УДК

РАЗРАБОТКА КРОССПЛАТФОРМЕННОЙ АВТОНОМНОЙ ТЕСТИРУЮЩЕЙ СИСТЕМЫ

Зганяйко Д.О., Милюков В.В.
Таврический национальный университет им. В.И. Вернадского
Кафедра компьютерной инженерии и моделирования
E-mail: zdo.str@gmail.com

Аннотация:
Зганяйко Д.О., Милюков В.В. Разработка кроссплатформенной автономной тестирующей системы. *Проведен анализ существующих программных решений для проведения компьютеризированного тестирования. Определены требования к системе, разработана архитектура, спроектированы и разработаны ключевые программные узлы системы.*

Общая постановка проблемы

Тестирование занимает важное место в современной системе образования. Непрерывно возрастающие масштабы и сложность тестов ставят перед преподавателями проблему обеспечения качества, скорости и удобства проведения тестирования.

Существующие программные решения, которые разработаны с целью решить данную задачу, не позволяют нам в полной мере использовать их. В связи с этим было принято решение о разработке собственной системы тестирования.

Исследования

Программные продукты, имеющиеся на рынке, не позволяют в полной мере реализовать их потенциал по следующим причинам:

1. Развертывание этих систем сопряжено с определенными сложностями, как то: требование к наличию у преподавателя умения устанавливать и конфигурировать данное программное обеспечение. Также встает вопрос о наличии кроссплатформенной поддержки в имеющихся программах, что, в свою очередь, также осложняет процесс развертывания искомого ПО. Эти факты создают перед преподавателями дополнительные сложности, что в значительной степени уменьшает эффективность проводимого тестирования.
2. Ресурсоемкость существующих программных продуктов делает в значительной степени малоэффективным запуск их на устаревших компьютерах или системах со слабой конфигурацией.

Исходя из перечисленных недостатков существующих на рынке программных систем было принято решение о разработке своей собственной тестирующей системы. Последняя должна обладать, в свою очередь, следующими характеристиками (требования к системе):

1. Кроссплатформенность. Система должна качественно работать на всех наиболее распространенных операционных системах, таких как GNU/Linux, Windows, Mac OS X, FreeBSD.
2. Малое потребление памяти и процессорных ресурсов компьютера. Система должна

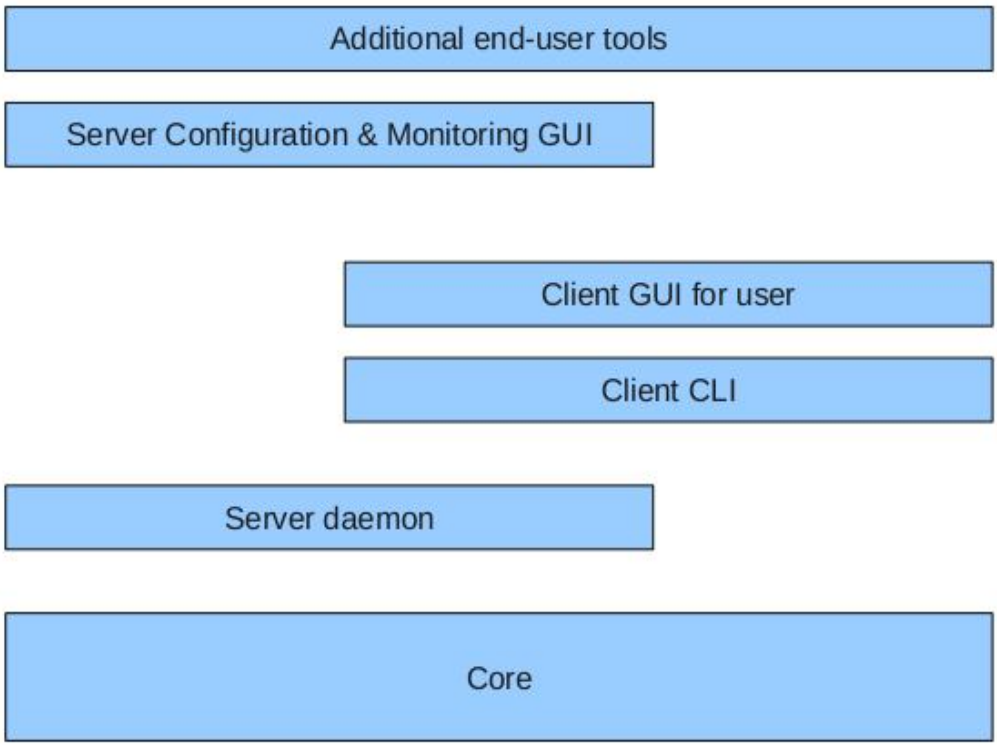
работать на компьютерах со слабой конфигурацией.

