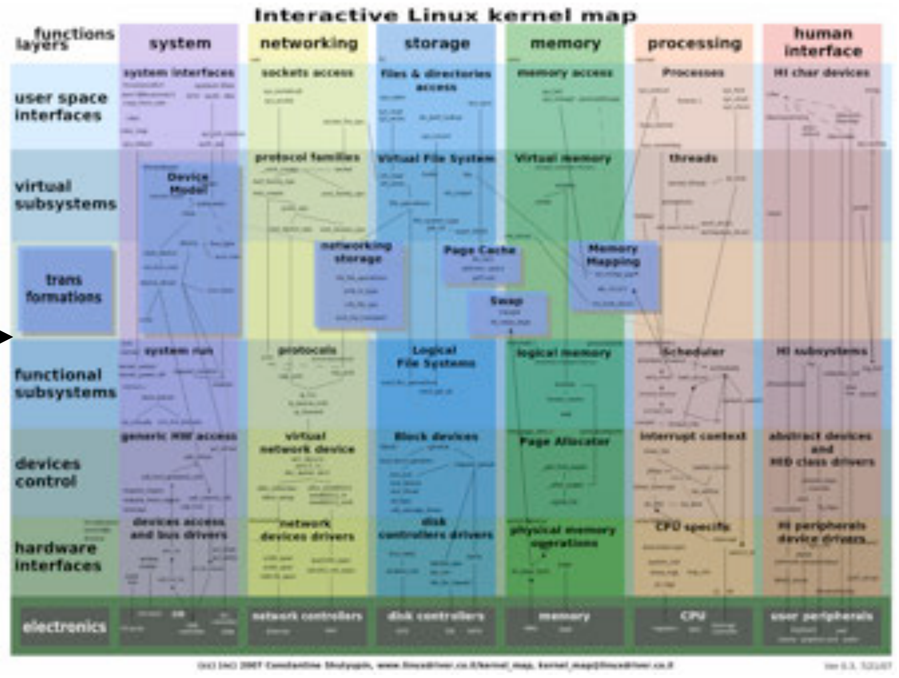
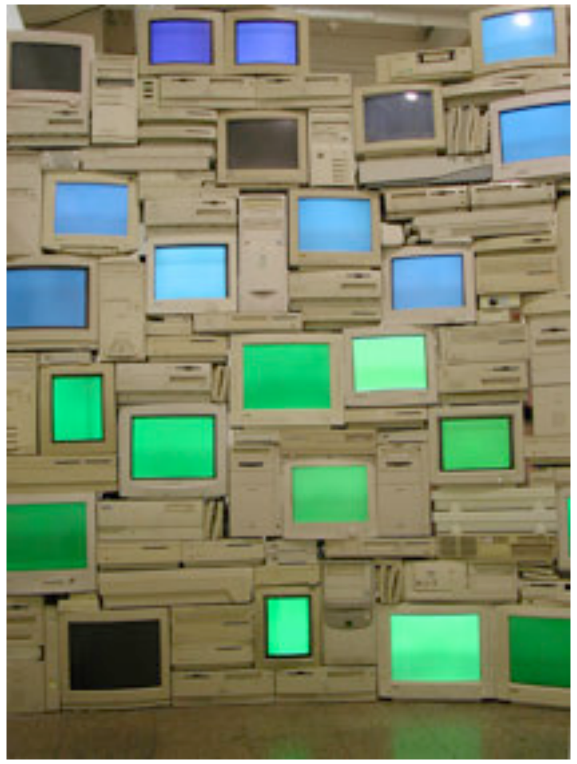
Software Development
The Big Picture

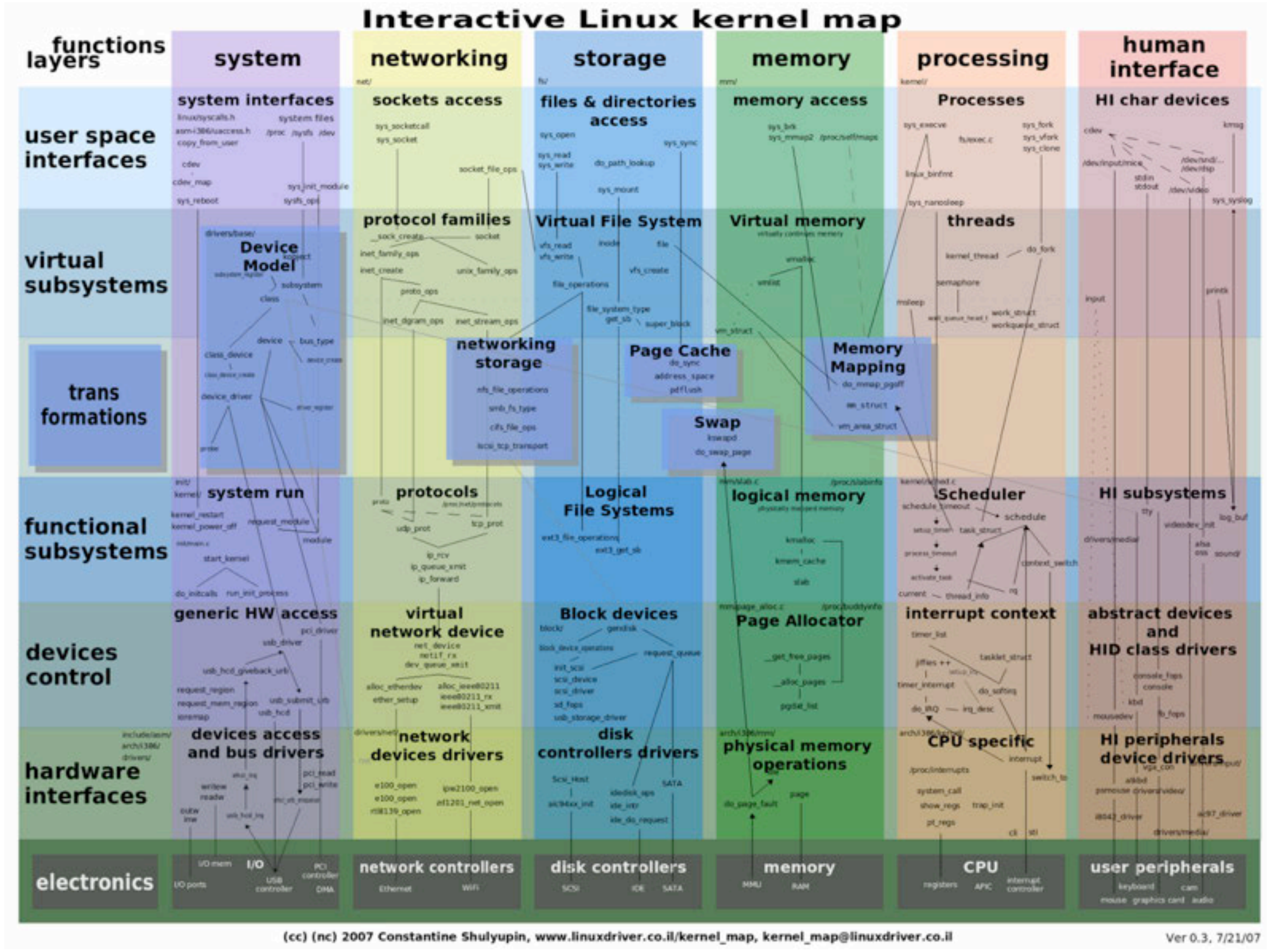
- Software Development is more than programming.





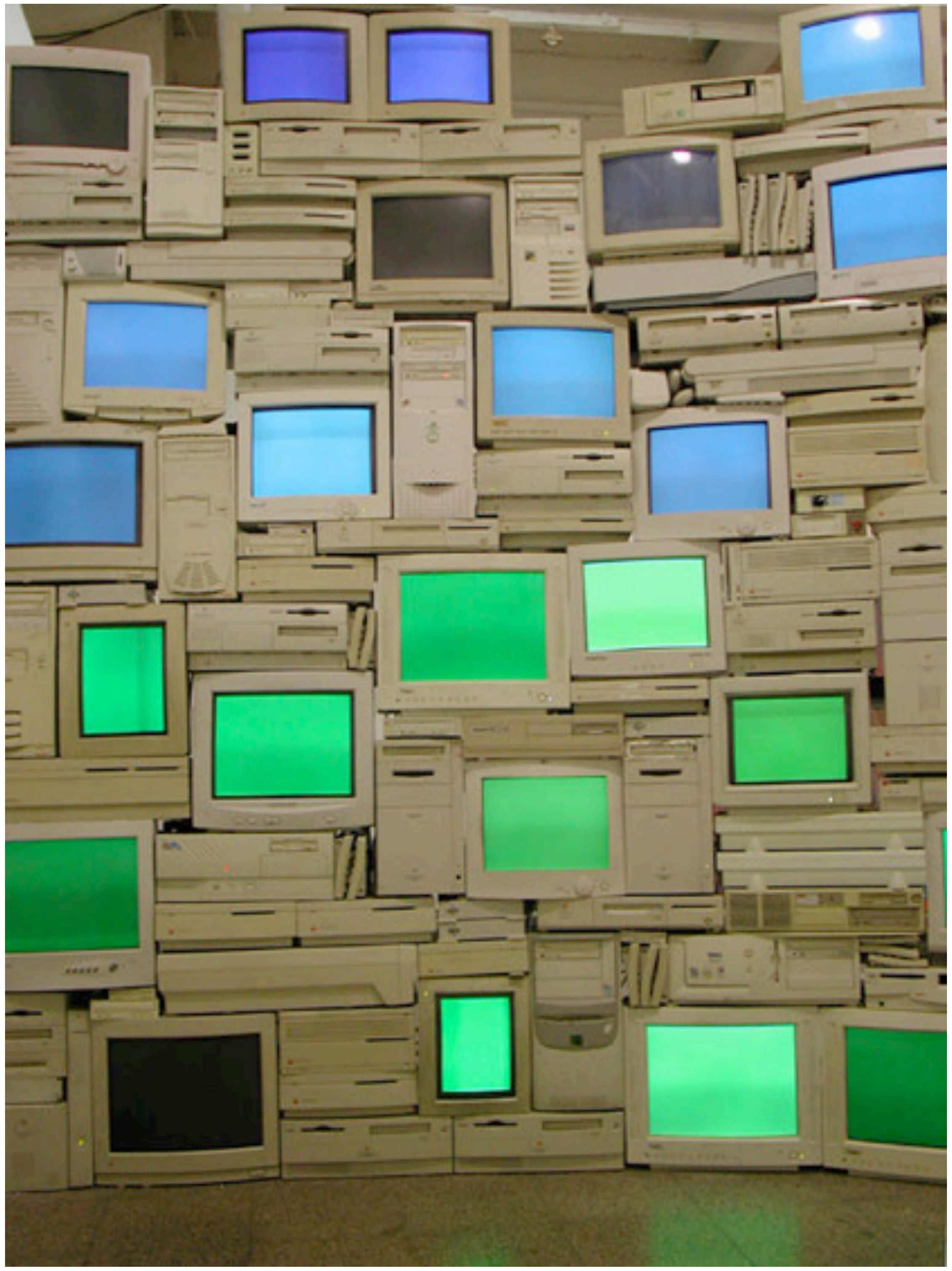
- Stages of software development.

- Requirements gathering



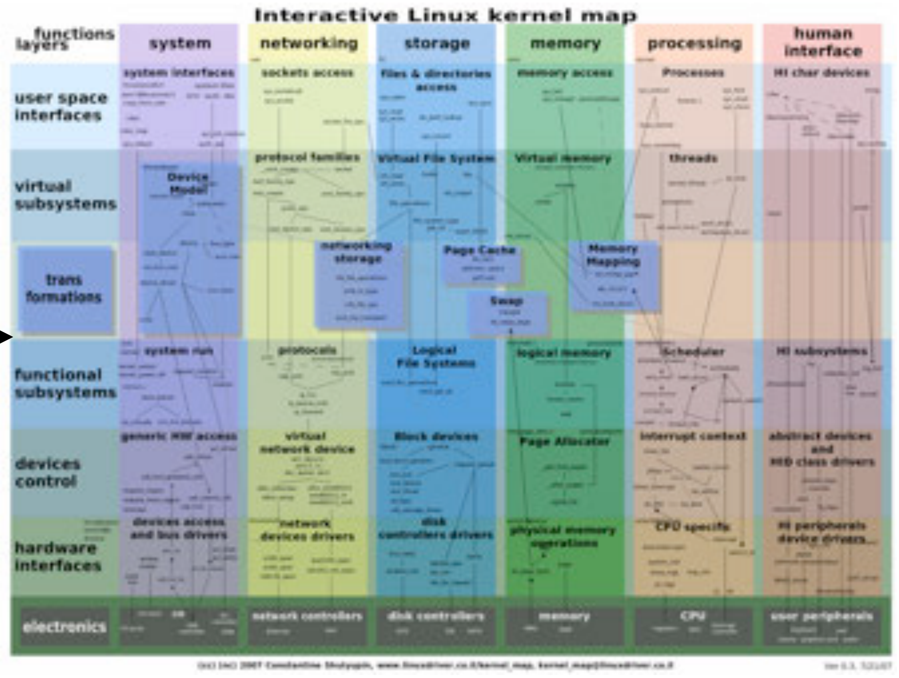
- Design Stage

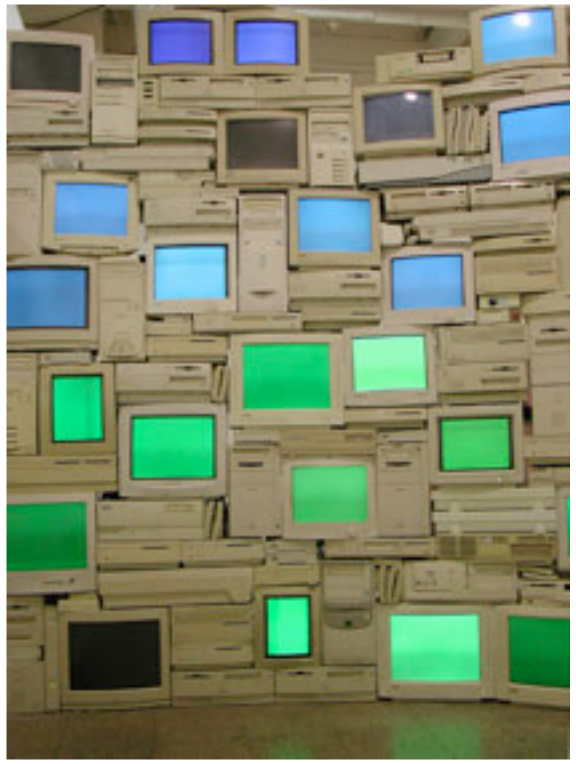
- Testing



- Installation

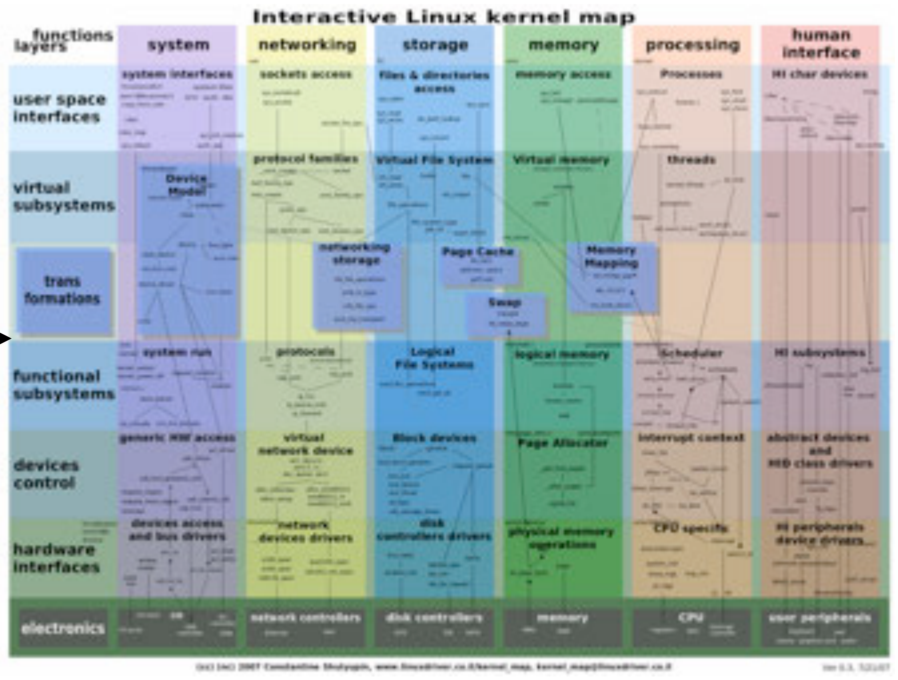
- Maintenance



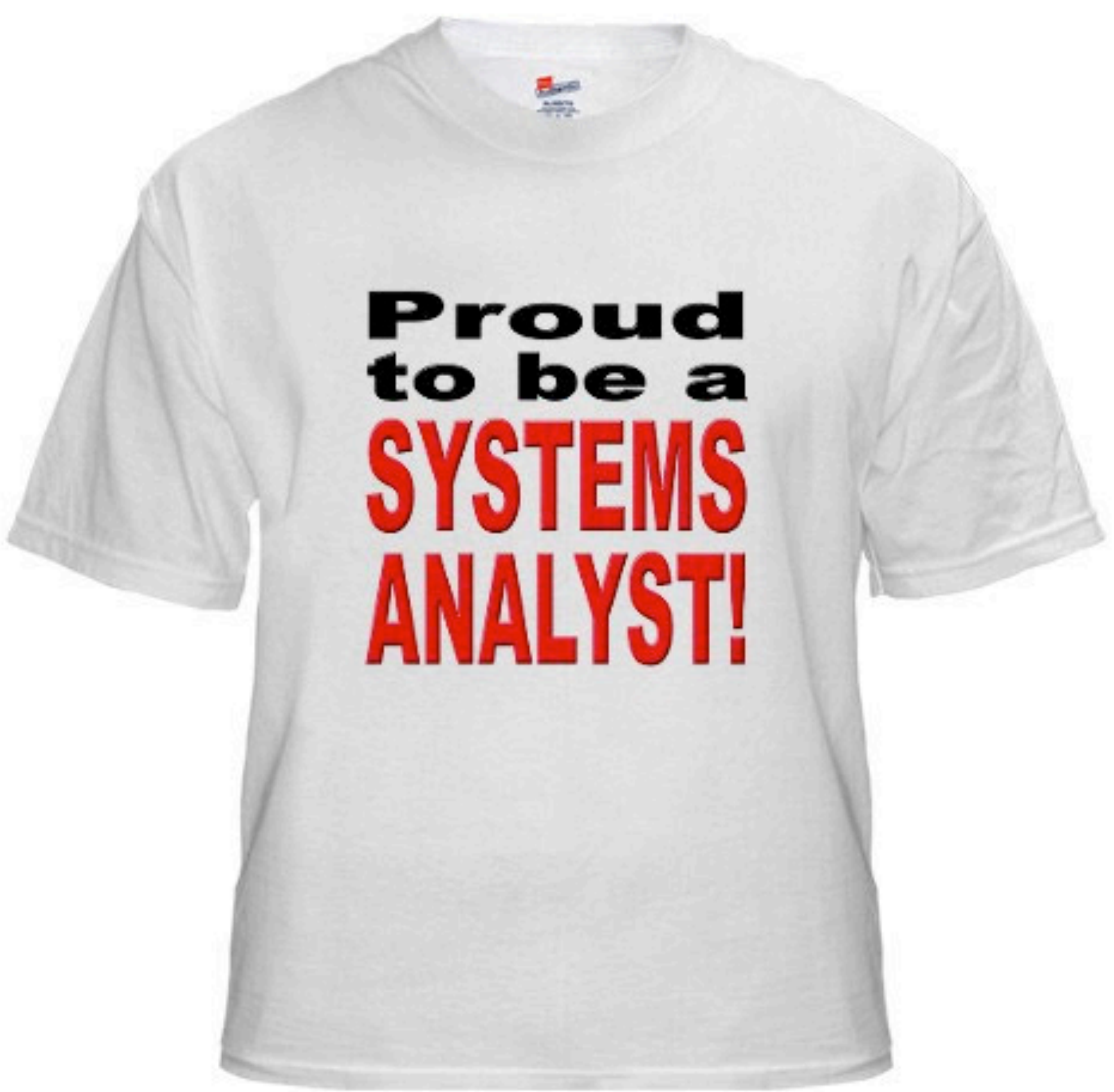


- Software Life Cycle

$\rightarrow$

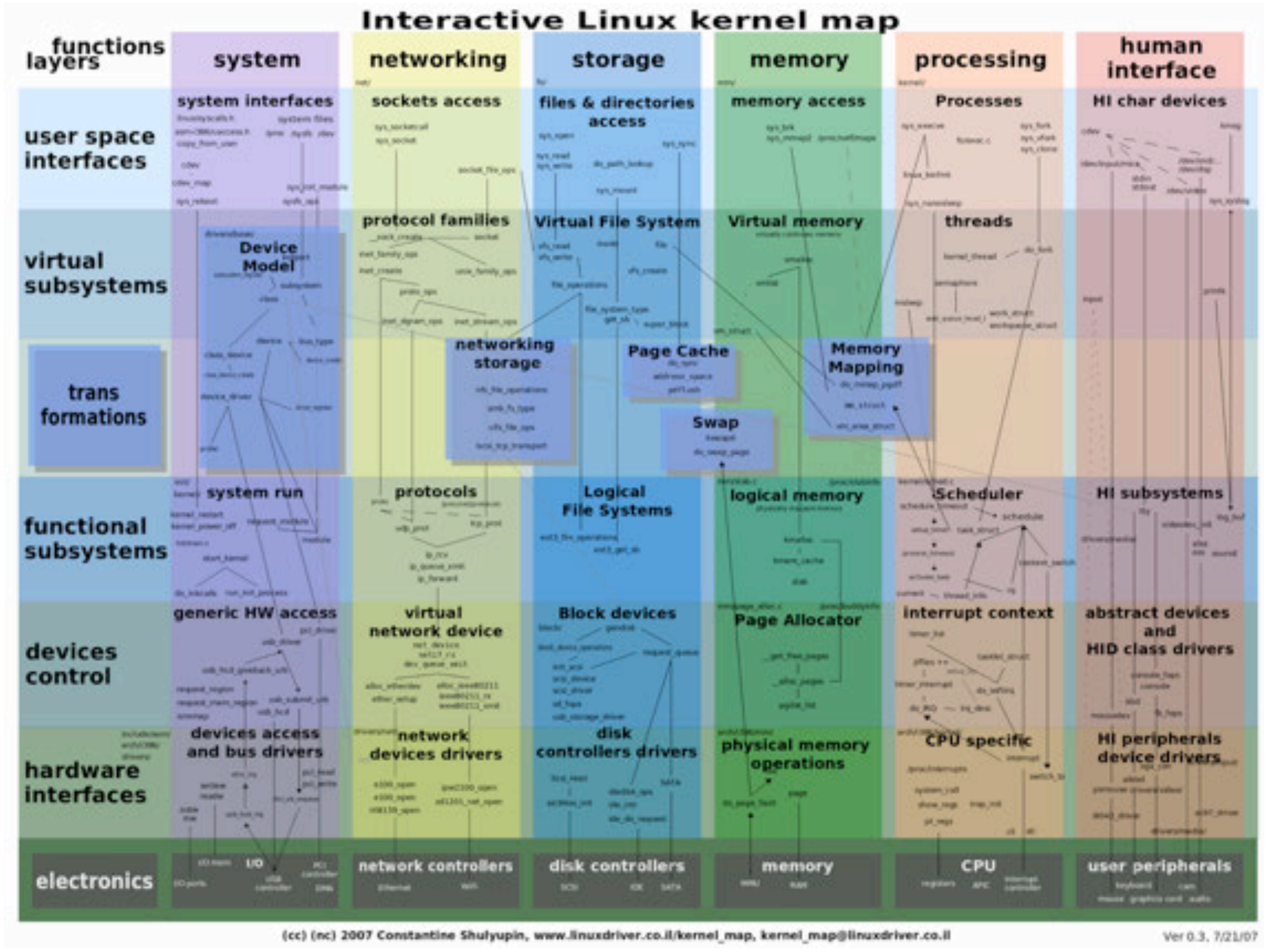


- The focus of this course.

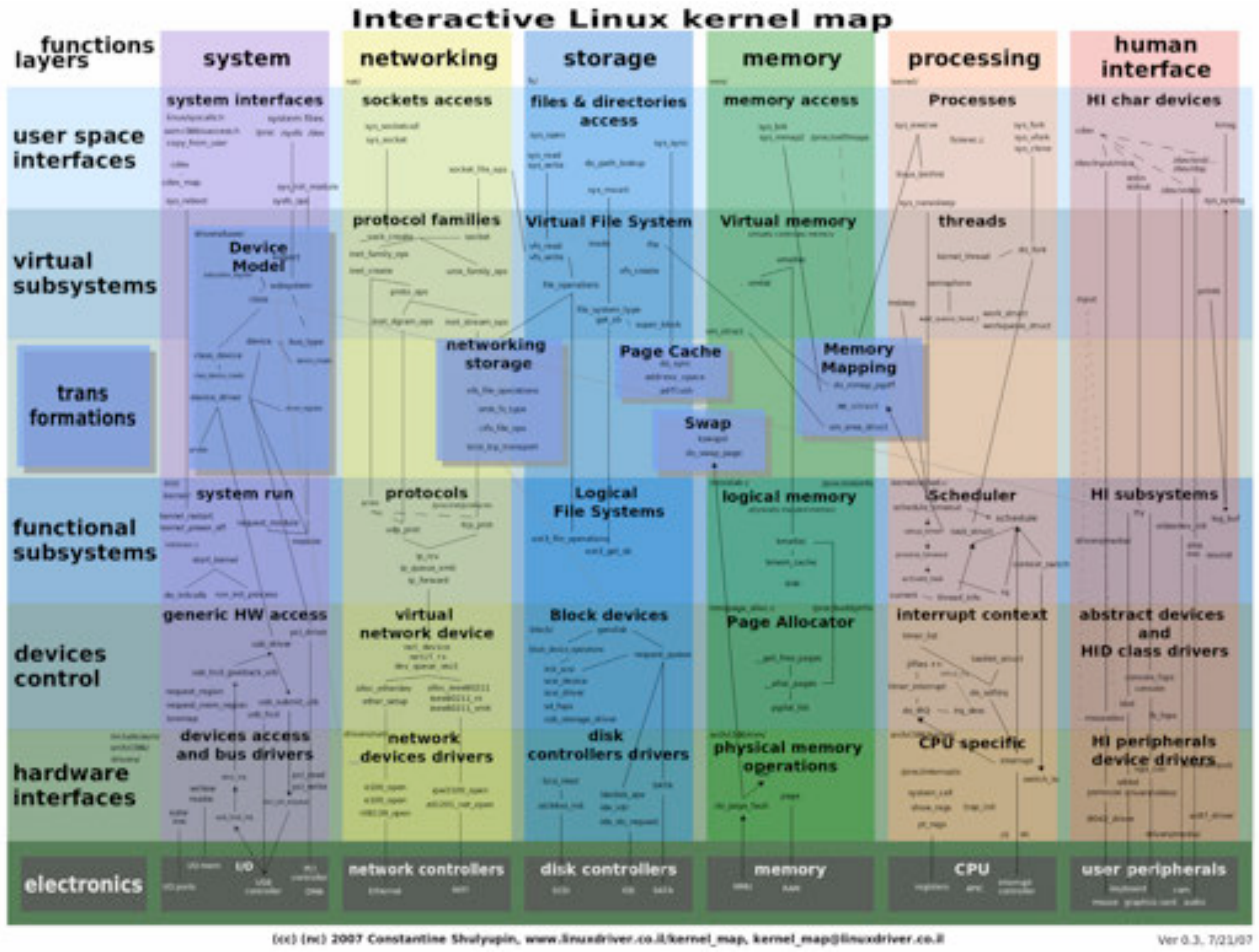


- There is a job known as Systems Analyst

- What are requirements?



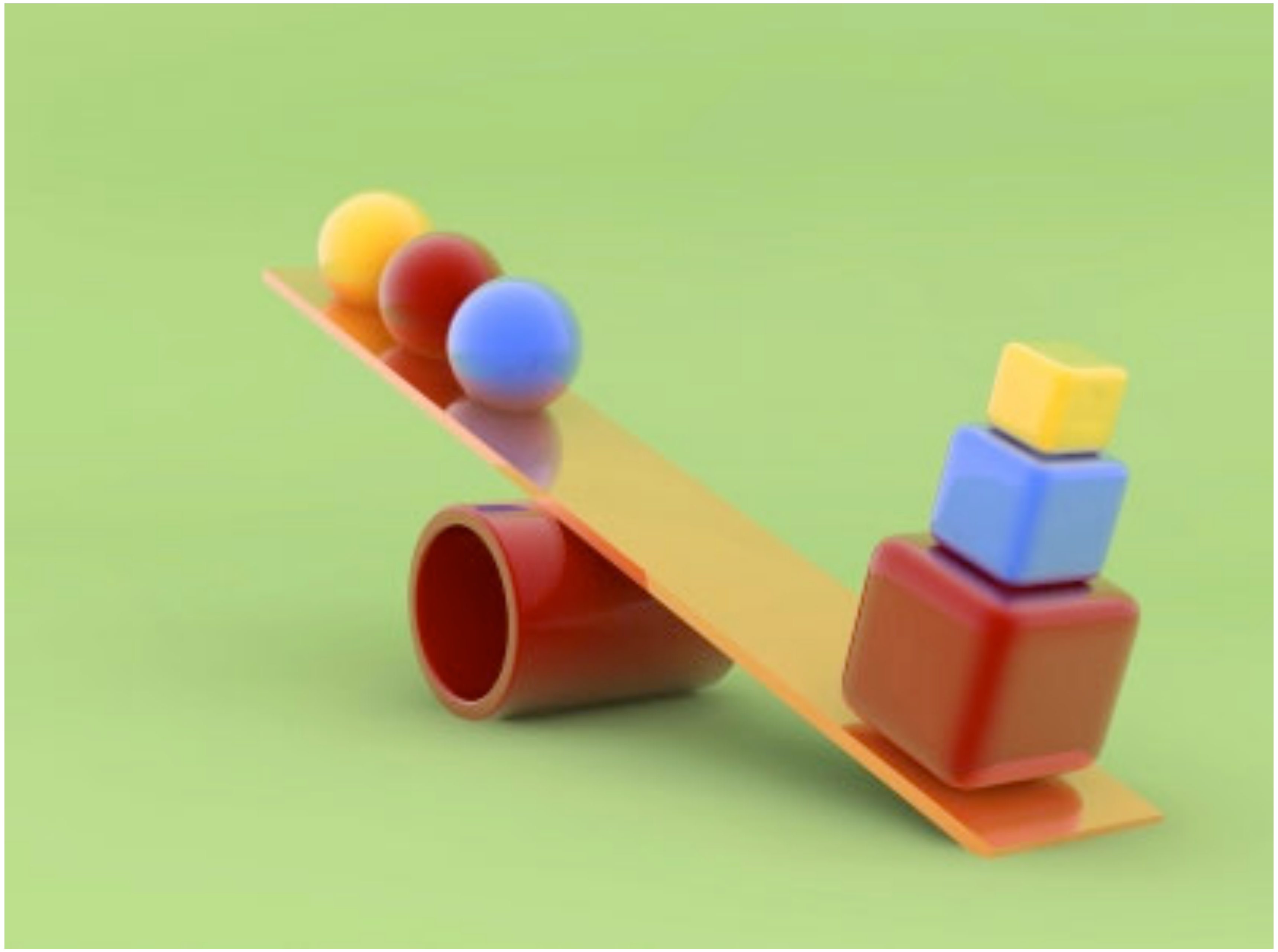
- What is design?



#1 #2

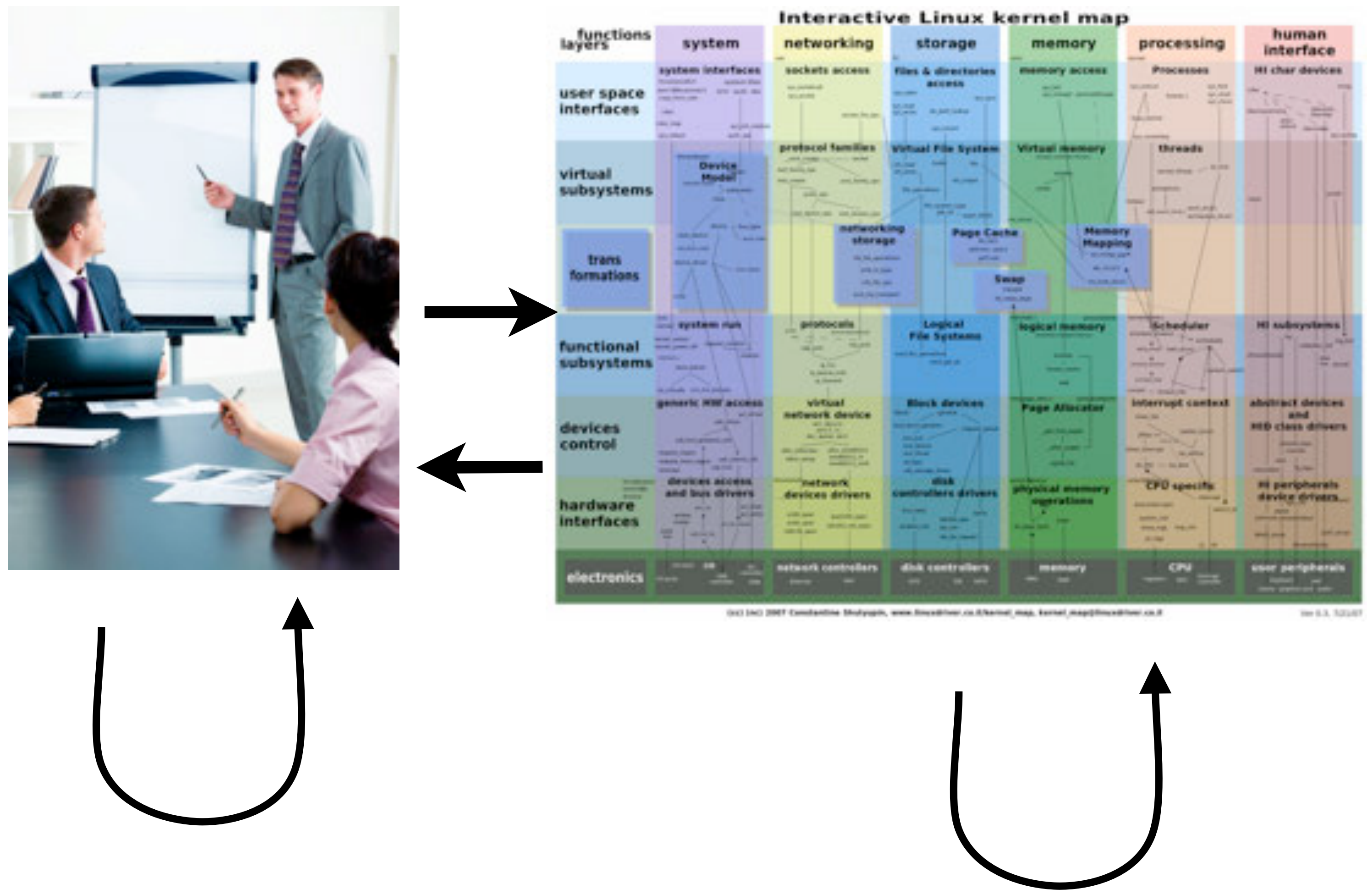
- Why two separate stages?

- Making decisions too early is a problem.



- Balance competing requirements.

- Design cannot begin without requirements.



- Iterations between and within the steps can occur.

- We need an SQL database
- We need a system to organize inventory
- Runs under Windows OS
- Must be able to generate daily reports
- Must be able to handle 100 transactions per second
- Must run on our existing computer system
- Make it as cheap as possible

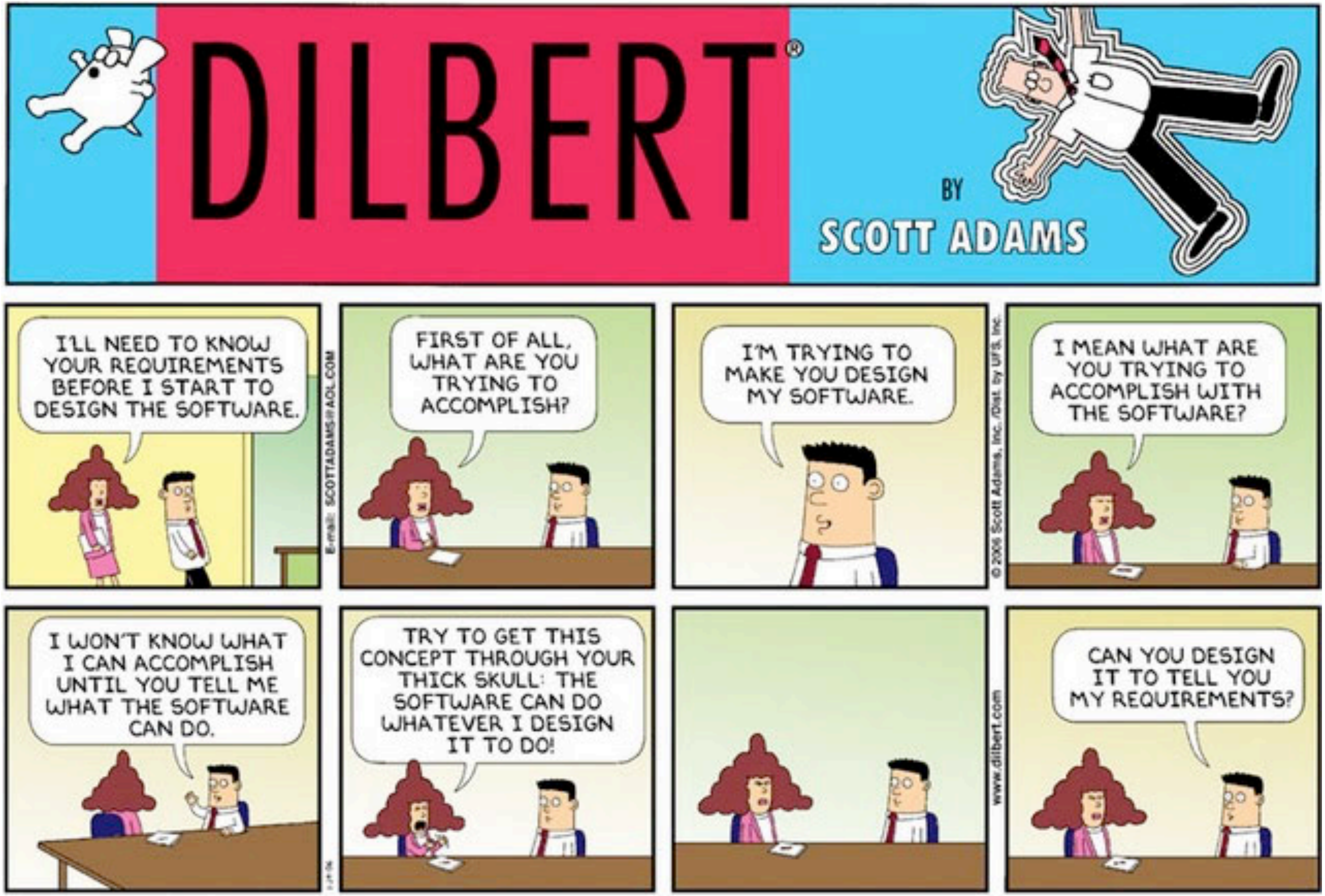
- We will want demos during development every two weeks
- We need some software to solve our problem
- Must be usable by someone with a grade six education
- Must implement all current legal expectations for this type of system
- Must use the most modern security available

Sample Problem

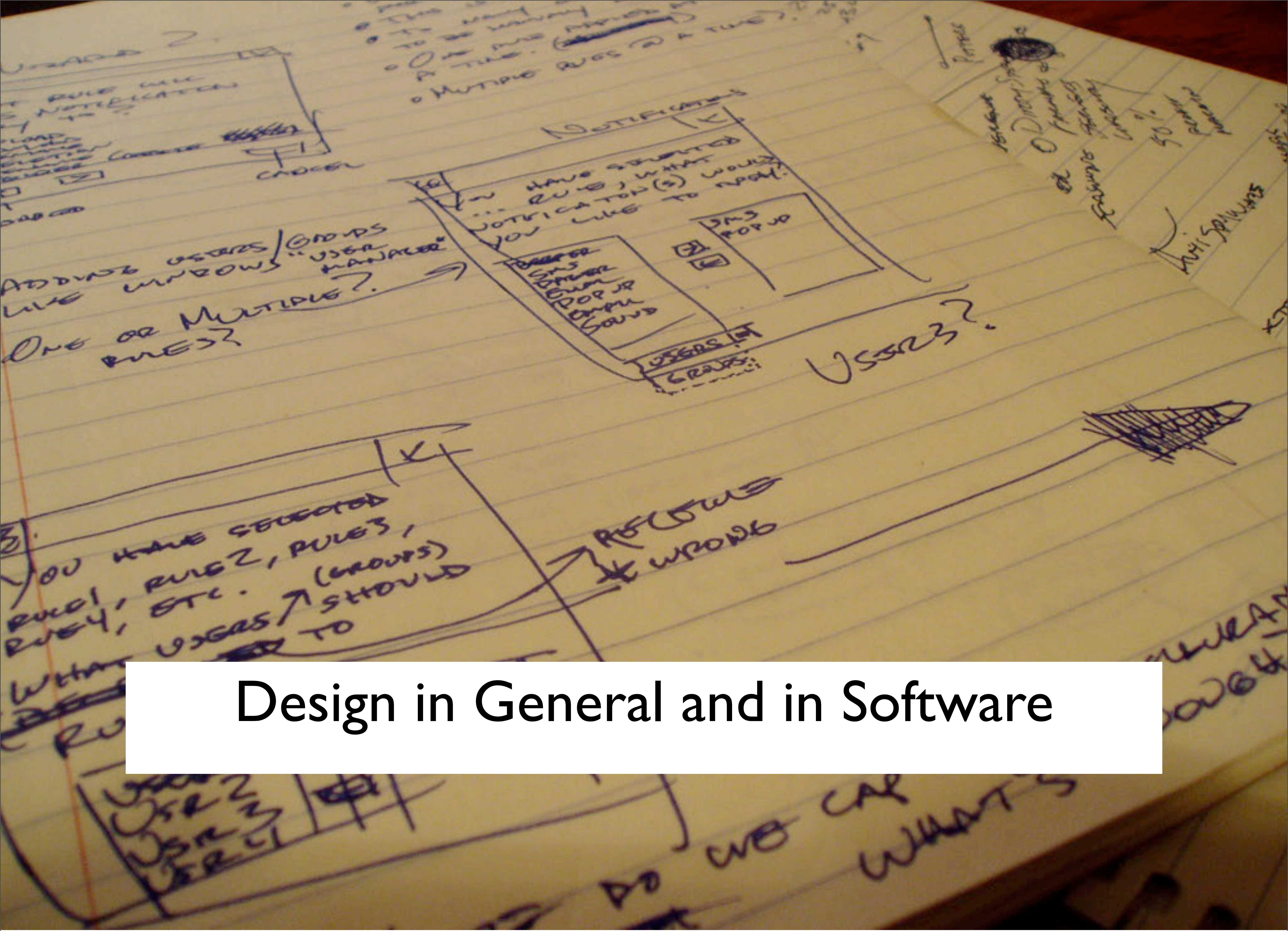
- A System Analyst was asked to develop a computer system to store different forms in a database. The office that managed the forms was having a lot of trouble keeping track of the different forms. The forms were all similar and were being filed in the wrong places or lost.

Solutions

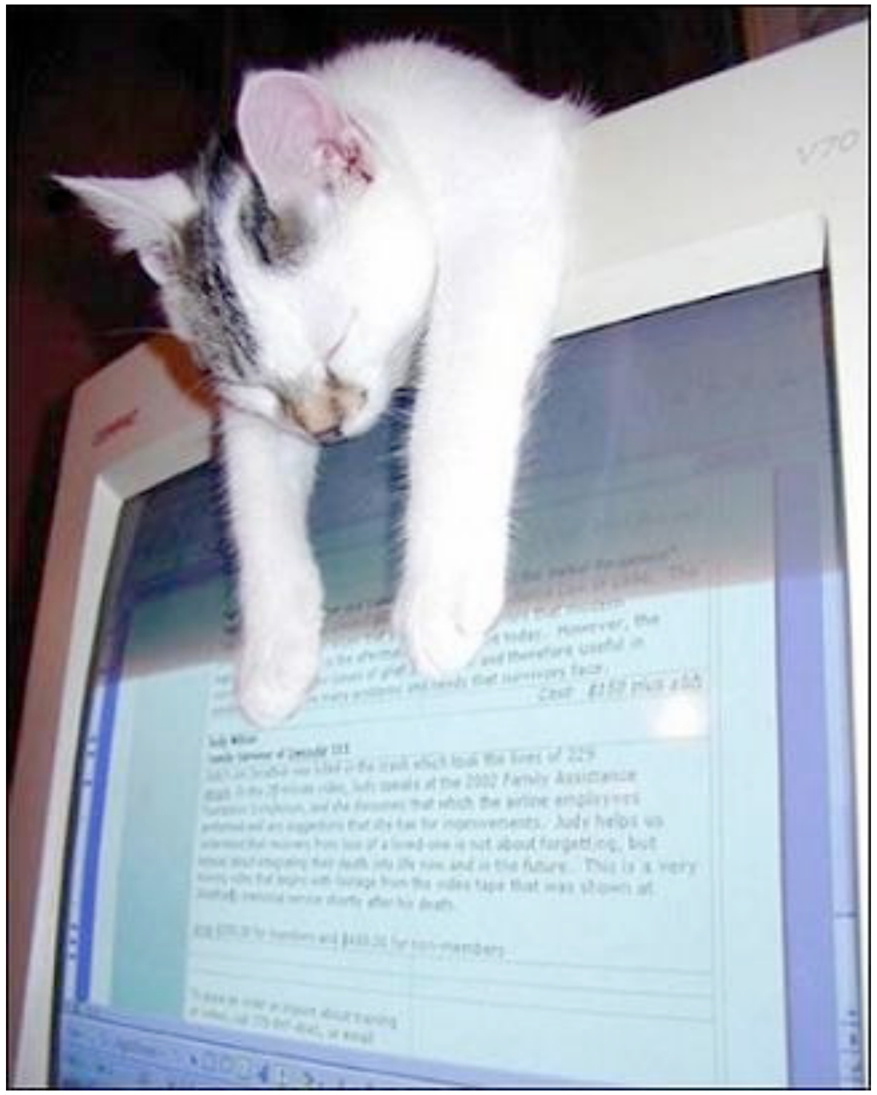
- Build the system they want
- Colour the forms so they cannot be mistaken (this was the chosen solution)


DILBERT®
BY SCOTT ADAMS
I'LL NEED TO KNOW YOUR REQUIREMENTS BEFORE I START TO DESIGN THE SOFTWARE.
E-mail: SCOTTADAMS@AOL.COM
FIRST OF ALL, WHAT ARE YOU TRYING TO ACCOMPLISH?
I'M TRYING TO MAKE YOU DESIGN MY SOFTWARE.
© 2006 Scott Adams, Inc. /Dist. by UFS, Inc.
I MEAN WHAT ARE YOU TRYING TO ACCOMPLISH WITH THE SOFTWARE?
I WON'T KNOW WHAT I CAN ACCOMPLISH UNTIL YOU TELL ME WHAT THE SOFTWARE CAN DO.
TRY TO GET THIS CONCEPT THROUGH YOUR THICK SKULL: THE SOFTWARE CAN DO WHATEVER I DESIGN IT TO DO!
www.dilbert.com
CAN YOU DESIGN IT TO TELL YOU MY REQUIREMENTS?

- http://www.monologuer.com/wp-content/uploads/big-painting.jpg
- http://stackoverflow.com/questions/1936919/what-is-a-good-example-to-show-to-a-non-programmer-to-explain-what-programming-l
- http://www.cafepress.com/+proud_systems_analyst_white_tshirt,84577362
- http://www.artofbi.com/index.php/2010/03/requirements-gathering-change-management-techniques/
- http://agsc.ca/servcies_ba.html
- http://iamanush.com/category/linux/
- http://en.wikiversity.org/wiki/File:ComputerTesting.jpg
- http://badbanana.typepad.com/weblog/2008/09/index.html
- http://www.freedigitalphotos.net/
- http://www.dilanchian.com.au/index.php?option=com_content&task=view&id=90&Itemid=144
- http://visualpop.wordpress.com/2008/01/26/making-sense-of-visual-culture/



Design in General and in Software

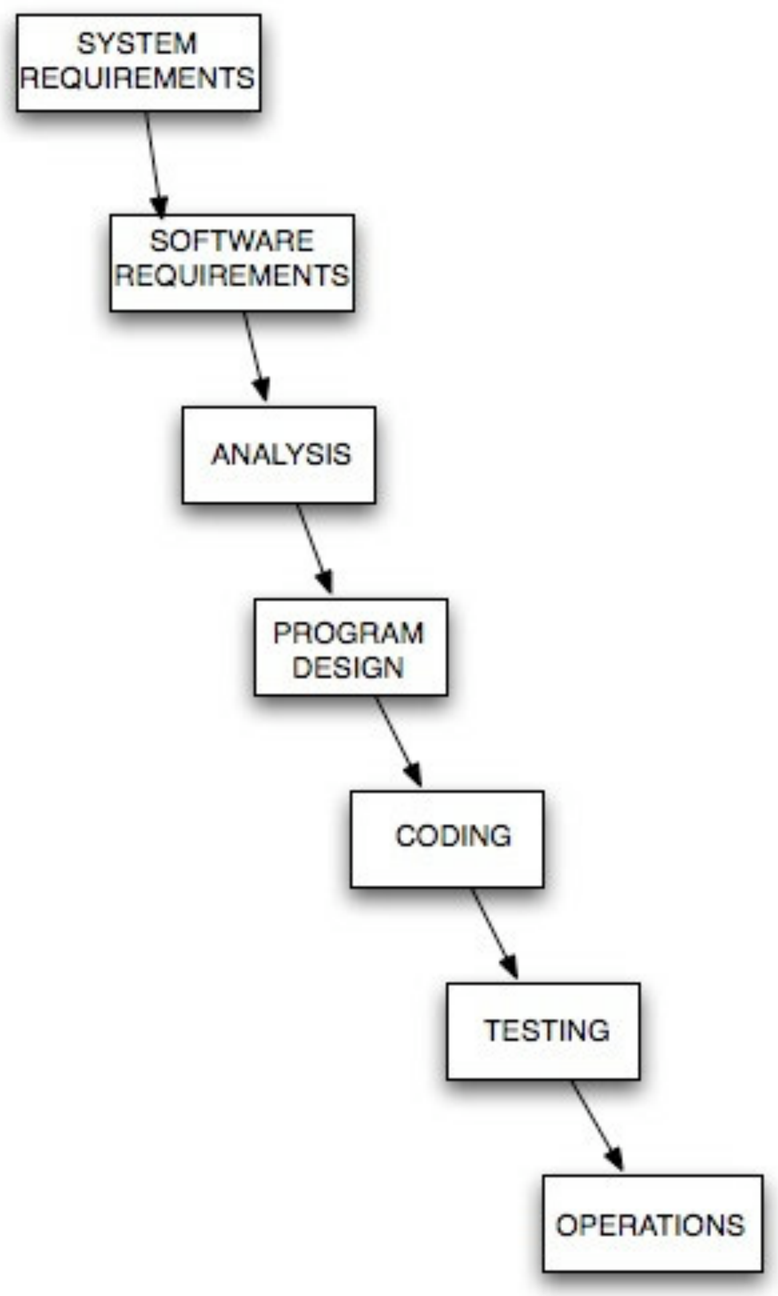


- System Design is not Artistic Design

- Design = plan, implementation, interaction

- Implementation reveals design problems

- Routine designs do exist for software
- Original designs tend to be for new things

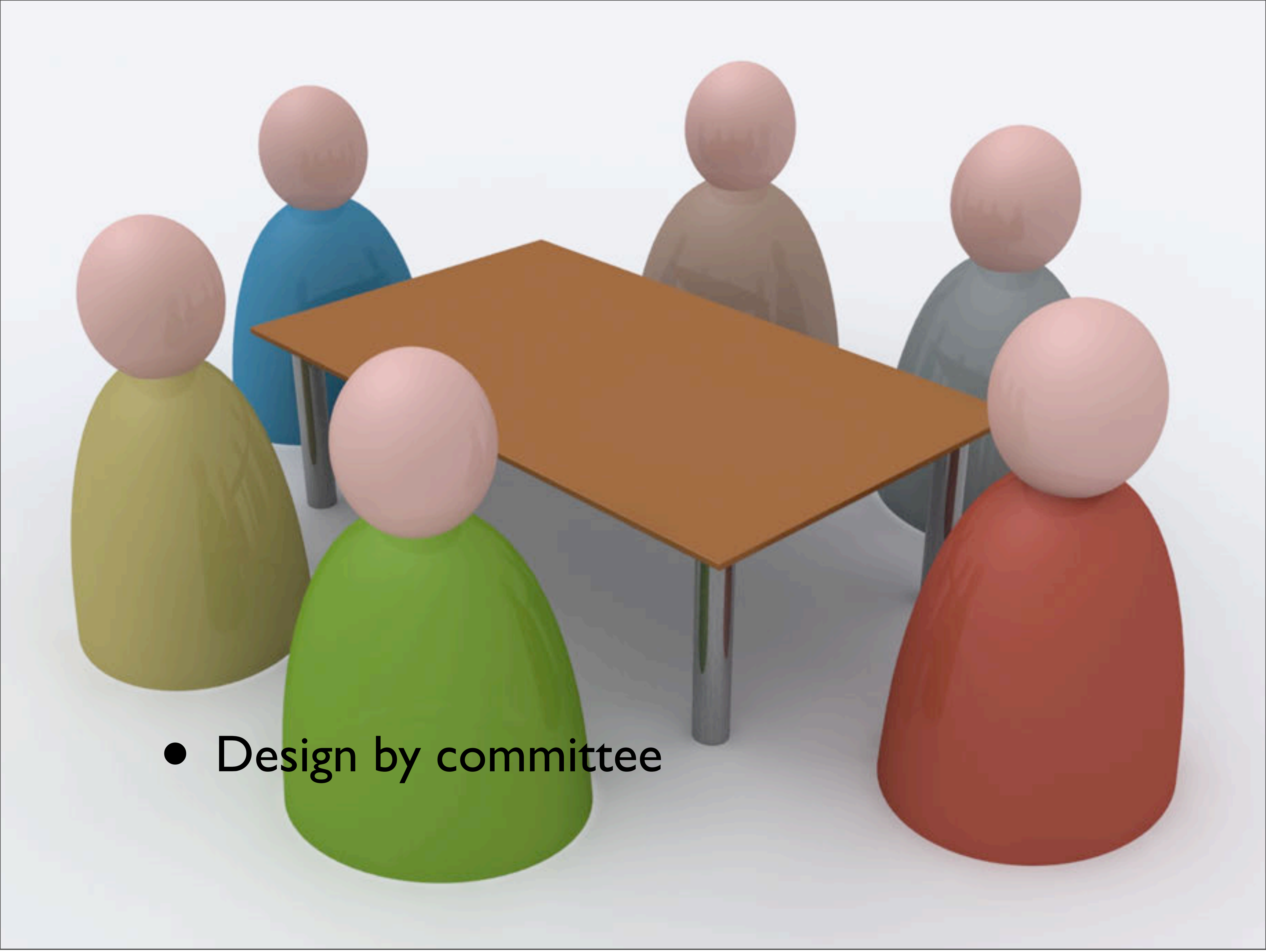


- Waterfall model

- What worked with the Waterfall model?

