Н. В. Сергиевич, М. И. Полоз

АВТОМАТИЗАЦИЯ ПРОВЕРКИ РЕШЕНИЙ ЗАДАЧ ПО ПРОГРАММИРОВАНИЮ

Алгоритмический стиль мышления предполагает высокую мыслительную активность, сопоставимую, а временами и превосходящую по интенсивности и сложности даже математическое мышление [1]. Хорошо развитое алгоритмическое мышление предполагает хорошо

развитое владение навыками анализа и синтеза информации. Один из теоретиков программирования Д. Кнут, рассматривая алгоритмическое мышление, выделяет следующие основные категории, характерные для наиболее типичного мыслительного процесса программистов [2]: некоторая связь – операции с формулами; сильное использование – отражение действительности, сведение к более простому случаю, абстрактные рассуждения, использование структур данных, алгоритмы.

Для успешного освоения раздела «Алгоритмизация и программирование», помимо знания основных алгоритмов, учащимся нужно иметь хорошо развитое абстрактное мышление, владеть навыками анализа и синтеза и, кроме того, обладать хорошими знаниями по основным естественнонаучным дисциплинам – алгебре и началам анализа, геометрии, физике. При этом одной из самых важных и сложных задач является проверка объема и, главное, качества усвоенного материала.

В программировании, где ***критерием правильности работы программы*** *является получение верных результатов на всех возможных наборах входных данных, ограниченных исходной областью определения задачи*, наиболее распространённым способом оценки является проверка разработанного учащимися алгоритма и его программной реализации на одном из языков программирования на некотором наборе входных данных, именуемом *тестом* [3].

Ранее тестирование и оценка решений выполнялись вручную, что являлось весьма трудоёмким занятием. На различных соревнованиях по программированию именно здесь рождались многие судейские ошибки. Помимо человеческого фактора, существуют и другие недостатки этой формы проведения соревнований. Например, нельзя сдавать решение до окончания олимпиады, а после сдачи – вносить в него изменения. Очевидным решением данной проблемы является использование компьютера для автоматизации проверки решений.

Большинство из разработанных в настоящее время так называемых *автоматизированных систем тестирования* (АСТ) решений задач достаточно узко специализированы и ориентированы непосредственно лишь на автоматизацию только проверки решений. Однако с учётом получающей в настоящее время все большее распространение дистанционной формы обучения подобная система тестирования, основанная на использовании web-технологий, позволила бы учителям школ и преподавателям вузов при изучении дисциплин, связанных с программированием, не только повысить качество сугубо контроля практических навыков учащихся, но и выйти на качественно новый уровень обучения. Ниже рассматриваются базовые принципы работы АСТ «MasterTest», ориентированной на работу как в локальной сети, так и в сети Интернет [4].

Процесс тестирования решения алгоритмической задачи на компьютере приблизительно можно описать следующим образом: составляется условие задачи, решить которую учащемуся требуется с помощью одного или нескольких изученных алгоритмов, причем в условии четко регламентируется результат, выдаваемый программой во всех возможных ситуациях, а также задаются области значений всех входных параметров. При этом стиль написания программы не учитывается, но строго отслеживается корректность ее работы.

Правильность работы программы оценивают по заранее подготовленному набору тестов, в котором каждый тест состоит из входных и выходных данных: программе даётся входной набор данных, а полученные на выходе результаты сравниваются с эталонными. Способы сравнения могут варьироваться. Например, это может быть побайтное сравнение файлов либо подстановка ответа в условие задачи.

Успешное прохождение полного набора тестов оценивается в 100 баллов (либо процентов). При этом удельный «вес» каждого теста может либо вычисляться как средний вес всех тестов набора, либо индивидуально оцениваться в зависимости от его сложности (если сложность тестов различна и/или прохождение данного теста принципиально).

Обычно наборы составляются так, чтобы наиболее комплексно протестировать программу, проверив ее корректную работу на всех возможных наборах входных данных, удовлетворяющих условиям поставленной задачи. При этом можно также учитывать и некоторые дополнительные ограничения, такие, как время исполнения программы, объём используемой оперативной памяти и др. Такой способ оценки результатов может быть достаточно просто автоматизирован.

