TALLINNA ÜLIKOOL
Informaatika Instituut

Jaan Lont

TARKVARA LOKALISEERIMISE TESTIMINE

Bakalaurusetöö

Juhendaja: lekt. Inga Petuhhov

Tallinn 2008

Содержание:

Содержание: 2
Введение: 3
1 Тестирование программного обеспечения 5
2 Уровни тестирования 7
2.1 Тестирование «белого ящика» и «черного ящика» 7
2.2 Статическое и динамическое тестирование 8
3 Проверка качества програмного обеспечения 11
4 Процесс локализации 13
4.1 Типичные ошибки 15
4.2 Автоматическое тестирование локализации 16
Заключение 18
Kokkuvõtte 19
Используемая литература : 22

Введение:

С каждым годом компьютерные технологии все глубже проникают в нашу жизнь и круг пользователей постоянно расширяется. Поэтому задача локализации программ становится в последнее время все более актуальной. Внедрение продукции на зарубежных рынках является одной из ключевых задач во все более глобализирующемся мире. Для обеспечения успеха на мировом рынке, крайне важно общаться с клиентами на их региональных языках. Перевод и локализация, таким образом, становятся важнейшими факторами для международного успеха. Локализация открывает компаниям новые возможности для развития рынков и увеличения прибыли.

Я считаю, что данная тема очень актуальна в наше время, ведь с каждым днём на IT-рынке появляются новые программы и для их локализации нужны грамотные специалисты по тестированию. Тестирование в нашей стране набирает обороты с каждым днём, появляются новые фирмы занимающиеся тестированием различных программ, систем и даже телефонов. Тестинг в Эстонии очень молод и он только начинает развиваться и набирать обороты. Европейские фирмы в последнее время заключают договора именно с эстонскими фирмами по тестированию локализованных программ. Так как многие эстонские фирмы уже успели закрепиться на рынке IT- услуг, а также зарекомендовать себя, предлогая профессиональные и качественные услуги. Локализованное тестирование – это одна из самых важных тенденций в промышленности программного обеспечения, которая делает программное обеспечение настолько легким в использовании насколько это возможно. Пригодность программного обеспечения на родном языке пользователя – важная часть этого процесса.

В данной работе я постараюсь полностью раскрыть тему локализованного тестирования, что поможет многим людям которые до этого не сталкивались с тестированием и не знали что из себя это предсталяет. А также надеюсь что данная работа сможет воспроизвести картину самого процесса тестинга и познакомить читателя с наиболее эффективными и распространёнными приёмами тестирования.В конечном результате надеюсь что у читателей сложится целостное представление о локализованном тестировании программ, а начинающие тестеры получат

необходимые теоретические сведения по данной теме. Данную работу можно считать научным пособием для начинающих тестеров. В работу я добалю ещё немного примеров различных программ, с которыми мне лично приходилось столкнуться в процессе работы тестером в одной эстонской фирме, занимающиеся тестированием программ, а также локализацией. За время, которое я проработал в этой фирме, я приобрёл много новых знаний, которые смог применить на практике, а также полностью понять суть тестирования и его процесс.

Несколько слов о структуре работы. В ведении пишется о актуальности темы, а также о целе работы и для кого она будет полезна. Основную часть я разделил на четыре части. В первой части говорится о тестировании программного обеспечения. Во второй части будут описаны уровни тестирования. В третьей части будет говориться о проверке качества программного обеспечения и в посленей четвёртой части рассмотрим сам процесс локализации. В заключении подводятся итоги о проделаной работе.

Хотелось бы также упоминуть тот факт, что количества матерьяла про тестинг очень мало, если несказать что его вовсе нет, особенно на русском и эстонском, поэтому данную работу можно считать первопроходцем в данной сфере. Работая над данной темой мне пришлось немало перелапатить матерьяла на англиском языке прежде чем я занялся написанием самой работы.

1 Тестирование программного обеспечения

Начать, я думаю, лучше с объяснения кто такой тестер и чем он знимается, а уж потом раскажу, что такое тестирование программного обеспечения и что входит в это понятие. И так, тестер это человек, в профессиональные обязанности которого входит обнаружение, локализация и отслеживание различных ошибок в программе, описание их самих, а также шагов для их воспроизведения. Практически это тот человек, который совместно с разработчиком обеспечивает качество программы. Тестер является скорее связующим звеном между разработчиком и пользователем. Это специалист, который способен рассматривать проблему с точки зрения пользователя, а обсуждать ее с разработчиком на языке программиста. Теперь я думаю привести качесва которыми должен обладать тестер: наблюдательностью, педантичностью, усидчивостью и конечно же любовью к работе, иначе Вы долго не задержитесь на этой работе.

