для любых двух работ можно просмотреть их разницу). Описанный цикл выполнения задания продемонстрирован на рисунке.

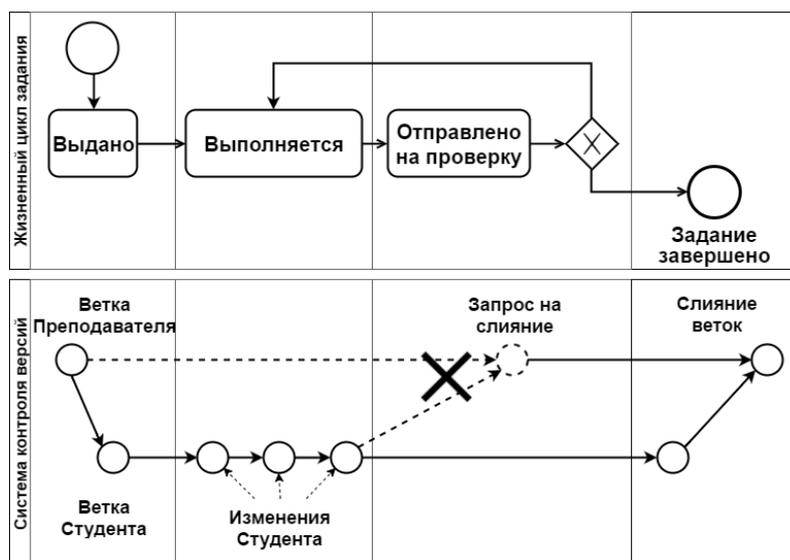


Рис. 1. Формализация жизненного цикла учебного задания на основе системы контроля версий

4. Архитектура

В качестве архитектуры проекта была выбрана сервис-ориентированная архитектура [14]. Информационная система состоит из клиента, набора независимых сервисов и мастер-сервиса, предоставляющего прикладной интерфейс доступа к сервисам системы. Такая архитектура, в противовес архитектуре с монолитным сервером, обеспечивает гибкость, расширяемость и модифицируемость системы. Кроме этого, сильно возрастает отказоустойчивость: при выходе из строя одного из сервисов, другие останутся работать. Так же этот подход обеспечивает горизонтальную масштабируемость системы: сервисы могут работать на разных физических узлах сети.

При выбранном архитектурном подходе используется «толстый» клиент. Помимо положительного пользовательского опыта, одним из преимуществ «толстого» клиента является независимость его реализации от сервера. Клиент и сервер связываются по REST API (Representational state transfer), что позволяет разрабатывать разные версии клиентов для разных платформ при наличии одного сервера. Основным клиентом для разрабатываемой информационной системы является Web-клиент, спроектированный как компонентно-ориентированное одностраничное Web-приложение [15]. Архитектура представлена на рисунке.

Точкой входа в систему является мастер-сервис, который предоставляет удобный прикладной интерфейс доступа ко всем остальным сервисам. Он реализует логику работы приложения. Сервис аутентификации – это сервис-адаптер к существующему сервису аутентификации МГТУ, работающему по технологии CAS. Сервис интеграции с информационными системами МГТУ обеспечивает своевременное получение необходимых для функционирования системы данных из информационных систем МГТУ, в первую очередь, из системы «Электронный университет», официального сайта МГТУ, сайта библиотеки МГТУ, службы активных каталогов МГТУ. Это информация о структуре университета, контингенте ВУЗа, списках преподавателей, учебных планах, текущей успеваемости,

расписании и публикациях авторов. Сервис получает, обрабатывает, связывает, разрешает возможные коллизии и конвертирует данные во внутреннюю схему данных. Остальные сервисы получают требуемые данные уже от него. Сервис подсистемы уведомлений обеспечивает гарантированную отложенную приоритезированную доставку уведомлений о любых происходящих в системе событиях (изменение расписания, выдача и сдача учебных заданий и пр.). Сервис спроектирован таким образом, чтобы поддерживать все существующие каналы доставки (e-mail, sms, мессенджеры, веб-уведомления, push-уведомления). Сервис приема и выдачи заданий реализует описанный выше жизненный цикл задания. Сервисом добавления и выдачи учебных материалов реализуется функциональность хранилища учебных материалов. Через него осуществляется добавление к учебным планам дополнительных материалов. Сервис интеграции с системой контроля версий с помощью очереди отложенных задач управляет репозиториями в соответствии с жизненным циклом заданий и по предоставляет интерфейс доступа к системе управления репозиториями для всех остальных сервисов системы. Сервис управления календарями и событиями обеспечивает работу календаря пользователя: заполняет его персональным расписанием, добавляет кафедральные и университетские события и с помощью подсистемы уведомлений уведомляет об этом пользователей.

