УДК 378:681.3(06)

А.В. Романенко

Блочная архитектура системы контроля знаний

Рассмотрен новый подход в проведении компьютерного контроля знаний и принцип использования блочной архитектуры. Приведен пример использования блоков.
Ключевые слова: контроль знаний, методы контроля знаний, блочная архитектура программы, интерпретатор кода.

Введение

Существует множество различных систем контроля знаний. Одни разработчики используют уже освоенные принципы проектирования систем контроля знаний, другие разрабатывают собственные решения. Так или иначе существующие системы контроля знаний можно отнести к какой-либо группе.

Одни системы не учитывают в процессе тестирования уровень знаний и подготовки обучаемых и используют самые простые методы контроля знаний [1–3]. В таких системах имеется база вопросов и ответов к ним. В процессе тестирования из базы случайным образом выбирается один из вопросов, затем также случайно выбирается второй и т.д. В подобных системах порядок выбора задания из базы может формироваться различными способами: выбор происходит случайным образом из всего множества заданий; выбор равномерно распределен по определенным группам заданий (например, различные темы дисциплины). Плюсы таких систем очевидны – простота разработки как самой системы тестирования, так и базы вопросов. В силу данных плюсов такой тип систем самый распространенный, но не значит, что самый объективный. Данные системы не адаптивны к пользователям, что не позволяет учесть уровень знаний и подготовки обучаемого. Помимо этого, конечный набор вопросов предполагает повторное их использование, что приводит к появлению баз ответов к тестам, и тест теряет свою значимость (в продаже появляются базы готовых ответов).

Другие системы обладают некоторой адаптивностью, что достигается путем использования более сложных методов контроля знаний [4–6]. Самый простой и распространенный метод – выбор задания по сложности. В данном методе выбор следующего задания определяется текущим уровнем знаний пользователя. Если этот уровень низкий, то и задания будут выбираться несложные, и наоборот. Если взять достаточно длинный ряд заданий, то результаты тестирования пользователя в первой и второй группах систем будет примерно одинаков. Тогда зачем нужно использовать более сложные методы? Необходимость их использования обусловлена возможностью добавления некоторых особенностей в движении контроля (перехода от одного задания к другому) – изменение длины пути контроля (количество предложенных заданий). Если пользователь находится на высоком уровне (отвечает на сложные вопросы) в течение определенного времени, то можно сделать вывод, что он обладает достаточными знаниями и может получить оценку. Либо если пользователь находится определенное время на низком уровне, то можно сделать вывод о плохом уровне подготовки обучаемого и предложить ему повторное изучение теоретического материала по теме тестирования. В итоге в зависимости от ответов длина пути контроля может изменяться. В данных методах каждое задание должно иметь определенный вес, который будет влиять на выбор этого задания из множества других. С точки зрения разработчика данные системы также достаточно просты в разработке, что должно влиять на их распространение, но, тем не менее, в действительности системы первой группы много более распространены.

Развитие контроля знаний привело к появлению более сложных методов [7]. Тут, согласно идеям разработчиков, появляются различные модификации первых двух групп методов либо же особые методы, которые чаще всего привязаны к предметной области тестирования (например, тестирование пилотов в специальной системе, которая имитирует поведение самолета). Можно также выделить метод с использованием графа переходов [8]. Данный метод сильно адаптирован к пользователю, так как позволяет оценивать ответ как в целом, так и в частном, задавать наводящие вопросы и т.д. Соответственно, сложность разработки таких заданий гораздо выше, чем в предыдущих методах. Использование специфичных методов контроля ограничивает область применения системы кон-

троля знаний в пределах собственной специфики. В настоящее время такие системы не получили распространения.

Как и в случае самых простых систем, существует вероятность перебора всех заданий и составления банка ответов к ним. Для решения этой проблемы используют генераторы заданий [9, 10]. Генераторы позволяют определять задание и решения в процессе работы системы контроля знаний. Генераторы могут применяться в различных частях системы контроля знаний, но чаще всего их используют для проведения выборки вариантов заданий и ответов к ним и при генерации параметров какой-либо задачи. Помимо этого, генератор также увеличивает разнообразие заданий за счет перестановки частей задания и ответов. Генераторы обычно привязаны к системе разработки, и возможность их использования в других системах маловероятна. Также разработка генераторов достаточно сложна с точки зрения проектировки и реализации. Поэтому если в системах и используют генераторы, то чаще всего разработчики ограничиваются генерацией набора вариантов ответов из множества.

Выбор методов контроля разрабатываемой системы контроля знаний, применения генераторов относится к обязанностям разработчиков. Если в процессе разработки или эксплуатации системы возникнет потребность внесения изменений в процесс контроля, то система должна будет подвергнуться переработке, и скорее всего значительной. Это обусловлено тем, что в некоторых случаях не требуется сложной модели поведения системы контроля, в таких ситуациях прибегают к простым методам. Может возникнуть потребность в применении генератора в уже действующей системе. Таким образом, необходимо иметь систему, которая позволяла бы оперировать с различными методами контроля, генераторами, имела возможность модификации модели тестирования. Такая система разработана автором.