Ну а теперь перейдём к понятию тестирования программного обеспечения. Это процесс помогающий определить корректность, полноту и качество разработанного программного обеспечения (ПО). Вместе с тем, тестирование никогда не может полностью установить корректность программы. Только процесс формальной проверки может доказать, что ошибки отсутствуют. (Хотя, так как программы для тестирования программного обеспечения пишут обычные люди, которые тоже могут ошибаться, мы не имеем права быть полностью уверенными даже в методах формальной проверки.)

Есть множество подходов к задаче локализованного тестирования ПО, но эффективное тестирование сложных программных продуктов — это процесс в высшей степени творческий, не сводящийся к следованию строгим и четким процедурам или созданию таковых. Одно из определений тестирования — «процесс опроса продукта с целью оценить его», где «вопросы» — суть действия, которые тестер пытается совершить с данным продуктом, на которые продукт отвечает своим поведением, реакцией на тестовые испытания. Хотя большинство мыслительных процессов при тестировании почти одинаковы с таковыми при обзоре и экспертизе, в данном значении термин «тестирование» употребляется в смысле динамического анализа продукта, пошаговый запуск продукта. [2]

Качество приложений обычно сильно отличается в различных системах, но есть несколько общих критериев качества программного обеспечения, таких как: надёжность, стабильность, переносимость, удобство обслуживания, простота использования (usability). Всё это должен учитовать тестер во время своей работы над программой. Более полный список атрибутов и критериев, которые должен знать тестер приведён в стандарте ISO 9126 Международной организации по стандартизации. Состав и содержание документации, сопутствующей процессу тестирования, определяется стандартом IEEE 829-1998 Standard for Software Test Documentation. [8]

В целом, тестеры различают ошибки локализации программного обеспечения и сбои. В случае сбоя программа ведет себя не так, как ожидает пользователь. Ошибка — это ошибка, которая может быть (а может и не быть) следствием сбоя, часто встречается при локализации продукта.

Общепринятая практика состоит в том, что после завершения продукта и до передачи его заказчику независимой группой тестеров проводится тестирование ПО. Эта практика часто выражается в виде отдельной фазы тестирования, которая часто используется для компенсирования задержек, возникающих на предыдущих стадиях разработки. Другая практика состоит в том, что тестирование начинается вместе с началом проекта и продолжается параллельно созданию продукта до завершения проекта. Второй путь обычно требует больших трудозатрат, но качество тестирования при этом будет выше.

2 Уровни тестирования

Ниже я приведу наиболее используемые уровни тестирования, и объясню зачем используются данные виды тестирования при локализации продукта. Во время работы мне лично приходилось сталкиваться с многими из них, так как закачики часто просили проверить продукт различными способами.

- **Модульное тестирование** (юнит-тестирование) — тестируется минимально возможный для тестирования компонент, например, отдельный класс или функция.

- **Интеграционное тестирование** — проверяет, есть ли какие-либо проблемы в интерфейсах и взаимодействии между интегрируемыми компонентами — например, не передается информация, передается некорректная информация.

- **Системное тестирование** — тестируется интегрированная система на её соответствие исходным требованиям

  - **Альфа-тестирование** — имитация реальной работы с системой штатными разработчиками, либо реальная работа с системой потенциальными пользователями/заказчиком на стороне разработчика. Часто альфа-тестирование применяется для законченного продукта в качестве внутреннего приёмочного тестирования.

  - **Бета-тестирование** — в некоторых случаях выполняется распространение версии с ограничениями (по функциональности или времени работы) для некоторой группы лиц, с тем чтобы убедиться, что продукт содержит достаточно мало ошибок. Иногда бета-тестирование выполняется для того, чтобы получить обратную связь о продукте от его будущих пользователей.

2.1 Тестирование «белого ящика» и «черного ящика»

В терминологии профессионалов тестирования (программного и некоторого аппаратного обеспечения), фразы «тестирование белого ящика» и «тестирование черного ящика» относятся к тому, имеет ли разработчик тестов доступ к исходному коду тестируемого ПО, или же тестирование выполняется через пользовательский

интерфейс либо прикладной программный интерфейс, предоставленный тестируемым модулем.

