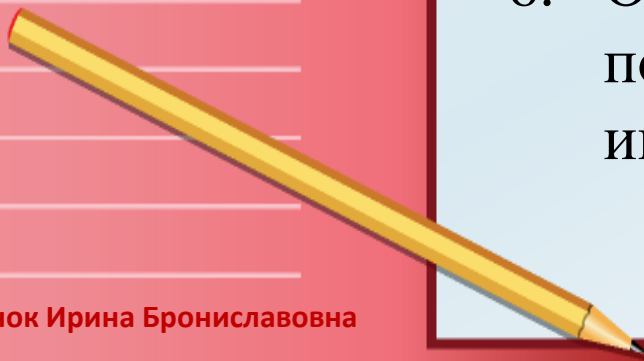





Глава 4

«Критерии качества интерфейса пользователя»



Тема 11. Количество человеческих ошибок.

1. Природа существования ошибок. Типы ошибок.
 2. Ошибки, вызванные недостаточным знанием предметной области.
 3. Опечатки.
 4. Ошибки, вызванные не считыванием показаний системы.
 5. Моторные ошибки.
 6. Два уровня ошибок и обратная связь.
- 

Количество человеческих ошибок

Важным критерием
эффективности
интерфейса является
количество человеческих
ошибок.

В некоторых случаях одна или две человеческих ошибки погоды не делают, но только тогда, когда эти ошибки легко исправляются.

Однако часто минимальная ошибка приводит к совершенно катастрофическим последствиям.

Пример 1 За одну секунду оператор в банке может сделать кого-то богаче, а банк, в свою очередь, беднее (впрочем, обычно беднее становятся все).

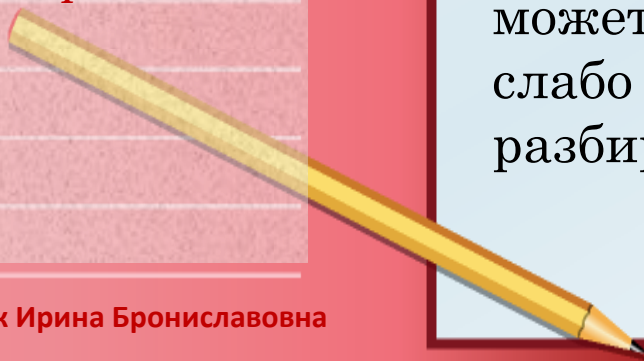
Пример 2 Классический сюжет из жизни летчика, который после взлета хотел убрать шасси, но вместо этого включил систему аварийного катапультирования, возник отнюдь не на пустом месте.



Природа существова ния ошибок

Вначале необходимо
сказать главное: *человек
при работе с компьютером
постоянно совершает
ошибки.*

Компьютеры (как и все
сложные технические
системы) *вообще не могут
быть используемы
человеком без совершения
ошибок.*




Компьютеры требуют от человека:

- ☐ точности,
- ☐ логического мышления,
- ☐ способности абстрагироваться от
идей реального мира.

Человек же практически на это не
способен.

Человек не цифровая система,
неспособная на ошибку, но система
аналоговая.

Именно благодаря этому он плох в
логике, зато имеет интуицию, не
приспособлен к точности, зато
может подстраиваться к ситуации,
слабо абстрагируется, зато хорошо
разбирается в реальном мире.



Человеческая ошибка

Термин «**человеческая ошибка**» до сих пор существует только по двум причинам.


❑ Во-первых, люди в ошибках системы склонны винить себя, поскольку по собственному эгоцентризму полагают, что подобные вещи происходят только с ними.

❑ Во-вторых, существующее положение вещей очень выгодно всякому руководству: гораздо легче уволить кого-либо, нежели признать, что система спроектирована плохо.


Под словосочетанием «**человеческая ошибка**» нужно понимать **«действие пользователя, не совпадающее с целью действий этого пользователя»**.



Типы ошибок





Наибольшее количество человеческих ошибок при использовании ПО раскладывается на четыре типа (сильно упрощенно, разумеется):

- ☐ Ошибки, вызванные недостаточным знанием предметной области.
 - ☐ Опечатки.
 - ☐ Ошибки, вызванные не считыванием показаний системы.
 - ☐ Моторные ошибки
- 



Типы ошибок


***Ошибки, вызванные
недостаточным
знанием предметной
области.***



Теоретически, эти *ошибки* методологических проблем не вызывают, сравнительно *легко* *исправляясь обучением пользователей.*

Практически же, роль этих ошибок чрезвычайно велика – никого не удивляет, когда оператора радарной установки перед началом работы оператором долго учат работать, и в то же время все ожидают должного уровня подготовки от пользователей ПО, которых никто никогда ничему целенаправленно не обучал.

Еще хуже ситуация с *сайтами*, у которых *даже справочной системы почти никогда не бывает.*



Типы ошибок

Опечатки


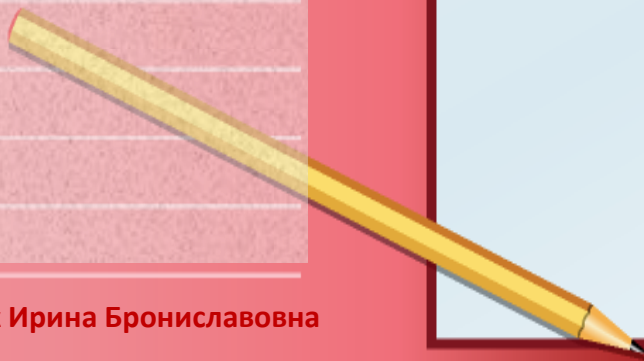
Опечатки происходят в двух случаях:

- ❑ когда не все внимание уделяется выполнению текущего действия (этот тип ошибок характерен, прежде всего, для опытных пользователей, не проверяющих каждый свой шаг);
- ❑ когда в мысленный план выполняемого действия вклинивается фрагмент плана из другого действия (происходит преимущественно в случаях, когда пользователь имеет обдуманное текущее действие и уже обдумывает следующее действие).



Типы ошибок


*Ошибки, вызванные
не считыванием
показаний системы*



Данные ошибки одинаково охотно производят как опытные, так и неопытные пользователи.

Первые не считывают показаний системы потому, что у них уже сложилось мнение о текущем состоянии, и они считают излишним его проверять.

Вторые – потому что они либо забывают считывать показания, либо не знают, что это нужно делать и как.



Типы ошибок

Моторные ошибки

Моторные ошибки – фактически их количество, пренебрежимо мало, но к сожалению, не так мало, чтобы вовсе их не учитывать.

Сущностью этих ошибок являются ситуации, когда пользователь знает, что он должен сделать, знает, как этого добиться, но не может выполнить действие нормально из-за того, что физические действия, которые нужно выполнить, выполнить трудно.

Так, никто не может с первого раза (и со второго тоже) нажать на экранную кнопку размером 1 на 1 пиксель. При увеличении размеров кнопки вероятность ошибки снижается, но почти никогда не достигает нуля.

Соответственно, единственным средством избежать этих ошибок является снижение требований к точности движений пользователя.

Моторные ошибки

Для успешного пользования любой системой пользователям необходима определенная степень бдительности, но эта же бдительность пользователям и неприятна.

