**Лекция 3 ПрЧМИ 2024**

**2.6. Определение низкоуровневых и высокоуровневых функций**

***Сведения из теории***

Существует два принципиально разных подхода к определению функциональности системы. При первом подходе система снабжается максимальным количеством функций, при этом результаты многих из них являются суммой результатов других функций. При втором подходе разработчик снабжает систему набором базовых условно-элементарных функций, из которых пользователь может «собрать» более сложные необходимые ему функции.

Оба подхода имеют как недостатки, так и достоинства. Подход, при котором количество функций ограничено, позволяет упрощать интерфейс, но при этом от пользователя требуется понимание, как из многих низкоуровневых функций «собирать» функции более сложные. Подход, при котором помимо низкоуровневых функций есть высокоуровневые, позволяет потенциально обеспечивать большую скорость работы (за счет отсутствия пауз между низкоуровневыми функциями), но зато от пользователя требуются знания о том, где эти высокоуровневые функции найти и как с ними работать, при этом они перегружают интерфейс. Кроме того, остается возможность компромисса: всегда можно включить в систему средства автоматизации, чтобы пользователи получали возможность создавать (и распространять) свои метафункции. Этот подход является оптимальным.

**Практическая часть (Пример)**

К высокоуровневым функциям легенды относятся следующие:

- указание источника формирования элемента теста;

- указание режима работы;

- ввод пароля;

- создание теста;

- прохождение теста;

- выдача справки о приложении.

Низкоуровневыми функциями легенды являются выдача контекстной помощи о визуальных элементах легенды и выдача сообщений об ошибках.

К низкоуровневым функциям задания относятся:

- изменение цвета элементов (текста и фона);

- изменение вида элементов (шрифта и вида).

**2.7.** **Формализация бизнес-ролей пользователей**

***Сведения из теории***

Функциональность любой системы разделяется на несколько ролей пользователей: разным пользователям нужны разные блоки функциональности (в системах автоматизации эти роли называются бизнес-ролями). Навигация по системе прямо зависит от этих ролей, поскольку в пределах одной роли в навигацию нежелательно включать функции из чужой роли. Соответственно на этом этапе выделяются основные роли пользователей с относящимися к этим ролям функциями. Также на этом этапе проводятся собеседования с каждым из представителей определенной роли на предмет выявления особенностей данной роли и выяснения, какие дополнительные (по отношению к формальным) возможности следует предусмотреть.

**Практическая часть (Пример)**

В данном приложении предусмотрены три роли:

- пользователь, который проходит тест (первая группа);

- пользователь, который разрабатывает тест (вторая группа);

- пользователь-программист, который использует разработанные классы компонент в своей разработке (третья группа).

Для пользователей первой группы разрабатываются различные виды тестов, которые они проходят. Пользователи второй группы создают тесты для различных направлений.

**2.8.** **Формализация функциональности**

***Сведения из теории***

На этом этапе, основываясь на информации, выработанной на предыдущих этапах, окончательно формируется список функциональных возможностей новой версии системы. Ранее сформированное ТЗ порой не включает части необходимой функциональности либо содержит функциональность, реально не требующуюся пользователям.

**Практическая часть (Пример)**

Для пользователей второй группы при создании теста нужно указать источник. Если нужно указать файл, то его либо нужно выбрать в директории, либо задать путь.

Для пользователей первой группы для прохождения теста нужно указать местонахождение файла с тестом, для этого надо выполнить функции для файла, аналогичные для пользователей второй группы.

Функциональность разрабатываемых компонентов состоит в задании определенных параметров путём выбора некоторого набора или только одного параметра из заданного множества (каждый параметр определяется одним компонентом в активном или неактивном состоянии). Выбор осуществляется с помощью мыши или клавиатуры.

Дополнительная функциональность компонентов CheckBox и RadioButton состоит в изменяемости внешнего вида для разработчиков; для пользователей же – в измененном внешнем виде, что придает компоненту дополнительную гибкость в его использовании и позволяет получить соответствие с внешним видом системы-хозяина.