При тестировании белого ящика (англ. white-box testing, также говорят — прозрачного ящика), разработчик теста имеет доступ к исходному коду и может писать код, который связан с библиотеками тестируемого ПО. Это типично для юнит-тестирования, при котором тестируются только отдельные части системы. Оно обеспечивает то, что компоненты конструкции — работоспособны и устойчивы, до определенной степени.

При тестировании чёрного ящика (англ. black-box testing), тестер имеет доступ к ПО только через те же интерфейсы, что и заказчик или пользователь, либо через внешние интерфейсы, позволяющие другому компьютеру либо другому процессу подключиться к системе для тестирования. Например, тестирующий модуль может виртуально нажимать клавиши или кнопки мыши в тестируемой программе с помощью механизма взаимодействия процессов, с уверенностью в том, что эти события вызывают тот же отклик, что и реальные нажатия клавиш и кнопок мыши.

2.2 Статическое и динамическое тестирование

Описанные выше техники — тестирование белого ящика и тестирование чёрного ящика — предполагают, что код исполняется, и разница состоит лишь в той информации, которой владеет тестер. В обоих случаях это динамическое тестирование.

При статическом тестировании программный код не выполняется — анализ программы происходит на основе исходного кода, который вычитывается вручную, либо анализируется специальными инструментами. В некоторых случаях, анализируется не исходный, а промежуточный код (такой как байт-код или код на MSIL).

В ходе работы тестером мне приходилось заниматься лишь динамическим тестированием, так как в основном наши заказчики просят уделить максимум внимания локализации и интернационализации. Теперь предстоит узнать разницу между этими двумя схожими словами.

Понятие интернационализация можно представить как процесс адаптации продукта к языковым и культурным особенностям регионов, отличного от того, в котором разрабатывался продукт. В тестировании используется слово «internationalization» что в сокращение означает «i18n». Интернационализацию часто называют глобализацией так как у двух этих понятий одна идея и смысл в создании и развитие содержания продукта, а также различных программ и документации, которая смоглабы позволить лёгкую локализацию для целевых рынков, где различия будут лишь в языке и регионе.(На Рисунок 2.1 видно что ошибка состоит в том что после глобализации в локализованном продукте остался английский текст) [1,7]

Рисунок 2.1 – Английский текст при локализации продукта.

Есть важное различие между интернационализацией и локализацией.

Также интернационализация можноахарактиризовать как адаптация продукта для полнаценног использования во всех уголках мира, а вот в локализацию добавляют толко несколько специальных функций для использования в какихто определённых регионах.

Ниже я приведу примеры важных для интернационализации и локализации аспектов таких как:

- Язык
- Текст
- Алфавиты и шрифты; направление письма — слева направо, справа налево; системы нумерации. В большинстве современных систем проблема множества кодировок решается при помощи Юникода.
- Графическое представление текста
- Форматы даты и времени, включая различные календари
- Часовой пояс
- Валюта
- Изображения и цвета
- Названия и заголовки
- Телефонные номера, региональные и международные почтовые адреса и индексы
- Единицы мер и весов
- Форматы бумаги [1,7]

3 Проверка качества програмного обеспечения

Чтобы понять процесс локализованного тестирования, следует изначально выяснить для чего оно нужно и с чего начинается. И так, тестирование программного обеспечения — процесс, помогающий определить корректность, полноту и качество разработанного программного обеспечения.

Для лучшего понимания рассмотрим пример.

Софтверная фирма выпустила коммерческий программный продукт. Проект оказался очень успешным с точки зрения цена-качество. Компания принимает решение расширить рынок сбыта и перевести продукт на другой язык (локализовать). После принятия решения, начинается процесс локализации.

Процесс включает все стадии - маркетинг, разработка, перевод на локализованные языки и т.д. По окончании локализации, тест инженеры начинают работу тестирования.

При этом выявляются все виды ошибок - функциональные, локализационные, интернациональные, чтобы убедиться, что локализованный продукт полностью соответствует локализационным стандартам и нормам и функционирует так же как и оригинальный .