Одно из важных понятий инженерной психологии – бдительность, т.е. *способность оператора в течение продолжительного времени направлять существенную часть своего внимания на состояние системы.*

Как показывает практика, ни один человек не способен долгое время обеспечивать бдительность без существенных потерь: мозг стремится найти себе более интересное занятие, отчего накапливается усталость и стресс.

Как только бдительность снижается, количество ошибок возрастает в разы.

В систему можно ввести индикатор опасности текущего состояния, при этом пользователь получает право быть не слишком бдительным большую часть времени, но зато получает и обязанность быть максимально собранным, когда горит «красная лампочка».

Моторные ошибки

Заметим, что крайне ценная возможность последующей отмены (Undo) деструктивных действий сама по себе не служит уменьшению количества человеческих ошибок, но помогает только уменьшить урон от них.

В действительности надо стремиться минимизировать количество ошибок, поскольку только это позволяет сберечь время (т.е. повысить производительность) и сделать пользователей более счастливыми за счет отсутствия дискомфорта.

Суммируя, при борьбе с ошибками нужно направлять усилия на:

- ☐ *плавное обучение пользователей в процессе работы;*
- ☐ *снижение требований к бдительности;*
- ☐ *повышение разборчивости и заметности индикаторов;*
- ☐ *снижение чувствительности системы к ошибкам.*

Моторные ошибки

Для этого есть три основных способа, а именно:

- ☐ *блокировка потенциально опасных действий пользователя до получения подтверждения правильности действия*
- ☐ *проверка системой всех действий пользователя перед их принятием*
- ☐ *самостоятельный выбор системой необходимых команд или параметров, при этом от пользователя требуется только проверка.*

Самым эффективным является третий способ. К сожалению, этот способ наиболее труден в реализации.

Моторные ошибки

*Блокировка
потенциально опасных
действий до получения
подтверждения*

*Не делайте опасные для
пользователя кнопки
кнопками по
умолчанию.*

К типу блокировки относится снятие фокуса ввода с кнопок конечных действий, чтобы пользователь не мог, не разобравшись, нажать на кнопку **Enter** и тем самым начать потенциально опасное действие.

Если пользователям приходится прилагать какие-либо усилия, чтобы запустить действие, есть надежда, что во время совершения этих усилий он заметит вкраившуюся ошибку.

Обычно проще всего в опасных случаях не делать главную кнопку кнопкой по умолчанию.

Также, важно не делать кнопку **Отмена** кнопкой по умолчанию (как часто случается). Если это сделать, пользователи будут ошибочно закрывать окно, т.е. одна ошибка заменит другую.

Моторные ошибки

*Проверка действий
пользователя перед
их принятием*

*Всегда показывайте
границы диапазона во
всплывающей
подсказке.*

Этот метод гораздо лучше блокировки, но он тоже не без недостатка: трудно проверять команды.

Наиболее популярны два универсальных и работающих способа проверки.

- ❑ Во-первых, это меню. *В случаях, когда пользователь выбирает команду из списка, система может без труда делать так, чтобы в этот список попадали только корректные команды (это вообще достоинство любого меню).*
- ❑ Во-вторых, если действие запускается непосредственным манипулированием объектами, можно индицировать возможные действия изменением поведения этих объектов.

Например, если бы форматирование диска запускалось не нажатием кнопки, а перенесением пиктограммы диска в область форматирования, можно было бы показывать пользователю, как с выбранного диска исчезают все файлы и папки.

Моторные ошибки

Самостоятельный выбор команд

Система сама должна знать большинство из тех сведений, которые она запрашивает у пользователя. Источниками этих сведений являются:

- ☐ **здоровый смысл разработчика системы**
- ☐ **предыдущие установленные параметры**
- ☐ **наиболее часто устанавливаемые параметры.**

И, наконец, самый эффективный способ. Системе, лучше знать, какие именно команды или параметры для неё пригодны. Соответственно, чем меньше действий требуется совершить пользователю, тем меньше вероятность ошибки (при этом пользователь, которого избавили от рутинной работы, уже радуется).

Однако *цель этого метода* состоит не в том, чтобы провести пользователя за ручку по программе, оберегая его от всего, что может его испугать, но в том, чтобы сделать пользователя более счастливым.

Многим людям будет комфортней работать со сложной системой, нежели со слишком упрощенной.

Проблема этого метода заключается в том, что для его использования к проектированию системы нужно подходить значительно более творчески и тщательно, нежели обычно практикуется.

Два уровня ошибок и обратная связь


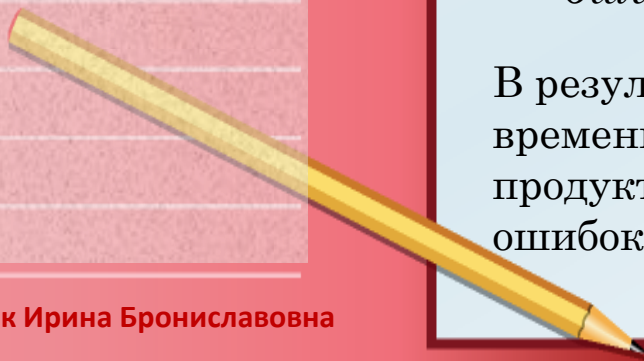
Существует ещё одна классификация ошибок, в которой ошибки расставлены по уровням их негативного эффекта:

- ☐ Ошибки, исправляемые во время совершения действия;
(например, пользователь перетаскивает файл в корзину и во время перетаскивания замечает, что он пытается стереть не тот файл).
- ☐ Ошибки, исправляемые после выполнения действия;
(например, после ошибочного уничтожения файла его копия переносится из корзины).
- ☐ Ошибки, которые исправить можно, но с трудом;
(например реальное стирание файла, при котором никаких его копий не остается).
- ☐ Ошибки, которые на практике невозможно исправить, т.е. ошибки, которые невозможно обнаружить формальной проверкой (т.е. невозможно обнаружить их не случайно).
(например: смысловая ошибка в тексте, удовлетворяющая правилам языка).



Два уровня ошибок и обратная связь

*Почему ошибки
первого типа
(«исправляемые во
время») гораздо
лучше ошибок
второго типа
(«исправляемых
после»)?*

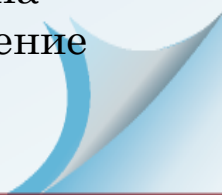


*Объективное объяснение: ошибки,
исправляемые после, снижают
производительность работы.*

Помним, что любое действие пользователя состоит из семи шагов. Всякий раз, когда пользователь обнаруживает, что он совершает ошибку, ему приходится возвращаться назад на несколько этапов. Более того, чтобы исправить совершенную ошибку, от пользователя требуется:

- ☐ *понять, что ошибка совершена,*
- ☐ *понять, как её исправить,*
- ☐ *потратить время на исправление ошибки.*

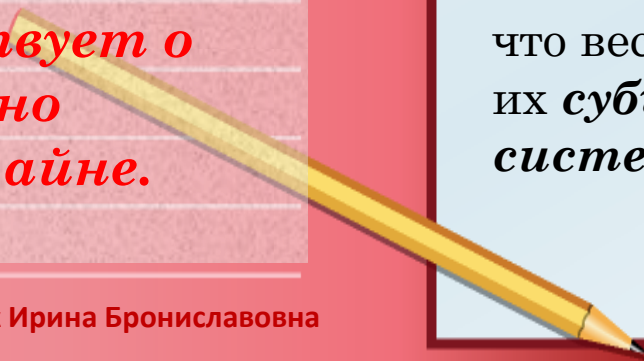
В результате значительный процент времени уходит не на действие (т.е. на продуктивную работу), а на исправление ошибок.





