Технологии разработки программного обеспечения

Составитель: Эверстов В.В.

Дата составления: 15/03/2016

Дата модификации: 15/03/2016

Что такое Функциональная спецификация?

- Функциональная спецификация описывает как все должно работать с точки зрения пользователя.
- Техническая спецификация описывает внутреннюю реализацию системы. Она описывает структуры данных, алгоритмы, структуру базы данных, выбор языка программирования, инструментов и т. д.

Что такое Функциональная спецификация?

- Если техническое задание затрагивает все, что касается отношений между заказчиком и разработчиком, то спецификации – уже **внутренний** документ. Внутренний, для разработки.

Что такое Функциональная спецификация?

- Во-первых, нужно разработать **функциональную спецификацию**. Основная ее задача – ответить на вопрос «**Что надо сделать**». При ее написании разрабатываемая система представляется черным ящиком. Т.е. что и как работает у нее внутри, нам сейчас не интересно, а интересно только то, как ее функционирование будет выглядеть снаружи.

Что такое Функциональная спецификация?

- Если описываете продукт, с которым будут работать конечные пользователи, сюда входит то, что называется «Пользовательским интерфейсом» – набор окон, элементов ввода-вывода информации и описание как можно большего количества ситуаций работы с системой.
- Если продукт является компонентом, который будет использоваться в более объемной системе, то в это описание должны входить все интерфейсные функции этого модуля.

Для чего она нужна?

- Когда вы проектируете ваш программный продукт на человеческом языке, вам нужно меньше времени, чтобы попытаться вдуматься в альтернативные пути, исправить ошибки и улучшить разработку. Менее болезненно, когда удаляет абзац в тексте, чем если вы реализуете без проектирования, итерационная разработка займет *недели*.

Для чего она нужна?

- Спецификации *сокращают информационные потоки*. Когда вы пишете спецификацию, вы должны *только один раз* выяснить, как программа работает. Каждый член команды может потом просто почитать спецификацию.

Для чего она нужна?

- Без детальной спецификации невозможно составить план. Почти в любой отрасли реального бизнеса вы просто обязаны знать, как долго все будет продолжаться, потому что разработка продукта стоит ***денег***.

Для чего она нужна?

- Распространена следующая ошибка: разработчики долго обсуждают, каким образом что-то должно быть реализовано, но после этого *никогда не принимают решение*.
- Написание спецификации – прекрасный способ точно выразить все эти раздражающие вопросы разработки. Даже маленькие решения могут быть выражены в спецификации.

Что должно быть в функциональной спецификации?

- Название,
- Автор,
- Общая информация,
- Сценарии,
- Детали,
- Проблемы,
- Заметки.

Что должно быть в функциональной спецификации?

- Автор
  - Спецификация должна иметь автора. Не стоит разрабатывать спецификацию группой. Во-первых, это займет много времени. Во вторых, если обнаружиться ошибка, то из группы сложно будет найти ответственного за ошибочное решение. Если у вас слишком большой проект, следует разбить его на несколько частей по областям и раздать разным исполнителям.

Что должно быть в функциональной спецификации?

- Общая информация
  - Здесь вы должны привести общее описание системы, блок-схему, архитектуру системы, прочитав, которую читатели должны составить себе общую картину того, что они делают.
  - Рамки системы, т.е. вы должны описать чего не будет делать ваша система.
  - Пригласить, всех к обсуждению или к исправлению ошибок.

Что должно быть в функциональной спецификации?

- Сценарии
  - При проектировании программных продуктов у вас всегда должны быть готовы реальные сценарии того, как люди будут пользоваться вашим продуктом. К этому моменту вы уже должны определиться с целевой аудиторией. И попытаться выбрать наиболее типичные случаи.

Что должно быть в функциональной спецификации?

- Детали
  - Детали наиболее важная часть функциональной спецификации. В большинстве случаев программисты решают, что некоторую детализацию они сделают тогда, когда это понадобиться, а не сейчас. Наипростейший способ построить детализацию это описать каждое окно по отдельности. Можно каждому окну давать свои имена. Очень важно здесь рассмотреть все всевозможные ошибочные ситуации, которые могут возникнуть в вашем продукте. Необходимо, принять политику действий при ошибочных ситуациях. И она должна быть описана в функциональной спецификации.

Что должно быть в функциональной спецификации?

- Проблемы
  - Вполне приемлемо, если первая версия вашей функциональной спецификации, будет содержать открытые проблемы, т.е. проблемы, которые требуют решения в будущем. Надо в спецификации каким-либо образом их помечать и решить их до того как программисты начнут этап реализации.

