МОДУЛЬ 9: «ТЕСТИРОВАНИЕ ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА»

ЦЕЛИ И ЗАДАЧИ ТЕСТИРОВАНИЯ ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА

ПОЛЬЗОВАТЕЛЬСКИЙ ИНТЕРФЕЙС – часть программной системы, обеспечивающая работу интерфейса с пользователем. Трудность тестирования пользовательского интерфейса заключается в двоякости восприятия термина “пользовательский интерфейс”.

1) Пользовательский интерфейс – часть программной системы.
   - ✓ Написание функциональных и низкоуровневых требований.
   - ✓ Составление тест-требований и тест-планов.
   - ✓ Определение реакции системы на каждое действие пользователя (при помощи клавиатуры, мыши или иного устройства ввода).
   - ✓ Оценка вывода информационных сообщений системы (на экране, на печатающем устройстве или на ином другом устройстве вывода).
   - ✓ Проверка функциональной полноты пользовательского интерфейса (соответствие реализованных функций требованиям; правильность вывода информации и т.д.).
2) Пользовательский интерфейс – “лицо” системы, и от его продуманности зависит эффективность работы пользователя с системой.
   - ✓ Проверка интерфейса на эффективность человеко-машинного взаимодействия (usability verification).
   - ✓ Удобство использования - учет в виде общих рекомендаций и принципов построения пользовательского интерфейса.

ФУНКЦИОНАЛЬНОЕ ТЕСТИРОВАНИЕ ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА

Функциональное тестирование пользовательского интерфейса состоит из пяти фаз:

- ✓ анализ требований к пользовательскому интерфейсу;
- ✓ разработка тест-требований и тест-планов для проверки пользовательского интерфейса;
- ✓ выполнение тестов и сбор информации о выполнении тестов;
- ✓ определение полноты покрытия пользовательского интерфейса требованиями;
- ✓ составление отчетов о проблемах в случае несовпадения поведения системы и требований, либо в случае отсутствия требований на отдельные интерфейсные элементы.

ТЕСТ-ПЛАНЫ представляют собой сценарии, описывающие действия пользователя при работе с системой. Сценарии записаны или на естественном языке, или на формальном языке системы автоматизации пользовательского интерфейса.

ВЫПОЛНЕНИЕ ТЕСТОВ производится или оператором в ручном режиме, или системой, которая эмулирует поведение оператора.

ВЫПОЛНЕНИЕ ТЕСТОВЫХ ПРИМЕРОВ оценивается или из анализа выводимых на экран форм и их элементов (в случае графического интерфейса), или из анализа выводимого на экран текста (в случае текстового интерфейса).

ПОЛНОТА ПОКРЫТИЯ ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА – при выполнении всех тестовых примеров необходимо, чтобы каждый интерфейсный элемент был использован хотя бы один раз во всех доступных режимах.

ОТЧЕТ О ПРОБЛЕМАХ пользовательского интерфейса: а) описание несоответствия требований и реального поведения системы; б) описание проблем в требованиях к пользовательскому интерфейсу.

ПРОВЕРКА ТРЕБОВАНИЙ К ПОЛЬЗОВАТЕЛЬСКОМУ ИНТЕРФЕЙСУ

1. ТИПЫ ТРЕБОВАНИЙ К ПОЛЬЗОВАТЕЛЬСКОМУ ИНТЕРФЕЙСУ.

Требования к внешнему виду пользовательского интерфейса и форме взаимодействия с пользователем.

1.1. Требования к размещению элементов управления на экранных формах.

1.2. Требования к содержанию и оформлению выводимых сообщений.

1.3. Требования к форматам ввода.

Требования по доступу к внутренней функциональности системы при помощи пользовательского интерфейса.

1.4. Требования к реакции системы на ввод пользователя.

1.5. Требования к времени отклика на команды пользователя.

1.1. ТРЕБОВАНИЯ К РАЗМЕЩЕНИЮ ЭЛЕМЕНТОВ УПРАВЛЕНИЯ НА ЭКРАННЫХ ФОРМАХ.

