УДК 004.4 (06)

ПОСТРОЕНИЕ ЭФФЕКТИВНОЙ АРХИТЕКТУРЫ «ОБЛАЧНЫХ» ВЕБ-ОРИЕНТИРОВАННЫХ БИЗНЕС-ПРИЛОЖЕНИЙ

[2]Братчиков И.А., [1]Шмидт И.А., [2]Елисеев А.С., [2]Анисимова Т.В.

[1]Пермский национальный исследовательский политехнический университет, Пермь;
[2]ГК «ИВС», Пермь, e-mail: shmidt@msa.pstu.ru

В статье представлены результаты исследований методов построения архитектуры генерируемых платформой Flexberry «облачных» веб-ориентированных приложений. Рассмотрен понятийный аппарат и терминология применительно к созданию архитектуры приложения. Произведено сравнение основных типов архитектуры приложения с разными вариантами компоновки. Обосновывается применение архитектуры Single-page Application, характеризующейся тем, что с сервера загружается HTML-страница, представляющая собой JavaScript-приложение, которое строит визуальную часть, используя веб-интерфейс. Сформулированы критерии для оценки функционирования веб-приложений, построенных с использованием разных архитектур. Критерии были сформулированы авторами с двух точек зрения: разработчика и заказчика (пользователя). Результаты сравнительного анализа методов построения веб-ориентированных бизнес-приложений сведены в таблицу. По результатам сравнения авторами было принято решение применить для создаваемых приложений архитектуру на основе Single-page Application.

Ключевые слова: бизнес-приложение, облачные приложения, архитектура приложения, программная платформа

BUILDING AN EFFECTIVE ARCHITECTURE OF CLOUD WEB-BASED BUSINESS APPLICATIONS

[2]Bratchikov I.A., [1]Shmidt I.A., [2]Eliseev A.S., [2]Anisimova T.V.

[1]Perm National Research Polytechnic University, Perm;
[2]GC «ICS», Perm, e-mail: shmidt@msa.pstu.ru

The article presents the results of research of methods for constructing the architecture generated by the platform Flexberry «cloud» web-based applications. Considered the conceptual apparatus and terminology as applied to the creation of the application architecture. The article made a comparison of the main types of architecture of application with different layout options. Explains the application architecture, Single-page Application, characterized by the fact that the server loads the HTML page representing a JavaScript application that is building a visual part using the web interface. Were formulated criteria to assess the performance of web applications built using different architectures. A comparative analysis of methods of building. Criteria formulated from two points of view: the developer and the customer (user). Web-based business applications in the table. Comparison results the authors decided to use for the generated application architecture based on Single-page Application.

Keywords: business application, cloud applications, application architecture, software platform

В [8] был рассмотрен метод построения веб-ориентированных бизнес-приложений, основанный на применении метамоделей с возможностью модификации исходного кода. Выполненные работы позволили реализовать инструментарий создания веб-ориентированных бизнес-приложений доступный непрофессиональным программистам. Создаваемые бизнес-приложения соответствуют критериям промышленной разработки, но ориентированы в первую очередь на функционирование в «классическом» хостинге, а не в «облачной» среде. В первую очередь это связано с задачей горизонтального масштабирования приложений [1, 3]. Данная тематика послужила основой для новых исследований, которые должны дать ответ на вопрос: какие требуются доработки для полноценной поддержки «облачной» среды со стороны создаваемых веб-приложений?

Кроме вопросов масштабирования приложений в последнее время важную роль приобрели вопросы импортозамещения. С целью обеспечения защиты инвестиций потенциальные клиенты платформы разработки бизнес-приложений в первую очередь ориентируются на отечественные решения либо на решения с открытым исходным кодом и со свободной лицензией (свободное программное обеспечение, СПО). Реализация всего многообразия возможностей силами одной отечественной компании – очень трудоёмкая задача, поэтому актуальным вариантом является выбор в пользу применения сторонних СПО-решений. Отсюда возникает новое требование: в платформе разработки должна быть возможность создавать решения, основанные на свободном ПО. Это позволит, в свою очередь, выпускать реализованные приложения также под свободной лицензией.