Кто должен писать функциональную спецификацию?

- Если вы когда-либо заботились почему программисты более заботятся о элегантности кода, чем о привлекательности продукта вам необходим менеджер проектов. Если когда-либо задумывались почему люди лучше пишут на C++, чем по-русски или по-английски, то вам необходим менеджер проектов.

Кто должен писать функциональную спецификацию?

- Менеджеры проектов это отдельный класс специалистов. Они должны быть технически очень подкованы, но не обязаны быть хорошими кодерами. Менеджеры проектов должны изучать как строить GUI, встречаться и контактировать с большим количеством разных людей и уметь писать функциональные спецификации.

Обновления функциональной спецификации

- Функциональная спецификация всегда должна оставаться актуальной. Она должна обновляться и во время реализации проекта если были приняты новые проектные решения.

Техническая спецификация

- Техническая спецификация описывает внутреннюю реализацию системы. Она описывает структуры данных, алгоритмы, структуру базы данных, выбор языка программирования, инструментов и т.д.

План работы

- После того, как вы разберетесь со спецификациями, вы будете готовы к составлению плана ваших действий.
- Во многих компаниях-разработчиках игр на пресс-релизах на их сайтах вы можете прочитать, что следующая версия игры выйдет такого-то числа. Именно благодаря плану и рассчитывается примерная дата выпуска продукта.

Для чего нужен план?

- План позволяет не только распланировать действия, которые вы должны совершить, для достижения конечной цели, но и позволяет
  - примерно рассчитать дату выпуска продукта,
  - Распределить задачи по исполнителям,
  - Отслеживать выполняемость всего проекта.

Что должно быть в плане?

- Минимально план должен включать:
  - Задачи,
  - Исполнителя,
  - Длительность реализации задачи,
  - Приоритет.
- Так же в нее можно включать:
  - Начальная оценка выполняемости,
  - Текущая оценка,
  - Прошло,
  - осталось

Кто его должен составлять?

- **Только программист, который будет писать данный код, может распланировать его.** Любая система, где начальство пишет план и вручает его программистам, обречена на неудачу. Только программист, который будет делать данную работу, способен определить, какие шаги нужно предпринять, чтобы реализовать данное свойство. И только программист сможет оценивать, сколько времени займёт каждый шаг.

Пример

Свойство	Задача	Приор	Нач Оцн	Тек Оцн	Прошло	Осталось
Проверка орфографии	Добавить пункт меню	1	12	8	8	0
Проверка орфографии	Главный диалог	1	8	12	8	4
Проверка орфографии	Словарь	2	4	4	4	0
Проверка грамматики	Добавить пункт меню	1	16	16	0	16

Подсказки

- Каждое свойство должно состоять из нескольких задач.
- Ставьте очень мелко дроблёные задачи.
- Добавьте строки для Праздников, Отпуска, и т. д.
- Поместите в план время отладки!
- Добавьте в план время на интеграцию.
- Добавьте буфер в план.
- Время выполнения задачи, которое вы запланировали, лучше умножить на 3, тогда вы получите более реальные сроки выполнения.

Анализ плана

- Если вы составили план, но оказывается, что не успеваете в сроки, тогда надо сделать одно из двух:
  - Передвинуть сроки релиза,
  - Убрать некоторые из свойств, передвинуть их на следующие версии.
- Отслеживайте начальную и текущую оценку.

GUI

Пользователь

- Придумайте самого типичного пользователя, который будет работать с вашей программой.

Счастье

- Психологическая теория доктора Мартина Е. П. Селигмана (Dr. Martin E.P. Seligman) под названием "Приобретенная беспомощность" (Learned Helplessness).
- Состояние депрессии часто вырастает из чувства *беспомощности,* когда человек ощущает, что не может контролировать происходящее.

Центральная аксиома

- **Хороший дизайн пользовательского интерфейса подразумевает, что программа соответствует ожиданиям пользователей о том, как она должна себя вести.**

Центральная аксиома

- Суть пользовательского интерфейса в следующем: реагирует ли пользовательский интерфейс так, как пользователь того *ожидает*? Если нет, пользователь будет ощущать собственную беспомощность и невозможность контролировать ситуацию,

Модель пользователя

- Когда новый пользователь приступает к работе с новой программой, его голова наполнена опытом прошлых встреч с компьютером. У него есть определенные ожидания по поводу того, как новая программа будет работать. У него могут быть совершенно разумные мысли о том, как будет работать интерфейс данной программы. Все это называется *моделью пользователя*: представление о том, для чего и как программа будет работать.