Блочная структура

Для осуществления поставленной задачи было принято решение разбиение процесса работы системы контроля знаний на подпроцессы (блоки). Каждый блок представляет собой некоторый набор параметров и процедуру, которая может быть вызвана из другого блока. Каждый блок обладает своими внутренними параметрами, но также может оперировать и глобальными. Результатом работы блока служит вызов процедуры другого блока, выбор которого определяется внутренним устройством. В результате получаем набор связанных между собой блоков. Таким образом, заменяя один или несколько блоков, можно кардинально изменить поведение системы контроля.

В результате работы был определен минимальный набор блоков. Блоки можно разделить на два вида: управляющие и визуальные. Соответственно, первые служат для управления процессом тестирования, а вторые – для отображения какой-либо информации и для взаимодействия системы с пользователем. К управляющим блокам относятся:

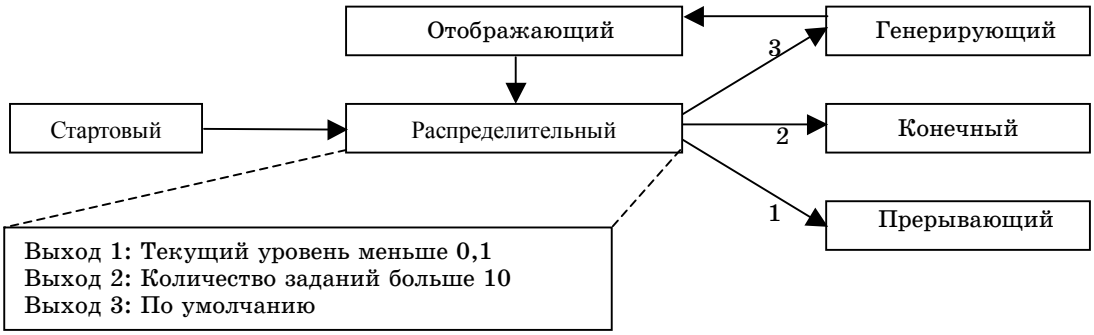
- *стартовый.* В данном блоке осуществляется инициализация параметров системы. Этот блок может использоваться более одного раза, если возникает необходимость обнуления состояния системы. К параметрам системы относятся: длина пути, количество выданных заданий, количество верных ответов, текущий уровень пользователя, время тестирования и т.д.;
- *генерирующий.* Данный блок отвечает за выборку задания из базы, согласно текущему состоянию системы, то есть в зависимости от параметров. Помимо выборки, происходит компоновка задания и ответов к нему с помощью соответствующих генераторов, которые были применены при составлении задания;
- *распределительный.* Данный блок предназначен для вызова процедуры одного из связанных с ним блоков, в зависимости от параметров системы.

К визуальным блокам относятся:

- *отображающий.* Работа данного блока заключается в непосредственном взаимодействии с пользователем. В результате работы данного блока на экран выдается сформированное ранее задание и ожидаются действия пользователя. В зависимости от действий пользователя будут изменены параметры системы;
- *конечный.* Данный блок активируется, когда совокупность параметров системы может определить оценку результатов тестирования пользователя. Пользователь получает результат, и работа системы прекращается;
- *прерывающий.* Данный блок активируется, когда совокупность параметров системы позволяет судить о недостаточных знаниях пользователя. В данном случае пользователю предлагается перейти к лекционным материалам для повторного изучения.

Приведенного набора блоков достаточно для обработки практически любой ситуации в контроле знаний. Тем не менее для предоставления более гибкого использования системы был введен неопределенный (пользовательский) блок. Он может так же, как и другие блоки, быть включен в цепочку, но заранее никаких действий не производит. Для управления его работой был использован разработанный автором для данной системы интерпретатор кода. Путем написания соответствующего кода на языке интерпретатора можно управлять поведением блока и всей системы в целом. Данный язык поддерживает основной набор функций и управляющих конструкций: арифметические операции, вызов функций, 2 цикла (с предусловием, с постусловием), условный оператор, использование ранее написанного кода (модули). Пользователь может определять внутренние переменные блока, которые будут использованы в процессе его работы и уничтожены после выполнения. Помимо этого пользователю доступны глобальные переменные, которые характеризуют состояние системы.

Приведем на рисунке пример возможной схемы контроля, которая использует простой метод контроля.



Пример схемы контроля

Рассмотрим работу приведенной схемы. Начало работы начинается с вызова процедуры стартового блока – происходит инициализация переменных. Затем происходит вызов процедуры распределительного блока, в которой идет проверка условий выходов по порядку (тем самым можно задавать приоритет активации (вызова соответствующей процедуры) связанных блоков, когда условия удовлетворяют одновременно нескольким выходам). Если пользователь достиг низкого уровня, то происходит прерывание контроля и предлагается повторное изучение лекционных материалов. Если пользователь решил 10 заданий, то происходит вызов процедуры конечного блока, в которой будет определена оценка пользователя. Если первые два условия не выполнились, то будет активирован генерирующий блок, который будет выбирать задание из базы. Алгоритм выбора может быть различным. В данном случае метод контроля простой, и выбор осуществляется случайным образом. После выбора задания активируется отображающий блок, который выводит текст задания и управляющие элементы на экран и ожидает действий пользователя. После выбора ответа пользователем заново активируется распределительный блок, и работа продолжается.