- What didn't work in Waterfall? Knowing what needs to be done.

- More Waterfall problems. Designers don’t think that way.



- Design by committee

- Requirements bloat and creep

- Collaboration

- Is collaboration always better?

- Why the change from solo to team design?

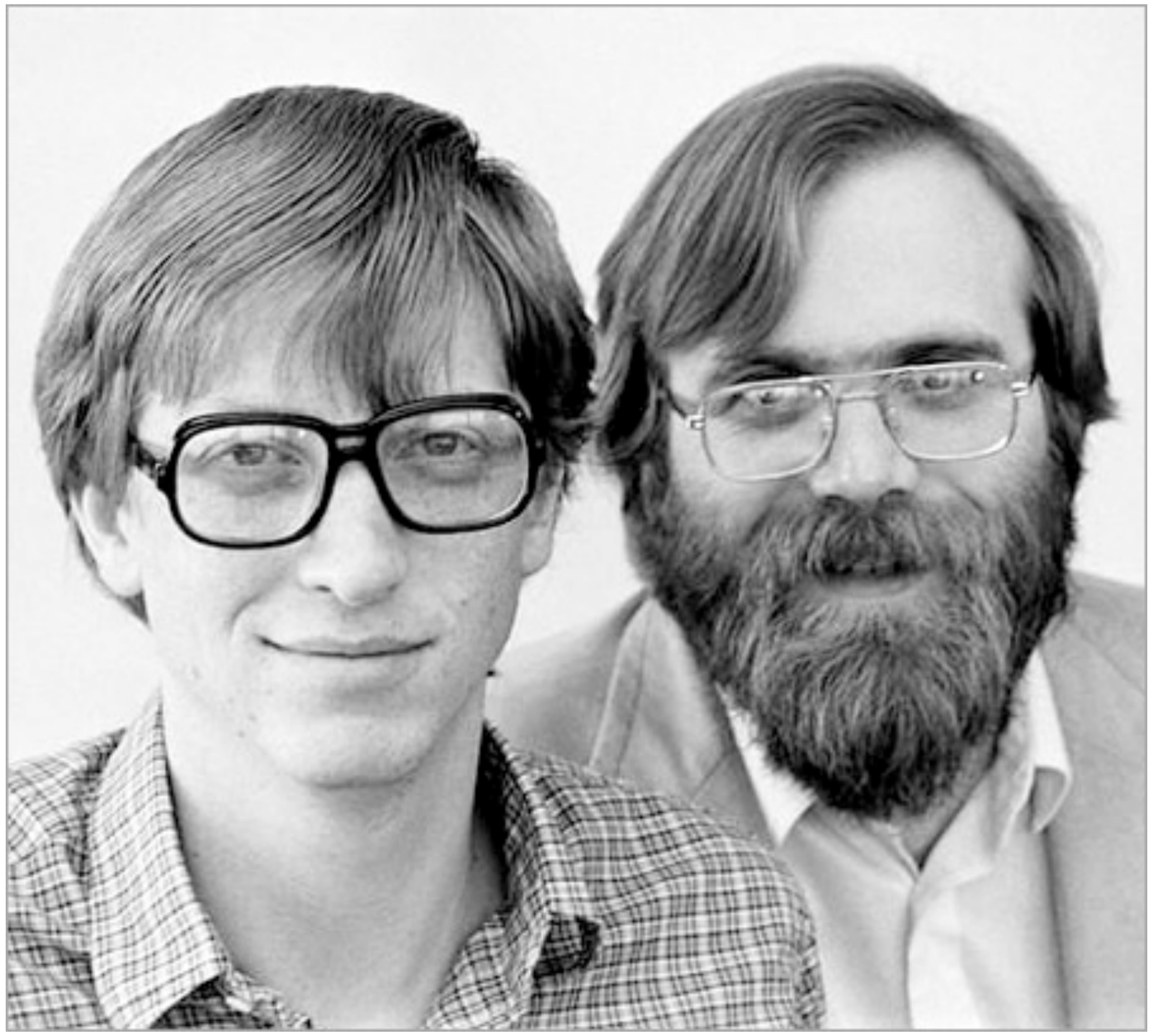
- Why the change from solo to team design?

- Cost of collaboration

- Cost of collaboration

- The challenge of conceptual integrity

- When collaboration helps



- The exception - Two person teams

SUBVERSION

CVS - Concurrent Versions System

Redmine

- Software tools for collaboration

- Reference: The Design of Design by Frederick P. Brooks, Jr., Addison-Wesley, 2010.

- http://yourdon.com/strucanalysis/wiki/index.php?title=Chapter_5
- http://www.pantherkut.com/2007/10/21/cat-sleeping-on-computer-screen/
- http://www.britannica.com/blogs/2009/02/optical-illusion-of-the-day-truck-art/
- http://www.cyberpresse.ca/la-tribune/estrie/200902/16/01-827896-de-lautoneige-a-la-motoneige.php
- http://thedigitalmagpie.com/post-graduate-certificate/essay
- http://www.wired.com/science/discoveries/news/2008/04/dayintech_0404
- http://qwickstep.com/search/francis-crick-and-james-watson.html
- http://www.freedigitalphotos.net/
- http://morguefile.com/
- http://www.sxc.hu/

Design in General and in Software
Part II

- Rational versus Empiricism

-Rationalists believe you can design something solely by thinking about it. Empiricists believe that thought is not enough. They believe that people will inevitably make mistakes in defining the design. To address this they use prototyping, early user testing, iterative implementation testing on a large number of test cases, and regression testing after changes are made.

- User and Use models

-Describe what you know about the user of the software and how the software will be used.
-Often a good idea to start by writing this down (although it isn't often done).
-They allow you to think about the problem in detail and improve the quality of the design.
-All designers have a user and use model but it may subconscious (not explicit).
-Unless they are explicit, different designers will make different assumptions about the design

- What if there is no user information?

-Once you try to build user and use models it becomes apparent that there is a lot that the designer doesn't know about the user.
-This forces you to ask questions earlier in the process.
-Q: What do you do when you cannot get the information?
-A: Guess. It's better to have a poor assumption than no assumption. A poor assumption is explicit so you are aware of it and can fix it later if it is wrong. You will make the assumption anyway so it is better to make it explicit than leave it vague.

Millimetres

Seconds

Memory

Power

Memory Bandwith

Heat

Days

Pages of Text

Water

Politics

- Budgeted Resources

-Something used during development which is scarce and is rationed.
-Popular industrial measures are cost or performance to cost ratio although these are often not used as much as they are talked about.
-Students often get overly enthusiastic about minor performance optimizations.
-Surrogates are often used instead of cost. Lines of code can describe programmer productivity, hours of testing can describe how few bugs are left.

- Budgeted Resources

-The budgeted resource can change in the middle of a design. What if your competitor releases a product with twice the memory of yours during your design process? What if your chip supplier suddenly makes a new model which is twice as fast.
-Explicitly identify budgeted resources and track them. Let the design team know what they are and the status of them during the project. Control them carefully and be cautious about using them up.

- Constraints

-Constraints are both good and bad. They shrink the design space but they limit the choices which the designer can make.
-Having no constraints actually makes designing more difficult.

- Types of constraints

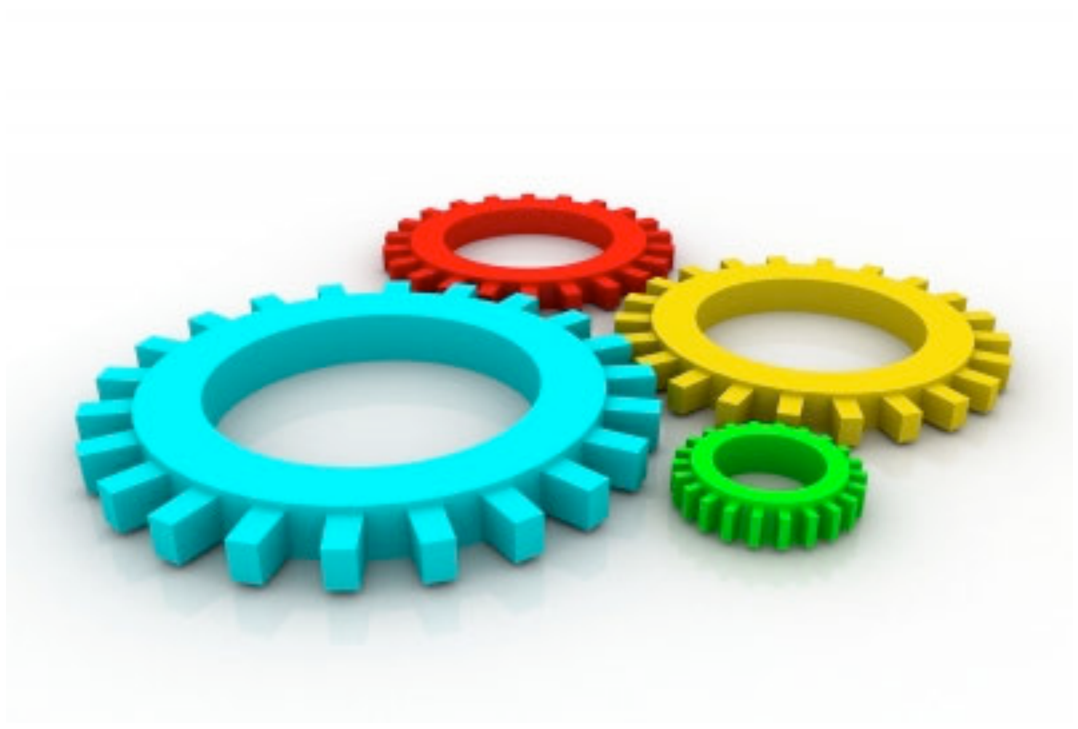
-When working with users you need to determine if the constraints are real or not.
-Real constraints need to be accepted.
-Obsolete constraints are normally not important due to technological advancement.
-Misperceived constraints are ones which the clients believe are important but are not.
-There are also intentional constraints which are not necessary to the design but are applied.These don't show up in software very often. They occur in artistic designs.

- Exemplars

-Definition: A typical example.
-They provide safe models for designers. They contain implicit lists of design tasks and features. The lead away from potential mistakes and act as a starting point for radical new designs.
-A good example is the current GUI standard. If you were making a new graphical interface would it be a good idea to ignore those that currently exist?

- Without exemplars you have only your own experiences.

-By examining other designer's solutions you can learn how they solved problems and benefit from their experiences.
-It isn't lazy to use existing designs which have been proven to work well. It is knowledge of your discipline.
-It is not unoriginal to use exemplars. Originality is not an excuse for ignorance of your discipline.



- Design Process or Design Innovation

-Product process attempts to ensure quality through the development and maintenance process used to build it.
-Innovative or inspired design says that good software is more a factor of good designers than of process.
-Is process a method to try and force mediocre designers to achieve acceptable results?

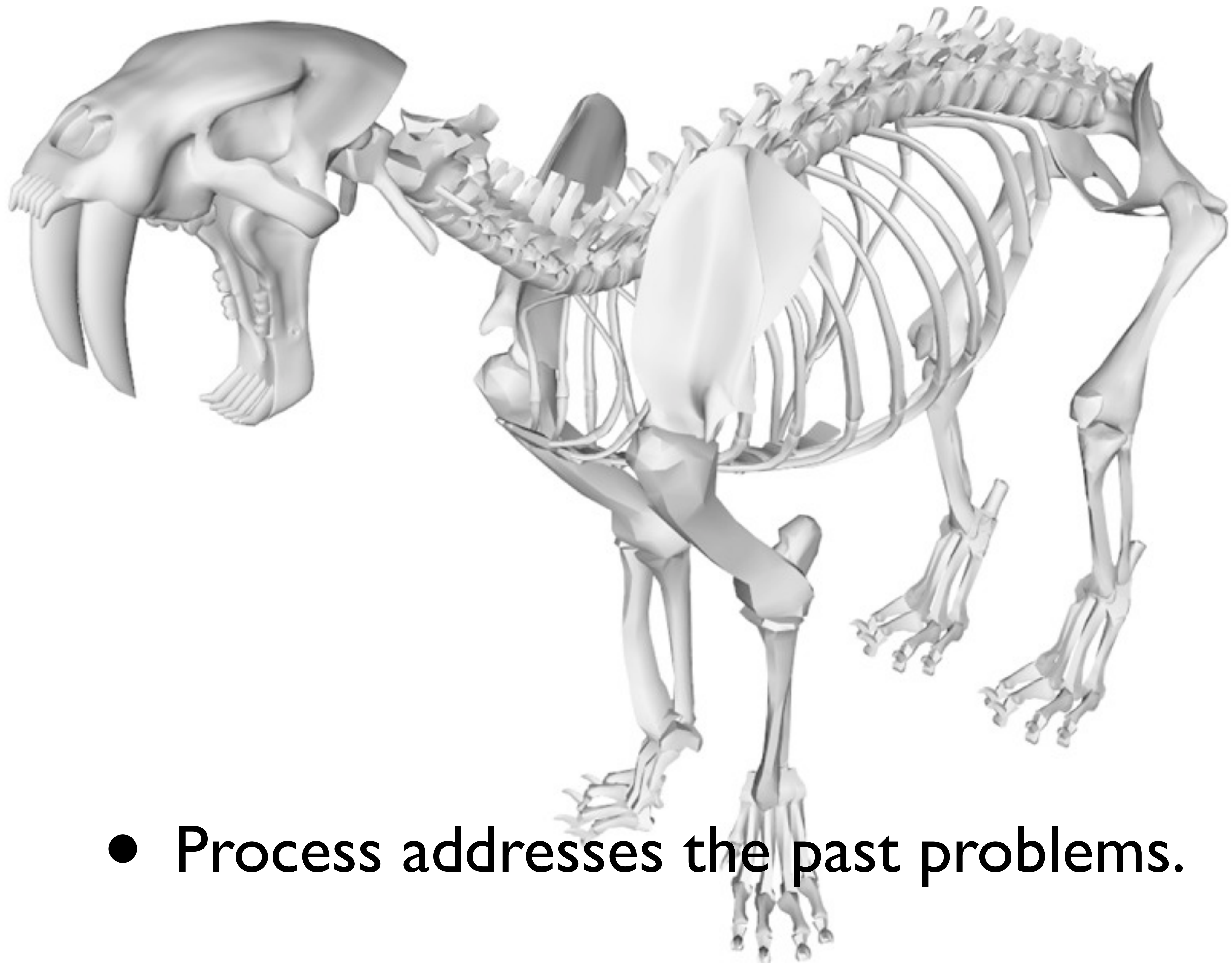
iPhone	Fortran	Penicillin
Unix	Python	Pascal
Apple Interface	Atomic Bomb	Nuclear Submarine

- A lot of great products have been made outside of the normal design process.

-Each of these was made by a small team which was intentionally set apart from the normal production process.
-This may not guarantee success but it raises some important questions.

- Does product process limit great design?

-Process is used to order the development of new products.
-It is conservative by nature. It is easier to create similar but slightly different things than it is to create something very unusual and innovative. The innovative ideas don't fit into the framework of the process.
-The purpose is to create predictable results for the next product. Great designs are often not easy to predict.



- Process addresses the past problems.

-When designing something new the processes which worked in the past may no longer be appropriate.

- Process is veto oriented

-It is designed to block bad ideas and catch oversights. It aims to inhibits products that wont sell, cost too much, and to limit functions and schedules which cannot be met.
-Corporate process control restricts products which might compete with your current products.

- Process is driven by consensus.

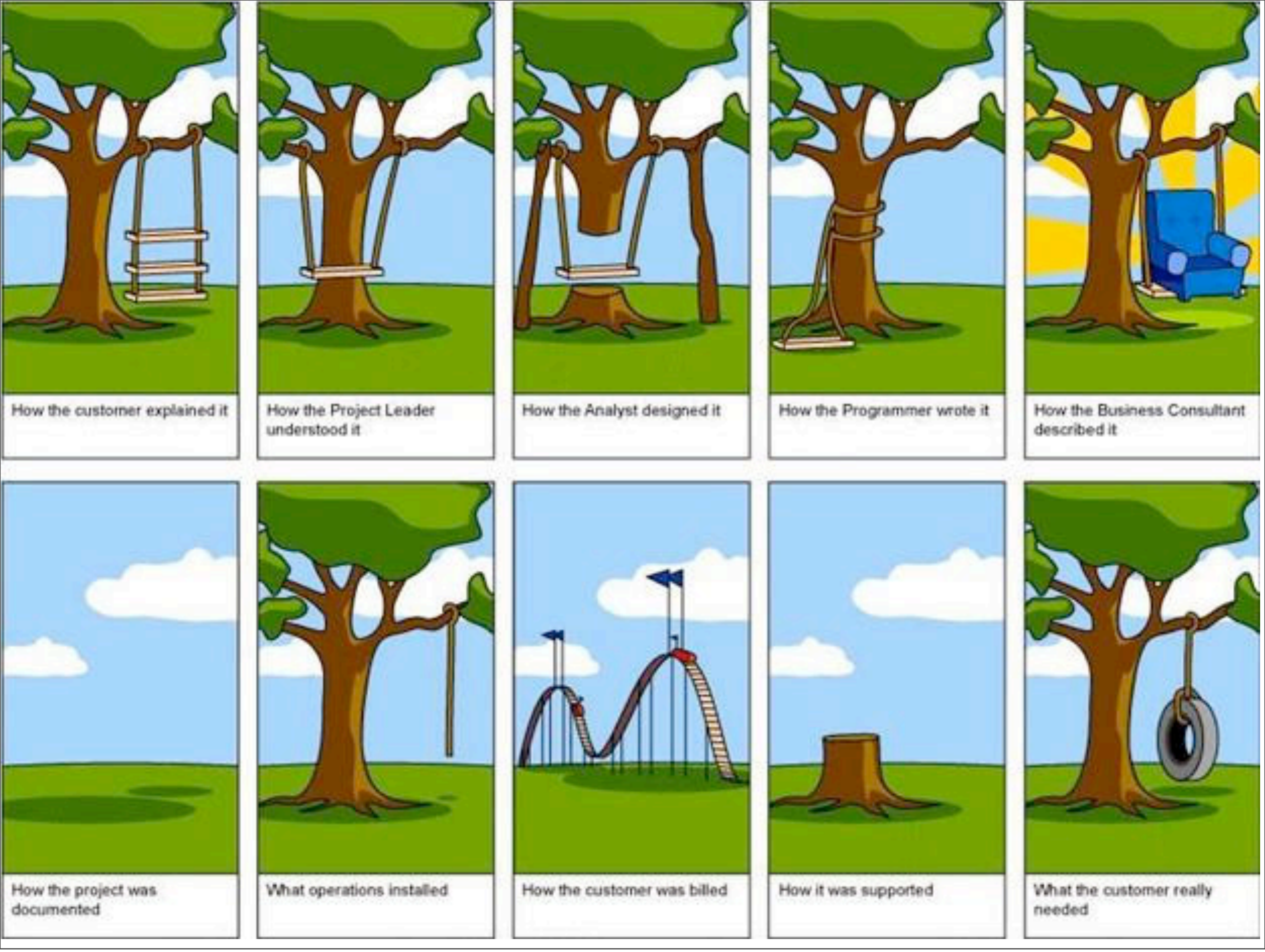
-The process is typically driven by consensus of the experts(committee).
-Experts are expected to avoid mistakes in their area of expertise. They aren't there to create insightful designs.
-If new ideas are not vetoed it is often diluted through compromise.

- Do we need process at all?

-All the the above problems occur when process inhibits innovation.
-The trick is to allow an innovative design to be developed before too many restrictions are considered. This gives the basic ideas a chance to be seriously explored without being prematurely discarded.
-At some point corporate approval is needed, experts need to be consulted, and a schedule and budget must be created.

- Updated products

-Not all new designs need to be innovative. Many are incremental improvements applied to existing products.
-Once users have adopted a product the opportunity to change it dramatically is limited. The users do not want to lose the functionality they currently have with the product.


How the customer explained it
How the Project Leader understood it
How the Analyst designed it
How the Programmer wrote it
How the Business Consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed

- Reference: The Design of Design by Frederick P. Brooks, Jr., Addison-Wesley, 2010.

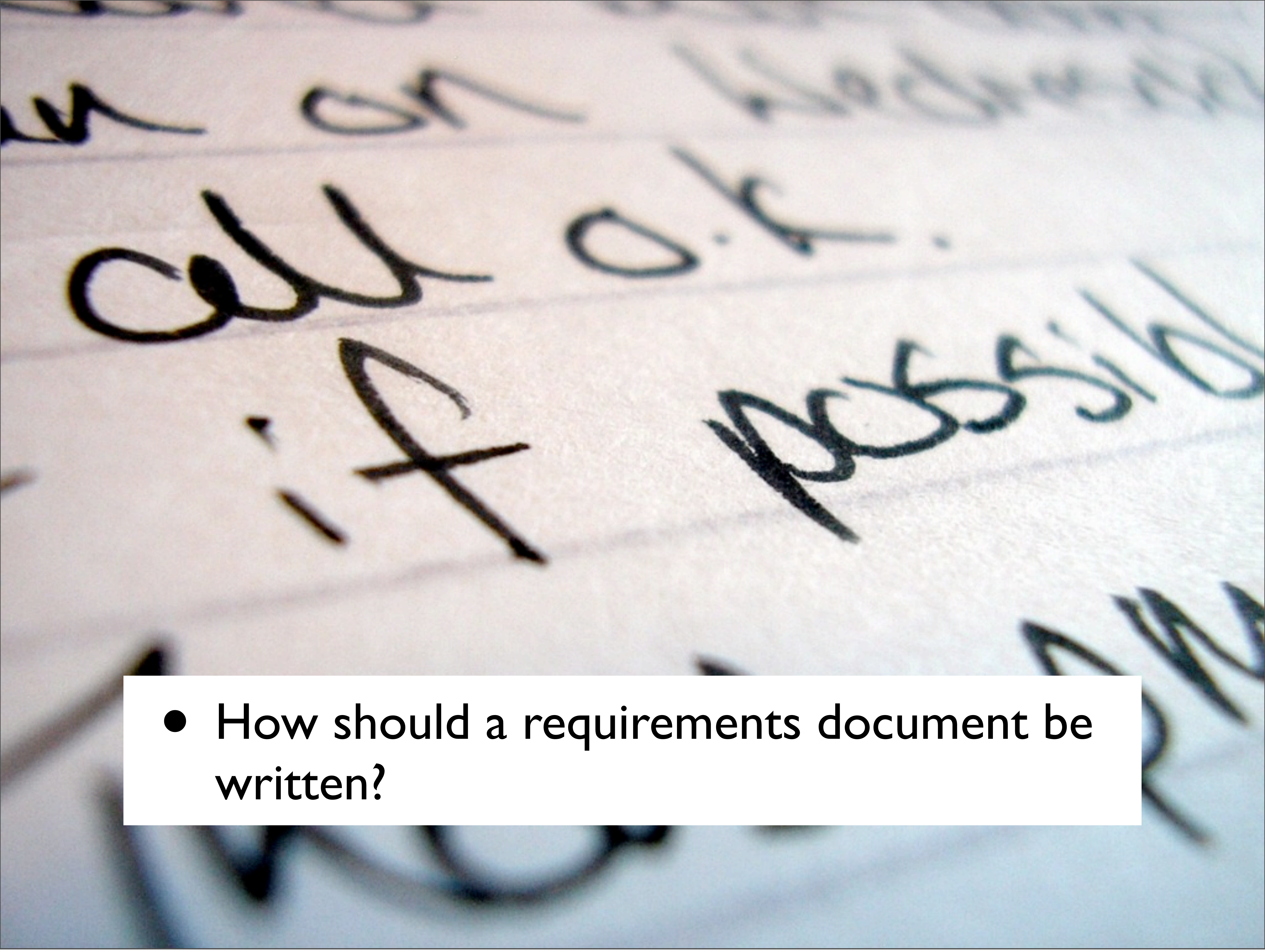
Requirements Analysis

- Stakeholders

-People affected by the system.
-People who influence the requirements.
-This includes users, managers, those who develop and maintain the system, external bodies who regulate the company (eg. for safety, taxation).

- What is a requirements document?

–An official statement of the system requirements for the users and developers.
–Other names include ‘functional specification’, ‘requirements definition’, ‘software requirements specification (SRS)’, ‘safety/reliability plan’.



- How should a requirements document be written?

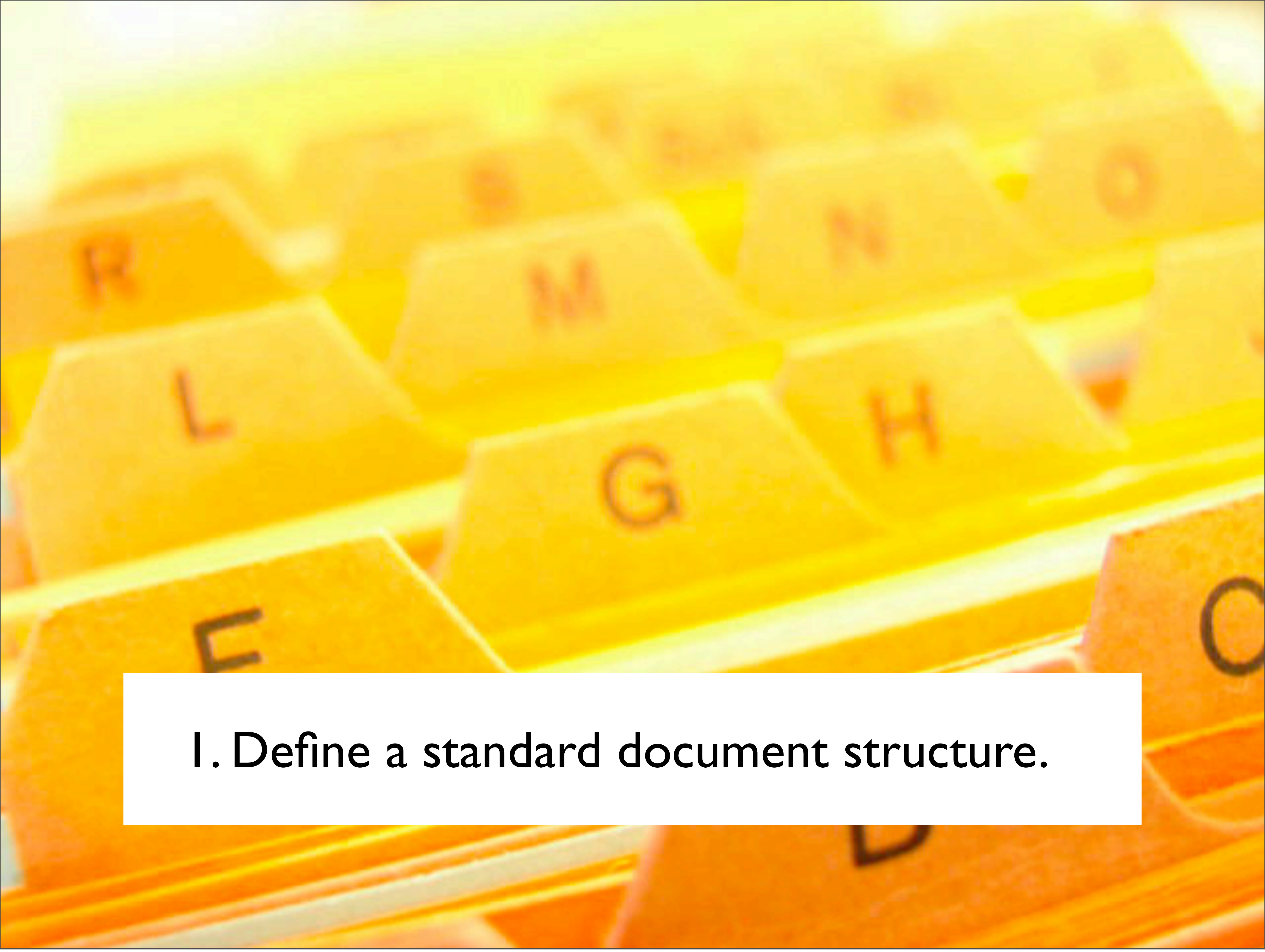
-It should be written so that its audience can understand it.
-It can be written using language which reflects the background of the source. If an engineer provides requirements then it may be written using engineering terms.
-It will be written using natural language.
-Avoid vernacular and attempts to be funny. They become annoying.

- The requirements gathering process.
- Elicitation, Analysis and Negotiation, and Validation

-Elicitation -discover the requirements by consulting with the stakeholders, from existing documentation, and domain knowledge.
-Analysis and Negotiation -analyze the requirements and negotiation with stakeholders to decide which will be accepted. Deal with inconsistencies.
-Check the requirements for consistency and make sure they address all of the problem.

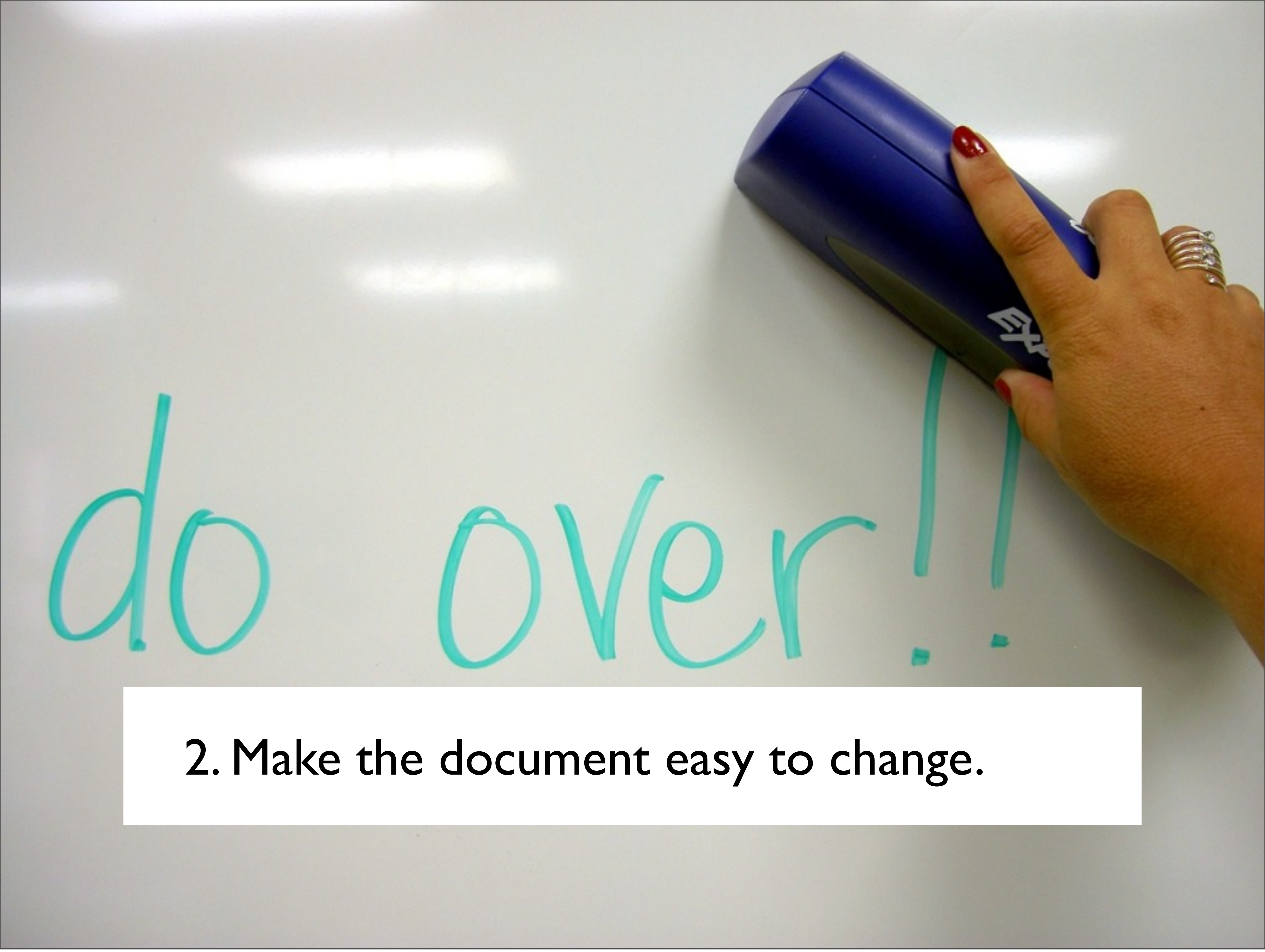
- The Top Ten Requirements Guidelines

-Most organizations have their own methods for dealing with requirements gathering and documenting but these ten points will always be useful.
-Most of them are simple things you can do to check the quality of the requirements list.



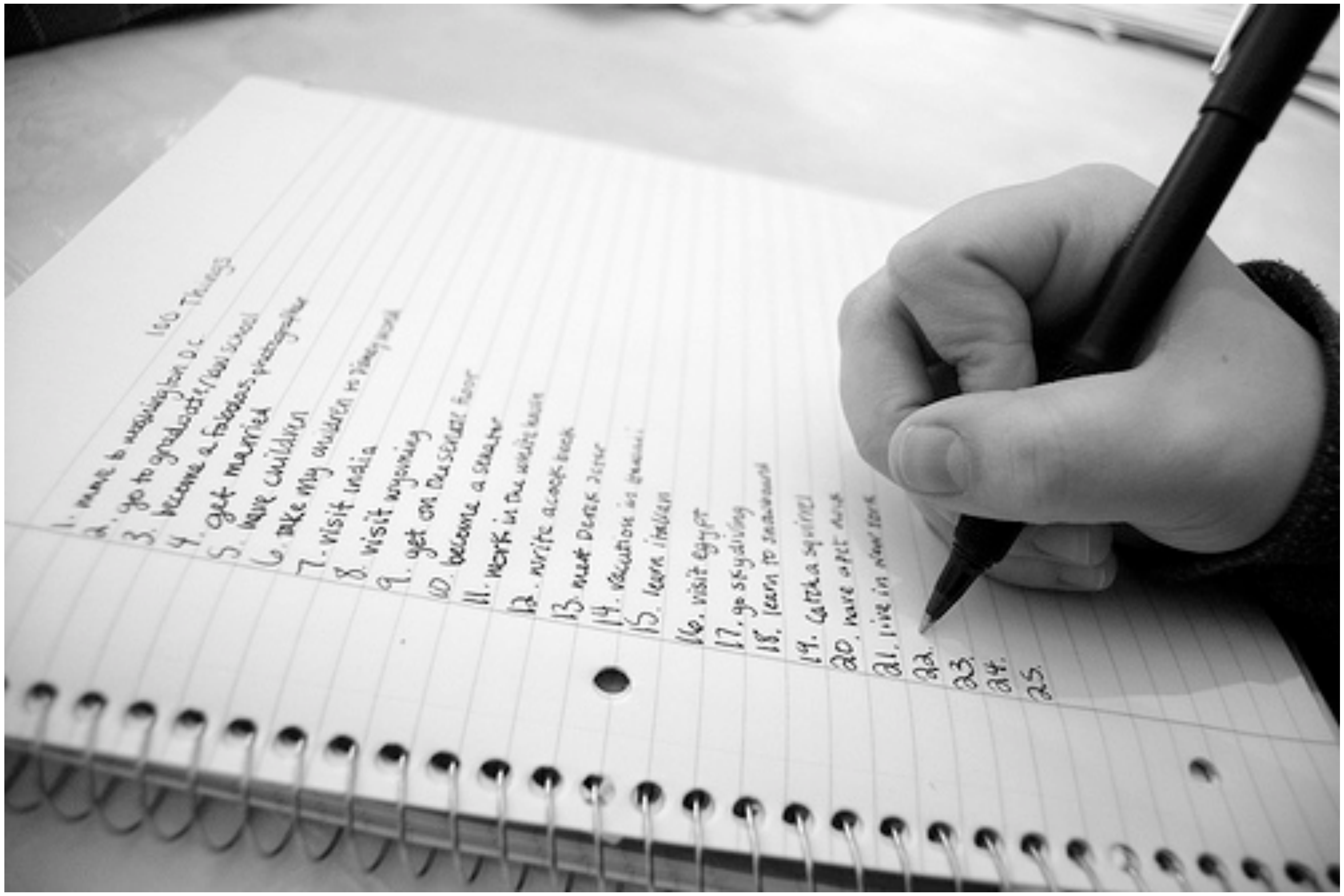
1. Define a standard document structure.

-Readers will become familiar with the format and know how to use it.
-Acts as a checklist for the writer so they don't accidentally omit something.
-Software templates can be developed to help format the document.



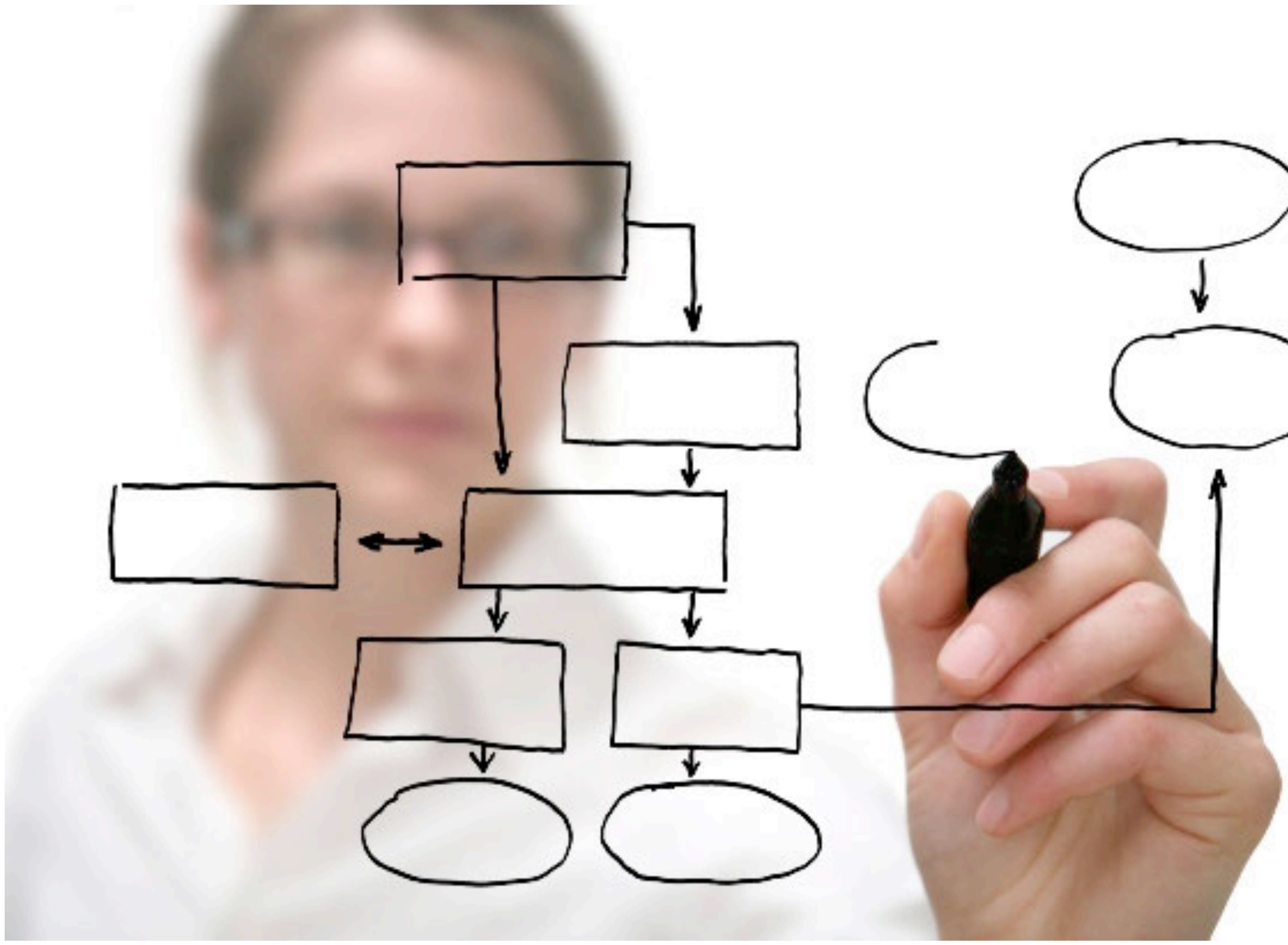
2. Make the document easy to change.

-Writing, reviewing, and distributing new documents is expensive and time consuming. Don't make it more difficult than it needs to be.
-If the cost of making changes is too high then changes may be collected and applied in a batch. This means the document can be out of date while the changes wait to be applied.
-Online versions of the documents can lessen the effect of these problems.



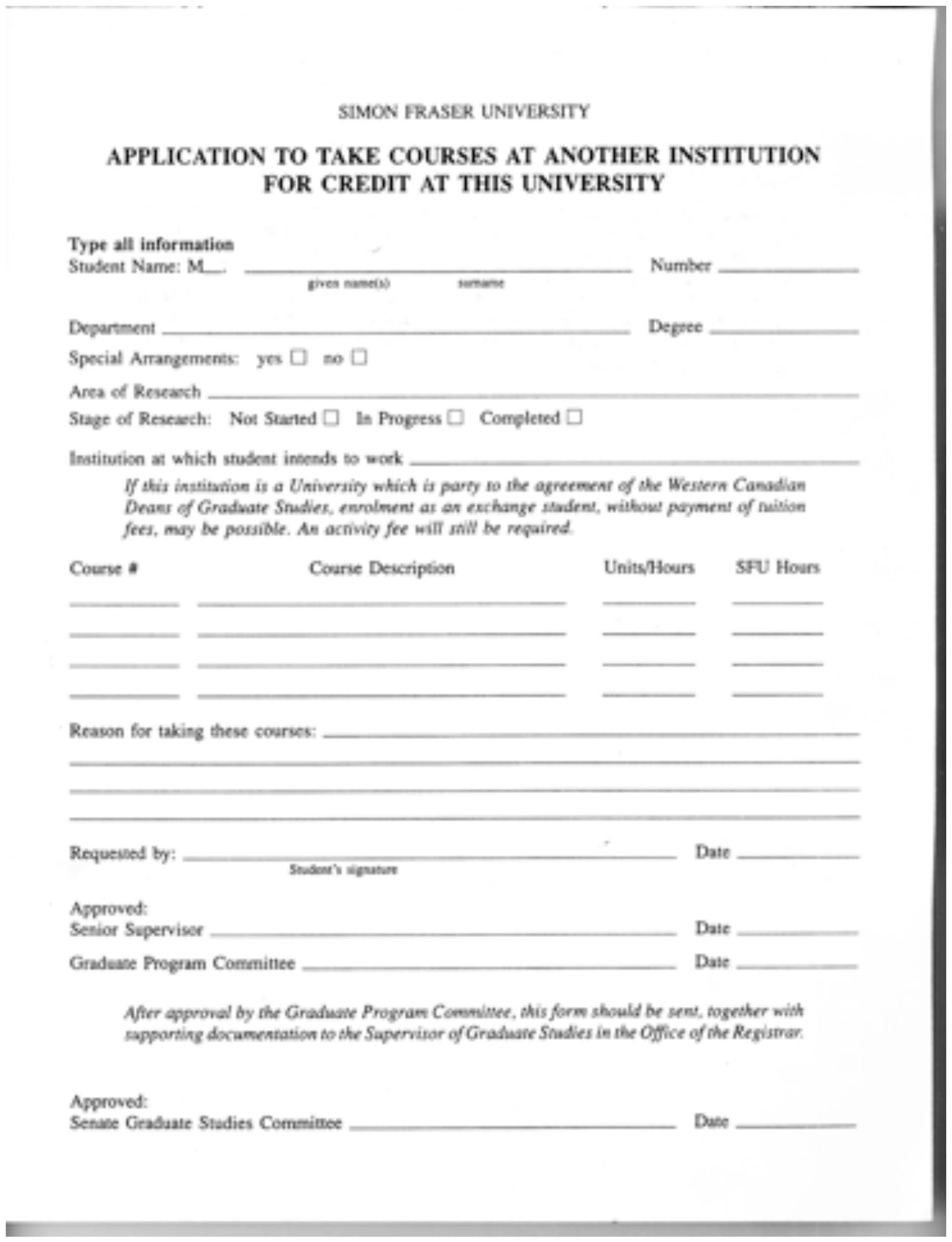
3. Uniquely identify each requirement.

-Give each requirement a unique number.
-They can be referenced through the number in the document.
-The numbers can be used to show which requirements have created other requirements. For example, “Requirements 3.1 and 3.2 were added to meet the needs of requirement 2.0.”



4. Define policies for requirements management.

-Explicitly tell people what they are expected to do and why it is done.



SIMON FRASER UNIVERSITY

APPLICATION TO TAKE COURSES AT ANOTHER INSTITUTION FOR CREDIT AT THIS UNIVERSITY

Type all information

Student Name: M__. ____________________ Number __________

given name(s) surname

Department ____________________ Degree __________

Special Arrangements: yes ☐ no ☐

Area of Research ____________________

Stage of Research: Not Started ☐ In Progress ☐ Completed ☐

Institution at which student intends to work ____________________

If this institution is a University which is party to the agreement of the Western Canadian Deans of Graduate Studies, enrolment as an exchange student, without payment of tuition fees, may be possible. An activity fee will still be required.

Course #	Course Description	Units/Hours	SFU Hours

Reason for taking these courses: ____________________

Requested by: ____________________ Date __________

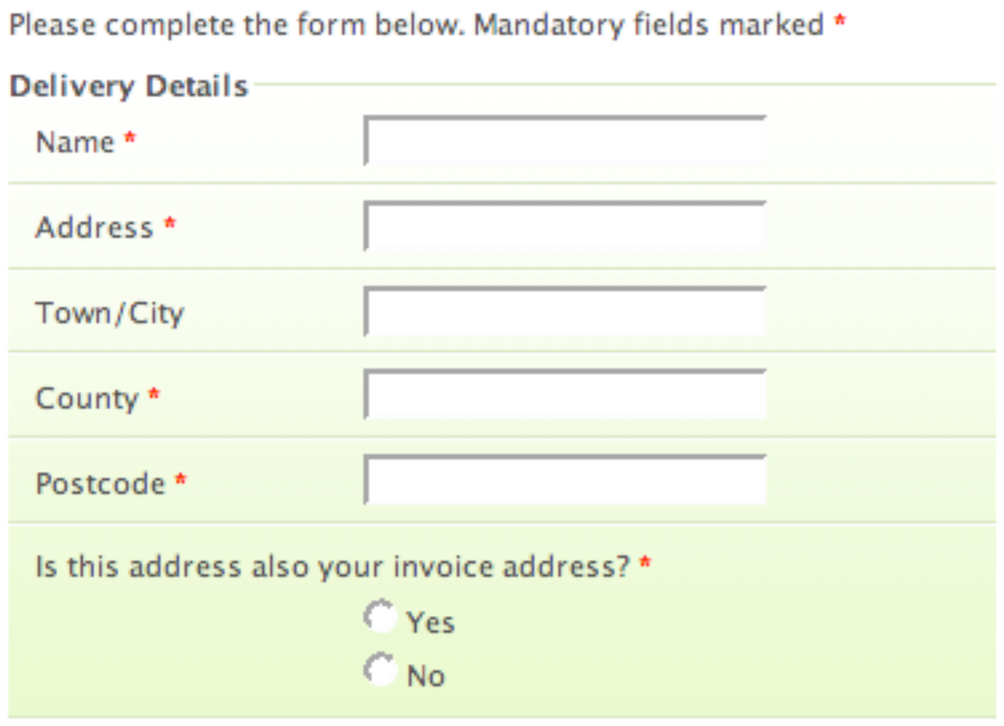
Student's signature

Approved:
Senior Supervisor ____________________ Date __________

Graduate Program Committee ____________________ Date __________

After approval by the Graduate Program Committee, this form should be sent, together with supporting documentation to the Supervisor of Graduate Studies in the Office of the Registrar.

Approved:
Senate Graduate Studies Committee ____________________ Date __________



Please complete the form below. Mandatory fields marked *

Delivery Details

Name *	
Address *	
Town/City	
County *	
Postcode *	

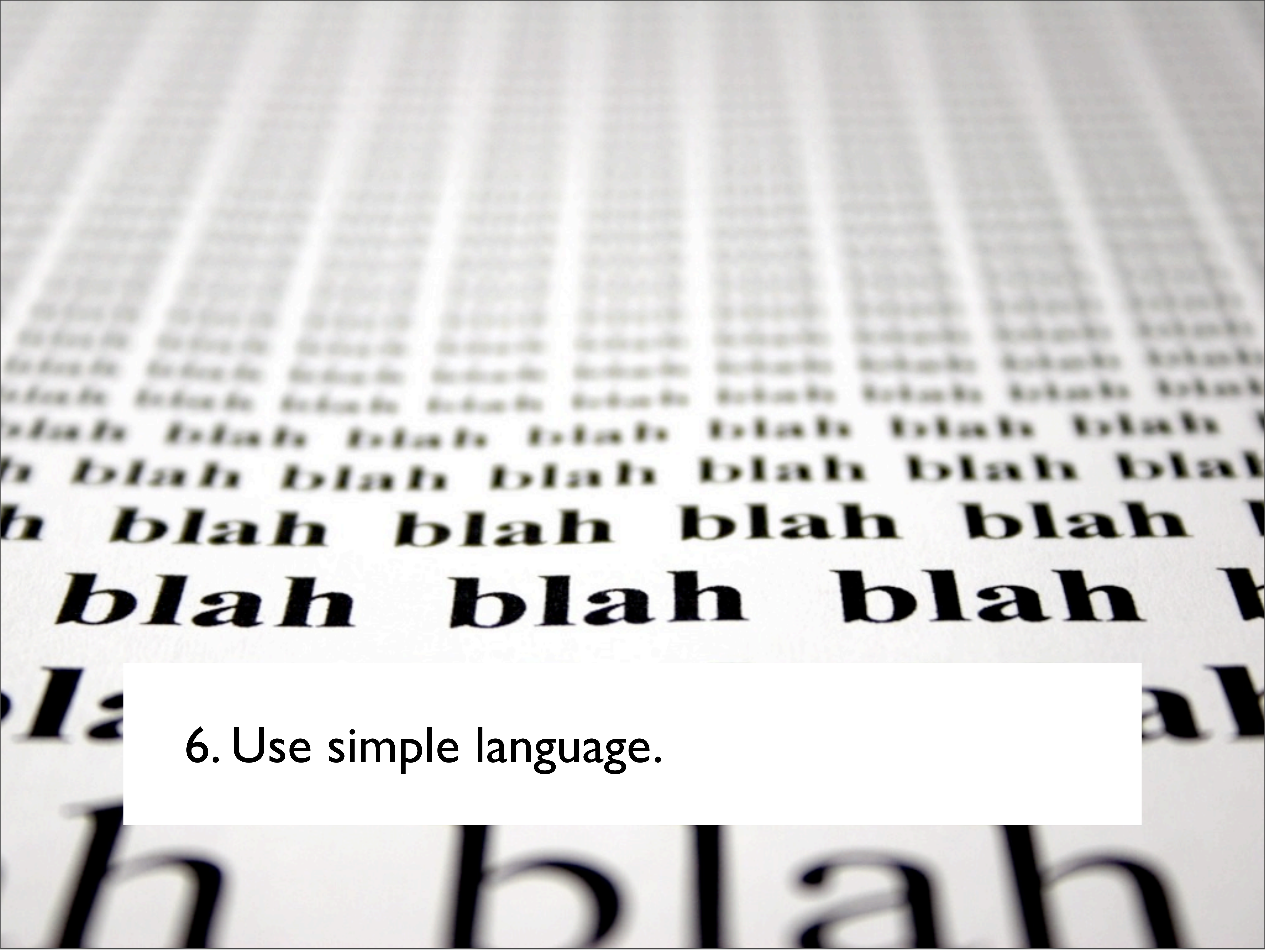
Is this address also your invoice address? *

Yes

No

5. Define templates for requirements description.

-They make the requirement easier to read once the reader understands the format.
-They make gathering and writing easier because they provide a checklist of things to include.
-This is similar to a standard document structure but it deals with individual requirements only.



6. Use simple language.

-Simple language is easier to read and understand. You are writing a technical document and ornate writing makes it more difficult for the reader.
-Be consistent in your description. If you give a name to something then use the same name all the time.
-Short sentences are easier to understand. Use one idea per sentence.
-Don’t use structures which add nothing to the sentence (eg. However, ...).