Требуется провести анализ вариантов изменения архитектуры генерируемых веб-приложений. Целью анализа является

нахождение оптимального распределения логики приложения между его функциональными компонентами для обеспечения требований по поддержке «облачной» среды исполнения (лёгкой масштабируемости) и возможности создания СПО [2, 6, 7].

Наиболее систематизированное и полное изложение понятийного аппарата и терминологии применительно к созданию архитектуры приложения приведено в руководстве [4]. Самый общий вид архитектуры приложения, компоненты которого сгруппированы по функциональным областям, как организации или структуры системы, где система представляет набор компонентов, выполняющих определенную функцию или набор функций, представлен на рис. 1.

Функциональные области приложения разделяются на многослойные группы (уровни). Представленный вид архитектуры описывает функции, слои и уровни, которые могут присутствовать в приложении. Эффективное разделение функциональности достигается за счет применения многослойного архитектурного стиля, который позволяет отделить логику представления от бизнес-логики и логики доступа к данным. Многослойная архитектура представляет систему как единое целое, обеспечивая при этом достаточно деталей для понимания ролей и ответственностей отдельных слоев и отношений между ними. Функциональность каждого слоя объединена общей ролью или ответственностью. Слои слабо связаны, и между ними осуществляется явный обмен данными. Правильное разделение приложения на слои помогает поддерживать строгое разделение функциональности, что в свою очередь обеспечивает гибкость, а также удобство и простоту обслуживания.

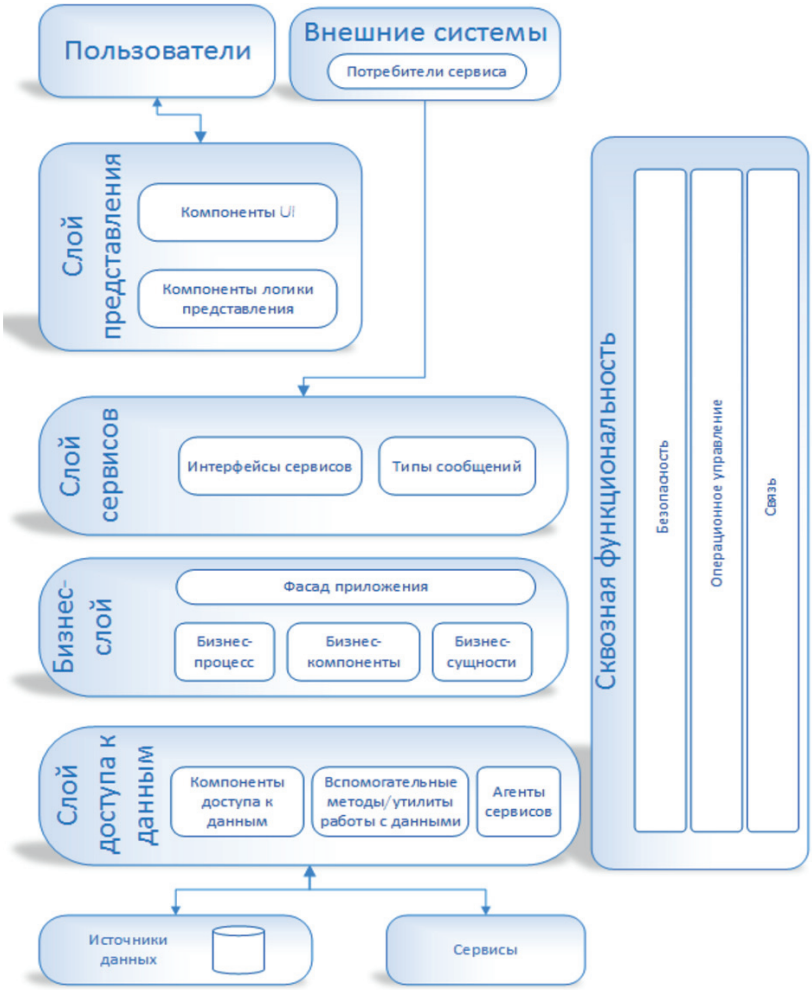

Рис. 1. Общий вид архитектуры приложения (слои, компоненты, сервисы)

Помимо разделения функций на слои, архитектура описывает разложение дизайна на отдельные функциональные или логические компоненты и сервисы, предоставляющие четко определенные интерфейсы, содержащие методы, события и свойства. Компоненты проектируются с минимальными зависимостями от других компонентов и предоставляют интерфейсы, позволяющие вызывающей стороне использовать их функциональность, не раскрывая при этом детали внутренних процессов, внутренние переменные или состояние.