Модель программы

- Программа тоже обладает моделью поведения, которая, в отличие от модели пользователя, закодирована в биты и самым последовательным образом выполняется CPU – *модель программы*, и она есть -- **Закон**. Если модель программы соответствует модели пользователя, у нас получился удачный пользовательский интерфейс.

Как узнать ожидания пользователей?

- **Спросите их!**
- Выберите наугад 5 человек с работы, или друзей, или родственников. Далее, задайте пару вопросов и попробуйте составить мнение об их модели пользователя.

Как узнать ожидания пользователей?

- Следующий этап - проверка ваших предположений. Постройте модель или прототип вашего интерфейса и дайте нескольким людям задания. Попросите их комментировать свои действия в то время, как они решают поставленную задачу. Ваша цель заключается в том, чтобы понять, чего они ожидают.

Сколько людей привлечь?

- Специалисты советуют ограничить число пользователей до **пяти - шести**. Если вы привлечете большее количество народу, то увидите повторяемость результатов.

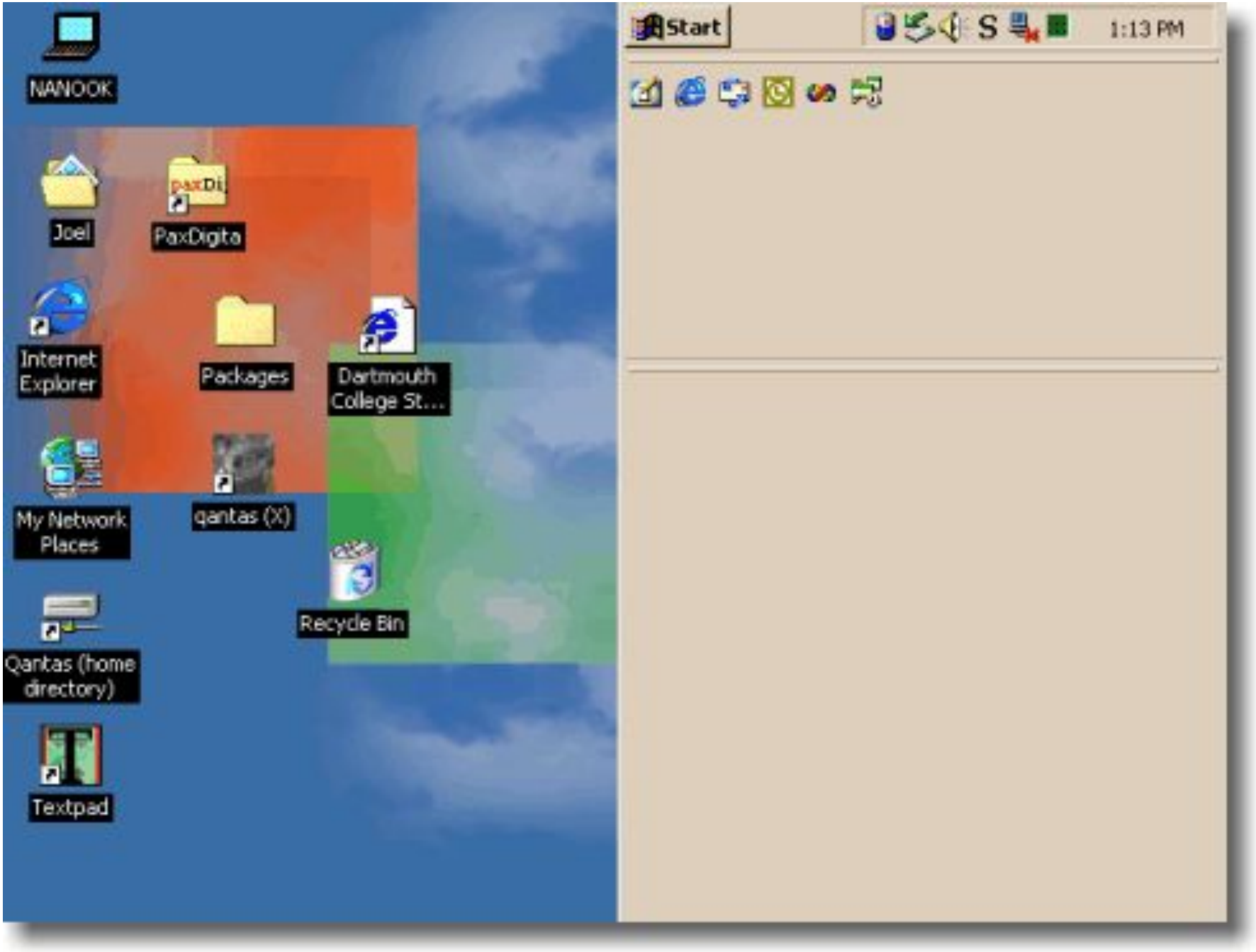
Как приспособить модель?