Два уровня ошибок и обратная связь


Наличие человеческих ошибок, которых нельзя обнаружить и исправить до окончательного совершения действия, всегда свидетельствует о недостаточном хорошем дизайне.



Субъективное объяснение:

- ☐ ошибки, исправляемые после, воспринимаются пользователем как ошибки.
- ☐ ошибки же, исправляемые во время, как ошибки не воспринимаются, просто потому, что для пользователей это не ошибки вообще: все человеческие действия до конца не алгоритмизированы, они формируются внешней средой (так не получилось и так не получилось, а вот так получилось).


Ошибка же, не воспринимаемая как таковая, пользователей не раздражает, что весьма положительно действует на их **субъективное удовлетворение от системы.**





Два уровня ошибок и обратная связь

*Как избавиться
от ошибок,
исправляемых
после?*




Чтобы дать пользователям исправлять их действия на ходу, этим пользователям надо дать обратную связь.

К сожалению, вводить в систему обратную связь получается не всегда.

Её ненавидят программисты. Мотивируют они своё отношение тем, что она плохо влияет на производительность системы. Обычно они обманывают. На самом деле им просто лень её реализовывать.

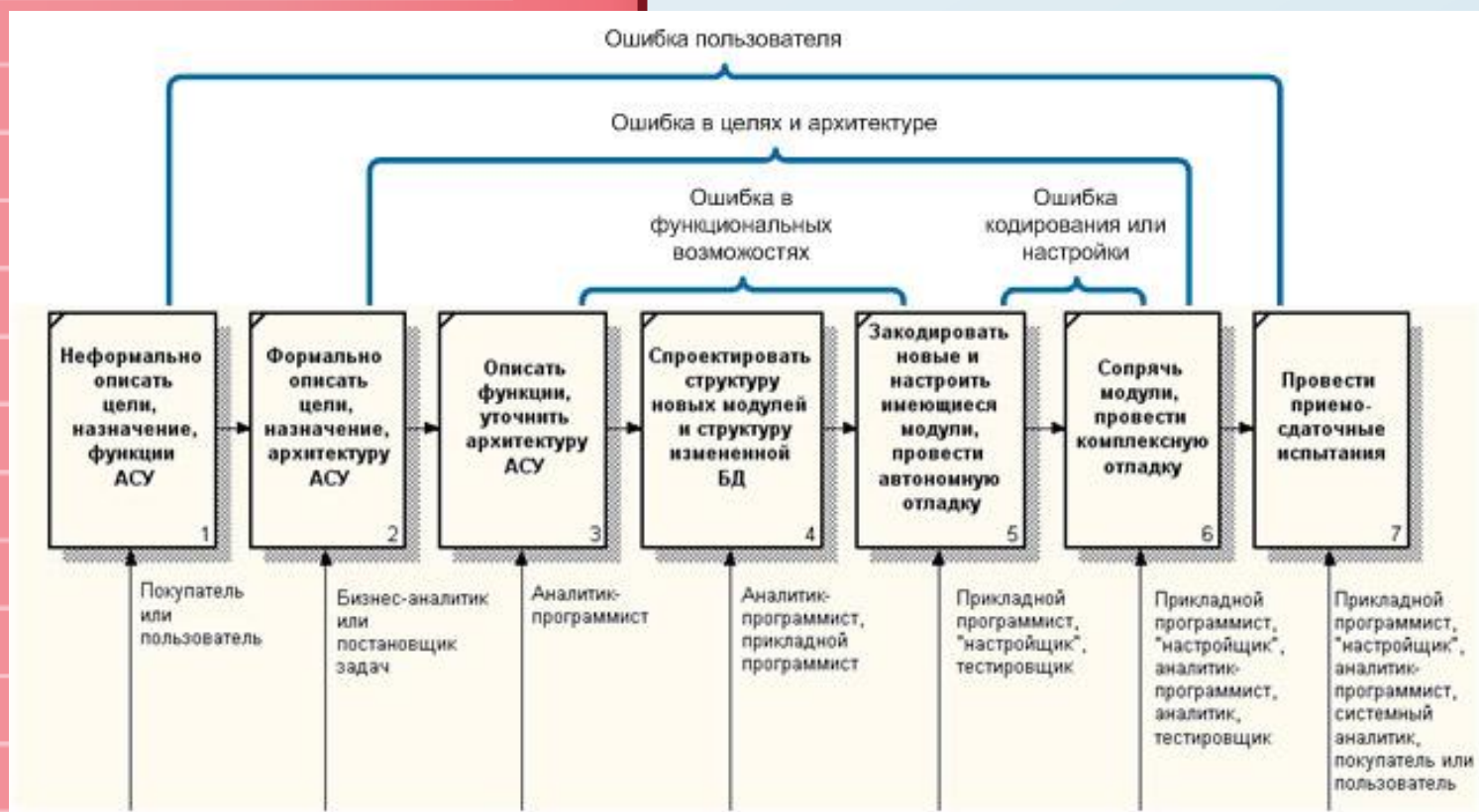
Иногда, впрочем, соображения о производительности системы и вправду имеют место.



Ошибки при создании программного обеспечения

Дольше всех остаётся незамеченной ошибка **пользователя**. Её «мертвое время» – от начала работ и до сдачи – покрывает все внутренние этапы разработки.


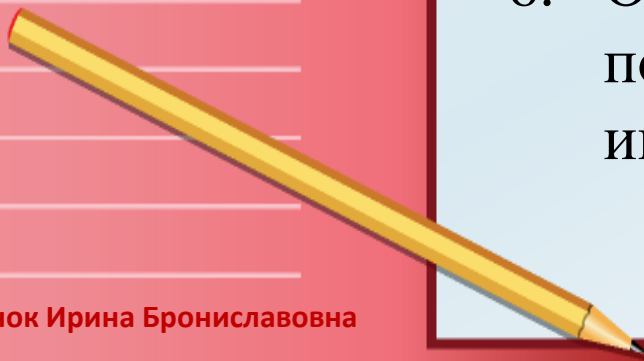
Самое **короткое время жизни у ошибки программиста** или настройщика. Их ошибки обнаруживаются уже на соседнем этапе, во время комплексной отладки.






Глава 4

«Критерии качества интерфейса пользователя»



Тема 12. Скорость обучения


1. Средства обучения пользователей.
 2. Понятность системы.
 3. Построение ментальной модели.
 4. Правила применения метафор.
 5. Способы передачи аффорданса.
 6. Обеспечение мобильности пользовательского интерфейса.
- 



Скорость обучения

Проблема обучения пользователей работе с компьютерной системой чрезвычайно важна.

Рассчитывайте на средних пользователей, а не новичков или на профессионалов.




Поэтому в дополнении к самой системе разрабатывается *методология обучения её будущих пользователей*, а также разрабатываются *нормативы на пользователей*.

Если человек будет сочтен неподходящим, к системе его просто не допустят.

Напротив, с ПО и сайтами ситуация принципиально иная: как цель ставится возможность работы с системой для любого человека, независимо от его свойств и навыков, при этом целенаправленное обучение пользователей, как правило, не производится.

