УДК 004.89

О.Г. Молчанова, А.Ю. Соколов

Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ», Харьков

ИНТЕГРАЦИЯ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ В СИСТЕМАХ УДАЛЕННОГО КОНТРОЛЯ ЗНАНИЙ В ВУЗАХ В ЕДИНУЮ ИНФОРМАЦИОННУЮ СИСТЕМУ

Рассматриваются вопросы интеграции информационных технологий в системах тестирования высокой ответственности. Задача актуальна для организации, проведения, анализа тестов, например, в национальных системах оценивания. Предложена модель формирования программных модулей, выполняющих различные функции в системе, построенных по открытому типу. Приведены иллюстративные примеры модулей для анализа теста, автоматизации проверки эссе и других. Предложенный подход открывает перспективы автоматизации многих процессов в системах тестирования в единой форме и может быть полезен в учебных заведениях, которые занимаются разработкой и внедрением автоматизированных систем тестирования.

Ключевые слова: *интеграция систем тестирования, модульный подход, автоматизация.*

Введение

В настоящее время практически все ВУЗы оснащены компьютерами, имеют свои локальные сети, доступ к сети Internet, что позволяет перейти от традиционных методов обучения и оценки полученных знаний к новым обучающим технологиям.

Можно выделить пять этапов в эволюции развития контроля знаний (КЗ), которые отражают формы организации КЗ и роль преподавателя в этом процессе:

1. Традиционный контроль знаний. Для оценки знаний студентов в учебном процессе традиционно используются такие формы контроля знаний, как: контрольная работа, лабораторная работа, курсовая работа, курсовой проект, реферат, домашнее задание, собеседование, тестирование, зачет, экзамен, дипломная работа.

2. Контроль знаний с использованием бумажных (не компьютерных) средств. При данном подходе для контроля используются заранее подготовленные бланки, содержащие контрольные задания (тесты).

3. Контроль знаний с использованием технических устройств. При данном подходе студент, получив от преподавателя индивидуальный набор заданий, выполняет его и вводит в устройство номер своего варианта и результат решения каждого задания, а устройство проверяет введенные ответы, рассчитывает и выводит оценку за работу.

4. Компьютерный контроль знаний.

Контроль знаний обеспечивают специальные компьютерные программы, в которых осуществляется:

1) формирование индивидуального набора контрольных заданий каждому обучаемому;

2) вывод заданий на экран;

3) анализ ответов обучаемого;

4) выставление оценки;

5) хранение результатов контроля и данных о работе студента с обучающей программой, которые могут быть впоследствии использованы преподавателем и др.

5. Удаленный контроль знаний. Данный подход связан с широким использованием в учебном процессе возможностей сети Internet. Отличительными чертами удаленного КЗ является:

1) применение современных технических средств связи и передачи информации между обучаемым и преподавателем;

2) свобода выбора обучаемым темпов обучения, времени и места обучения.

По сравнению с традиционными формами КЗ, компьютерный контроль знаний, умений и навыков имеет ряд преимуществ:

1) использование новейших методик проверки и оценки знаний студентов;

2) использование современных информационных технологий;

3) возможная адаптация к индивидуальным характеристикам студентов.

На сегодняшний день существует проблема, связанная с тем, что в большинстве высших учебных заведений наиболее часто применяются первые два описанные вида контроля знаний (традиционный контроль знаний и контроль знаний с использованием бумажных средств). Тогда как скорость процесса проверки знаний может быть существенно увеличена, если ввести в процесс обучения компьютерные системы КЗ. Кроме этого при условии грамотно спроектированной системы КЗ её преимущества над традиционными формами КЗ позволят значительно улучшить качество оценки знаний.

Целью данной работы является разработка системы удаленной оценки знаний учащихся для тестирования студентов ВУЗов.

Актуальность работы связана с тем, что:

1) обладатели системы удаленной оценки знаний смогут осуществлять поиск, устанавливать контакт с абитуриентами, а также вести сбор данных о них;

2) рост доступности сети интернет и развитие информационных технологий требует использования их в создании новых способов улучшения образования;

3) на текущий момент не существует комплексного программного продукта, позволяющего проводить тестирование и анализ результатов тестов;

4) все существующие программные продукты обладает целым рядом недостатков, которые необходимо устранить.

