для многослойного персептрона. Разрешение блоба равно 30x30 пикселей, каждый из которых содержит три компонента(R, G и B). Следовательно, входной вектор признаков будет состоять из 2700 , что неприемлемо для работы системы в реальном времени. Необходимо сократить количество компонентов входного вектора признаков следующим образом. Общее количество входных узлов персептрона – 63. Значения для 3 узлов – это нормированные усредненные максимальные значения пикселей MR, MG и MB, для 30 – полученные из вертикальной гистограммы, и для оставшихся 30 – из горизонтальной. Последняя часть – распознавание образов, реализована с помощью обученной нейронной сети с прямым распространением сигнала. Многослойные нейронные сети являются одной из новых находок и нашли широкое применение во многих приложениях[2]. Обучение персептрона проводится с помощью алгоритма обратного распространения ошибки. Для обучения нейронной сети была использована база дорожных знаков IMM[3].

Тестирование точности и быстродействия разработанной системы показало что из 48 знаков, правильно определены – 44, количество неправильно определенных – 2, пропущенных – 2. Среднее время обработки кадра равно 53 мс.

Таким образом, была разработана система фотофиксации дорожных знаков и их распознавания на основе многослойного персептрона. В дальнейшем планируется адаптация данной системы для работы на мобильном устройстве (смартфоне) на базе операционной системы Google Android.

Список использованных источников:

1. Журавель И.М. Краткий курс теории обработки изображений. – М., 1999
2. L. Fausett – Fundamentals of Neural Networks Architectures, Algorithms, and Applications / Prentice Hall Upper Saddle River – New Jersey, 1994.
3. Institut für Neuroinformatik, Dataset for The Traffic Sign Recognition Benchmark. [Электронный ресурс] – Режим доступа: http://benchmark.ini.rub.de – Дата доступа: 22.02.2015

СИСТЕМА АВТОМАТИЗАЦИИ ВЫДАЧИ И ПРОВЕРКИ ЛАБОРАТОРНЫХ РАБОТ ПО АЛГОРИТМИЗАЦИИ И ЯЗЫКАМ ПРОГРАММИРОВАНИЯ

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Костенич А. М.

Костенич Д. М. – магистр технических наук

Степень интеграции информационных технологий в учебный процесс в текущей системе образования все еще довольно низка, При проводимых реформах остается открытым вопрос автоматизации выдачи и проверки лабораторных работ, в том числе и по предметам, охватывающим области алгоритмизации и языков программирования.

Именно такую задачу решает разрабатываемая система. В базе данных хранятся условия задач, распределенные по определенным предметам и курсам, с соответствующими им тестовыми значениями, которые представляют собой словарь вида: {входное_значение1, входное_значение2…: выходное_значение}, наподобие того, что можно встретить в ACM подобных системах.

Каждый студент регистрируется в системе и по ходу выполнения им задач получает новое случайное задание. Целью студента является создание такой программы, которая пройдет все имеющиеся тесты. В таком и только таком случае задача считается выполненной, за что студент получает базовую оценку.

Роль преподавателя в таком процессе заключается в том, что он сам устанавливает критерии повышения оценки. Например, создание пользовательского интерфейса над прошедшим тесты “ядром” программы или покрытие исходного кода модульными тестами для получения наивысшей оценки. Преподаватель может уделить больше времени на анализ качества исходного кода или на теоритические области предмета. Помимо этого, такая система изначально заставляет студентов думать о структуре исходного кода, выделяя тестируемые участки своих программ в отдельные модули, не связанные с остальными участками кода.

К сожалению, подобный подход к тестированию не позволяет проводить автоматическую проверку лабораторных работ по предметам, не связанным с программированием, однако уже только на них можно ощутить насколько разумнее расходуется время студентов и преподавателя.

Система состоит из четырех логически связанных модуля: модуль учебного процесса, отвечающий доступ к текущему прогрессу для каждого студента, модуль обучения, отвечающий за выдачу лабораторных работ, модуль проверки заданий и модуль планирования, который ответственен за распределение выполняемых системой задач.

Перед разработкой продукта был проведен анализ существующих инструментов и был сделан выбор в пользу языка программирования Ruby и фреймворка Ruby on Rails, которые позволяют писать программный код с большей скоростью, одновременно проводя его полное покрытие модульными тестами.