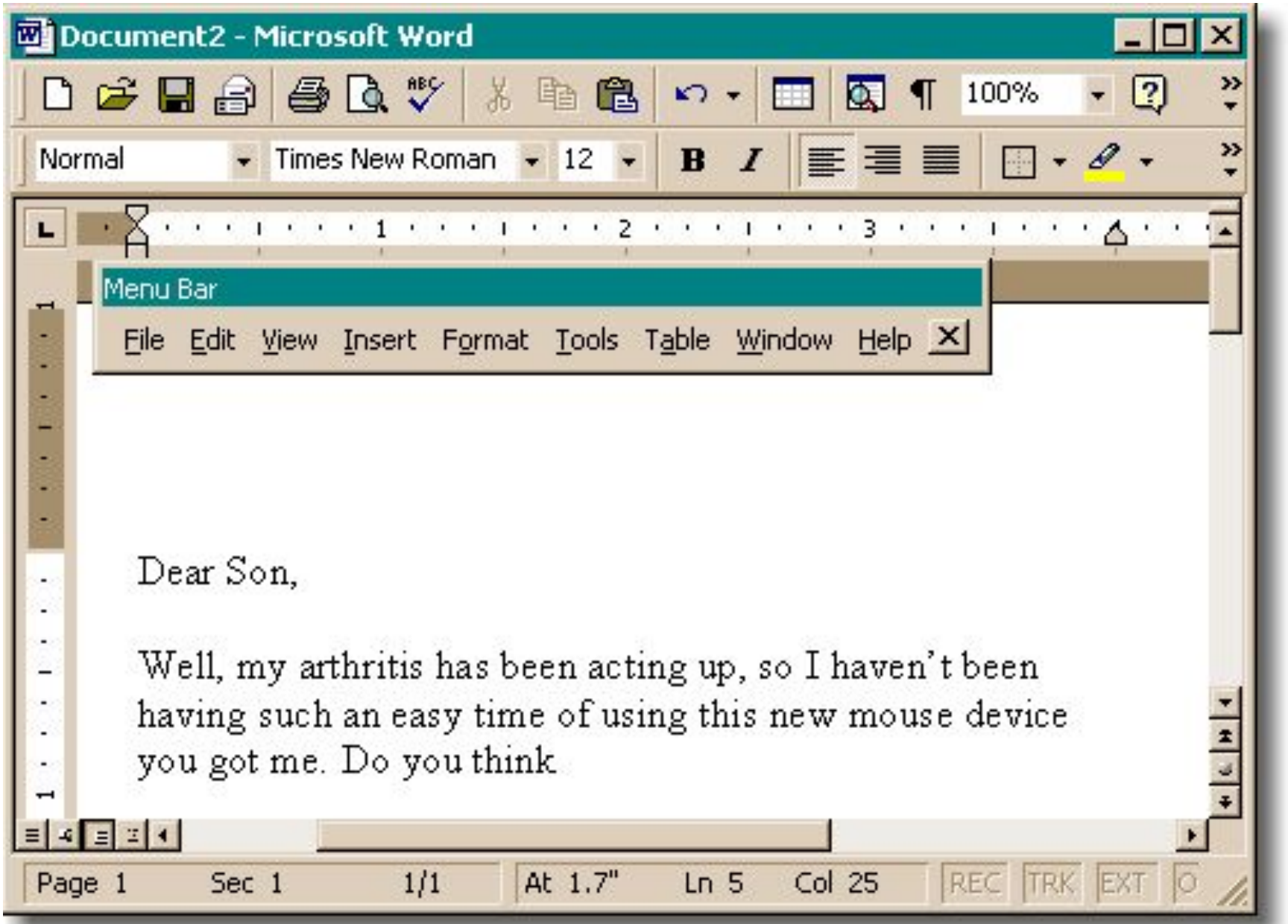
- Приспособить модель программы к модели пользователя - нелегкое дело, даже когда модели простые. Оно становится практически невыполнимым, когда сложность моделей возрастает. Поэтому идеальный вариант - использовать простейшую модель из всех, которые кажутся вам возможными.