7. Organize requirements inspections.

-A group of people should meet and systematically check for problems in the requirements document. The requirements engineer (systems analyst) presents each requirement. The group comments about problems or concerns. The comments are recorded so the problems can be addressed later.
-It should be a formal meeting, with a Chairperson and an agenda.

8. Define a validation checklist.

-This helps the people validating the requirements identify if something is incorrect.
-Questions include questions such as:
  -are the requirements complete
  -do you understand what the requirements all mean
  -are they ambiguous
  -are there any contradictions

9. Use checklists of requirements problems.

-Identify problems which you have experienced in the past and look for them.
-Things that you could look for include:
   -design choices being listed as requirements
   -combining two requirements into one
   -are there unnecessary requirements
   -is it an ambiguous requirement
This is really suggesting that you should carefully inspect the requirements.

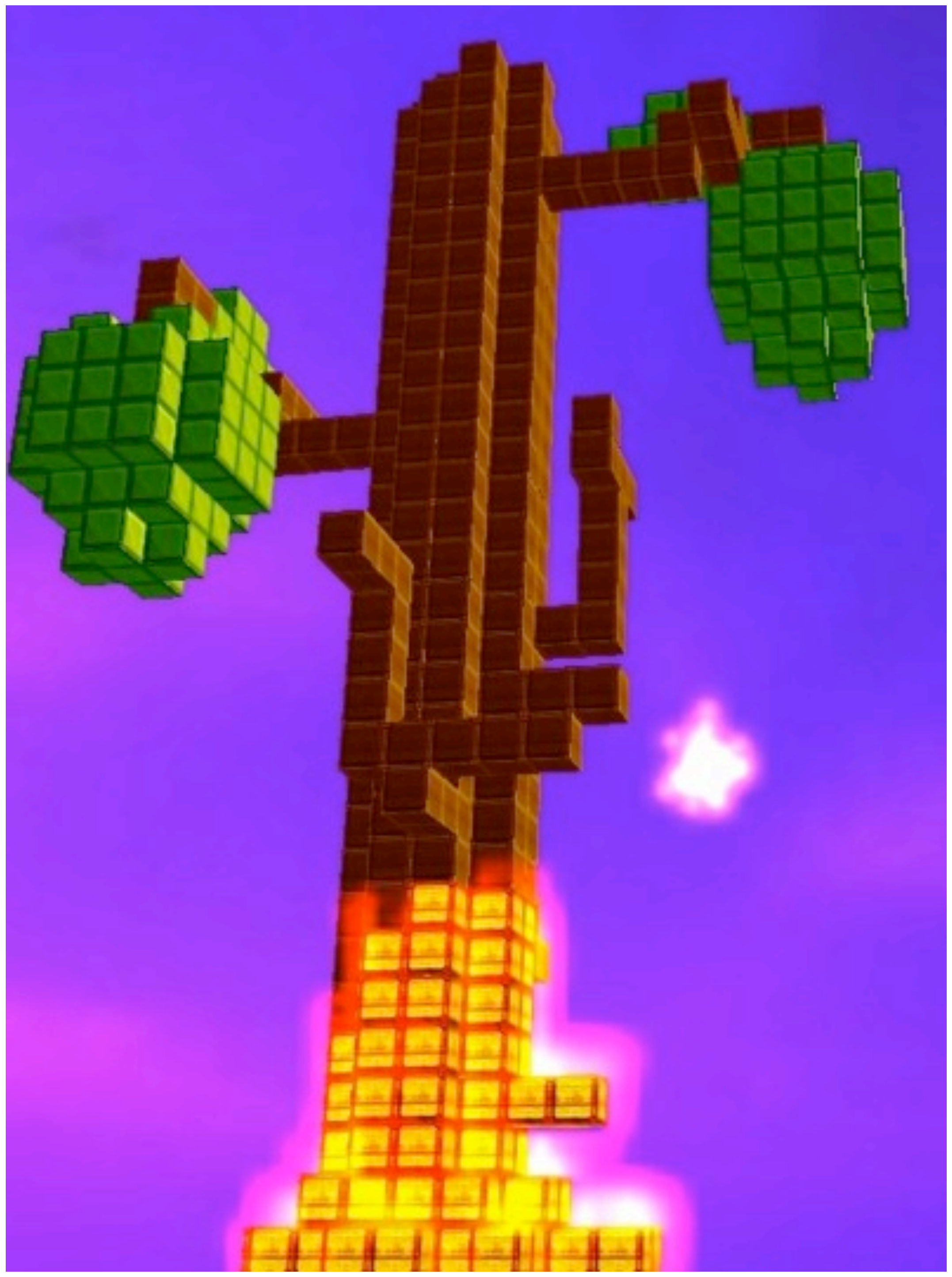
Conflict Resolution
Various facial expressions are used to communicate. Physical aggression is often avoided by giving just the right "look".
"You are starting to bother me"
"You are really bothering me"
"I've had enough and I'm going to do something about it!"

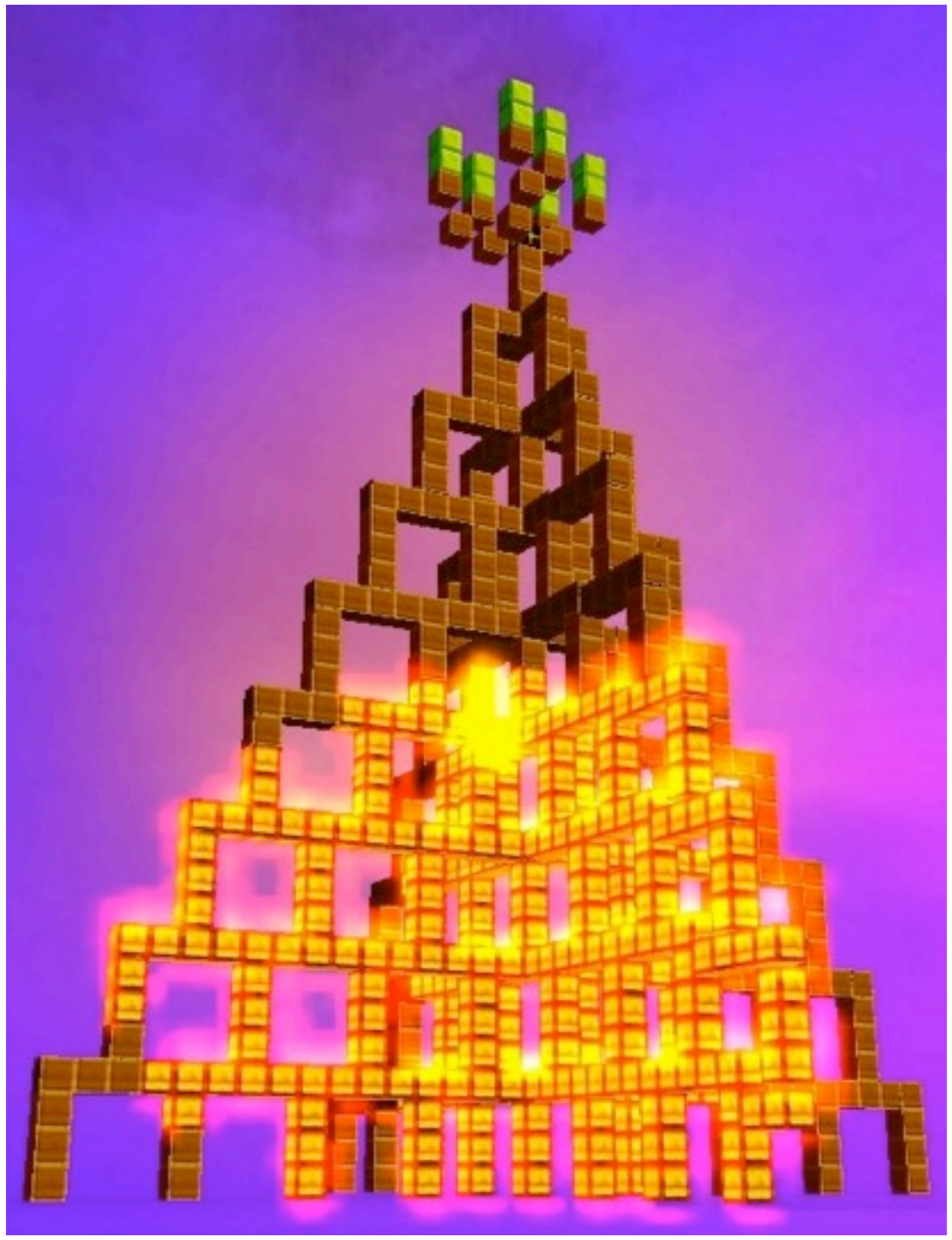
10. Plan for conflicts and their resolution.

-There will always be conflicts or omissions in the gathered requirements.
-Arrange meetings to resolve problems through negotiation. Don't assume you can solve the problem without input from the stakeholders.
-All stakeholders may be involved.

- Requirements Engineering: A Good Practice Guide by Ian Sommerville and Pete Sawyer, Wiley and Sons, 1997.

Design Document for Fireball (Play with Fire)





- A Design Document for a video game

-Play with Fire (initially named Fireball) is a game written for the Playstation 2.
-The design document for the game is 16 pages of text and images.
-The objective of the game is to move a ball of fire around a three dimensional world and burn different objects.

Overview
Vision Statement
Branding Information

- The document contains extra information.

-This includes information for marketing the game.
-A Vision statement and Branding Choices are identified which will often not be found in a design document.
-An Overview reveals many requirements choices.

- Question: What are the Implicit Requirements for a game?

It should:
-be enjoyable to play for some target audience (casual to hard core)
-challenge the player without annoying them
-engage them sufficiently that they will buy the game
These are true for most games.

Fireball is a budget game for PS2.

- Overview Statements

-These give ideas as to the requirements which are specific to this game.
-The platform (PS2) is a design choice but it sounds like it was made early in the design process. Perhaps before any requirements were considered.
-The budget sale price suggests that the game wont have a huge development budget.

The player controls a ball of fire, and traverses a landscape made of blocks of different materials. As the player sets fire to these blocks, they grow hotter, and can set fire to more and more different types of blocks. The fireball the player controls can also rise up in height and the hotter the player gets, the higher they can jump in this fashion.

- Overview

-This is a short and concise description of how the game is played.
-The basic gameplay for the entire game is summed up in this paragraph.
-Try to achieve this level of concise description.

On each field (level) the player has an ultimate goal of igniting the torch (brazier) and thus clearing the field – but the torch is generally positioned at a high point and out of reach. The player must use a combination of platform skills and dynamic environmental features (for instance, by burning the supports under the torch down to the ground) in order to clear the field.

- Overview

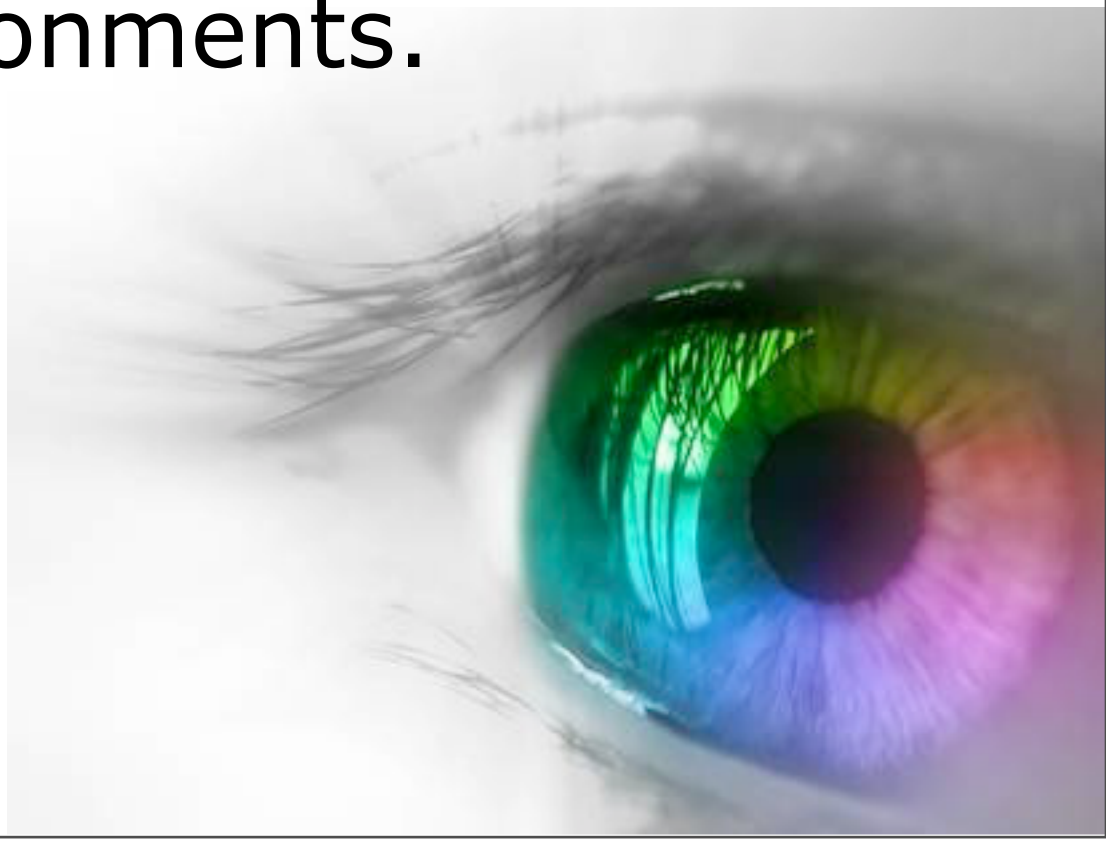
–The goal of the game and more detailed gameplay description can be summed in a second paragraph.

Simple, clean cut graphics and controls combine to give an easy to learn but engaging play experience.

- Overview

-One more sentence sums the graphical appearance, difficulty of the game, and user experience.

Effortless play originating from a simple control scheme.
Unique experience – the only game to be based around setting fires.
Varied solutions to the mini-puzzles as the player works out the best places to start fires.
Exploration of small environments.



- Vision Statement

–Provides guidance for the developer as to what kind of experience the user should have while playing the game (easy to play), the type of game (puzzle), and the scope (small environments).

Avatar concerns the player's ability to negotiate the landscape.
Temperature concerns the ignition of blocks in the playing field, and how fire spreads between blocks.
Gravity concerns the collapse of objects and blocks in the playing field as a result of fires.

- Game Subsystems

-These describe the things which will be controlled by the software.
-They are the components which must be developed.
-In other types of software they can involve things like storing data, using a network, performing calculations, etc.

The player's Avatar is a glowing ball of fire, considered to be 1 unit in diameter. The player's abilities are as follows:

Move around the environment. The player turns left and right, and pushes forward to move (relative controls).

- Each subsystem is described in detail.

-This is so the developers have enough information to implement them.
-The size of the player is described (1 unit).
-The types of motion and how they are implemented (relative to controls)

Jump up to a (relative) height determined by the heat of the ball. The characteristics of this jump are that the player rises rapidly up to their maximum (relative) height, and then slowly descend.

- Subsystems continued.

-Movement requirements are further explained with relative motion selected over absolute motion.
-Some terms which will need to be clarified at a later time (slowly).

Burning blocks is achieved simply by pushing into them. If the player is just hot enough to ignite a block, they will need to push into the block for a short while to start a fire – but if they are considerably hotter, fires will start just by them passing by.

...

These are all the player's abilities.

- Subsystems continued.

-The method of interaction with objects in the world is explained (burning).
-The final sentence makes it clear that nothing else needs to be considered.

The basis of fire starting rests in a simple system of temperature based upon colours. The avatar increases in heat permanently when it touches a burning block that is hotter than the avatar's current temperature.

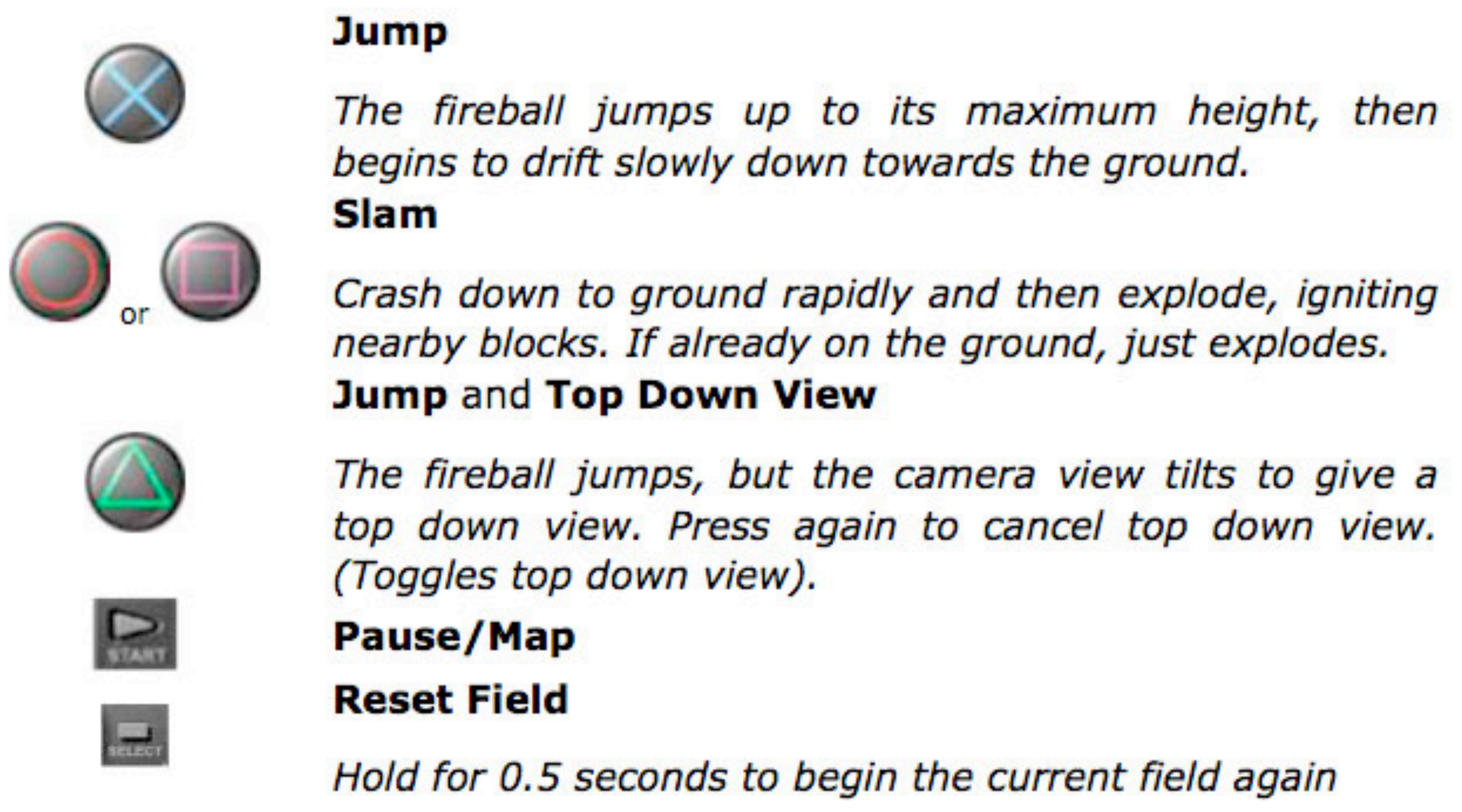
- The Temperature Subsystem

-It is explained with the same concise detail as the motion subsystem had been.

Colour	Jump Height	Description
1. Yellow	+2 Units	Yellow flames; bunsen burner style
2. Orange	+4 Units	Bright orange flames.
3. Red	+8 Units	Glowing red with heat haze.
4. Blue	+12 Units	Blue-white flame, like a blowtorch.
5. White	+16 Units	Bright white glow – very bright.

- The Temperature Subsystem

-When it is appropriate the data can be displayed in a table.
-More graphical information is described in the table. This might considered out of place but it is related to temperature so perhaps it is in the correct location. Given the short length of the document this isn't a large concern.



Jump

The fireball jumps up to its maximum height, then begins to drift slowly down towards the ground.

Slam

Crash down to ground rapidly and then explode, igniting nearby blocks. If already on the ground, just explodes.

Jump and **Top Down View**

The fireball jumps, but the camera view tilts to give a top down view. Press again to cancel top down view. (Toggles top down view).

Pause/Map

Reset Field

Hold for 0.5 seconds to begin the current field again

- User Interface

-Since this program was intended to run on a game console the user interface is defined by the controller. Other systems can have more complex control options which can make the decisions regarding the interface more difficult.

The player’s goal is always to move, burn and melt their way around the environment in order to reach the torch – a symbolic brazier item – which they ignite on contact.

Advanced players will attempt to clear the playing field in the shortest play time, and/or cause the biggest Chain – achieved by having large number of blocks burning at the same time.

- The Player’s Goal

-In other types of software this would be a description of what the user could expect to achieve using the software. It would let the reader of the design document know for what purpose the software was written.
-Instructions regarding more experienced users can be included. Not all users are novices. You should describe how all users could use the software.

3.1 Components
3.1.1 Blocks
The environment is entirely constructed out of 1 unit cubes (actually about 4 m per side, therefore 1 unit = 4 m). These cubes have different colours, and are textured to resemble specific materials.

3.1.2 Objects
Objects are simply clusters of blocks. For instance, a vertical column of ten blocks is an object. Four such columns with a flat plane of blocks across the top is a 'Table' object.
Clearly, because objects are made of cubic blocks, they are abstract in nature, but the player will still be able to make out what these objects represent.

- The Environment

-In more general terms this means the components of the system which the user will interact with. In other software packages it can involve screens or windows which provide access to different functions. This lets the user know which operations and associated windows they can access.
-Notice that sections are numbered for easy reference.

3.2. Gravity
Gravity always pulls blocks and objects downwards. The gravity value is 10 units per second per second, with a terminal velocity of 5 units per second.

3.4. Burning
3.4.1 Ignition
The temperature at any point in the game field (for the purposes of checking for ignition) is based upon the temperatures of the surrounding blocks. The process of determining if any given block ignites is as follows:
· Check for neighbouring blocks of the same material that are on fire, and have been burning for at least as long as the ignition time. If they exist, the current block catches fire.
...

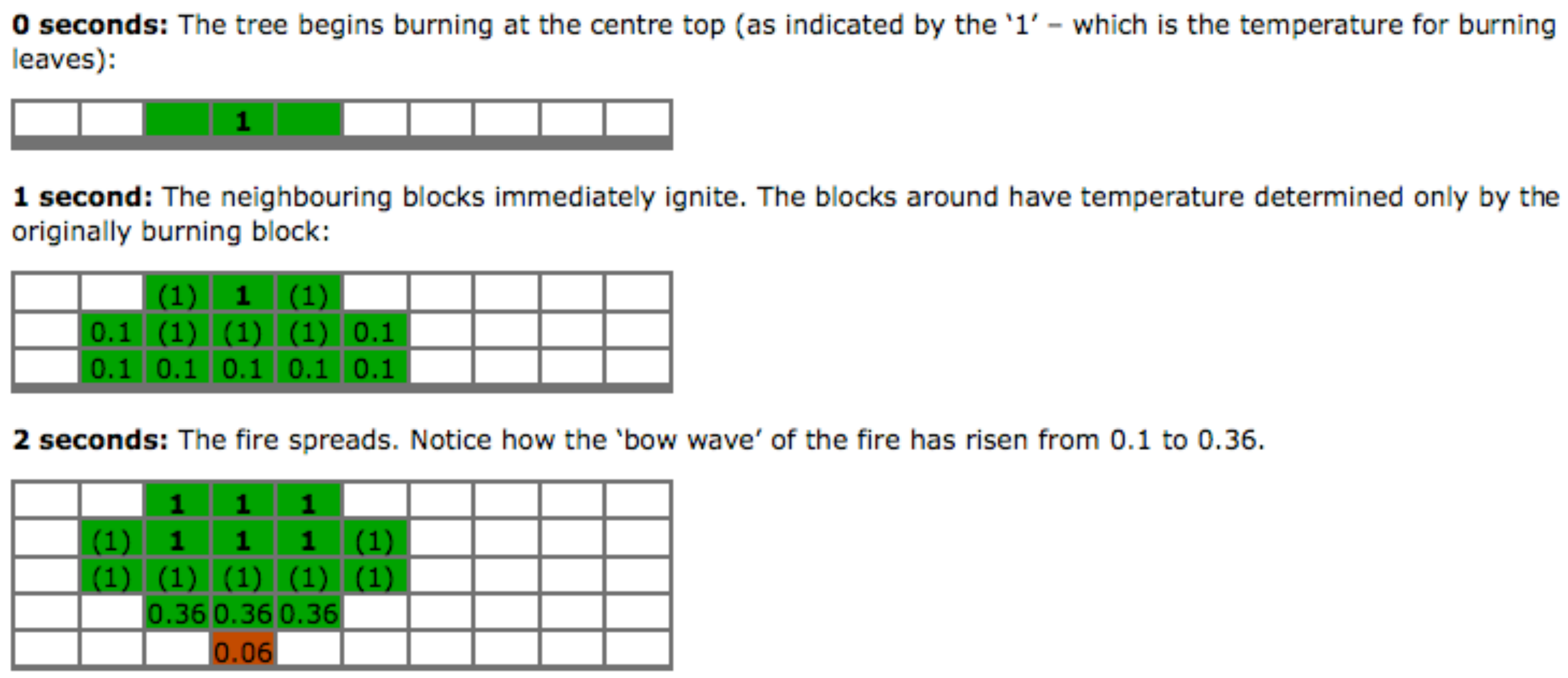
- Complex Behaviours

-The most complex behaviours in the game are how gravity operates and how components burn. They have their own sections which contain detailed descriptions of the operations.
-Algorithms are described in easy to understand terms. Mathematics are used when necessary.
-Initial values for parameters are specified.

Note
The three parametric values above are the temperature radiation coefficients (TRC1, TRC2 and TRC3). The values given should be considered default values – tweaking will be required.

- Unknown Values

–Values which are not known during the system can be tested are clearly indicated. If something is not known then it should be carefully identified.



0 seconds: The tree begins burning at the centre top (as indicated by the '1' – which is the temperature for burning leaves):

		🟩	1	🟩					

1 second: The neighbouring blocks immediately ignite. The blocks around have temperature determined only by the originally burning block:

		(1)	1	(1)					
	0.1	(1)	(1)	(1)	0.1				
	0.1	0.1	0.1	0.1	0.1				

2 seconds: The fire spreads. Notice how the 'bow wave' of the fire has risen from 0.1 to 0.36.

		1	1	1					
	(1)	1	1	1	(1)				
	(1)	(1)	(1)	(1)	(1)				
		0.36	0.36	0.36					
			0.06						

- Complex Behaviour Diagrams

-Diagrams can be used to represent complex behaviours. If diagrams will help to illustrate a complex operation then they should be included.
-In this case it illustrates how an object will burn in the game.

3.4.2 Burning Out
After the Burn Time for a block has expired, the block is removed from the world completely.
Any blocks or objects that were resting upon that block then fall down. Thus the shape of the world changes over time as more objects and blocks burn to nothing.

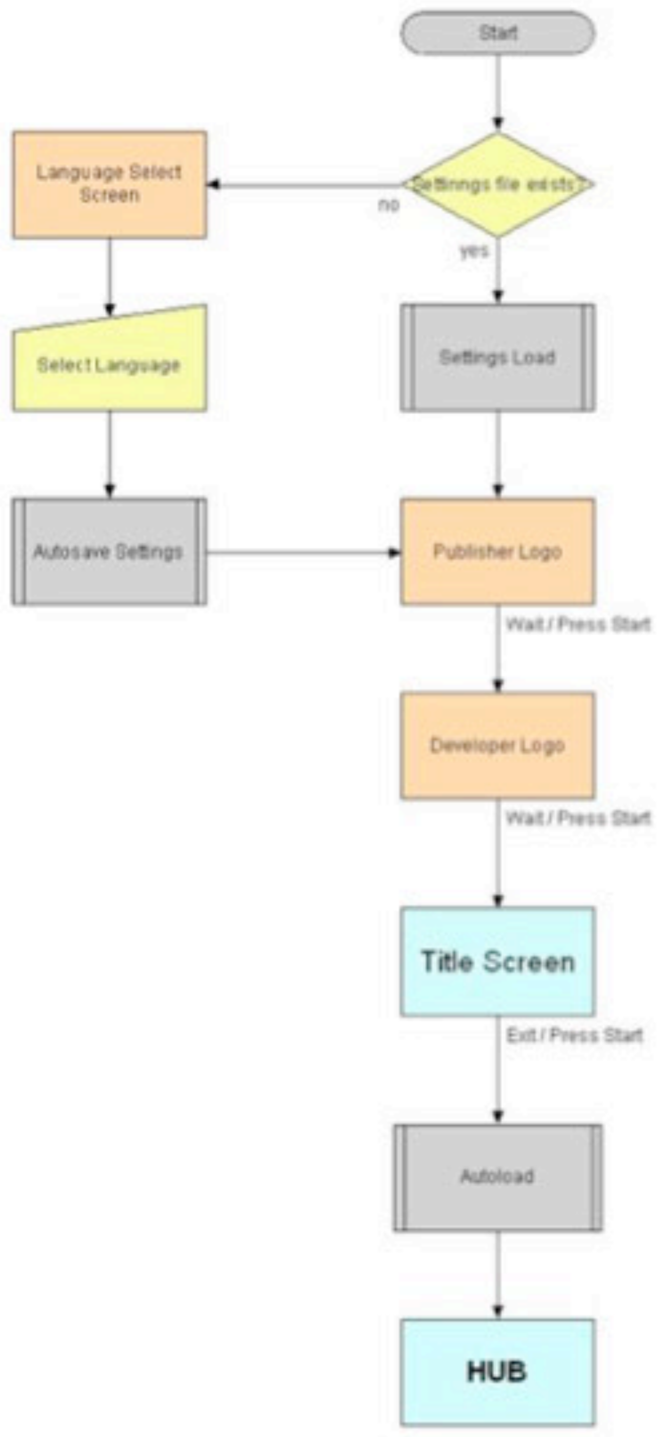
- Behaviour Changes in the System

-If the system's behaviour changes due to the user's actions then the new behaviour should be identified.
-This can include menu items being made unavailable (greyed out) or made available, windows can pop-up, or the operations which keys perform can change (with modal interfaces).

4. Structure
4.1. Overview
The game is divided into **field lists** or **Quests** which consist (in general) of six fields. (The term Quest is used in game documentation, but in the design document the term Field List is used). The player must play through all six fields to complete a field list. However, they can exit their current set of six fields by pressing start and choosing exit from a pause menu.

- Navigating the Interface

-If the system has a sequence of operations which must be performed in a given order then this should be explained.



Start Up
State Diagram
for Fire ball

- System Operation Sequences

-State diagrams can be used to illustrate the sequence in which operations are performed and how different operations are selected.
-Include a description of the states so the reader understands what they all mean.
-Don’t forget a starting and ending state.
-Break the diagram into parts if it is too large to comfortably fit on one page.

6.4.2 Field Filenames
The following filename format is used for fields:

XXXXXXXXXX.XX.XXXX.fsf

The first ten characters are an informal description of the level. The next two characters represent the Stage (see below). The next four characters represent the difficulty, either:

· Easy
· Mid
· Hard
· Hell

So an example field filename might be:
Smallhouse.LW.easy.fsf
This allows anyone building field lists to know that the field designer intends this particular field to be easy to complete.

- Naming Conventions

–Specify naming conventions for resources such as files and databases.
–If the name encodes some information then explain how this operates.

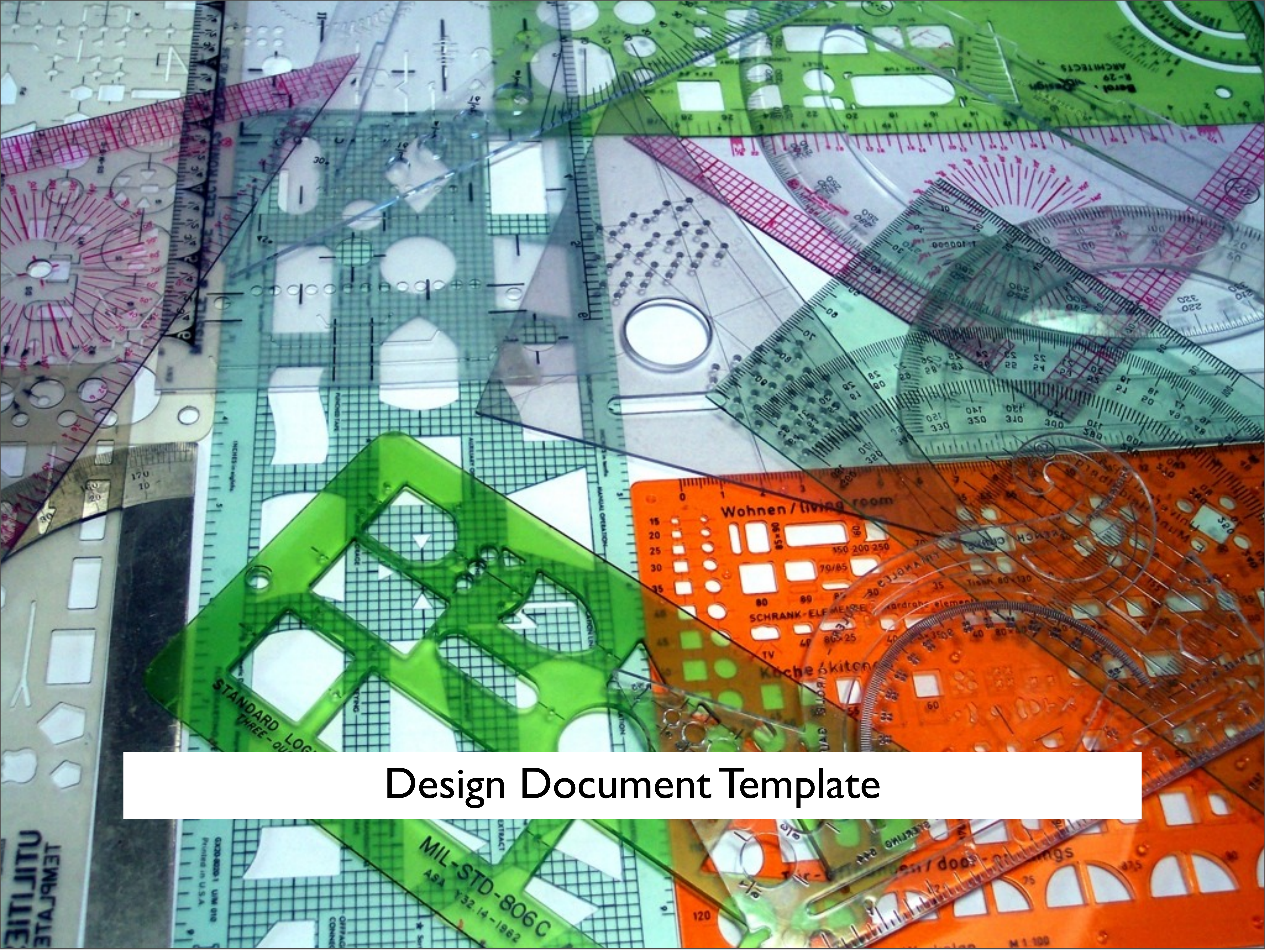
8.Delta Log
Version 1.0
Orange squares represent sales primarily from hardcore gamers, whilst blue squares represent sales to a mass market audience.
Transition from H2 to C3 or C2 (dotted arrow) is most likely be a male H2 showing the game to friends, and therefore uptake from these vectors will be lower Conversely, transitions from H3 to C3 or C2 are more likely to be a friend making a recommendation for purchase, and therefore should help drive sales.Core design incorporated from Concept version 1.0
Version 1.05
Changed references that read ‘Combos’ to saying ‘Chains’ (i.e. renamed ‘Combos’ to ‘Chains’). Added a section about Chains. Added a Template section giving advice to field designers.
Version 1.10
Added the concept of ignition time. A block has to have been on fire for a (short) amount of time (known as the ignition time) before it can set fire to other blocks. Ignition times added to the block data table. Added reference to the parameters for the fire effects. (Appendix I in this document). Target Audience titled ‘Appendix II’.

- Change Logs

-These show the changes which have been made to the document over time. It is a list of changes and the reasons for each change.
-It is important to track changes and retain the information for the future.
-This can be an appendix in a paper document. In an electronic document it is usually a history of changes that can be recalled.

References

- http://www.gamasutra.com/view/feature/1709/design_document_play_with_fire.php



Design Document Template

A sample structure for a design document or design specification from R. Pressman.

Document Structure

- Introduction
- Data Design
- Architecture and Component Design
- User Interface Design
- Restrictions, Limitations, Constraints
- Testing Issues
- Supplemental Information

Each of these is described later.

Introduction

- Goals and Objectives
- Scope
- Context
- Major Constraints

–Scope –A description of the software is presented. Major inputs, processing functionality, and outputs are described without regard to implementation detail.
–Context –The software is placed in a business or product line context.
–Constraints –Any business or product line constraints that will impact he manner in which the software is to be specified, designed, implemented or tested are noted here.

Data Design

- Internal software data structure
- Global data structure
- Temporary data structure
- Database description

-A description of all data structures including internal, global, and temporary data structures.
-Internal -Data structures that are passed among components.
-Global -Available to major portions of the architecture.
-Temporary -Files created for interim use are described.
-Database -Database(s) created as part of the application.

Architectural and Component Design

- Program Structure
- Description of Components and Sub-Components
- Interface Description

-Program Structure -Architecture diagrams and alternatives to the selected architecture.
-Description - detailed description of each software component contained within the architecture is presented. This includes a description of its processing, interface, algorithm, restrictions, data structures, and constraints for each component.
-Interface Description -interfaces to other computers, machines, and humans are described.

User Interface Design

- Description of the user interface
- Interface design rules
- Components available
- User Interface Development System

-Description -Screen images and a description of objects and their actions.
-Design Rules -Conventions and standards used for designing the interface.
-Components -GUI components are listed.
-Development System -system tools and library described.

Restrictions, limitations, and constraints

- Special design issues which impact the design or implementation of the software are noted here.

Testing Issues

- Classes of Tests
- Expected software response
- Performance bounds
- Identification of critical components

Test strategy and preliminary test case specification are presented in this section.
-Classes of tests -types of tests to be conducted.
-Expected response -expected results.
-Performance bounds -Special performance requirements.
-Critical components -those which demand particular attention during testing.

Appendices

- Requirements traceability matrix
- Packaging and installation issues
- Design metrics to be used
- Supplementary information

Presents information that supplements the design specification.
-Traceability Matrix -traces stated components and data structures to software requirements.
-Packaging and installation -special concerns.

Reference

- Roger S. Pressman and Associates, Inc.
- http://www.rspa.com/docs/Designspec.html

Technical Reviews

- Technical Reviews can be applied to requirements, designs, or to code.

-Reviews are used to improve the quality of the system. They can involve requirements, design, or code.

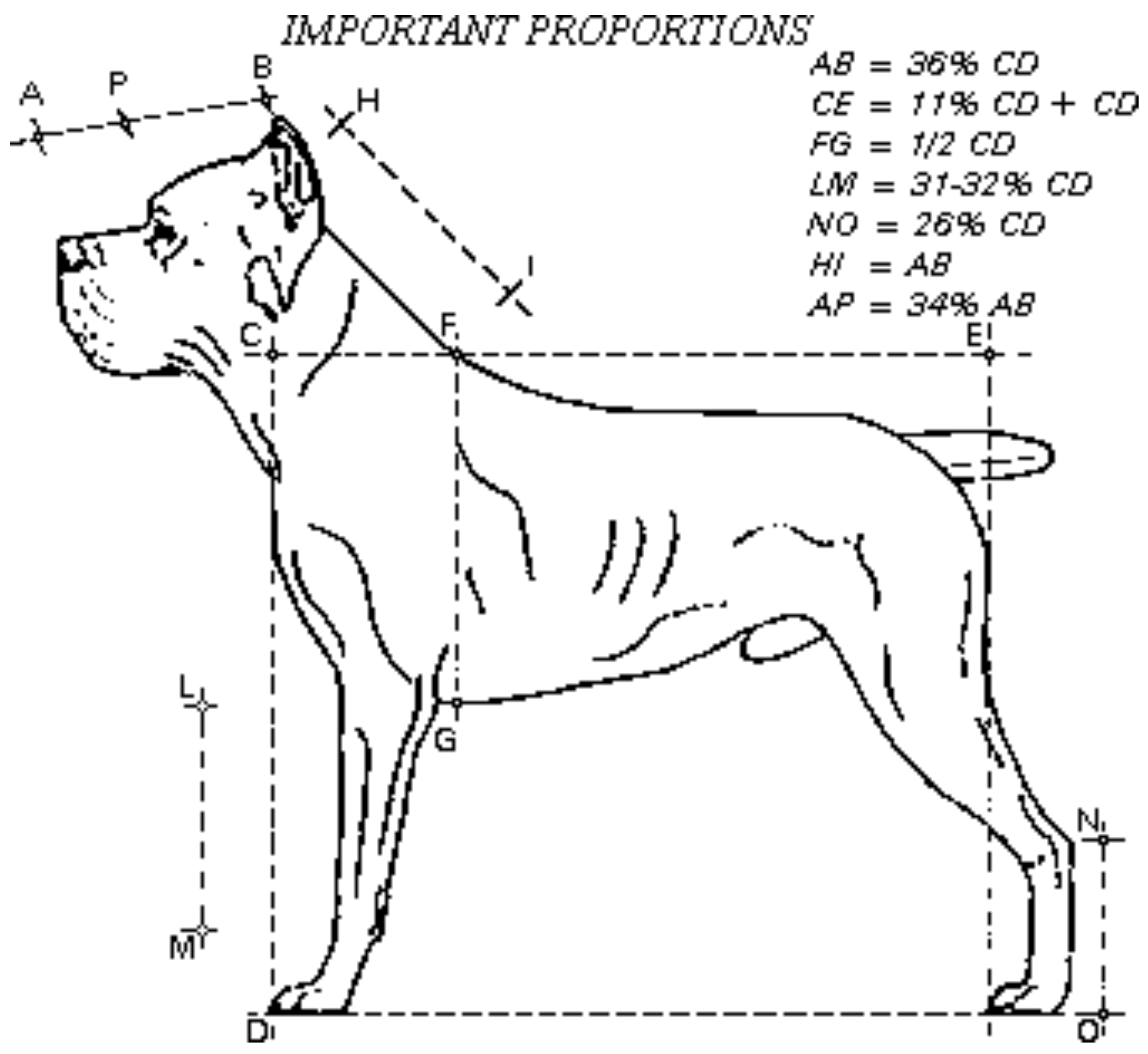
Objectives of a Review

- Uncover errors.

-Errors in code include functional, logic, or implementation in the software.
-Errors in design include logic of the system, functions performed the system or by components which the system accesses.

- To verify requirements.

-To ensure that the software being reviewed meets its requirements.



- To ensure the software meets a standard.

-This ensures that the system is being designed as expected. There are no surprises in the development due to standards not being followed.
-Standards could be internal (the developers, the client) or external (government).
-These standards could include tools, formatting of work, regulatory bodies, interaction with other systems.

- For system uniformity.

-To ensure that different parts of the system have a uniform development style.
-For example, structure of program components or the language used in design or requirements should be uniform.

- To make projects more manageable.

-To provide a point where managers can influence the design and development of the system.
-Goals of this include the previous points. The idea is that a manager has the opportunity to affect the development so they can manage it.

- Training

–The technical review meeting serves as an opportunity for junior developers to observe different approaches to analysis, design, and construction.

- Continuity of the team.

-The review process ensures that number of people who are familiar with parts of the system is larger than it would normally be. This ensures that if people leave the team there will be others who understand the system they were working on.

- Reviews can take several forms.

-They can be walkthroughs, inspections or others small group assessments.
-A review is conducted as a meeting. It must be planned (have an agenda), managed (have a chair), and attended by the appropriate developers.

- Every review should follow some rules.

-Three to five people should be involved. Too many and it becomes difficult to contribute.
-Each person should prepare but should require no more than two hours to prepare. If it requires more than this then it is asking for too much to be accomplished.
-The meeting should be less than two hours. More time means you have too large a goal for one meeting.

- Focus on a specific and small part of the system.

-Do not review an entire design. Focus on a component or collection of small component.
-A narrow focus is more likely to uncover errors.

- Initiating a review.

–Once a components (e.g. partial requirement specification, component design, or section of source code) is ready for review its developer requests that the project leader schedule a review.
–The person in charge of the review determines if the component is ready. If it is they schedule the review and send the materials for the review to those who will attend.

- Review attendees.

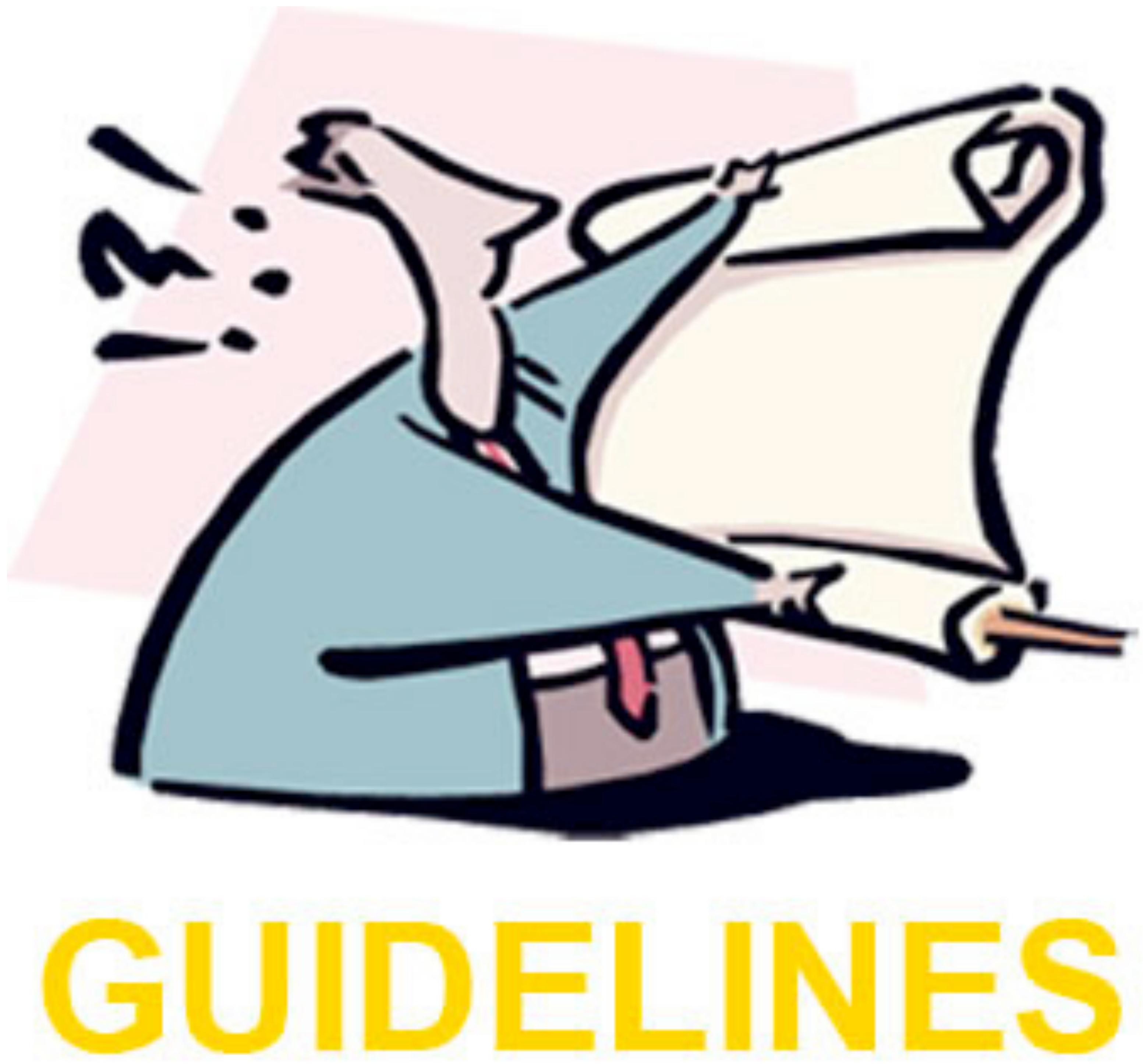
-The meeting chair (review leader), the reviewers, and the developer of the item being reviewed.
-One reviewer takes notes of important issues raised during the review.

- Review process.

-The developer gives a brief introduction to the component.
-The developer then walks through (describes) the component.
-Reviewers raise issues based on their preparation and the developer's description.
-It can be useful to have someone other than the developer perform the walk through. They will provide a literal interpretation of the component which will make errors more obvious.

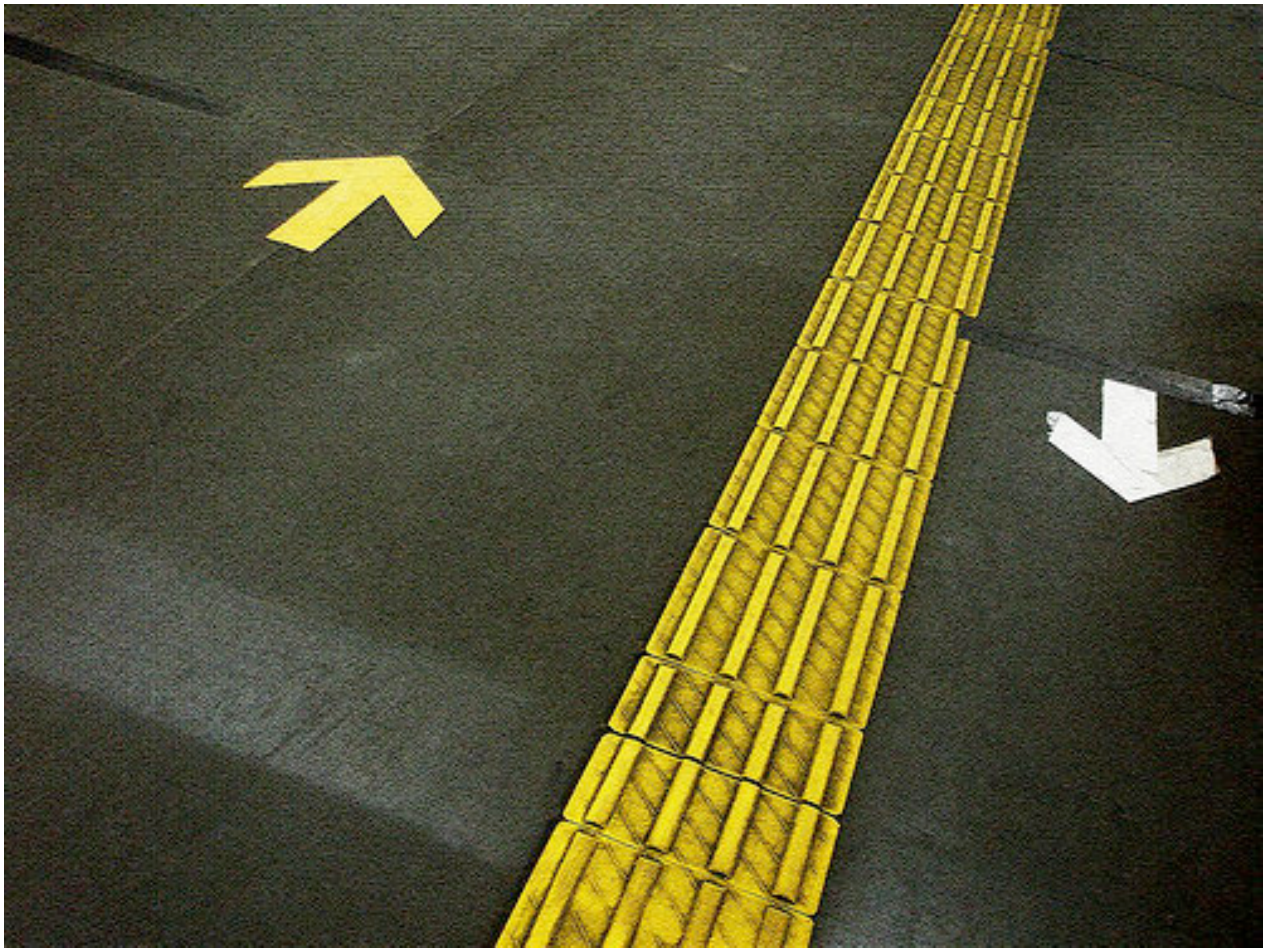
- At the end of the review.

-The committee decide whether to either accept the component as it is or with minor changes, or to reject it due to severe errors.
-Rejected components must be reviewed once errors have been corrected.
-Someone is given the responsibility to ensure that the changes are implemented.



- Guidelines for reviews.

–Review the product, not the producer. The objective is to be constructive and not to embarrass the developer. The process is said to be egoless because it is directed towards the product.
–Set and agenda and maintain it. Meetings can drift off topic and waste a lot of time. Do not be afraid to redirect the meeting if it is off topic.



- Guidelines for reviews.

-Limit debate and rebuttal. Not all issues can be resolved during the review meeting. Do not waste time in the meeting trying to resolve difficult issues. The meeting is there to identify potential problems not to solve them. If there is not a quick resolution then the issue should be recorded and discussed out side of the meeting.

- Guidelines for reviews.

–Make notes so the issues are properly remembered. Notes can be displayed during the meeting so meeting reviewers can comment on the wording and priorities.

- Sample driven reviews.

-Reviewing all parts of a design or a program is probably the best solution but can be impractical.
-The solution is to identify the components of the system with the largest number of faults and then focus the reviews on those items.