1. Анализ существующих компьютерных систем контроля знаний

Основные программные продукты, имеющиеся на рынке на текущий момент, их основные достоинства и недостатки представлены ниже.

XCalibre

Достоинства:

- Позволяет проводить полный анализ психометрических показателей для IRT моделей.
- Позволяет работать одновременно с различными типами тестовых заданий.
- User-friendly interface.
- Хорошая система результирующих отчетов.
- Возможность импорта результирующих файлов в нескольких форматах.

Недостатки:

- Платный продукт. Отсутствие возможности дальнейшей доработки программного продукта, создания дополнительных модулей.
- Отсутствие системы хранения и обработки тестовых заданий, конструирования тестов.
- Расчет психометрических показателей выполняется только для IRT моделей.

SPSS

Достоинства:

- Мощный статистический аппарат.
- Хорошая система результирующих отчетов.
- Возможность преобразования исходных данных к желаемому виду.
- Возможность импорта результирующих файлов в нескольких формата.
- Позволяет работать одновременно с различными типами тестовых заданий.

Недостатки:

- Платный продукт. Отсутствие возможности дальнейшей доработки программного продукта, создания дополнительных модулей.
- Отсутствие системы хранения и обработки тестовых заданий, конструирования тестов.
- Более глубокий анализ тестовых заданий (расчет IRT показателей) реализуется через скрипты и требует дополнительных навыков в программировании.
- Избыточная функциональность усложняет использование программного продукта для расчета психометрических показателей.

Winsteps

Достоинства:

- Проводит полный анализ психометрических показателей для IRT моделей.
- Позволяет работать одновременно с различными типами тестовых заданий.

Недостатки:

- Платный продукт. Отсутствие возможности дальнейшей доработки программного продукта, создания дополнительных модулей.
- Отсутствие системы хранения и обработки тестовых заданий, конструирования тестов.
- Расчет психометрических показателей выполняется только для IRT моделей.
- Плохое графическое представление результатов исследования.

Таким образом, можно выделить ряд схожих недостатков у всех имеющихся программ на текущий момент:

- программы являются платными;
- невозможность изменения функционала в виду ограничений лицензий, закрытости исходного кода, его избыточная сложность;
- отсутствие возможности создания своих тестов и проведения тестирования среди целевой аудитории;
- возможность анализа данных представленных только в одном специальном виде;
- перегруженность и неудобность пользовательского интерфейса.

Ввиду перечисленных недостатков, актуальным является разработка собственного программного продукта, устраняющего данные недостатки.

2. Основные задачи системы оценки контроля знаний

Если проанализировать требования к системе удаленной оценки знаний и управления тестами в целом, то можно выделить ряд подзадач, которые она должна решать. Все эти подзадачи на высоком уровне детализации можно изобразить на схеме, представленной на рис. 1.

Рассмотрим кратко каждую из подзадач и требования к разрабатываемой информационной системе.

1. Разработка задания.

Каждый тест состоит из целого ряда заданий. Поэтому одним из основных требований к функциональности программного комплекса является возможность создавать тестовые задания. Программа должна предоставлять удобный и простой интерфейс для создания таковых. Эта подзадача предполагает прямое взаимодействие оператора и программы.