Вторая аксиома

- **Каждый раз, предлагая опцию, вы просите пользователя сделать выбор.**

- Ваши программы пользователи используют для решения *своих* задач. И все, что их волнует, -- это решение их задач.


NANOOK
Joel
PaxDigita
Internet Explorer
Packages
Dartmouth College St...
My Network Places
qantas (X)
Recycle Bin
Qantas (home directory)
Textpad
Start
1:13 PM


Document2 - Microsoft Word
100%
Normal
Times New Roman
12
Menu Bar
File Edit View Insert Format Tools Table Window Help
Dear Son,
Well, my arthritis has been acting up, so I haven't been having such an easy time of using this new mouse device you got me. Do you think
Page 1 Sec 1 1/1 At 1.7" Ln 5 Col 25 REC TRK EXT

Вторая аксиома

- В общем, вам следует всегда минимизировать количество вопросов, по которым пользователь должен принять решение.

Метафоры

- Иногда у пользователей *просто нет* конкретного представления о том, как работает программа и для чего она предназначена. В таком случае вам придется найти способ подсказать им, как функционирует ваша программа. В графических интерфейсах используется метод *метафор*.

Увеличительное стекло -- полновесная метафора из реального мира. Вы не опасаетесь, что приближение изменит размер картинки, потому что знаете, что увеличительное стекло сделать этого не может. **Метафора, пусть даже и не самая удачная, лучше чем ее отсутствие.**

Приглашения

- Удачно разработанный дизайн хорош именно тем, что он позволяет сразу понять, как объект функционирует.

Приглашения

- На некоторых дверях на уровне руки расположены большие металлические пластины. Единственное, что вы можете проделать с ними -- толкнуть их.
- Как сказал Доналд Норман, они *приглашают* вас толкнуть их. На других дверях вы увидите большие закругленные ручки, которые просто вынуждают вас *потянуть* за них. Они заранее предполагают, как вам взяться за ручку, чтобы удобно открыть дверь. Ручка *приглашает* вас потянуть дверь на себя.

Приглашения

- Многие приложения в нижнем правом углу имеют выпуклый квадратик с тремя диагональными линиями. Он *приглашает* вас потянуть за него.

Постоянство дизайна

- То, что это делает *Microsoft*, не значит, что это правильно», они создают пользовательские интерфейсы, неоправданно отличающиеся от того, к которому люди привыкли.

Постоянство дизайна

- Так ли это на самом деле?
  - Пусть это и неправильно, но если *Microsoft* использует это в таких известных приложениях как *Word*, *Excel*, или *Internet Explorer*, миллионы людей будут думать, что это правильно, или, по крайней мере, является стандартом.
  - *Microsoft* тратит больше денег на тестирование *usability* чем вы. Они ведут детальную статистику данных, полученных на основе анализа миллионов звонков в службу технической поддержки.