Таким образом, *основная цель создания системы тестирования – полное исключение человека из процесса проверки с целью обеспечения быстрой, качественной и, главное, объективной оценки решения задачи.*

Можно выделить следующие основные требования, предъявляемые к подобной системе [4]:

1. *Независимость от платформы системы участника соревнования.*

В современном мире существует большое количество различных операционных систем, структура и принципы функционирования которых значительно отличаются. Разработка программного обеспечения, в частности, клиентской части, для какой-либо определенной платформы привело бы к потере значительной части потенциальных пользователей данной системы. Следовательно, программный продукт не должен зависеть от платформы клиента.

2. *Автоматическое тестирование в кратчайшие сроки.*

Передача функции проверки программе позволяет увеличить скорость тестирования, исключить человеческий фактор и тем самым повысить качество результата.

3. *Обслуживание нескольких пользователей одновременно.*

Одним из способов обеспечения этой возможности является включение в тестирующую систему двух автономных модулей – *ядра* и *серверной части*.

Сервер берет на себя функции общения с подмножеством клиентов системы. Именно он реализует независимость АСТ от платформы пользователя. Следует отметить, что самим процессом тестирования данный модуль заниматься не должен. В любой момент сервер должен быть готов принять поступившее от пользователя решение некоторой задачи и поставить его в очередь тестирования. Непосредственно же тестированием пришедших от пользователей решений занимается **ядро** тестирующей системы [4].

В общем виде вся последовательность действий, осуществляемых тестирующей системой при проверке программы пользователя, выглядит следующим образом:

1. Получение результата решения задачи – программы (алгоритма) пользователя.
2. Определение набора тестов для этой задачи.
3. Выполнение тестирования.
4. Формирование оценки.

При этом, во-первых, процедура передачи решения от пользователя к системе тестирования должна быть простой и, во-вторых, сама система тестирования должна быть фактически изолирована от пользователей с целью нормального функционирования и предохранения наборов тестов от просмотра пользователями.

Таким образом, помимо ядра и серверной части АСТ, упомянутых выше, для полноценного взаимодействия пользователя с системой требуется также наличие соответствующего *интерфейса*, который максимально упрощает и делает прозрачной процедуру общения пользователя с системой, но в то же время решает ряд сложных задач по существлению безопасности работы системы.

Одной из главных задач программных модулей, составляющих АСТ, является продуманный выбор инструментария, необходимого для создания данного программного обеспечения.

При разработке ядра и серверной части тестирующей системы [4]-[6] использовались язык C# и СУБД MySQL, а также следующие сторонние библиотеки:

- ***Система протоколирования*** log4net, являющаяся частью проекта «Apache Logging Services». Данная библиотека распространяется по свободной лицензии Apache License 2.0.
- ***Драйвер для подключения баз данных*** MySQL Connector/NET 5.1, распространяющийся по GPL лицензии.

Модуль, реализующий интерфейс пользователя и АСТ [3], был разработан с использованием следующего программного комплекса:

- Web-сервер Apache 2.00.52;
- Язык программирования Perl 5.8.6;
- СУБД MySQL 3.23.58.

Основными критериями выбора данных программ являлась их бесплатность и в то же время достаточно большая функциональность. Кроме того, они достаточно просты в использовании и имеют реализации, написанные для различных операционных систем.

Анализ существующих программных решений в области сетевых технологий и теории тестирования программ позволил сделать вывод о том, что, возможно, наиболее действенные результаты при создании АСТ может дать использование стремительно развивающихся в данное время web-приложений, и, в частности, технологии CGI (Common Gateway Interface) [7]. Выбор обусловлен несколькими причинами.

1. Существует достаточно большое количество различных инструментов, которые позволяют решать различные задачи с использованием данной технологии.
2. Работа приложений построена по технологии «Клиент-Сервер», предоставляющей необходимые возможности по защите данных, находящихся на сервере [8].
3. Технология базируется на протоколе HTTP, который является, в свою очередь, надстройкой над семейством протоколов TCP/IP. Вследствие этого пользователь практически не стеснен в выборе программы-клиента (браузера), а главное – в выборе операционной системы [9].
4. Простота доступа к нужной информации.

Для обеспечения возможности взаимодействия ядра и серверной части системы необходимо разработать соответствующий механизм. Поскольку ядро системы работает с очередью задач, а серверная часть активизируется по запросу от клиента, то прямой связи между ними в этом случае быть не может. Следовательно, нужно искать возможность обмена информацией без прямого соединения, но с использованием некоторого промежуточного «хранилища». Возможно, наилучшим вариантом для этого является использование базы данных.

