Особенности автоматизированного тестирования знаний студентов в области программирования

Алыкова А.Л., канд. техн. наук

Рассматриваются основные принципы автоматизированного тестирования знаний с учетом целей, направленных на повышение качества обучения, и особенностей, связанных с выявлением истинных умений и навыков, полученных студентом в области программирования. Приводится описание системы автоматизированного тестирования, построенной с учетом этих принципов.

Современные тенденции развития высшей школы направлены на повышение качества образования, обеспечение объективности оценки знаний, расширение доступа к образовательным услугам [1, 2]. Решение этих задач в значительной степени зависит от внедрения компьютерных технологий в учебный процесс.

Современные технические и программные средства, развитые сетевые и мультимедийные технологии позволяют создавать электронные учебники, электронные лекционные курсы, проводить дистанционное обучение в режиме реального времени. Несколько сложнее обстоит дело с контролем уровня знаний. Очень заманчивой выглядит идея автоматизировать этот процесс, так как это позволит исключить человеческий фактор, а следовательно, повысить объективность и независимость оценки. Но автоматизация предполагает формализацию процесса, а это не всегда возможно. Даже в средней школе при введении единого государственного экзамена (ЕГЭ) задания разделены на три уровня [3], и последний третий уровень – творческое задание – проверяет эксперт! И в ближайшие годы вряд ли представится возможным получить с помощью компьютера объективную оценку литературного сочинения или философского трактата.

Наиболее легко поддается автоматизации процесс контроля знаний в тех областях, где можно сформулировать вопросы, задания и задачи, предполагающие конкретный и однозначный ответ, например: в математике, физике, истории в той ее части, где не предполагается развернутый анализ исторической ситуации, экономике и некоторых других. Практически все современные тестовые системы [4, 5, 6] разработаны таким образом, что используют в качестве ответов так называемую «угадайку», число или, в крайнем случае, строку символов. Остановимся на каждом из этих вариантов ответов.

В вопросах типа «угадайка», когда тестируемому необходимо сделать выбор одного или нескольких вариантов из предложенного набора, проконтролировать правильность ответа наиболее легко, но такой тип вопросов накладывает определенные ограничения на формулировку самого вопроса и в той или иной степени содержит подсказку, так как заранее известно, что хотя бы один вариант является верным. Несложно проверить и числовой ответ. Здесь в отдельных случаях, возможно, придется задаться дополнительным параметром – точностью ответа. Несколько больше проблем может возникнуть при анализе строки символов. Не забываем, что вопрос сформулирован так, что ответ должен быть конкретным и однозначным. Но даже в этом случае тестируемый не застрахован от опечаток и орфографических ошибок (к сожалению, уровень грамотности студентов в последние годы оставляет желать лучшего).

Перейдем от общих проблем автоматизированного тестирования к тестированию знаний в области программирования. Компьютер непосредственно задействован в учебном процессе, является инструментом в руках студента, и, казалось бы, что может быть проще, чем проверка знаний с помощью этого компьютера. Но так может рассуждать только человек, не посвященный во все проблемы, связанные с обучением программированию.

Основная цель любого курса, связанного с обучением программированию (независимо от его названия), – привить студенту навыки решения задач на компьютере. Она ни в коем случае не должна сводиться к элементарному умению кодировать на том или ином языке программирования. Процесс решения задачи начинается с ее формулировки, определения математической модели, далее следуют выбор методов решения, разработка алгоритма, составление программы на том или ином языке программирования, наконец, завершают процесс отладка программы и собственно решение задачи. Таким образом, для адекватной оценки знаний, умений и навыков необходимо обеспечить возможность тестирования способностей студента как выполнять отдельные этапы решения задачи, так и решать задачу в целом.

Как показывает многолетняя практика преподавания, лучшей формой проверки знаний студентов в области программирования является решение нетривиальных задач. Не следует отказываться от такого подхода и при автоматизированном контроле знаний. Здесь, возможно, подошел бы опыт проведения международных олимпиад по программированию (см. сайт www.acm.timus.ru), где работоспособность программ проверяется на ряде тестов, и если хотя бы один тест не прошел, программа считается составленной неверно. Тестовых систем, построенных по такому принципу, достаточно много. Одна из них разработана в ИГЭУ, используется при проведении областных олимпиад по программированию среди школьников. Принцип, несомненно, хорош, но вряд ли пригоден в учебном процессе, поскольку дает возможность оценивать знания только по двухбалльной системе: неудовлетворительно или отлично. Студенту можно нанести непо-

правимую психологическую травму, если все время выносить ему отрицательную оценку, особенно если он этого не заслуживает.