Схему, по которой работает любая фирма, занимающаяся тестированием можно примерно представить в следующем виде:

- Разрабатывается новых продукт (или новая версия), переводится на другие языки,
- Происходит тестирование продукта – находятся ошибки,
- Инженеры исправляют ошибки, отправляют непереведённые строчки переводчикам,
- Переводчики делают переводы – инженеры вставляют их в продукт,
- Происходит тестирование нового билда – находятся ошибки.

Как упоминалось выше, при тестировании обращается внимание на функциональные, локализационные , интернациональные ошибки. Чаще всего сталкиваться придётся именно с локализационными, под которыми понимаются:

- непереведённый текст,
- обрезанные диалоги,
- дублирующиеся "горячие" клавиши (если встречаются в меню),
- выбранные по умолчанию значения,
- базовая функциональность,
- отсутствующие или непереведённые меню,
- нарушено расположение элементов на странице,
- неправильное отображение локальных символов,
- сообщения об ошибках,
- проблемы с графикой,
- неверный формат даты / времени (Рисунок 3.1 – тут была допущена ошибка в формате времени). [5]

Рисунок 3.1 – Неверный формат времени.

4 Процесс локализации

Весь процесс локализации состоит из трёх этапов, первый этап это подготовка программы к самой локализации, затем идёт у нас второй этам это перевод ресурсов и конечный этап самый сложный это поиск всех ошибок и их исправление.

Прежде всего начать нужно с подготовки программы к локализации, для этого нужно отделить локализуемые ресурсы от кода затем нужно подготовить программу чтобы она корректно работала с переводом ресурсов. После того как были устранены все ошибки препятсвующие локализации продукта начинается перевод рессурсов, этим занимается переводчик и программист так как часто приходится вставлять переведёный текст прямо в код. Тут программисты обычно используют различные программы по переводу на другие языки, но мне приходилось сталкнуться именно с программой Passolo.

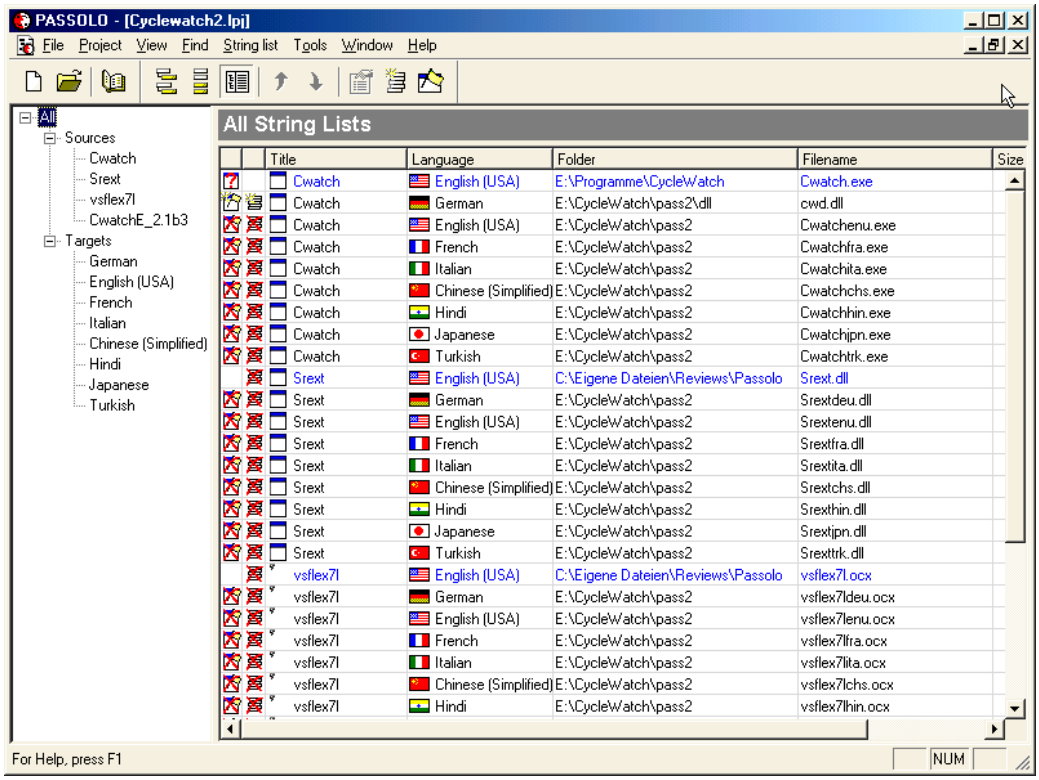

Рисунок 4.1 – Программа Passolo.

