**Проектирование пользовательского интерфейса**

Существуют четыре версии прототипа. Особый интерес представляют первые две. Третья и четвертая используются редко, т.к. не сильно отличаются от готовой программы.

Первая версия. Бумажная.

Вторая версия. Презентационная.

Третья версия. Псевдореальная.

Четвертая версия. Реальная.

**Первая версия. Бумажная**

Необходимо нарисовать на бумаге все экраны и диалоговые окна (или распечатать соответствующие части схемы). Нужно только убедиться, что все интерфейсные элементы выглядят единообразно и сколько-нибудь похоже на реальные. Эта распечатка и является первым прототипом. На нём вполне можно тестировать восприятие системы пользователем и её основную логику.

**Вторая версия. Презентационная.**

После исчерпания возможностей бумажной версии прототипа стоит создать новую версию. Для этого точно так же рисуется интерфейс, но уже не на бумаге, но в какой-либо презентационной программе. С этой версией прототипа можно тестировать значительно более сложное взаимодействие человека с системой, нежели с бумажной. С другой стороны, исправление найденных ошибок значительно более трудоемко.

**Третья версия. Псевдореальная.**

В тех случаях, когда в интерфейсе появляются нестандартные элементы или необходимо проверить реальную скорость взаимодействия пользователя с системой, создается еще одна версия прототипа – реально выглядящая, но лишенная каких-либо алгоритмов и, соответственно, не показывающая реальных данных. Делать этот вариант можно как в средах разработки, благо в них есть визуальные инструменты создания интерфейсов, так и в редакторах изображений, что обычно быстрее. Фактически при этом создаются фальшивые снимки экрана, на которых и производят тестирование. Понятно, что существенно модифицировать эти экраны затруднительно, так что лучше не увлекаться такой работой, не получив каких-либо гарантий ее правильности.

**Четвертая версия. Реальная.**

Иногда необходимо тестировать взаимодействие пользователя не только с интерфейсом системы, но и с обрабатываемыми системой данными. Например, работая с графической программой, пользователь не только нажимает на экранные кнопки, но также создает и модифицирует изображения мышью. Область же редактирования данных зачастую вообще не содержит каких-либо визуальных интерфейсных элементов, из чего вовсе не следует, что интерфейса в ней нет, его, наоборот, много. Другой разговор, что счет в нем идет не на кнопки и переключатели, но на пиксели и миллисекунды.

Понятно, что создание прототипа в таких условиях не поможет, поскольку прототип вообще не будет отличаться от проектируемой системы. В таких условиях лучше всего написать нужные участки кода до написания всего остального, и проводить тестирование уже на реальной системе.

**Анализ создаваемого интерфейса**

* **Предсказание скорости**
* В 1983 году Кард, Моран и Ньювел создали метод оценки скорости работы с системой, названный аббревиатурой GOMS (Goals, Operators, Methods, and Selection Rules – цели, операторы, методы и правила их выбора).
* **Предсказание скорости**

Идея метода очень проста: все действия пользователя можно разложить на составляющие (например, взять мышь или передвинуть курсор). Ограничив номенклатуру этих составляющих, можно замерить время их выполнения на массе пользователей, после чего получить статистически верные значения длительности этих составляющих. После чего предсказание скорости выполнения какой-либо задачи, или, вернее, выбор наиболее эффективного решения, становится довольно простым делом – нужно только разложить эту задачу на составляющие, после чего, зная продолжительность каждой составляющей, всё сложить и узнать длительность всего процесса.

* Исследователи обнаружили, то для большей части сравнительного анализа задач, включающих использование клавиатуры и мыши, вместо проведения измерений для каждого отдельно пользователя можно применить набор стандартных интервалов. С помощью тщательных лабораторных исследований был получен набор временных интервалов, требуемых для выполнения различных действий.

|  |  |  |  |
| --- | --- | --- | --- |
| **Тип** | **Действие** | **Длительность, с** | **Комментарии** |
| K | Нажатие на клавишу клавиатуры | 0,28 | Включая клавиши Alt, Ctrl, Shift |
| M | Нажатие на кнопку мыши | 0,1 |  |
| П | Перемещение курсора мыши | 1,1 | Время затрачиваемое на перемещение курсора зависит как от дистанции, так и от размера цели. Тем не менее это число представляет достаточно точный компромисс. |
| В | Перемещение руки с мыши на клавиатуру или наоборот. | 0,4 |  |
| Д | Ментальная подготовка. | 1,2 | Время, необходимое пользователю для того, чтобы умственно подготовиться к следующему шагу. |
| Р | Время реакции системы | от 0,1 до бесконечности | Для базовых операций, таких как работа с меню, это время можно не засчитывать. |