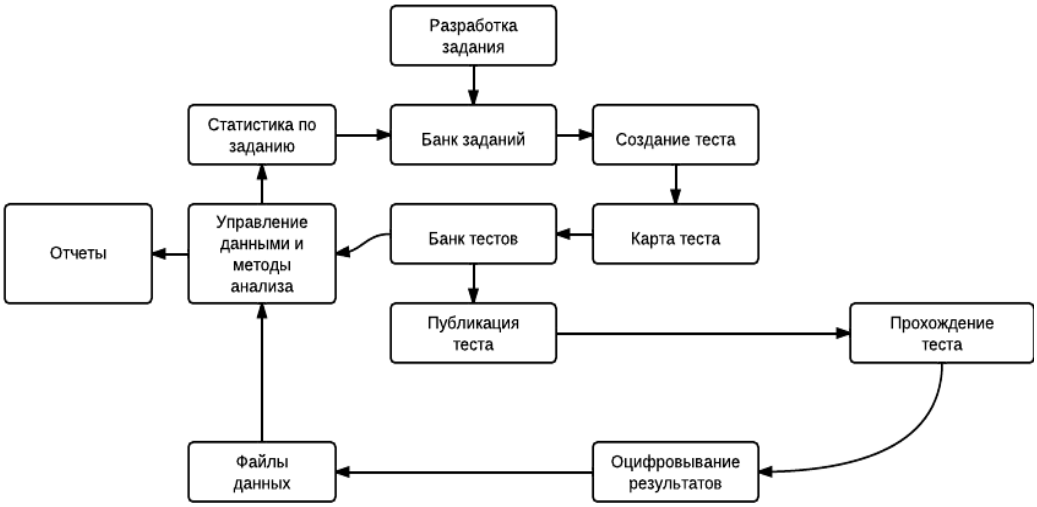

Рис. 1. Взаимосвязь подзадач в системах тестирования

2. Создание теста.

Каждое тестовое задание по отдельности не представляют собой особого интереса. Для того чтобы пользоваться системой для оценки своих знаний, конечные пользователи должны иметь возможность проходить полноценный тест. Т.е. это должен быть сгруппированный тематически набор тестовых заданий, ограниченный по времени (ограниченным по времени может быть как сам тест, так и каждое тестовое задание в частности).

3. Публикация теста.

В этом случае прохождения теста в электронном виде оператор, используя разрабатываемую программу, редактирует информацию о существующем тесте, указывая даты, в которые данный тест пользователи будут иметь возможность пройти. После того, как эти даты заполнены, программа считает их, и дальше будет следить за тем, чтобы по наступлению указанного периода времени данный тест оказался в списке тестов, доступных к прохождению. В качестве отдельного вопроса здесь можно выделить задачу ограничения по целевой аудитории.

4. Прохождение теста.

Прохождение теста реализовывается только для случая его прохождения в электронном виде.

В этом случае программа должна предоставить конечному пользователю возможность зарегистрироваться. После регистрации пользователю предлагается выбор одного из доступных тестов. После того, как пользователь уведомляется о том, что пользователь действительно хочет начать прохождение выбранного теста, система обращается к Банку тестов и выбирает указанные в тесте задания. Далее система предлагает пользователю в определенной последовательности ответить на указанные вопросы. В ходе прохождения теста система отслеживает время, которое пользователь тратит на ответы на вопросы, а также на весь тест в целом. Система должна предоставлять возможность пользователю возвращаться к отвеченным вопросам и изменять выбранный вариант ответа.

5. Оцифровывание результатов.

Этот шаг предпринимается только в случае ручного проведения теста. Этот шаг представляется достаточно сложным, поэтому в рамках данной работы реализован не будет. Единственное, что можно отметить, это факт того, что для выполнения этой подзадачи должна существовать отдельная программа, способная распознать текст бланков ответов и создать файлы в специальном формате, которые далее смогут быть распознаны программой, ответственной за управление данными и их анализ.

6. Управление данными и методы анализа.

Целью этой подзадачи является сбор данных об ответах пользователей, их обработка. В случае электронного прохождения теста на вход этой подзадаче подаются данные из Банка Ответов Пользователей. В случае же ручного прохождения тестов этот шаг ответственный в первую очередь за то, чтобы считать данные из файлов и сохранить их Банк Ответов Пользователей, чтобы затем извлечь их оттуда для последующей обработки.

После того, как данные считаны из Банка Ответов, программный модуль предоставляет возможность для выполнения двух отдельных подзадач:

7. Создание статистики по заданию.