3. Высокая производительность и устойчивость к большой нагрузке. В свою очередь, разрабатываемая система должна выдерживать большую нагрузку и уметь быстро обрабатывать получаемые запросы, то есть, обладать высокой производительностью.
4. Распознавание ошибок во входящих данных. Любая опечатка в файлах конфигурации, используемых программой, должна детектироваться программой и определяться ее точное местоположение и причина. Как простые синтаксические ошибки, так и более сложные в нахождение семантические ошибки, должны быть точно определены с указанием администратору системы конкретного места в объектной иерархии, где произошла ошибка.
5. Простота развертывания. Одно из самых важных требований, из-за которого, по сути, мы и отказались от использования существующих программных продуктов. Наше приложение должно быстро и просто разворачиваться на всех операционных системах, которые были перечислены в первом требовании.
6. Легкость в первичной и всех последующих настройках. Как развитие предыдущего требования, все процессы конфигурации системы должны быть легкими в осуществлении без какой-либо предварительной подготовки. В нашем случае, это будет обеспечиваться удобными графическими утилитами, входящими в стандартный пакет программ.

Исходя из поставленных требований, были предложены следующие технологии для разработки:

- язык программирования — C стандарта ANSI 1989 года. Использование несколько устаревшего стандарта используется с той целью, чтобы система могла «нативно» компилироваться на всех перечисленных операционных системах и аппаратных платформах, а компилятор C от Microsoft для ОС Windows не поддерживает стандарты позднее стандарты 1989 года;
- в качестве основной библиотеки используется Glib, которая позволяет использовать готовые структуры данных, такие как списки, массивы, массивы указателей, связные списки, двусвязные списки и т. п.;
- для чтения и записи файлов конфигурации со сложной иерархической структурой, коими являются файлы с информацией о тестировании, используется библиотека libconfig;
- для вспомогательных графических утилит для конечного пользователя используются такие тулкиты, как GTK+ и Qt;
- libssl планируется использовать для шифрования трафика, проходящего по сетевым интерфейсам.

Следуя поставленным требованиям и предложенным технологиям, была разработана следующая архитектура:



Ядро реализует функционал, который будет использоваться повсеместно по всей программной системе (загрузка конфигурационных файлов, валидация сложной иерархической система объектов), и будет представлять собой статически подключаемую библиотеку.

На базе ядра реализуются так называемые «standalone binary applications» - сервер и клиент. Дополнительные графические утилиты являются своего рода надстройкой над сервером и клиентом, которые дают конечному пользователю удобный frontend для быстрой конфигурации и мониторинга состояния сервера.

Развертывание системы будет предельно простым. Для ОС Windows необходимо будет просто скопировать исполняемые файлы программы и запустить их, либо запустить специальные инсталлятор, который установит приложение по аналогии со всеми остальными. Для UNIX-подобных ОС пользователь может собрать приложение из исходников и установить с помощью стандартной цепочки «configure && make && make install», либо скачать приложение из репозиториев наиболее популярных дистрибутивов Linux.

В качестве примера использования можно рассмотреть ситуацию, когда преподаватель запускает серверную версию приложения у себя на ноутбуке с ОС Windows, а студенты запускают клиентскую версию программы либо у себя на ноутбуках, либо на стационарных компьютерах, которые расположены в специальных компьютерных аудиториях.

Вывод

Была разработана система, позволяющая самым удобным, на наш взгляд, способом проводить компьютерное тестирование. Она получилась кроссплатформенной, высокопроизводительной, легкой в разворачивании и настройке. Хотя система требует некоторых доработок, которые облегчат ее использование неподготовленными пользователями, она уже готова к реальному использованию в учебных заведениях.

Литература

1. Робачевский А. Операционная система UNIX. – 2-е изд., перераб. и доп. – СПб.: БХВ-Петербург, 2007. – 656 с.
2. Реймонд Э. Искусство программирования для Unix.: Пер. с. англ. - М.:Издательский дом “Вильямс”, 2005. - 544 с.