**2.9.** **Формализация сценариев действий пользователей**

***Сведения из теории***

На этом этапе частично изучаются, частично разрабатываются типовые сценарии действий пользователя: формализуются данные, необходимые пользователям для выполнения работы, последовательность самой работы, критерии завершенности этой работы.

Цель этого этапа – написать словесное описание взаимодействия пользователя с системой, не конкретизируя, как именно проходит взаимодействие, но уделяя возможно большее внимание всем целям пользователей. Количество сценариев может быть произвольным, главное, что они должны включать все типы задач, стоящих перед системой, и быть сколько-нибудь реалистичными. Сценарии очень удобно различать по именам участвующих в них вымышленных персонажей.

Возвратимся к примеру с тостером. Это очень простая система, поэтому для нее достаточно одного сценария.

«*Виктор Х. отрезает кусок хлеба и кладет его в тостер. Он включает тостер и ждет, пока хлеб не поджарится*»

Но это просто, ради такого результата нет необходимости применять какие-то специализированные методы проектирования. Обычно всё существенно сложнее.

Предположим, что необходимо разработать сценарии для будущей почтовой программы. Судя по всему, для этой задачи необходимо три сценария:

1. *Елизавета Петровна запускает почтовую программу. Она включает процесс скачивания новой почты. Получив почту, она читает все сообщения, затем часть их удаляет, а на одно сообщение отвечает. После чего выключает почтовую программу.*
2. *Андрей Фёдорович делает активным окно уже открытой почтовой программы и включает процесс скачивания новой почты. Получив почту, он ее читает. Одно сообщение он пересылает другому адресату, после чего удаляет его, а еще одно печатает. После чего переключается на другую задачу.*
3. *Пришло новое сообщение, и системный администратор Андрей реагирует на соответствующий индикатор. Он делает активным окно почтовой программы и открывает полученное сообщение. Он читает его, после чего перемещает его в другую папку. Затем переключается на другую задачу.*

Эти сценарии имеют двойную ценность. Во-первых, они будут полезны для последующего тестирования, поскольку тестируется не выполнение абстрактных задач, а выполнение конкретных, входящих в эти сценарии, операций. Во-вторых, сам факт их написания обычно, хоти и не всегда, приводит к лучшему пониманию устройства проектируемой системы, побуждая сразу же оптимизировать будущее взаимодействие. На таких сценариях очень хорошо заметны ненужные шаги. Например, в третьем сценарии системный администратор Андрей после получения индикатора не смог сразу же открыть новое сообщение, но должен был открыть окно системы, найти нужное сообщение, открыть его и только тогда прочесть. Понятно, что от этих ненужных этапов можно смело избавиться уже на этой ранней стадии проектирования.

**Практическая часть (Пример)**

Первый типовой сценарий «*Пользователь решил создать тест по математике по разделу «Решение простых дифференциальных уравнений». Для этого он предварительно создал в текстовом файле тест*».

Второй типовой сценарий «*Пользователь решил пройти тест. Для этого он должен указать расположение файла с тестом и получить оценку в некоторой шкале за этот тест*».

Для компонента CheckBox возможен следующий типовой сценарий «*Допустим, пользователь нажал левой клавишей мыши на некотором компоненте CheckBox. Если до этого компонент был неактивен, он переходит в активное состояние. Если он был в активном состоянии, он может перейти в неактивное состояние либо в промежуточное состояние («не уверен») – в зависимости от настроек компонента*».

Для компонента RadioButton возможен следующий типовой сценарий «*Допустим, пользователь нажал левой клавишей мыши на некотором компоненте RadioButton. Если до этого компонент был неактивен, он переходит в активное состояние. Если он был в активном состоянии, он в нём и остаётся. Далее пользователь может продолжать аналогичные действия с данным компонентом или другими, ему аналогичными, в этой группе или другой. При этом в группе может быть только один активный компонент*».

