3) Защита данных	
– Общие рекомендации	– соблюдение мер противостояния SQL-инъекциям, указанных выше; – создание резервных копий и хранение их на отдельных накопителях; – обеспечение защиты авторского права доступных для загрузки файлов (например, защиту загружаемого с сервера файла паролем на открытие / редактирование/распечатывание, подписывание файла цифровой подписью, внедрение цифровых водяных знаков и пр.); – использование файла robots.txt для ограничения роботам доступа к содержимому веб-сайта.

Разумеется, приведенный в таблице список рекомендаций по мерам безопасности далеко не полон и не отражает частных деталей, однако является полезным ориентиром при внедрении и сопровождении информационных систем, использующих LAMP. В заключение отметим, что проектирование системы безопасности требует комплексного подхода, а обеспечение безопасности любого ресурса – не разовое мероприятие, а непрерывный многогранный процесс.

Список цитированных источников

1. Яремчук, С. Возьми индейца под защиту / С. Яремчук // Хакер, 2007. – №10(106). – С. 154.

2. Бойцев, О. Взлом и защита веб-сервера – на каждый яд есть свое противоядие / О. Бойцев // Компьютерная газета. – 2010. – №5.

3. Матвеев, А. На лезвии ножа / А. Матвеев // Хакер, 2005. – №3(75). – С. 52.

4. Зобнин, Е. Остаться на плаву. Обвески для Web-сервера, без которых не обойтись / Е. Зобнин // Хакер, 2010. – №10(133). – С. 128.

УДК 004.514.62

ПРОГРАММНАЯ СИСТЕМА ТЕСТИРОВАНИЯ ЭФФЕКТИВНОСТИ ОКОННЫХ ИНТЕРФЕЙСОВ

Шитиков А.В.

УО «Брестский государственный технический университет», г. Брест

Научный руководитель – Костюк Д.А., к.т.н., доцент

До самого недавнего времени оконные интерфейсы всех популярных операционных систем в той или иной мере были построены на основе метафоры «рабочего стола», получившей первое коммерческое воплощение во времена компьютеров Apple Macintosh. Согласно этой устоявшейся парадигме, существует основное (т. н. «корневое») окно графической среды, которое может полностью или частично перекрываться остальными окнами. Корневое окно отображает фоновое изображение, ряд основных элементов управления графической оболочки (обычно – средства переключения фокуса окон и запуска приложений), и, как правило, один из каталогов файловой системы, содержимое которого размещается в произвольном порядке поверх фонового изображения, по аналогии с предметами, лежащими на письменном столе.

Метафора рабочего стола на сегодняшний день воплощает многолетний опыт разработки графических интерфейсов, ориентированных на управление с помощью мыши. Однако интерфейсы на основе емкостных сенсорных экранов, управляемых пальцами,

получившие существенное распространение в последние годы в портативных устройствах (сначала в смартфонах, а затем в планшетных компьютерах ощутимо большего размера), сформировали новый набор требований к графическим оболочкам. Обострилось противоречие между размерами элементов интерфейса, которые должны быть увеличены для точной активации прикосновением пальца, и размером экрана, более ограниченным в сравнении с настольным компьютером. Понадобилось разработать и альтернативные подходы к управлению окнами. Вместе с тем многие планшетные компьютеры построены на той же аппаратной архитектуре, что и настольные ЭВМ, и могут успешно исполнять стандартное программное обеспечение при условии адаптации его графического интерфейса под сенсорный экран. В связи с этим в настоящее время разработчики графических оболочек предпринимают шаги, направленные на отказ от классических элементов интерфейса и от метафоры рабочего стола.

Много ресурсов уделяется поискам универсальных решений, пригодных для управления как с помощью мыши, так и средствами сенсорного экрана. Теоретически такой подход может повысить эффективность работы пользователя: интерфейсы приложений, упростившиеся в ходе адаптации к запуску на планшетном компьютере, легче в освоении и способствуют меньшей утомляемости, т. к. не перегружают пользователя большим числом мелких деталей. Однако отказ от классического подхода в управлении окнами, практикуемый разработчиками последних экспериментальных версий графических оболочек, является не таким однозначным: достигаемая универсальность может иметь в качестве побочного эффекта большую трудоемкость и меньшую наглядность взаимодействия.

Настоящая работа посвящена одному из способов прояснения этого вопроса: разработке средств оценки эффективности взаимодействия пользователей с оконным интерфейсом. В ходе работы решалась задача создания программной системы, функционирующей в связке со стандартным периферийным и измерительным оборудованием, и позволяющей протестировать эффективность выполнения пользователем некоторого числа однотипных операций в многооконной среде.