На этом этапе программа должна рассчитать ряд психометрических показателей каждого задания (таких как сложность, дискриминативность и др.) и сохранить их в Банке Заданий. Таким образом, на Банк Заданий ложится дополнительная ответственность – хранения общих просчитанных статистик по каждому из заданий. Это делается с целью упрощения создания отчетов для клиентов программы по каждому заданию. В частности для оптимизации скорости генерации отчетов.

Данный шаг должен предприниматься автоматически без участия конечного пользователя.

8. Создание отчетов.

Эта задача предполагает прямое взаимодействие пользователя, анализирующего статистики и показатели заданий. Программа должна предоставлять пользователю удобный и простой интерфейс для того, чтобы он мог выбрать интересующий его тест или же заданий и увидеть основные или детальные статистики. Программа на этом шаге будет достаточно активно взаимодействовать как с Базой Ответов Пользователей (для получения данных и построения отчетов о пользователях, прошедших тест), так и с Базой Заданий (для того, чтобы получить основные показатели по каждому заданию). Поэтому крайне важным является высокая степень оптимизации скорости получения данных.

Программа должна предоставлять пользователю не только возможность визуально проанализировать статистики, но также и сохранять их на локальной машине в нескольких форматах.

3. Архитектура программного обеспечения

Предлагается трехуровневая архитектура для web-приложений (рис. 2).

Специфика данного программного продукта заключается в том, что задачи, возникающие в процессе организации и проведения тестирования, могут меняться как во времени, так и в различных организациях (например, новые аналитические отчеты и исследования, не предусмотренные в начале внедрения системы тестирования). Поэтому необходимо выбрать такую реализацию архитектуры, в которой дополнение программного обеспечения новыми функциями станет возможным без изменения ранее разработанных модулей.

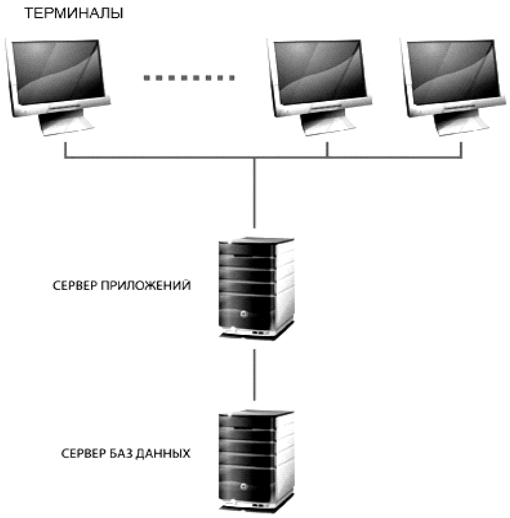

Рис. 2. Модель трехуровневой архитектуры

С точки зрения программного обеспечения данная архитектура может быть реализована набором взаимосвязанных модулей (рис. 3).

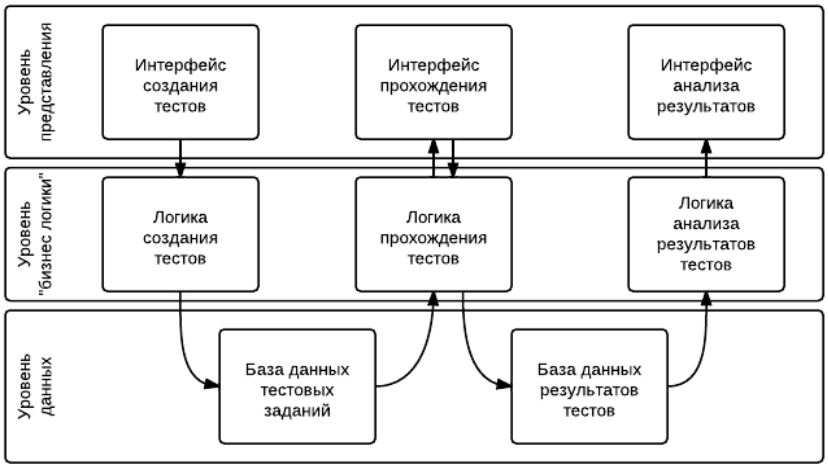

Рис. 3. Модули ИС

Существует несколько способов реализации такого механизма. Одними из самых доступных являются два способа:

- Реализовать новый мета-язык, описывающий преобразования над данными и их конечный вид.
- Реализовать поддержку пользовательских скриптов, которые будут описывать логику расчета показателей, а также конечный вид информации.