References

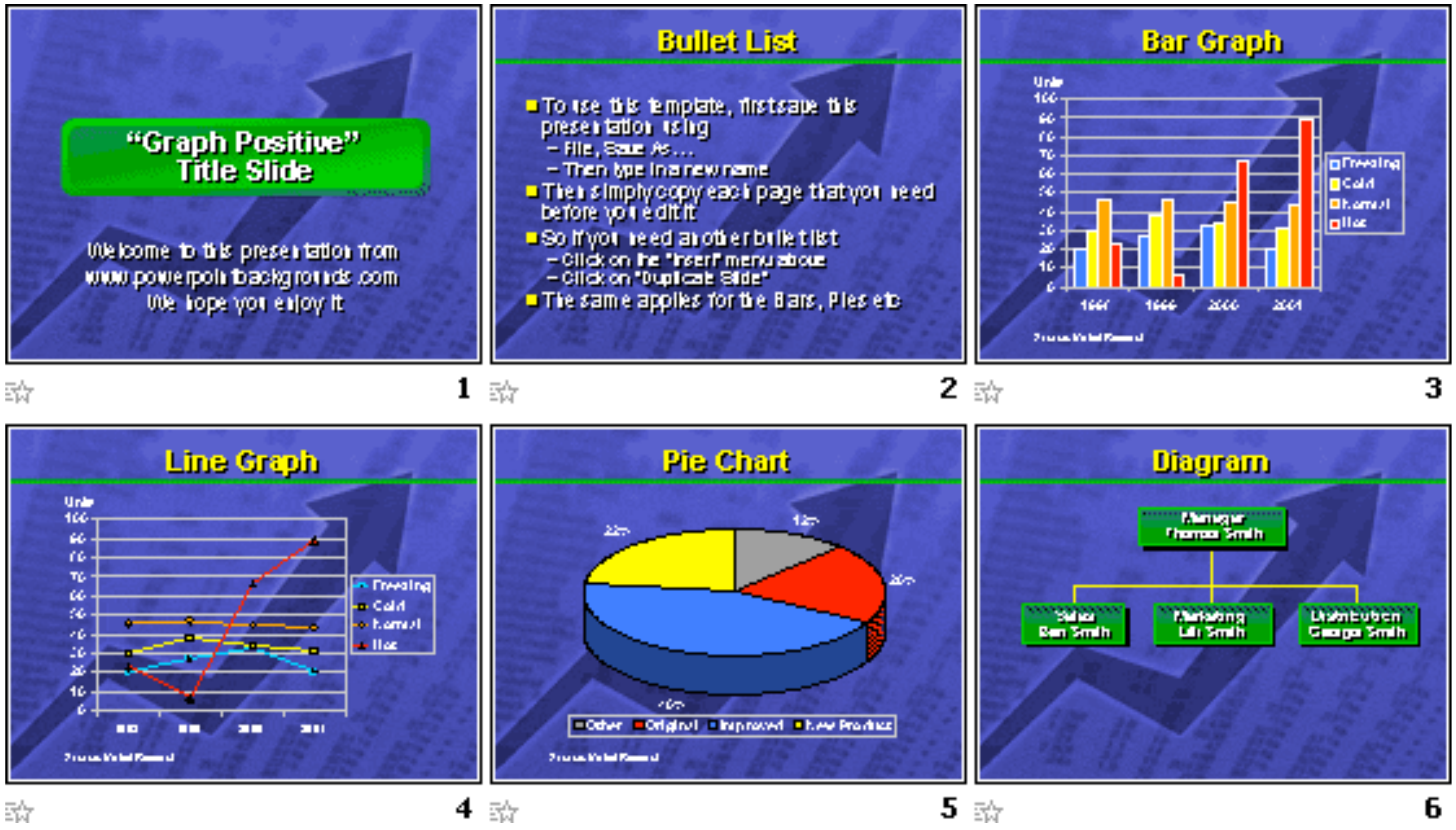
- Software Engineering: A Practitioner's Approach, sixth edition, by Roger S. Pressman, McGraw Hill, 2005.

AS YOU CAN CLEARLY SEE IN SLIDE 397...
www.dilbert.com scottadams@aol.com
GAAAAH!
Z
Z
© 2000 United Feature Syndicate, Inc.
"POWERPOINT" POISONING.

Presentations

- Two Approaches

-The two approaches to presentations we will consider are the traditional presentation and the more design oriented presentation.
-The traditional approach is often used for more complex technical talks but it is not a requirement.
-The design approach is more effective at engaging the audience.
-Both types can be effective but you need to decide which is appropriate for your audience. Traditional is easier to create.


"Graph Positive" Title Slide
Welcome to this presentation from www.powerpointbackgrounds.com We hope you enjoy it
Bullet List
To use this template, first save this presentation using
– File, Save As…
– Then type in a new name
Then simply copy each page that you need before you edit it
So if you need another bullet list
– Click on the "Insert" menu above
– Click on "Duplicate Slide"
The same applies for the Bars, Pies etc
Bar Graph
Line Graph
Freezing
Cold
Normal
Hot
Pie Chart
Other
Original
Improved
New Product
Diagram
Manager Thomas Smith
Sales Ben Smith
Marketing Lili Smith
Distribution George Smith
1
2
3
4
5
6

- The Traditional Presentation

-Slides contain a fair amount of text.
-There are normally three to seven points on a slide. More than this is too dense.
-All of the key points are listed.
-The text is a guide for the audience and the speaker. They shouldn't be able to get lost during the presentation because everything is listed.
-These are most common in business environments. In some cases too much flashy design may not appear professional. The large amount of text is seen as an accomplishment.

- The Traditional Presentation

-The presentation can become tiresome because of the quantity of information.
-The audience may spend more time reading the text instead of listening to the speaker.
-Is it appropriate for the slides to act as a guide or should the focus more on conveying the information?
-Background images are often pointless and distracting.
-Images are generally not important.

$$\begin{tikzcd}
\text{Introduction} \arrow[dr] & & \\
& \text{Body} \arrow[dr] & \\
& & \text{Conclusion}
\end{tikzcd}$$

- The Traditional Presentation

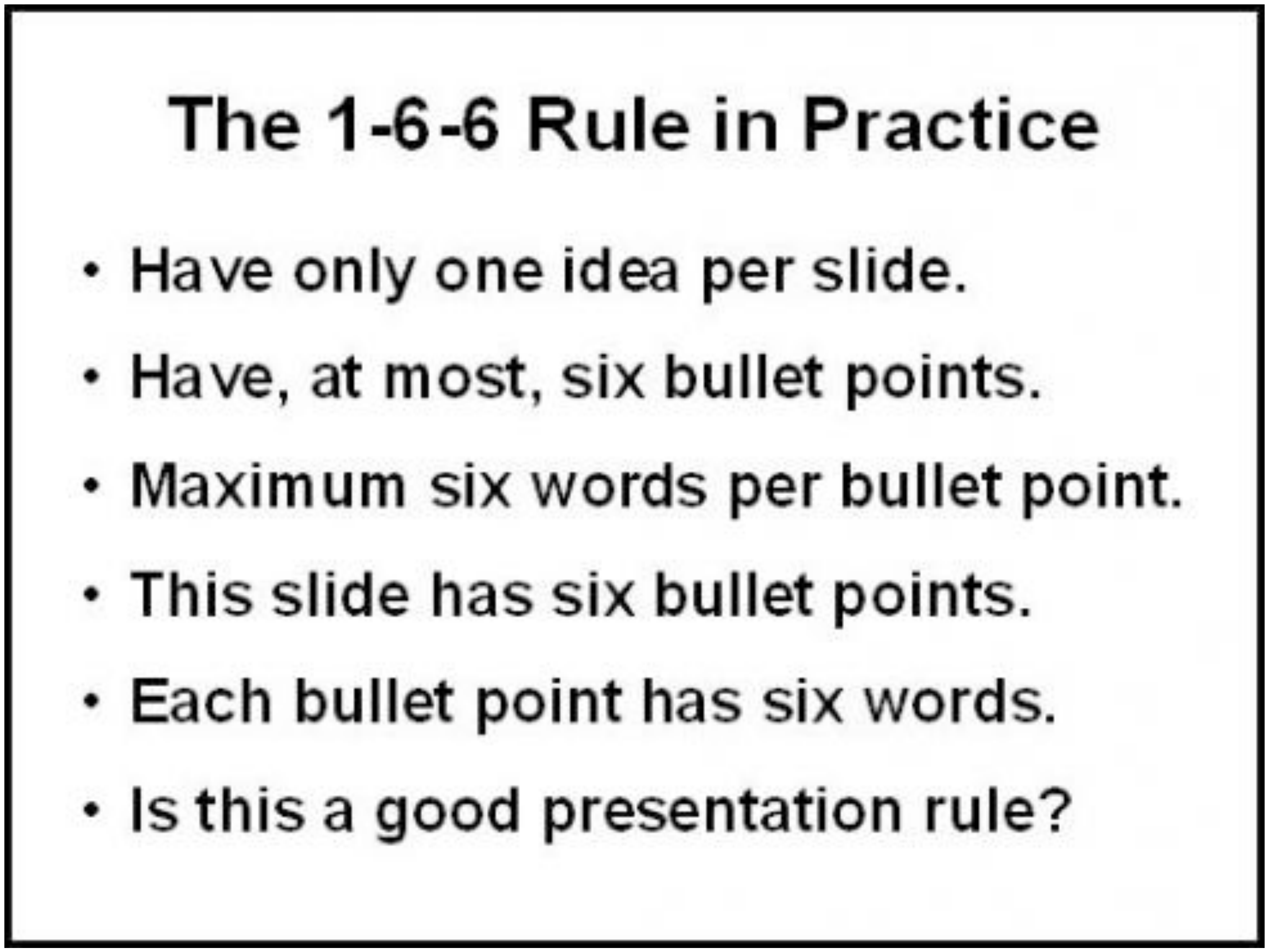
-There is normally an introduction, body, and conclusions.
-The introduction should explain to the audience why they should care about the presentation. You are not writing a mystery novel. Explain the importance at the start.
-The introduction can contain a list which lists all of the parts of the coming presentation. It can be appropriate to give this information to the audience but it is pointless to spend much time on it. This is often used to fill time and doesn't add much to the presentation.

- The Traditional Presentation

-The body is where all of the important details are presented. This should be the largest part.
-A common mistake is to provide too much background material and too little about the topic you are discussing. If the audience is familiar with the background then they don't need to hear you discuss it. Keep the background material short.
-The conclusion is often a restating of everything which has been presented. Don't dwell on the less important parts. Don't simply repeat what has been previously stated. Provide a summary, not a list of points.

- The Traditional Presentation

-The number of slides in the presentation should not be too large or too small.
-If you are advancing the slides too often it becomes distracting (say one a minute on average).
-The opposite is also true. If there is a long gap between slides then you should consider if there are enough slides.

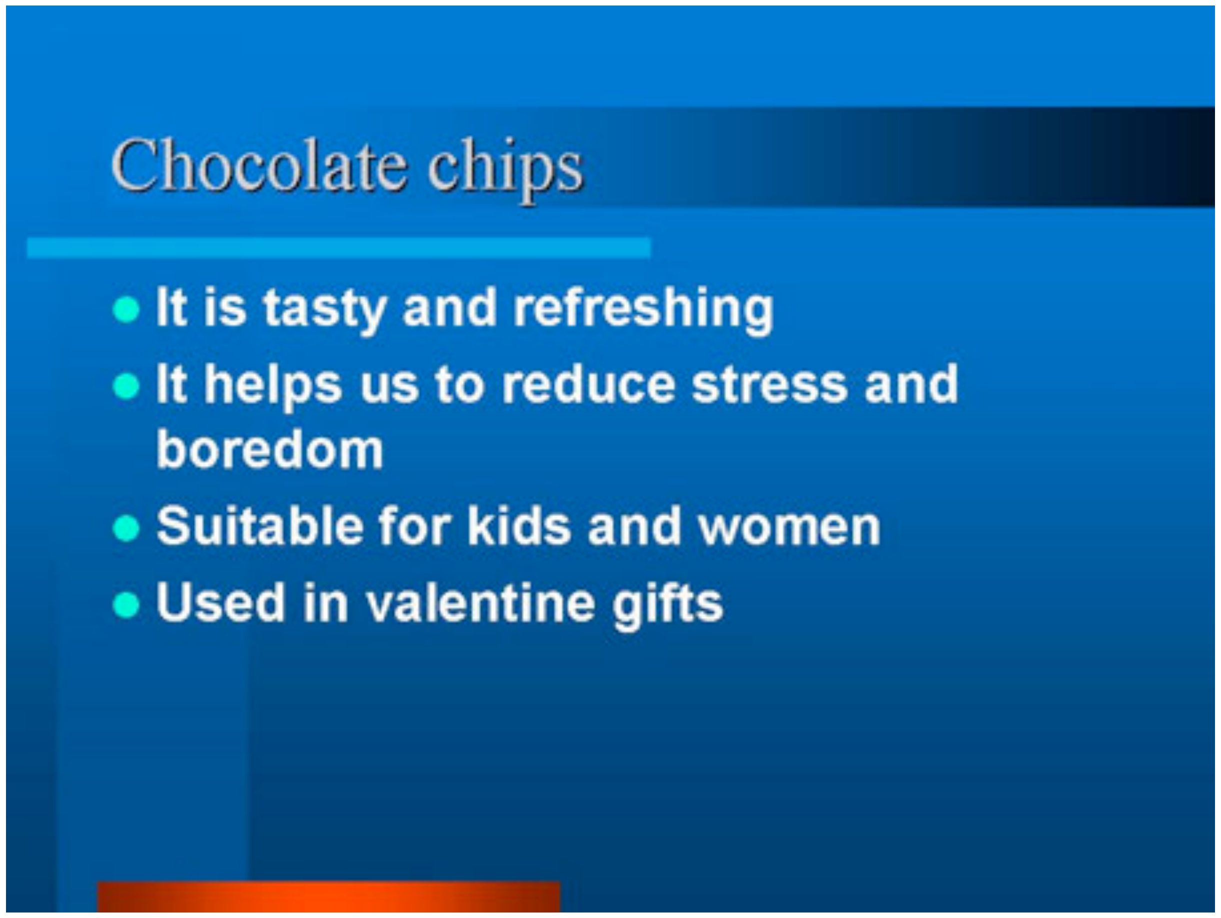


The 1-6-6 Rule in Practice

- Have only one idea per slide.
- Have, at most, six bullet points.
- Maximum six words per bullet point.
- This slide has six bullet points.
- Each bullet point has six words.
- Is this a good presentation rule?

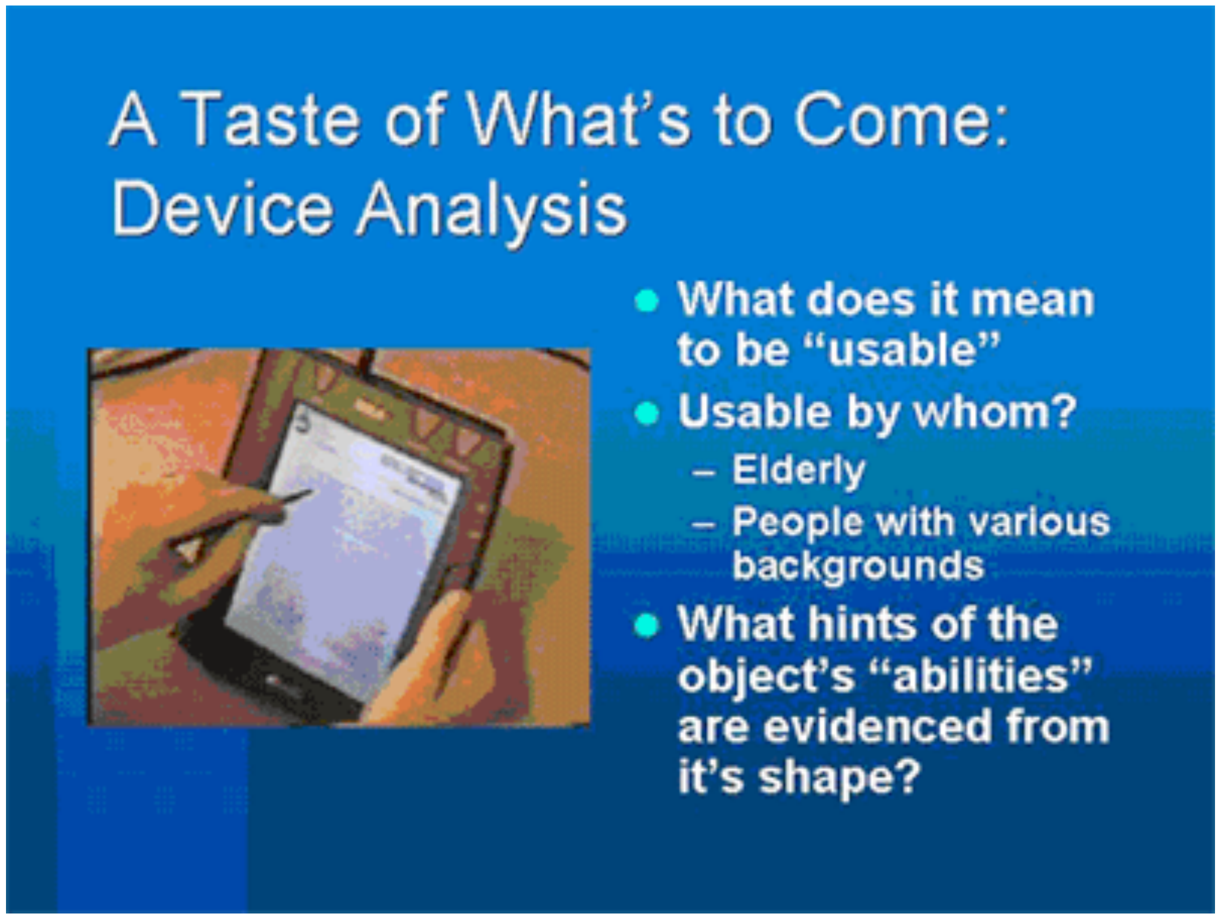
- The 1-6-6 or 1-7-7 Rule.

-This is a rule of thumb for traditional presentations.
-There should be one idea per slide, no more than six or seven lines of text, and no more than six or seven words per line.
-The idea is to not overwhelm the audience but this can lead to very dense, hard to read, and tiresome slides.



Chocolate chips

- It is tasty and refreshing
- It helps us to reduce stress and boredom
- Suitable for kids and women
- Used in valentine gifts

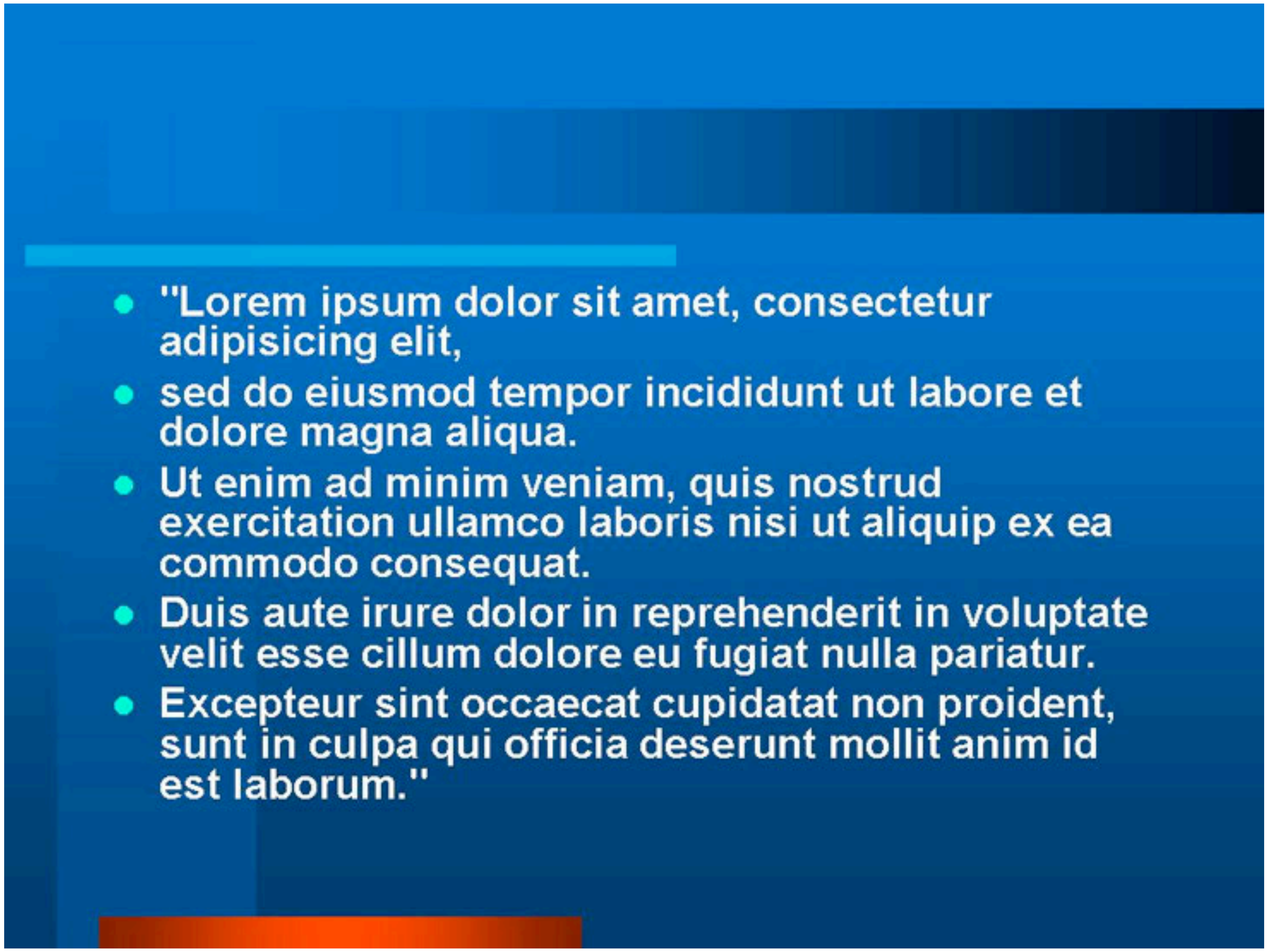


A Taste of What's to Come: Device Analysis

- What does it mean to be "usable"
- Usable by whom?
  - Elderly
  - People with various backgrounds
- What hints of the object's "abilities" are evidenced from it's shape?

- Traditional Slides

-They look like this.



- "Lorem ipsum dolor sit amet, consectetur adipisicing elit,
- sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
- Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.
- Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur.
- Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum."

- Common Slide Mistakes

-Too much text on the slide.

$$\rho\left(\frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x}+v\frac{\partial u}{dy}+w\frac{\partial u}{\partial z}\right)=$$

$$\rho g_x-\frac{\partial p}{\partial x}+\frac{\partial}{\partial x}\left[2\mu\frac{\partial u}{\partial x}+\lambda\nabla\cdot\mathbf{V}\right]+\frac{\partial}{\partial y}\left[\mu\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right)\right]+\frac{\partial}{\partial z}\left[\mu\left(\frac{\partial w}{\partial x}+\frac{\partial u}{\partial z}\right)\right]$$

$$\rho\left(\frac{\partial v}{\partial t}+u\frac{\partial v}{\partial x}+v\frac{\partial v}{dy}+w\frac{\partial v}{\partial z}\right)=$$

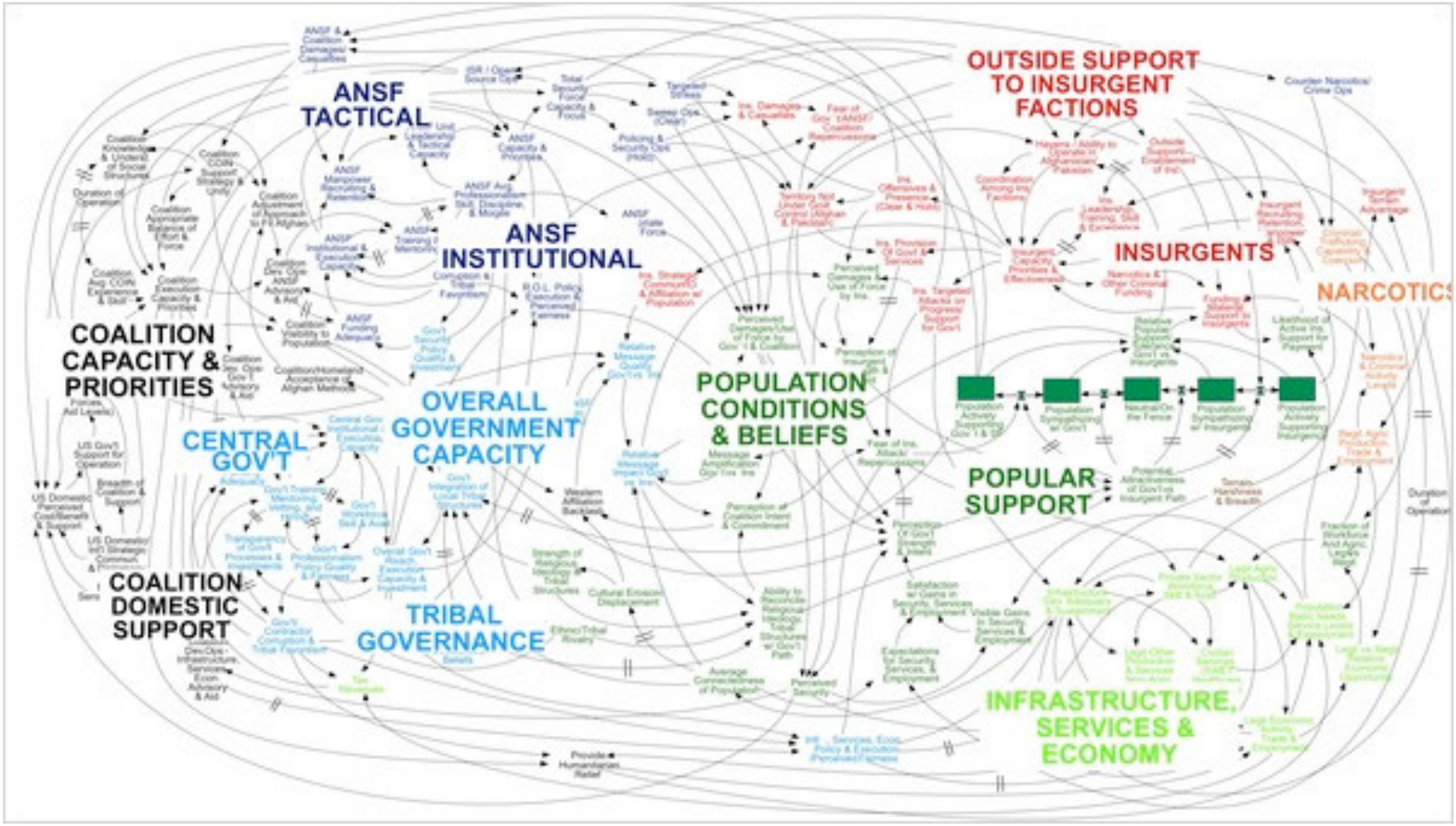
$$\rho g_y-\frac{\partial p}{\partial y}+\frac{\partial}{\partial y}\left[2\mu\frac{\partial v}{\partial y}+\lambda\nabla\cdot\mathbf{V}\right]+\frac{\partial}{\partial z}\left[\mu\left(\frac{\partial v}{\partial z}+\frac{\partial w}{\partial y}\right)\right]+\frac{\partial}{\partial x}\left[\mu\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right)\right]$$

$$\rho\left(\frac{\partial w}{\partial t}+u\frac{\partial w}{\partial x}+v\frac{\partial w}{dy}+w\frac{\partial w}{\partial z}\right)=$$

$$\rho g_z-\frac{\partial p}{\partial z}+\frac{\partial}{\partial z}\left[2\mu\frac{\partial w}{\partial z}+\lambda\nabla\cdot\mathbf{V}\right]+\frac{\partial}{\partial x}\left[\mu\left(\frac{\partial w}{\partial x}+\frac{\partial u}{\partial z}\right)\right]+\frac{\partial}{\partial y}\left[\mu\left(\frac{\partial v}{\partial z}+\frac{\partial w}{\partial y}\right)\right]$$

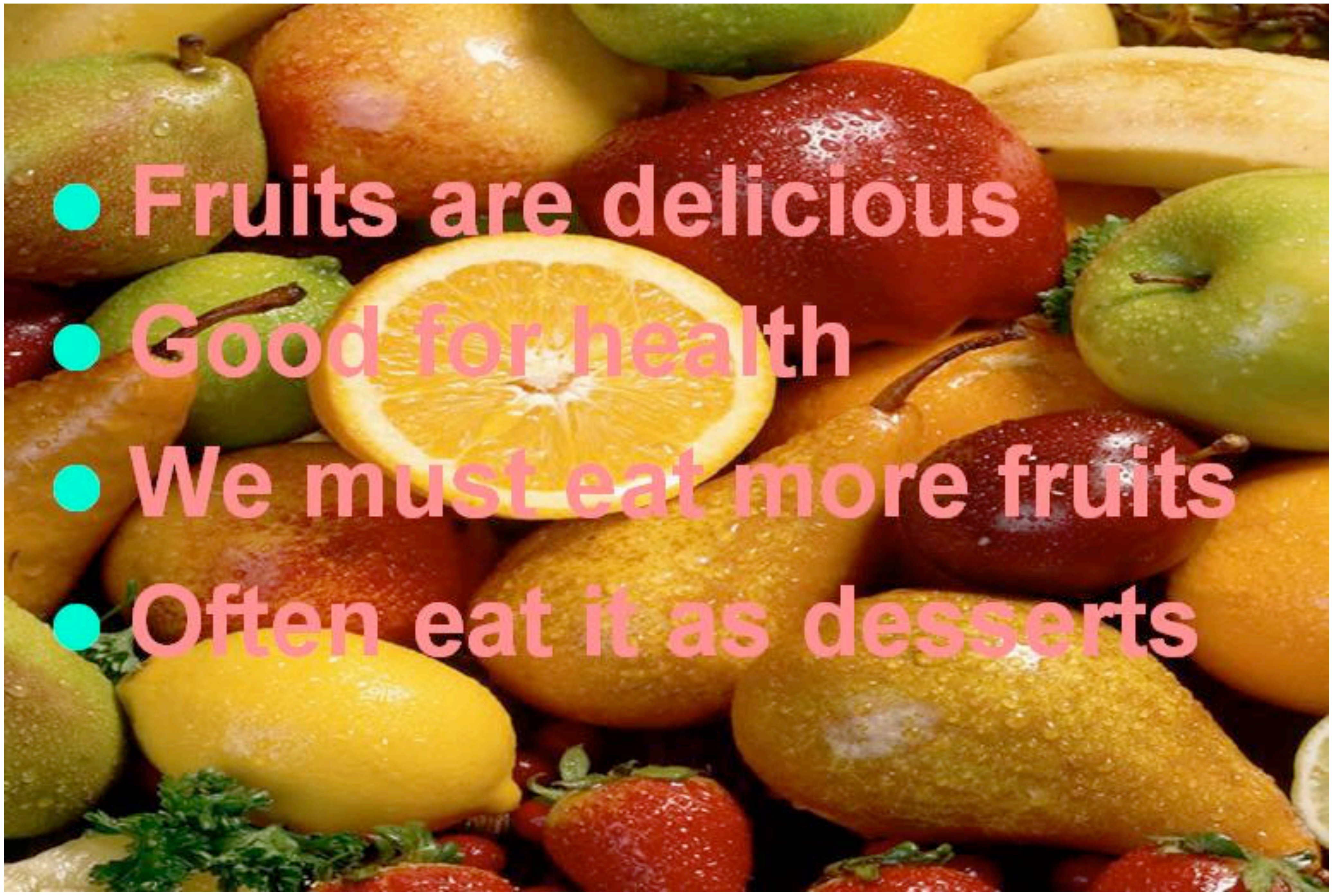
- Common Slide Mistakes

-Too much math (equations) on the slide.


ANSF TACTICAL
OUTSIDE SUPPORT TO INSURGENT FACTIONS
ANSF INSTITUTIONAL
INSURGENTS
NARCOTICS
COALITION CAPACITY & PRIORITIES
POPULATION CONDITIONS & BELIEFS
OVERALL GOVERNMENT CAPACITY
CENTRAL GOV'T
POPULAR SUPPORT
COALITION DOMESTIC SUPPORT
TRIBAL GOVERNANCE
INFRASTRUCTURE, SERVICES & ECONOMY

- Common Slide Mistakes

-Too busy a background texture.
-Too complex an image or diagram.
-Overuse of slide transitions.



- Fruits are delicious
- Good for health
- We must eat more fruits
- Often eat it as desserts

- Common Slide Mistakes

-Don’t get too creative with text colours and background colours.
-Many colour combinations which look good on a computer monitor are difficult to read when projected onto a presentation screen.
-Keep the contrast between the text and background high.
-Don’t change the fonts. Pick one or two and use them consistently.

- Presentation Mistakes

-Don’t read the slides. The audience doesn’t need to you read to them. They need to you to explain the ideas.
-Look at the audience.
-Practice the presentation. Especially if you are new to giving presentations or are nervous.

- Presentation Mistakes

-Time yourself so you don't go on too long (this is rude). Bring a watch.
-It looks amateurish if you have to skip past the last few slides because your presentation is too long and you cannot finish presenting the material.

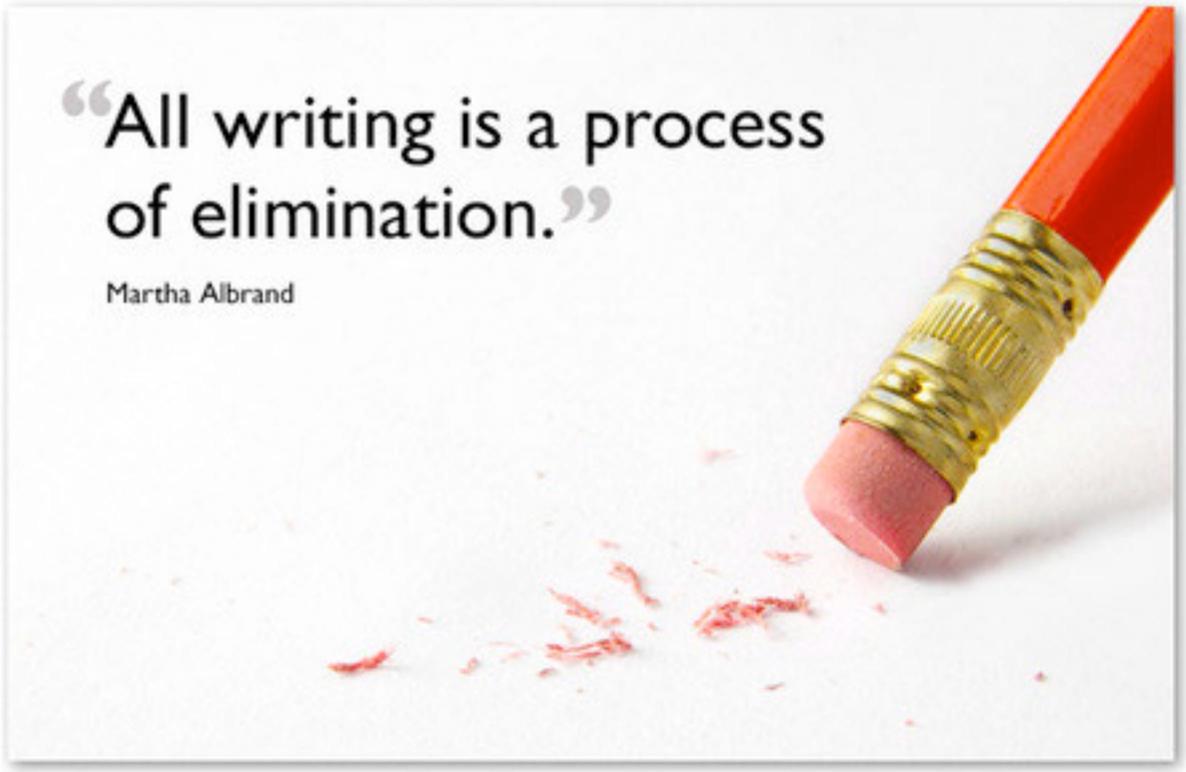
- Presentation Mistakes

-Avoid laser pointers unless there is a real need to point and even then be careful in using them. They are mostly used improperly. Don't shine them at the audience.
-Try to avoid repeated patterns in your behaviours. These can be speech patterns, gestures, or breathing. Speech patterns are repeated phrases which don't really add anything to the statement (e.g. in any case, so on and so on, basically). Breathing problems can come from nervousness. These can include holding your breath and occasionally gasping or deep sighs.

- Presentation Mistakes

-Relax (if possible). It's easier to think about and discuss your ideas if you are relaxed.
-If you are very nervous then the solution is preparation and practice. Prepare by thinking about the likely questions you will be asked. Practice so you are more comfortable.
-Practice will also let you time the presentation so you will know how long it will be. People often speak faster during the real presentation than during practice.
-Develop a plan and keep to it. Avoid the temptation to ad lib unless you are absolutely sure it is necessary.

Eat only
until 80% full.



- The Designer's Approach to Presentations

-It does not aim to provide as much detail on the slides.
-Detailed information is provided through other sources such as handout notes, links to web sites, or references to books.
-The slides from this kind of presentation are not a document. They are a collection of points and images which support the verbal component of the presentation.
-Making the slides into a handout tends to create a poor handout and poor support for the presentation.

- The Central Point

-This approach focusses on conveying a central point. This is the one thing that the audience should remember from the presentation.
-You should answer two questions during the presentation which are: What is your point and why does it matter?
-Think about your audience and try to answer these questions from their perspective. It is easy to forget that they don't have the same experiences as you. Things which are obvious to you may not be obvious to the audience.

- The Elevator Test

-Could you explain the main idea of the presentation in 30-45 seconds (the time of an elevator ride)?
-If you cannot then you don’t have a clear statement of the central point.
-For extremely technical topics this means that almost all details are removed and the results are the primary message.

Questions about the presentation.

- How much time do I have?
- Who is the audience?
- What is their background?
- Why was I asked to speak?
- What is the purpose of my talk?
- What do they expect from me?

-You should know the answers to these before you make the presentation.

- Making Messages Stick.

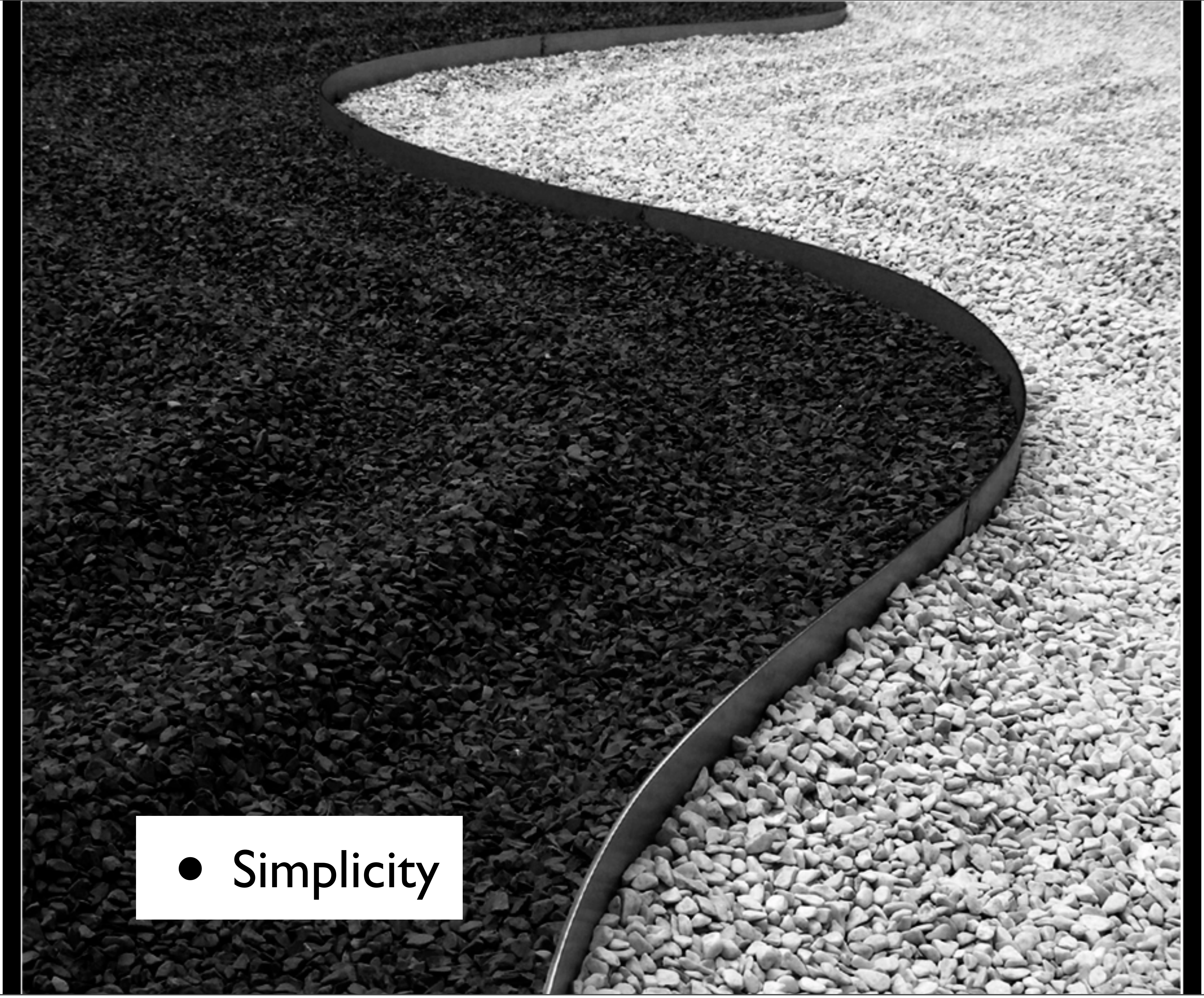
-How do you choose the message so the audience will remember it?
-Do not use abstractions which the audience wont find compelling. Use language and ideas which the audience will understand and appreciate.

Our mission is to become the international leader in the space industry through maximum team-centered innovation and strategically targeted aerospace initiatives.

Or

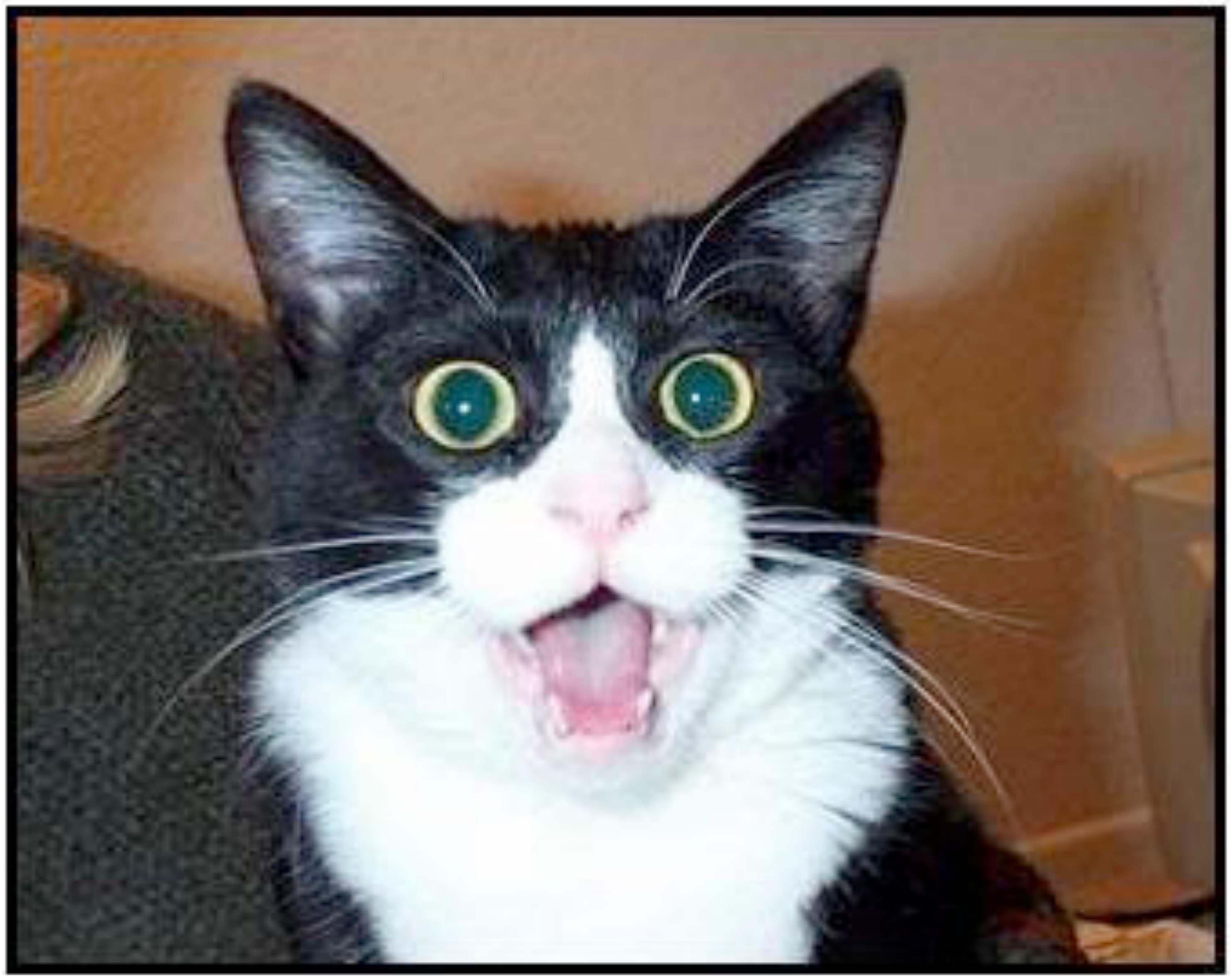
...put a man on the moon and return him safely by the end of the decade.

-The two messages address the same idea but which is more compelling?
-The first is CEO-speak and is neither comprehensible or memorable.
-The second is easy for people to relate and easy to understand.



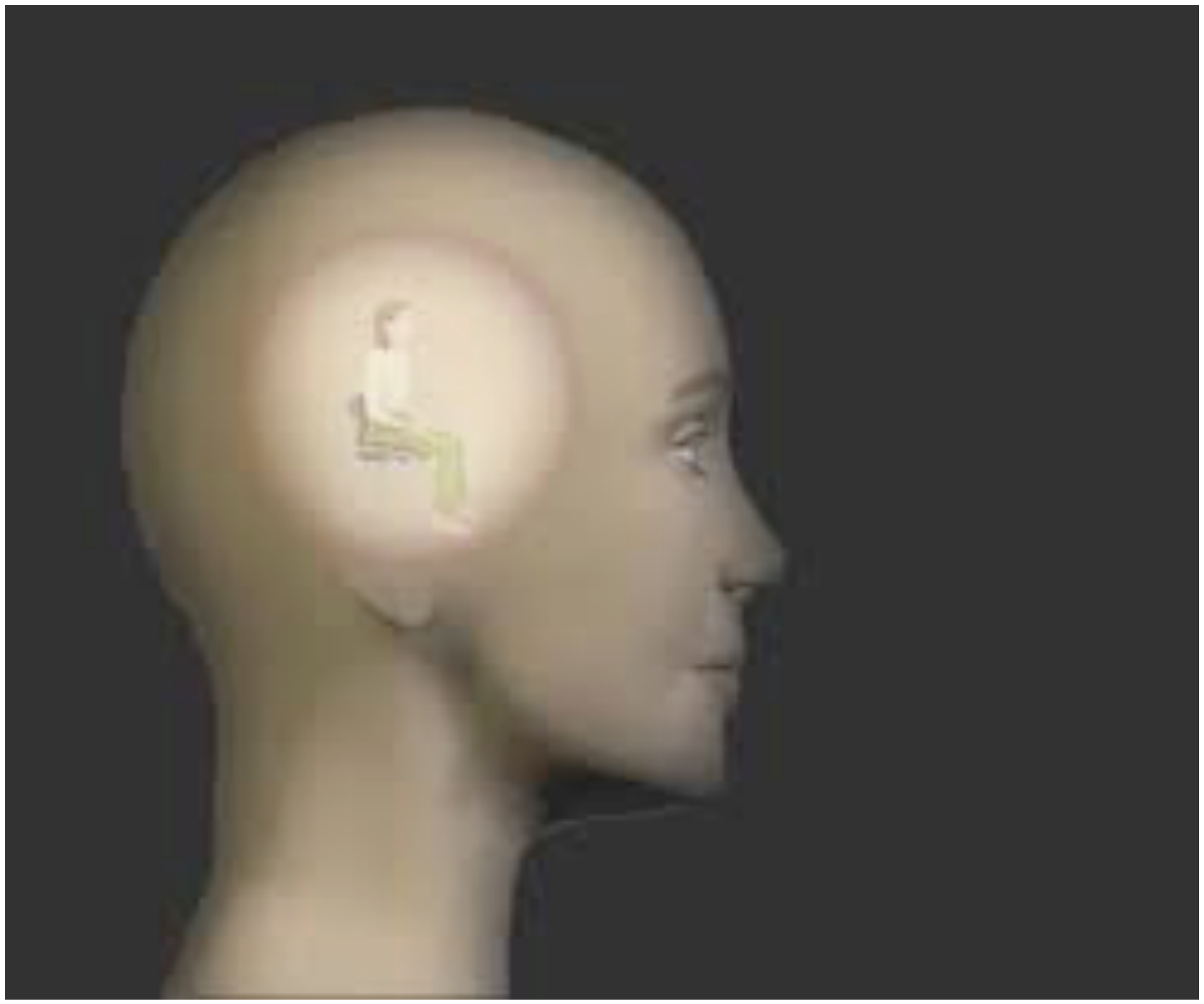
- Simplicity

-You normally cannot put all of the details into the presentation. If you try then the main idea will be difficult to understand because of the excess in details.
-Simplify the message to its core. This does not mean to “dumb down” the message but to reduce it to the essential meaning.
-What is the key point?



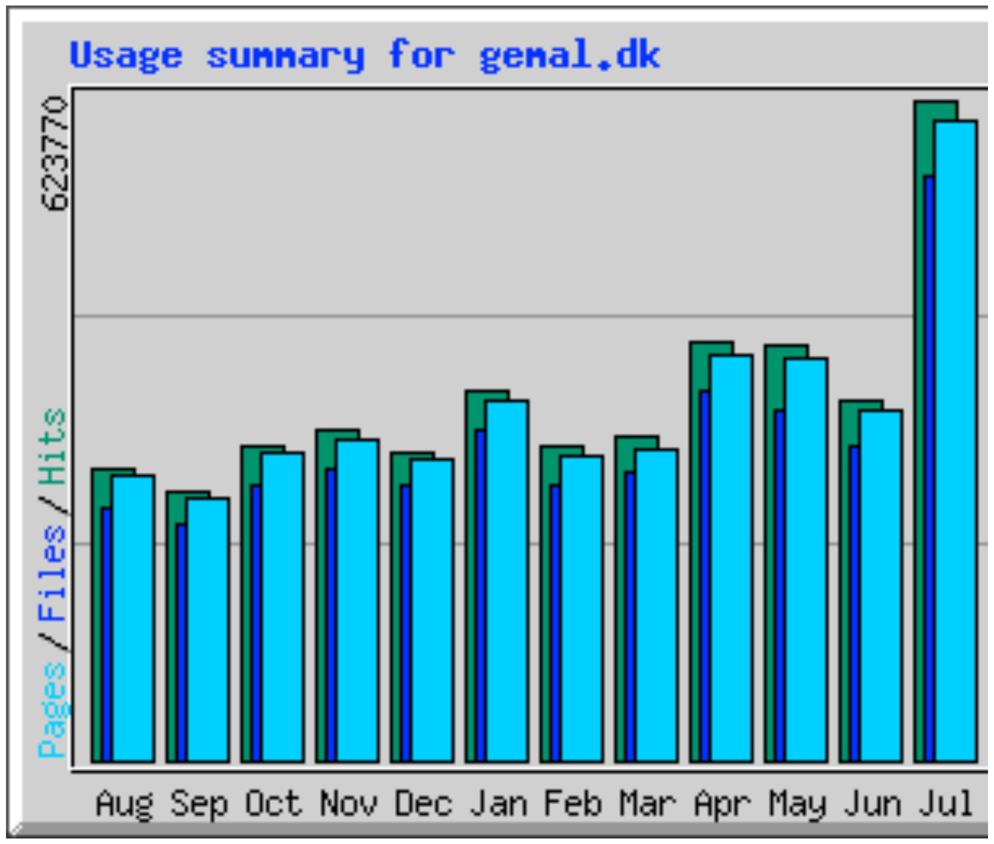
- Unexpectedness

-People will be interested if you do the unexpected. Surprise them and you will get their interest.
-A good way to do this is to pose questions which the audience will want to know the answer and then fill in the gaps in their knowledge by answering the question.