Структура разработанной системы представлена на рис. 1. Программа тестирования взаимодействует с графической оболочкой, создавая несколько окон, с которыми должен работать пользователь. Пользователю выделяется фиксированный промежуток времени, в течение которого необходимо выполнить в графическом интерфейсе с помощью заданных средств управления (мыши, трекбола либо тачпада) максимальное количество однотипных манипуляций. В качестве такого типового действия выбрано копирование чисел через буфер обмена. Система фиксирует в файл журнала все сделанные пользователем ошибки, а также регистрирует его состояние в ходе тестирования с помощью отдельного датчика сердечного ритма, что позволяет оценить не только скорость и точность выполнения действий, но и сопутствующую нагрузку на оператора. Информация с датчика поступает на первичный накопительный узел, сохраняющий минимальный, максимальный и средний ритм за период эксперимента для последующего анализа результатов тестирования.

Тестирование проводится в два этапа. На первом этапе пользователь имеет дело с двумя окнами: «Source» и «Destination». Окна отображаются с размером, обеспечивающим их перекрытие, что принуждает многократно использовать механизм переключения окон, предоставляемый графической оболочкой.

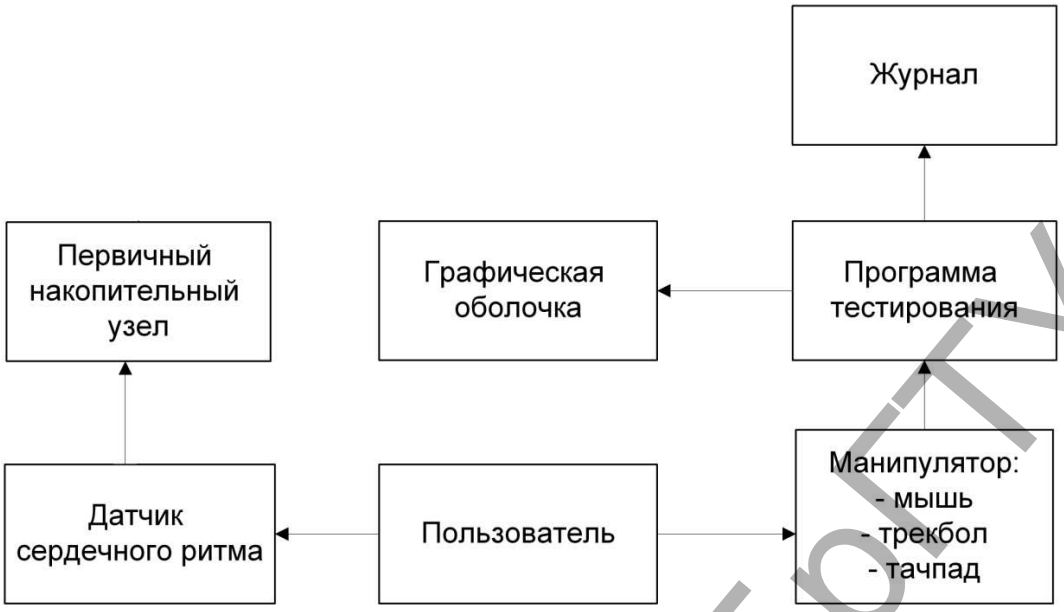

Рисунок 1 – Структура системы

Окно «Source» содержит текстовое поле и кнопку «Next», по нажатию которой в текстовом поле генерируется псевдослучайное число. Пользователь должен скопировать это число в буфер обмена путем выбора пункта «Копировать» через контекстное меню. Далее средствами графической оболочки необходимо переключить фокус на окно «Destination», куда сгенерированное число должно быть вставлено из буфера обмена с помощью пункта «Вставить» контекстного меню. При этом клавиатура не используется по условиям эксперимента. С точки зрения графической оболочки окна запускаются как независимые приложения. Этап позволяет оценить эффективность механизма переключения окон разнотипных приложений.