Первый способ связан с очень большими затратами и трудностями при разработке. В дополнение к этому – разработанный мета-язык все равно будет иметь целый ряд ограничений, который могут сделать невозможной реализацию необходимого конечного отчета. Поэтому необходим подход, который не будет накладывать никаких ограничений на свободу действий при реализации пользовательских алгоритмов. Таким подходом является поддержка пользовательских скриптов, написанных на одном из полноценных существующих языков программирования. Таким языком является язык Groovy.

Groovy – объектно-ориентированный язык программирования, разработанный для платформы Java как альтернатива языку Java. Groovy может быть использован в любом Java проекте.

Groovy-скрипты представляют собой обычные текстовые файлы с расширением .groovy, которые компилируются при непосредственном запуске.

Используя последний подход, в системе была реализована поддержка пользовательских скриптов, работающая по следующей схеме:

- Приложение имеет конфигурационный параметр, описывающий папку на файловой системе, в которой будут храниться скрипты.
- Пользователь разрабатывает groovy скрипт таким образом, что он содержит groovy-класс, реализующий определенный интерфейс. Этот класс реализовывает алгоритм расчета статистических метрик.
- Когда пользователь переходит на страницу с пользовательскими отчетами, приложение сканирует сконфигурированную папку и создает в памяти экземпляры описанных groovy классов.
- Далее программа по очереди проходит через все созданные экземпляры классов, вызывая у них определенный метод. На вход этому методу передается информация о пройденном тесте, а в качестве выходного значения выступает информация, описывающая конечный отчет.
- Программа, используя информацию о типах полученных отчетов, отрисовывает их в HTML формате, после чего те передаются пользователю.
- На стороне пользователя полученные отчеты визуализуруются с помощью Google Visualization API.

На рис. 4 – 6 приведены соответственно:

- Диаграмма классов, которые используются в groovy-скриптах.
- Интерфейс, реализуемый groovy-классом в скрипте.
- Общая схема интеграции с groovy-скриптом.