Определяют общие принципы размещения элементов пользовательского интерфейса и их конкретных элементов.

Пример общего требования.

> *Каждое окно приложения должно быть разбито на три части: строка меню, рабочая область и статусная строка. Строка меню должна быть горизонтальной и прижатой к верхней части окна, статусная строка должна быть горизонтальной и прижата к нижней части окна, рабочая область должна находиться между строкой меню и статусной строкой и занимать всю оставшуюся площадь окна.*

ТЕСТИРОВАНИЕ. Достаточно определить, что в каждом окне системы присутствуют три части, которые расположены и прижаты согласно требованиям (при изменении размеров окна, его сворачивании/разворачивании, перемещении по экрану, при перекрытии его другими окнами).

Пример требования к конкретному элементу.

> *Кнопка «Начать передачу» должна находиться непосредственно под строкой меню в левой части рабочей области окна.*

ТЕСТИРОВАНИЕ. Необходимо определить, сохраняется ли расположение элемента при изменении размера окна, а также при нажатии.

1.2. ТРЕБОВАНИЯ К СОДЕРЖАНИЮ И ОФОРМЛЕНИЮ ВЫВОДИМЫХ СООБЩЕНИЙ.

Определяют текст сообщений, выводимых системой, его шрифтовое и цветовое оформление, а также и в каких случаях выводится то или иное сообщение.

Пример.

> *Сообщение «Невозможно открыть файл» должно выводиться в статусную строку прижатым к левому краю красным цветом полужирным шрифтом, если открываемый файл недоступен по чтению.*

ТЕСТИРОВАНИЕ. Необходимо проверить, что при возникновении указанной ситуации сообщение выводится согласно требованиям.

Пример.

> *Сообщения об ошибках должны выводиться в статусную строку, прижатыми к левому краю красным цветом полужирным шрифтом.*

ТЕСТИРОВАНИЕ. Необходимо проверять форматы всех возможных сообщений об ошибках программы во всех возможных ошибочных ситуациях.

1.3. ТРЕБОВАНИЯ К ФОРМАТАМ ВВОДА.

Определяет, в каком виде информация поступает от пользователя в систему.

ТЕСТИРОВАНИЕ. Проверка, как корректного ввода данных, так и некорректного ввода данных.

1.4. ТРЕБОВАНИЯ К РЕАКЦИИ СИСТЕМЫ НА ВВОД ПОЛЬЗОВАТЕЛЯ.

Определяет связь внутренней логики системы и интерфейсных элементов.

Пример.

> *При нажатии кнопки «Сброс» значение таймера синхронизации передачи должно сбрасываться в 0.*

ТЕСТИРОВАНИЕ. В тестовом примере необходимо сымитировать нажатие на кнопку “Сброс”, потом проверить значения таймера.

Пример реакции системы.

> *При нажатии кнопки «Отложенный сброс» должно выводиться окно «Ввод значения времени для отложенного сброса».*

ТЕСТИРОВАНИЕ. Имитация ввода пользователя и анализ появляющихся интерфейсных элементов.

1.5. ТРЕБОВАНИЯ К ВРЕМЕНИ ОТКЛИКА НА КОМАНДЫ ПОЛЬЗОВАТЕЛЯ.

Операции продолжительностью более 1 секунды пользователем воспринимаются как длительные. Система должна сообщать пользователю о выполнении какой-либо операции, чтобы пользователь не считал, что система зависла или работает в неверном режиме.

ТЕСТИРОВАНИЕ. Значения предельного времени и равномерность увеличения значений индикатора прогресса, указывающего выполнение операции.

2. ТЕСТОПРИГОДНОСТЬ ТРЕБОВАНИЙ К ПОЛЬЗОВАТЕЛЬСКОМУ ИНТЕРФЕЙСУ.

ТЕСТОНЕПРИГОДНЫЕ ТРЕБОВАНИЯ – требования, описывающие субъективные характеристики интерфейса, которые не могут быть точно определены или измерены при выполнении тестовых примеров.

Пример тестонепригодного требования.

