Студенческий Вестник
Электронный научно-технический журнал Октябрь 2015

СОВРЕМЕННЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ В АВТОМАТИЗИРОВАННОЙ СИСТЕМЕ КОНТРОЛЯ ЗНАНИЙ [1]

М. В. Аршинский, Н. В. Божков, **А.И. Якимов**

Аннотация. В статье представлены основные принципы построения автоматизированных систем контроля знаний на основе тестов. Показаны пути реализации принципов на основе использования методов языка программирования C# и платформы .NET Framework.

Ключевые слова: информационные технологии, Экзаменатор, учебный процесс, контроль знаний.

1. Введение

В учебном процессе применяют тесты – задания, дающие возможность быстро выявить и оценить уровень знаний, умений, навыков. Тестирование хорошо поддается машинной обработке. Сильная сторона тестирования – возможность экономно проверить усвоение большого по объему материала (особенно, если речь идет компьютерном тестировании), а также его объективность, независимость от субъективной оценки преподавателя [1, 2]. Разработаны теория и практика педагогических измерений, формы и методы массового тестирования [3].

Использование широкого спектра информационных технологий в тестирующих информационных системах позволяет предложить экзаменуемому простой и интуитивно понятный интерфейс путем его оптимизации, организовать проведение экзамена в компьютерном сетевом классе, провести декомпозицию функционала программного обеспечения, реализовать блокировку процессов на компьютере экзаменуемого для повышения качества оценки знаний, ввести дополнительные функции для реализации аналитических возможностей информационной системы.

2. Принципы построения автоматизированных систем контроля знаний

Опыт эксплуатации автоматизированных систем контроля знаний позволяет определить несколько основных принципов, в соответствии с которыми они должны проектироваться.

Принцип распределенного контроля знаний подразумевает одновременный опрос учащихся в аудитории. Для реализации принципа распределенного контроля знаний может быть использован сетевой компьютерный класс. При проведении контроля знаний один компьютер выбирается в качестве экзаменационного сервера; остальные – клиенты, на которых осуществляется тестирование.

Принцип декомпозиции функционала заключается в разделении программной системы на отдельные функциональные программные модули, чтобы сложность освоения системы контроля не оказывала влияние на оценку качества знаний, умений, навыков. Устанавливается программное обеспечение на сервере: программные модули экзамена-

[1] Работа выполнена в порядке личной инициативы по НИРС

тора; экзаменационные базы данных (БД) и файлы с экзаменационными вопросами; на клиенте: программные модули для проведения тестирования.

Принцип чистоты контроля знаний подразумевает запрет на использование различных вспомогательных источников информации (шпаргалок), которые будут нарушать чистоту контроля знаний. Поскольку тест выполняется на персональном компьютере, нельзя гарантировать, что при выполнении теста не будут запущены различные сторонние приложения, облегчающие ответы на вопросы тестов.

Принцип интеллектуализации интерфейса подразумевает использование в программном модуле функций, способных облегчить выполнение некоторых операций экспертом – преподавателем. Например, автоматическая проверка результатов тестирования, сортировка результатов по различным параметрам, автоматическая генерация сложных паролей и т. д. Для анализа множества результатов тестирования, удобнее иметь инструмент, позволяющий обрабатывать и хранить результаты тестирования автоматически.

Принцип оптимизации интерфейса пользователя заключается в создании удобного и интуитивно понятного интерфейса, обеспечивающего комфортные условия при работе с тестирующей программой [4].

3 Реализация принципов построения автоматизированной системы контроля знаний на основе информационных технологий

Для организации работы по компьютерной сети (во исполнение принципа распределенного контроля знаний) с использованием операционной системы Windows не ниже XP с платформой .NET Framework с версией не ниже 1.0 возможно подключение библиотеки .NET Remoting. .NET Remoting – компонент API (Application Programming Interface), созданный компанией Microsoft, для межпроцессного взаимодействия. Этот компонент является реализацией протокола SOAP (Simple Object Access Protocol). Выпущен компонент .NET Remoting в 2002 году вместе с версией 1.0 пакета .NET Framework.