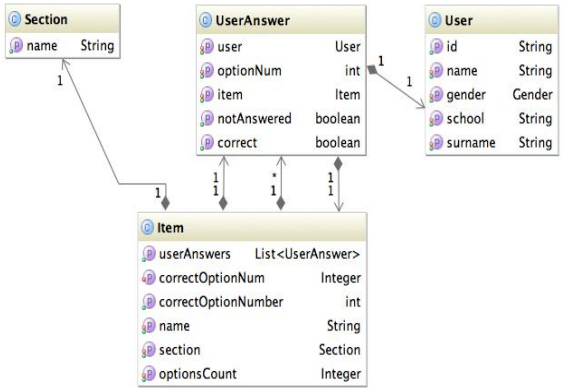

Рис. 4. Диаграмма классов, которые используются в groovy-скриптах

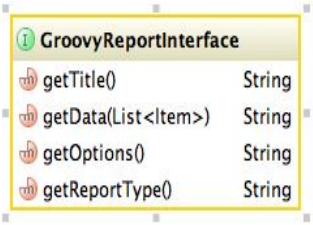

Рис. 5. Интерфейс, реализуемый groovy-классом в скрипте

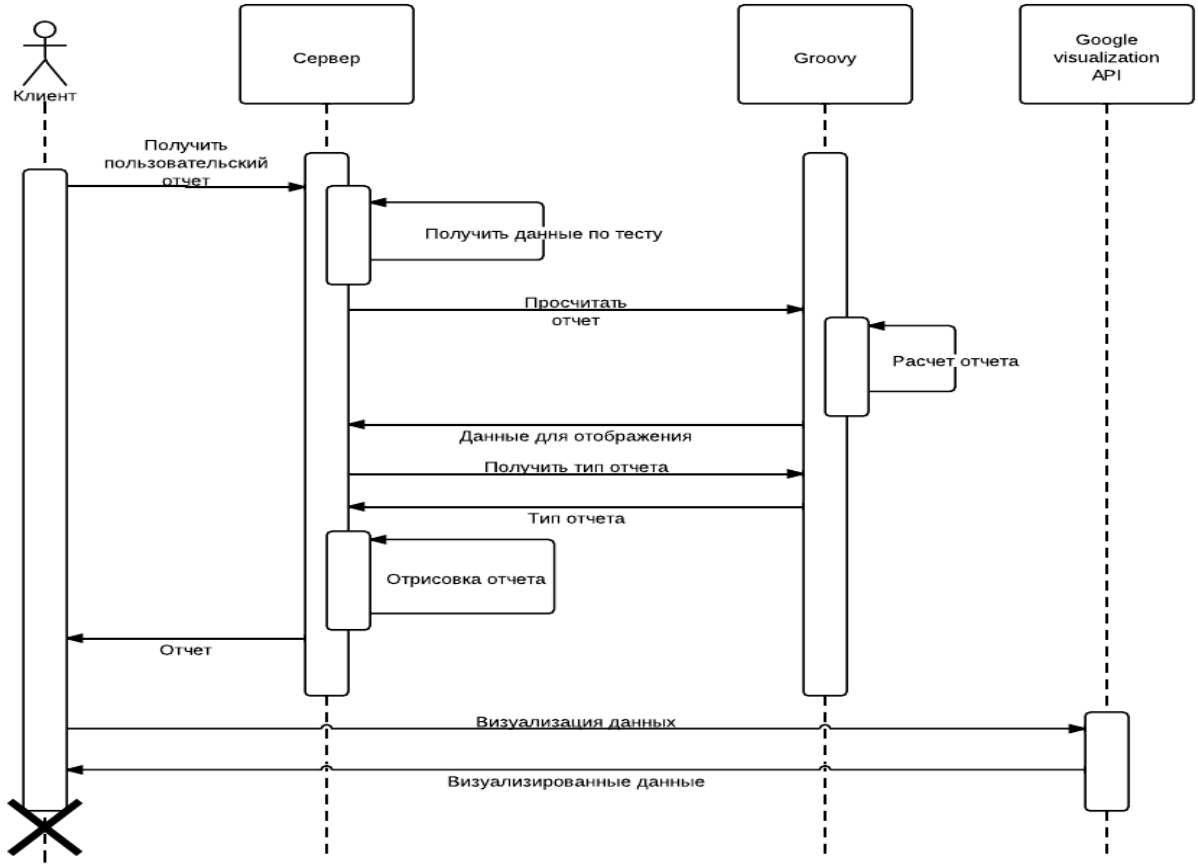

Рис. 6. Общая схема интеграции с groovy-скриптом

4. Пример

В качестве примера приведем скрипт для расчета количества не отвеченных заданий в тесте (рис. 7).

```
public class Report {

    public String getTitle() {
        return "Not Answered - Table";
    }

    public String getData(items) {
        StringBuilder sb = new StringBuilder();

        sb.append("[['Item','Answers'],");
        for (item in items) {
            sb.append("['");
            sb.append(item.getName());
            sb.append("',");

            int notAnswered = 0;
            for (def answer : item.getUserAnswers()) {
                if (answer.isNotAnswered()) {
                    notAnswered++;
                }
            }
            sb.append(notAnswered);
            sb.append("],")
        }
        sb.setLength(sb.length()-1);
        sb.append("]");

        return sb.toString();
    }

    public String getOptions() {
        return null;
    }

    public String getReportType() {
        return "table";
    }

}
```

Рис. 7. Скрипт для расчета количества неотвеченных заданий

Основные части, из которых состоит этот класс, следующие:

1. Метод getTitle – отвечает за название отчета, которое будет отображено на клиентской стороне.