Рассмотрим простой пример. Пусть студенту поставлена такая задача: «Даны вещественные числа x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4. Точки с координатами (x_1, y_1) и (x_2, y_2) определяют первую прямую на плоскости, а точки с координатами (x_3, y_3) и (x_4, y_4) – вторую прямую. Составьте программу вычисления координат точки пересечения этих прямых». Не будем подробно рассматривать решение задачи, ограничимся лишь замечанием, что для полного решения необходимо кроме общего случая, когда обе прямые расположены под некоторыми углами к оси ординат, рассматривать два исключения: прямые параллельны и одна из прямых перпендикулярна оси. В том и другом случае, если они не обрабатываются программой отдельно, возникает деление на ноль, и соответствующие тесты не проходят. Какую оценку следует поставить студенту, если он не учел одно или два исключения? Имеющиеся системы в этих случаях всегда будут давать неудовлетворительную оценку, но объективно в данной задаче за каждое неучтенное исключение следует снимать один балл, то есть, если прошел основной тест, то оценка должна быть не ниже «3». Эти особенности необходимо учитывать при разработке автоматизированных тестов по программированию.

Если говорить о проверке знаний и навыков в выполнении отдельных этапов решения задачи, то в большинстве случаев можно ограничиться вопросами, предполагающими ответы перечисленных выше типов. Особого внимания заслуживает этап разработки алгоритма. Как оценить способность студента разрабатывать правильные алгоритмы? Вероятно, наилучший способ состоит в том, чтобы дать возможность студенту изобразить алгоритм, а затем его протестировать. Следовательно, система тестирования должна включать соответствующий инструмент.

К сожалению, известные на данный момент инструментальные средства по разработке автоматизированных тестирующих систем не учитывают проблем, связанных с оценкой знаний в области программирования, и не обладают необходимым инструментом. Поэтому на кафедре программного обеспечения компьютерных систем Ивановского государственного энергетического университета в течение ряда лет разрабатывается система автоматизированного тестирования знаний студентов, ориентированная, прежде всего, на дисциплины из области программирования.

Система отвечает практически всем требованиям, предъявляемым к инструментальным системам в классе задач автоматизированного обучения. Включает средства разработки тестов и средства проведения тестирования, причем тестирование может проводиться как в локальной сети, так и на удаленном компьютере. Поэтому программное обеспечение системы состоит из двух частей: серверной и клиентской (см. рисунок).

Сервер приложения содержит подсистемы, обеспечивающие функционирование системы. Ядром системы является подсистема тестирования, которая обеспечивает запуск процесса тестирования, его приостановку, завершение тестирования и ожидание подключения к серверу новых пользователей (студентов). Подсистема отображает список подключенных к серверу пользователей и их состояние (подключен, отключен, нет задания, задание принято, задание завершено). Кроме того, подсистема отвечает за формирование и отправку заданий активным клиентским приложениям, за получение и оценку результатов тестирования и рассылку этих результатов студентам.

Работа студента в системе начинается с регистрации. Студент вводит свою фамилию, имя, отчество, курс, группу и пароль. Данные сверяются со списком студентов, допущенных к работе с системой, в базе данных, что предотвращает несанкционированный доступ к ресурсам системы. Пароль необходим для обеспечения доступа студента к своему набору данных в базе, а также для защиты интересов студента при попытке входа в систему постороннего пользователя под его именем. Студент может работать в трех режимах: тестирования, самотестирования, обучения.

Работу в режиме тестирования обеспечивает отдельная подсистема тестирования, размещенная со стороны клиентского приложения. Она оповещает студента о поступившем задании, отображает задание с помощью Internet Explorer, обеспечивает заполнение ответов в соответствии с указаниями, сформулированными преподавателем, отправляет на сервер результаты работы, получает от сервера и отображает результирующую оценку. При необходимости подсистема тестирования клиента подключает редактор блок-схем или редактор текстов программ. Результаты тестирования сохраняются в базе данных. Причем сохраняется не только итоговая оценка, а вся информация полностью: дата и время тестирования, какие вопросы были получены студентом, как он на них отвечал, какую получил оценку. При необходимости можно восстановить весь ход тестирования.

В режиме самотестирования студенту предоставляется возможность самостоятельно выбрать дисциплину и тему для тестирования из имеющегося набора. Студенту направляются отдельные вопросы в заданном порядке или путем случайного выбора.

По окончании процесса тестирования подсистема генерации отчетов позволяет составить протокол для каждого студента. Если тестирование одновременно проходит группа студентов, подсистема по запросу сформирует ведомость результатов. Помимо этого подсистема позволяет сгенерировать SQL-запрос и получить выборку по интересующим параметрам, причем сту-

дент имеет доступ только к своим данным, преподаватель – к данным всех студентов.

