Лабораторная работа №2

Низкоуровневое проектирование интерфейса:

СОЗДАНИЕ и оценка Прототипов. CWT-АНАЛИЗ ИНТЕРФЕЙСА ПРОГРАММЫ

Цели работы:

1. Закрепить теоретические знания по разработке пользовательского интерфейса.
2. Развить навыки создания версий прототипов интерфейса пользователя.
3. Провести CWT-анализ интерфейса программы, рассмотрев все репрезентативные задачи (не более 3-х)

2.1 CWT-анализ интерфейса

CWT анализ (Cognitive Walkthrough, т.е. познавательный сквозной контроль) – это формализованный способ представления мыслей и действий людей, когда они пользуются интерфейсом в первый раз. Всё начинается с того, что у вас есть прототип или детальное описание интерфейса, и вы знаете, кто будет пользователем системы. Вы выбираете одну из задач, решение которых интерфейс должен поддерживать, затем формируете полный письменный список действий, необходимых для выполнения задачи при помощи интерфейса. Далее вы пытаетесь рассказать «правдоподобную историю» о каждом действии, которое должен выполнить пользователь. Для этого необходимо давать мотивацию каждого действия пользователя исходя из его предполагаемых знаний и подсказок и реакций интерфейса. Для каждого действия могут обнаруживаться проблемы, мешающие правильному его выполнению. Эти проблемы записываются, но затем вы предполагаете, что они исправлены, и переходите к следующему действию. В результате у вас остаётся список проблем, что является руководством к действию по исправлению (улучшению) интерфейса.

CWT-анализ позволяет обнаружить несколько типов проблем с интерфейсом.

1. Поставить под сомнение ваши первоначальные и не вполне обоснованные предположения о том, как мыслит пользователь.
2. Выявлять элементы управления, которые очевидны для разработчика, но могут быть непонятны пользователю.
3. Выявлять затруднения с надписями и подсказками.
4. Обнаруживать неадекватную обратную связь, что может заставить пользователя сомневаться в результате и повторять всё с начала, хотя всё было сделано правильно.
5. Показывать недостатки в текущем описании интерфейса.

CWT-анализ фокусируется в основном на проблемах, которые пользователи испытывают при первом взаимодействии, не проходя предварительных тренировок. Такая постановка вопроса чрезвычайно важна для некоторых критических систем, таких как банкоматы или терминалы оплаты. Но та же самая ситуация возникает и в сложных программных системах, когда пользователь выполняет какую-либо задачу впервые. Пользователи часто изучают сложные программы постепенно, углубляясь в детали интерфейса по мере возникновения в том необходимости. Поэтому проблемы "первого знакомства" могут возникнуть даже при взаимодействии с давно используемой программой. Если система построена с уважением принципов CWT-анализа, то она позволяет пользователю плавно подниматься с уровня новичка до уровня эксперта.

Многие упускают необходимость сформировать полный и точный список действий для выполнения задачи. То есть они сами точно не знают, как выполнить задачу, и блуждают по интерфейсу, пытаясь отыскать правильную последовательность действий, а потом они оценивают сложность этого блуждания. Иногда, действительно, бывает полезно оценить сложность такого блуждания, но это не метод CWT. В CWT вы должны начинать, имея в руках полный список отдельных элементарных действий, необходимых для выполнения задания. Если в ходе анализа выясняется, что пользователь испытывает затруднения в определении и выполнении одного из действий, то вас интересует не то, как пользователь выйдет из положения, а сам факт того, что проблема возникла и интерфейс нуждается в доработке.

Основные рекомендации по проведению CWT-анализа сводятся к следующему. Вы определили задачу, класс пользователей, интерфейс и корректную последовательность действий. Далее рекомендуется собрать вместе группу разработчиков и других заинтересованных лиц (в учебных целях вы можете действовать в одиночку). И после этого начинается процесс анализа. Вы пытаетесь рассказать историю о том, почему пользователь выбрал бы каждое действие в списке корректных действий. Вы критикуете историю, чтобы сделать её правдоподобной. Рекомендуется держать в уме четыре вопроса:

* Будут ли пользователи пытаться произвести тот или иной эффект, который даёт действие?
* Видят ли пользователи элемент управления (кнопку, меню, переключатель и т.д.) для осуществления действия?
* Если пользователи нашли элемент управления, поймут ли они, что он производит тот эффект, который им нужен?
* После того как действие сделано, будет ли понятен пользователям тот отклик, который они получают, чтобы перейти к следующему действию с уверенностью?

По результатам CWT-анализа необходимо исправить интерфейс. Большинство исправлений будут очевидны. Сложный случай возникает тогда, когда у пользователя вообще нет никаких причин думать, что действие должно быть сделано. Самое верное решение этой проблемы – исключить это действие; пусть система сама выполнит его. Если так не получается, то необходимо перестроить задачу так, чтобы пользователи получали бы логическое побуждение к выполнению "проблемного" действия.

CWT-анализ позволяет выявить много проблем с интерфейсом, но не даёт ответа на вопрос, насколько сложен интерфейс, т.е. сколько времени тратит пользователь на выполнение задачи.

2.2 Пример CWT-анализа

Рассмотрим программу CDCopy. Пусть репрезентативная задача звучит так: преобразовать третий трек аудио компакт-диска в формат MP3 и сохранить на жёстком диске компьютера.

Последовательность действий:

1. Загрузить аудио диск в устройство чтения компакт-дисков компьютера;
2. запустить программу CDCopy;
3. в появившемся списке треков выбрать трек № 3;
4. в списке форматов файлов выбрать MP3(MPEG 1 Lay. 3);
5. нажать кнопку "Start copying".