2. Метод getData – отвечает за возвращение данных для отображения отчета в специально сформированном виде. Этот вид представляет из себя JavaScript массив, который будет передан на вход Google Visualization API. Описание структуры массива можно найти на официальном сайте Googe Visualization API.

3. Метод getOptions – представляет собой дополнительные параметры для отображения отчета посредством Google Visualization API. Описание этих параметров также содержится на официальной веб-странице Google Visualization API. Описание может иметь ряд свойств, специфичных для каждого вида отображаемого отчета – таблица, линейный график, столбчатая диаграмма, круговая диаграмма и т.д.

4. Метод getReportType – возвращает строку, означающую, какой тип отчета будет показан. Некоторые из возможных значений:

a) “table” – отчет будет отображен в виде таблицы;

b) “LineChart” – отчет будет отображен в виде линейного графика;

c) “ColumnChart” – отчет будет отображен в виде столбчатой диа-граммы;

d) “AreaChart” – отчет будет отображен в виде круговой диаграммы.

Полный список возможный видов отображения отчета может быть найден на web-странице Google Visualization API.

В качестве примера визуализации пользовательского отчета приведем визуализацию отчета, описанного выше (рис. 8).

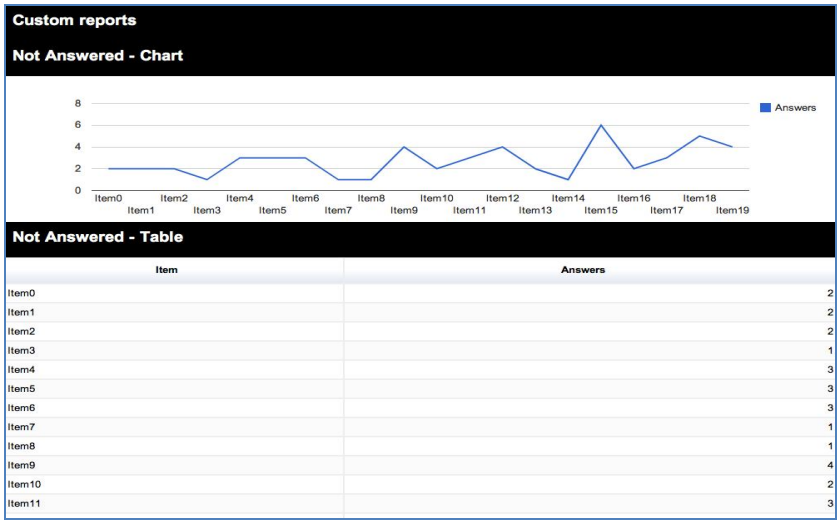

Рис. 8. Пример визуализации пользовательского отчета

Выводы

В заключении предложены способы реализации информационной системы в комплексе следующих технологий:

- Java 6 – выступает в роли основного языка, на котором написана серверная часть информационной системы.
- Spring MVC – технология, которая используется для построения клиент-серверного web-приложения.

- Spring Security – технология для реализации механизмов безопасности (аутентификации и авторизации).
- HTML.
- CSS.
- Google Visualization API – технология компании Google, предоставляющая собой возможность визуализации данных, используя множество различных представлений.
- jUnit – основная технология для написания юнит-тестов к коду.
- Mockito – отдельная библиотека, расширяющая возможности при написании юнит-тестов (предоставляет возможность использования Mock-объектов).
- Log4j – библиотека, используемая для логирования процессов в информационной системе.
- MySQL – СУБД для управления базой данных пользователями.

Как следует из указанного списка, все программные технологии являются открытыми и позволяют конфигурировать информационную систему в виде открытых кодов, что важно в системах критического применения, к которым относятся системы тестирования высокой ответственности.

Список литературы

1. Самылкина Н.Н. Современные средства оценивания результатов обучения: учеб. пособие / Н.Н. Самылкина. – М.: Лаборатория Базовых Знаний, 2007. – 172 с.