**2.10.** **Обзор интерфейса конкурирующих систем**

***Сведения из теории***

Большая часть аудитории любой системы обладает навыками использования нескольких конкурирующих систем; если разрабатываемый интерфейс полностью несхож с конкурентами, пользователям придется переучиваться. Кроме того, конкурирующие системы часто содержат эффективные решения, которые полезно перенять (или, что случается чаще, учесть при проектировании интерфейса).

Как и в случае экспертной оценки текущего интерфейса системы, отчет по выполнению этого этапа работ содержит перечень удачных и неудачных интерфейсных решений; в целом, однако, отчет более сфокусирован на удачных решениях.

**Практическая часть (Пример)**

На рис. 1, 2 приведены интерфейсы приложений тестирования по учебным дисциплинам.

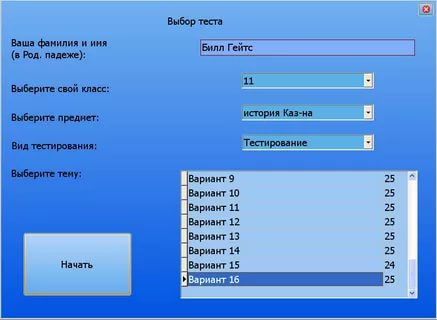
[](https://yandex.ru/images/search?pos=14&img_url=http://img.allsoft.ru/Screens/mig/2012/04/03/354617.png&text=%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81%D1%8B+%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC+%D0%BE%D1%80%D0%B3%D0%B0%D0%BD%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D0%B8+%D1%82%D0%B5%D1%81%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F+%D0%BF%D0%BE+%D1%83%D1%87%D0%B5%D0%B1%D0%BD%D1%8B%D0%BC+%D0%B4%D0%B8%D1%81%D1%86%D0%B8%D0%BF%D0%BB%D0%B8%D0%BD%D0%B0%D0%BC&rpt=simage&lr=5&source=wiz)

Рис.1.

На рис.1 показан интерфейс системы тестирования. На рис. 2 показан интерфейс системы создания тестов и и прохождения тестов.

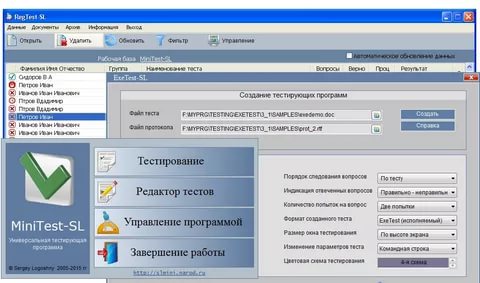
[](https://yandex.ru/images/search?pos=13&img_url=http://slmini.narod.ru/3t.png&text=%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81%D1%8B+%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC+%D0%BE%D1%80%D0%B3%D0%B0%D0%BD%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D0%B8+%D1%82%D0%B5%D1%81%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F+%D0%BF%D0%BE+%D1%83%D1%87%D0%B5%D0%B1%D0%BD%D1%8B%D0%BC+%D0%B4%D0%B8%D1%81%D1%86%D0%B8%D0%BF%D0%BB%D0%B8%D0%BD%D0%B0%D0%BC&rpt=simage&lr=5&source=wiz)

Рис.2.

В приведенных примерах систем тестирования нет ничего, заслуживающего внимания для заимствования.

**2.11.** **Формализация привычек и ожиданий пользователей**

***Сведения из теории***

На данном этапе изучаются субъективные ожидания пользователей от системы. Без этого исследования трудно или невозможно предугадать отношение пользователей к будущей системе.

**Практическая часть (Пример)**

Работа с указанными выше компонентами является обычной для пользователя, привыкшего работать в среде Windows, так как она осуществляется с помощью стандартных, наиболее часто используемых клавиш «Пробел» и левой кнопки мыши, а также клавиши Tab для перемещения.