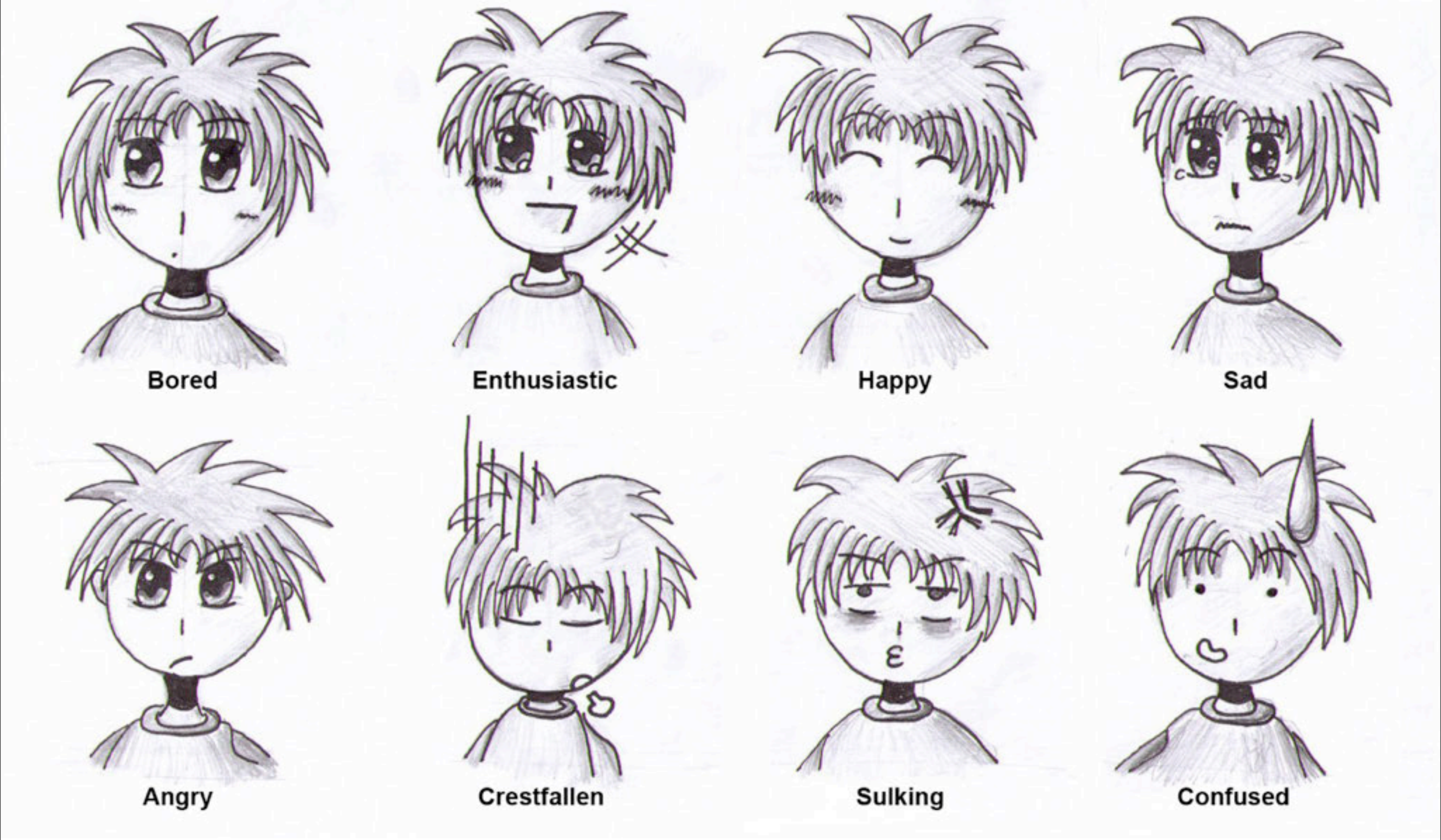
- Concreteness

-Use natural speech and give real examples. Do not use abstractions.
-The presentation will be easier to remember.
-People will remember things they can visualize.



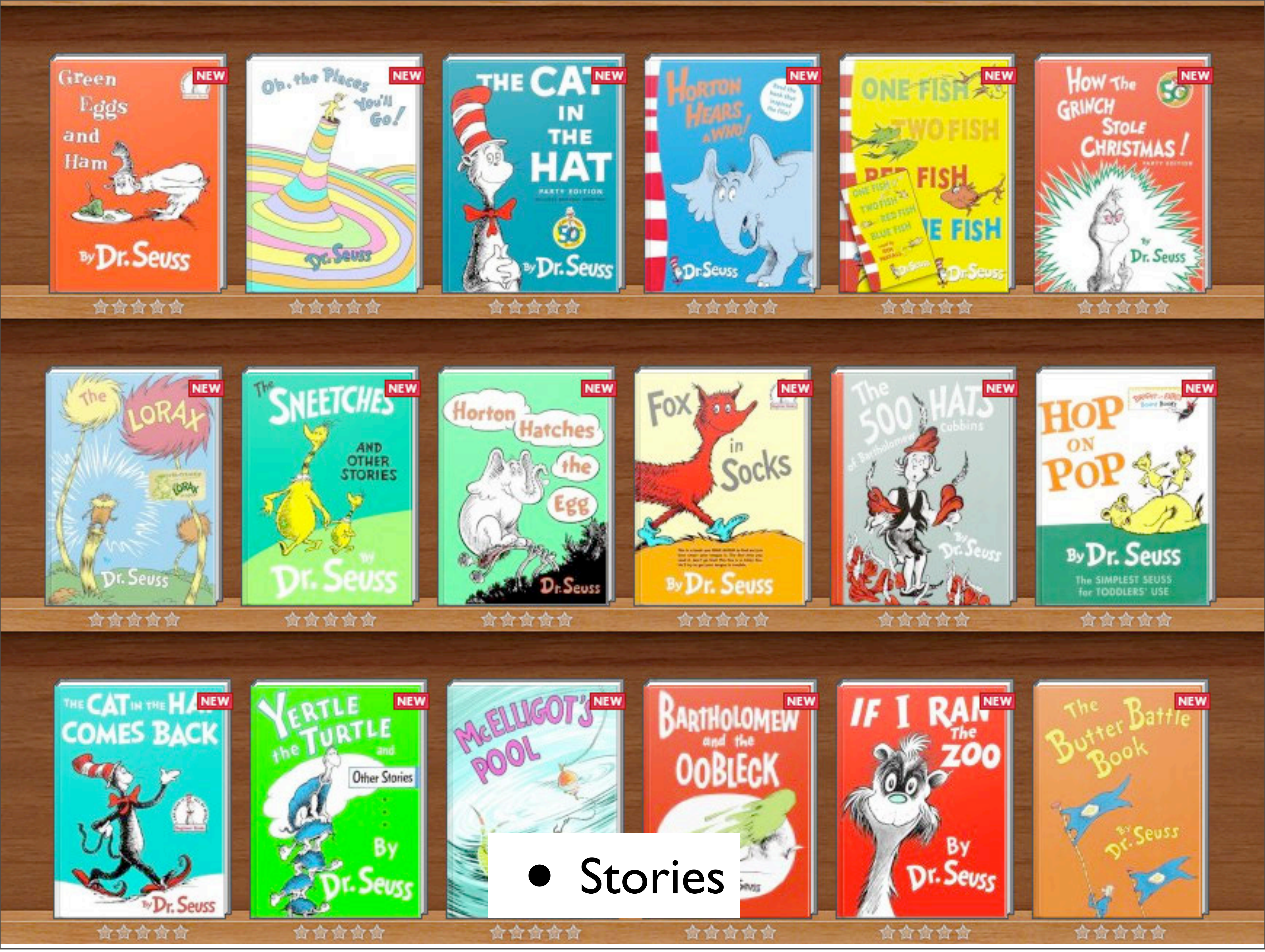
- Credibility

-We can use numbers and statistics to support claims. They are most useful when presented so the audience can relate to them. Instead of listing the numbers put them into a context the audience can understand (e.g. you can browse the web twice as fast, your database needs half of the maintenance).
-Quotes from experts, clients, or the press can be useful.



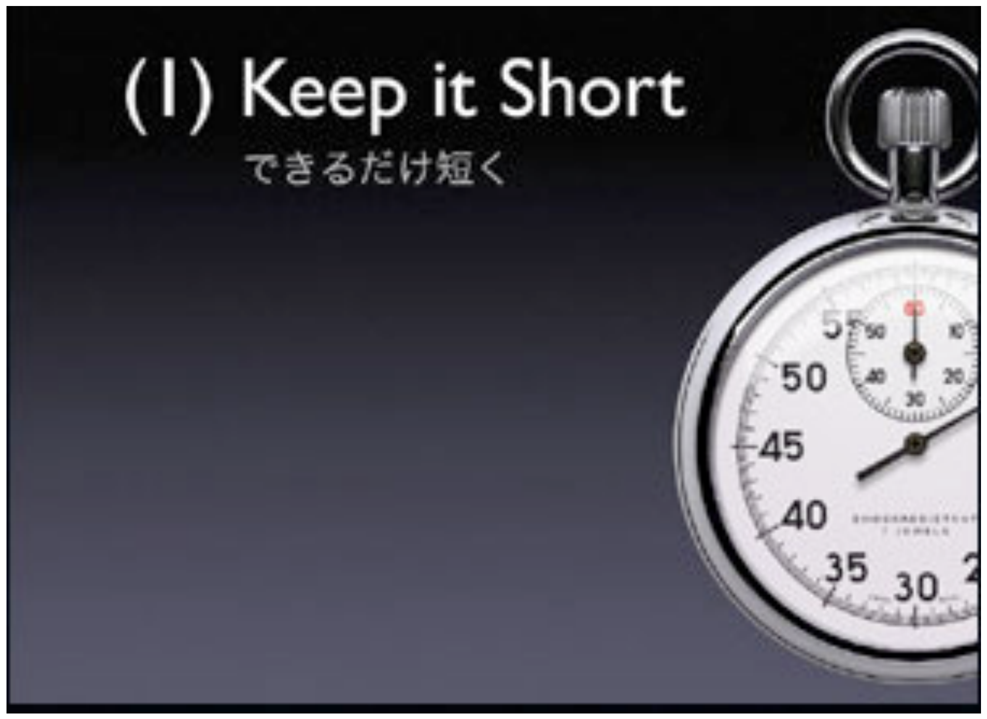
- Emotions

-Images can cause strong emotional responses in the viewer and make them feel something about the topic.



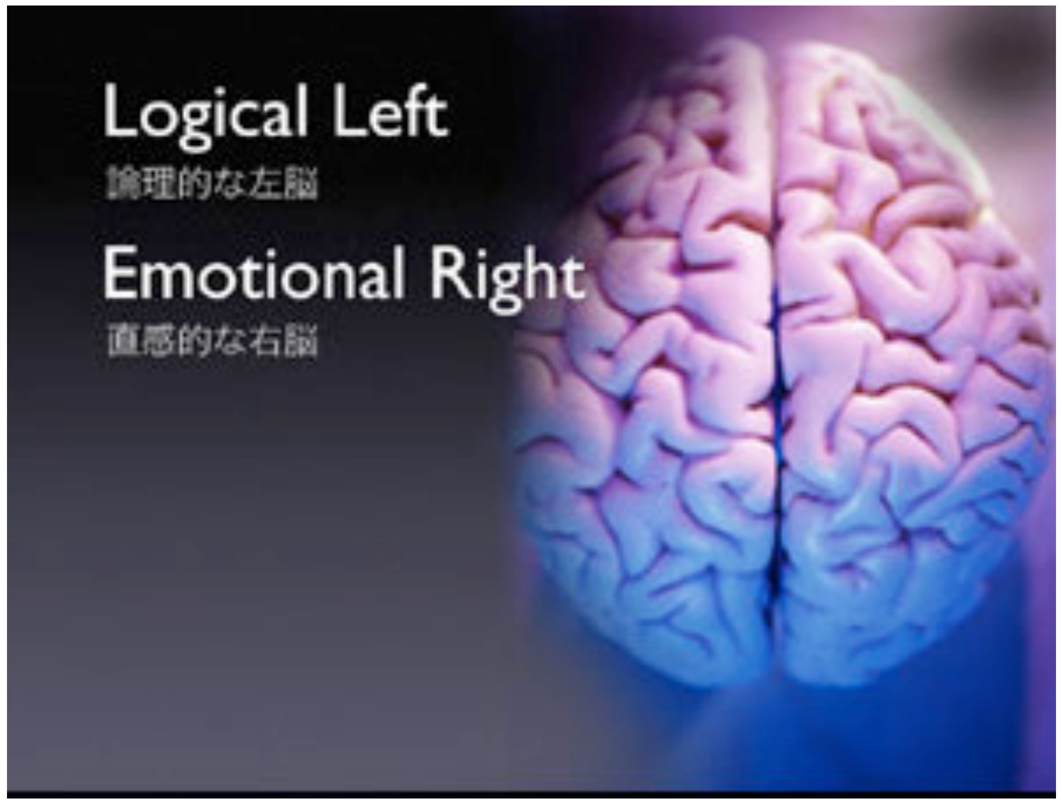
- Stories

-Stories provide a good way to make ideas memorable.
-They provide a context which is easier to remember than a list of points.
-They make the presentation seem authentic. Not just a collection of ideas which have been stuck together.



(1) Keep it Short

できるだけ短く



Logical Left

論理的な左脳

Emotional Right

直感的な右脳

- Sample Designer Slides

-The emphasis is on images with small amounts of text.

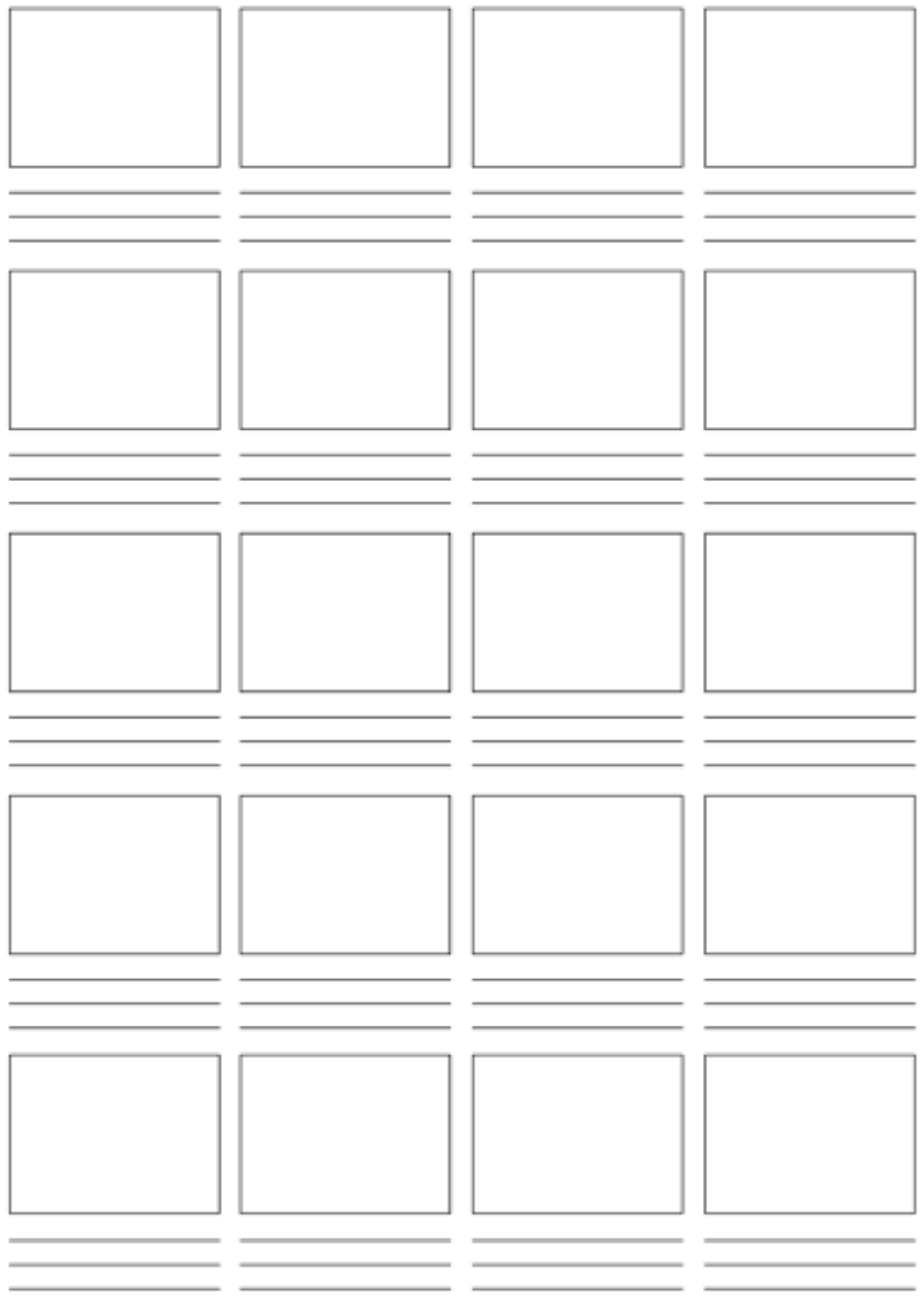
Steps in Developing a Designer Presentation

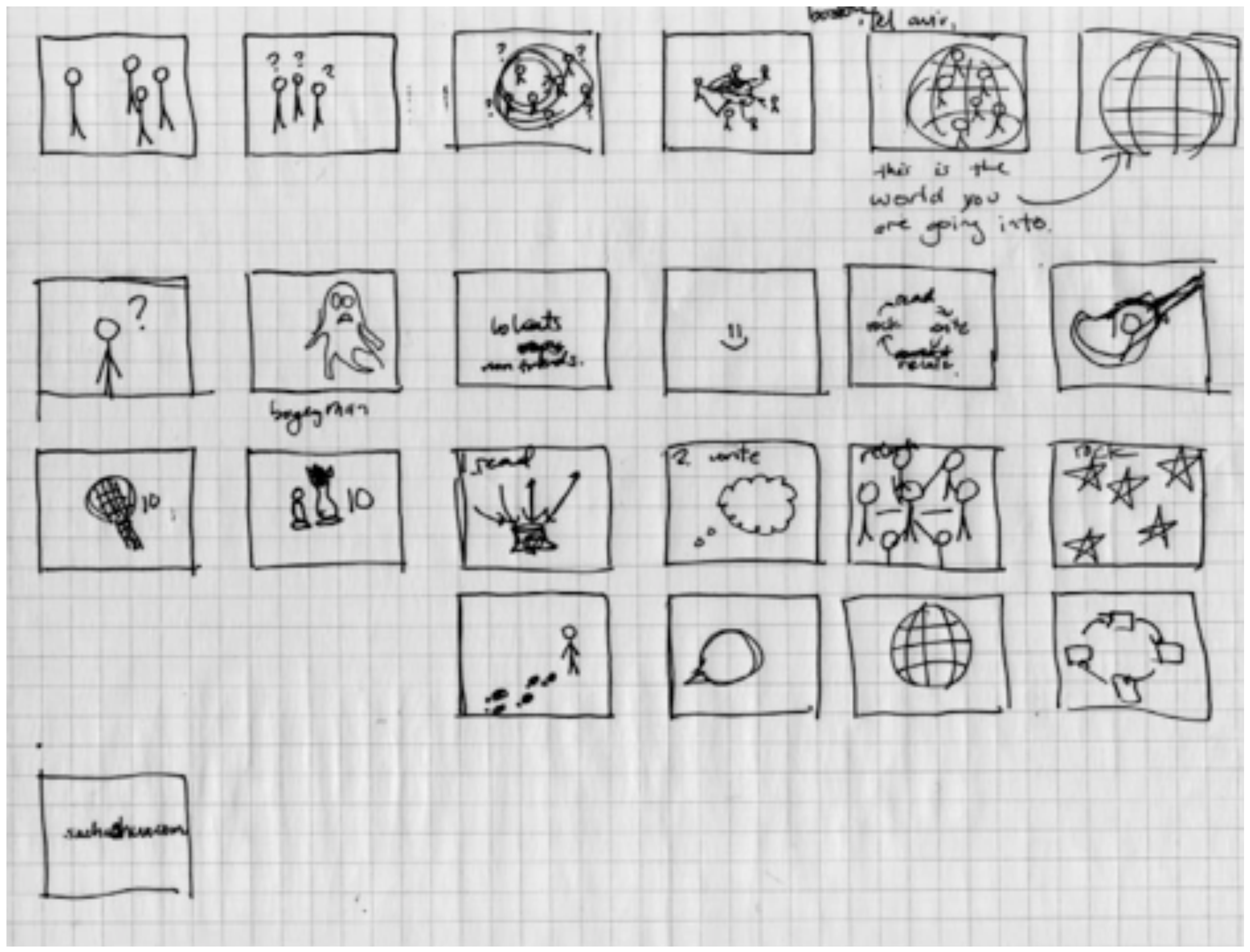
- Brainstorming

-Collect ideas for the presentation.
-Don’t edit them much at this point. That can happen later.
-Attempt to understand all aspects of the issue.
-If there is a client then this can be similar to requirements gathering.

- Group and identify core ideas.

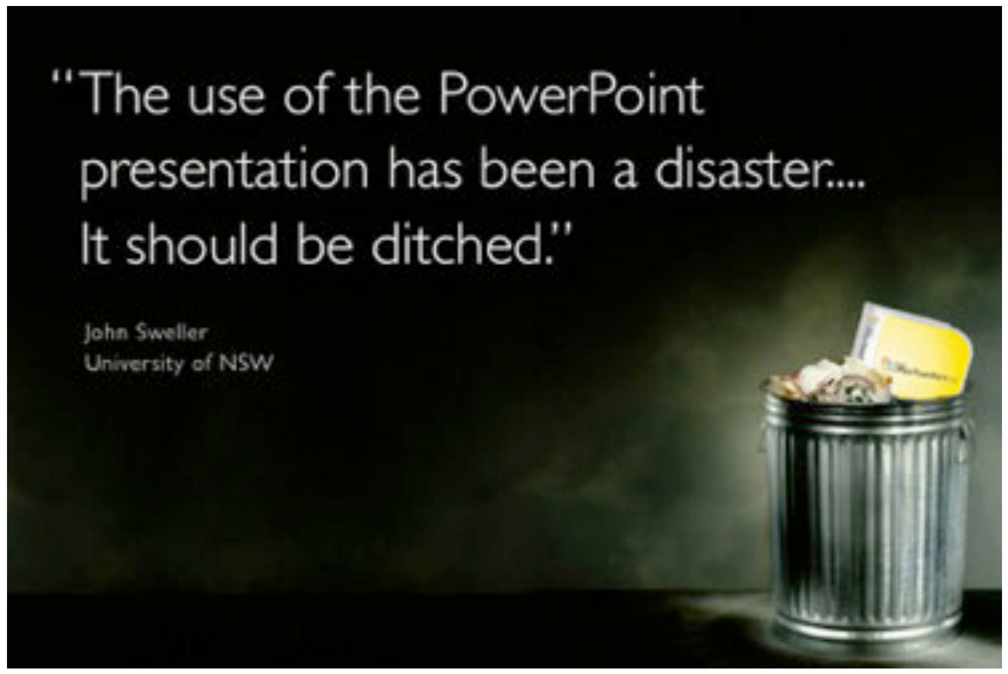
-Identify the one idea which you want the audience to remember.
-If you have multiple ideas which are similar then group them together and look for a unifying theme.
-Divide the presentation into sections. Each section supports the theme but can present different ideas. Three sections are a good number because it provides a memorable structure but the number of sections is not fixed.



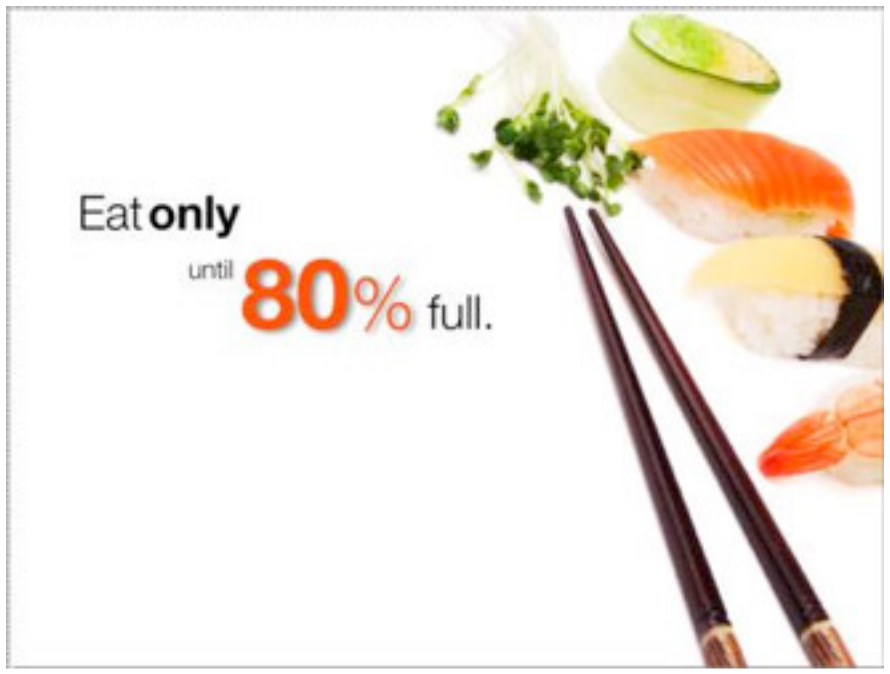


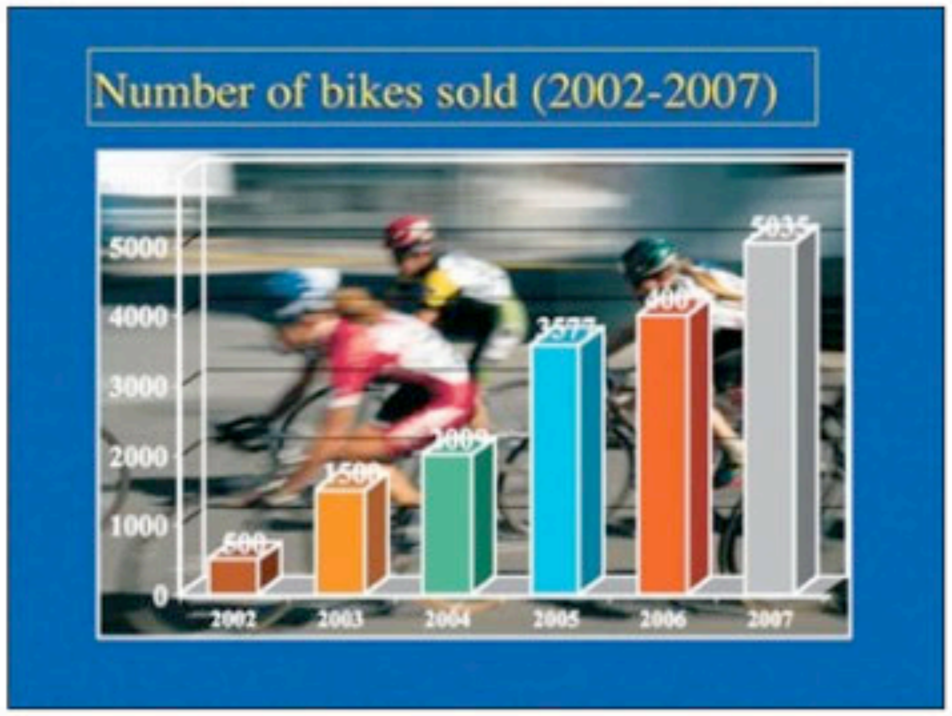
- Storyboarding

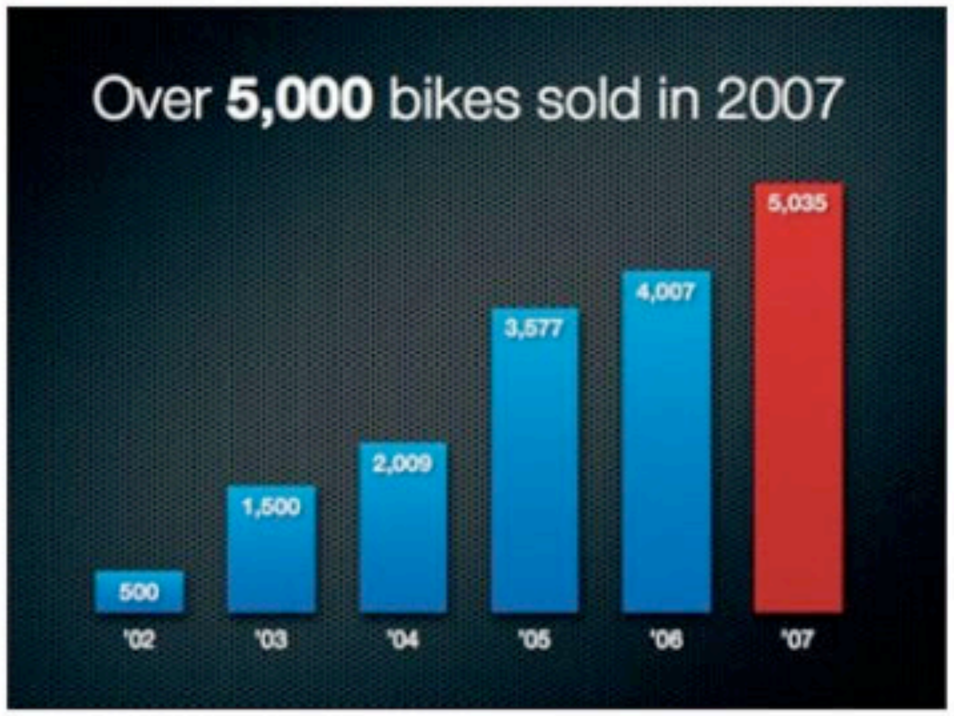
-Organize key points and sketch images. Both are done on paper.
-This allows you to built the story without committing too much effort to the implementation. It is always possible to write another note or change a picture.
-The sketches for the slides can be rearranged easily at this point.
-You can skip this step if you have a clear idea for the structure.


"The use of the PowerPoint presentation has been a disaster.... It should be ditched."

John Sweller
University of NSW







- Create the slides

-Using the storyboard create the slides for the presentation.
-Normally start with a description of the problem and then follow it with the sections which describe the solution.
-When you are done go back to the start of the presentation and edit it. Remove anything that isn't essential. When in doubt, remove it.

- Not everyone agrees.

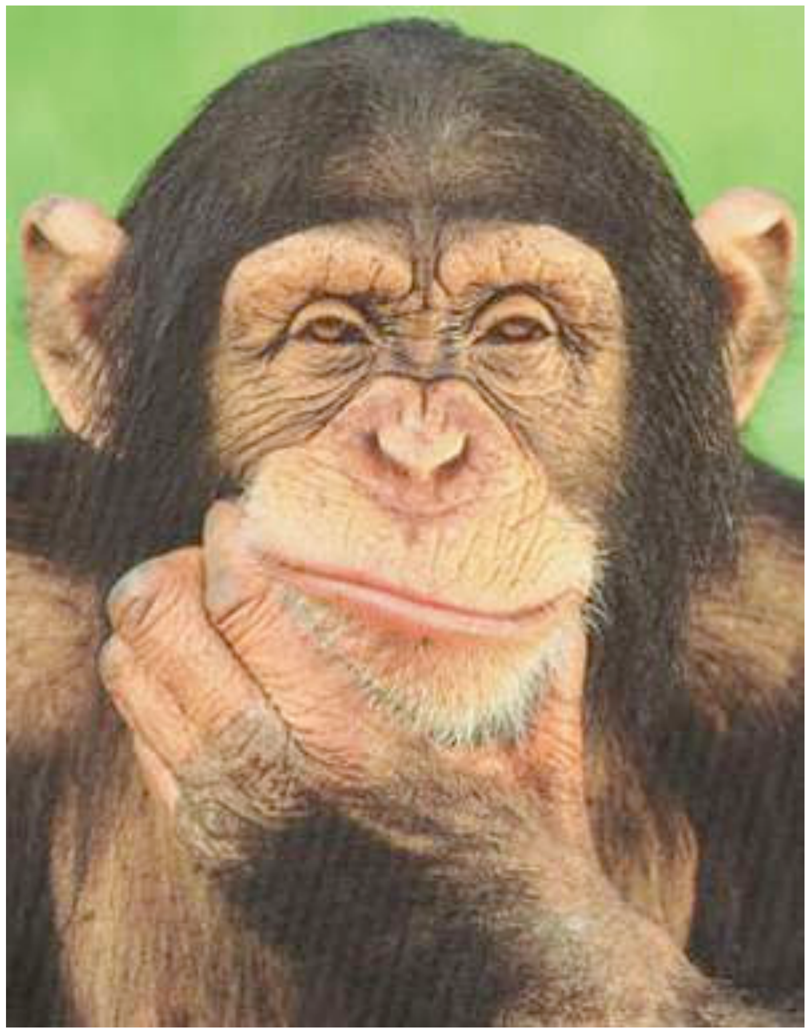
-Elaborate presentations are not universally seen as the best way to do business.
-Perhaps neither the traditional or designer’s approach is appropriate.

- "when the anthropologists look back on the 1980s and 1990s and do the archaeological digs, and get their callipers and brooms and microscopes out, they will blame the massive reduction in productivity during the 1980s and 1990s entirely on Microsoft Office."   -Scott McNealy

-Scott McNealy is the chairman, president, and founder of Sun Microsystems.
-The PC is ubiquitous, and every desktop in every office, of every programmer, secretary, manager or filing clerk has a full-blown office 'productivity' suite.
-It focusses thoughts into bullet points when broader thinking may be more appropriate.

- "...now we've got highly paid people spending hours formatting slides because it's more fun to do that than concentrate on what you're going to say..."

-It would be more efficient to give that work to lower paid employees who could do it faster and probably with better results.



- Millions of executives around the world are sitting there going, 'Arial? Times Roman? Twenty-four point? Eighteen point?

–Quote from Scott McNealy in The New Yorker, May 28, 2001.

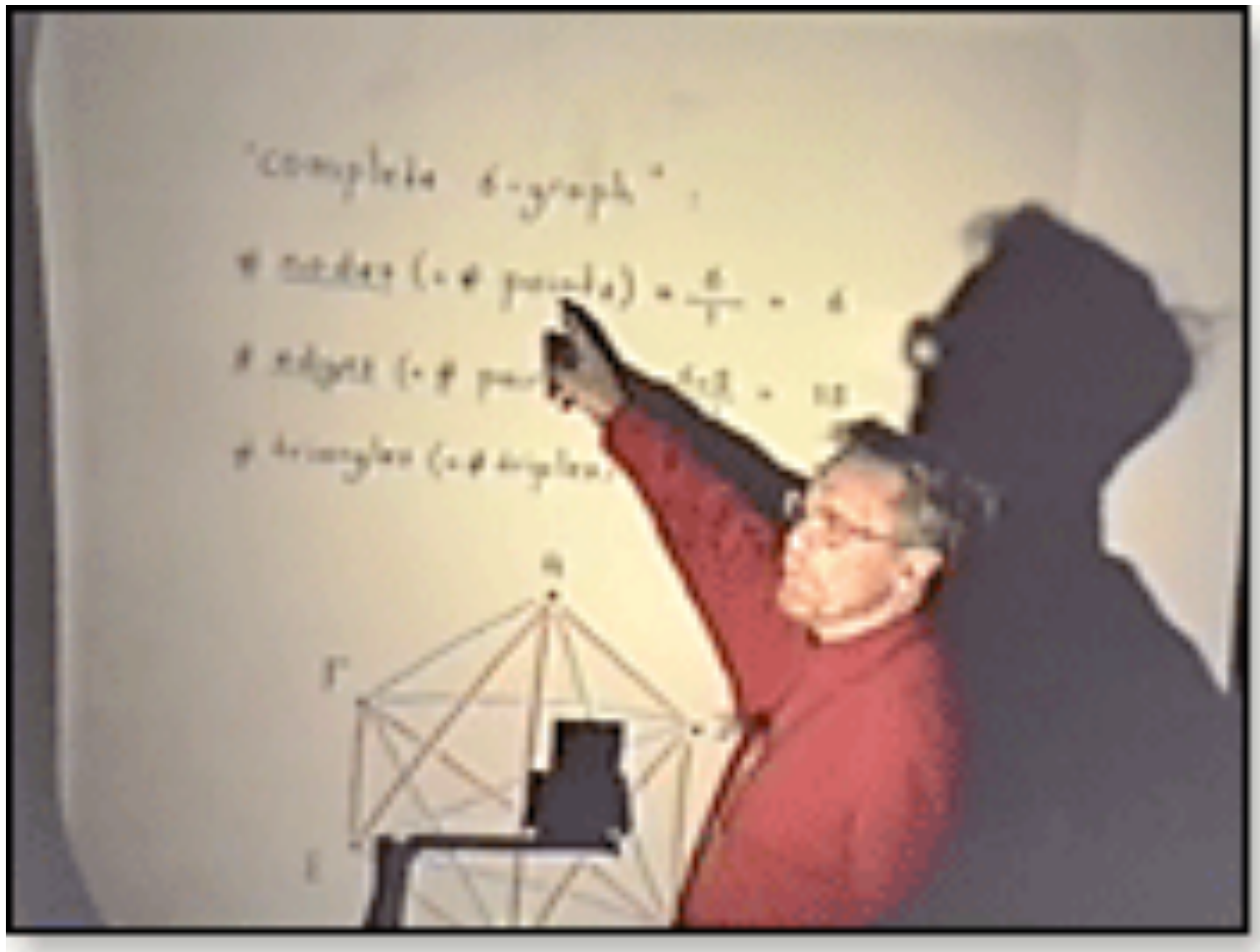
More from Scott McNeily

- “We had 12.9 gigabytes of PowerPoint slides on our network. And I thought, 'What a huge waste of corporate productivity'. So we banned it. And we've had three unbelievable record-breaking fiscal quarters since. Now I would argue that every company in the world, if it would just ban PowerPoint, would see its earnings skyrocket. Employees would stand around going: 'What do I do? Guess I've got to go to work.’”

-From San Jose Mercury, 3 August 1997.
-The office suite moved from the secretaries desk to the computer of everyone in the organization.
-The negative effect of this is that it promotes elaborate communications when simpler methods would suffice. It is now expected that elaborate documents will be provided.

McNealy's Solution

- "give everybody plastic Mylar sheets and all the pens they need to scribble on them"



-In large corporations it is common for spend a lot of time preparing documents which exist only to impress managers but are largely unread.
-A full office suite encourages managers to expect to receive good looking documents.

- Microsoft estimated that Powerpoint was used to create 30 million presentations a day in the year 2000.

-How many hours were spent creating 30 million presentations?

Others Agree

- The US Navy Secretary Richard Danzig has announced that he is no longer willing to sit through slide shows, saying that they were necessary only if the audience was "functionally illiterate". Too much time and effort is spent messing with PowerPoint, and not enough is spent on the message.

THIS CONCLUDES MY PRESENTATION. ARE THERE ANY QUESTIONS?
www.dilbert.com scottadams@aol.com
HOW DO I GET THE BOREDOM OUT OF MY HEAD?!!
3/5/99 © 1999 United Feature Syndicate, Inc.
THE FUNNY THING IS THAT I'LL LIST THIS ON MY ANNUAL ACCOMPLISHMENTS.
AIR! I NEED AIR!!!

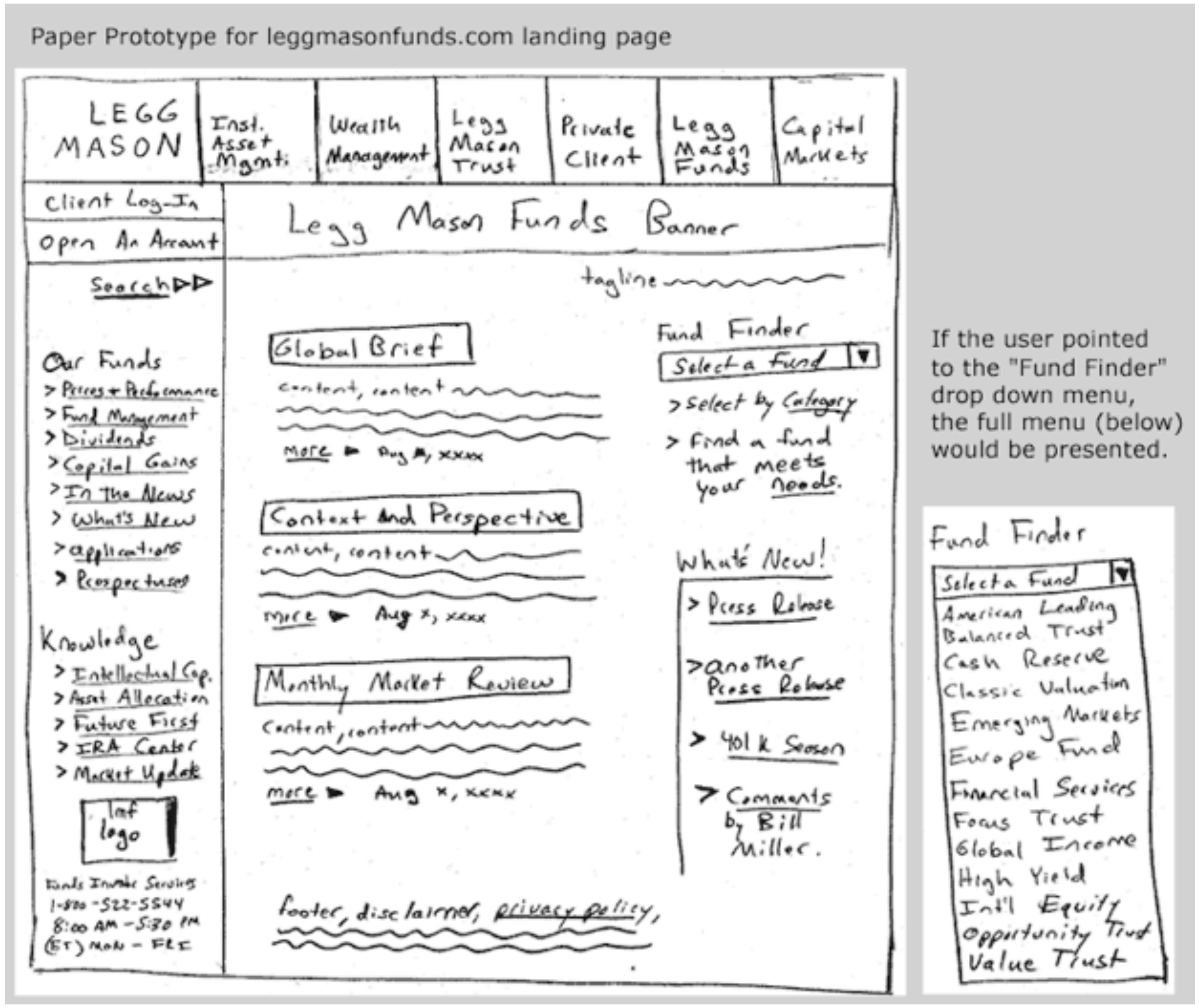
References

- Presentation Zen: Simple Ideas on Presentation Design and Delivery, by Garr Reynolds, New Riders, 2008.
- http://www.presentationzen.com/
- http://tuxdeluxe.org/node/38

Paper Prototyping

- A Prototype

-A model which something is based on. An early design.
-It is built with the understanding that it may be incorrect and will probably need to change.
-They are used to try out ideas.
-They are generally not complete models.

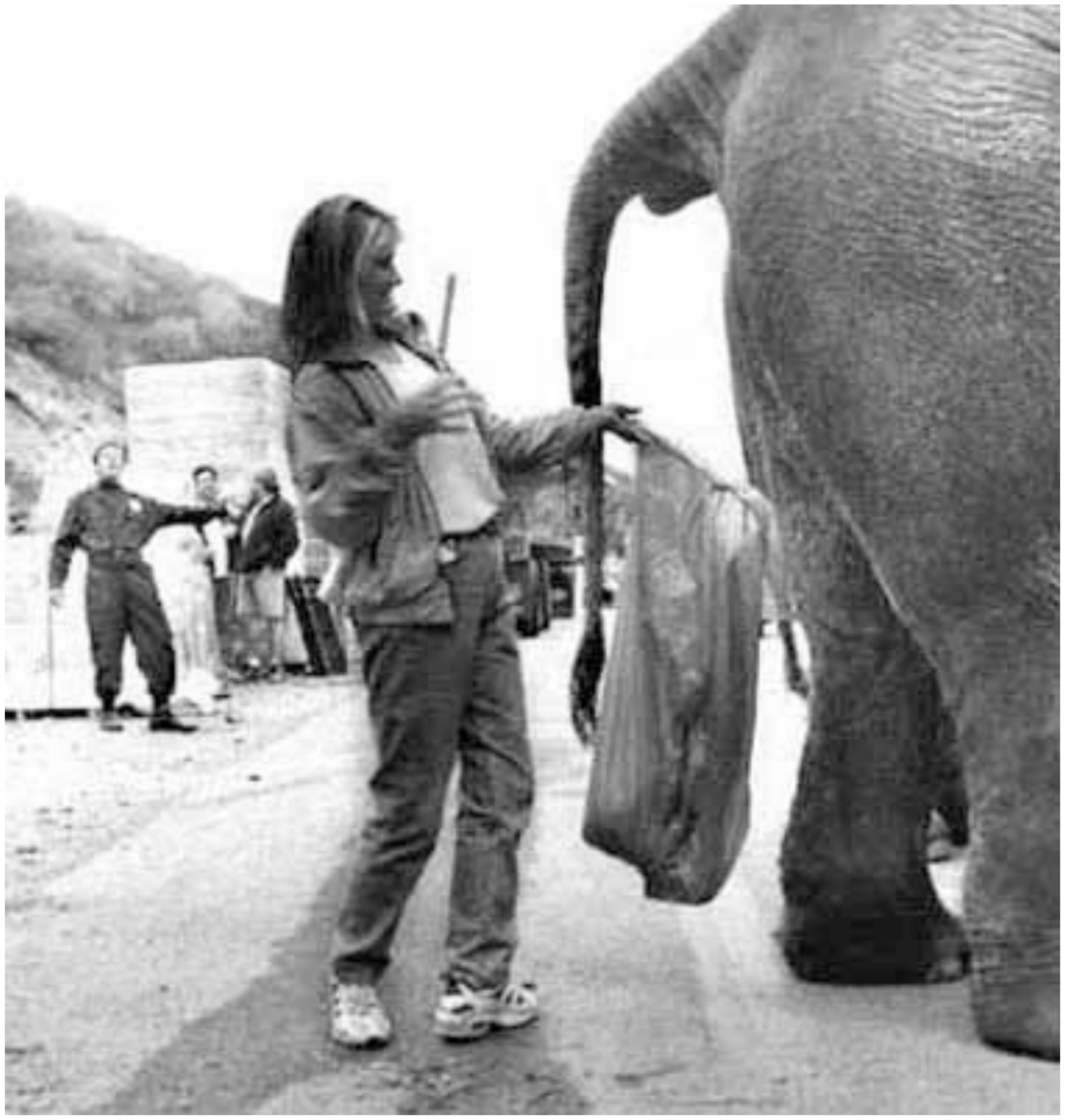


- What is a paper prototype?

-It’s a way to brainstorm, design, test, and communicate user interfaces.
-It can be used for almost any interface, including telephones, car dashboards, handheld games, or computer applications.

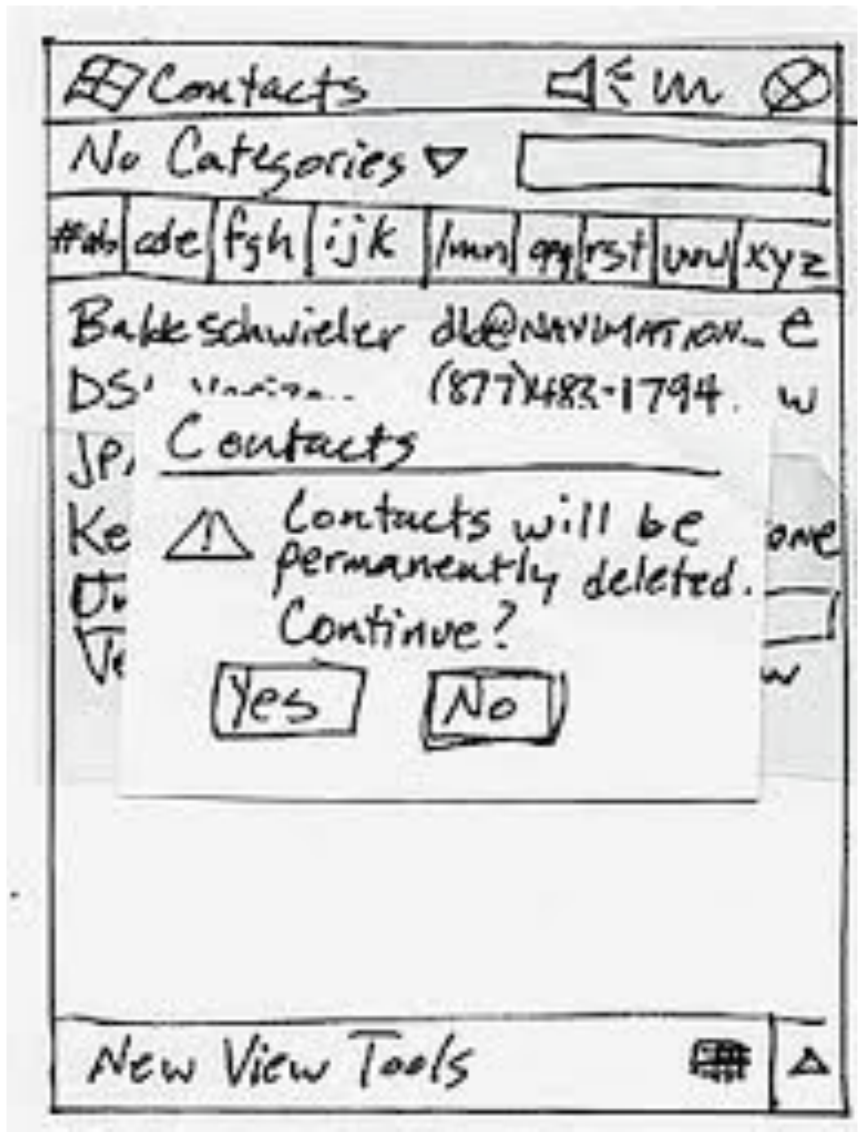
- How does it work?

1. You meet with the development team and choose the type of user who is most important audience for the interface.

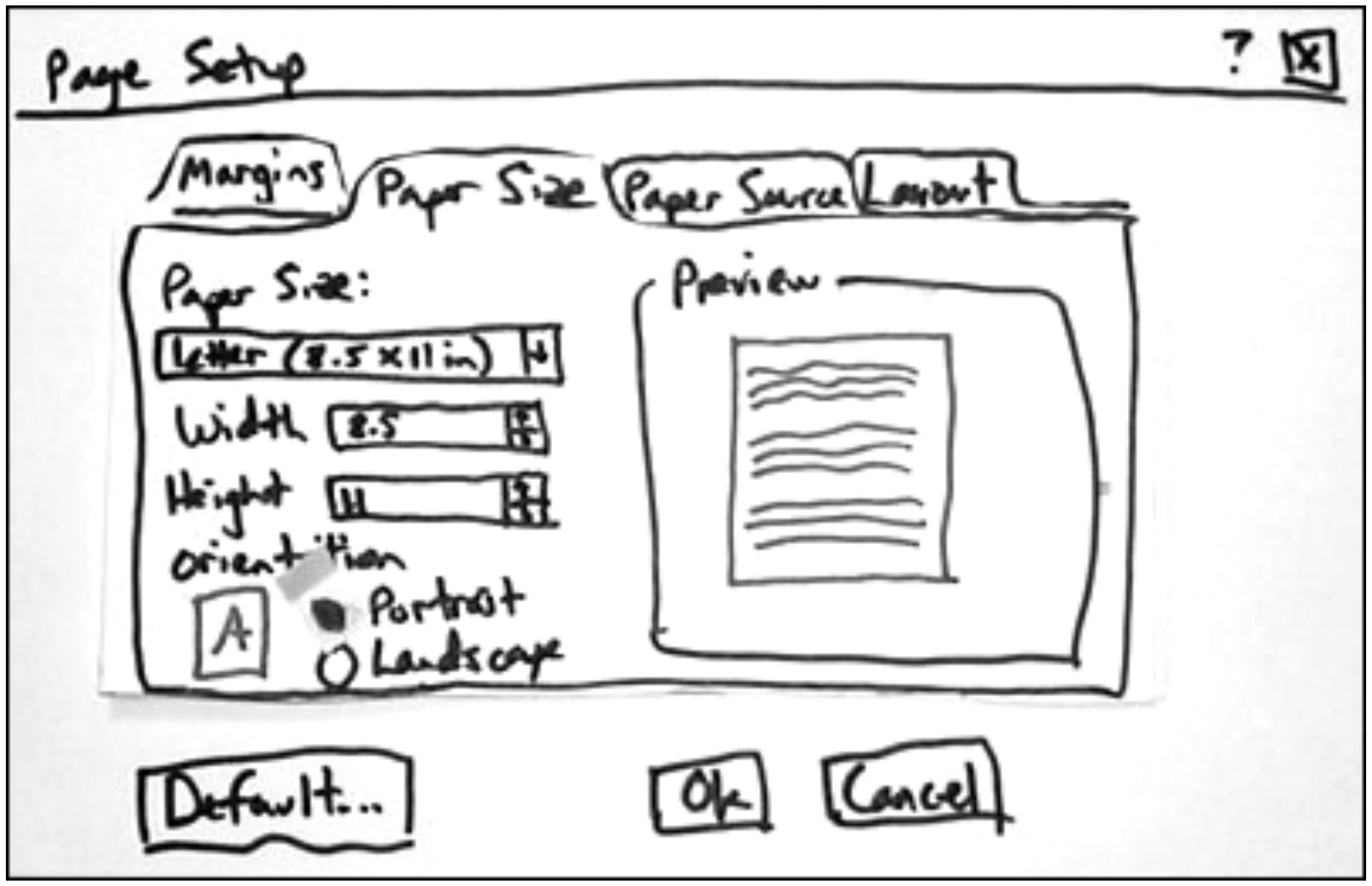


- How does it work?