Реализация

Так как для системы контроля знаний был разработан интерпретатор, то было принято решение использовать его для реализации схем. При добавлении в схеме контроля блока формируется код на языке разработанного интерпретатора. Данный подход обеспечивает единообразие в описании схемы контроля. Как уже было сказано выше, существует неопределенный блок, поведение которого может быть определено пользователем на разработанном языке. По сути, все блоки единообразны, только каждый имеет свой, уже заданный код на разработанном языке. Помимо этого, существует возможность вызова функций из динамических связываемых библиотек. Это дает возможность использовать функции, описанные другими разработчиками ранее. Также поддерживается запуск сторонних приложений, причем происходит контроль кода завершения этих приложений для изменения параметров системы. Например, отображающий блок запускает программу, которая выводит задание на экран.

Использование языка для описания схемы контроля позволяет вносить дополнительные возможности в управление блоками. Так, например, в распределяющий блок можно добавить циклические конструкции или в стартовый внести дополнительные глобальные переменные и т.д.

Система разрабатывалась в рамках автоматизированного комплекса разработки электронных учебных курсов EduCAD [11].

Применение

Разработанная система применялась для создания системы контроля знаний для электронных учебных комплексов по дисциплине «Физика» в двух частях, предназначенных для использования в учебном процессе студентами ТМЦ ДО [12]. В данных электронных учебниках для контроля и самоконтроля знаний методистами были составлены списки задач по каждой теме дисциплины. Задача представляет собой описание, для отображения которого используются язык гипертекстовой разметки системы EduCAD, набор параметров и одна или несколько конечных формул ответа. Параметры описываются интервалами допустимых значений, при этом значения одних параметров могут ограничивать значения других. Для определения значения параметров был использован соответствующий генератор, который входит в набор генераторов разработанной системы.

Литература

1. Соколова М.В. Программный продукт «Exam Tool» – новое поколение систем оценки знаний / М.В. Соколова, С.С. Пучнин, А.В. Иванов // Новые информационные технологии: Матер. Седьмой междунарю студ. школы-семинара (Крым, май 1999 г.). – Судак, 1999. – С. 152–153.

2. Бабешко В.Н. Реализация системы контроля знаний при дистанционном обучении на базе Lotus Notes / В.Н. Бабешко, М.И. Нежурина // Там же. – С. 157–158.

3. Jennifer R. Parham An assessment and evaluation of computer science education // Journal of Computing Sciences in Colleges. – 2003. – Vol. 19, Issue 2 (December 2003). – P. 115–127.

4. Аавтоматизированная система контроля знаний «1th-ТЕСТ» [Электронный ресурс]. – Режим доступа: http://www.1th.ru, свободный.

5. Автоматизированная система Auto Control [Электронный ресурс]. – Режим доступа: http://www.fbit.ru, свободный.

6. Niklaus Wirth. Computing science education // ACM SIGCSE Bulletin. – 2002. – Vol. 34, Issue 3 (September 2002). – P. 1–3.

7. Гусева А.И. Методы адаптивного контроля знаний // Информатика и образование. –2003. – №7. – С. 108–111.

8. Зайцева Л.В. Модели и методы адаптации к учащимся в системах компьютерного обучения // Educational Technology & Society. – 2003. – № 6(3). – С. 204 – 212.

9. Кручинин В.В. Генераторы в компьютерных учебных программах. – Томск: Изд-во Том. ун-та, 2003. – 200 с.

10. Романенко А.В. Генераторы в контроле знаний // Электронные средства и системы управления. Опыт инновационного развития: Докл. Междунар. Науч.-практ. конф. (Россия, Томск, 31 октября – 3 ноября 2007 г.). – Томск: В-Спектр, 2007. – Ч. 2. – С. 295–297.

11. Романенко В.В. Развитие автоматизированного комплекса разработки компьютерных учебных пособий EduCAD // «Автоматизированные системы обработки информации, управления и проектирования»: Сборник трудов ТУСУР. – Томск: ТУСУР, 2002. – Т. 7. – 192 с.

12. Козырев А.В. Мультимедийный учебник по механике, молекулярной физике и термодинамике. / А.В. Козырев, А.А. Мицель, В.В. Романенко, А.В. Романенкои др. // Современное образование: инновационный потенциал «умной экономики» России. – Томск: Изд-во ТУСУР. – 2007. – С. 164–165.

Романенко Александр Васильевич
Аспирант каф. автоматизированных систем управления ТУСУРа
Тел.: +7-923-415-92-29
Эл. почта: fess@ms.tusur.ru

A.V. Romanenko

The use of block architecture in knowledge testing

A new approach for computer-based knowledge test is considered in the paper. The principle of block architecture using is analyzed. Example of the use of blocks is shown.

Key words: knowledge test, methods of knowledge test, block architecture of program, code interpretation.