Запомните, что одного стимула для пользователя недостаточно, если он не знает, за что этот стимул дается.



Скорость обучения

Средства обучения

Обычно считается, что в случае ПО есть два способа повысить эффективность обучения (помимо метода «обучения плаванию посредством выбрасывания из лодки»), а именно бумажная документация и «оперативная справка».

Существуют более эффективные способы, ЭТО :

- ☐ общая «понятность» системы
- ☐ обучающие материалы.

Понятность системы

Ментальная модель

Понимание сущности
системы называется
ментальной моделью.

Термин «понятность»
включает в себя три
составляющих:

- ☐ ментальную модель,
- ☐ метафору,
- ☐ аффорданс,
- ☐ стандарт.

(Почти всегда, чтобы успешно пользоваться
какой-либо системой, человеку необходимо
однозначно понимать, как она работает.
При этом необязательно точно понимать
сущность происходящих в системе процессов,
более того, и необязательно правильно их
понимать.)

Понятность системы

Метафора

Чтобы научиться пользоваться системой, пользователю нужно построить ментальную модель этой системы.

Избавить его и от этой работы можно применением **метафоры**, которая позволяет пользователю не создавать новую модель, а воспользоваться готовой моделью, которую он ранее построил по другому поводу.

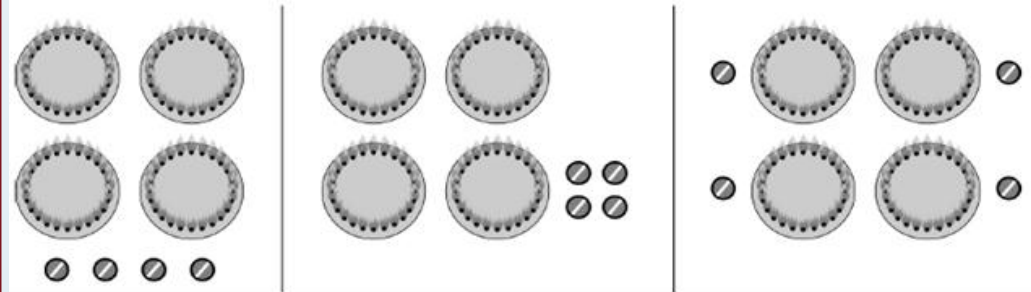
Анализируя опыт применения метафор, можно вывести следующие правила:

- ☐ опасно полностью копировать метафору, достаточно взять из неё самое лучшее;
- ☐ не обязательно брать метафору из реального мира, её смело можно придумать самому;
- ☐ эффективнее всего метафорически объяснять значение отдельных объектов: например, для графической программы слои можно представлять как положенные друг на друга листы стекла (этот пример подходит и для предыдущего пункта);
- ☐ если метафора хоть как-то ограничивает систему, от неё необходимо немедленно отказаться.

Понятность системы

Аффорданс

Аффордансом называется ситуация, при котором объект показывает субъекту способ своего использования своими неотъемлемыми свойствами.



Пример: Слева стандартный вариант, в центре и справа варианты с аффордансом

Например, надпись «На себя» на двери не является аффордансом, а *облик* двери, который подсказывает человеку, что она открывается на себя, несет в себе аффорданс.

Польза аффорданса заключается в том, что он позволяет пользователям обходиться без какого-либо предварительного обучения, благодаря этому аффорданс является самым эффективным и надежным средством обеспечения понятности.

Аффорданс

Способы передачи аффорданса:

- ☐ *маппинг, или повторение конфигурации объектов конфигурацией элементов управления (этот способ работает хорошо в реальном мире, но не очень хорошо на экране, поскольку предпочтительней непосредственное манипулирование);*
- ☐ *видимая принадлежность управляющих элементов объекту;*
- ☐ *визуальное совпадение аффордансов экранных объектов с такими же аффордансами объектов реального мира (кнопка в реальном мире предлагает пользователю нажать на неё, псевдотрехмерная кнопка предлагает нажать на неё по аналогии);*
- ☐ *изменение свойств объекта при подведении к нему курсора (бледный аналог тактильного исследования).*

Понятность системы

Стандарт

*Последовательность в
реализации интерфейса
есть первое условие
качества результата*

Самый мощный, но зато и самый ненадежный способ обучения – **стандарт**.

Дело в том, что если что-либо нельзя сделать «самопроизвольно» понятным, всегда можно сделать это везде одинаково, чтобы пользователи обучались только один раз.

Например, кран с горячей водой всегда маркируют красным цветом, а кран с холодной – синим.

Чтобы стандарт заработал, он должен быть популярен.

Популярность стандарта может быть достигнута двумя способами:

- ☐ во-первых, он может быть во всех системах,
- ☐ во-вторых, он может быть популярен внутри отдельной системы.

(Например, стандарт интерфейса MS Windows популярен почти во всех программах для Windows)

Стандарт

*Стандарты,
обеспечивающие
интерфейсы
пользователей с
операционной средой*

Мобильность информационных систем включает *унификацию* и сохранение *технологии взаимодействия* и всех деталей интерфейса пользователей с приложениями при изменении аппаратных и операционных платформ — ***мобильность пользователей.***

Для сохранения применяемых операционных систем, прикладных программ и баз данных и обеспечения возможности их переноса на иные платформы необходима унификация концепции, архитектуры, функций и методов визуализации пользовательских интерфейсов, организующих их взаимодействие с различными аппаратно-программными реализациями терминалов.

Стандарт

Задача обеспечения мобильности пользовательского интерфейса включает стандартизацию:

1. визуализации и непосредственного взаимодействия пользователей с различными типами терминалов;
2. интерфейсов прикладных программных средств, обеспечивающих визуализацию, с операционной системой;
3. интерфейсов программных средств визуализации с приложениями;
4. интерфейсов прикладных программных средств и баз данных с операционной системой (API).

Стандарт

1. Программные средства визуализации и взаимодействия с пользователями по отношению к операционной системе выступают как прикладные программы и их интерфейсы с ней должны соответствовать стандартам **API POSIX** и в частности стандарту **IEEE 1003.4** для систем реального времени.

Стандартизация визуализации и непосредственного взаимодействия пользователей с различными типами терминалов затруднена широким спектром функционального назначения и особенностей отображаемой информации и классов информационных систем.

2. Наибольшие успехи достигнуты в международной стандартизации архитектуры интерфейсов программных средств визуализации с операционными системами. Применение этих стандартов значительно облегчает переносимость пользовательского интерфейса и приложений на иные платформы. Основная совокупность стандартов поддерживает графические пользовательские интерфейсы (Graphical User Interfaces — **GUI**).

Ключевой проблемой остается выработка единой методологии создания программ графических систем.

Стандарт

Основные особенности современного интерфейса с пользователями :


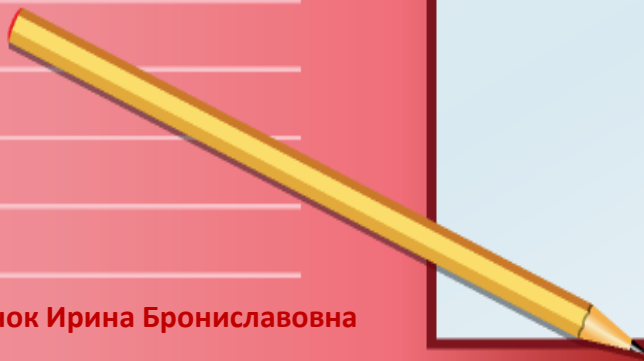
- ☐ наличие механизмов управления окнами;
- ☐ использование готовых графических символов (икон) для отображения управляемых объектов;
- ☐ непосредственное манипулирование графическими объектами и окнами посредством "мыши";
- ☐ объектно- и проблемно-ориентированное проектирование диалоговых систем.