Как известно, никакое тестирование невозможно без набора вопросов и заданий. Эту работу выполняет преподаватель с определенными для него правами доступа. Подсистемы, обеспечивающие создание новых тестов и обучающих модулей, расположены на сервере приложения, куда пользователь с правами студента доступа не имеет. Работа преподавателя начинается также с регистрации. Обязательно вводится пароль, предназначенный для защиты информации: каждый преподаватель может работать только со своими учебными курсами. Основными подсистемами, обеспечивающими работу преподавателя, являются редактор вопросов и редактор сценариев.

Редактор вопросов предоставляет преподавателю возможность создавать и редактировать вопросы в режиме визуального конструирования, размещая на экране форматированный текст, специальные элементы диалога, рисунки, гиперссылки. Каждому вопросу приписывается ряд дополнительных атрибутов: дисциплина и тема, назначение, тип, коэффициент сложности. Назначение вопроса используется системой при автоматическом формировании пакета вопросов и заданий студенту (билета) в соответствии с заданным сценарием тестирования. Так, вопросы, имеющие назначение «Экзамен», не будут использоваться при проведении других видов контроля знаний. Тип вопроса имеет значение для определения необходимости использования специальных подсистем для подготовки ответа и оценки его правильности. Вопрос типа «Алгоритм» требует загрузки у клиента редактора блок-схем, а вопрос типа «Программа» – редактора текстов программ, и оба типа вопросов требуют загрузки подсистемы тестирования программ на сервере.

Редактор блок-схем представляет собой графический редактор с предопределенным набором графических элементов, отображающих логические блоки структурного программирования. При составлении блок-схемы пользователю достаточно выбрать необходимые блоки, разместить их на поле экрана и связать их линиями потоков. Недопустимые связи контролируются редактором и не отображаются. Текстовое содержание блоков должно быть записано в соответствии с синтаксисом языка программирования Паскаль.

Редактор текстов программ является несложным текстовым редактором, обеспечивающим подсветку синтаксических конструкций языка Паскаль.

При создании вопроса преподаватель обязан указать правильный ответ или задать набор тестов для проверки работоспособности алгоритмов и программ, сопроводив их необходимыми атрибутами.

В целом работа с редактором вопросов проста, интуитивно понятна и не требует специальной подготовки. Вопросы сохраняются в отдельных файлах в формате HTML, что обеспечивает их компактность. Ссылки на файлы записываются в базу данных.

Редактор сценариев необходим для того, чтобы преподаватель мог задать правила тестирования. В сценарии он может определить количество вопросов и заданий, которые получит студент за один сеанс тестирования, темы, которые будут задействованы в тестировании, наличие заданий специальных типов (алгоритм, программа), установить ограничение по времени, возможность одномоментного дублирования вопроса на разных компьютерах, среднюю сложность вопросов для обеспечения создания примерно равных по сложности билетов. Далее по заданному сценарию подсистема тестирования сервера приложения автоматически генерирует контрольные билеты, используя имеющийся в базе данных набор вопросов, и отправляет их студентам для проведения тестирования. Сценарии сохраняются в базе данных с целью их многократного использования.

Изначально система проектировалась как система автоматизированного тестирования знаний, однако в процессе разработки выяснилось, что кроме контроля знаний целесообразно обеспечить возможность включения элементов обучения. Эта проблема в данной версии решена с помощью гиперссылок.

Для хранения и обработки данных используется СУБД MS SQL Server, где кроме таблиц с данными содержатся справочники. Функции для работы с базами данных обеспечивают возможность добавления как новых типов вопросов, так и их новых назначений. Таким образом, система представляется достаточно гибкой. Средствами СУБД решается и проблема защиты информации.

В целом система является достаточно удобным инструментом для разработки автоматизированных тестирующих программ. Но следует заметить, что эффективность тестирования и его объективность могут быть обеспечены только при наличии необходимого количества вопросов и заданий.

Список литературы

1. Нуждин В.Н., Кадамцева Г.Г. Стратегическое управление качеством образования: Учеб. пособие. – Иваново, 2003. – 88 с.

2. Афанасьев В.В., Тыщенко О.Б., Афанасьева И.В. Оценка уровня усвоения знаний с применением компьютерной техники: Тез. докл. I Всерос. науч.-техн. конф. «Компьютерные технологии в науке, проектировании и производстве», часть V. – Нижний Новгород, 1999. – С. 15.

3. Справка по результатам проведения пробного единого государственного экзамена по русскому языку в Санкт-Петербурге (22 апреля 2005) / Багге М.Б., Казаков В.П.: www.ege.spb.ru.