* На практике указанные значения могут варьироваться в широких пределах. Для опытного пользователя, способного печатать со скоростью 135 слов/мин., значение К может составлять 0,08 с, для обычного пользователя, имеющего скорость 55 слов/мин., - 0,2 с, для среднего неопытного пользователя, имеющего скорость 40 слов/мин., - 0,28 с, а для начинающего – 1,2 с. Нельзя сказать, что скорость набора не зависит от того, что именно набирается. Для того чтобы набрать одну букву из группы случайно взятых букв, большинству людей требуется около 0,5 с. Если же это какой-то запутанный код (например адрес электронной почты), то у большинства людей скорость набора составит около 0,75 символов в секунду. Значение К включает в себя и то время, которое необходимо пользователю для исправления сразу замеченных ошибок.
* Более сложным является определения точек, в которых пользователь остановится, чтобы выполнить бессознательную ментальную операцию, - интервалы ментальной подготовки М. Вот основные правила, позволяющие определить, в какие моменты будут проходить ментальные операции:
* Операторы М следует устанавливать перед всеми операторами К, а также перед всеми операторами Р, предназначенными для выбора команд; но перед операторами Р, предназначенными для указания на аргументы этих команд, ставить оператор М не следует.
* Если оператор, следующий за оператором М, является полностью ожидаемым с точки зрения оператора, предшествующего И, то этот оператор М может быть удален. Например, если вы перемещаете мышь с намерением нажать ее кнопку по достижении цели движения, то в соответствии с этим правилом следует удалить оператор М, устанавливаемый по первому правилу. В этом случае последовательность РМК превращается в РК.
* Если срока вида МКМКМК принадлежит когнитивное единице, то следует удалить все операторы М, кроме первого. Когнитивной единицей является непрерывная последовательность вводимых символов, которые могут образовывать название команды или аргумент. Например Y, перемещать, Елена Троянская или 4567.34 являются примерами когнитивных единиц.
* Если оператор К означает лишний разделитель, стоящий в конце когнитивной единицы (например, разделитель команды, следующий сращу за разделителем аргумента этой команды), то следует удалить оператор М, стоящий перед ним.
* Если оператор К является разделителем, стоящим после постоянной строки (например, название команды или любая последовательность символов, которая каждый раз вводится в неизменном виде), то следует удалить оператор М, стоящий перед ним. (Добавление разделителя станет привычным действием, и поэтому разделитель станет частью строки и не будет требовать специального оператора М.) Но если оператор К является разделителем для строки аргументов или любой другой изменяемой строки, то оператор М следует сохранить перед ним.
* Любую часть оператора М, которая перекрывает оператор Р, учитывать не следует.
* Этот метод имеет определенные недостатки:
* он применим в основном для предсказания действий опытных пользователей;
* он никак не учитывает ни прогресса в обучении, ни возможных ошибок, ни степени удовлетворения пользователей;
* он плохо применим при проектировании сайтов из-за непредсказуемого времени реакции системы.
* ***Измерение эффективности интерфейса***
* Чтобы сделать правильную оценку времени, необходимого на выполнение задачи с помощью самого быстрого интерфейса, прежде всего следует определить минимальное количество информации, которое пользователь должен ввести, чтобы выполнить задачу. Это минимальное количество не зависит от модели интерфейса.
* Если методы работы, используемые в предполагаемом интерфейсе, требуют введения такого количества информации, которое превышает минимальное, это означает, что пользователь делает лишнюю работу, и поэтому интерфейс можно усовершенствовать.
* ***Информационная производительность*** интерфейса Е определяется как отношение минимального количества информации, необходимого для выполнения задачи, к количеству информации, которое должен ввести пользователь. Параметр Е может изменяться в пределах от 0 до 1.
* Если никакой работы для выполнения задачи не требуется или работа просто не производится, то производительность составляет 1.
* Производительность Е может равняться и 0 в случаях, когда пользователь должен ввести информацию, которая совершенно бесполезна. Следует отметить, что в интерфейсах можно встретить немало деталей, которые имеют сомнительную ценность из-за параметра Е = 0. Примером такого бесполезного элемента может быть диалоговое окно, в котором есть только одна-единственная возможность для действия пользователя, например кнопка ОК.
* В параметре Е учитывается только информация, необходимая для задачи, и информация, вводимая пользователем. Два или более методов действия могут иметь одинаковую производительность Е, но иметь разное время выполнения. Возможно даже, что один метод имеет более высокий показатель Е, но действует медленнее, чем другой метод.
* Информация измеряется в битах. Один бит, который представляет собой один из двух альтернативных вариантов (таких как 0 или 1, да или нет), является единицей информации. Например, чтобы выбрать один из каких-либо четырех объектов, потребуется 2 бита информации. Чтобы сделать выбор из группы восьми элементов, потребуется 3 бита. Из шестнадцати элементов — 4 бита, и т. д. В общем случае при количестве n равновероятных вариантов суммарное количество передаваемой информации определяется как степень 2, равная n: lg2(n)
* Количество информации для каждого варианта определяется как lg2(n)/n
* Если вероятности для каждой альтернативы не являются равными и i-я альтернатива имеет вероятность p(i), то информация, передаваемая этой альтернативой, определяется как lg2(1/p(i))\*p(i)
* Информационное содержание интерфейса, в котором возможно сделать только нажатие единственной клавиши (а не нажатие клавиши не допускается), составляет 0 бит:1\*lg2(1)
* Можно оценить объем информации, содержащейся в сообщении, только в контексте всего набора возможных сообщений. Чтобы подсчитать количество информации, передаваемой некоторым полученным сообщением, необходимо знать в частности вероятность, с которой это сообщение может быть отправлено.
* Количество информации в любом сообщении не зависит от других сообщений, которые были в прошлом или могут быть в будущем, не связано со временем или продолжительностью и не зависит от каких-либо иных событий.
* Для данного рассмотрения достаточно использовать упомянутые вероятности отдельных, единичных событий, при этом будем исходить из того, что все сообщения являются независимыми друг от друга.
* Однако действия, которые совершает пользователь при выполнении задачи, можно с большей точностью смоделировать в виде процесса Маркова, в котором вероятность последующих действий зависит от уже совершенных пользователем действий.