Для воссоздания принципа шифрования машины «Энигма» был использован язык программирования C# из-за его простоты и удобства в работе с объектами. Для большей наглядности процесса для входящего и выходящего текста использовались массивы символов и из метода возвращалась буква шифроалфавита, индекс которой соответствовал бы индексу входящей буквы. В предлагаемой программной версии опущены математические преобразования, проводимые с буквой при переходе с ротора на ротор, которые присутствовали на более поздних «Энигмах». Также в программе для упрощения восприятия опущено разбиение шифротекста на блоки типа «XXXXXXXXXXXX». Результат работы программы при попытке зашифровать фразу «attackatdawn» приведён на рисунке 2:

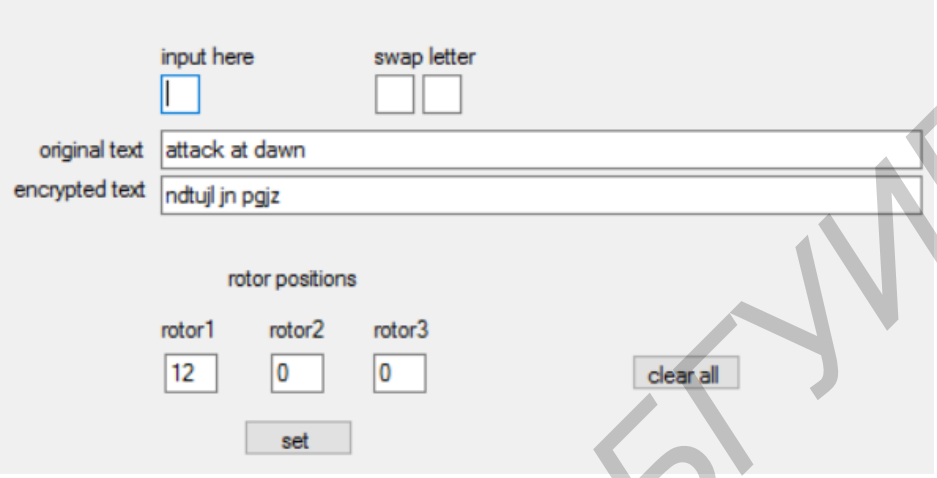

Рис. 2 – Результат работы программы

Основные преимущества данного шифра:
- простота реализации;
- большое количество комбинаций символов шифротекста;
- шифр не поддаётся сугубо математическому анализу (на данный момент).

Основной недостаток данного шифра: если шифротекст записан на грамматически строгом языке и взломщик владеет открытым текстом – расшифровать другое сообщение, зашифрованное с той же расстановкой шифрующих элементов, представляется возможным в короткие сроки.

Данная работа будет неплохим подспорьем в преподавательской деятельности, а также для наглядной демонстрации всем желающим.

В завершение хотелось бы отметить, что некоторые тексты, зашифрованные «Энигмой» и повреждённые (оставшиеся только в зашифрованном варианте) по сей день не поддаются расшифровке и восстановлению, из-за чего мы до сих пор не знаем всего о некоторых интересных событиях Второй Мировой войны.

Список использованных источников:

1. Сингх, С. Книга шифров: тайная история шифров и их расшифровки / С. Сингх ; пер. с англ. – М. : АСТ: Астрель, 2007. – 447 с.
2. Молдовян, Н. А. Криптография: от примитивов к синтезу алгоритмов / Н. А. Молдовян, А. А. Молдовян, М. А Еремеев. – СПб. : БХВ-Петербург, 2004. — 448 с.
3. Энигма. Материал из Википедии – свободной энциклопедии [Электронный ресурс]. – 2016. – Режим доступа : https://ru.wikipedia.org/wiki/%D0%AD%D0%BD%D0%B8%D0%B3%D0%BC%D0%B0.
4. The Late Tony Sale's Codes and Ciphers Website [Электронныйресурс]. – 2004. Режим доступа : http://www.codesandciphers.org.uk/enigma/rotorspec.htm.

ОРГАНИЗАЦИЯ АВТОМАТИЗИРОВАННОГО ТЕСТИРОВАНИЯ WPF ПРИЛОЖЕНИЙ

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

Костюкевич В.В.