Хотелось бы рассказать пару слов о программе Passolo которую используют для перевода программ с английского на локализованные языки. Эту программу считают шаблонным редактором ресурсов и при грамотном использовании помогает

переводчику сэкономить уйму сил и времени при локализациях программ. В данной программе предусмотрена функция автоматического перевода, а также функция проверки переведённого текста на наличие большинства типичных ошибок. Функция автоматического перевода использует словари восновном от Microsoft, ну а также и других производителей. В программе Passolo (Рисунок 4.1 – изображон интерфейс программы Passolo) очень развита визуальная корректировка диалоговых окон в переводимых программах, а также эта программа умеет работать не только со стандартными файлами ресурсов (*.exe. *.dll, *.scr), но и со многими другими, в частности файлами Java (*.java, *.properties, *.class, *.jar). [6]

Процесс локализации решает следующие задачи:

- Перевод пользовательского интерфейса, включая изображения, голосовые сообщения и документацию.
- Обеспечение работы системы с принятыми в регионе единицами измерения и стандартами.
- Модификация фрагментов кода, обрабатывающих регионально-зависимые данные (даты, время, номера телефонов и т. п.).
- Обеспечение корректности лексикографической сортировки строк в соответствии с правилами целевого языка.
- Настройка печати под стандартные для рассматриваемого региона форматы бумаги.
- Проверка соответствия и допустимости в культуре целевой аудитории используемых символов, рисунков, цветовых комбинаций, музыкальных фрагментов и т. п. [3,5,7]

Вышеприведенный перечень наглядно демонстрирует многообразие работ по адаптации программного продукта на местных рынках. Это сложный процесс, требующий привлечения большого числа специалистов разного профиля: переводчиков, редакторов, инженеров, программистов, специалистов по тестированию, верстальщиков. Все это в свою очередь требует наличия у компании-поставщика серьезной производственной базы.

4.1 Типичные ошибки

А теперь расскажу о наиболее типичных ошибках, с которыми мне приходилось встречаться после перевода программ в Passolo. Существуют такие диалоговые окна, в которых содержится очень много элементов управления и все они находятся друг на друге. Но в рабочей программе отображаются не все эти объекты, а только часть из них. Эти объекты отображаются в зависимости от выполняемый в данный момент задачи и поэтому, конечный пользователь даже и не догадывается об этой мешанине объектов. Такие ресурсы в обычных редакторах довольно трудно редактировать из-за того, что видна только часть объектов. А в Passolo для таких случаев предусмотрено временное скрытие объектов. Достаточно скрыть часть объектов и все остальные появятся и тестирование не вызовет больших трудностей. Также, объекты диалогов можно объединять в группы и все действия по редактированию будут выполняться с выбранной группой.

Бывает так что переведённый объект становится настолько длинным, что не вмещается в объект (Рисунок 4.2) – как видно на этом рисунке то здесь длина текста превышает длину элемента управления. А также бывает что элемент управления перекрывает другой элемент управления. Но здесь не всегда бывает ошибка, потому что бывают диалоги со сложной структурой, когда одни элементы перекрывают другие. В этом случае приходится уже тестировать в рабочей программе, то есть новый переведённый продукт посылают к тестерам на проверку.