Чтобы понять, как масштабируется всё приложение, обратим внимание на то, как каждый отдельный слой в архитектуре поддерживает масштабирование. Под масштабированием будем рассматривать возможности обслуживания пользователей со стороны приложения при существенном приросте их количества (пользователей, одновременно работающих с приложением). Конкретная реализация архитектуры сильно влияет на то, как увязаны между собой слои, поэтому рассмотрим основные типы архитектуры с разными вариантами компоновки.

Метод построения архитектуры на основе Server-side HTML

Server-side HTML – это самая распространенная архитектура, именно она используется в веб-приложениях, сгенерированных платформой Flexberry первого поколения. Главный принцип работы приложения заключается в том, что в ответ на каждый запрос клиента сервер генерирует HTML-контент и возвращает его клиенту как полноценную HTML-страницу, как это показано на рис. 2.

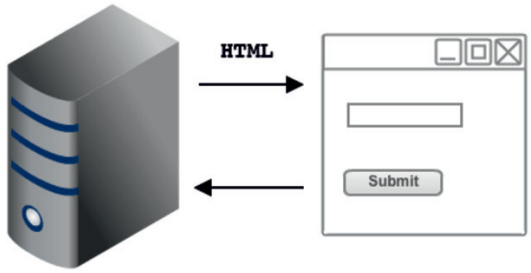

Рис. 2. Архитектура Server-side HTML [7]

Рассмотрим данную архитектуру относительно всех слоёв. Схема взаимодействия слоев приведена на рис. 3.

В рамках одного запроса к серверу происходит вызов каждого слоя в определённой последовательности, это привносит свои особенности. Данное решение, с одной стороны, позволяет достаточно гибко управлять выходными данными, поскольку все слои объединены контекстом одного запроса, но, с другой стороны, вопрос масштабирования для данной архитектуры заключается в увеличении числа веб-серверов, при этом приложение требуется изменить таким образом, чтобы состояние приложения было в едином месте и было доступно всем веб-узлам. Также стоит отметить, что количество внутренних операций веб-сервера на один запрос к нему в данной архитектуре одно из самых больших, что в целом негативно сказывается на требуемом количестве серверных ресурсов.

Ввиду того, что архитектура подразумевает выполнение большого числа операций на каждый запрос, уровень тестируемости данного решения невелик. Требуется обеспечивать проверку работы как каждого отдельного слоя, так и их взаимодействия, что не всегда технически легко реализуется.

Таким образом, пользователь каждый раз получает всю страницу с сервера. С одной стороны, это даёт полный контроль и обеспечивает более высокий уровень безопасности приложения, с другой стороны, пользователю приходится ожидать загрузки одних и тех же данных, что негативно сказывается на реактивности системы. Из плюсов можно выделить то, что поисковые роботы хорошо работают с таким типом страниц [9].

Данный вариант архитектуры наиболее близок разработчикам, которые хорошо умеют использовать серверное программирование и не очень сильны в клиентских технологиях (HTML, JavaScript, CSS). Производительность работы таких разработчиков в условиях этой архитектуры будет достаточно высокой.

Метод построения архитектуры на основе Single-page Application

Метод построения архитектуры на основе Single-page Application (SPA) характеризуется тем, что с сервера загружается HTML-страница, представляющая собой JavaScript-приложение, которое строит визуальную часть и, используя определенный веб-интерфейс, получает от сервера бизнес-данные, т.е. часть функциональности, обеспечивающая логику представления, переносится на клиентскую сторону. В качестве программного веб-интерфейса предполагается использование открытого веб-протокола для запроса и обновления данных. Схема работы приложения приведена на рис. 4.