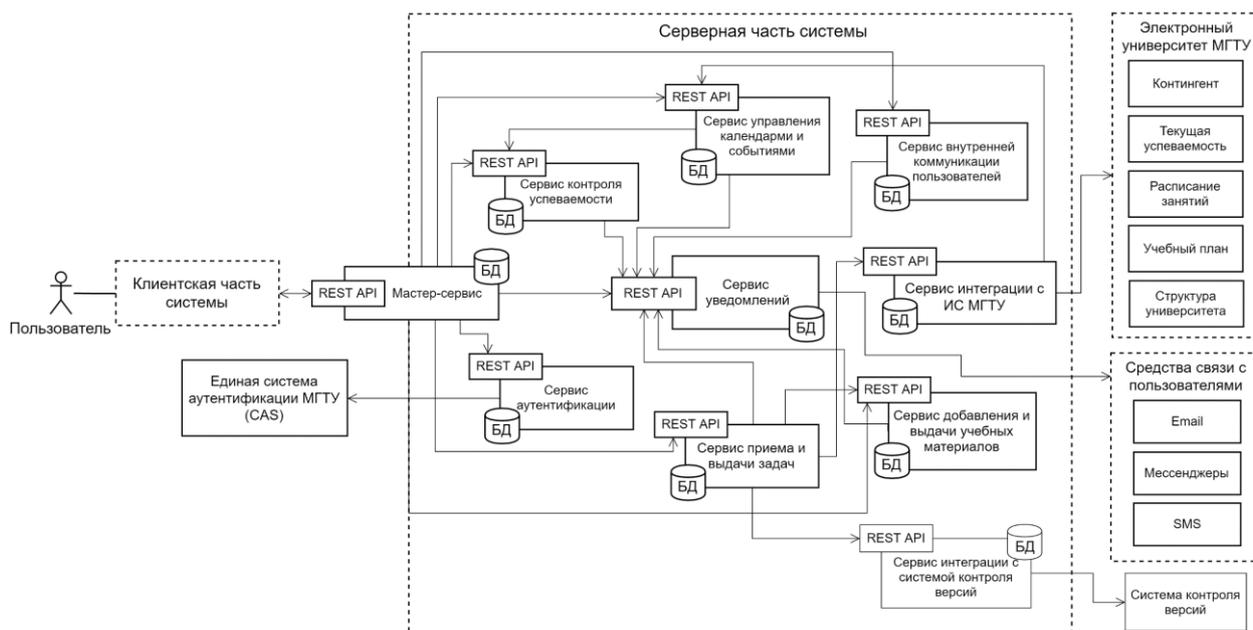


Рис. 2. Архитектура кафедральной LMS

Заключение

Разработка LMS ведется силами кафедры. Формулирование требований и координация проекта осуществляется преподавателями, а проектирование и разработка - студентами в рамках курсовых и дипломных проектов. Помимо практической пользы, проект несет и важный методический смысл: студенты приобретают опыт участия в большом проекте, которым сами пользуются. Кроме того, проект имеет потенциал превращения в самостоя-

тельный отечественный продукт. Есть планы по выпуску LMS под открытой лицензией и привлечению к его разработке участников из других ВУЗов.

К моменту написания статьи готова первая версия LMS, в которой реализована подсистема выдачи и сдачи учебных заданий, подсистема календарей и уведомлений, интеграция с информационными системами МГТУ, личные страницы студентов и преподавателей. Проводится тестирование на ограниченном контингенте пользователей. Ввод в эксплуатацию планируется в осеннем семестре этого года. Однако уже можно сделать вывод, что принятые конструкторские решения, связанные в том числе с использованием системы контроля версий для реализации жизненного цикла задания – верные и перспективные.

