**Лекция 2 ПрЧМИ 2024**

Целью практических занятий по курсу «Проектирование интерфейса компьютерных приложений» является знакомство с технологией разработки и тестирования интерфейсных элементов. При этом выполняется анализ требований, предъявляемых к разрабатываемым компонентам, реализуются основные этапы проектирования интерфейса, его тестирования, создается система обучения и документации. В качестве средства разработки компонентов используются различные среды программирования.

Для указанного визуального компонента нужно сформулировать легенду, т.е. формулировку задачи.

При реализации легенды могут использоваться другие визуальные элементы. Надо реализовывать требования только для указанных визуальных элементов в задании.

Практикум включает в себя текстовый отчет и разработанную программу. Программа должна быть разработана при завершении практикума.

При разработке приложения необходимо полностью выполнить требования для визуальных элементов, указанных в задании, и учесть требования или пожелания преподавателя при создании кода для легенды.

Структура отчета по лабораторным работам следующая

Все пункты и подпункты отчета рассматриваются сначала с точки зрения легенды, а затем с точки зрения задания.

Структура отчета по лабораторному практикуму:

1. Титул
2. Оглавление
3. Содержание
4. Приложение

Оглавление содержит следующие пункты:

1. Описание предметной области

2. Постановка задачи

2.1. Формализация контекста использования

2.2. Формализация объективных критериев успеха

2.3. Определение необходимой функциональности системы

2.4. Анализ целей

2.5. Анализ действий пользователей

2.6. Определение низкоуровневых и высокоуровневых функций

2.7. Формализация бизнес-ролей пользователей

2.8. Формализация функциональности

2.9. Формализация сценариев действий пользователей

2.10. Обзор интерфейса конкурирующих систем

2.11. Формализация привычек и ожиданий пользователей.

3. Проектирование интерфейса

3.1. Проектирование структуры экранов системы

3.2. Проектирование навигационной системы

3.3. Построение прототипа ПИ

3.4. Низкоуровневое проектирование.

4. Тестирование интерфейса

5. Система помощи и документация

Описание предметной области относится к легенде, для которой рассматриваются основные понятия и ситуации в данной области.

Свойства компонент, указанные в задании, должны быть доступны пользователю либо в виде нажатия каких-то клавиш, либо в виде приведенного дополнительного или контекстного меню, либо еще каким-то образом.

Отчет по лабораторному практикуму формируется путем наращивания количества страниц в отчете путем добавления сделанных лабораторных работ. Студентам групп 3-41 и 3-42 можно вместо лабораторной работы выполнять задание по выбранной теме для дисциплин «Конструирование программного обеспечения», «Разработка мобильного приложения» или выполнять тему по ВКР.

Завершением лабораторного практикума является разработанные отчет и программа.

В методических указаниях рассматриваются лабораторные работы, содержащие методические указания и пример.

Далее согласно приведенному оглавлению будем рассматривать пункты с точки зрения теории.

**2. Постановка задачи**

***Сведения из теории***

Приводится формулировка задания и формулировка легенды. Формулировка легенды должна представлять собой задание на разработку приложения.

***Методические указания***

Приводится формулировка задания и формулировка легенды (формулировки задачи для некоторой предметной области). Легенда должна представлять собой задание на разработку приложения, в котором используется наряду с другими визуальными элементами визуальный компонент задания.

#### Практическая часть(пример)

***Задание***

Разработать визуальные компоненты, реализующие элементы управления CheckBox и RadioButton и позволяющие изменять свой внешний вид путем задания их цвета или вида элементов из файла формата BMP. Компонент должен поддерживать темы ОС Windows.

***Легенда***

Создать приложение, реализующее тестирование с помощью применения формирования элементов теста. Элемент теста представляет собой вопрос и набор вариантов ответов. Правильных ответов может быть один или несколько. Вариантов ответов может быть не больше пяти. Элемент теста может формироваться в диалоге или вводиться из текстового файла. В процессе диалога вводится вопрос и варианты ответов. Если элемент теста вводится из файла, то структура файла следующая: несколько строк может занимать вопрос, признаком окончания вопроса является символ точка с запятой; затем в каждой из пяти строк содержится вариант ответа, признаком окончания текста ответа является символ точка с запятой; после этого следуют номера правильных ответов, перечисленные через запятую и ограниченные символом точка с запятой. Если вариантов ответов меньше, чем пять, то формируются пустые строки.