Как известно, при разработке любого программного продукта необходимо четко определить его структуру. Наличие данного этапа в разработке программного обеспечения обусловлено некоторыми объективными причинами:

1. Существуют различные алгоритмы, которые решают одни и те же поставленные задачи. Одни из них отличаются высокой скоростью решения, другие – функциональностью, третьи – простотой разработки [10]. Следовательно, зная перечень основных проблем, можно выбрать наиболее эффективный алгоритм их решения, используя для оценки все эти критерии.
2. В процессе разработки программы с изначально некорректной структурой могут возникнуть непреодолимые проблемы в решении поставленных задач, в некоторых особых случаях способные и вовсе приостановить разработку программного продукта.

Как уже говорилось ранее, СУБД в данном случае является не только хранилищем данных, но и одним из основных механизмов взаимодействия автоматизированной системы тестирования и web-интерфейса. Во-первых, система тестирования является многопользовательской, следовательно, помимо всего прочего, в БД также необходимо хранить информацию и обо всех пользователях. Второй важной составляющей является информация о задачах, которые будут предоставлены учащимся для решения. Для работы с АСТ также требуется хранить и сведения о поступивших на тестирование решениях и, соответственно, результатах тестирования предыдущих решений.

Выше был описан минимальный набор данных для нормального взаимодействия системы тестирования и web-интерфейса. В процессе разработки программы возникла необходимость в хранении также еще и дополнительной информации, необходимой для удобного администрирования системы, создания дружественного интерфейса и получения оперативной статистики по решениям. Поскольку эти данные требуются исключительно для функционирования web-модуля, они не накладывают никаких ограничений на работу системы тестирования.

Инфологическое и логическое проектирование базы данных АСТ на основании всех поставленных требований и, далее, физическое проектирование БД уже непосредственно для реляционной СУБД MySQL позволило определить структуру базы данных системы, схема которой приведена на рисунке.

В БД выполнена частичная нормализация отношений. В частности, все таблицы (отношения) приведены к нормальной форме Бойса-Кодда (или усиленной третьей нормальной форме).

Во всех таблицах есть первичные индексы (помечены флагом PK – primary key), необходимые для однозначного определения любой записи таблиц. В некоторых таблицах добавлены вторичные индексы (по одному или нескольким полям) для ускорения работы операций, связанных с выборкой данных (выделены на схеме «жирным» шрифтом). Вторичные индексы добавлялись только в тех случаях, когда по данному полю или группе полей осуществлялся отбор данных. Следствием добавления вторичных индексов стала значительно возросшая скорость обработки некоторых запросов к БД.

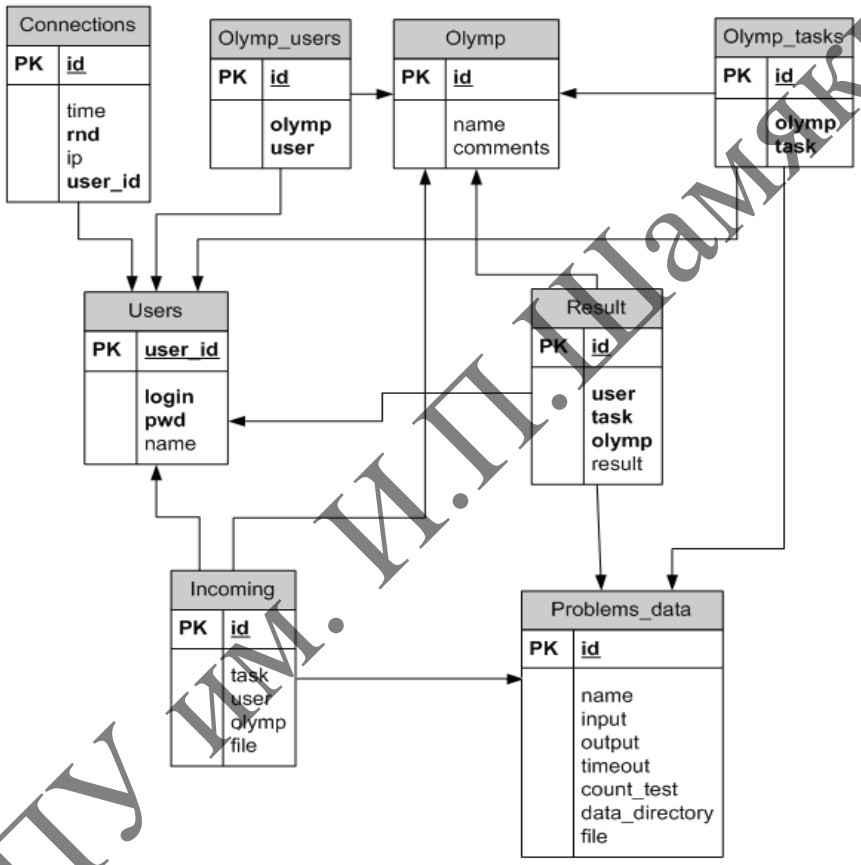

Рисунок – Структурная схема базы данных АСТ «MasterTest»