Для реализации интерфейсов создаются и используются библиотеки технологических интерактивных программ, позволяющих использовать устройства ввода команд управления и графических элементов при наличии обратной связи, отображающей на дисплее результаты манипулирования ими.




Глава 4

«Критерии качества интерфейса пользователя»



Тема 13. Субъективное удовлетворение пользователей

1. Использование эстетически привлекательных объектов.
 2. Объективное и субъективное ощущение времени.
 3. Сообщения об ошибках.
 4. Использование паролей.
- 

Субъективное удовлетворение

Натан Мирвольд, бывший вице-президент Microsoft, некогда высказал скандальную по тем временам сентенцию «Крутота есть веская причина потратить деньги» (*Cool is a powerful reason to spend money*). Слово «Cool» — русск. «Крутота», к сожалению, плохо переводится на русский язык, впрочем, засилье американской культуры привело к тому, что все мы неплохо представляем его смысл.

Предположение Мирвольда оправдалось.

Исследования показали, что пользователи воспринимают **одинаково положительно** как *убогие, но приятные интерфейсы*, так и *простые, эффективные, но сухие и скучные*.

Таким образом, субъективные факторы имеют тот же вес, что и объективные.

И то и другое важно.

Эстетика

Все знают, что значительно легче и приятнее пользоваться эстетически привлекательными объектами. Это наблюдение породило весь промышленный дизайн, включая дизайн одежды, интерьеров и так далее.

В то же время в дизайне интерфейсов, это наблюдение до сих пор как следует, не утвердилось: бои между *пуристами* (интерфейс должен быть, прежде всего, работоспособным) и *маньеристами* (красота – это страшная сила) никоим образом не затихают.

В то же время «срединный путь» до сих пор не найден, интерфейсы, равно удобные и эстетически привлекательные, до сих пор существуют в единичных экземплярах.

Кто бы что ни говорил, но массовые представления о прекрасном за последние сто лет не выросли.

Эстетика

Принципы многих направлений дизайна вполне применимы к дизайну интерфейса, при этом донорами преимущественно выступают книжный, коммуникационный и промышленный дизайны.

- ❑ *Внимание к деталям.* Интерфейс состоит из отдельных деталей, каждая из которых действует сравнительно независимо, поскольку раскрывает различную функциональность. Это сближает дизайн интерфейса в целом с книжным дизайном, характерными, как раз пристальным вниманием к мелочам.
- ❑ *Интерфейс не самоценен.* Опять сближение с книжным дизайном (никто не покупает книгу из-за качества её верстки).
- ❑ *Интерфейс передает информацию своему пользователю.* Опять книжный дизайн и коммуникационный дизайн вообще. Фактически, плакат со схемой метро обладает явно выраженным интерфейсом, другой разговор, что этот интерфейс более однонаправленный, нежели двусторонний.
- ❑ *Интерфейс обычно предназначен для длительного использования.* Это серьезно отличает его от графического дизайна вообще (никто не будет рассматривать журнальный разворот часами), но зато сближает опять с книжным дизайном и дизайном среды обитания.

Эстетика

- ❑ *Интерфейс функционален.* Очень часто приходится искать компромисс между эстетикой и функцией. Более того, интерфейс сам по себе зарождается в функциональности, «интерфейс ни к чему» просто не может существовать. Это сближает дизайн интерфейса с промышленным дизайном.
- ❑ *Интерфейс готового продукта образуется не сам по себе, но в результате промышленного производства.* Дизайнер интерфейса не сам производит интерфейс – за него это делают программисты, имеющие свои ограничения (стоимость, технология и так далее).

Эстетика

Конструируемый предмет должен быть незаметен в процессе его использования.

Он должен приятно ощущаться на бессознательном уровне.

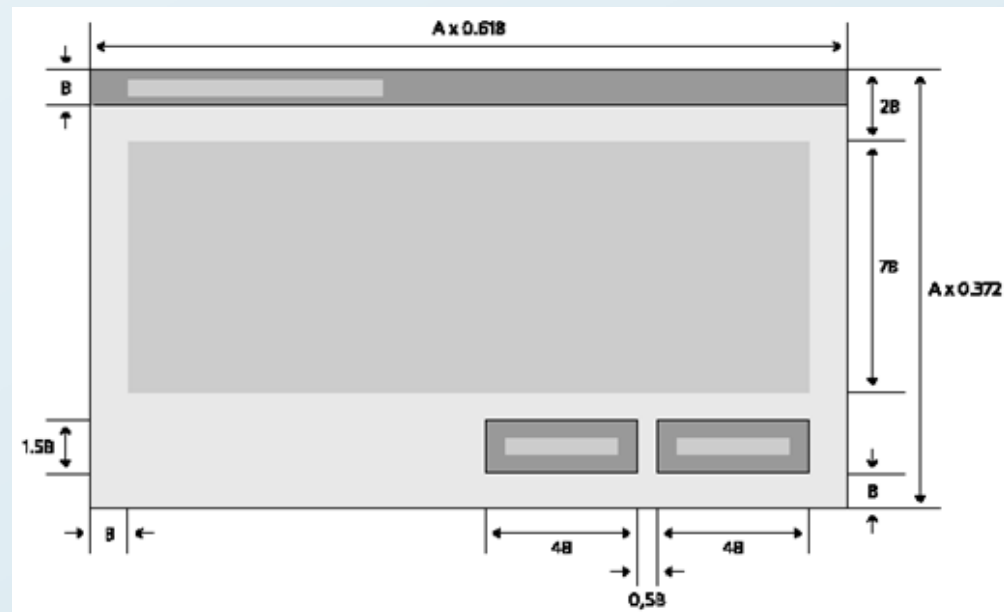
Для этого:

- ❑ **Избегайте развязности в изображении.** Лучше, чтобы он был скромнее. Во что бы то ни стало, добивайтесь того, чтобы интерфейс был неоощуаем.
- ❑ **Избегайте ярких цветов.** Существует очень немного цветов, обладающих и яркостью, и мягкостью (т.е. не бьющих по глазам). На экране их значительно меньше, поскольку в жизни такие цвета обычно моделируются как собственно цветом, так и текстурой, с чем на экране есть проблемы.
- ❑ **Избегайте острых углов в изображении.**
- ❑ **Старайтесь сделать изображение максимально более легким и воздушным.**
- ❑ **Старайтесь добиваться контраста не сменой насыщенности элементов, а расположением пустот.**

Эстетика

Визуальный дизайн: использование компонентов

Пример
закономерностей в
диалоговом окне.



- ☐ *Старайтесь минимизировать количество констант* (тем более, что двух констант обычно хватает на все).
- ☐ *Придерживайтесь во всей системе единожды примененных закономерностей.*
- ☐ *Правильно используйте компоненты визуального дизайна.*

Эстетика

*"предоставленная
возможность"
(affordance)*

Хорошо выполненный дизайн выглядит чистым, простым и аккуратным. Его можно понять одним взглядом.