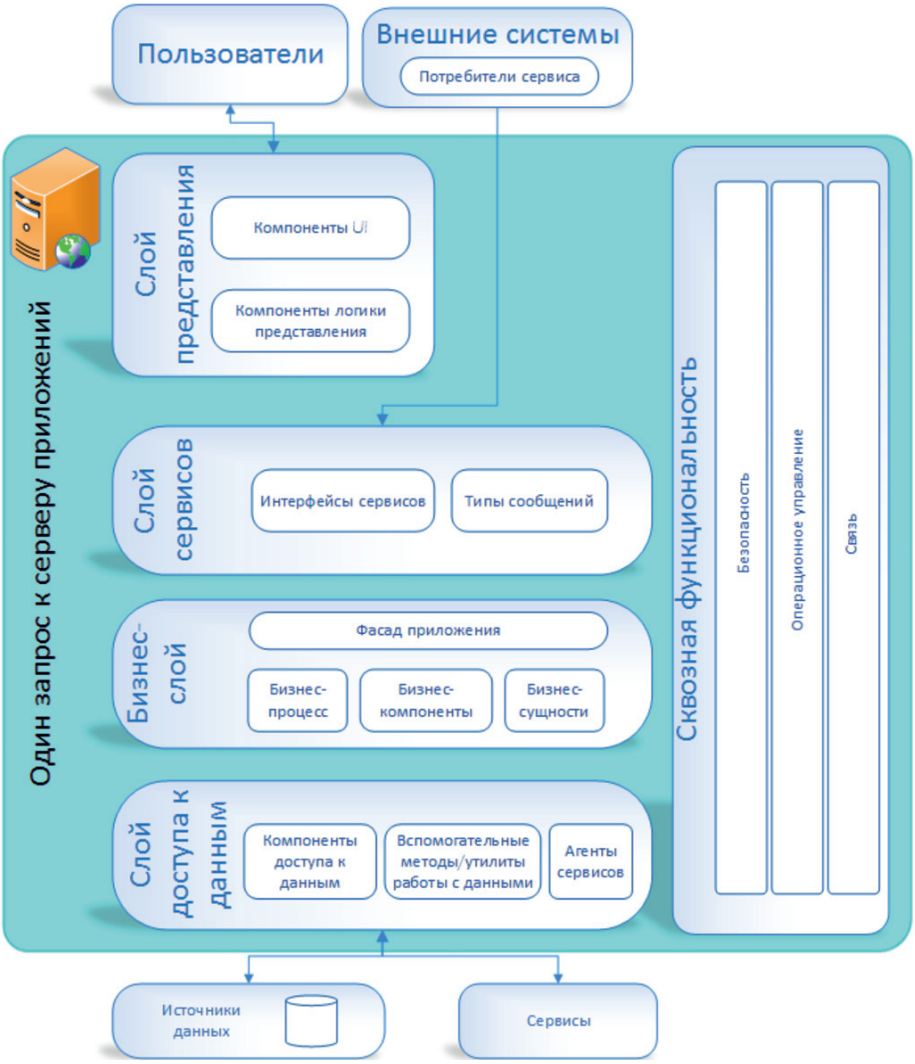

Рис. 3. Архитектура Server-side HTML в разрезе слоёв

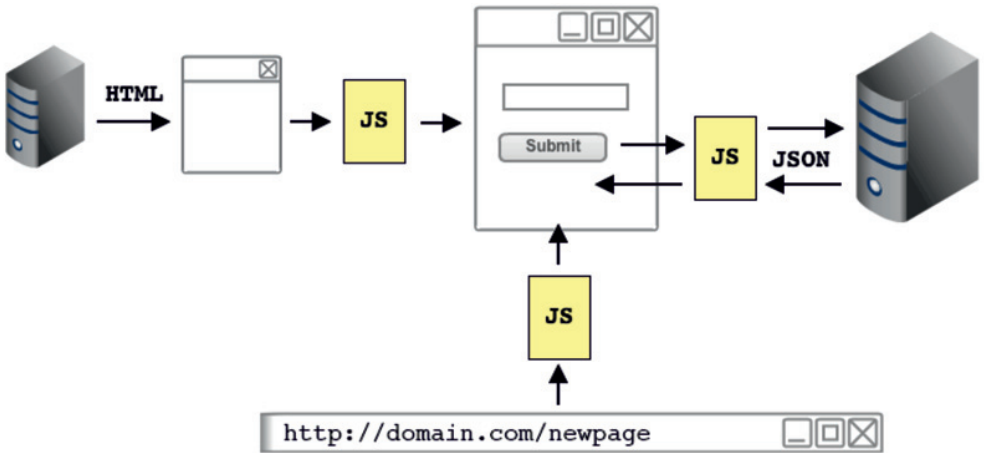

Рис. 4. Архитектура Server-side HTML

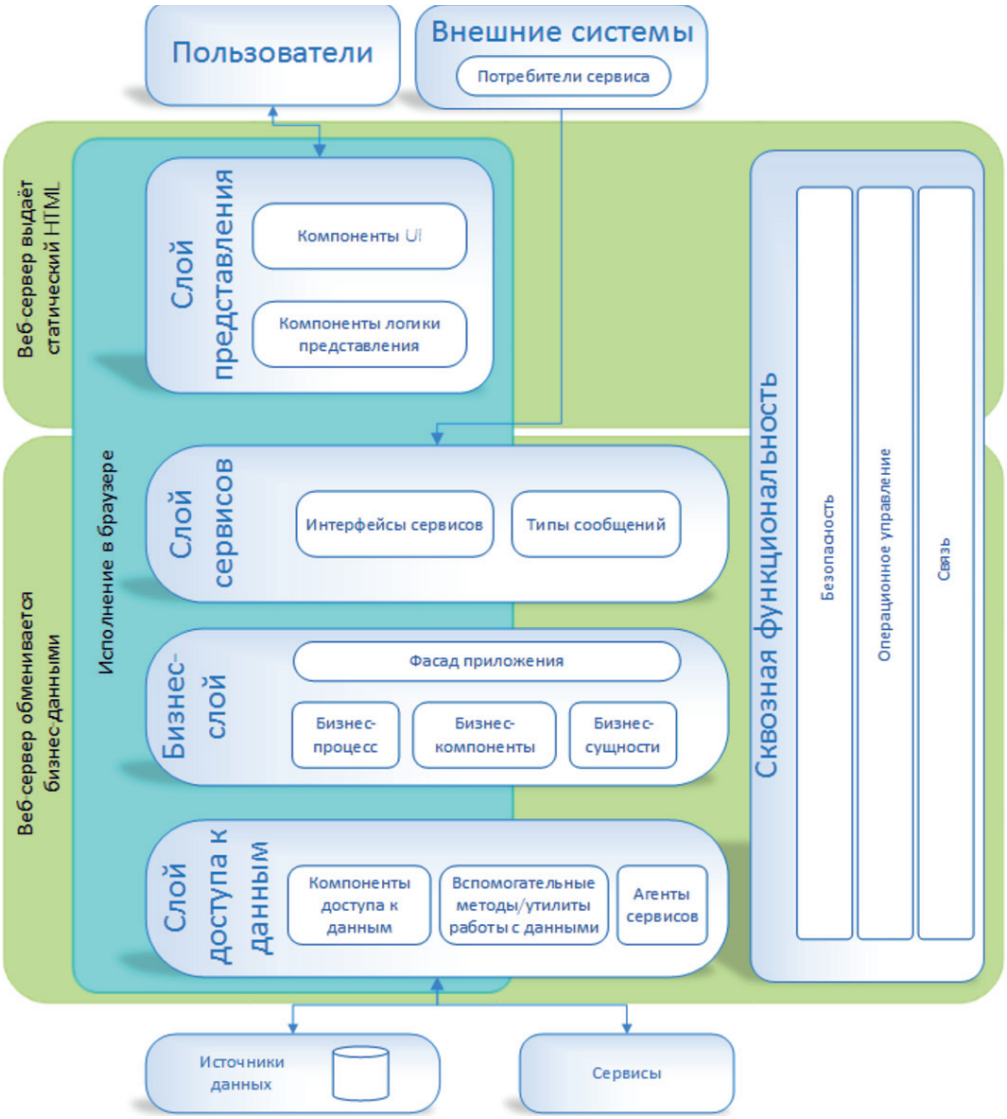

Рис. 5. Архитектура Single-page Application в разрезе слоёв

Рассмотрим данную архитектуру относительно всех слоёв. Схема приведена на рис. 5.

Скорость разработки приложений для этой архитектуры ниже, поскольку применяются разнородные технологии на серверной стороне и на стороне клиента, при этом сам объём логики как на одной, так и на другой стороне сопоставим. Часть логики может дублироваться, например валидация данных может происходить как на стороне клиента с целью удобства пользователя, так и на стороне сервера с целью безопасности и целостности данных (поскольку клиентская сторона не обеспечивает защиту от модификации приложения).

Возможности по масштабированию в этом варианте гораздо шире. Также являются характерными относительно небольшие затраты на доставку слоя представления, поскольку он часто представляет собой статические файлы, загружается один раз при старте приложения и может быть вынесен на уровень frontend-сервера. В дальнейшем приложение получает только бизнес-данные. При увеличении количества пользователей в первую очередь требуется масштабирование только сервисов, отдающих бизнес-данные [2].

За счёт слабой связности между компонентами в Single-page Application мы получаем возможность выполнения тести-