2. Куклин В.Ж. О компьютерной технологии оценки качества знаний / В.Ж. Куклин, В.И. Мешалкин, В.Г. Наводнов, Б.А. Савельев // Высшее образование в России. – 1993. – № 1. – С. 146-153.

3. Звонников В.И. Современные средства оценивания результатов обучения: учеб. пособие / В.И. Звонников, М.Б. Челышкова – М.: Издательский центр "Академия", 2007. – 223 с.

4. Андреев А.Б. Компьютерное тестирование: системный подход к оценке качества знаний студентов / А.Б. Андреев // Высшее образование в России. – 2001. – № 5. – С. 19 – 26.

5. Щапов А.Н. Тестовый контроль в системе рейтинга / А.Н. Щапов, Н. К. Тихомирова, С.А. Ершиков, Т.Е. Лобова // Высшее образование в России. –, 1995. – №3. – С. 100-102.

6. Гутгарц Р.Д. Особенности дистанционного тестирования в Интернете / Р.Д. Гутгарц. // Высшее образование в России. – 2001. – № 6. – С. 37 – 26.

7. Аллахвердиева Д.Т. Опыт применения тестов для дидактической экспертизы обучения / Д.Т. Аллахвердиева // Высшее образование в России. – 1993. – №2. – С. 1 – 104..

8. Хубаев Г.А. О построении шкалы оценок в системах тестирования / Г.А. Хубаев // Высшее образование в России. – 1996. – № 3. – С. 122-125.

9. Васильев А.Н. Java. Объектно-ориентированное программирование / А.Н. Васильев. – Санкт-Петербург, Питер, 2011. – 400 с.

10. Брюс У. Перри. Java сервлеты и JSP. Сборник рецептов / У. Перри Брюс. – Москва, КУДИЦ-Пресс, 2009. – 768 с.

11. Башар Абдул-Джавад. Groovy и Grails. Практические советы / Абруд-Джавад Башар. – М.: ДМК Пресс, 2010. – 392 с.

12. Гарнаев А.С. WEB-программирование на Java и JavaScript / А.С. Гарнаев, С.С. Гарнаев. – С-Пб.: БХВ-Петербург, 2005. – 1040 с.

Поступила в редколлегию 2.02.2012

Рецензент: д-р техн. наук, проф. Н.Д. Кошевой, Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ», Харьков.

ІНТЕГРАЦІЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ В СИСТЕМАХ ВІДДАЛЕНОГО КОНТРОЛЮ ЗНАНЬ У ВУЗАХ В ЄДИНУ ІНФОРМАЦІЙНУ СИСТЕМУ

О.Г. Молчанова, О.Ю. Соколов

Розглядаються питання інтеграції інформаційних технологій в системах тестування високої відповідальності. Задача актуальна для організації, проведення, аналізу тестів, наприклад, в національних системах оцінювання. Запропоновано модель формування програмних модулів, що виконують різні функції в системі, побудованих з відкритого типу. Наведено ілюстративні приклади модулів для аналізу тесту, автоматизації перевірки есе та інших. Запропонований підхід відкриває перспективи автоматизації багатьох процесів в системах тестування в єдиній формі і може бути корисний у навчальних закладах, які займаються розробкою та впровадженням автоматизованих систем тестування.

Ключові слова: *інтеграція систем тестування, модульний підхід, автоматизація.*

INTEGRATION OF INFORMATION TECHNOLOGIES IN THE REMOTE CONTROL OF KNOWLEDGE IN UNIVERSITIES IN UNATED INFORMATION SYSTEM

O.G. Molchanova, O.Yu. Sokolov

The problems of integrating information technology systems, testing of high responsibility are considered. The problem is relevant to the organization, conduct, analysis, tests, for example, national systems of evaluation. A model of the formation of software modules is proposed that perform various functions in a system built on an open type. Illustrative examples are presented for the analysis of test modules, test automation, and other essays. The proposed approach opens up prospects for the automation of many processes in the test systems in a single form and may be useful in educational institutions that are engaged in the development and implementation of automated testing systems.

Keywords: *Integration of systems testing, a modular approach, automation.*