> *Пользовательский интерфейс должен быть интуитивно понятным.*

ТЕСТИРОВАНИЕ. Без определения четких критериев интуитивной понятности проверка такого требования невозможна.

Пример детерминированного критерия.

> *Под интуитивной понятностью интерфейса понимается доступность любой функции системы при помощи не более чем 5 щелчков мыши по интерфейсным элементам.*

ТЕСТИРОВАНИЕ. Возможно как ручное, так и автоматизированное тестирование.

Пример вероятностного критерия.

> *Под интуитивной понятностью интерфейса понимается, что пользователь обращается к руководству пользователя не чаще, чем раз в пять минут на этапе обучения и не чаще, чем раз в 2 часа на этапе активного использования системы. Значения должны быть получены на репрезентативной выборке пользователей не менее 1000 человек.*

ТЕСТИРОВАНИЕ. Есть четкий критерий, при использовании которого результаты тестирования могут быть воспроизведены.

ПОЛНОТА ПОКРЫТИЯ ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА

Включает следующие уровни:

- ✓ функциональное покрытие – покрытие требований к пользовательскому интерфейсу;
- ✓ структурное покрытие – каждый интерфейсный элемент должен быть использован в тестовых примерах хотя бы один раз;
- ✓ структурное покрытие с учетом состояния элементов интерфейса – каждый элемент интерфейса должен быть приведен во все возможные состояния (например, для чек-боксов – отмечен/не отмечен, для полей ввода – пустое/заполненное не целиком/заполненное полностью и т.п.);
- ✓ структурное покрытие с учетом состояния элементов интерфейса и внутреннего состояния системы – каждое различимое поведение интерфейсного элемента должно быть проверено (например, система имеет два режима работы – для начинающего пользователя, с появлением всплывающей подсказки и нормальный режим).

При определении степени покрытия учитывается, что реакция на некоторые интерфейсные элементы определяется не программной системой, а на уровне операционной системы (например, реакция на использование многих интерфейсных элементов стандартного диалогового окна открытия файла определяется операционной системой и может не тестироваться).

При недостаточном уровне покрытия интерфейса тестами необходимо уточнение требований к пользовательскому интерфейсу или снижение подробности тестирования.

МЕТОДЫ ТЕСТИРОВАНИЯ ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА

Функциональное тестирование пользовательского интерфейса:

- ✓ ручное тестирование, при непосредственном участии оператора;
- ✓ автоматическое тестирование, при помощи различного инструментария.

1. РУЧНОЕ ТЕСТИРОВАНИЕ.

РУЧНОЕ ТЕСТИРОВАНИЕ пользовательского интерфейса проводится тестировщиком на основе тестовых примеров в виде набора сценариев.

СЦЕНАРИЙ – перечисление последовательности действий тестировщика и описание результатов тестирования ответных реакций системы.

ТАБЛИЦА – форма записи сценария. Первая колонка – описание действий; вторая колонка – ожидаемая реакция системы; третья колонка – совпадение или не совпадение ожидаемой реакции системы с реальной (Таблица 7 [Синицын 2006]).

Таблица 7. Пример сценария для ручного тестирования пользовательского интерфейса

№ п/п	Действие	Реакция системы	Результат
1	Щелкните на пиктограмме System и выберите пункт меню ‘System Management Applet’.	Появится окно ввода логина и пароля	Верно
2	Введите в появившееся окно ввода имя пользователя ‘guest1’ и пароль ‘guest’. Затем нажмите кнопку ‘Login’.	Появится окно ‘System Management Applet’. В верхнем правом углу должно быть выведено имя вошедшего пользователя guest1. Все опции в окне должны быть отключены (выведены серым цветом).	Неверно Окно имеет название ‘System Management Application’
3	Завершите сеанс работы с апплетом щелчком по пиктограмме ‘Logout’.	Окно ‘System Management Applet’ должно быть закрыто.	Верно

УДОБСТВО РУЧНОГО ТЕСТИРОВАНИЯ.

- ✓ Контроль корректности интерфейса проводится человеком, т.е. “потребителем” данной части программной системы.