Калугина М.А.- канд. физ.-мат. наук, доцент

Рассматривается задача организации автоматизированного тестирования WPF-приложений в соответствии с наиболее популярными решениями этой же задачи в рамках веб-приложений.

Библиотека БГУИР

В настоящее время автоматизированное тестирование становится все более популярным. Говоря о преимуществах автоматизации тестирования, чаще всего упоминают сокращение времени, затрачиваемого на проверку разработанного программного продукта. Однако, существует и много других преимуществ:

-снижение стоимости итерации тестирования,

-увеличение скорости тестирования без ущерба для результата,

-повышение надежности систем за счет улучшения качества тестирования,

-обеспечение возможности многократного использования разработанных тестовых сценариев без увеличения стоимости тестирования,

-усиление контроля процесса обеспечения качества,

-прозрачность и простота планирования времени для проведения тестирования программного обеспечения,

-автоматическое формирование отчётов о тестировании.

Эта сфера является достаточно развитой в рамках веб-приложений, но решения для WPF- приложений довольно сыры и значительно отличаются от тех, к которым привыкли разработчики веб-автоматизации.

Основным отличием между технологиями являются локаторы элементов. Веб-автоматизация имеет несколько наиболее удобных и эффективных способов нахождения элементов:

- спомощью XML Path Language,
- по свойствам CSS,
- по атрибуту id,
- по атрибуту name.

Используя XML Path Language и CSS, можно не просто указать, какой элемент необходимо найти, но и указать конкретных его родителей. Это значительно упрощает поиск. В существующих open-source решениях по автоматизации WPF-приложений поиск может производиться только по свойствам и атрибутам необходимого элемента, а значит, родительский элемент приходится находить отдельно.

Это ограничение увеличивает размеры классов страниц и делает неэффективным одно из наиболее оптимальных в автоматизации архитектурных решений − паттерн PageObject.

Все сказанное выше является причиной разработки фреймворка для работы с WPF-приложениями, которое будет иметь такой же интерфейс, как уже существующие решения для веб-приложений.

Разработанный фреймворк основывается на классе AutomationElement из .NET библиотеки. Реализация поиска элементов с использованием XML Path Language производится путем преобразования графического интерфейса тестируемого приложения в XML-документ. Основным отличием от веб-реализации является то, что поиск родительских элементов указанных пользователем в XML Path Language локаторе производится в фоновом режиме, а в веб-приложениях родительские элементы искать нет необходимости, так как DOM − это и есть XML-документ. Данная особенность говорит о том, что точность поиска не отличается от старого решения, но и необходимости в создании отдельных методов для нахождения родительских элементов нет.

Разработанное решение позволяет:

- существенно сократить время автоматизации WPF-приложений,
- упростить создание уникальных локаторов,
- облегчить переход из веб-автоматизации в WPF,
- сократить размеры создаваемых классов,
- упростить использование архитектурного шаблона Page Object,
- проводить экспертизу по возможности автоматизации конкретных приложений.

Список использованных источников:

1. ЭлфридДастин, ДжеффРэшка, ДжонПол Automated Software Testing. − ЛориРоссия, 2003. − 592 с.

ПРОГРАММНАЯ РЕАЛИЗАЦИЯ ДЕТЕКТОРОВ ДВИЖЕНИЯ С ПРИМЕНЕНИЕМ УДАЛЁННОГО МОНИТОРИНГА

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

Лагодич Ю.В.

Сиротко С. И. – канд.физ.-мат.наук, доцент

Активно развивающиеся технологиипо обработке и распознаванию изображений позволяют анализировать огромные объёмы графического материала и производить поиск похожих элементов за считанные секунды, несмотря на невозможность какого-либо нормального структурирования информации традиционными подходами. Разработка новых алгоритмов и их применение в данной сфере позволила бы перейти на качественно новый уровень решения многих задач в этой области, а также найти применение в других областях.

На текущий момент алгоритмы сравнения изображений среди больших объёмов графической информации активно используются в поисковых системах и сайтах, где нежелательно дублирование графического (а вместе с ним, зачастую, и текстового) материала в виду специфики работы (фотостоки,

Библиотека БГУИР