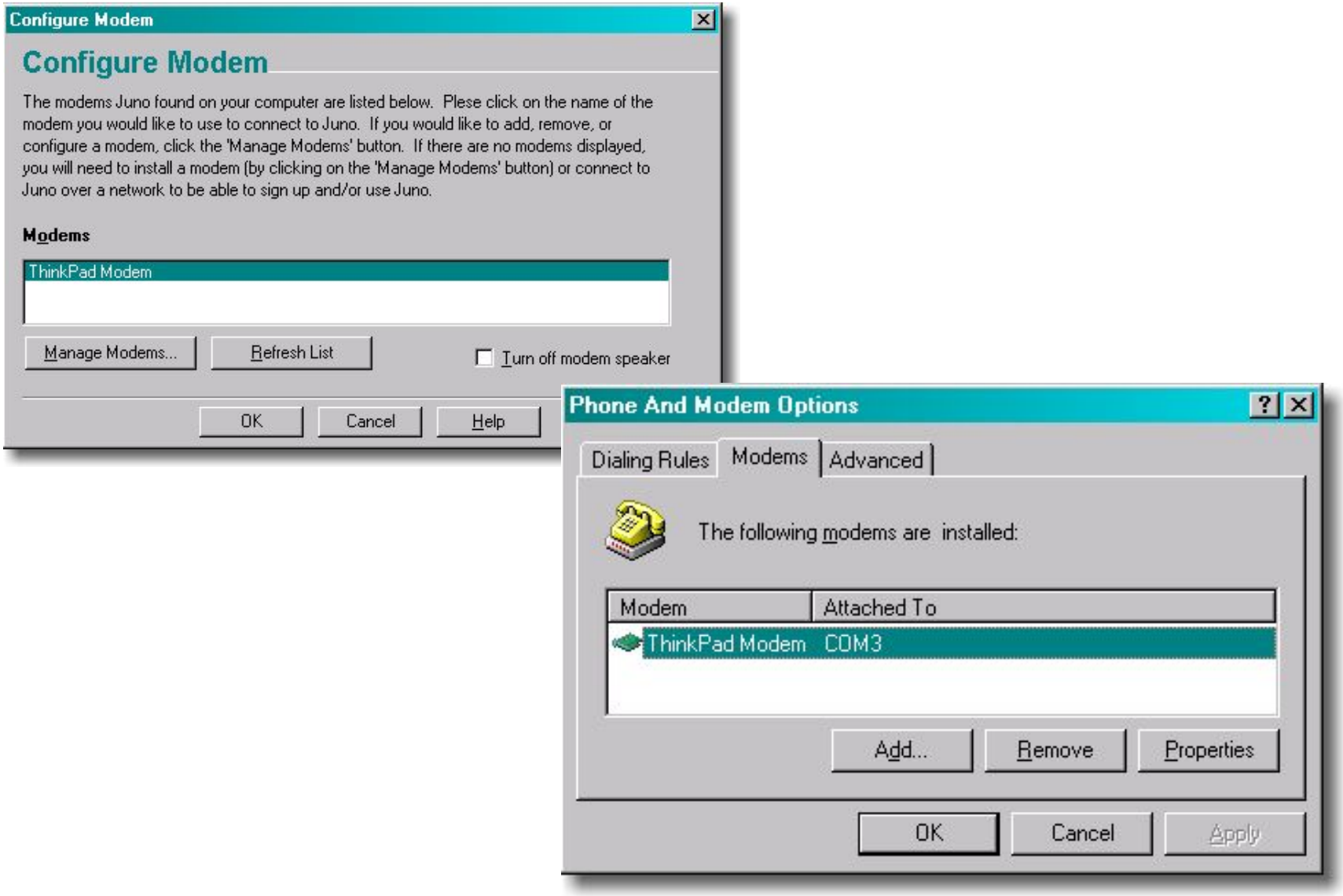
Примеры подражания

- Microsoft
- Amazon.com
- Adobe Photoshop

Чего не делают пользователи?

- Не читают руководств,
- Не умеют мастерски манипулировать мышью,
- Не помнят ничего.

Не читают

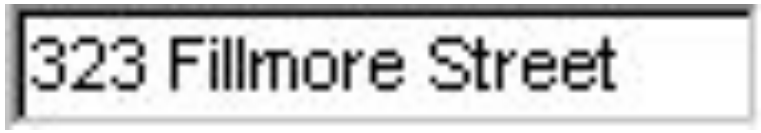

Configure Modem
Configure Modem
The modems Juno found on your computer are listed below. Plese click on the name of the modem you would like to use to connect to Juno. If you would like to add, remove, or configure a modem, click the 'Manage Modems' button. If there are no modems displayed, you will need to install a modem (by clicking on the 'Manage Modems' button) or connect to Juno over a network to be able to sign up and/or use Juno.
Modems
ThinkPad Modem
Manage Modems...
Refresh List
Turn off modem speaker
OK
Cancel
Help
Phone And Modem Options
Dialing Rules
Modems
Advanced
The following modems are installed:
Modem
Attached To
ThinkPad Modem COM3
Add...
Remove
Properties
OK
Cancel
Apply

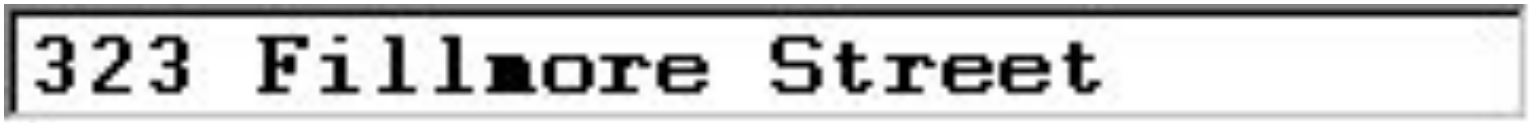
Не дружат с мышью

- Иногда приходится пользоваться не самыми оптимальными манипуляторами -- трэкболами, трэкпадами, и т.д.
- Иногда приходится пользоваться мышью не в самых благоприятных условиях.
- Иногда человек, сидящий за компьютером – новичок.
- Некоторые люди никогда не смогут развить эти самые навыки, по разным причинам.
- Некоторые люди считают, что постоянное применение мыши замедляет работу.