"Предоставленные возможности" это визуальные характеристики объектов, которые сигнализируют о том, что с ними можно сделать.

Например, поля ввода, приглашают пользователя ввести любое текстовое или числовое значение. Если же набор вводимых величин ограничен, то лучше использовать выпадающий список.

Вы также должны понимать принципы визуальных сообщений. **Размер, цвет, яркость, местоположение, форма и текстура** – все это средства, которые используются для того чтобы сгруппировать элементы вместе по важности или по похожести.

Если у вас есть возможность воспользоваться услугами профессионального визуального дизайнера, не пренебрегайте ею.

Эстетика

Красота

Любая красота со временем надоедает и в лучшем случае перестает восприниматься.

Именно поэтому в интерфейсах обычно не место красоте.

Красота понятие относительное.

(Для одних красивыми могут считаться только живописные закаты, для других картины художника Кустодиева, а для третьих – комбинация вареных сосисок, зеленого горошка и запотевшей бутылки пива.)

Это делает красоту вещь не слишком универсальной.

Элегантность и гармония гораздо лучше.

- ☐ Во-первых, они не надоедают.
- ☐ Во-вторых, они редко осознаются потребителями, обеспечивая неощущаемость.
- ☐ В-третьих – они приносят эстетическое удовольствие независимо от культурного уровня потребителя (так, древнегреческие и слегка менее древние римские здания воспринимаются нами красивыми, несмотря на абсолютную разницу культур и времени).
- ☐ В-четвертых, в производстве они гораздо удобнее красоты, поскольку сравнительно легко ставятся на поток.

Эстетика

Элегантность

Каким образом надо действовать, чтобы добиться элегантности?

Старайтесь сделать интерфейс максимально насыщенным визуальными закономерностями.

Есть универсальное правило — чем больше закономерностей, тем больше гармонии. Даже самые незначительные закономерности всё равно воспринимаются.

Под **закономерностью** понимают любое методически выдерживаемое соответствие свойств у разных объектов. (Например, высота кнопок может быть равна удвоенному значению полей диалогового окна.)


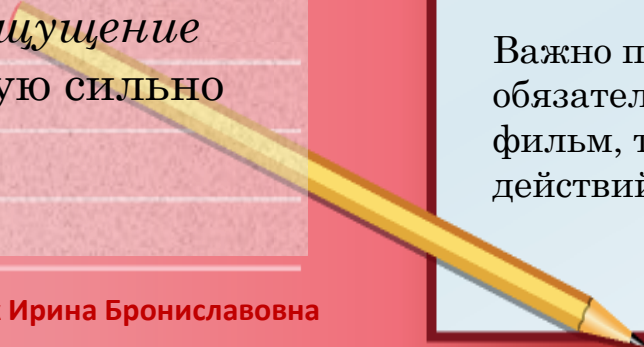
Стремитесь не столько к красоте интерфейса, сколько к его элегантности

- ❑ *Всемерно старайтесь использовать модульные сетки*, т.е. привязывайте все объекты к линиям (лучше узлам) воображаемой сетки, которую выдерживайте во всем интерфейсе.
- ❑ *Старайтесь привязывать все размеры и координаты* (как минимум пропорции диалоговых окон) к золотому сечению (0.618×0.382).



Ощущение времени

Любой человек хочет работать быстро. Если работу (или, понимая шире, любое действие) можно выполнить быстро, у человека возникает приятное ощущение. Хитрость тут в том, что *субъективное ощущение времени* зачастую сильно отличается от *объективного*.



Человеческое восприятие времени устроено своеобразно.

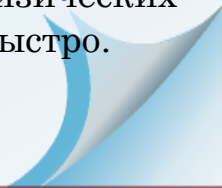
С одной стороны, пользователи способны обнаружить всего 8-процентное изменение длительности в двух или четырехсекундном времени реакции системы.

С другой стороны – не могут точно определить суммарную длительность нескольких последовательных действий.

Более того, воспринимаемая продолжительность действий напрямую зависит от уровня активности пользователя, так что субъективная длительность последовательности действий всегда ниже такой же по времени паузы.

Это наблюдение вовсе не результат напряженных исследований: все знают, что при бездействии (скуке) время течет невыносимо медленно.

Важно понимать, что действие может быть не обязательно физическим: лежа на диване и смотря фильм, т.е. не совершая почти никаких физических действий, время можно потратить очень быстро.



Ощущение времени

Субъективное ощущение времени

Таким образом, субъективную скорость работы можно повысить двумя способами:

❑ **Заполнение пауз между событиями.**

Есть данные о том, что если в периоды ожидания реакции системы пользователям показывается индикатор степени выполнения, субъективная продолжительность паузы существенно снижается. Судя по всему, чем больше информации предъявляется пользователям в паузах, тем меньше субъективное время. С другой стороны, эта информация может вызвать стресс в кратковременной памяти, так что пользоваться этим методом надо осторожно.

❑ **Разделение крупных действий пользователей на более мелкие.**

При этом количество работы увеличивается, но зато субъективная длительность снижается. Плох этот метод тем, что увеличивает усталость.

Ощущение времени

*Объективное
ощущение времени*

С другой стороны, повышение объективной скорости работы зачастую способно повысить и субъективную скорость. В каждом конкретном случае это нужно проверять секундомером, сравнивая объективную длительность действия с его субъективной длительностью.

Психологическое напряжение

Нет ничего более
неприятного, чем
психологическое
напряжение,
иначе говоря – **стресс**.

Большинство
компьютерных программ и
сайтов не требует от
пользователя высокой
степени психологического
напряжения.

Однако, почти всё время пользователь может что-либо испортить и знает это. Он может отформатировать жесткий диск, может стереть или испортить нужный файл. Неудивительно, что пользователь **часто боится**. А если не боится, то склонен **недооценивать свои возможности к разрушению**, отчего снижается бдительность. Куда ни кинь, всюду клин.

Единственным полноценным решением является **возможность отмены пользователем своих предыдущих действий, без ограничения количества уровней отмены и типа отменяемых действий**.

Задача эта непростая, но зато результат крайне существенен.


Реалистичным решением является давно уже существующая практика **прятать опасные для пользователя места интерфейса**. Проблема заключается в том, что при этом логично прятать все функции, изменяющие данные.

Пользователь, зная, что он *не может* совершить ошибку, испытывает радость и умиротворение.



Чувство контроля над системой

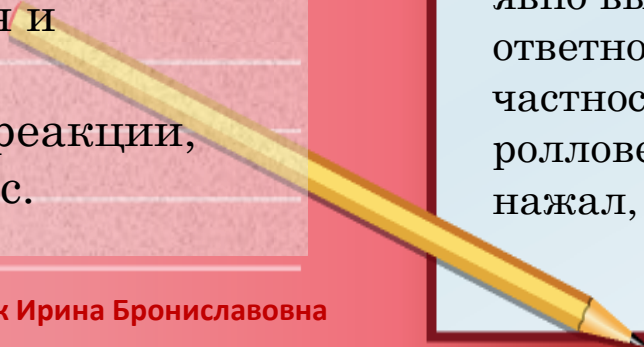
Функции, работающие в автоматическом режиме, но время от времени просыпающиеся и требующие от пользователей реакции, вызывают стресс.