Архитектура проекта изначально подразумевает под собой возможные расширения, такие как мобильные приложения для современных операционных систем и миграцию к одностраничному веб-приложению, поэтому был разработан слой доступа к данным и вынесен в отдельную веб-службу (Web API) на основе фреймворка Grape.

Планирование задач в системе выполняется на основе системы обмена сообщениями RabbitMQ, благодаря которой исчезает проблема жесткой связности модулей, появляется устойчивость к высоким нагрузкам и возможность расширяемости разрабатываемого продукта.

Новизна продукта заключается в том, что до текущего момента не встречались случаи внедрения подобных технологий в белорусскую систему образования, с их последующей интеграцией, базирующейся на принципах простоты, открытости и дружелюбности к использующим его пользователям.

Список использованных источников:

1. RabbitMQ in Action / Manning. - Shelter, New York, 2012.
2. Programming Ruby / The Pragmatic Bookshelf. - Dallas, Texas, 2009.
3. Agile Web Development with Rails / The Pragmatic Bookshelf. - Dallas, Texas, 2014.
4. Grape wiki [Электронный ресурс]. - Электронные данные. - Режим доступа : https://github.com/inridea/grape/wiki.
5. ACM homepage [Электроонный ресурс]. - Электронные данные. - Режим доступа : http://icpc.baylor.edu/welcome.icpc.

ОЦЕНКА ПОДХОДОВ К РАЗРАБОТКЕ КРОССПЛАТФОРМЕННОГО ГРАФИЧЕСКОГО ИНТЕРФЕЙСА

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Куница Е. Ю., Макоед В. Н.

Искра Н. А. – ассистент

На сегодняшний день существует большое количество инструментов разработки графического интерфейса пользователя (Graphical User Interface, GUI), предлагающих создание программ на различных языках программирования и реализующих различные подходы к созданию интерфейса. Существуют стандартизированные наборы примеров программ, представляющие собой решения типовых задач, возникающих при разработке графического интерфейса [1]. Кроме того, определены метрики, позволяющие выполнить объективное сравнение технологий и подходов к разработке GUI [2].

В рамках исследования были подробно изучены две задачи: CRUD (рис. 1) и Circle Drawer (рис. 2). Были спроектированы и разработаны программы на языках программирования C++ и Java на платформах QT [3] и JavaFX [4] соответственно. Оба подхода являются кроссплатформенными, что позволило успешно протестировать приложения на операционных системах Windows, Linux и OS X. Была произведена оценка скорости и удобства разработки решений на данных платформах.

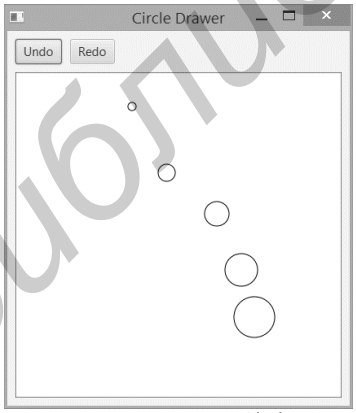

Рис. 1 – окно программы Circle Drawer

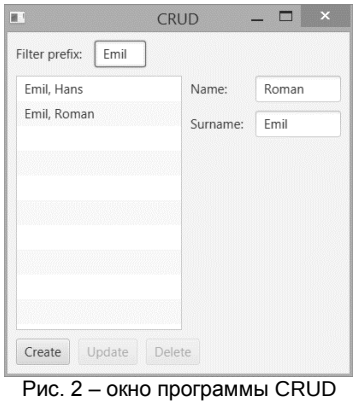

Рис. 2 – окно программы CRUD

При разработке программы для рисования кругов были решены следующие задачи: реализация функциональности отмены действия и повторения отменённого действия (функции Undo и Redo); рисование примитивной графики; работа с диалоговыми окнами.

При разработке CRUD были решены следующие задачи: разделение бизнес-логики и логики интерфейса; реализация изменяющихся элементов интерфейса; нетипичная компоновка элементов интерфейса.

Сходства и различия в возможностях платформ, обнаруженные при разработке приложений, приведены в таблице 1.