В качестве дальнейшего развития предполагается более полная интеграция с существующими информационными системами МГТУ (в том числе на запись изменений), поддержка открытого стандарта LMS SCORM, поддержка мобильных клиентов и расширение средств коммуникации пользователей.

Список литературы

- [1]. Bruff D.O., Fisher D.H., McEwen K.E, Smith B.E. Wrapping a MOOC: Student Perceptions of an Experiment in Blended Learning // Journal of Online Learning and Teaching. 2013. Vol. 9. No. 2.
- [2]. Русская система обучения ремеслам. Том I. М.: НОЦ «Контроллинг и управленческие инновации»; ООО «Высшая Школа Инженерного Бизнеса». 2015. 238 с.
- [3]. Scott Chacon, Ben Straub. Pro Git. 2nd ed. Publisher: Apress. 2014. 426 с. DOI: 10.1007/978-1-4842-0076-6
- [4]. Tools for modern developers. GitLab unifies chat, issues, code review, CI and CD into a single UI // Site: GitLab. Режим доступа: <https://about.gitlab.com/> (дата обращения 01.07.2016)
- [5]. Martin Fowler. Continuous Integration // The Next Web. 01.05.2006. Режим доступа: <http://martinfowler.com/articles/continuousIntegration.html> (дата обращения 01.07.2016)
- [6]. Балдин А.В. Информационные технологии в управлении университетом / Федеральное агентство по образованию, Приоритетный нац. проект "Образование", МГТУ им. Н. Э. Баумана. М.: Изд-во МГТУ им. Н. Э. Баумана. 2007. 46 с.
- [7]. Рыбальченко М. А. Архитектура единой службы аутентификации, авторизации и учета пользователей информационных ресурсов университета // Молодежный научно-технический вестник. Электрон. журн. МГТУ им. Н.Э. Баумана. 2014. №1. Режим доступа: <http://sntbul.bmstu.ru/doc/681183.html> (дата обращения 01.07.2016)
- [8]. E-Learning Market Trends and Forecast 2014 – 2016. Report. Docebo. 2014. 47 p. Режим доступа: <https://www.docebo.com/landing/contactform/elearning-market-trends-and-forecast-2014-2016-docebo-report.pdf> (дата обращения 01.07.2016)
- [9]. Sharable Content Object Reference Model (SCORM®) 2004 2nd Edition Overview // Advanced Distributed Learning. 2006. Режим доступа: <http://www.eife->

l.org/publications/standards/elearning-standard/scormoverview/english_release (дата обращения 01.07.2016)

- [10]. Downes A., Shahrazad A., Smith R. Sharing between LRSs: a collaborative experiment in practical interoperability. 2015. Режим доступа: <https://tincanapi.com/wp-content/uploads/2015/03/whitepaper.pdf> (дата обращения 01.07.2016)
- [11]. Hicks K. Understanding The Top Learning Management Systems // Edudemic. 2016. Режим доступа: <http://linkis.com/НрFnD> (дата обращения 01.07.2016)
- [12]. Bradford P., Porciello M, Balkon N. The blackboard learning system: The Be All and End All in Educational Instruction? // The Journal of Educational Technology Systems. 2007. Vol. 35. Is. 3. P. 301-314.
- [13]. Kor B., Tanrikulu Z. Evaluation of Learning Management Systems with Test Tools. // ED-MEDIA 2008. World Conference on Educational Multimedia, Hypermedia and Telecommunications / Editors: Joseph Luca; Edgar R. Weippl. 2008. P. 5261–5266.
- [14]. Kress J., Maier B., Normann H., Schmeidel D., Schmutz G., Trops B., Utschig C., Winterberg T. SOA and User Interfaces // Oracle Technology Network Articles. October, 2013. Режим доступа: <http://www.oracle.com/technetwork/articles/soa/ind-soa-ui-2028423.html> (дата обращения 01.07.2016)
- [15]. Klaus Nygard. Single page architecture as basis for web applications: abstract of master's thesis Degree Programme in Computer Science and Engineering. Aalto University School of Science. 2015. P. 65.