У пользователей для которых использование компьютера не является действием привычным, ощущение того, что они не способны контролировать работу компьютера, является сильнейшим источником стресса. Для остальных пользователей отсутствие чувства контроля не приносит стресса, но всё равно приводит к неудовольствию.

Таким образом, *пользователей нужно всемерно снабжать ощущением, что ничего не может произойти, пока этого не захочется самому пользователю.*

В любом случае, стоит всеми силами внушать пользователям мысль, что только явно выраженное действие приводит к ответному действию системы (это, в частности, главный аргумент против ролловеров – пользователь ещё ничего не нажал, а уже что-то произошло).



Сообщения об ошибках

Ни один пользователь не может долго и продуктивно работать с системой, которая его огорчает и обижает. Тем не менее, такие «скандальные» системы являются нормой. Виной тому **сообщения об ошибках**.

Почему сообщения об ошибках плохи?

Дело в том, что большинство сообщений об ошибках в действительности не являются собственно сообщениями об ошибках.

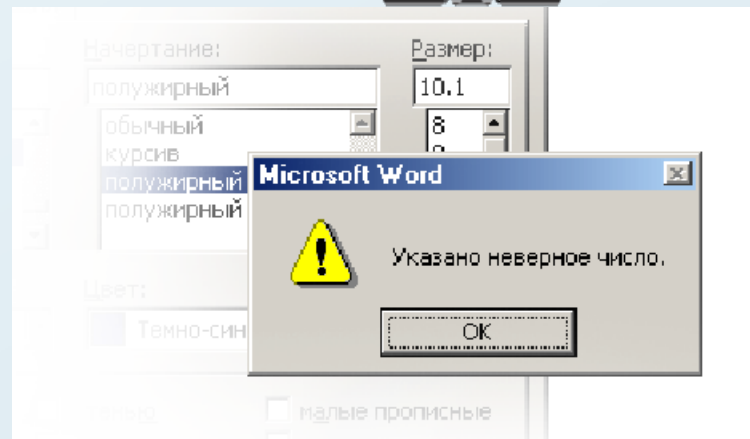
Сообщения об ошибках показывают пользователю, что система, которой он пользуется:

- ☐ недостаточно гибка, чтобы приспособиться к его действиям
- ☐ недостаточно умна, чтобы показать ему его возможные границы его действия
- ☐ самоуверенна и считает, что пользователь дурак, которым можно и нужно помыкать.

Сообщения об ошибках

Недостаточная гибкость

Многие программы искренне «уверены», что пользователь царь и бог, и когда оказывается, что пользователь хочет невозможного (с их точки зрения), они начинают «чувствовать» разочарование. Проявляют же они свое чувство диалогами об ошибках.



Вот что бывает, если пользователь попытается ввести значение, которое ему нужно, но которое система не умеет обрабатывать. Тут возможно три альтернативных решения.

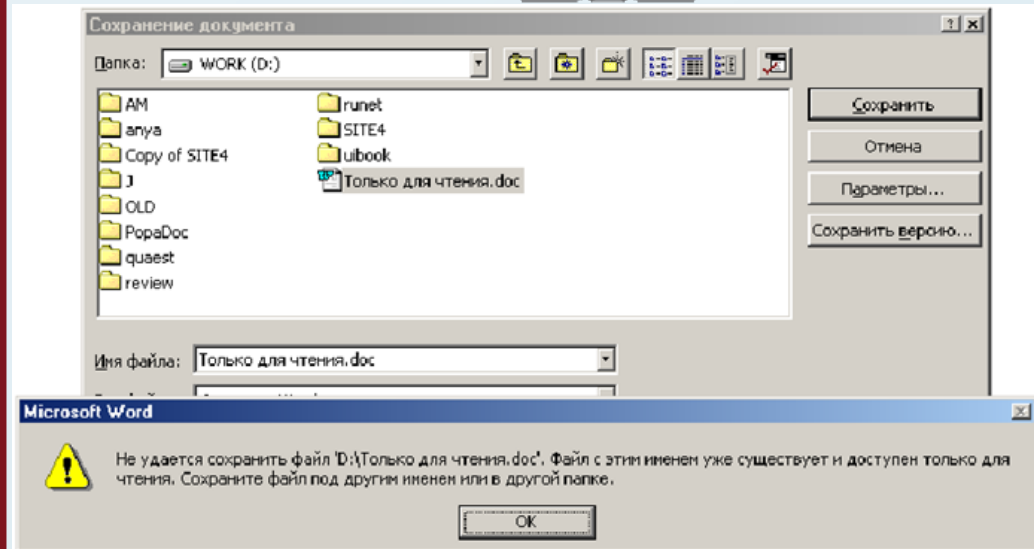
- ☐ Во-первых, при проектировании системы можно более тщательно подходить к выбору её функциональности.
- ☐ Во-вторых, можно просто игнорировать неправильность значения, округляя его до ближайшего возможного (индицируя, возможно, самостоятельность действий системы однократным миганием соответствующего поля ввода).
- ☐ В-третьих, вместо обычного поля ввода можно использовать крутилку.

Пользователю часто нет дела, можно добиться точного результата или нет. Показывать ему в таких случаях диалог об ошибке глупо, поскольку пользователю не нужно ничего знать. Ему нужен результат.

Сообщения об ошибках

Нежелание показать границы действия

Во многих случаях пользователь совершает действия, которые воспринимаются программой как неправильные, не потому, что он дурак, но потому, что система не показала ему границ возможного действия.



Типичное сообщение об ошибке, вызванное нежеланием системы показать пользователю границы его действий. С одной стороны, оно разумно – файл не может быть записан под этим именем.

С другой стороны, это сообщение показывается пользователю каждый раз при попытке перезаписать такой файл.

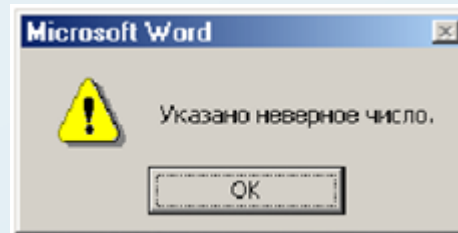
Если бы названия всех защищенных от записи файлов отображались бы не черными, но серыми, это сообщение пришлось бы показывать пользователю только один раз в его жизни. © Microsoft.

Все такие сообщения порочны, поскольку их можно было бы избежать, просто блокировав возможность совершения некорректных действий или показав пользователю их некорректность до совершения действия.

Сообщения об ошибках

Самоуверенность

Нормой также являются случаи, когда система пытается выставить дело так, как будто пользователь идиот, а система, наоборот, есть воплощение безошибочности и правоты.

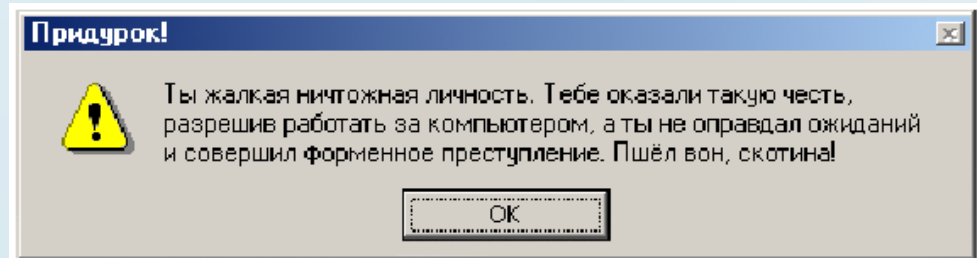


Для кого неверное?
И кто, собственно, виноват, система или пользователь?