**2.1. Формализация контекста использования**

***Сведения из теории***

На этом этапе собирается большинство сведений о пользователях. Описываются следующие свойства аудитории системы:

* характеристики пользователей: их опыт работы с компьютером, знание предметной области, мотивы, размер/важность групп пользователей, образцы (типовые ситуации) использования;
* цели и задачи пользователей;
* задачи проекта (причина создания проекта, его этапы, результаты, которые должны быть получены, необходимая информация);
* технология разработки и платформа, на которой будут работать пользователи;
* среда, в которой будет создаваться и использоваться проект (физическая, рыночная, организационная и культурная).

***Методические указания***

1. Описать характеристики пользователей вашего создаваемого приложения:

* опыт работы с компьютером, знание предметной области, мотивы, размер/важность групп пользователей, образцы (типовые ситуации) использования;
* цели и задачи пользователей;
* задачи проекта (причина создания проекта, его этапы, результаты, которые должны быть получены, необходимая информация);
* технология разработки и платформа, на которой будут работать пользователи;
* среда, в которой будет создаваться и использоваться проект.

***Практическая часть (Пример)***

Сведения о пользователях

Пользователей разрабатываемых компонентов можно разделить на две основные группы:

1. Конечные пользователи систем, в которых будут использоваться разрабатываемые компоненты. К этой группе относятся люди, имеющие минимальные навыки работы с компьютером (основные приемы работы с Windows, программами MS Office, почтовыми системами и т.п.), либо не имеющие этого опыта вообще.
2. Программисты, которые будут использовать компоненты CheckBox и RadioButton и элемент теста в своих разработках.

Цели и задачи пользователей

Целью первой группы пользователей является создание приложения, представляющего собой некоторый тест.

Целью второй группы пользователей является удобное и простое подключение компонентов к разрабатываемой системе и обеспечение их корректной работы в рамках проекта в целом.

Задачи проекта

Компонент элемент теста является составной частью большинства разрабатываемых тестовых систем. Значительное упрощение процесса составление тестов является задачей проекта.

Такие компоненты, как CheckBox и RadioButton, применяются практически на каждой интерфейсной форме в любой из компьютерных систем. Однако стандартные компоненты имеют единообразное практически ненастраиваемое исполнение, что снижает удобство и наглядность их использования. В связи с этим задачей проекта является добавление к стандартным компонентам функций, позволяющих изменить внешний вид этих компонентов (задание их цвета или вида элементов из файла формата BMP).

Технология разработки и платформа

Компоненты разрабатываются в среде Visual Studio с использованием языка C#. Требуемая операционная среда – Windows 7 и более поздние версии Windows.

**2.2. Формализация объективных критериев успеха**

***Сведения из теории***

Выделяются объективные критерии оценки эргономичности интерфейса (показатели эффективности, продуктивности, удовлетворенности пользователей); создается реальное задание на проектирование интерфейса, например:

1. Группа пользователей постоянно меняет свой состав, и разрабатываемая программа будет использоваться редко. Необходимо акцентировать внимание на простоте интерфейса.
2. Одна и та же задача повторяется многократно, а группа пользователей довольно большая. Необходимо акцентировать внимание на эффективности использования. На 20 % снизить количество человеческих ошибок и т.д.
3. ***Методические указания***

1. Выделить объективные критерии оценки эргономичности интерфейса (показатели эффективности, продуктивности, удовлетворенности пользователей).

2. Создать реальное задание на проектирование интерфейса.

**Практическая часть(Пример)**

Основными критериями успеха у пользователей являются простота, удобство и наглядность использования компонентов.

Интерфейс должен обладать свойством быстрого освоения работы с ним пользователей-непрофессионалов.

Для программистов разработанные компоненты интерфейса должны обладать свойством удобного подключения компонентов к проектам.

***Пояснения к сведениям из теории***

Существует четыре основных критерия эргономичности (качества) любого интерфейса, а именно:

Скорость работы пользователей

Количество человеческих ошибок

Скорость обучения

Субъективная удовлетворенность пользователей.

Эти критерии являются показателями эффективности, продуктивности и удовлетворенности пользователей.

Существует две разных производительности - производительность компьютера и производительность пользователя.