Решение задачи требует от пользователя всего пять действий. Выполним их подробный анализ. Первые два действия вроде бы не связаны с интерфейсом программы и должны обеспечиваться интерфейсами других систем, не подвергающихся анализу. Но всегда ли для пользователя естественна такая последовательность действий? Многие пользователи предпочитают сначала запускать нужную им программу, а уже затем вставлять диск (в противном случае может сработать автоопределение и запуститься совсем другое приложение). Высказанное замечание ещё не означает, что в интерфейс программы нужно вносить изменения: может быть, программа будет корректно работать и при другой последовательности действий. Но значение метода CWT в том, что любое замечание является поводом дальнейших проверок. И действительно, при дальнейших исследованиях оказывается, что если вначале выполнить действие 2, то программа выдаёт окно с надписью "No media present" и кнопкой "OK"; при закрытии окна программа запускается с пустым списком треков. Это может сбить с толку неопытного пользователя и заставить его закрыть программу и попытаться начать всё с начала в другой последовательности. Анализ первых двух действий сразу даёт подсказку к улучшению интерфейса: если программа запускается до загрузки диска, то не нужно выдавать предупреждающее сообщение, а после запуска программы с пустым списком треков выдать пользователю явный запрос на загрузку диска. Итак, мы выявили некоторые слабости интерфейса в первых двух действиях, но теперь предположим, что они устранены, и перейдём к третьему действию. Третье действие заключается в выборе нужного трека в списке. Здесь интерфейс стандартен и однозначен. Никаких возможностей для непонимания пользователем, как правильно выполнить это действие, мы не находим. Замечаний нет. Четвёртым действием пользователь должен выбрать формат MP3 в списке форматов. Сразу возникает вопрос: как он найдёт этот список? На самом деле имеется в виду выпадающий список, в котором сейчас выводится слово WAV. Но пользователь может не знать, что такое WAV. Ведь его задача состоит в сохранении аудио трека в формате MP3. Он может вообще не знать других слов, кроме MP3. Если он знает, что WAV это один из форматов, то он может догадаться, что элемент управления с этим словом и есть список форматов. Очевидное решение проблемы снабдить список форматов меткой с текстом типа "Output format". Считаем, что эта проблема исправлена, и переходим к пятому действию. Пятое действие. Опять возникает проблема с нахождением кнопки "Start copying". Может возникнуть желание нажать на большую кнопку с похожей надписью "CD copy", но она выполняет совсем другое действие. На самом деле имеется в виду кнопка в верхнем ряду с изображением дискеты. Всплывающая надпись "Start copying" появляется при наведении на неё указателя мыши. Можно улучшить ситуацию, если при описании действий пользователя использовать картинку с изображением кнопки. Но всё равно, многие привыкли к тому, что кнопка с изображением дискеты сохраняет файл без преобразования формата. Лучше было бы изменить вид кнопки. После нажатия на кнопку начинается процесс копирования, его ход отображается в новом окне. По завершении процесса программа без всяких вопросов переходит в исходное состояние. Но у пользователя возникает вопрос: если преобразованный файл был сохранён, то где? Здесь мы сталкиваемся с типичным недостатком интерфейса отсутствием необходимой обратной связи. Пользователь не видит результата своего действия (в предыдущих действиях результат был ясно виден, т.е. обратная связь была вполне нормальной). Предпринятые изыскания показали, что файл был сохранён по месту установки программы (\Program Files\CDCopy) в папке ec Если присмотреться, то видно, что название папки взято из CD-ID, отображаемого над списком треков (см. рис. р375). Отметим, что если бы программа была запущена не администратором системы, то копирование не смогло бы выполниться, т.к. папка Program Files и все её внутренние файлы недоступны для записи другим пользователям. Программа не работает в многопользовательском режиме. Простая доработка интерфейса устраняет проблемы: запросить у пользователя путь для сохранения файла с помощью стандартного диалогового окна (как это делают многие программы); начальный путь брать из переменной среды USERPROFILE (в ней записан путь к каталогу текущего пользователя). Итак, мы провели CWT-анализ интерфейса программы CDCopy на примере решения одной из репрезентативных задач. Был выявлен ряд недостатков интерфейса и предложено 4 простых доработки, их устраняющих.

* 1. Задание к лабораторной работе

1. Имея результаты проектирования интерфейса на высоком уровне, разработать варианты моделей – прототипы экранных форм для каждого из функциональных блоков. Создать:

* по 3 различных бумажных прототипа для трех наиболее важных форм программы (рисунки должны быть нарисованы быстро и неаккуратно, но понятно и читаемо);
* по одному электронному прототипу для каждой из этих форм (т.е. всего создать 3 прототипа), в одной из сред разработки ПО, имеющей визуальные средства создания интерфейса.

1. Выполнить эвристическую оценку качества одной из проектируемых экранных форм:

* сравнить все разработанные бумажные прототипы для выбранной формы по степени соответствия свойствам эффективного пользовательского интерфейса;
* выбрать наилучший прототип, выбор аргументировать.

1. Провести CWT-анализ.

2.4 Требования к оформлению отчета

Отчет должен содержать:

* название и цели работы;
* внешний вид разработанных экранных форм с описанием навигации по ним (рукописные рисунки бумажных прототипов, их отсканированные изображения или фотографии; скриншоты электронных прототипов);
* описание навигации по приложению с использованием элементов управления на спроектированных формах;
* проведение CWT-анализа.