4. Пантелеев Е.Р. Среда разработки программ дистанционного обучения и профильного тестирования ГИПЕРТЕСТ: логистическая модель и архитектура // Информационные технологии. – 2001. – №5. – С. 30–36.

5. Степанцов В.А. Комплексный подход к разработке автоматизированных обучающих систем // Международный конгресс конференций «Информационные технологии в образовании» (ИТО-2003). Секция 2: ИКТ в учебном процессе: www.ict.edu.ru.

6. СДО ПРОМЕТЕЙ 4.X. Руководство тьютора: www.prometeus.ru.

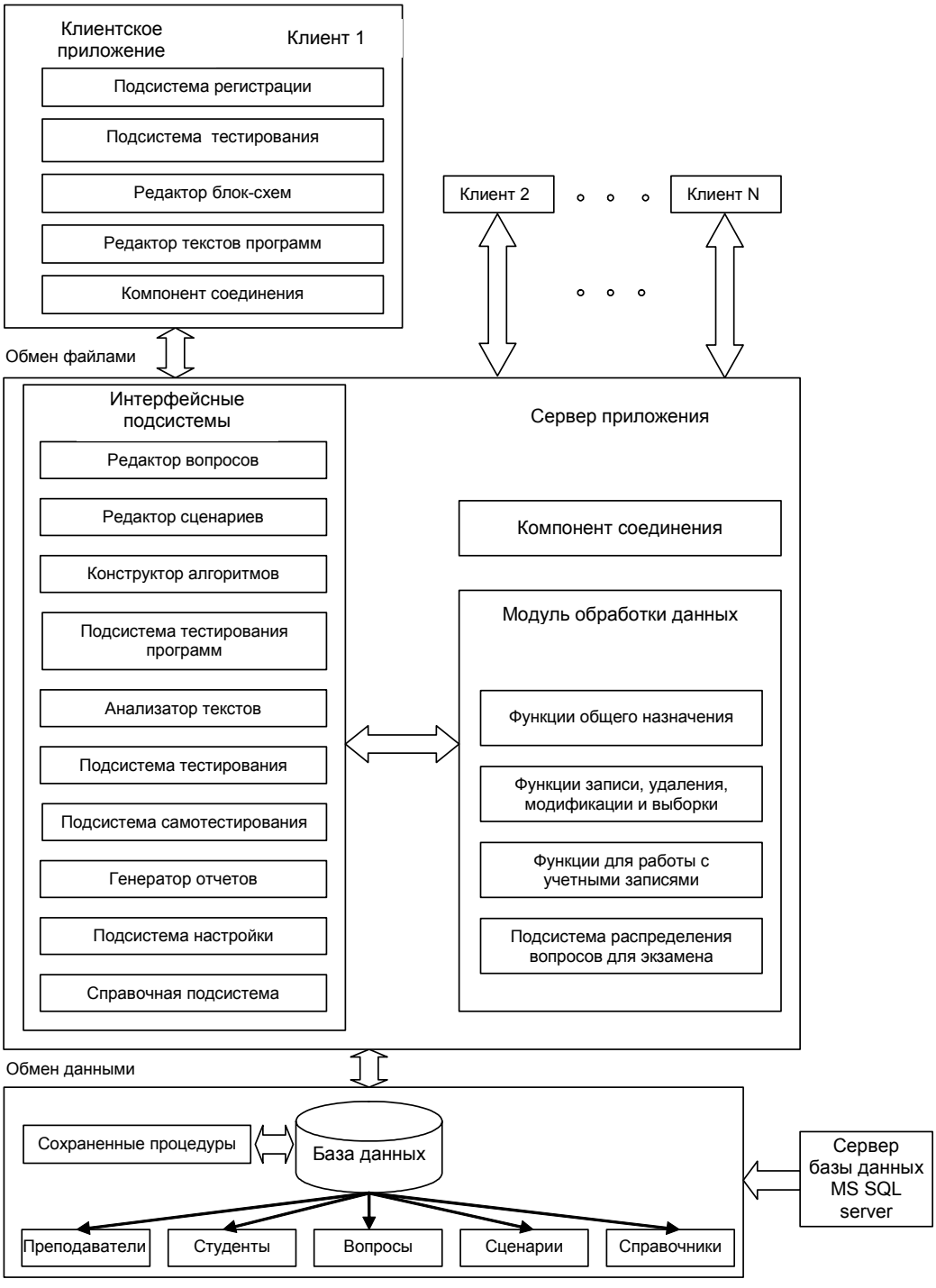

Рис. 1. Структура системы автоматизированного тестирования