Производительность компьютера – широко известное техническое понятие и для ее увеличения существует множество методов. Увеличение производительности компьютера ускоряет все процессы, повышает эффективность их выполнения и уменьшает стоимость одной операции.

***Производительность пользователя*** находится в прямой зависимости от длительности выполнения работы пользователем. Длительность выполнения работы пользователем состоит из:

* Длительности восприятия исходной информации
* Длительности интеллектуальной работы
* Длительности физических действий пользователя
* Длительности реакции системы

Длительность реакции системы, как правило, является наименее значимым фактором.

Взаимодействие пользователя с системой (не только компьютерной) состоит из семи шагов:

* формирование цели действий;
* определение общей направленности действий;
* определение конкретных действий;
* выполнение действий;
* восприятие нового состояния системы;
* интерпретация состояния системы;
* оценка результата.

Длительность восприятия исходной информации зависит от многих индивидуальных качеств пользователей (какой вид информации воспринимается лучше, физиологические качества органов чувств и т.д.)

При взаимодействии пользователя с системой основное время занимает умственная деятельность

Повышение скорости размышлений приводит к увеличению скорости работы пользователей с системой.

Повысить скорость мышления, присущего человеку практически невозможно, зато можно уменьшить влияние факторов, усложняющих процесс мышления.

Перечислим факторы, уменьшающие время интеллектуальной работы пользователей:

- упрощение манипулирования;

- компенсация потери фокуса;

- ограничение принятия решения;

- закон Хика.

Более подробно эти факторы будут рассмотрены при изучении следующих этапов проектирования пользовательского интерфейса.

Методы, уменьшающие длительность физических действий пользователя:

* Закон Фиттса;
* Методы повышения доступности кнопки;
* Уменьшение числа манипуляций;
* Уменьшение необходимости ввода данных;
* Память программы.

Также более подробно эти методы будут рассмотрены при изучении следующих этапов проектирования пользовательского интерфейса.

Рассмотрим следующие этапы разработки постановки задачи.

**2.3. Определение необходимой функциональности системы**

***Сведения из теории***

На первом этапе необходимо определить функциональность будущей системы. Это исключительно важный этап, поскольку именно функциональность будет определять весь интерфейс. Очень важно сознавать, что практически невозможно убрать из уже продающейся системы какие-либо функции. Поэтому оптимальным вариантом работы почти всегда является проектирование функциональности сразу на несколько версий вперед.

Существуют два основных способа для определения нужной функциональности - анализ целей и анализ действий пользователей. Эти способы фактически не конфликтуют друг с другом, более того, в процессе определения функциональности желательно использовать оба.

***Методические указания***

1. Использовать анализ целей для определения функциональности.

2. Использовать анализ действий пользователей для определения функциональности.

3. Используя результаты анализа, описать функциональность.

**Практическая часть (Пример)**

Компонент элемент теста формируется в диалоговом режиме или путем ввода информации из файла. Для установления режима ввода предназначен компонент CheckBox. Если галочка не установлена, то происходит ввод в диалоге с помощью текстовых полей, если галочка установлена, то происходит ввод информации из файла.

Компонент CheckBox используется для того, чтобы пользователь с помощью мыши или клавиатуры мог указать свое решение типа да/нет или да/нет/не уверен (в последнем случае флажок устанавливается, но выглядит блеклым). Это решение отражается в специальном свойстве компонента, доступном как для чтения, так и для записи. В составе диалогового окна может быть несколько компонентов CheckBox, причём состояние любого из них никак не зависит от состояния остальных. Также в компоненте должна существовать возможность определения положения текста надписи (левое или правое выравнивание). В соответствии с заданием позволяется изменять цвет и внешний вид элементов.

В отличие от компонентов CheckBox, компоненты RadioButton представляют собой переключатели, предназначенные для выбора одного из нескольких взаимоисключающих решений. Выбор осуществляется также с помощью мыши или клавиатуры. На форму (точнее, в компонент-контейнер) помещается, по меньшей мере, два таких компонента. Они могут иметь только два состояния, определяемых соответствующим свойством. Если в одном компоненте это свойство принимает значение True, во всех других компонентах, расположенных в том же контейнере, эти свойства принимают значения False. Аналогично компоненту CheckBox задаётся свойство выравнивания и свойства, позволяющие изменять цвет и внешний вид элементов.