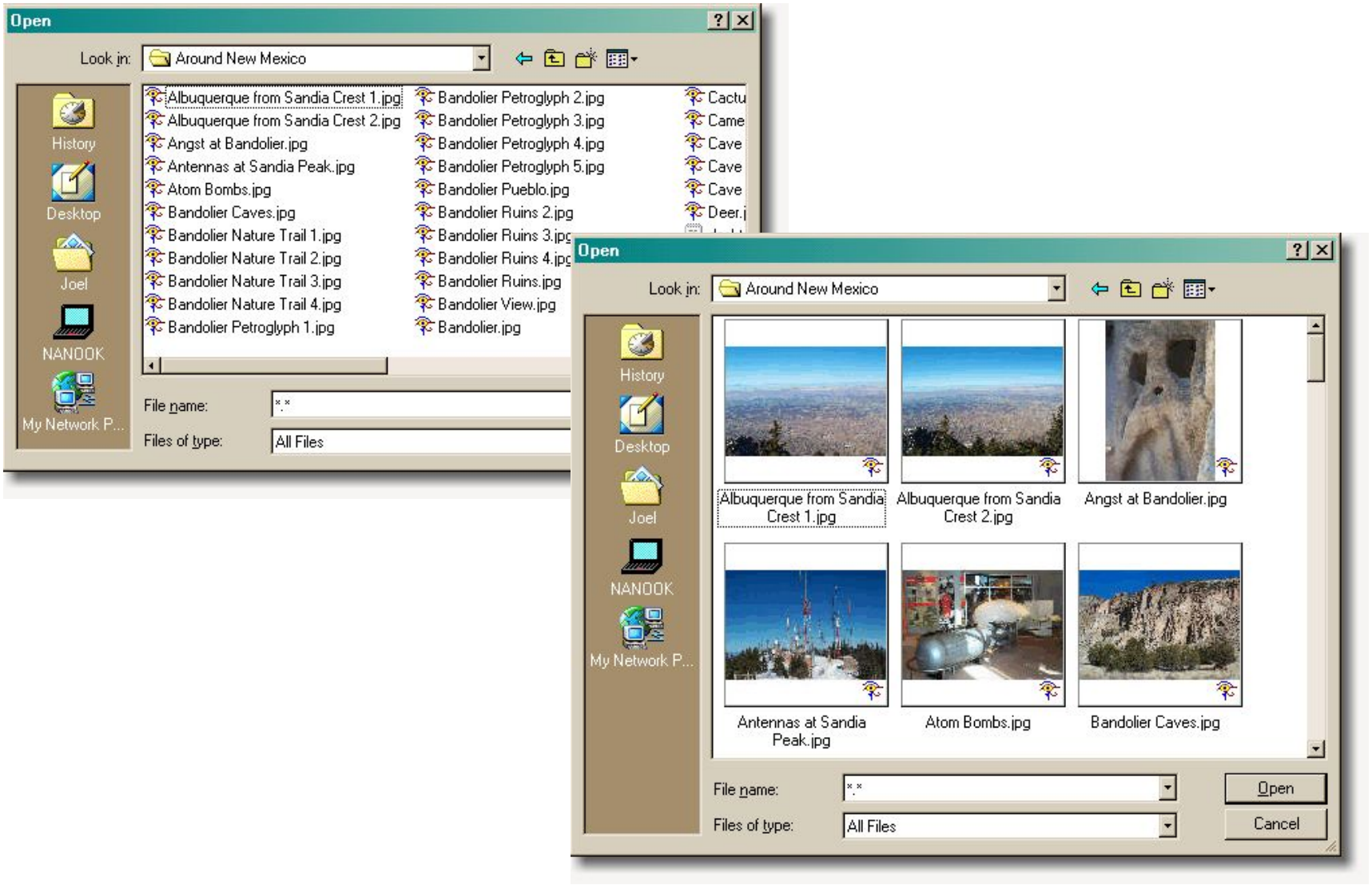
Выпадающие списки

Поля ввода

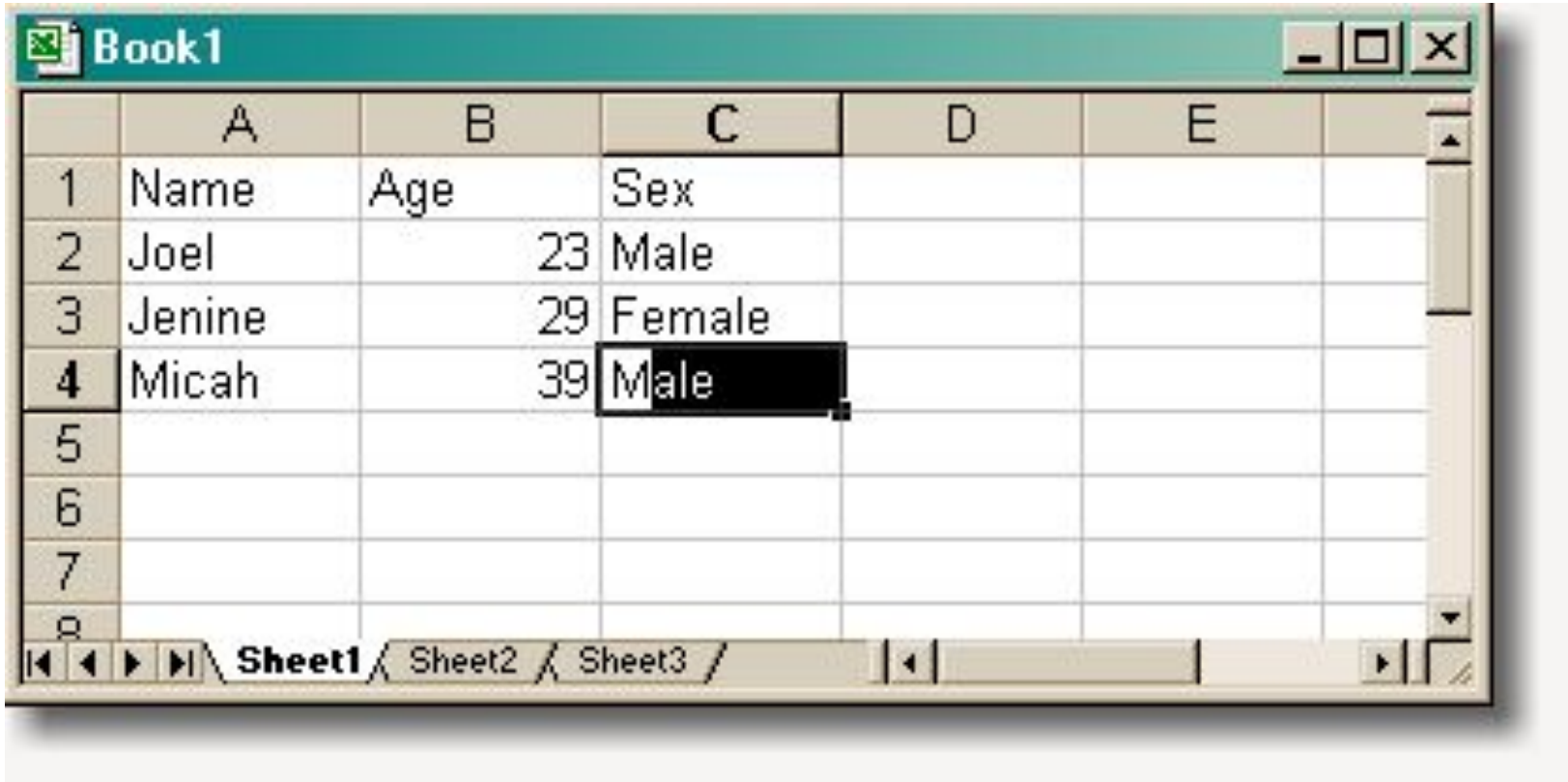




Не помнят ничего



Не помнят ничего



Пользователи – идиоты?

- Если вы постараетесь разработать программу, которой могут пользоваться даже идиоты, вы создадите удобную для пользователя программу, которая понравится людям и станет популярной. И вы удивитесь тому, что такие, казалось бы, незначительные улучшения привели к вам такое огромное количество новых покупателей.

Как разработать хороший GUI?

- Планирование Деятельности.
  - По методу Планирования Деятельности, вы сначала описываете те виды деятельности, которые могут заинтересовать пользователя. Достаточно побеседовать с потенциальными пользователями, и вы сможете составить такой список.
  - Теперь, вместо того, чтобы рассуждать о программе как программист («какие функции нужны для того, чтобы сделать открытку»), вы думаете как пользователь: какие действия он совершает,

ИТОГ

- Придумать воображаемых пользователей
- Продумать виды деятельности пользователей
- Узнать модель пользователя -- как он будет выполнять деятельность, основываясь на своем опыте
- Сделать первый набросок дизайна
- Изменять дизайн, все больше и больше делая его простым в использовании, до тех пор, пока продукт не окажется в рамках способностей воображаемых пользователей