.NET Remoting позволяет приложению создать объект (именуемый *remotable object*), доступный в рамках *remoting boundaries* и расположенный в домене приложения внутри или одного процесса, или в другом процессе, исполняющемся на этом же компьютере, или даже на другом компьютере, соединённом сетью. .NET Remoting делает ссылку на удалённый (remotable) объект доступной клиентскому приложению, которое затем направляет запросы к экземпляру удалённого объекта так, как если бы это был локальный объект. Однако, фактическое исполнение кода происходит на серверной стороне.

В процессе выполнения запросов любые вызовы методов, направленные объекту, включая идентификатор метода и передаваемые параметры, сериализуются в байтовый поток и передаются посредством канала связи, реализованного для конкретного протокола. Передача происходит путём записи данных в транспортный ввод канала. На серверной стороне прокси-сервер читает поток данных из вывода канала и выполняет вызов удалённого компонента от лица клиента. Результаты сериализуются и передаются через канал клиенту, где прокси-сервер читает результат и передаёт его вызывающему приложению.

Реализация принципа декомпозиции функционала предполагает разработку тестирующих программ в виде нескольких программных модулей так, чтобы оставить функции тестирования в программном модуле, доступном только для экзаменуемого. Так, например, автоматизированный тестирующий комплекс Экзаменатор 2 состоит из сле-

дующих программных модулей: администратор (Administrator.exe), редактор вопросов (QuestionEditor.exe), экзаменатор (Examiner.exe.). Программные модули Administrator.exe и QuestionEditor.exe предназначены для организатора проведения экзамена. Examiner.exe обладает минимальным функционалом и размещается на компьютере экзаменуемого [5].

В ходе эксплуатации программного комплекса декомпозиция функционала была реализована и в программном модуле, предназначенном для организатора тестирования (рисунок 1, рисунок 2)

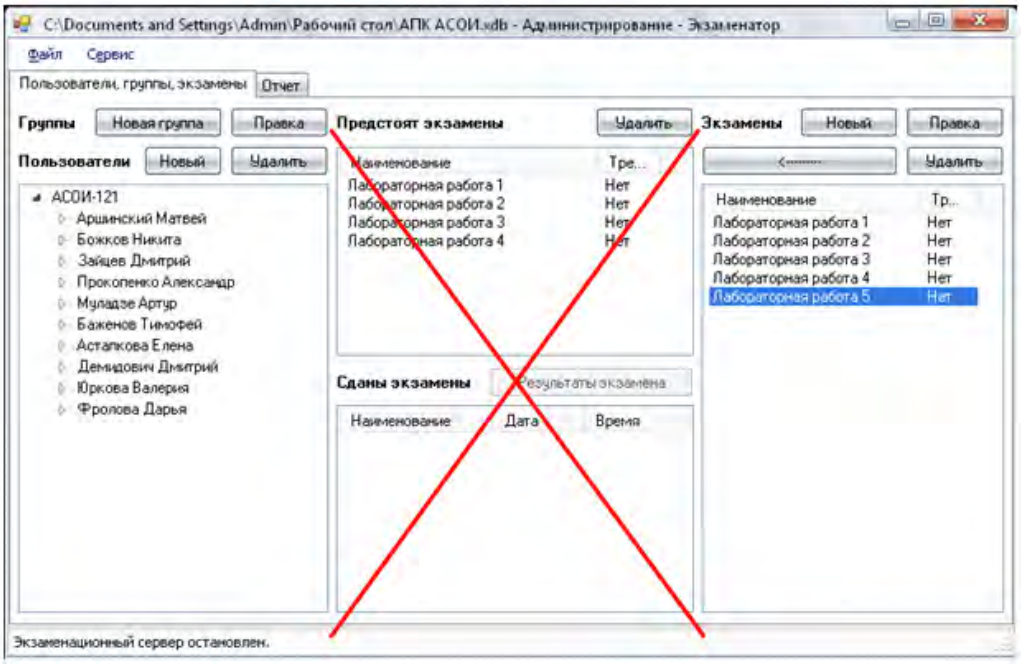

Рис. 1. Главное меню программы Экзаменатор (администратор)