- ✓ Анализ успешности теста при косметических изменениях в интерфейсе системы (например, при перемещении кнопок управления на несколько пикселей) будет выполняться согласно человеческому восприятию.

НЕДОСТАТКИ РУЧНОГО ТЕСТИРОВАНИЯ.

- ✓ Требуются значительные человеческие и временные ресурсы.
- ✓ При проведении регрессионного тестирования и вообще любого повторного тестирования – на каждой итерации повторного тестирования пользовательского интерфейса требуется участие тестировщика.

2. СЦЕНАРИИ НА ФОРМАЛЬНЫХ ЯЗЫКАХ.

АВТОМАТИЗАЦИЯ ТЕСТИРОВАНИЯ пользовательского интерфейса – использование программных инструментов, эмулирующих поведение тестировщика при ручном тестировании. В качестве входной информации используются сценарии тестов, записанные на некотором формальном языке.

Для определения ожидаемого состояния пользовательского интерфейса используются либо анализ снимков экрана и сравнение их с эталонными, либо доступ к данным интерфейсных элементов средствами операционной системы.

При передаче информации в тестируемый интерфейс и при получении информации для анализа используются следующие доступы к элементам интерфейса:

- ✓ позиционный, при котором доступ к элементу осуществляется при помощи задания его абсолютных (относительно экрана) или относительных (относительно окна) координат и размеров;
- ✓ по идентификатору, при котором доступ к элементу осуществляется через получение интерфейсного элемента при помощи его уникального идентификатора в пределах окна.

ПЕРВЫЙ МЕТОД. При регрессионном тестировании после внесения изменений в пользовательский интерфейс будет выявлено много не прошедших тестов (достаточно изменения местоположения одного ключевого интерфейсного элемента, как все сценарии начнут работать неверно). Поэтому необходимо менять значительную часть сценариев в системе тестов при каждом изменении интерфейса системы. Такой метод автоматизации тестирования подходит для систем с устоявшимся и редко изменяемым интерфейсом.

ВТОРОЙ МЕТОД. Этот метод более устойчив к изменению расположения интерфейсных элементов, но изменения тестовых примеров могут потребоваться и здесь при изменении логики работы интерфейсных элементов.

> *Пусть в первой версии системы при нажатии на кнопку “Передать данные“ передача данных начинается сразу и выводится окно с индикатором прогресса. Сценарий тестового примера включает в себя имитацию нажатия на кнопку и обращение к индикатору прогресса для получения значения прогресса в процентах.*

> *Если во второй версии системы после нажатия на кнопку “Передать данные“ вначале выводится окно «Вы уверены?» с кнопками «Да» и «Нет», то для проверки работы индикатора прогресса в тестовый сценарий необходимо добавить имитацию нажатия кнопки «Да».*

ТЕСТИРОВАНИЕ УДОБСТВА ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА

УДОБСТВО ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА (USABILITY) – простота доступа пользователя к функциям системы через человеко-машинный (пользовательский) интерфейс.

Тестирование удобства пользовательского интерфейса должно учитывать знания, как в области программной инженерии, так и в физиологии, психологии и эргономике.

На удобство использования пользовательского интерфейса влияют факторы:

- ✓ легкость обучения – быстро ли человек учится использовать систему;
- ✓ эффективность обучения – быстро ли человек работает после обучения;
- ✓ запоминаемость обучения – легко ли человек запоминается все, чему он научился;
- ✓ ошибки – часто ли человек допускает ошибки в работе;
- ✓ общая удовлетворенность – является ли положительным общее впечатление от работы с системой.

Эти факторы, несмотря на свою неформальность, могут быть измерены. Для этого выбирается группа типичных пользователей системы, и измеряются показатели их работы (например, количество допущенных ошибок), а также анализируются впечатления от работы с системой при помощи заполнения опросных листов.