2. Determine some typical tasks the that you expect the user to do.

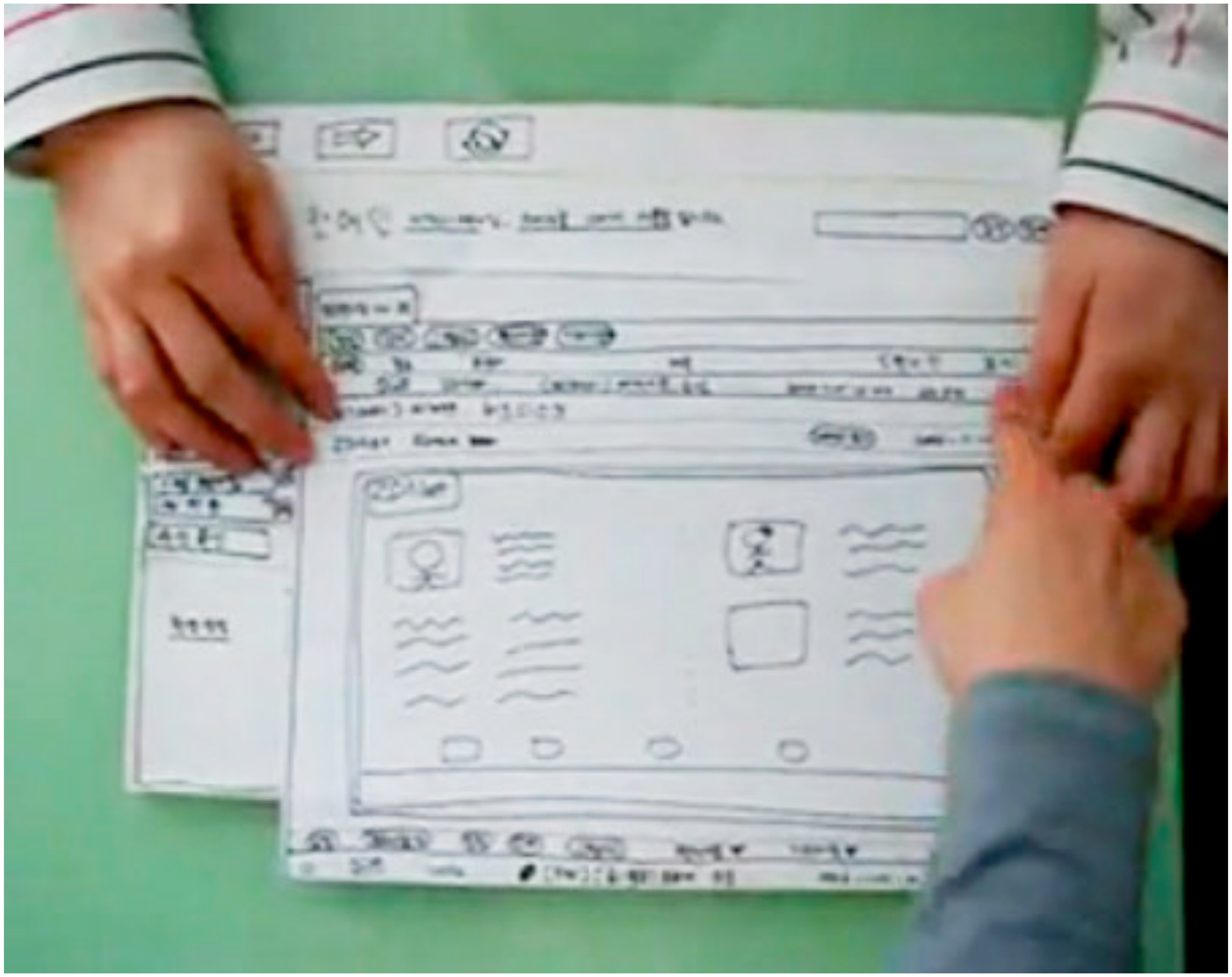


Contacts
No Categories
#ab cde fgh ijk lmn opq rst uvw xyz
Contacts
Contacts will be permanently deleted. Continue?
Yes
No
New View Tools



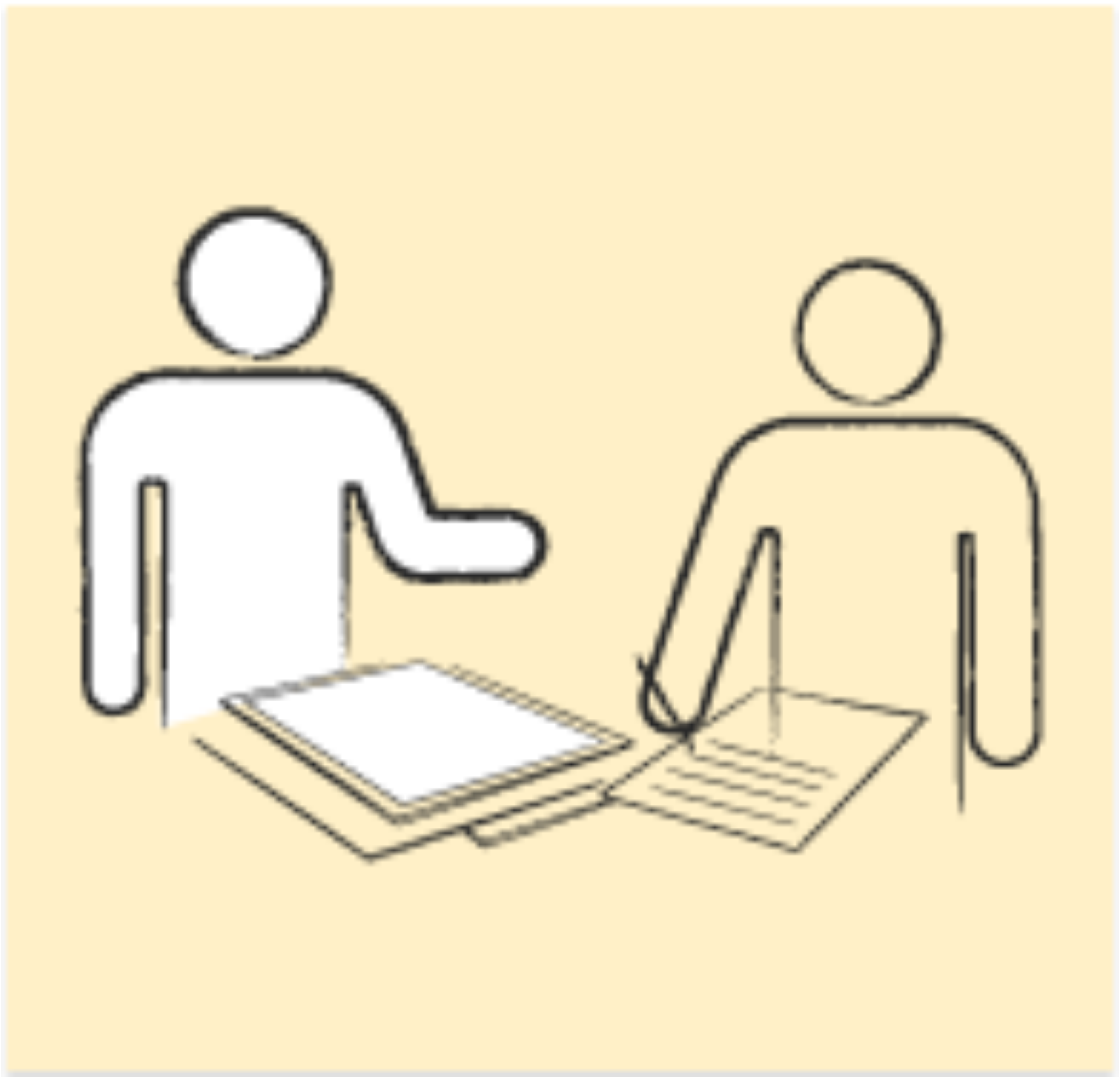
- How does it work?

3. Make screen shots or sketches of all of the windows, menus, dialog boxes, pages, data, pop-up messages, and other interface components that are needed to perform those tasks. This can also be done on a whiteboard.



- How does it work?

4. Perform a usability test. Get a user that the team previously identified to interact with the prototype. They “click” things by touching them and they fill in text by writing on the prototype. One of the developer plays the role of the computer and manipulates the paper to simulate the interface. Another developer acts as a facilitator and manages the session. The user explores the interface by interacting with it. An observer takes notes.

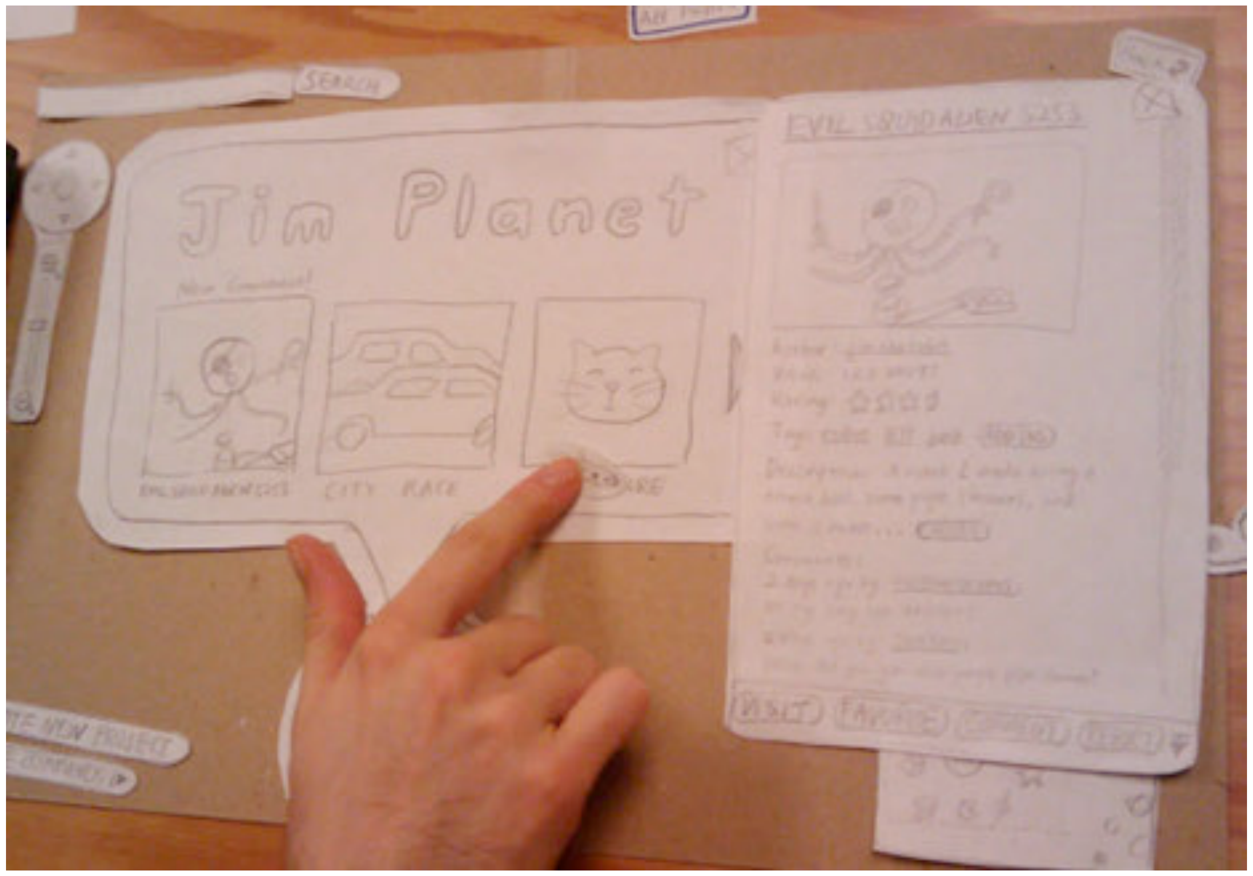


- What does this tell you?

-You will find out which parts of the interface work well and which are trouble spots.
-Since the prototype is done on paper you can easily modify it after or during the test.
-You can conduct several tests quickly and it generally doesn't take long to identify a pattern in the feedback you receive.
-You can rapidly iterate the design based on real user input before any code is written.

- Roles in a usability test.

-User, computer, facilitator, and observer.
-There can be multiple people in each role, especially observers.



- What is and is not required?

Required
-Interactivity with the user. -Real text.
Not Required
-High quality final interface designs that include colour, font, artwork, and layout choices.
-Filler text.
-A storyboard or flowchart showing all interaction. This is not interactive.

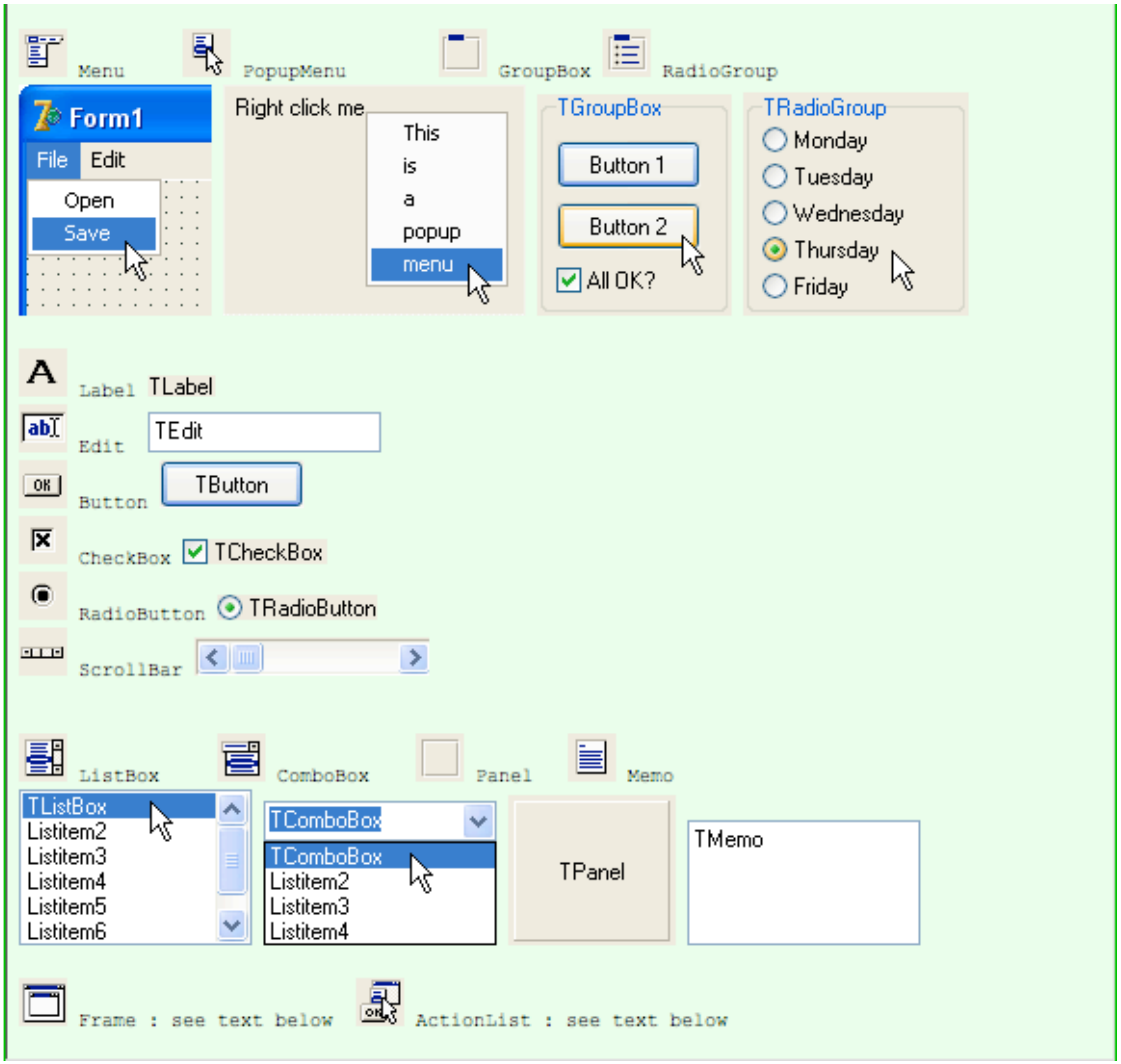
- Benefits.

-Feedback before any code is written.
-Promotes rapid iteration. You can experiment with different ideas rather than just one.
-Facilitates communication within the development team and with the user.
-Does not require any technical skill. You can train almost anyone to participate.
-Encourages creativity in the development process.

Developer watching videotape of usability test.

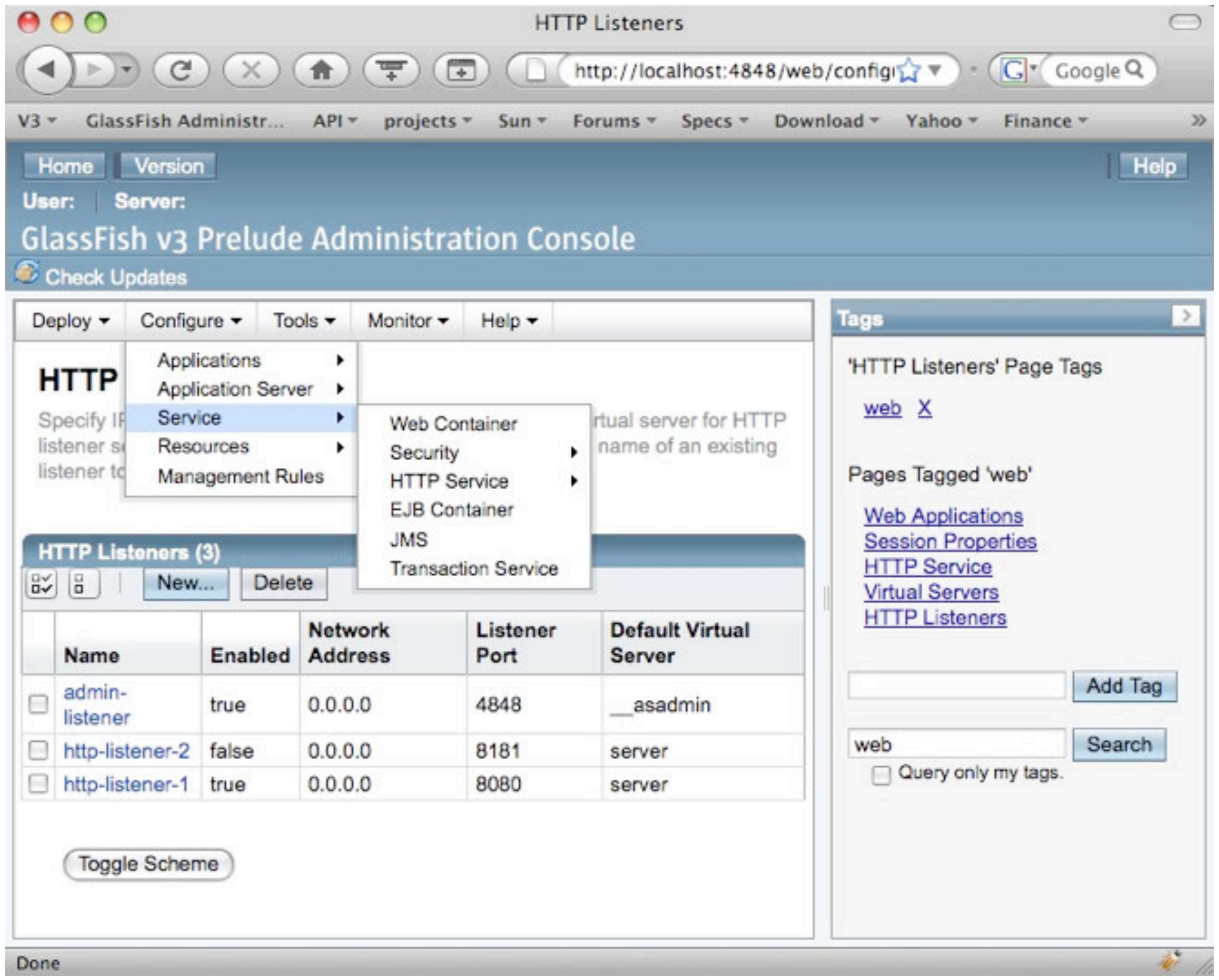
- Usability or User Centred Design.

-The goal is to make the interface better for its intended audience and purpose.
-A quote from Donald Norman, "you know you've got it right when your customers don't talk about how usable the product is...they are too busy raving about how you've make their life better." In other words, the interface disappears and doesn't get in the way of using the product.



- Graphical Interface Components

-Common interface components include, pull down and pop up menus, buttons, radio buttons, labels, checkboxes, text fields, scroll bars, list boxes, window, title bars, menu bars.



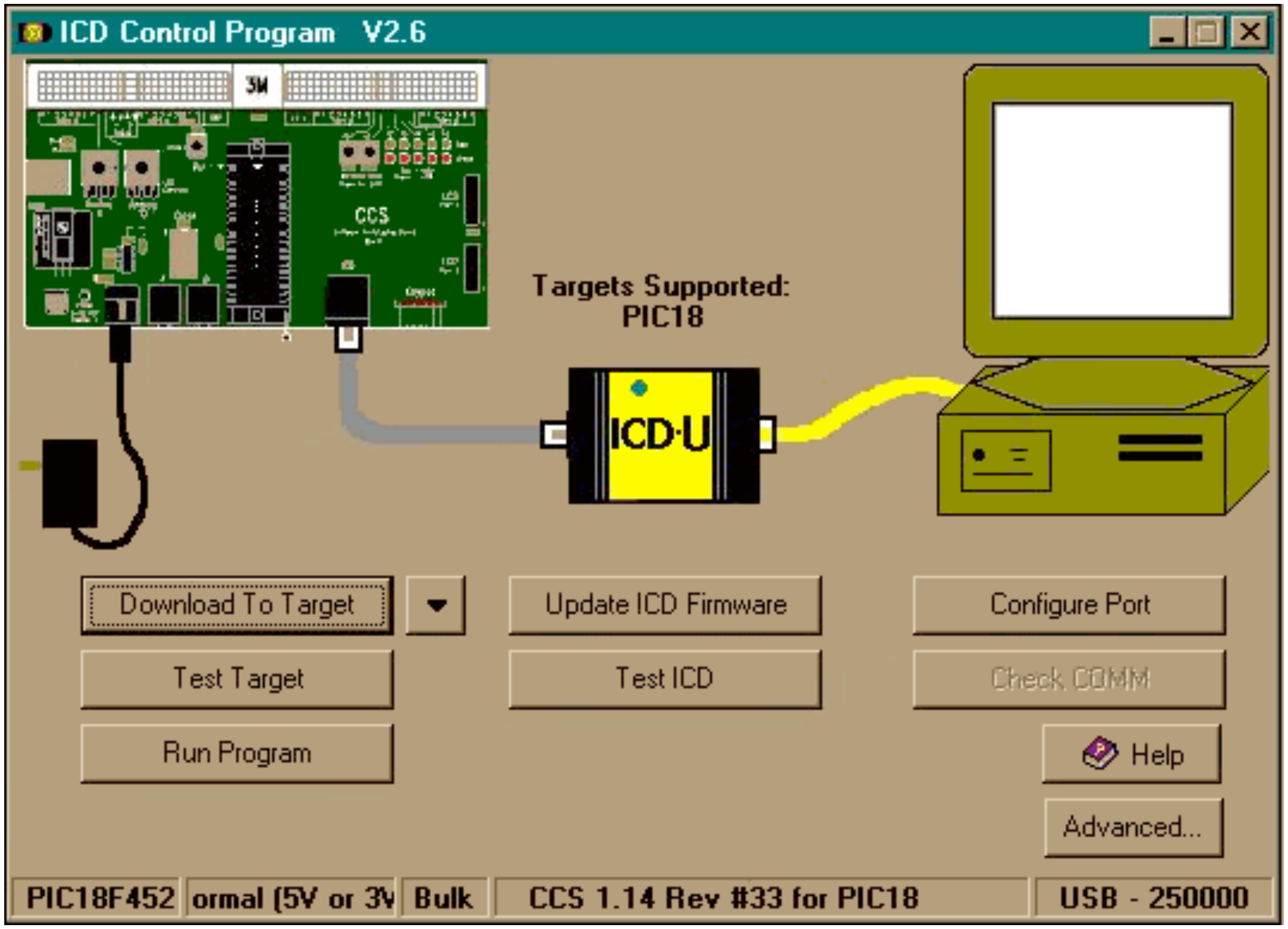
Name	Enabled	Network Address	Listener Port	Default Virtual Server
admin-listener	true	0.0.0.0	4848	__asadmin
http-listener-2	false	0.0.0.0	8181	server
http-listener-1	true	0.0.0.0	8080	server

- Don’t try to build a working prototype.

-Don’t try to build the application before you know what is required.
-Drawing the interface using a software tool is easy but that isn’t sufficient to prototype it.
-The hard part is linking the interface components to the code. Code must be written and it must be linked to the interface components. It is much more time consuming than paper prototyping and is much more difficult to change.
-No matter how fancy the interface tool is, it isn’t as easy to use or change as paper.
-The coding effort with paper is always zero.

- Other Benefits.

-An roughly drawn prototype encourages the user to provide feedback.
-Since it seems like it is still a work in progress they are more comfortable giving feedback.
-A slick, finished looking diagram can make the user think that everything has been decided and they shouldn’t suggest anything too drastic.
-Users are encouraged to participate by the rough appearance prototype.



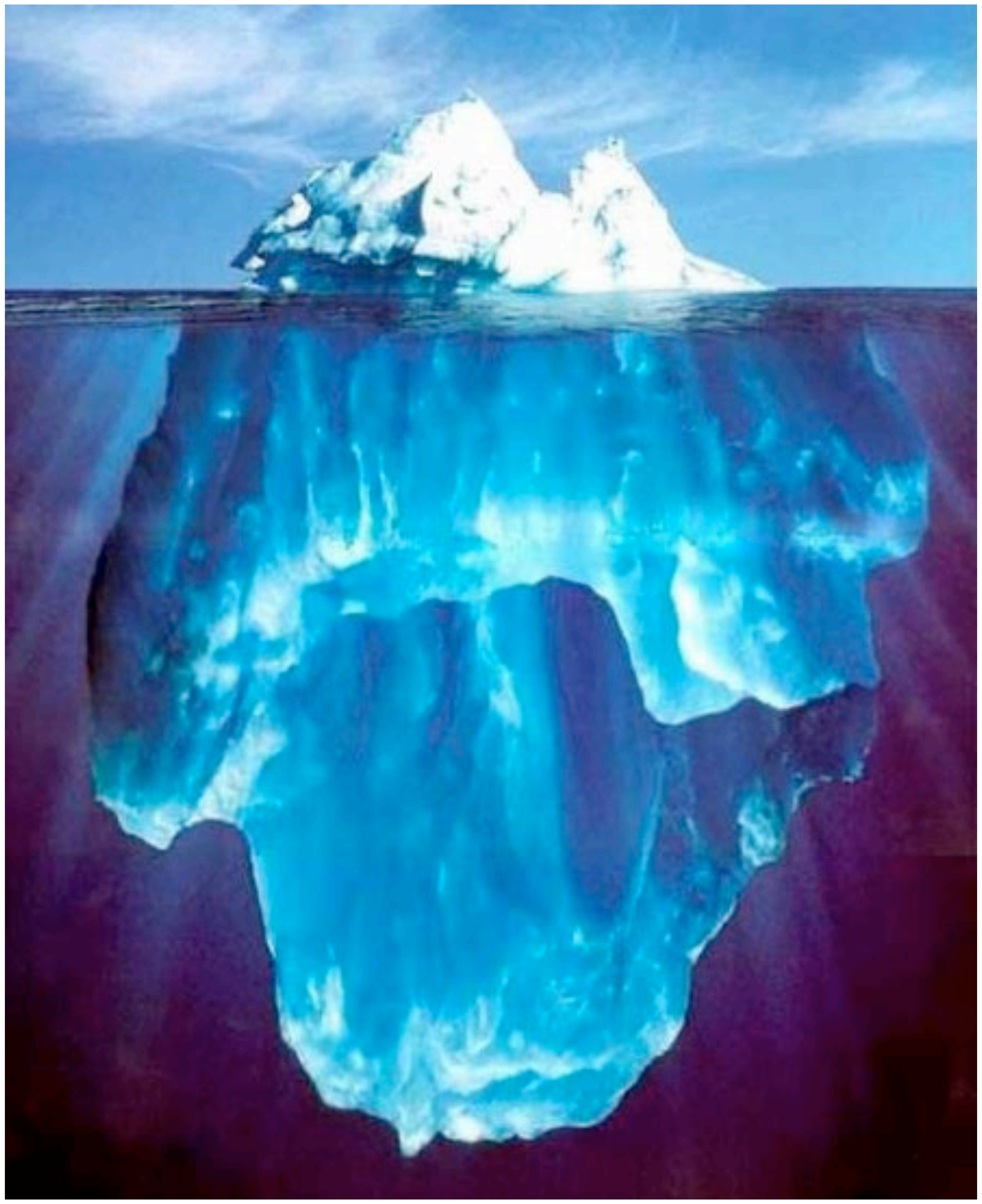
PIC18F452 | ormal (5V or 3V | Bulk | CCS 1.14 Rev #33 for PIC18 | USB - 250000

- Other Benefits.

-The user wont be encouraged to provide feedback about minor visual problems.
-When something appears finished the minor flaws stand out and will draw the user’s attention. Comments like, “those don’t line up” or “I don’t like that shade of green” are more likely to occur.

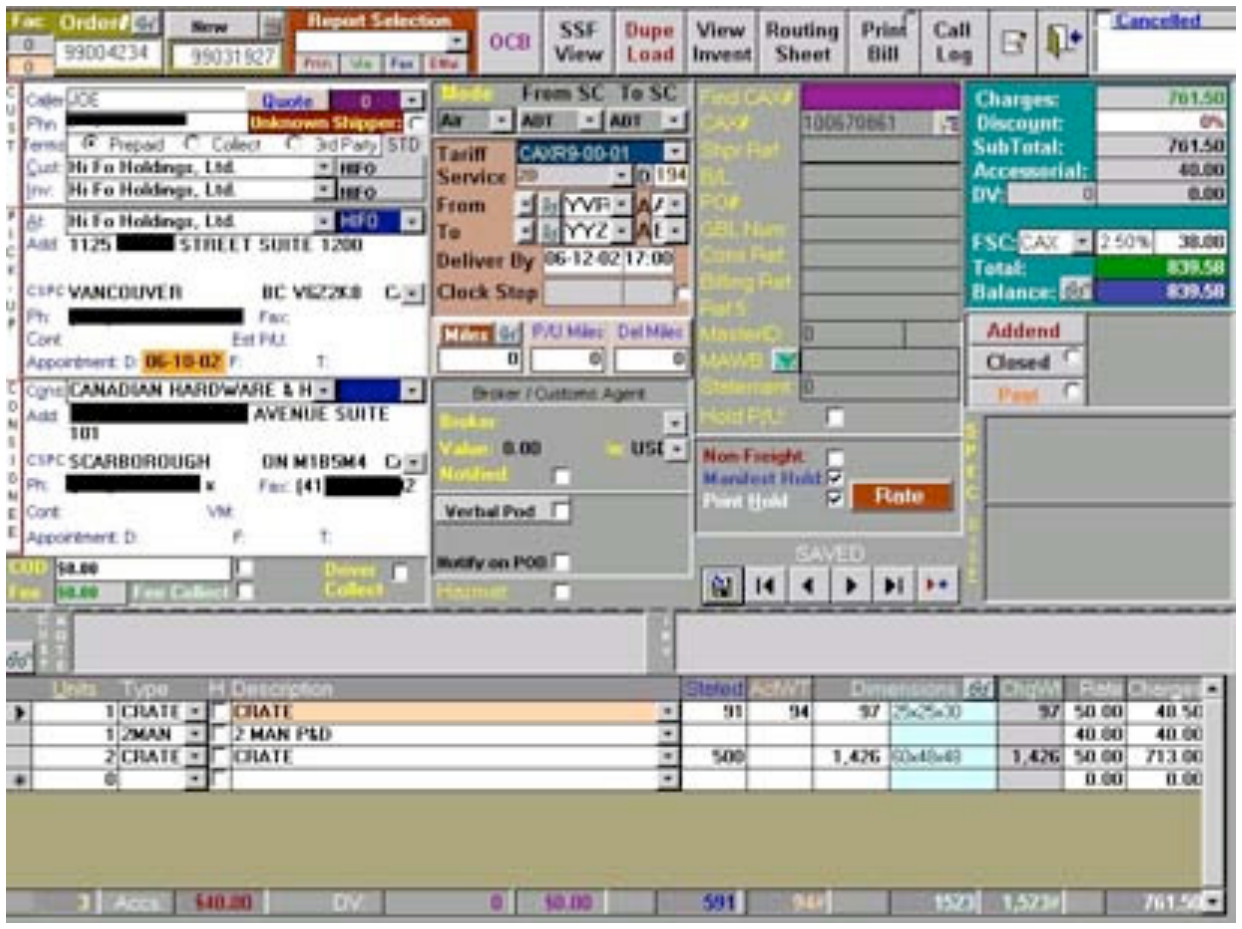
- Other Benefits.

-The user wont like the interface too much. If the interface looks too good then the users may resist changing it. They can resist change even if it is required.



- The Iceberg Secret.

-From Joel Spolsky.
-Just as an iceberg is 90% underwater, most of the work which goes into a software application cannot be seen. The interface accounts for about 1-5% of the development effort.
-This is something that non-programmers do not understand.



Fac: 0 0 Order# 99004234 New 99031927

Caller JOE Quote 0
Phn [redacted]
Terms (•) Prepaid ( ) Collect ( ) 3rd Party STD
Cust Hi Fo Holdings, Ltd. HIFO
Inv: Hi Fo Holdings, Ltd. HIFO
At: Hi Fo Holdings, Ltd. HIFO
Add 1125 [redacted] STREET SUITE 1200
CSPC VANCOUVER BC V6Z2K8 C
Ph: [redacted] Fax:
Cont Est P/U:
Appointment: D: 06-10-02 F: T:
Cgne CANADIAN HARDWARE & H
Add [redacted] AVENUE SUITE 101
CSPC SCARBOROUGH ON M1B5M4 C
Ph: [redacted] x Fax: (41[redacted]2
Cont VM:
Appointment: D: F: T:
COD $0.00
Fee $0.00 Fee Collect
Driver Collect

Mode Air From SC ADT To SC ADT
Tariff CA/R9-00-01
Service 20 D 194
From YVR
To YYZ
Deliver By 06-12-02 17:00
Clock Stop
Miles 0 P/U Miles 0 Del Miles 0
Broker:
Value: 0.00 USD

CA# 100670861
MasterID 0
Statement 0
Hold P/U
Non Freight
Manifest Hold ✓
Print Hold ✓

Charges: 761.50
Discount: 0%
SubTotal: 761.50
Accessorial: 40.00
DV: 0 0.00
FSC: CAX 2.50% 38.08
Total: 839.58
Balance: 839.58

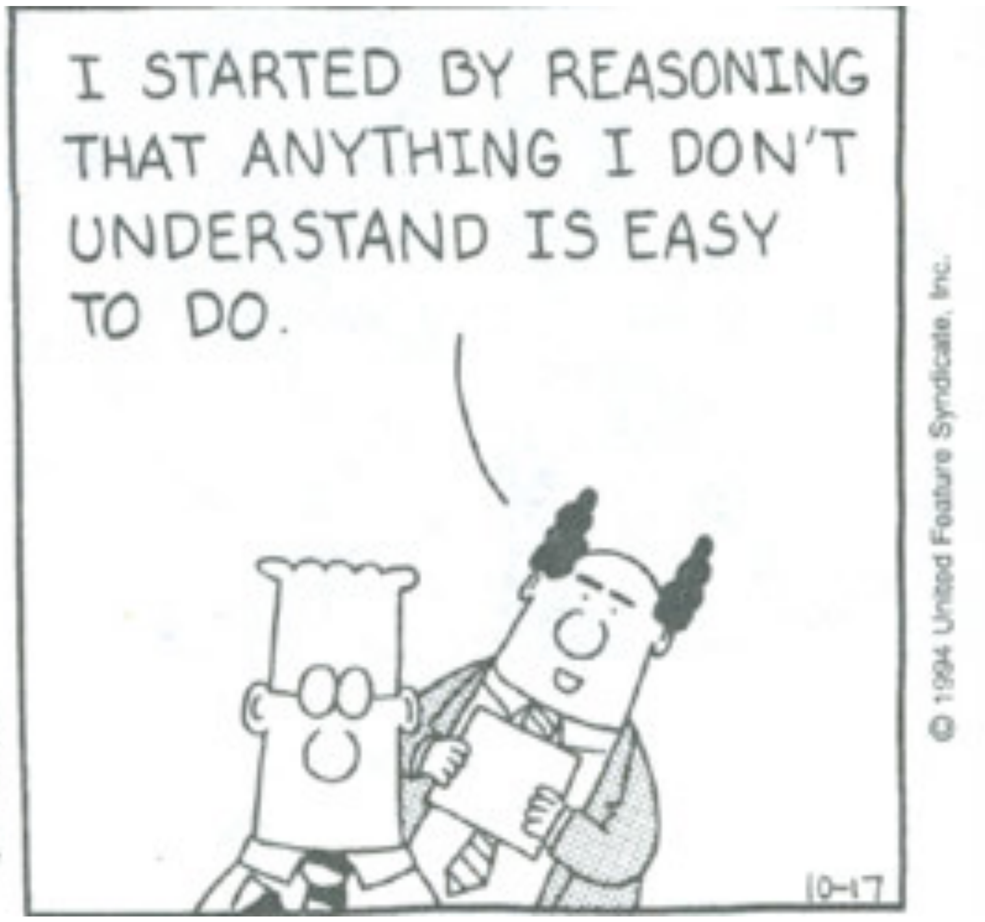
Units	Type	H	Description	Stated	ActWT	Dimensions		ChgWt	Rate	Charges
1	CRATE		CRATE	91	94	97	25x25x30	97	50.00	48.50
1	2MAN		2 MAN P&D						40.00	40.00
2	CRATE		CRATE	500		1,426	60x48x48	1,426	50.00	713.00
0									0.00	0.00

3 Accs $40.00 DV 0 $0.00 591 94# 1523 1,523# 761.50

Important Corollary One

- If you show a nonprogrammer a screen which has a user interface that is 90% worse, they will think that the program is 90% worse.

-If you have 100% of the functionality finished but the interface looks incomplete the client will complain about how poor it looks.


I STARTED BY REASONING THAT ANYTHING I DON'T UNDERSTAND IS EASY TO DO.
S. Adams
© 1994 United Feature Syndicate, Inc.
10-17

Important Corollary Two

- If you show a nonprogrammer a screen which has a user interface that is 100% beautiful, they will think the program is almost done.

-People who aren't programmers think the interface is the program. If it looks good then it must be good.
-If it takes you a long time to finish the functionality they wont see what you are doing and think it is nothing.

- Other Prototyping Benefits - Creativity.

-More creativity for the designers. They can get an opportunity to radically redesign the interface once they have had some experience with it. It helps them build an understanding of the system and lets them develop different ideas.

- Other Prototyping Benefits - Multidisciplinary.

-Multidisciplinary teams can participate. Customer support representatives and trainers can provide insights into what customers find confusing.
-Anyone who has knowledge of the subject area can contribute. Such as technical writers, marketers, trainers, etc.
-Paper prototyping facilitates incorporating many people’s ideas.
-It allows early communication across disciplines.

- Other Prototyping Benefits - Solves Arguments.

-Arguments based on opinions on the best way for users to do something can waste a lot of time. They are nasty disagreements that are usually based on unspoken assumptions and where there is no good evidence either way.
-A paper prototype allows you to determine which assumptions about the interface are true. The argument turns into, “we’ll test it with some users.”

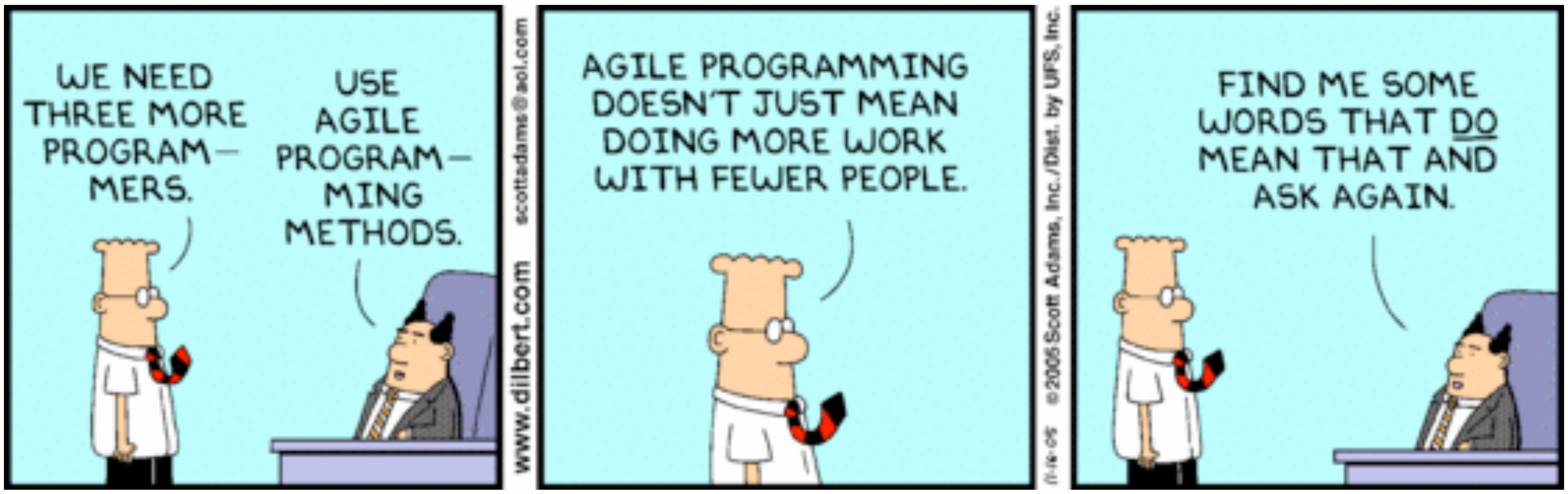
Useful Supplies

- something to hold the prototype, such as cardboard
- paper, post-it notes
- markers, a highlighter
- scissors
- tape
- glue stick
- overhead transparencies

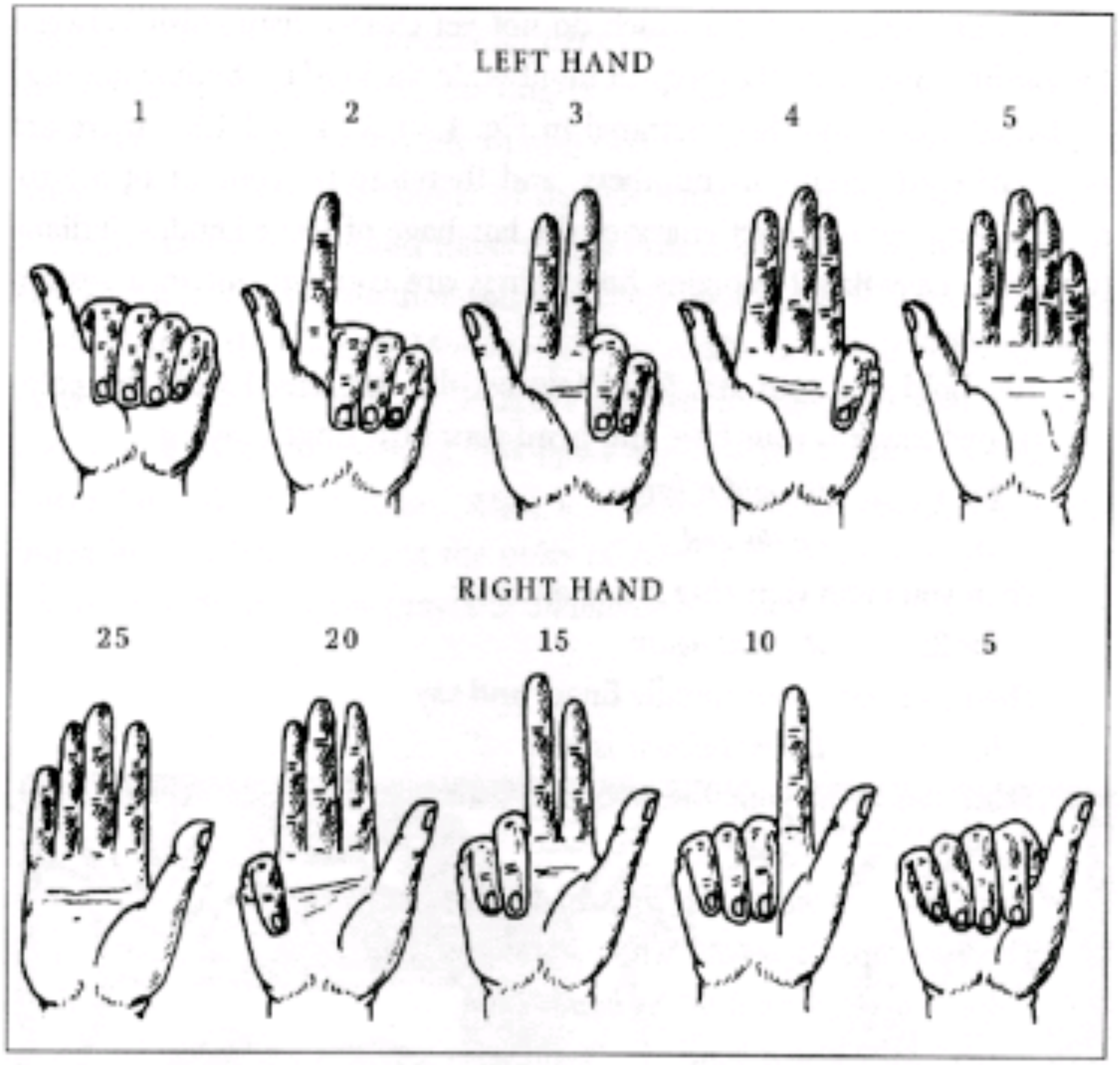
-The overheads can be placed over the prototype so the user can write text where they would type.

References

- Paper Prototyping: The Fast and Easy Way to Design and Refine User Interfaces, by Carolyn Snyder, Morgan Kaufman Publishers, 2003.
- Joel Spolsky web site: www.joelonsoftware.com


WE NEED THREE MORE PROGRAM-MERS.
USE AGILE PROGRAM-MING METHODS.
AGILE PROGRAMMING DOESN'T JUST MEAN DOING MORE WORK WITH FEWER PEOPLE.
FIND ME SOME WORDS THAT DO MEAN THAT AND ASK AGAIN.
www.dilbert.com scottadams@aol.com
11-16-05 © 2005 Scott Adams, Inc./Dist. by UFS, Inc.

Agile Software Development



- Iterative Software Development

-Iterative and incremental development.
-Requirements and solutions evolve through collaborative teams.

- Reaction to previous methods.

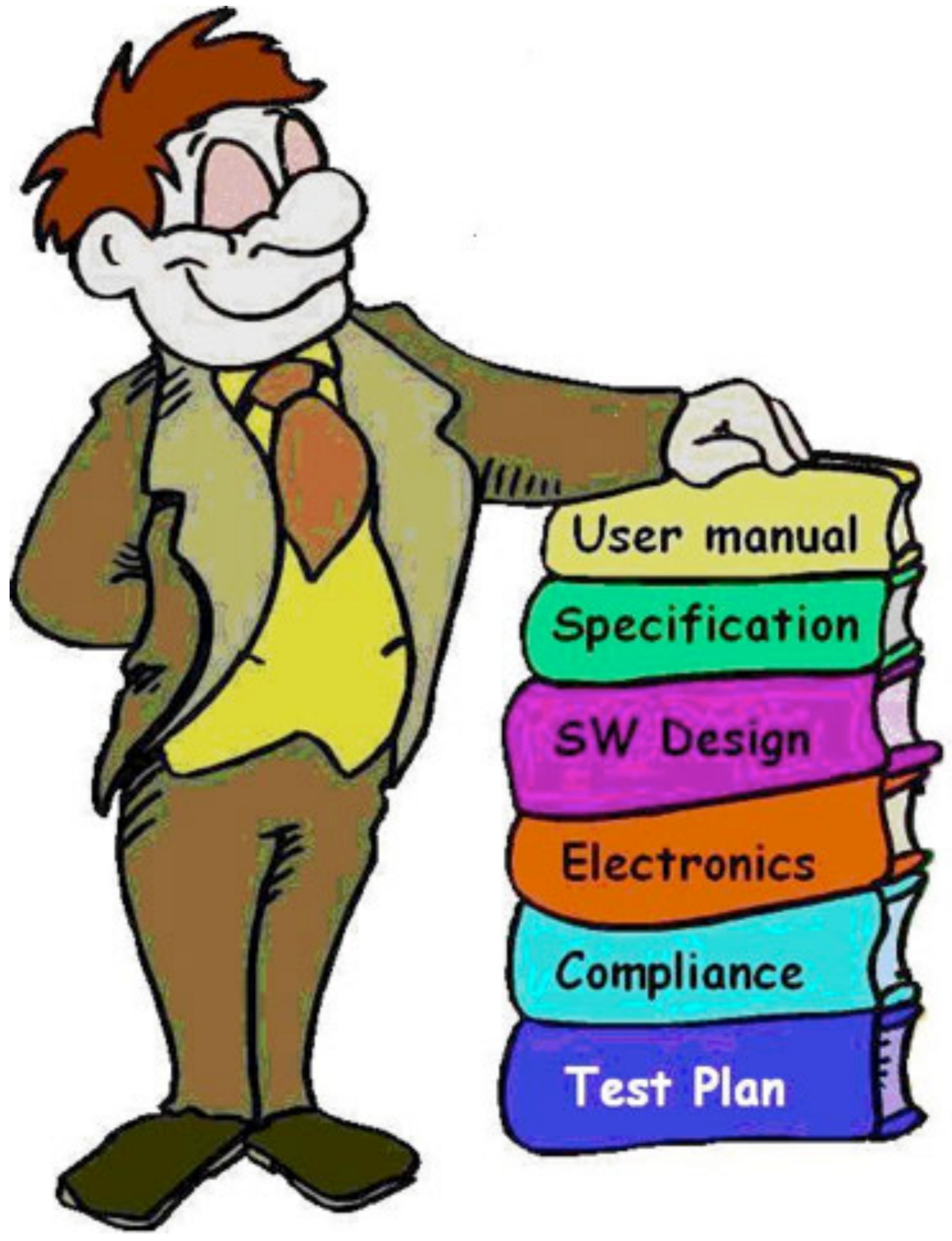
-A response to the heavy weight, non agile, methods such as Waterfall.
-It’s creators suggest that it is a return to software development methodologies from earlier.

weblogcartoons.com
SHUT UP
SHUT UP
SHUT UP
SHUT UP
SHUT UP
SHUT UP
BLAH BLAH BLAH BLAH BLAH BLAH BLAH BLAH BLAH BLAH BLAH BLAH BLAH BLAH BLAH BLAH

Agile Values

- Individuals and interactions over process and tools.

-Agile developers identified four values which they believe are most important for software development.
-The first is that the active involvement of people during development is more important than rigidly following a process.



- Working software over comprehensive documentation.

-Software is the most important part of the development so it should take precedence over documenting the process.

- Customer collaboration over contract negotiation.

-It is better to have customers involved during the development process then it is to try to negotiate all of the possible options before development begins.

"Look what I found in the dumpster! A perfectly good business plan!"

- Responding to change over following a plan.

-Following a plan should not be more important than changing to a more correct solution.
-A plan should not be used as an excuse to produce an incorrect solution.

Twelve Principles

- Customer satisfaction through rapid delivery of useful software.
- Welcome changing requirements, even late in the development.

-The creators of the agile manifesto also proposed twelve important principles.

- Working software is delivered frequently (weeks instead of months).
- Working software is the principle measure of progress.

- Sustainable development, able to maintain a constant pace.
- Close, daily cooperation between business people and developers.

- Face-to-face conversation is the best form of communication.
- Projects are built around motivated individuals, who should be trusted.

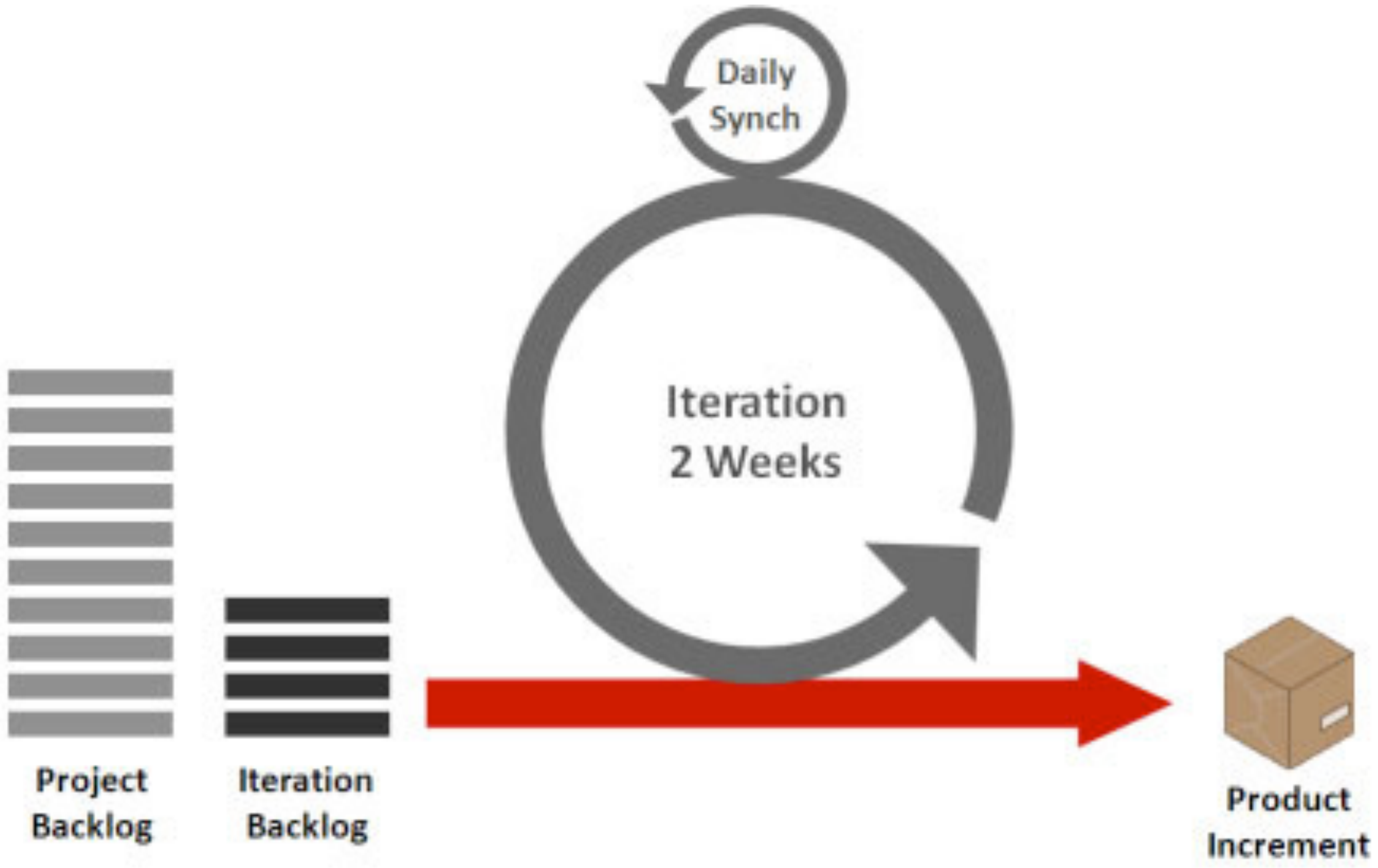
- Continuous attention to technical excellence and good design.
- Simplicity.