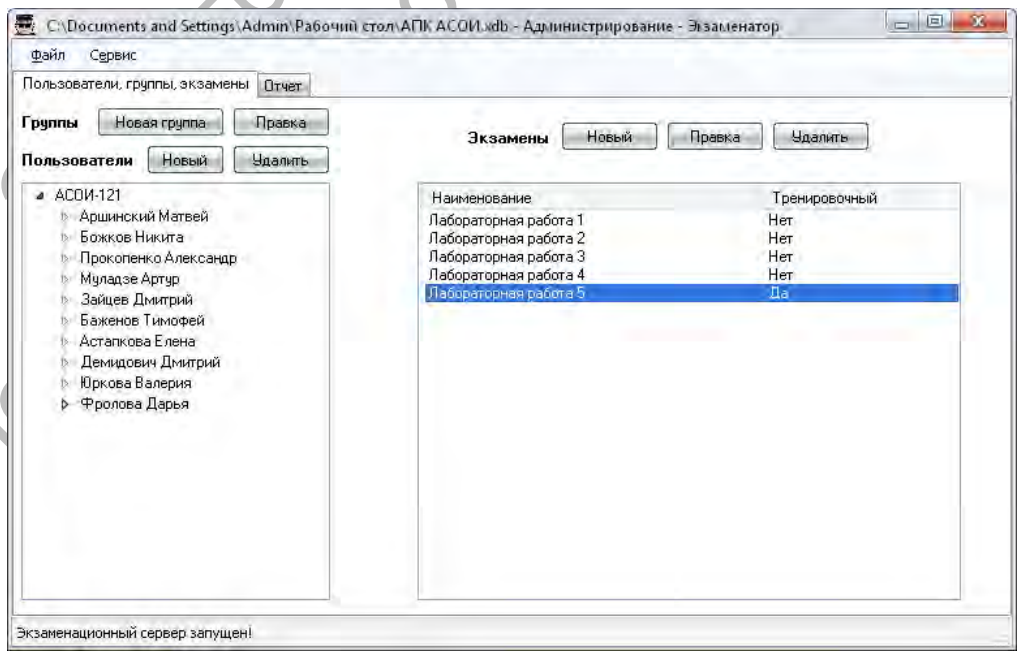

Рис. 2. Главное меню программы Экзаменатор 2.0 (администратор)

Для контроля запуска сторонних приложений во исполнение принципа чистоты контроля знаний возможно использование библиотеки Diagnostics, реализованной на платформе .NET Framework 1.0 и старше. Во время выполнения теста, с помощью класса Process, определенного в пространстве имен Diagnostics, производится сканирование запущенных на компьютере процессов. Если среди процессов обнаруживается программа из черного списка, дальнейшее выполнение теста можно запретить, на экран выводится сообщение о том, что результаты отправлены на сервер, и приложение будет закрыто, например, через 30 секунд. В версии Windows XP запускается звуковой сигнал на материнской плате.

В соответствии с принципом интеллектуализации интерфейса для анализа множества результатов тестирования удобнее иметь инструмент, позволяющий обрабатывать и хранить результаты тестирования автоматически. Для этих целей использован стандартный инструмент ListView. Он предоставляет возможность увидеть отчет о пройденных тестах в простой и наглядной форме таблицы, разделенной на столбцы. По всем столбцам можно проводить сортировку. Возможен импорт таблицы результатов в Excel. Также возможно просмотреть ответы каждого пользователя для ознакомления с вопросами, на которые был дан неправильный ответ.