В действительности не пользователь сделан для системы, но система для пользователя. Таким образом, как-либо ущемлять пользователя неправильно.

Пользователи ненавидят сообщения об ошибках.

Именно так пользователи воспринимают любые сообщения об ошибках.



Невозможно полноценно работать с системой, которая по несколько раз за день тебя ругает.

Сообщения об ошибках

Каким должно быть сообщение об ошибке?

Идеальное сообщение об ошибке должно отвечать всего на три вопроса:

- ☐ В чем заключается проблема?
- ☐ Как исправить эту проблему сейчас?
- ☐ Как сделать так, чтобы проблема не повторилась?

Отвечать на эти вопросы нужно возможно более вежливым и понятным пользователям языком.

В качестве примера идеального сообщения об ошибке попытаемся улучшить уже упоминавшееся сообщение о невозможности перезаписать заблокированный файл.

Учитывая:

- ☐ Во-первых, никогда не забывайте показывать текст сообщений об ошибке техническому писателю.
- ☐ Во-вторых, всемерно старайтесь делать текст сообщения возможно более коротким.
- ☐ В-третьих, диалоговое окно не самый лучший способ показывать сообщения об ошибках, во всяком случае, в ПО.

Сообщения об ошибках

Файл защищен от записи

Вы пытаетесь сохранить документ в файл, защищенный от записи. Этот файл, вероятнее всего, вы скопировали с компакт-диска. В этом случае нет ничего страшного в перезаписи файла. С другой стороны, этот файл может иметь важное значение для Windows, в этом случае перезапись приведет к нарушению работы операционной системы.

Если вы уверены в себе, нажмите кнопку "Разблокировать файл". В ином случае сохраните файл под другим именем или в другой папке.

Разблокировать файл

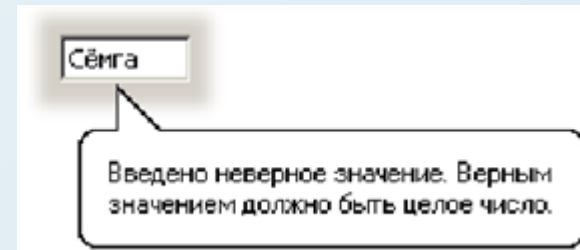
Закреть

Улучшенное сообщение об ошибке. Обратите внимание, что кнопка *Закреть* выбрана по умолчанию, чтобы снизить вероятность перезаписи важных файлов.

Конечно, лучше всего было бы, чтобы ОС сама снимала с копируемых с компакт-диска файлов метку Read Only. Многие проблемы при этом бы исчезли, поскольку защищенными от записи остались только действительно важные для ОС файлы.

Сообщения об ошибках

В Windows появился элемент управления, значительно лучше предназначенный для показа сообщений. Называется этот элемент весьма поэтично:
пузырь



Пузырь, по сравнению с диалоговым окном, имеет существенные достоинства.

- ☐ Во-первых, он гораздо слабее сбивает фокус внимания, нежели окно.
- ☐ Во-вторых, чтобы закрыть пузырь, пользователям не обязательно нажимать на какую-либо кнопку, достаточно щелкнуть мышью в любом месте экрана.
- ☐ В-третьих, он не перекрывает значимую область системы.
- ☐ В-четвертых, что самое главное, он показывает, в каком именно элементе управления была допущена ошибка. Все это делает пузырь вещь совершенно незаменимой.

По статистике через пару лет 80 процентов всех сообщений об ошибках будет появляться в пузырях.

Сообщения об ошибках

пузырь


К сожалению, пузыри имеют и недостатки.

- ❑ Во-первых, в них не может быть никаких элементов управления, так что использовать пузырь в предыдущем примере, например, было невозможно. В Windows пузыри вообще реализованы довольно бедно.
- ❑ Во-вторых, пузырей вообще нет в интернете.

Так что правило тут простое – *используйте пузыри всегда, когда это возможно, и рыдайте, когда их нет.*



Пароли



Не только пароли, но сама по себе идея секретности вызывает у всех пользователей изжогу.

Объясняется это просто – соблюдение секретности требует от пользователей выполнения действий, не приносящих пользы, но уменьшающих *потенциальный* вред.


Человеческая же природа такова, что потенциальное кажется неважным, а вот действия, которые приходится совершать, важны чрезвычайно.

В результате *пароли не любит никто*.



Пароли

Пароли есть явное и несомненное зло, вопрос состоит лишь в том, как это зло уменьшить?



Пароли имеют три принципиальных проблемы.

- ❑ Во-первых, как уже было сказано, пользователи не любят их вводить.
- ❑ Во-вторых, пользователи либо забывают пароли, либо пароли не работают, т.е. ничего не защищают, поскольку пользователи используют те пароли, которые они помнят и которые, соответственно, не проблематично подобрать.
- ❑ В-третьих, в интернете часто происходит так, что пользователи, не желая вводить пароль (и регистрироваться, к слову говоря) так никогда не попадают в главную часть сайта.

Пароли

Стандартный способ
человеколюбивого
использования паролей:
слева во время
регистрации, справа в
дальнейшей работе.

От паролей можно отказаться. Этот путь почти всегда предпочтителен, проблема в том, что он вызывает существенное отторжение у сетевых администраторов (которые все как один параноики). В самом деле, зачем требовать от пользователя пароль, если подобрать этот пароль злоумышленник сможет без труда?

Имя пользователя

Пароль

Повторите пароль

Имя пользователя

Пароль

☒ Запомнить пароль

Стандартный способ человеколюбивого использования паролей: слева во время регистрации, справа в дальнейшей работе.

Эффективнее всего максимально **здорово** *определить степень важности защищаемой информации, после чего выбрать адекватную схему защиты, стараясь, чтобы она была максимально лёгкой для пользователей.*

Как правило, их субъективное удовлетворение важнее потенциального вреда от потери информации.

Вывод

Пользователи хотят выразить себя и в программах, которыми они пользуются.

Соответственно, возможность настроить систему под свои нужды является мощной причиной *субъективного удовлетворения*.

Проблема здесь в том, что это очень дорого стоит. Времени на самовыражение уходит крайне много.

- ❑ Во-первых, пользователи не склонны удовлетворяться первым получившимся вариантом (творчество процесс итерационный).
- ❑ Во-вторых, когда выходят новые версии системы (или имеющаяся версия разрушается от излишнего самовыражения), все приходится начинать сначала. (по грубым прикидкам, на такой процесс самовыражения в среднем уходит около 45 минут в неделю, получается, что организация всего с сотней работников теряет на этом еженедельно 75 часов, что почти равняется потере недельной работы двух человек).

С другой стороны, настроенный под свои нужды интерфейс, судя по всему, снижает усталость работников и повышает их рабочее настроение.

В таких условиях потеря 45 минут в неделю может оказаться скомпенсированной благодаря повышению производительности труда.

Таким образом, ***в продуктах, продаваемых пользователям напрямую, возможность настройки под конкретного пользователя обязательна.***