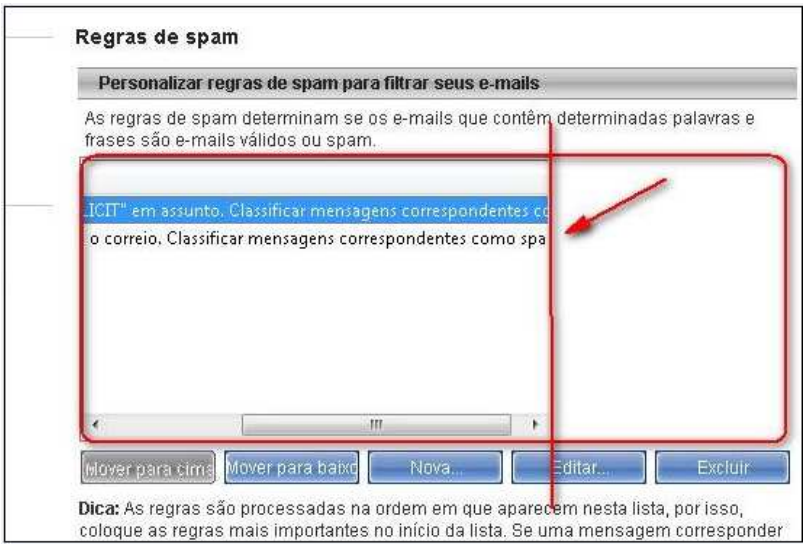

Рисунок 4.2 – Длинна текста привышает длинну элемента управления

В конечном итоге находятся все ошибки и их отсылают изготовителям продукта на исправление. Ошибки восноном бывают незначительные и обычно они косметические например длинна иконки изменилась или орфографические ошибки. Здесь обычно происходит самое длительное тестирование и исправление ошибок до тех пор пока продукт тестанет идеально чистым от ошибок, что в свою очередь займёт много времени и средств. Но так как процесс выпуска обновлений для программ незанимает обычно больше года то процесс локализации и его тестирование бесконечен так как с каждой новой версией приходится заного проходить все этапы локализации.

4.2 Автоматическое тестирование локализации

Даллее я бы хотел поговорить о автоматическом тестирование локализованных програм и сранить это тестирование с человеческим. Лично мне не приходилось с этим всталкиваться, но много слышал от знакомых, которые занимаются тестированием локализованных программ для мобильных телефонов. Этот процесс тестирования частично автоматизирован.

И так прежде всего, автоматизация не может заменить ручную проверку, когда мы говорим о локализации. Однако автоматизация может быть полезна, чтобы избежать человеческих ошибкок при попытке достичь результата. Например, переводчикам которые не очень квалифицированны в компьютерах.

Автоматизация также экономит много времени: можно запустить установку программы на тестирование локализации в ночное время, так что продукт будет готов к моменту прибытия тестеров в первой половине дня. Но так как автоматическим программам нельзя доверять тестирование локализации, тестерам или переводчикам нужно вручную проверять заключительную фазу локализации программы. Но при этом автоматическое тестирование локализации может быть очень сложным процессом, в котором большинство программ для автоматизации включают в себя Сценарий Recorder / Playback utility, где тестирование локализации может стать бесполезным по причине огромного количества ошибок.

Рисунок 4.3 – Увеличение количества символов при переводе.

В конечном итоге, ошибки всё равно придёться исправлять человеку вручную, так как в мире огромное количество различных языков и диалектов. Это в свою очередь делает тестирование локализованных программ очень трудной и трудоёмкой работой (например при переводе с английского на русский количество букв в слове может увеличется, при этом длинна диалога или кнопки неменяется, получается ошибка – что и случилось на примере рисунка - Рисунок 4.3). В заключении про автоматизацию тестирования программного обеспечения хотелось бы сказать, что она никогда незаменит ручную проверку, хотя этот способ тестирования локализованного продукта имеет большие перспективы в будущем. [4]

Заключение

В данной работе автор постарался раскрыть полностью тему тестирования локализованного программного обеспечения, и также углубиться во все детали и этапы этого тестирования. Работа так же отражает те знания, которые были приобретены во время работы тестером в одной фирме, специализирующейся на тестировании локализации программ, где он и по сей день работает.

Данная тема показалась автору очень интересной и актуальной, ведь количество программ, выпускаемых в мире растёт с каждым днём. Поэтому задача локализации программ становится в последнее время всё более актуальной – внедрение продукции на зарубежный рынок является одной из ключевых задачь бизнеса. Поэтому локализованное тестирование программ это одна из самых важных тенденцый в промышленности программного обеспечения. Локализация делает программы пригодными и понятными на родном языке пользователя. В локализацию программ входит перевод пользовательского интерфейса (включая изображение), голосовые сообщения и документацию, а самое главное обеспечение работы системы с принятыми в регионе единицами измерения и стандартами.

Тестирование локализованной программы очень трудоёмкий процесс и может занимать от нескольких месяцов до года. Часто встречаемые ошибки при тестирование локализованных программ это обрезанные диалоги, непереведённый текст и неправильное отображение локальных символов.

Целью этой работы было познакомить читателя с наиболее эффективными и распространёнными приёмами тестирования локализованных программ. Автор надеятся на то, что после прочтения этой работы у читателей сложится целостное представление о локализованном тестировании программ, а начинающие тестеры получат необходимые теоретические сведения по данной теме.