Основными объектами базы данных являются так называемые «плоские» таблицы. Каждая из таблиц либо хранит информацию о каком-либо множестве объектов (справочники), либо содержит сведения, характеризующие некоторую совокупность объектов, описанных в других таблицах (сводные таблицы).

В справочниках хранится информация об объектах, а также уникальный ключ, который позволяет однозначно идентифицировать объект. В нашей базе данных имеются три справочника: Users, Problems_data, Olymp. В них хранится информация о пользователях, задачах

и курсах соответственно, зарегистрированных в системе. Курсы объединяют в себе подмножество различных задач комплекса, предлагаемых для решения, и подмножество пользователей, которые будут их решать.

Забрав решение из очереди, ядро тестирует его и формирует соответствующий протокол, сохраняя всю необходимую информацию о решении в базе данных. При таком подходе ядро тестирующей системы избавлено от необходимости заботиться о входящих данных. Этим занимается серверная часть, предоставляя ядру решение пользователя для проверки. Именно она заботится о регистрации пользователей, подписке их на курсы, получении условий задач, обеспечении возможности отсылки решений.

Литература

1. Сергиевич, Н.В. О преподавании алгоритмизации и программирования в средней школе / Н.В. Сергиевич // Инновационные технологии обучения физико-математическим дисциплинам : материалы Междунар. науч.-практ. интернет-конф., 27-31 окт. 2008 г., г. Мозырь / редкол.: В.В. Валетов (отв. ред.) [и др.]. – Мозырь : УО МГПУ им. И.П. Шамякина, 2008. – С. 144–147.

2. Knuth, Donald E. Algorithms in modern mathematics and computer science / Donald E. Knuth. – Lecture Notes in Computer Science, 122. – Berlin : Springer, 1981. – P. 82–89.

3. Лопато, В.М. О разработке автоматизированной системы тестирования / В.М. Лопато // Инновации-2004 : материалы XI Респ. студ. науч.-практ. конф., 22 апреля 2004 г., Мозырь : в 2 ч. – Мозырь : УО МГПУ, 2004. – Ч. 1. – С. 89.

4. Лещенко, В.В. О подходе к реализации тестирующего модуля в автоматизированной системе тестирования / В.В. Лещенко // Инновации-2004 : материалы XI Респ. студ. науч.-практ. конф., 22 апреля 2004 г., Мозырь : в 2 ч. – Мозырь : УО МГПУ, 2004. – Ч. 1. – С. 89.

5. Лещенко, В.В. Автоматизация тестирования решений задач по программированию / В.В. Лещенко // Инновации-2005 : материалы XII Респ. студ. науч.-практ. конф., 28 апреля 2005 г., Мозырь : в 2 ч. – Мозырь : УО МГПУ, 2005. – Ч. 1. – С. 94.

6. Лещенко, В.В. Выбор средств реализации автоматизированных систем тестирования / В.В. Лещенко // Материалы XIV Республиканской студенческой науч.-практ. конф., 21 апреля 2007. – Мозырь : УО МГПУ, 2007.

7. Хокинс, С. Администрирование Web-сервера Apache и руководство по электронной коммерции / С. Хокинс. – Киев : Вильямс, 2001. – 336 с.

8. Цимбал, А.А. Технология создания распределенных систем. Для профессионалов / А.А. Цимбал, М.Л. Аншина. – Санкт-Петербург : Питер, 2003. – 576 с.

9. Паркер, Т. TCP/IP. Для профессионалов / Т. Паркер, К. Сиян. – Санкт-Петербург : Питер, 2004. – 864 с.

10. Кормен, Т. Алгоритмы: построение и анализ / Т. Кормен, Ч. Лейзерсон, Р. Ривест. – 2-е изд. – М. : Вильямс, 2007. – 1296 с.