Для реализации принципа оптимизации интерфейса пользователя можно применить основные принципы построения пользовательского интерфейса, такие как группирование схожих элементов контроля в отдельные группы, сокрытие от пользователя лишней информации, обеспечение быстрого восстановления фокуса внимания при переключении между задачами. Это может быть достигнуто путем сокрытия части функционала в меню, создание пошаговых инструкций при настройке программы и т. д. [6, 7].

Методика проведения тестирования: один компьютер выбирается в качестве экзаменационного сервера; остальные – клиенты, на которых осуществляется тестирование. Устанавливается программное обеспечение на сервере: программные модули экзаменатора; экзаменационные БД и файлы с экзаменационными вопросами; на клиенте: программные модули Examiner.exe, ExaminerCore.dll. На сервере запускается Administrator.exe, открываются экзаменационные БД, запускается экзаменационный сервер.

Порядок тестирования: экзаменуемому выдается пароль, затем экзаменуемый запускает Examiner.exe, указывает имя экзаменационного сервера (имя компьютера или IP адрес), выбирает свое имя из списка, выбирает экзаменационный тест (отображаются тесты, предназначенные для его группы и конкретно этому пользователю) и вводит пароль, отвечает на экзаменационные вопросы (по нажатию на кнопку Готово ответы отправляются на экзаменационный сервер).

Просмотр и печать результатов экзамена – через список сданных экзаменов или через отчеты администратора. Печать – из Excel путем экспорта отчета [5].

4. Заключение

Использование методов языка программирования C# и платформы .NET Framework позволили создать автоматизированный тестирующий комплекс Экзаменатор 2, состоящий из следующих программных модулей: администратор (Administrator.exe), редактор вопросов (QuestionEditor.exe), экзаменатор (Examiner.exe.). Программные модули Administrator.exe и QuestionEditor.exe предназначены для организатора проведения экзамена. Examiner.exe обладает минимальным функционалом и размещается на компьютере экзаменуемого.

Литература

1. **Загвязинский, В.И.** Теория обучения: современная интерпретация : учебное пособие / В. И. Загвязинский. - М.: Академия, 2001. - 192 с.
2. **Аникова, Л. А.** Тесты. Информатика и информационные технологии / Л. А. Аникова. – М.: Дрофа, 2004. – 251 с.
3. **Ефремова, Н.** Тестовый контроль в образовании : учебное пособие / Н. Ефремова. – М. : Университетская книга, 2007. – 540 с.
4. **Купер, А.** Алан Купер об интерфейсе. Основы проектирования взаимодействия : пер. с англ. / А. Купер, Р. Рейман, Д. Кронин. – СПб.: Символ-Плюс, 2009. – 688 с.
5. **Божков, Н. В.** Совершенствование качества контроля знаний на основе современных информационных технологий / Н. В. Божков, М. В. Аршинский; науч. рук. А. И. Якимов // Новые материалы, оборудование и технологии в промышленности : материалы междунар. науч.-техн. конф. молод. ученых; редкол.: И. С. Сазонов (гл. ред.) [и др.], Могилев, 22–23 октября 2015 г. – Могилев: Белорус.-Рос. ун-т, 2015. – С. 151.
6. **Жарков, С.** Shareware: профессиональная разработка и продвижение программ / С. Жарков. – СПб.: BHV-СПб, 2001. - 320 с.
7. **Брауде, Э. Д.** Технология разработки программного обеспечения / Э. Д. Брауде. – СПб.: Питер, 2004. – 655 с.

Аршинский Матвей Вадимович
Студент электротехнического факультета
Белорусско-Российский университет, г. Могилев
Тел.: +375(29) 743-91-21
E-mail: matvik1103@gmail.com
Божков Никита Владимирович
Студент электротехнического факультета
Белорусско-Российский университет, г. Могилев
Тел.: +375 (29) 385-77-47
E-mail: prizzzrag@gmail.com
Якимов Анатолий Иванович
Доцент кафедры Автоматизированные системы управления, канд. техн. наук
Белорусско-Российский университет, г. Могилев
Тел.: +375(222) 25-24-47
E-mail: ykm@tut.by