Этапы тестирования удобства использования пользовательского интерфейса:

1. **ИССЛЕДОВАТЕЛЬСКИЙ** – после формулирования требований к системе и разработки прототипа интерфейса. Цель – провести высокоуровневое обследование интерфейса и выяснить, позволяет ли он решать задачи пользователя.
2. **ОЦЕНОЧНЫЙ** – после разработки низкоуровневых требований и детального прототипа пользовательского интерфейса. Цель – провести количественные измерения характеристик пользовательского интерфейса (количество обращений к системе помощи по отношению к количеству совершенных операций, количество ошибочных операций, время устранения последствий ошибочных операций и т.п.).
3. **ВАЛИДАЦИОННЫЙ** – ближе к этапу завершения разработки. Цель – провести анализ соответствия интерфейса программной системы стандартам, регламентирующим вопросы удобства интерфейса, проводится общее тестирование всех компонент пользовательского интерфейса с точки зрения конечного пользователя.
4. **СРАВНИТЕЛЬНЫЙ** – на любом этапе разработки интерфейса. Цель – сравнение двух или более вариантов реализации пользовательского интерфейса.

При тестировании удобства использования пользовательского интерфейса часто используются некоторые эвристические критерии, которые заменяют точные оценки в классическом тестировании программных систем. Якоб Нильсен выделил следующие 10 эвристических характеристик удобного пользовательского интерфейса [Nielsen 2004]:

1. **НАБЛЮДАЕМОСТЬ СОСТОЯНИЯ СИСТЕМЫ** – система всегда должна оповещать пользователя о том, что она в данный момент делает, причем через разумные промежутки времени.
2. **СООТНЕСЕНИЕ С РЕАЛЬНЫМ МИРОМ** – терминология, использованная в интерфейсе системы должна быть терминологией проблемной области пользователя, а не технической терминологией.
3. **ПОЛЬЗОВАТЕЛЬСКОЕ УПРАВЛЕНИЕ И СВОБОДА ДЕЙСТВИЙ** – предоставление определенного “аварийного выхода”, при помощи которого можно вернуться к предыдущему нормальному состоянию. К таким “аварийным выходам” относятся, например, функции обратного отката.
4. **ЦЕЛОСТНОСТЬ И СТАНДАРТЫ** – для обозначения одних и тех же объектов, ситуаций и действий должны использоваться одинаковые слова во всех частях интерфейса.

5. **ПОМОЩЬ ПОЛЬЗОВАТЕЛЯМ** в распознавании, диагностике и устранении ошибок – Сообщения об ошибках должны определять суть возникшей проблемы и предлагать ее конструктивное решение.
6. **ПРЕДОТВРАЩЕНИЕ ОШИБОК** – необходимо либо полностью устранить элементы, в которых могут возникать ошибки пользователя, либо проверять ввод пользователя и сообщать ему о потенциально возможном возникновении проблемы.
7. **РАСПОЗНАВАНИЕ, А НЕ ВСПОМИНАНИЕ** – минимизация нагрузки на память пользователя (объекты, действия и опции – ясные, доступные и явно видимые).
8. **ГИБКОСТЬ И ЭФФЕКТИВНОСТЬ ИСПОЛЬЗОВАНИЯ** – горячие клавиши в интерфейсе, т.е. система должна предоставлять два способа работы – для новичков и для опытных пользователей.
9. **ЭСТЕТИЧНЫЙ И МИНИМАЛЬНО НЕОБХОДИМЫЙ ДИЗАЙН** – окна не должны содержать не относящуюся к делу или редко используемую информацию.
10. **ПОМОЩЬ И ДОКУМЕНТАЦИЯ** – документация должна быть структурирована так, чтобы пользователь мог легко найти нужный раздел, посвященный решаемой им задаче. Каждый такой раздел должен содержать пошаговые руководства по выполнению задачи.

Эти эвристики могут использоваться при тестировании удобства использования пользовательского интерфейса. Один из эффективных методов проверки интерфейса на удобство – использование формальной инспекции.

ЛИТЕРАТУРА

[Синицын 2006] – Синицын С.В., Налютин Н.Ю. Верификация программного обеспечения. Курс лекций. Московский инженерно-физический институт. М. 2006.

[Nielsen 2004] – Nielsen J. Ten Usability Heuristics (http://www.useit.com/papers/heuristic_list.html)