**2.4. Анализ целей**

***Сведения из теории***

Разработчику необходимо четко осознавать, что пользователям не нужны инструменты сами по себе, нужны лишь результаты их работы. Никому не нужен текстовый процессор – нужна возможность с удобством писать тексты. Никому не нужна программа обработки изображений – нужны уже обработанные изображения. Это значит, что сами по себе функции никому не нужны и не важны. Людям нужно средство вообще, с помощью которого можно выполнять какую-либо работу.

Разницу подходов к выбору функциональности удобно проиллюстрировать на примере тостера. Стандартный подход, при котором функции выбираются фактически произвольно, в лучшем случае приведет к такому заданию:

*«Нужен ящик с узкой прямоугольной дыркой и нагревателем внутри».*

Анализ целей пользователя приведет к другой формулировке:

*«Нужен поджаренный хлеб. Похоже, что проще всего добиться этого созданием ящика с дыркой по форме куска хлеба и нагревателем внутри. С другой стороны, похоже, что этот способ не единственный».*

Второй вариант при полном развитии этого метода может привести не только к созданию тостера, но и ростера (т.е. устройства, в котором можно поджаривать не только хлеб).

Главное же другое. Ни в коем случае нельзя дать обмануть себя ненужной конкретикой, т.е. описанием того, каковадолжна быть будущая функциональность. Как правило, одного и того же результата можно добиться несколькими разными способами, при этом важно не только реализовать какой-либо способ, но и выбрать лучший. Анализ целей пользователя как раз и позволяет избежать ненужной конкретики.

Результатом этого процесса должен являться список целей. Например, для тостера финальный список целей должен выглядеть очень просто: *«Должен поджаривать мелкие кусочки пищи, преимущественно хлеб»*.

После того как истинные цели пользователей установлены (и доказано, что таких пользователей достаточно много, чтобы оправдать создание системы), приходит время выбирать конкретный способ реализации функции, для чего используется второй метод.

***Методические указания***

1. Используя результаты анализа целей пользователя проанализировать формулировку легенды и задания и выявить цели разработки.

2. Описать цели разработки, учитывая интересы пользователей и программистов

**Практическая часть (Пример)**

Как выше было сказано, целью первой группы пользователей является создание приложения, представляющего собой некоторый тест. Было упомянуто, что с помощью элемента CheckBox, условное имя которому «Указатель источника», устанавливается, каким образом формируется элемент теста (формирование с помощью диалога, либо формирование из файла). С помощью другого элемента CheckBox, условное имя которого «Режим работы», устанавливается, как работает приложение (создание теста или функционирование теста). Для защиты от исправления теста во время функционирования необходимо предусмотреть ввод пароля, который защищает тест от вмешательства в процесс его разработки, т.е. если введенный пароль верен, то приложение работает в режиме формирования теста, если не верен или не введен, то приложение работает в режиме тестирования.

Таким образом, цель первой группы пользователей содержит две подцели: указание режима работы и источника формирования элемента теста.

Целью второй группы пользователей является удобное и простое подключение компонентов к разрабатываемой системе и обеспечение их корректной работы в рамках проекта в целом.

Эта цель может быть реализована с помощью решения следующих задач:

- создание класса CheckBox1, включающего класс CheckBox;

- создание класса RadioButton1, включающего класс RadioButton;

- создание класса CreateTest, реализующего создание элемента теста.

**2.5. Анализ действий пользователей**

***Сведения из теории***

Достижение почти всех целей требует от пользователей совершения определенных действий. Разумеется, эти действия могут различаться при разных способах достижения.

В сложных интерактивных системах сами по себе выбранные стратегии действий влияют на требования к функциональности. Возвращаясь к примеру с тостером, можно указать, что помимо возможности поджаривать хлеб он должен включаться и выключаться, более того, он должен быть устроен таким образом, чтобы его можно было удобно мыть. С другой стороны, возможно, что тостер можно сделать так, чтобы он не требовал мойки, в этом случае функция «возможность мытья» становится лишней. Разумеется, взаимодействие человека с тостером очень просто, здесь можно дойти до всего чисто логическим анализом. В компьютерных же системах взаимодействие обычно сложнее, при этом логический анализ неприемлем. Единственным выходом является наблюдение за людьми, выполняющими свою задачу, пользуясь уже имеющимися инструментами, а именно системами конкурентов (если они есть) и предметами реального мира. Неплохим источником материала для анализа часто служит даже не наблюдение за людьми, а анализ результатов их работы – если оказывается, что результат работы практически не зависит от используемого инструмента, это значит, что нужна только та функциональность, которая оказала воздействие на результат (т.е. функции, которыми никто не воспользовался, не нужны).