- Self-organizing teams.
- Regular adaption to changing circumstances.

WE'RE GOING TO TRY SOMETHING CALLED EXTREME PROGRAM-MING.
FIRST, PICK A PARTNER. THE TWO OF YOU WILL WORK AT ONE COMPUTER FOR FORTY HOURS A WEEK.
THE NEW SYSTEM IS A MINUTE OLD AND I ALREADY HATE EVERYONE.
scottadams@aol.com
www.dilbert.com
© 2002 United Feature Syndicate, Inc.
1/9/03

- Agile Methods.

-There are many agile methods. These include Extreme Programming and Scrum.
-They promote development, teamwork, collaboration, and adaptability throughout the software lifecycle.
-Agile methods break tasks into smaller part with minimal planning and do not directly involve long term planning.

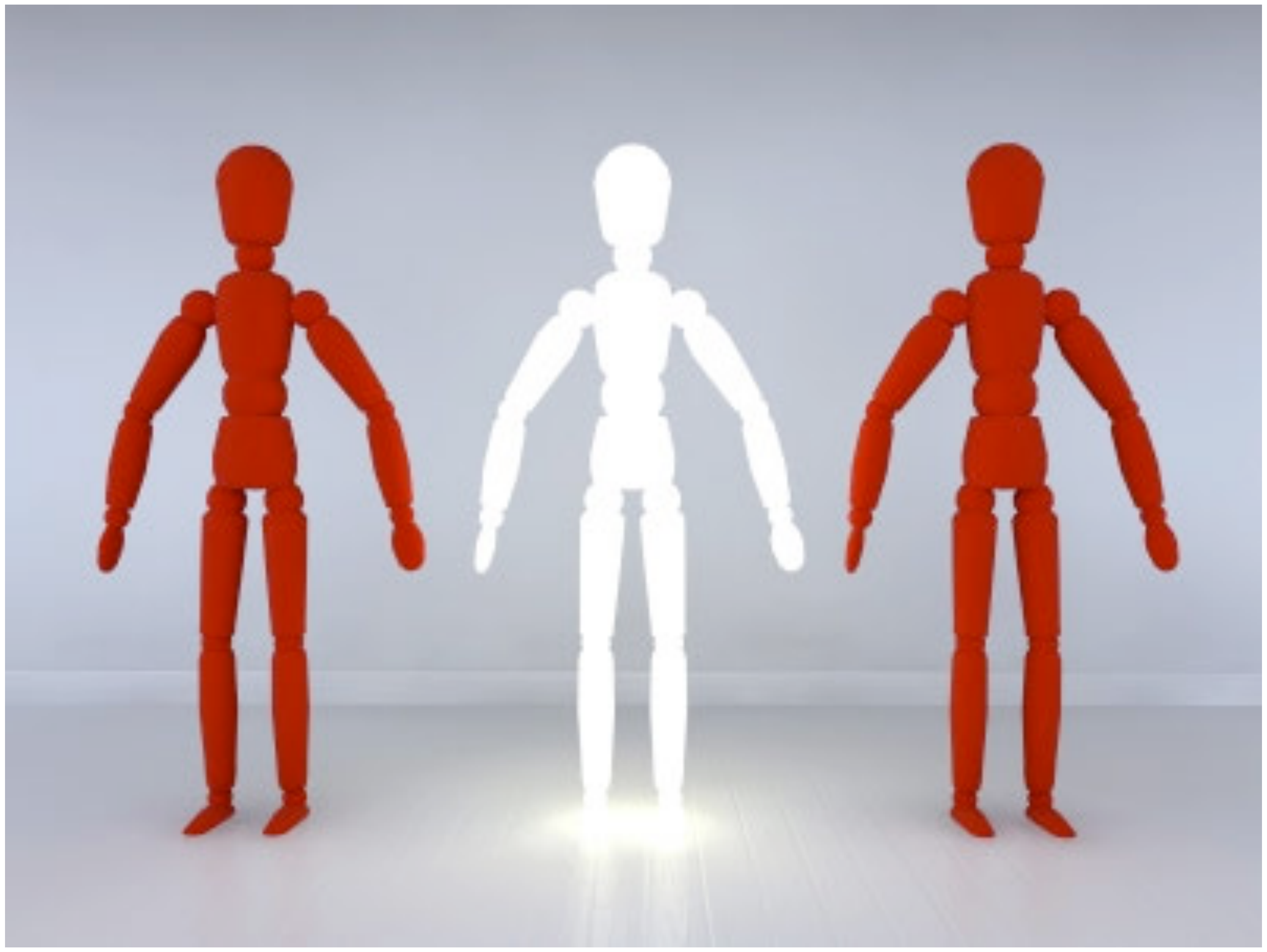


- Iterations

-Each iteration involves the team working through a full software development life cycle including planning, requirements analysis, design, coding, and unit testing.
-The final stage is acceptance testing where the product is demonstrated to the stakeholders.
-Iterations generally last from one to four weeks.
-Risk is minimized because of the short timespan and clear goals.
-One iteration may not be a completed product but the goal is to have a releasable product at the end of an iteration. Multiple iterations will generally be needed to complete a product.

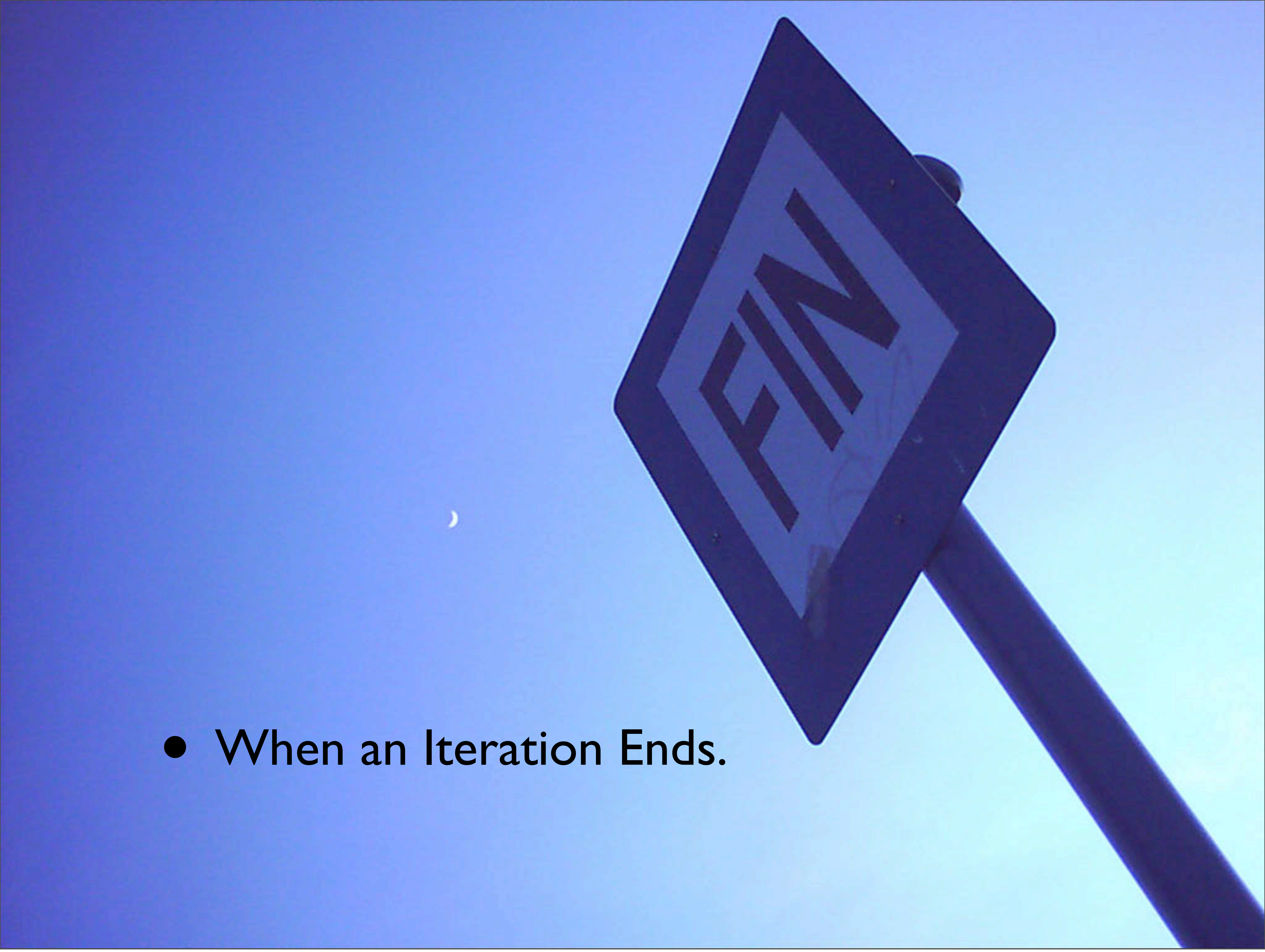
- Teams

-Teams normally organize themselves without consideration of existing corporate hierarchy.
-Team members take responsibility for the tasks which the current iteration requires.
-A team normally contains five to nine people which is small enough to simplify communications and collaboration.
-Larger projects can involve multiple teams which can work on a common goal or on different parts of an effort.



- The Customer Representative.

-Each team must have a customer representative who is appointed by the stakeholders to act on their behalf.
-They must be available during the iteration to answer domain specific questions.



- When an Iteration Ends.

-The stakeholders and the customer representative review the progress and re-evaluate priorities to ensure the project meets the goals.

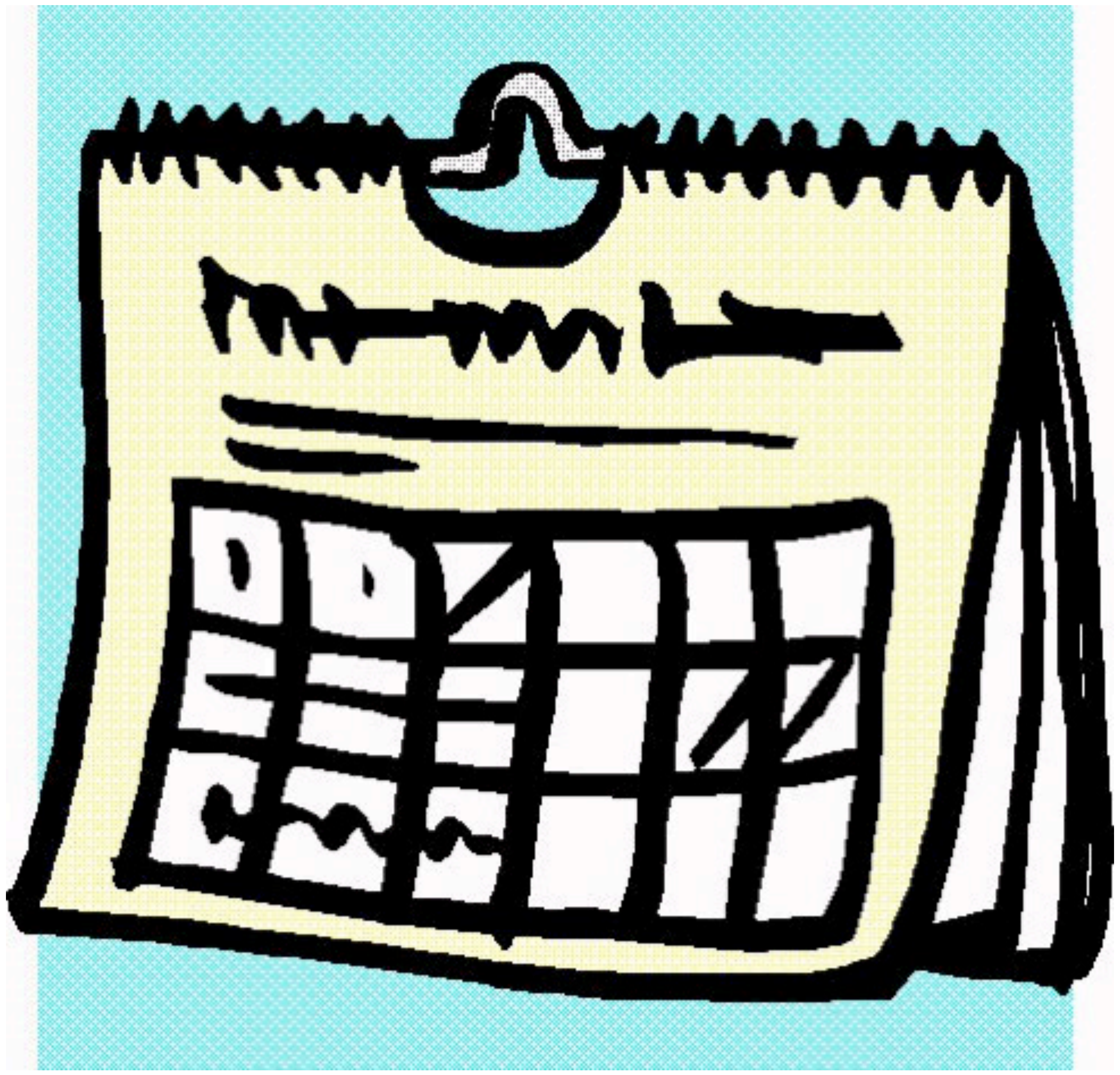
LET'S HAVE A LITTLE PREMEETING TO PREPARE FOR THE MEETING TOMORROW.
WHOA! DO YOU THINK IT'S SAFE TO JUMP RIGHT INTO THE PREMEETING WITHOUT PLANNING IT?
OKAY, LET'S GET THIS PRELIMINARY PREMEETING MEETING GOING.
YOU THINK YOU'RE FUNNY, BUT YOU'RE NOT.
3/31/98 © 1998 United Feature Syndicate, Inc.(NYC)

- Daily Meetings.

-Most agile methods have brief and daily, face to face team meetings.
-Team members report what they did yesterday, what they intend to do today, and what problems have appeared.
-This exposes problems when they arise.

- Adaptive versus Predictive Methods.

-Agile development is no unplanned or undisciplined. The teams may use highly disciplined techniques.
-It is more accurate to say they use adaptive methods to develop software instead of attempting to predict all development choices before the software is written.



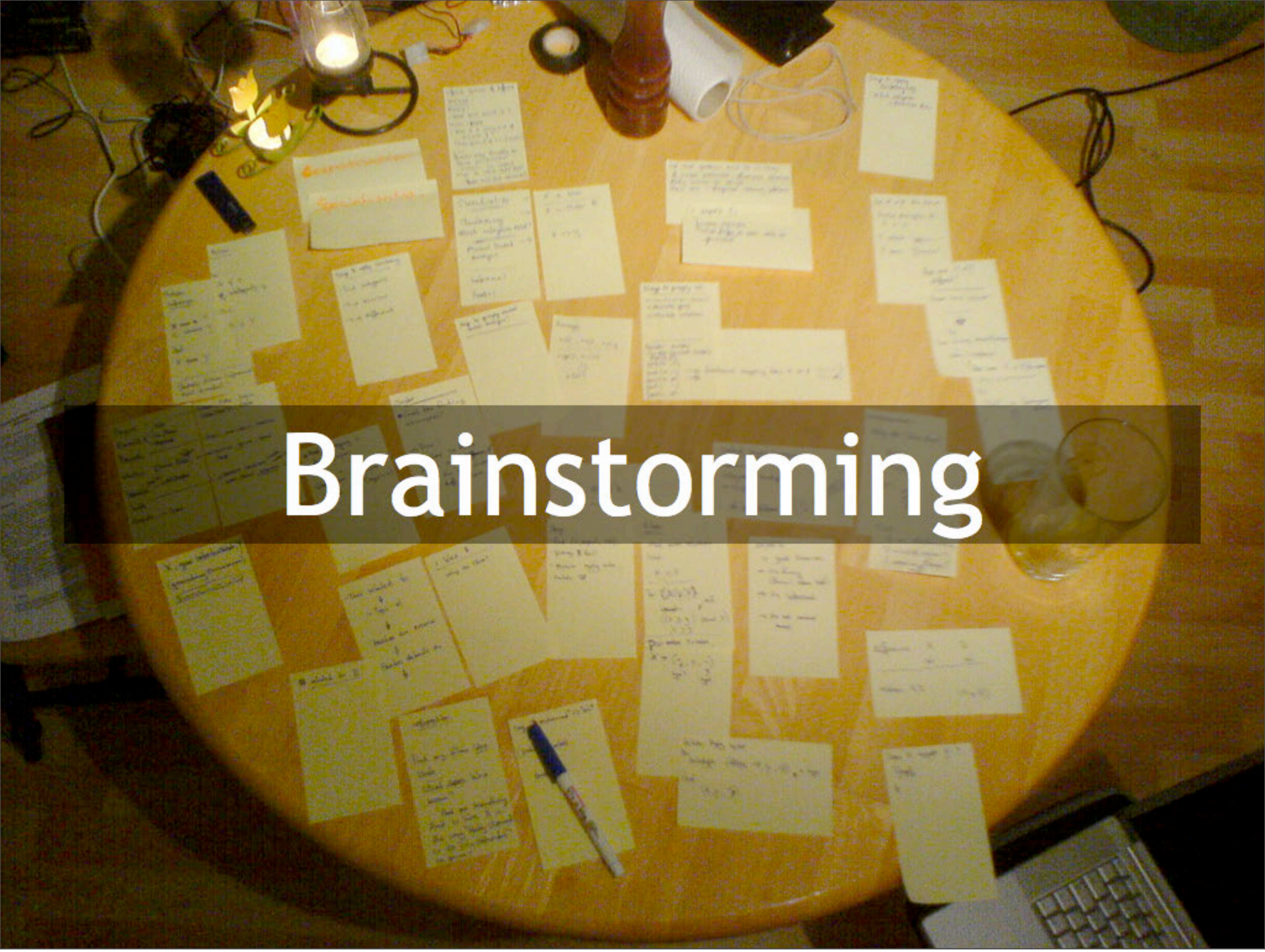
- Adaptive Methods.

-Adaptive methods focus on changing focus quickly when reality changes or when it becomes more clearly defined. When the needs of the project change, the team changes as well.
-The further away a date is the more vague an adaptive team will be about what will happen on that date.

WE'RE GOING TO TRY SOMETHING CALLED AGILE PROGRAMMING.
THAT MEANS NO MORE PLANNING AND NO MORE DOCUMENTATION. JUST START WRITING CODE AND COMPLAINING.
I'M GLAD IT HAS A NAME.
THAT WAS YOUR TRAINING.
www.dilbert.com
scottadams@aol.com
© 2007 Scott Adams, Inc./Dist. by UFS, Inc.
© Scott Adams, Inc./Dist. by UFS, Inc.

- What agile is not.

-It is not an unstructured free for all where the developers do what they feel is right.
-It is not an excuse to ignore all structure and just write some code. It involves a clearly defined and rigid process which is controlled.
-The flexibility and acceptance of change does not mean that it is trivial.



Brainstorming

-It is meant to be fun.

- Brainstorming

-Group creativity technique.
-Used to generate a large number of ideas to solve a problem.

- Good Features.

-Boosts morale.
-Enhanced work enjoyment.
-Improves team work.

- Not so good features.

-There is **no evidence** it produces more or better solutions than an individual.
-Can be distracting.
-Can lead t social loafing. When people work less hard because they are in a group. They feel their efforts are under appreciated, or if the task is not important, or if they think they can remain anonymous and not be evaluated.

RULES

1. YOU CAN....
2. YOU CAN'T...
3. YOU CAN....
4. YOU CAN'T

- Four Rules.

-The four rules are to reduce social inhibitions to participating and to stimulate ideas and creation in the group.

- Rule 1 - Quantity will lead to quality.

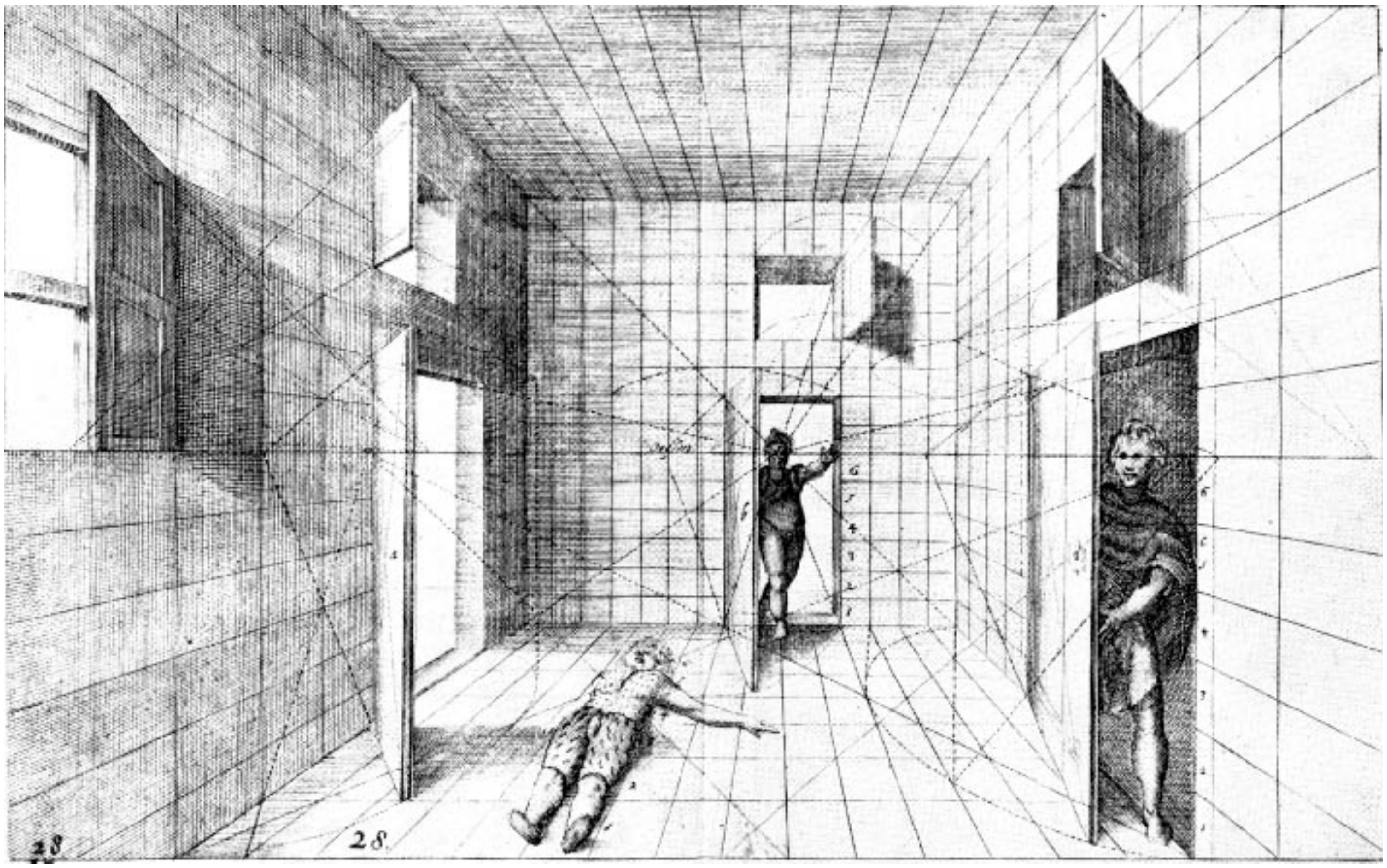
-The ideas is that the larger the number of ideas created, the greater the chance that a radical and effective solution will emerge.
-Is quantity really the route to quality?

- Rule 2 - Don't be critical.

-Criticism should be withheld until after the ideas have been generated.
-Participants should focus on generating ideas.
-By suspending judgement the participants will feel free to generate unusual ideas.

- Rule 3 - Welcome unusual ideas.

-To generate a long list of ideas the unusual must be accepted.
-They can be generated by looking at ideas from a **new perspective** and by **suspending assumptions.**
-The new way of looking at the problem may provide better solutions.
-It can be more fun and productive if wildly impractical ideas are presented.



- Looking at the problem with a new perspective.

-If the product is currently a boxed product can, it be made a web service?
-If it is a niche market, can it be expanded?
-If it requires specialized knowledge, can it be made simple for a new audience?
-If it is used by artists can, it be remade for scientists or business people?
-If it is specific to one task, can it be changed to another or can more be added?

- Suspend assumptions.

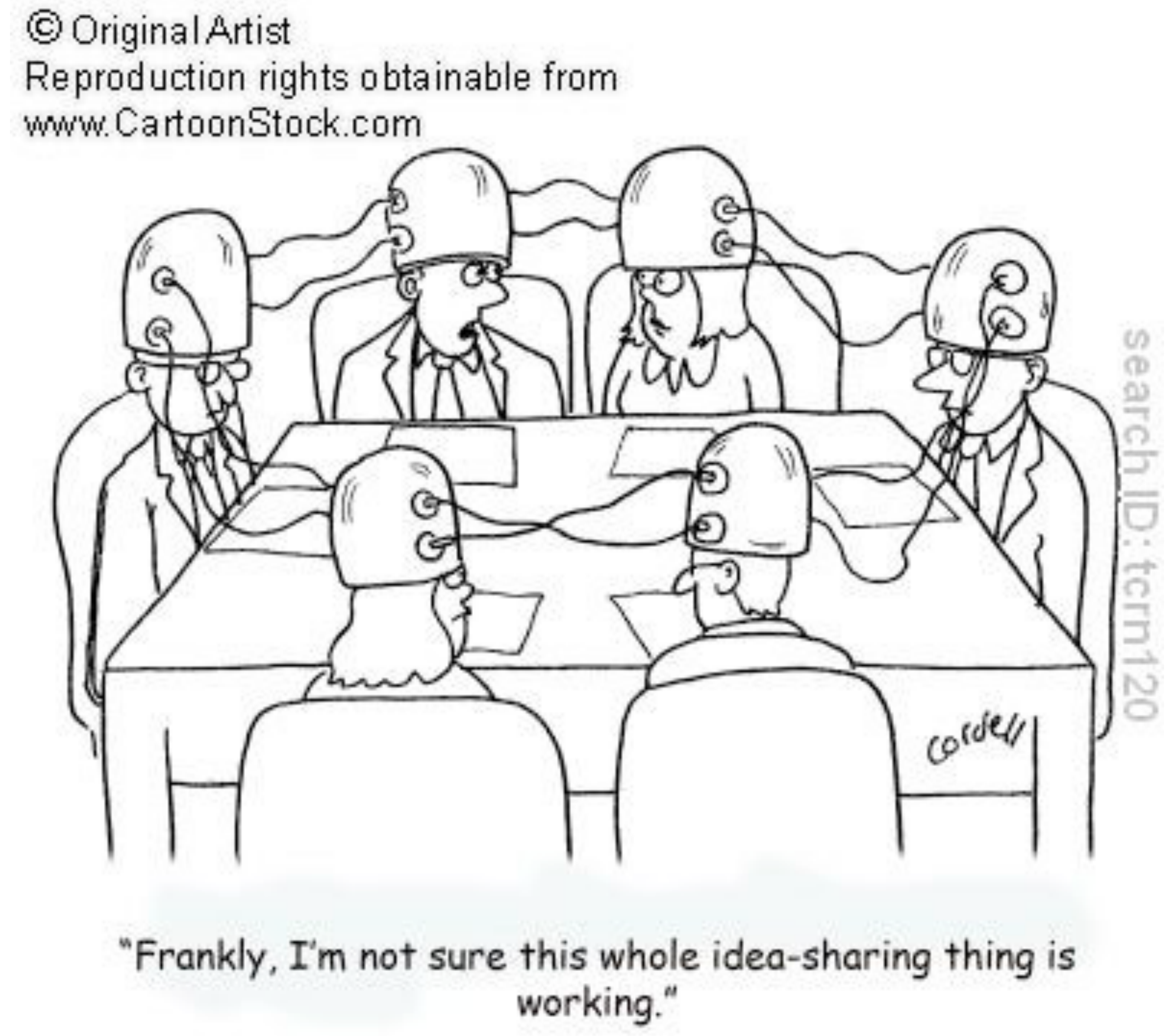
-We only write software for a particular type of client.
-It is impossible to make the software do something.
-There is no one who would use the software on that platform, in that industry.
-Only technical or non-technical people will use the product.
-Some group has no need for the product.
-Selling software is our only business.

- Rule 4 - Combine new ideas.

-The ideas may combine to form new and even better solutions.
-It builds ideas through associating different ideas with each other.

- Before the Brainstorming Meeting.

-Set a problem that is not too large to solve in the session.
-Make a question which is to be solved.
-The question should be specific. For example, “what does our product not currently do?”
-If the problem is too large it can be divided into subproblems. Each of these will have its own question.


© Original Artist
Reproduction rights obtainable from
www.CartoonStock.com
search ID: tcrn120
cordell
"Frankly, I'm not sure this whole idea-sharing thing is working."

- When You Call the Meeting.

-Provide information for the participants.
-Provide a description of the problem question and some sample solutions.
-Identify the participants for the meeting.
-From five to seven people are a good size for the group.
-Provide a time limit. Thirty minutes is often enough. The larger the group, the more time will be needed. Alternatively, provide an idea goal. This is a number of ideas which must be created before the meeting ends. Normally 50 or 100.

- During the Meeting.

-Someone facilitates (manages) the meeting and someone records ideas.
-If the group gets stuck then the facilitator suggests an idea in order to stimulate discussion.
-Ideas are not normally presented in a structured order. Participants present them as they think of them.

- During the Meeting.

-The person recording ideas may repeat their notes to ensure they understand the ideas.
-Managers may be discouraged from attending as it may cause people to feel uncomfortable presenting unusual ideas.
-At the end of the meeting the list is reviewed to ensure everyone understands it and duplicates are removed.

- A More Specific Brainstorming Method.

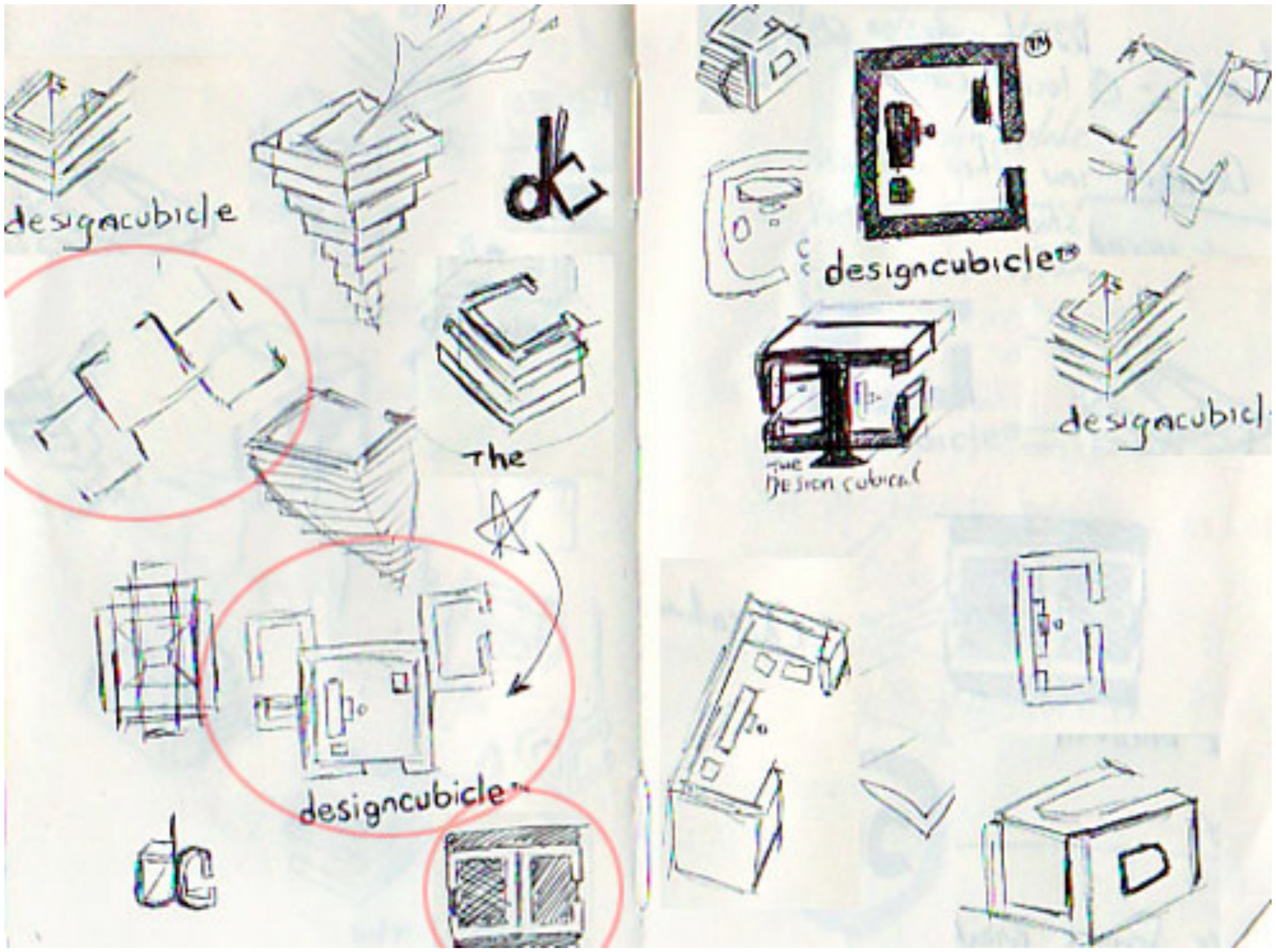
-It starts the same as previously mentioned with a 30 minute meeting and a goal of 50 or 100 ideas.
-Once it starts, participants present ideas which are written on a board or chart so everyone can see them.
-All ideas are accepted and written down. Laughing is encouraged but criticism is not.

- A More Specific Brainstorming Method.

-Once the time is up, select the five ideas which everyone agrees are the best. This can involve voting.
-Write down about five criteria which can be used to judge which ideas best solve the problem. These should start with the word "should," for example "it should be legal", "it should be finished by November 28th."

- A More Specific Brainstorming Method.

-Give each idea a score of zero to five depending on how well it meets the criteria.
-Once all ideas have a score for each idea, add up the scores.
-The ideas with the highest score is the best solution.
-Keep a record of the other ideas in case the best solution turns out to be infeasible.



- Visual Brainstorming.

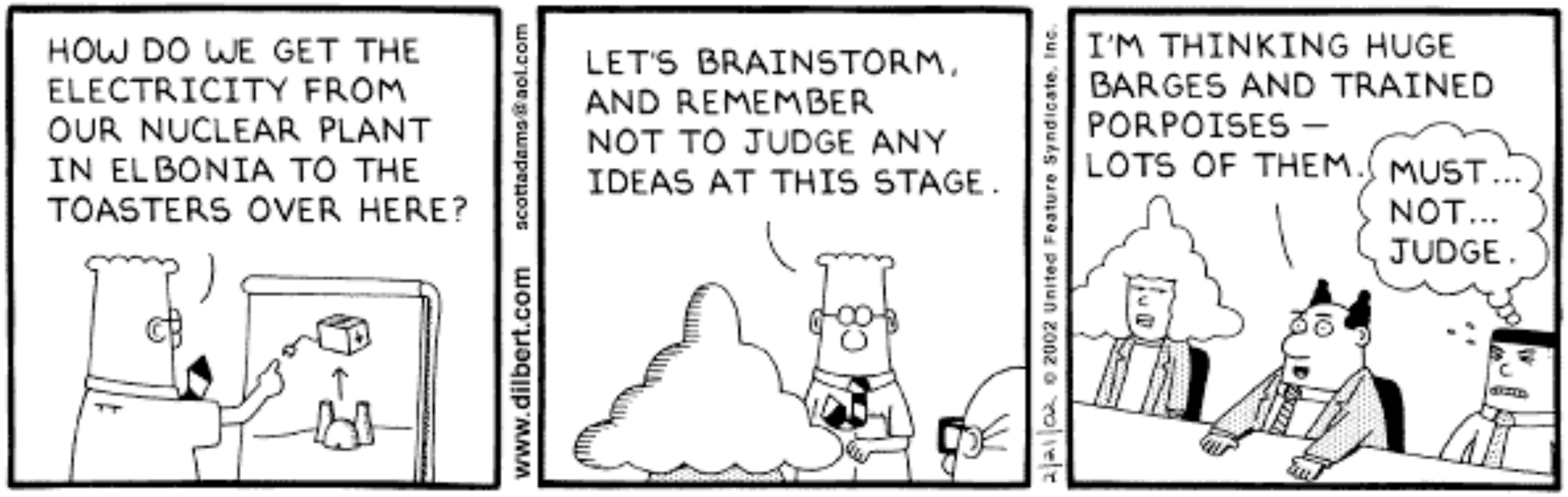
-A different approach which solves some of the problems found in regular brainstorming. These include criticizing ideas during the session, dominant personalities, fixating on a seemingly good idea too quickly, and the noisy chaotic environment of a session.
-The intention is to collaboratively generate ideas without a lot of speaking or writing.

- Visual Brainstorming.

-Items are used to imitate the product.
-This is much like paper prototyping but isn't restricted to interfaces.
-A question is posed at the start of the session.
-Instead of shouting out ideas, the team works to build a model of the product.
-Criticism is still prohibited.
-Talking is acceptable but any idea must be implemented in the model.

- Visual Brainstorming.

-The group make break into different teams if different ideas are being pursued.
-Once the model is completed the team presents the features and the logic behind the features.
-The ideas are then compiled into a report and evaluated.

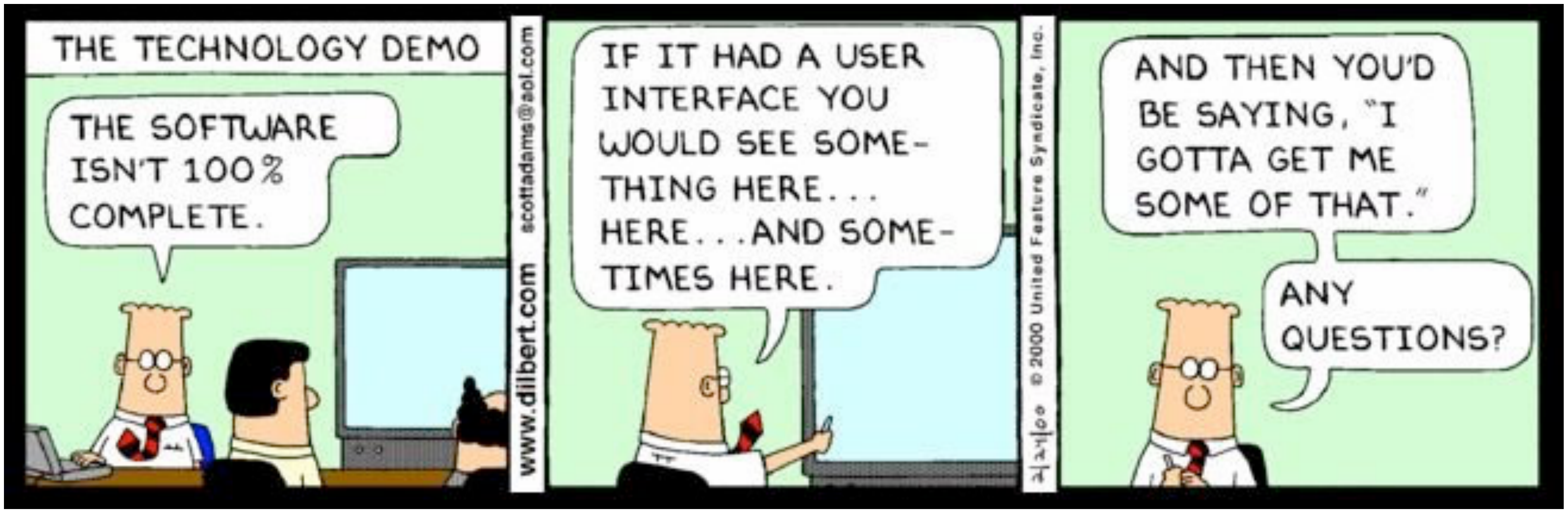

HOW DO WE GET THE ELECTRICITY FROM OUR NUCLEAR PLANT IN ELBONIA TO THE TOASTERS OVER HERE?
www.dilbert.com scottadams@aol.com
LET'S BRAINSTORM, AND REMEMBER NOT TO JUDGE ANY IDEAS AT THIS STAGE.
© 2002 United Feature Syndicate, Inc.
I'M THINKING HUGE BARGES AND TRAINED PORPOISES — LOTS OF THEM.
MUST... NOT... JUDGE.

- Visual Brainstorming.

The benefits of visual brainstorming:
-no one can avoid participating, if you aren't building something then it is obvious
-no one can dominate if everyone is working on part of the problem
-there are fewer distractions
-it is more difficult to be critical of other's work

References

- http://en.wikipedia.org/wiki/Brainstorming
- http://www.mindtools.com/brainstm.html
- http://www.jpb.com/creative/brainstorming.php
- http://www.jpb.com/creative/visual_brainstorming.php


THE TECHNOLOGY DEMO
THE SOFTWARE ISN'T 100% COMPLETE.
scottadams@aol.com
www.dilbert.com
IF IT HAD A USER INTERFACE YOU WOULD SEE SOMETHING HERE... HERE...AND SOMETIMES HERE.
© 2000 United Feature Syndicate, Inc.
2/24/00
AND THEN YOU'D BE SAYING, "I GOTTA GET ME SOME OF THAT."
ANY QUESTIONS?

Successful Software Designs

-What made some software successful?

LIVE FREE OR DIE
http://www.UNIX-systems.org
UNIX®
UNIX is a Registered Trademark of The Open Group

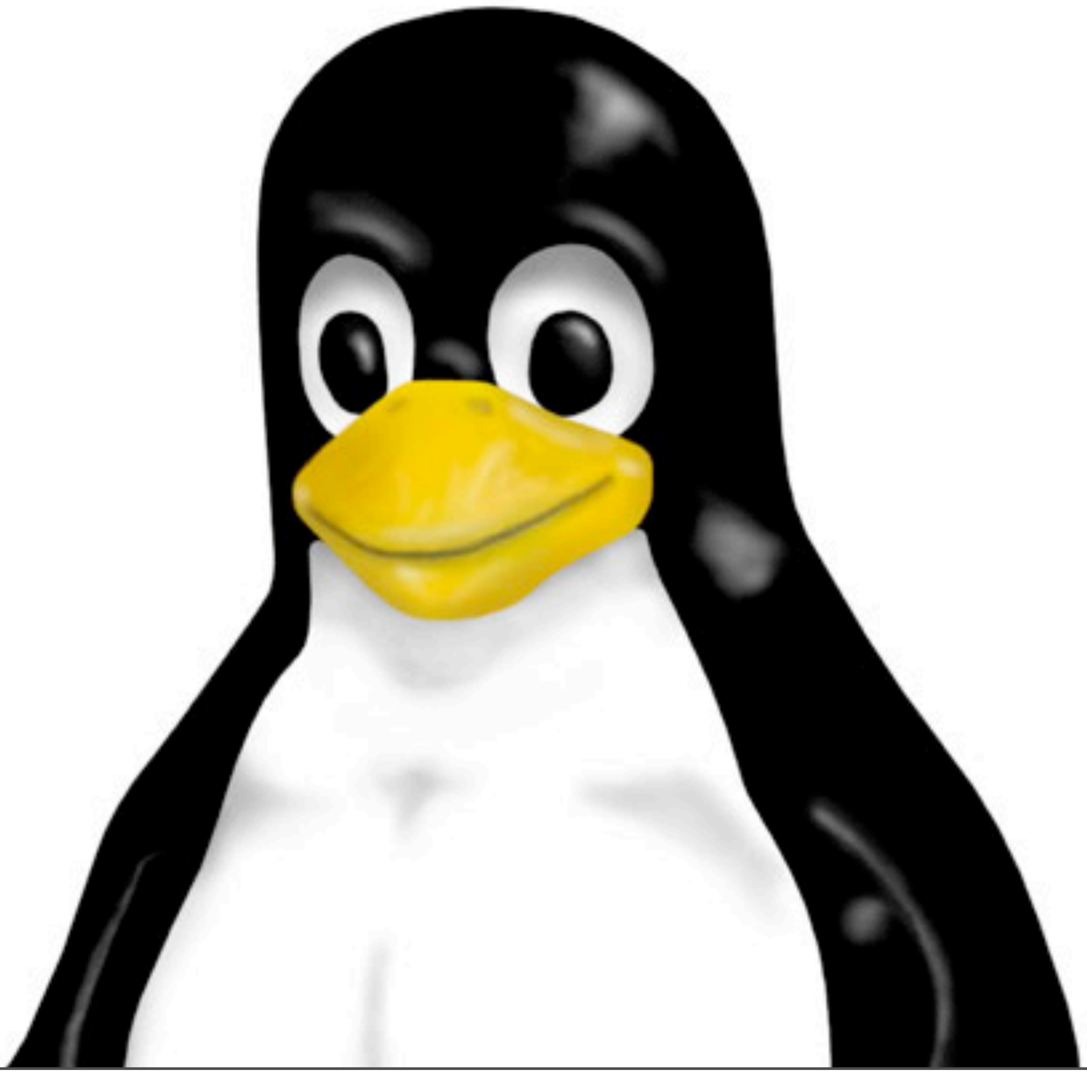
ubuntu
UNIX
gentoo linux
BSD
SUSE
Mandriva
MEPIS
Tru64
Linux
Mac
Universal
debian
SOLARIS
Novell
xandros
CentOS
slackware
Cygwin
KNOPPIX
fedora
redhat
SGI IRIX

- UNIX

-Ken Thompson and Dennis Ritchie and others, creators of Unix. Ritchie also created C at Bell Telephone Labs.
-Developed in 1969 at Bell Labs. Based on another operating system named Multics.
-Has been ported to almost every hardware platform.
-Open source versions such as Linux and BSD have become popular.
-Why is a 40 year old operating system still popular?

Unix was designed to be:

- Portable
- Multi-tasking
- Multi-user
- Time sharing

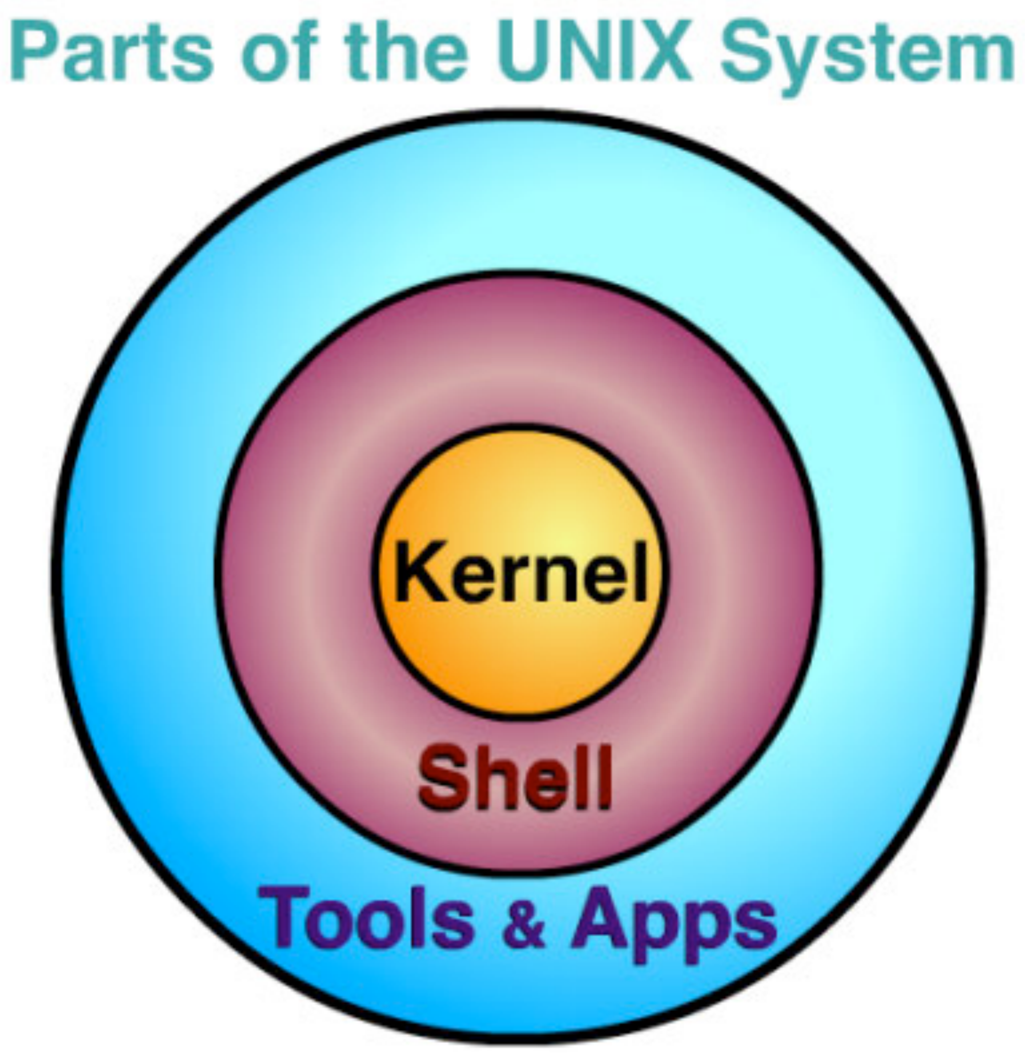


-The design ideas.
-Many were new at the time UNIX was created.
-It was written in a high level language instead of assembly language so it was easier to port to other platforms. Other operating systems did this (Multics, Burroughs) but UNIX popularized it.

The Unix philosophy:

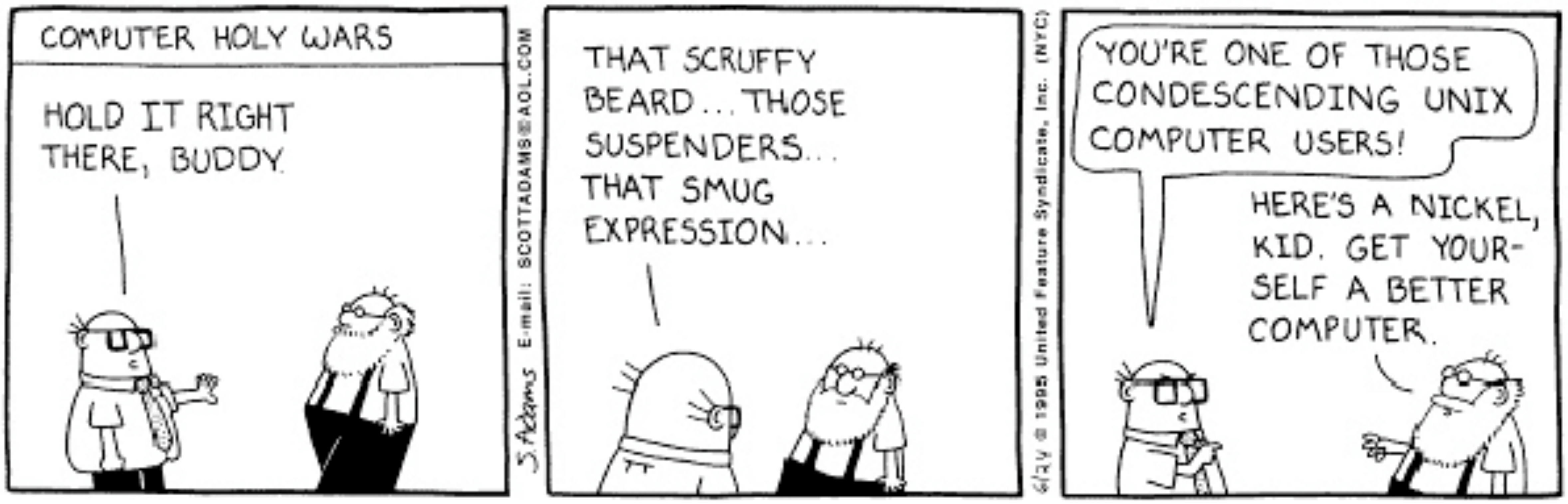
- store information as plain text
- hierarchical file system
- treating devices and some networking as files
- a large number of tools
- stringing together small programs through the shell and pipes

-The ideas which went into building it were useful and robust. They have demonstrated that they can be very powerful.
-Treating devices as files simplified access to them. It provided a uniform interface to printers, keyboards, disks, etc.
-Operating systems of the time often did not have hierarchical file systems. They divided disks up into sections that had a fixe number of levels, often only one.
-The shell allowed users to string together small programs into producer-consumer pipelines. Modularity and reusability became important concepts practices because of this.