рования с меньшими трудозатратами, чем в случае с Server-side HTML.

Отзывчивость и удобство использования у данных приложений наиболее высокие, поскольку объем данных, пересылаемых между клиентской и серверной стороной, минимален, а исполнение кода, реагирующего на действия пользователя, происходит прямо в браузере.

Дополнительно следует отметить, что требуются специальные меры для обеспечения безопасности приложения и данных, поскольку часть логики вынесена в клиентский JavaScript, который потенциально может быть модифицирован.

данных и недоступности данных для посторонних пользователей [5].

Результаты сравнительного анализа методов построения эффективной архитектуры приведены в таблице (приводятся экспертные оценки от 0 до 5, где 0 – плохой, а 5 – отличный уровень).

По результатам сравнения этих двух подходов было принято решение применить новый вид архитектуры создаваемых приложений на основе Single-page Application. Несмотря на то, что скорость разработки для команд, которые привыкли к использованию Server-side HTML, изначально будет ниже, а также потребуется

Сравнительный анализ методов построения эффективной архитектуры веб-ориентированных бизнес-приложений

	Критерий	Server-side HTML	Single-page Application
1	Скорость разработки	4	3
2	Масштабируемость	2	5
3	Тестируемость	3	5
4	Отзывчивость и удобство использования	3	5
5	Защищённость	5	3

Сравнительный анализ методов построения эффективной архитектуры «облачных» бизнес-приложений

Разные архитектуры по-разному распределяют логику веб-приложений между слоями и компонентами. Сформулируем критерии для оценки функционирования веб-приложений, построенных с использованием разных архитектур. Критерии будут сформулированы с двух точек зрения: разработчика и заказчика (пользователя).

Критерии разработчика:

● Скорость разработки – скорость добавления новой функциональности, изменения существующей;

● Масштабируемость – возможность по увеличению производительности создаваемых приложений. Увеличение производительности должно достигаться не только вертикальным увеличением ресурсов (количеством ядер, оперативной памяти одной машины), но и горизонтальным (добавлением множества машин);

● Тестируемость – возможность и легкость тестирования.

Критерии заказчика (пользователя):

● Отзывчивость и удобство использования – обновление данных на странице и переключение между страницами (время отклика);

● Защищённость – заказчик приложения (прежде всего корпоративный клиент) должен быть уверен в сохранности бизнес-

более внимательное отношение к вопросам безопасности, архитектура на основе Single-page Application выигрывает по более важным критериям, а именно: в вопросе масштабируемости (что важно для поддержки «облачной» среды исполнения), тестируемости, отзывчивости и удобства использования конечных приложений.