Kokkuvõtte

Iga aastaga hõivavad erinevad arvutitehnoloogiad üha enam meie igapäevaelu ning seoses sellega suureneb ka arvuti kasutajate hulk. Seepärast on programmide lokaliseerimise ülesanne muutunud viimasel ajal üha aktuaalsemaks teemaks.

Minu meelest on meie ajal antud teema väga tähtis, kuna iga päev jõuab IT-turule uusi programme ning nende lokaliseerimiseks vajab elu teadjaid testeerimis-spetsialiste. Iga päevaga laieneb testeerimist vajavate piirkondade hulk, avatakse uusi firmasid, mis tegelevad erinevate programmide, süsteemide ja isegi telefonide testeerimisega. Testimine on küllaltki uus nähtus Eestis, kuid see areneb pidevalt ja toimib üha suuremal teadusmaastikul.

Kõigepealt tuleb jutustada, mida tähendab testeerimine, ja alles siis, mida tähendab «localization testing» ning selle jaoks on vaja põhjalikult vaadelda kõiki samme, kuna lokaliseeritud testeerimist hakatakse teostama testimise viimases järgus.

Tarkvara testimine – protsess, mis aitab määrata väljatöötatud tarkvara korrektsust, täiuslikkust ja kvaliteeti. Samas, kunagi ei suuda testimine täielikult kindlaks teha programmi korrektsust. Ainult formaalse kontrolli protsess võib tõestada, et vigu ei eksisteeri. (Olgugi, et tarkvara testimise programme loovad tavalised inimesed, kes ka võivad eksida, ei saa me täiesti kindlad olla formaalse kontrolli meetodites.)

Tarkvara testimisülesande täitmiseks on hulgaliselt lähenemisviise, kuid keeruliste programmtoodete efektiivne testimine on kõrgemal tasemel loominguline protsess, mille käigus pole võimalik järgida rangeid ja ettekirjutatud protseduure või luua midagi sellesarnast. Üks võimalikest testimise mõistetest on see, et testimine so „toodangu küsitlus, mille eesmärgiks on viimase hindamine“, kus „küsimusteks“ on tegevuse põhimõte, mida testija püüab toime panna antud toodanguga, milledele toodang vastab oma käitumisega, reageerides testimise katsetele. Tavaliselt erineb lisade kvaliteet märgatavalt just mitmesuguste süsteemide poolest, kuid on olemas mõned ühised tarkvara kvaliteedi kriteeriumid, mille hulka kuuluvad: kindlus, stabiilsus, talutavus, kasutusmugavus ja –lihtsus (usability).

Atribuutide ja kriteeriumite täiuslikum nimekiri on esitatud Rahvusvahelise standartiseerimisorganisatsiooni ISO 9126 standardis. Dokumentatsiooni koostisosad ja

sisu, testimisega kaasnev protsess on standartiga IEEE 829-1998 Standard for Software Test Documentation SQA – Software Quality Assurance kindlaks määratud.

Antud protsessi paremaks mõistmiseks vaatleme näidet.

Firma Software töötas välja kommerts programmiprodukti. Hinna-kvaliteedi suhte seisukohalt osutus projekt väga edukaks. Firma võtab vastu otsuse: laiendada turgu ja tõlkida produkt teise keelde (lokaliseerida).

Pärast otsuse vastuvõtmist algab lokaliseerimise protsess. Protsess hõlmab kõiki staadiume – marketing, väljatöötamine, tõlge lokaliseeritud keeltesse jne. Lokaliseerimise lõpus alustavad testimistööd testijad-insenerid. Selgitatakse välja kõiki liiki vead – funktsionaalsed, lokalisatsiooni (L10n), internatsionaalsed (I18n), veendumaks, et lokaliseeritud produkt vastab igas mõttes lokaliseerimise standarditele ja normidele ning toimib samamoodi kui originaal (US).

Skeemi, mille alusel töötab ükskõik milline testimisega tegelev firma, võib esitada järgnevalt

a. luuakse uus produkt (või uus versioon), tõlgitakse teistesse keeltesse,
b. toimub produkti testimine – leitakse vead,
c. insenerid parandavad vead, saadavad tõlkimata jäänud read tõlkidele,
d. tõlgid tõlgivad – insenerid sisestavad need produkti,
e. toimub uue bildi testimine – leitakse vead.