Как уже было сказано, обычно есть несколько разных способов реализации одной и той же функции. Анализ действий пользователей как раз и позволяет определить, какой именно способ следует реализовывать.

Поскольку на этом этапе мы узнаём, какая именно функциональность нужна для каждого варианта, можно избрать верный путь по правилу: чем меньше действий требуется от пользователя, тем лучше. Не стоит забывать и про другое правило: чем меньше функций, тем легче их сделать.

***Методические указания***

1. Для типовых сценариев проанализировать действия пользователей при работе с визуальными элементами.

2. Описать типичные действия пользователей для типовых сценариев.

**Практическая часть (Пример)**

Первая группа пользователей для выбора альтернатив может использовать как клавиатуру, так и мышь. При этом у клавиатуры скорее вспомогательная роль, так как обычно при использовании подобных компонентов для пользователя более обычной и удобной является мышь.

Клавиатура в этих случаях используется достаточно редко, но её использование также должно быть предусмотрено для некоторых крайних случаев. Так, применение клавиши табуляции используется для перемещения между компонентами, а функцию, аналогичную функции левой кнопки мыши, здесь играет пробел.

Вторая группа пользователей использует созданные классы в привычной среде разработки приложения.

**Дополнительные сведения по теории**

Как было уже сказано, для накопления материала для анализа взаимодействия пользователя с интерфейсом логический анализ неприемлем. Причем логический анализ также можно проводить разными способами, с одной стороны быстрыми и не быстрыми, с другой стороны проводя небольшое количество испытаний или большое количество испытаний и т.д. Проведение большого количества испытаний является очень дорогостоящим мероприятием и займет много времени, потому что не один раз надо закодировать пользовательский интерфейс и проверять его на пользователях.

Существуют следующие относительно быстрые и относительно дешевые способы накопления материала для проверки и анализа взаимодействия пользователя с интерфейсом:

- юзабилити-тестирование;

- метод фокусных групп;

- проверка посредством наблюдения за пользователем;

- метод «мыслим вслух»;

- проверка качества восприятия;

- измерение производительности.

Кроме указанных способов тестирования существуют еще следующие:

- контрольные списки;

- измерение качества пользовательского интерфейса с помощью закона Фиттса;

- измерение качества пользовательского интерфейса с помощью закона Хика.

Применение первой группы методов тестирования возможно на любых этапах разработки. Это означает, что программного кода нет, а правильность проектирования пользовательского интерфейса можно проверить.

Вторую группу методов проверки возможно применять в том случае, когда программно пользовательский интерфейс частично или полностью разработан.

Возникает вопрос, а как проверить пользовательский интерфейс при отсутствии кода.

Для этого создается первая версия пользовательского интерфейса – бумажная. Сначала можно просто нарисовать, ручкой или в каком-то графическом интерфейсе. Затем можно на отдельных карточках написать или напечатать названия визуальных элементов. Собрать группу тестировщиков-пользователей. На западе им за это платят. Перед каждым пользователем сформулировать отдельную задачу. Карточки с названиями визуальных элементов разложить перед каждым пользователем в том порядке, который представляется разработчику и затем каким-то образом фиксировать последовательность выбора визуальных элементов каждым пользователем при решении предложенной задачи.

Организационно каждый из указанных методов тестирования выполняется по-разному.

Краткая характеристика методов тестирования.

**Юзабилити-тестирование представляет собой** постановку экспериментов с целью выявления специфичной информации, касающейся дизайна и пользовательских свойств исследуемого продукта. Юзабилити-тестирование производится на протяжении всего цикла разработки продукта. На ранних стадиях разработки тестирование предыдущей версии или конкурирующих продуктов позволяет команде проектировщиков наметить контрольные точки, которых необходимо достигнуть в процессе разработки. В середине работы над продуктом, тестирование проверяет корректность произведённого дизайна и предоставляет обратную связь, сообщая места, где дизайн нуждается в улучшении. На заключительных этапах тестирование удостоверяет, что продукт соответствует тем целям, для которых был спроектирован.