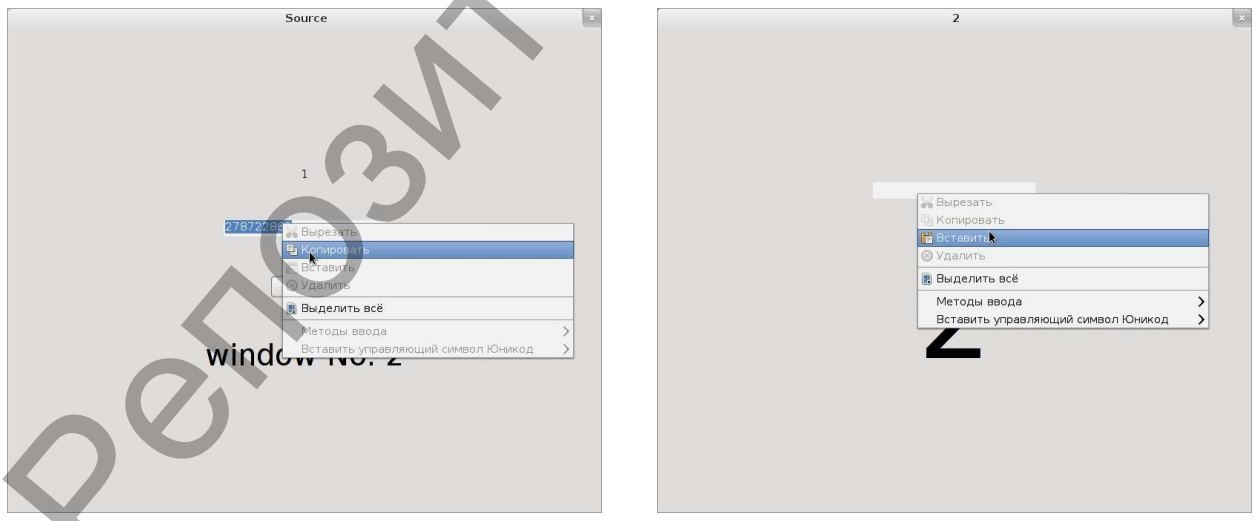

Рисунок 2 – Окна системы тестирования

На втором этапе используются *n* последовательно пронумерованных окон «Destination» и одно окно «Source», которое в дополнение к генерации псевдослучайного числа для копирования также генерирует число от 1 до *n* – номер окна «Destination», в которое следует произвести вставку. Данный этап выявляет эффективность управления окнами многооконных приложений и необходим, т.к. в последних версиях графических оболочек применяются средства группирования однотипных окон, что влияет на скорость доступа.

Для повышения концентрации внимания пользователя окна имеют схожий интерфейс, включая «фальшивую» кнопку «Next» в окне «Destination», все нажатия на которую также находят свое отражение в файле журнала. Среди типов ошибок пользователя, фиксируемых в журнале, можно выделить пропуск и дублирование чисел, ложное нажатие на кнопку и ошибочный выбор окна.

Система написана средствами языка C и библиотеки Gtk+ и предназначена в первую очередь для использования в графических оболочках Unix-подобных операционных систем. Однако исходный код может быть без существенной корректировки откомпилирован для операционных систем Windows и MacOS.

УДК 621.3.049.77

РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ АВТООПЕРАТОРА С РУЧНЫМ УПРАВЛЕНИЕМ

Шумель В.В.
УО «Белорусский национальный технический университет», г. Минск
Научный руководитель – Рудикова Л. В., к. ф.- м. н., доцент

Современный мир трудно представить без различных машин и механизмов. Они используются как в быту, повседневной жизни каждого человека, так и в составе различных производственных операций. Механизмы находят свое применение в процессах, требующих точного соблюдения временных и пространственных промежутков, небезопасных для человека или же слишком быстрых для человеческой реакции.

Ядром большинства механизмов являются микропроцессоры. Они работают по написанной специалистом программе. Сигналы с кнопок, различных датчиков (емкостных, температурных, индуктивных и пр.) являются входными данными программы процессора. Согласно входным данным и программе процессор устанавливает выходные сигналы, которые управляют работой двигателей, пневматических поршней, электромагнитных реле, клапанов и пр. Также различные интерфейсы могут связывать машину с ПЭВМ, дисплеем, принтером.

В статье изложены общие подходы к разработке программного обеспечения для автооператора с ручным управлением. Указанный автооператор собран на основе ПЛК – программируемого логического контроллера, т.е. законченного блока, включающего в себя процессор, память, коммуникационные интерфейсы и порты ввода-вывода сигналов [1].

Итак, автооператор с ручным управлением – это механизм, который служит для перемещения носителей с металлическими деталями между ваннами гальванической линии. Автооператор движется вдоль линии по направляющим металлическим путям с помощью электродвигателя. Другой двигатель поднимает и опускает каретку с носителем. Для аккуратного подъема/спуска носителя, а также точного позиционирования на позиции используются индукционные датчики. Отметим, что центром управления для автооператора является ПЛК, координирующий работу с помощью частотных преобразователей (рис. 1).