Nagu ülalpool mainitud, pööratakse testimisel tähelepanu funktsionaalsetele, lokalisatsiooni (L10n), internatsionaalsetele (I18n) vigadele. Küllaltki tihti puututakse kokku lokalisatsiooni vigadega, mille all mõistetakse:

- tõlkimata tekst,
- hakitud dialoogid,
- korduvad kiirvaliku klahvid (kui leiduvad menüüs),
- valitud vaikeväärtused,
- baasfunktsionaalsus,
- puuduolev või tõlkimata menüü,
- elementide asukoht leheküljel on rikutud,
- lokaalsete sümbolite ebaõige peegeldus,

- teated vigade olemasolust,
- probleemid graafikaga,
- kuupäeva/ kellaaja ebaõige suurusnäit.

Tarkvara lokaliseerimisprotsessi võib jagada mitmeks peamiseks etapiks: programmi ettevalmistus lokalisatsiooniks, ressursside üleviimine, otsing ja vigade parandus. Vaatleme neid põhjalikumalt.

Esimesel etapil on hädavajalik ette valmistada programm lokaliseerimiseks: eraldada lokaliseeritavad ressursid koodidest, kindlustada nende töö korrektsus pärast muudatuste sisseviimist/ ressursside tõlkimist, võimaldada tööd teiste keeltega. Programmi valmisolekut lokaliseerimiseks nimetatakse lokalisatsiooniks.

Parandades kõik lokaliseerimist segavad vead, võib alustada lokaliseerimise järgmist etappi – ressursside üleviimist. Ideaalvariandis peab selle ülesandega tegelema tõlk, kuid juhul, kui ressursid pole eraldatud koodidest, siis üsna tihti tuleb sellesse töösse kaasata programmeerija. Eriti raske on siis, kui tõlge tuleb sisestada otse koodi. Kuidas antud juhul reageerib programmeerija soovile tõlkida programm mitmesse keelde, võib ainult aimata.

Ja viimasel etapil, pärast seda, kui kõik ressursid on tõlgitud, tuleb üles leida kõik vead, mis tekkisid tõlkimise käigus. Vead võivad olla puht kosmeetilised, aga ka kriitilised, formaatrea ebaloomulikkuse tõttu. Sellel etapil võib ette tulla pikaajaline testimine iga toetatava keele tarvis.

Lõppsõna: Kaasaegsete lokaliseerimisvahendite ilmumisega on mitmekeelsete lisade väljatöötamine ja toetamine muutunud tunduvalt lihtsamaks. Vaatamata sellele, et mitmekeelsete lisade loomisele esitatavaid nõudmisi tuleb arvesse võtta väljatöötamise igal etapil, tõlke teostamisel võib programmeerijate osalus viia miinimumini.

Tänapäeval on tehtud lokaliseerimine nii universaalseks kui võimalik on ning lokaliseeritud tooteid soosib üha enam inimesi, sellepärast ongi tarkvara lokaliseerimine väga tähtsal kohal meie maailmas.

Используемая литература :

1) Internationalization and localization, 30 April 2008 – [4 Мая 2008] http://en.wikipedia.org/wiki/Software_localization

2) Что такое локализация и какой она должна быть в системе, ДенисСмирнов- [4 Мая 2008] - http://freesource.info/wiki/Lokalizacija/LokalizacijaProgramm?v=90n&

3) Globalization Step-by-Step, Localization Testing – [4 Мая 2008] http://www.microsoft.com/globaldev/getWR/steps/testing/test_l10n_test.mspx

4) Automating Localization Testing, Jose M. Ladero – [4 Мая 2008] http://www.stickyminds.com/sitewide.asp?Function=edetail&ObjectType=ART&ObjectId=6281

5) Using Pseudo-Locales for Localization Testing - [4 Мая 2008] http://msdn.microsoft.com/en-us/library/aa366908(VS.85).aspx

6) Teosofia – Passolo Редактор ресурсов – [4 Мая 2008] – http://www.teosofia.ru/Soft/passolo.htm

7) Интернационализация [4 Мая 2008] – http://ru.wikipedia.org/wiki/Интернационализация

8) ISO 9126 Международний стандарт программного обеспечения – [4 Мая 2008] - http://ru.wikipedia.org/wiki/ISO_9126