Список литературы

1. Дудин Е.Б., Сметанин Ю.Г. Облачные вычисления. обзор, Научно-техническая информация. Серия 1: Организация и методика информационной работы. – 2011. – № 11. – С. 16–21.

2. Мартиросян А.Г. Реализация масштабируемой архитектуры высоконагруженных веб-приложений // Наука и общество. – 2011. – № 1. – С. 108–110.

3. Петров Д.Л., Рябиков Э.М., Егоров В.Т., Сергучев А.А. Технологии облачных вычислений для масштабирования веб-приложений // Информационные технологии моделирования и управления. – 2011. – № 2 (67). – С. 227–233.

4. Руководство MICROSOFT по проектированию архитектуры приложений (2 издание), 2009, Корпорация Майкрософт.

5. Салыкова О.С., Лиценбергер Б.В., Иванова В.В. Уязвимости веб-приложений и способы борьбы ними, В сборнике: Современные инструментальные системы, информационные технологии и инновации сборник научных трудов XII-ой Международной научно-практической конференции в 4-х томах. Ответственный редактор: Горохов А.А. – Курск, 2015. – С. 31–34.

6. Хохряков И.А. Разработка распределённого web-приложения // RSDN Magazine. – 2011. – № 4. – С. 49–57.

7. Шкляр Л. Архитектура веб-приложений, принципы, протоколы, практика / Леон Шкляр, Рич Розен; [пер. с англ. М.А. Райтмана]. – М., 2011. Сер. Высший уровень.

8. Шмидт И.А. Архитектура платформы для разработки бизнес-приложений // Современные проблемы науки и образования. – 2014. – № 6; URL: http://science-education.ru/ru/article/view?id=17081.

9. 3 architectures the good, the bad, the ugly, Available at: http://frontend-architectures.appspot.com.

References

1. Dudin E.B., Smetanin Ju.G. Oblachnye vychislenija. obzor, Nauchno-tehnicheskaja informacija. Serija 1: Organizacija i metodika informacionnoj raboty. 2011. no. 11. pp. 16–21.

2. Martirosjan A.G. Realizacija masshtabiruemoj arhitektury vysokonagruzhennyh veb-prilozhenij // Nauka i obshhestvo. 2011. no. 1. pp. 108–110.

3. Petrov D.L., Rjabikov Je.M., Egorov V.T., Serguchev A.A. Tehnologii oblachnyh vychislenij dlja masshtabirovanija veb-prilozhenij // Informacionnye tehnologii modelirovanija i upravlenija. 2011. no. 2 (67). pp. 227–233.

4. Rukovodstvo MICROSOFT po proektirovaniju arhitektury prilozhenij (2 izdanie), 2009, Korporacija Majkrosoft.

5. Salykova O.S., Licenberger B.V., Ivanova V.V. Ujazvimosti veb-prilozhenij i sposoby borby nimi, V sbornike: Sovremennye instrumentalnye sistemy, informacionnye tehnologii i innovacii sbornik nauchnyh trudov XII-oj Mezhdunarodnoj nauchno-prakticheskoj konferencii v 4-h tomah. Otvetstvennyj redaktor: Gorohov A.A. Kursk, 2015. pp. 31–34.

6. Hohrjakov I.A. Razrabotka raspredeljonnogo web-prilozhenija // RSDN Magazine. 2011. no. 4. pp. 49–57.

7. Shkljar L. Arhitektura veb-prilozhenij, principy, protokoly, praktika / Leon Shkljar, Rich Rozen; [per. s angl. M.A. Rajtmana]. M., 2011. Ser. Vysshij uroven.

8. Shmidt I.A. Arhitektura platformy dlja razrabotki biznes-prilozhenij // Sovremennye problemy nauki i obrazovanija. 2014. no. 6; URL: http://science-education.ru/ru/article/view?id=17081.

9. 3 architectures the good, the bad, the ugly, Available at: http://frontend-architectures.appspot.com.