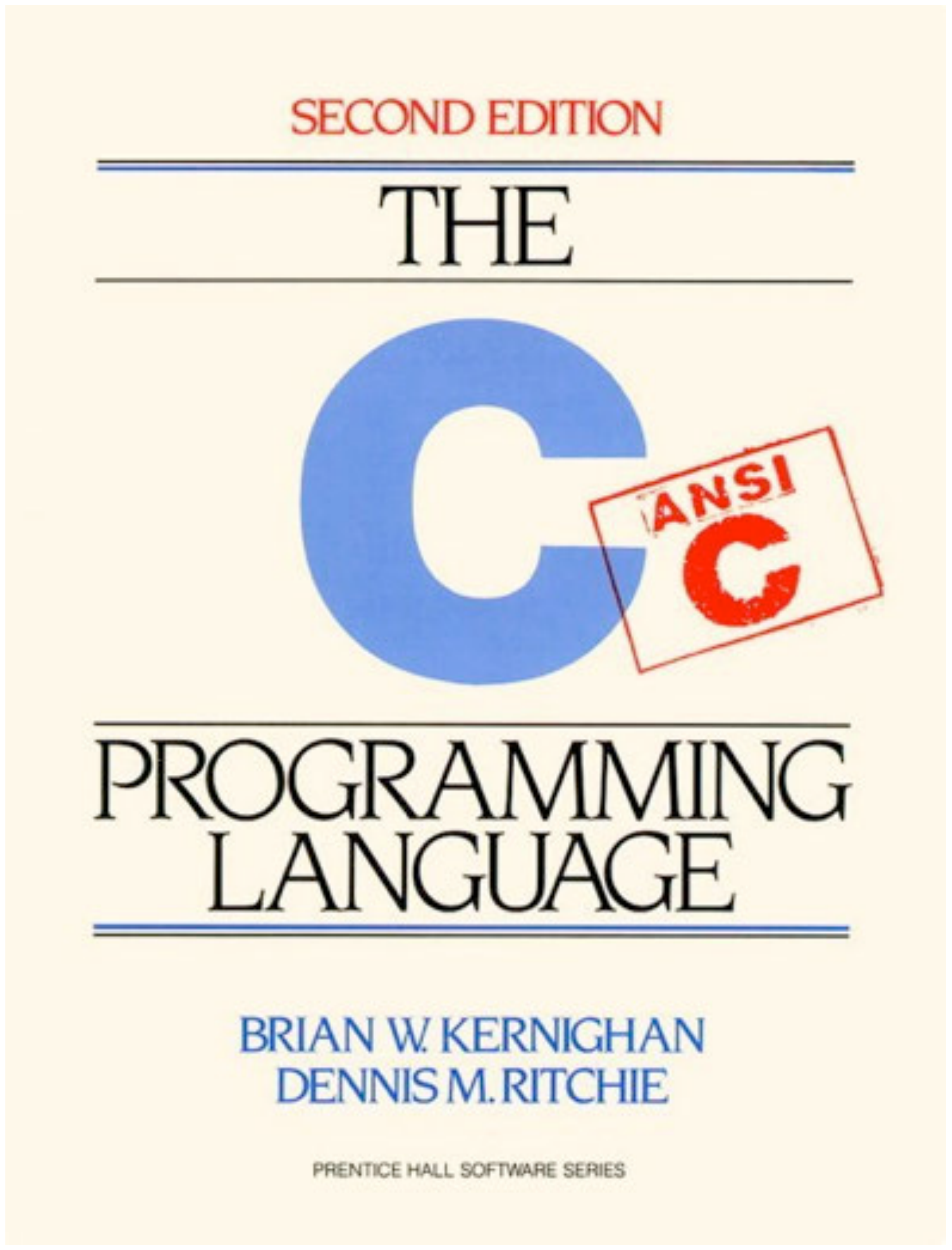
- The UNIX Kernel

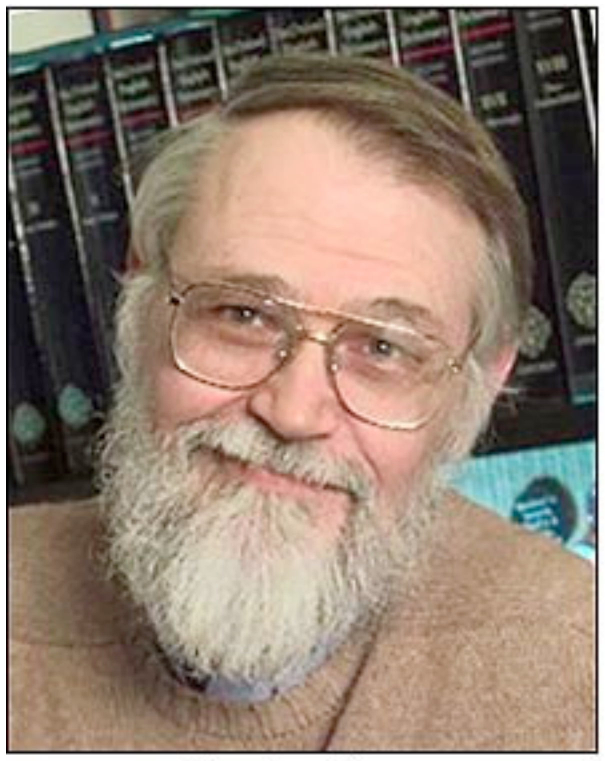
-A special control program which manages the system's resources.
-It controls the starting and stopping of programs (process control).
-It manages access to hardware. If two programs attempt to use the same device it allows them to share access.
-It manages the file system (disks).

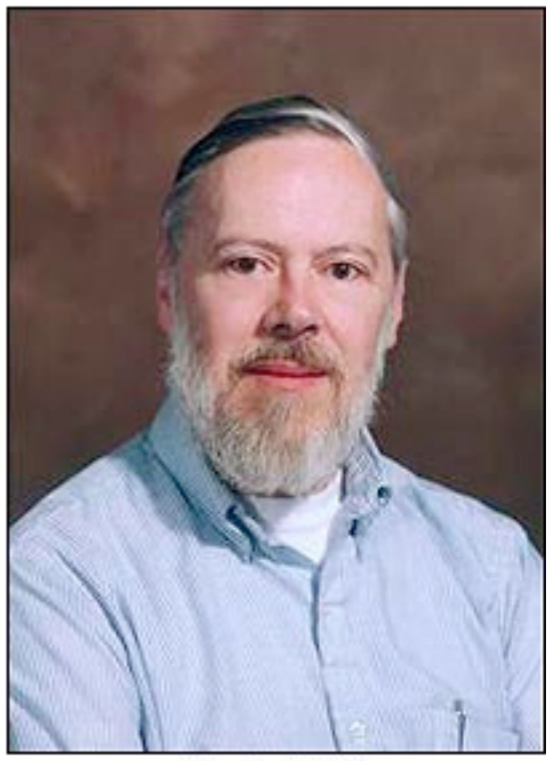

COMPUTER HOLY WARS
HOLD IT RIGHT THERE, BUDDY.
THAT SCRUFFY BEARD... THOSE SUSPENDERS... THAT SMUG EXPRESSION...
YOU'RE ONE OF THOSE CONDESCENDING UNIX COMPUTER USERS!
HERE'S A NICKEL, KID. GET YOURSELF A BETTER COMPUTER.
S. Adams E-mail: SCOTTADAMS@AOL.COM
6/24 © 1995 United Feature Syndicate, Inc. (NYC)

- Why was UNIX successful?

-It used many new technologies.
-It was easy to port to other platforms.
-It was stable and reliable.
-Many of the technologies involved proved to be powerful (e.g. hardware as files, linking programs together, many small tools, hierarchical file system.
-It was given away to academic institutions. Student’s learned about it and went to industry.
-It was easy to extend. Network sockets and a GUI were additions.

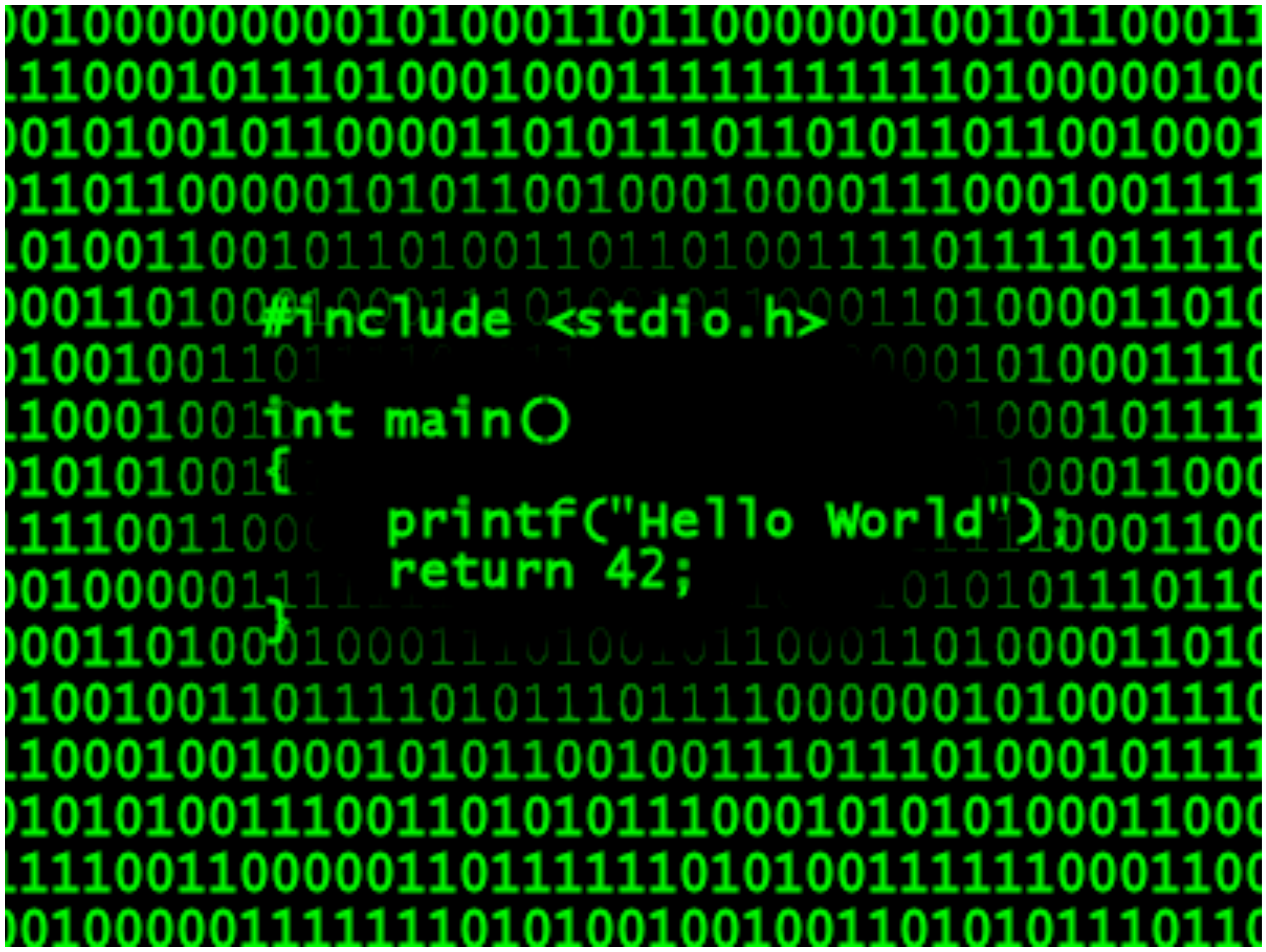



Brian Kernighan


Dennis Ritchie

- The C Programming Language

-C is a general purpose programming language developed in 1972 by Dennis Ritchie. It was made for use on the UNIX operating system.
-Although it was made to write system software it has been widely used for application software. It has been made an international standard.
-It has been ported to very many platforms.
-It has influenced the design of many other languages such as C++, Objective C, C#, Python, Perl, Java, Javascript, PHP, the C Shell.

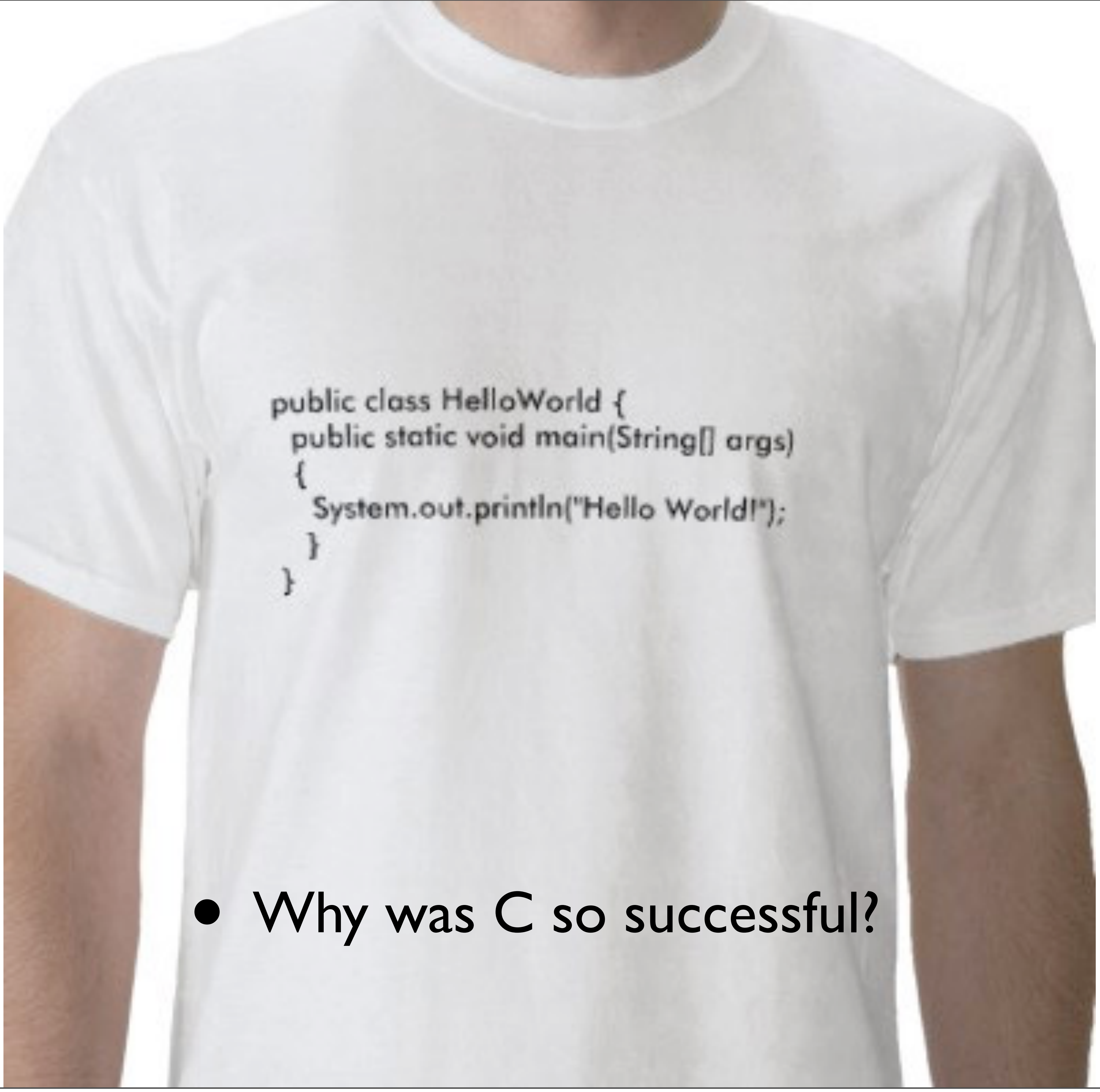


```
#include <stdio.h>

int main()
{
    printf("Hello World");
    return 42;
}
```

- Why was C so successful?

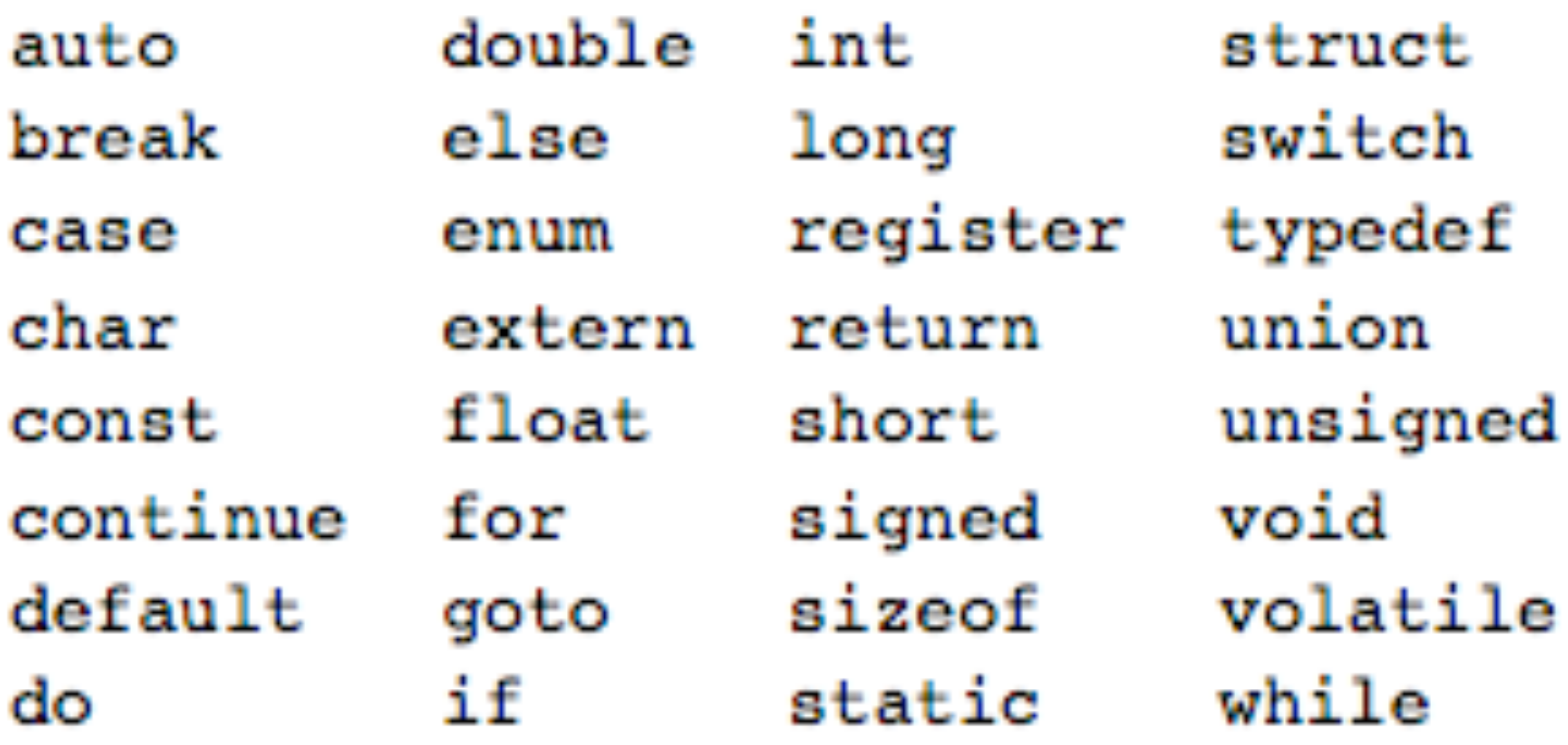
-It could be compiled into an efficient executable program. This reduced the need to use assembly language. It is easier to use than assembly language. (It was easy to create programs which executed efficiently).
-It provided direct access to things like memory, files, and processes. It didn't get in the way of communicating with the operating system resources.
-It is relatively easy to port a C program to a different platform with little change to the code.



```
public class HelloWorld {
  public static void main(String[] args)
  {
    System.out.println("Hello World!");
  }
}
```

- Why was C so successful?

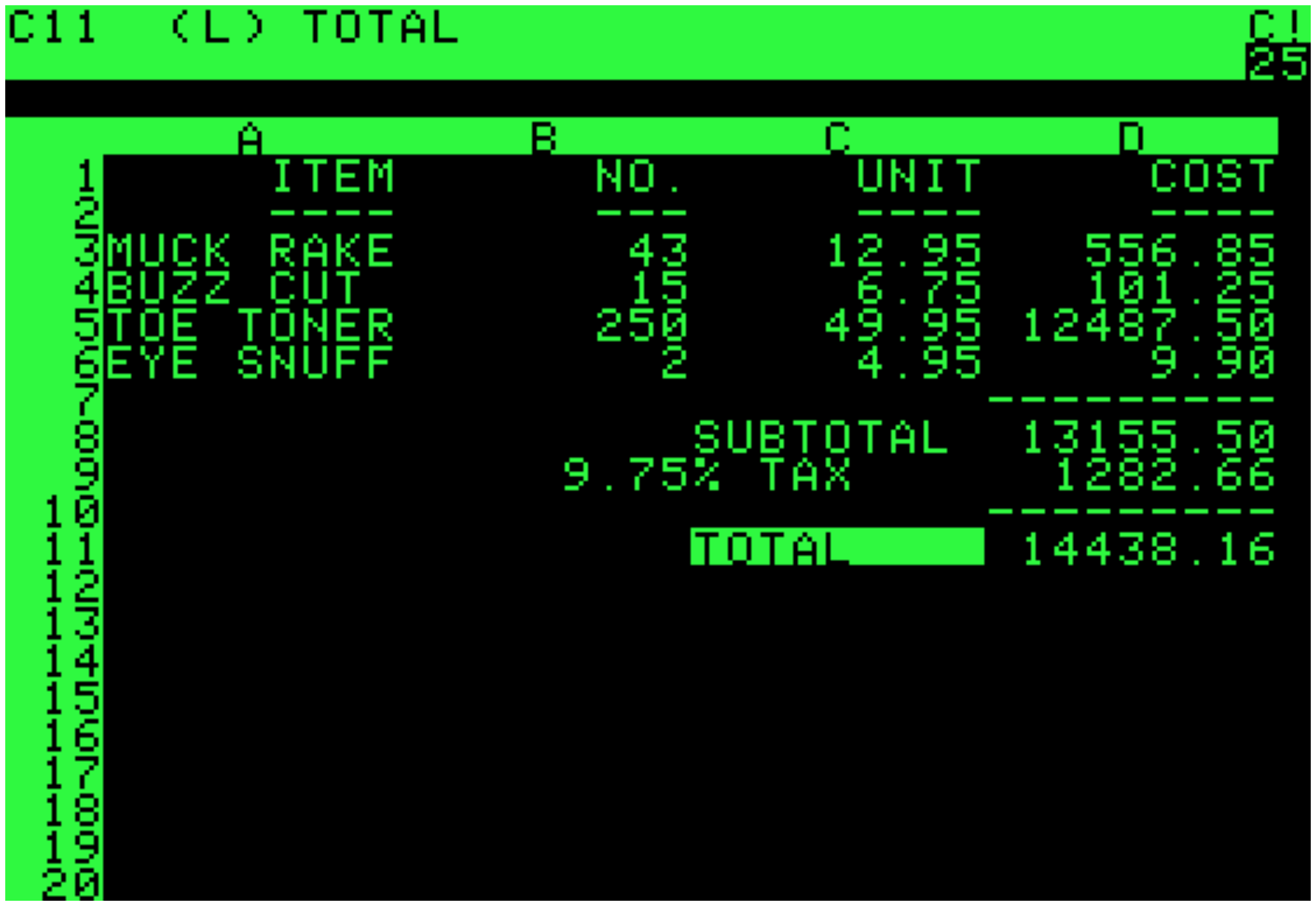
-Libraries are used to extend the functionality of the system. Users could create their own libraries to add more functionality.
-It allows dynamic memory allocation. So the amount of memory the program uses can be determined when the program is running.
-It provided a mix of high level language features (struct, loops, branches) and low level operations (memory access, file manipulation, communication with the operating system).



auto	double	int	struct
break	else	long	switch
case	enum	register	typedef
char	extern	return	union
const	float	short	unsigned
continue	for	signed	void
default	goto	sizeof	volatile
do	if	static	while

- The C Keywords.

-The C language is very small. You can memorize all of the language keywords easily.
-There are other rules that must be followed to use the language, such as how variable names are formed (must begin with a letter or number or _), how the keywords are used, and how other symbols are used (+,-,/,*).
-What is a language verses a library? Why are printf and the math functions not listed here? They aren’t part of the **language**. They are extensions in **libraries**.

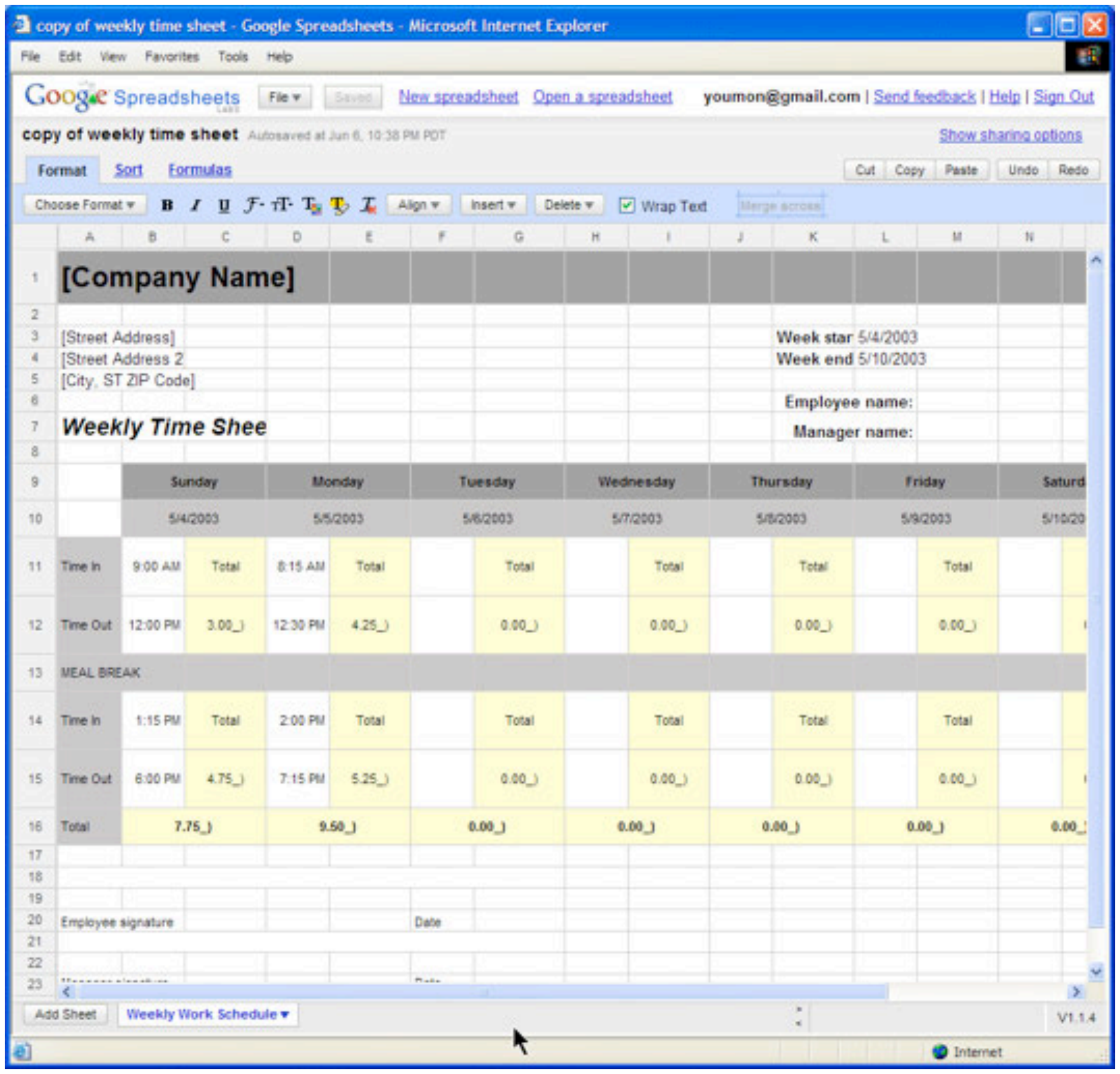


```
C11  (L) TOTAL                                    C!
                                                  25

           A          B          C          D
 1        ITEM        NO.       UNIT       COST
 2        ----        ---       ----       ----
 3MUCK RAKE            43      12.95     556.85
 4BUZZ CUT             15       6.75     101.25
 5TOE TONER           250      49.95   12487.50
 6EYE SNUFF             2       4.95       9.90
 7                                    ----------
 8                        SUBTOTAL     13155.50
 9                   9.75% TAX          1282.66
10                                    ----------
11                        TOTAL        14438.16
12
13
14
15
16
17
18
19
20
```

- The Spreadsheet

-VisiCalc was the first spreadsheet available for the personal computer.
-It is seen as helping the microcomputer (desktop computer) become a business tool. It has been called the first killer application for the personal computer.
-It had many successors which were more powerful and refined.



copy of weekly time sheet Autosaved at Jun 6, 10:38 PM PDT

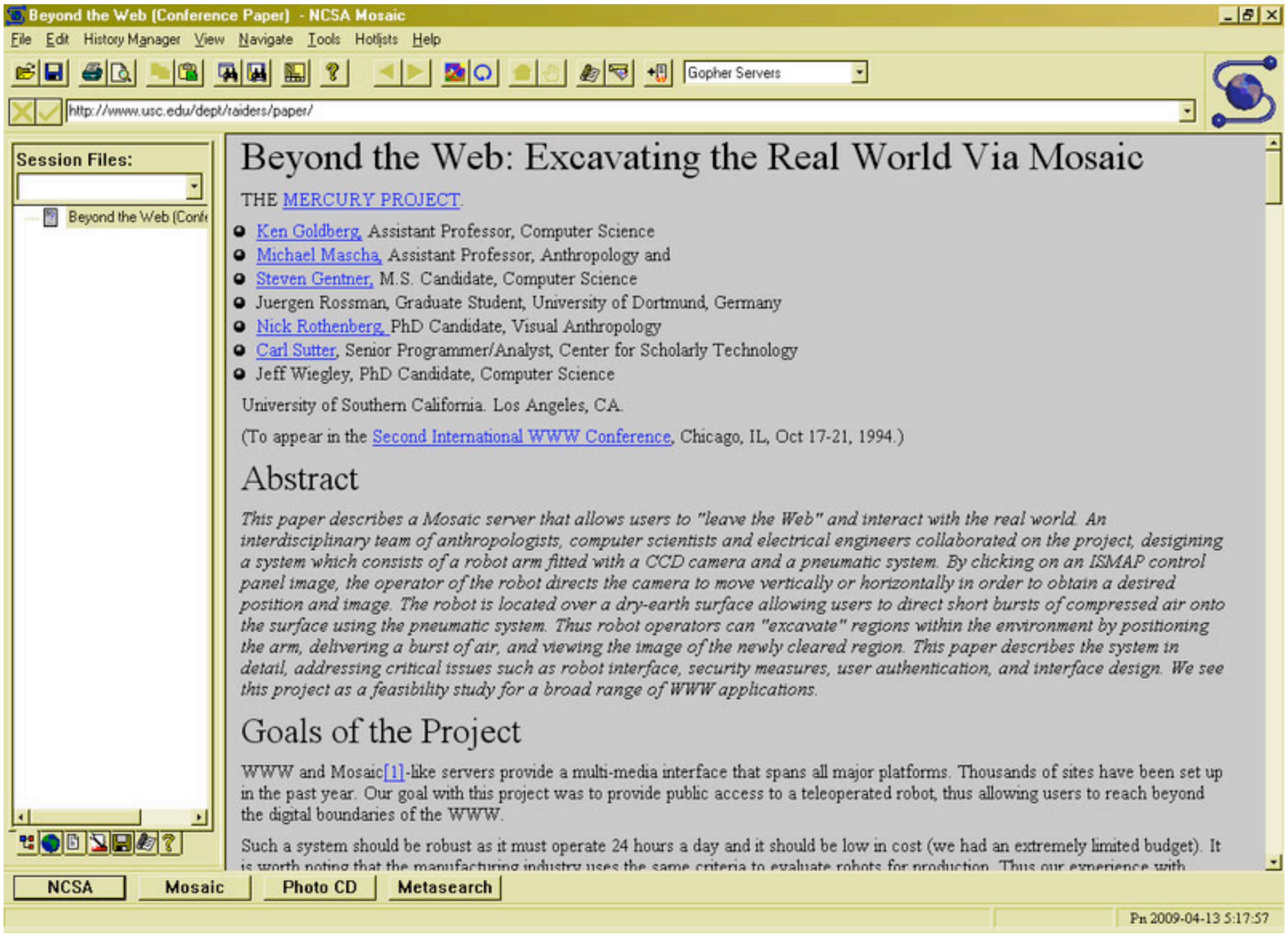
	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	[Company Name]													
2														
3	[Street Address]										Week star	5/4/2003		
4	[Street Address 2										Week end	5/10/2003		
5	[City, ST ZIP Code]													
6											Employee name:			
7	Weekly Time Shee										Manager name:			
8														
9		Sunday		Monday		Tuesday		Wednesday		Thursday		Friday		Saturd
10		5/4/2003		5/5/2003		5/6/2003		5/7/2003		5/8/2003		5/9/2003		5/10/20
11	Time In	9:00 AM	Total	8:15 AM	Total		Total		Total		Total		Total	
12	Time Out	12:00 PM	3.00_)	12:30 PM	4.25_)		0.00_)		0.00_)		0.00_)		0.00_)	
13	MEAL BREAK													
14	Time In	1:15 PM	Total	2:00 PM	Total		Total		Total		Total		Total	
15	Time Out	6:00 PM	4.75_)	7:15 PM	5.25_)		0.00_)		0.00_)		0.00_)		0.00_)	
16	Total	7.75_)		9.50_)		0.00_)		0.00_)		0.00_)		0.00_)		0.00_
17														
18														
19														
20	Employee signature					Date								
21														
22														
23														

- Why was the Spreadsheet successful?

-The user could define the data organization and the formulas applied to the data it was very flexible. It could be used for many different types of calculations.
-The table layout of rows and columns was easy to understand and mirrored many existing processes, especially financial calculations. It build on a familiar pre-existing model.

- The Database

-A database is an organized collection of data.
-It is used to quickly store and retrieve **large** amounts of data.
-This replaced the need to store large amounts of data on paper.
-It allows users to leverage the ability of computers to **store** large amounts of data on a disk and **search** through that data quickly.



Beyond the Web (Conference Paper) - NCSA Mosaic
File Edit History Manager View Navigate Tools Hotlists Help
Gopher Servers
http://www.usc.edu/dept/raiders/paper/
Session Files:
Beyond the Web (Confe
NCSA
Mosaic
Photo CD
Metasearch
Pn 2009-04-13 5:17:57

Beyond the Web: Excavating the Real World Via Mosaic

THE MERCURY PROJECT.

- Ken Goldberg, Assistant Professor, Computer Science
- Michael Mascha, Assistant Professor, Anthropology and
- Steven Gentner, M.S. Candidate, Computer Science
- Juergen Rossman, Graduate Student, University of Dortmund, Germany
- Nick Rothenberg, PhD Candidate, Visual Anthropology
- Carl Sutter, Senior Programmer/Analyst, Center for Scholarly Technology
- Jeff Wiegley, PhD Candidate, Computer Science

University of Southern California. Los Angeles, CA.

(To appear in the Second International WWW Conference, Chicago, IL, Oct 17-21, 1994.)

Abstract

This paper describes a Mosaic server that allows users to "leave the Web" and interact with the real world. An interdisciplinary team of anthropologists, computer scientists and electrical engineers collaborated on the project, designing a system which consists of a robot arm fitted with a CCD camera and a pneumatic system. By clicking on an ISMAP control panel image, the operator of the robot directs the camera to move vertically or horizontally in order to obtain a desired position and image. The robot is located over a dry-earth surface allowing users to direct short bursts of compressed air onto the surface using the pneumatic system. Thus robot operators can "excavate" regions within the environment by positioning the arm, delivering a burst of air, and viewing the image of the newly cleared region. This paper describes the system in detail, addressing critical issues such as robot interface, security measures, user authentication, and interface design. We see this project as a feasibility study for a broad range of WWW applications.

Goals of the Project

WWW and Mosaic[1]-like servers provide a multi-media interface that spans all major platforms. Thousands of sites have been set up in the past year. Our goal with this project was to provide public access to a teleoperated robot, thus allowing users to reach beyond the digital boundaries of the WWW.

Such a system should be robust as it must operate 24 hours a day and it should be low in cost (we had an extremely limited budget). It is worth noting that the manufacturing industry uses the same criteria to evaluate robots for production. Thus our experience with

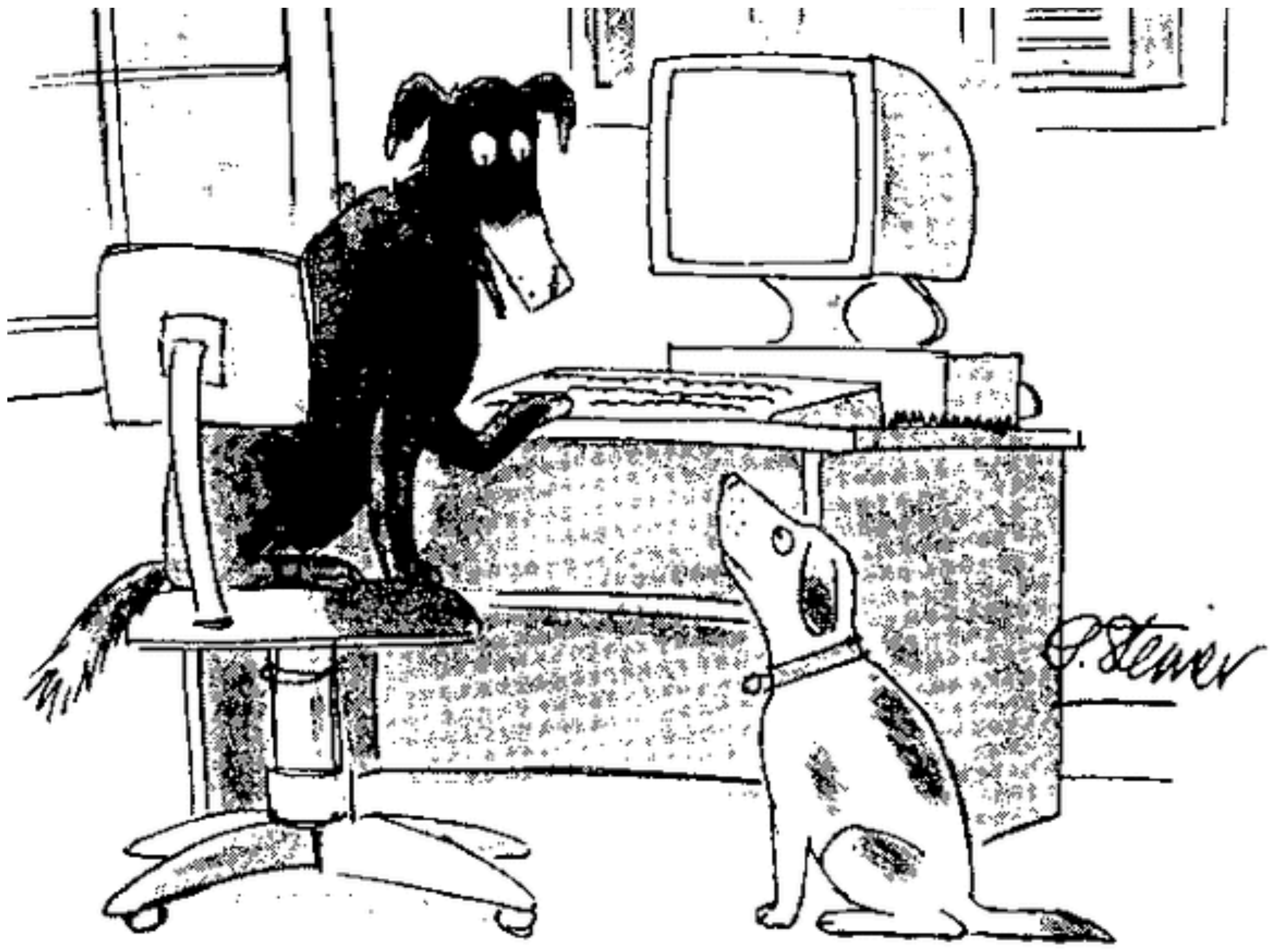
- The Web Browser

-The Mosaic web browser was released in 1993. Not the first browser, that was WorldWideWeb, which was later renamed Nexus. Mosaic is credited with popularizing the web.
-It was easy to understand, easy to install, reliable, and available on Windows. This made it accessible to the public.
-It was the first browser to allow images and text in the same window. Previous browsers opened separate image windows.

THE COMPANY PAYS ME TEN DOLLARS FOR EVERY BUG I FIX IN MY CODE, RATBERT.
I WANT YOU TO DO YOUR LITTLE RAT DANCE ON MY KEYBOARD SO I'LL HAVE LOTS OF BUGS TO FIX.
HOW AM I DOING?
NOT SO GOOD. YOU JUST AUTHORED A WEB BROWSER.
S. Adams E-mail: SCOTTADAMS@AOL.COM
11/14 © 1995 United Feature Syndicate, Inc.(NYC)

- Why was the web browser so successful?

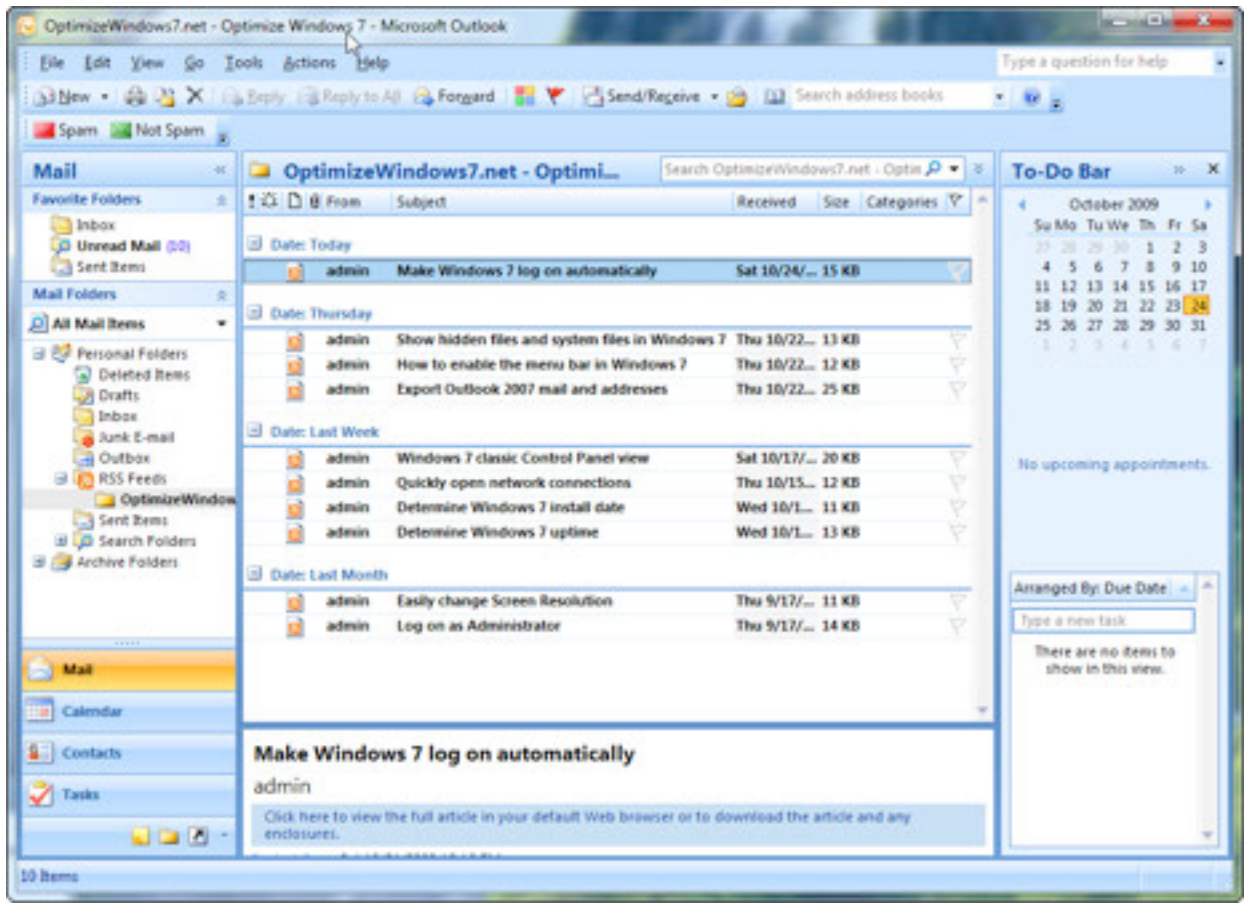
-It was easy to use.
-It made information more available. Especially more esoteric information that might not be published in books or magazines because it had too few readers.
-It made access to information easy for companies and individuals. Companies could put all of their customer support on the web and not have to directly interact with most of their clients. Individuals could put anything they wanted on the web which allowed much easier access to an audience.
-A lot of paper documentation became electronic and more accessible.



"On the Internet, nobody knows you're a dog."

- Why was the web browser so successful?

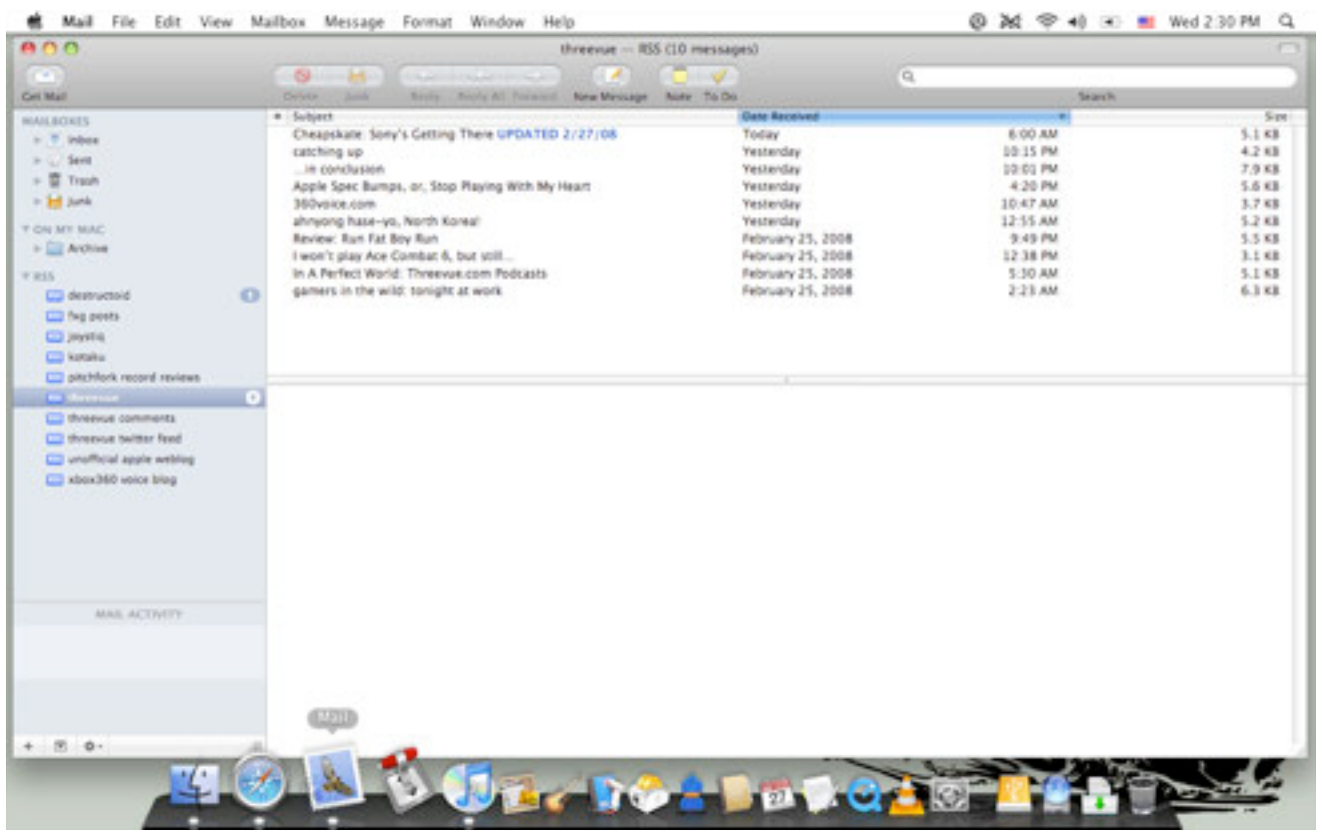
-It took the program off of the desktop and moved it to web servers. This meant the supplier of the software didn't need to support it on your computer. They needed to make its interface accessible in a web browser and they could run it on their local machines.
-It allowed companies to change their traditional, expensive, method of having buildings, storefronts, warehouses into electronic storefronts. Both large traditional retailers and small specialty companies benefited.


OptimizeWindows7.net - Optimize Windows 7 - Microsoft Outlook
File Edit View Go Tools Actions Help
New Reply Reply to All Forward Send/Receive
Mail
Favorite Folders
Inbox
Unread Mail
Sent Items
Mail Folders
All Mail Items
Calendar
Contacts
Tasks
To-Do Bar
October 2009
No upcoming appointments.
Arranged By: Due Date
There are no items to show in this view.

OptimizeWindows7.net - Optimi...

From	Subject	Received	Size
Date: Today			
admin	Make Windows 7 log on automatically	Sat 10/24/...	15 KB
Date: Thursday			
admin	Show hidden files and system files in Windows 7	Thu 10/22...	13 KB
admin	How to enable the menu bar in Windows 7	Thu 10/22...	12 KB
admin	Export Outlook 2007 mail and addresses	Thu 10/22...	25 KB
Date: Last Week			
admin	Windows 7 classic Control Panel view	Sat 10/17/...	20 KB
admin	Quickly open network connections	Thu 10/15...	12 KB
admin	Determine Windows 7 install date	Wed 10/1...	11 KB
admin	Determine Windows 7 uptime	Wed 10/1...	13 KB
Date: Last Month			
admin	Easily change Screen Resolution	Thu 9/17/...	11 KB
admin	Log on as Administrator	Thu 9/17/...	14 KB

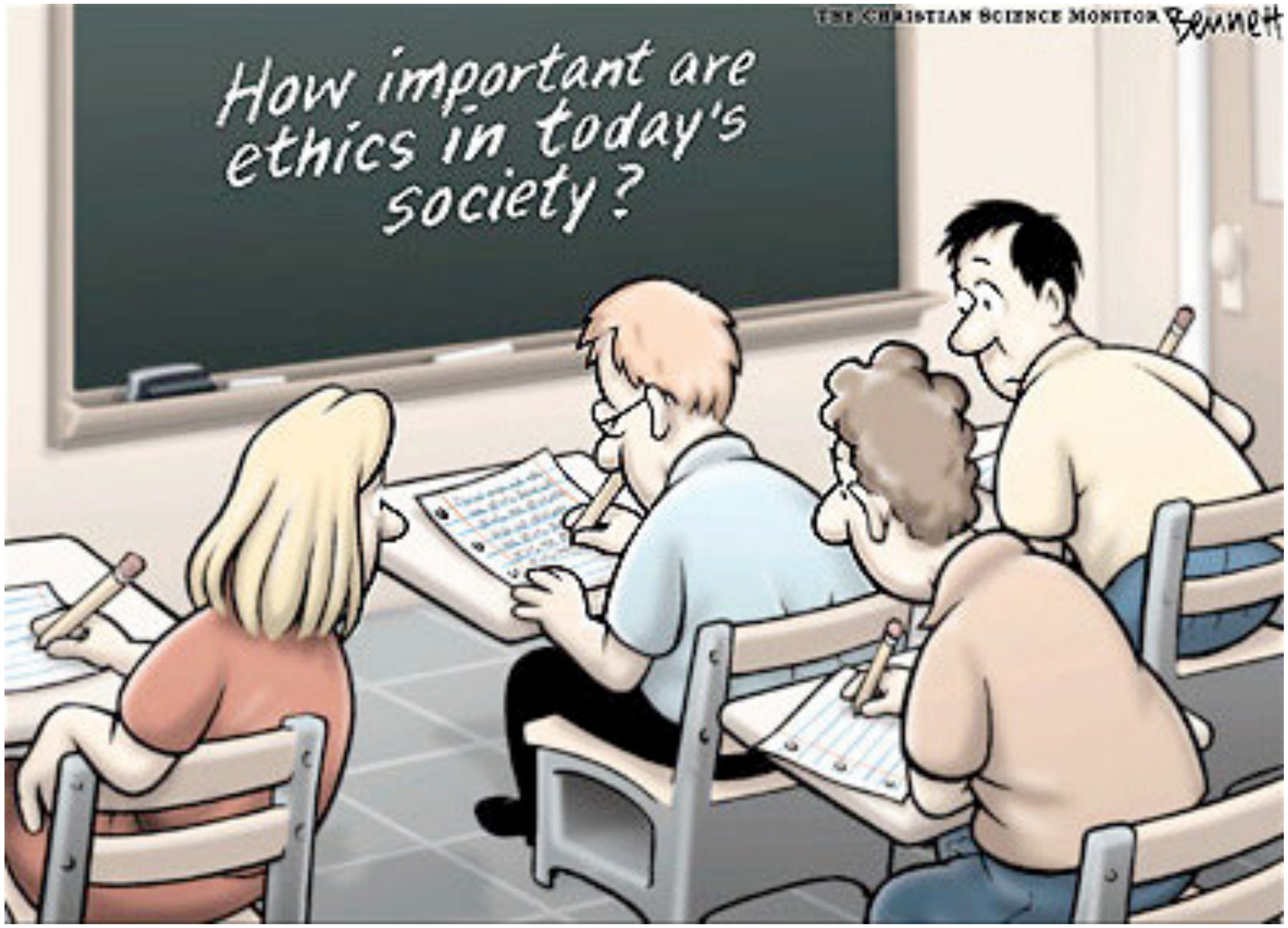
Make Windows 7 log on automatically
admin
Click here to view the full article in your default Web browser or to download the article and any enclosures.


Mail File Edit View Mailbox Message Format Window Help
Wed 2:30 PM
threevue — RSS (10 messages)
Delete Junk Reply Reply All Forward New Message Note To Do
Search
MAILBOXES
Inbox
Sent
Trash
Junk
ON MY MAC
Archive
RSS

Subject	Date Received		Size
Cheapskate: Sony's Getting There UPDATED 2/27/08	Today	6:00 AM	5.1 KB
catching up	Yesterday	10:15 PM	4.2 KB
...in conclusion	Yesterday	10:01 PM	7.9 KB
Apple Spec Bumps, or, Stop Playing With My Heart	Yesterday	4:20 PM	5.6 KB
360voice.com	Yesterday	10:47 AM	3.7 KB
ahnyong hase-yo, North Korea!	Yesterday	12:55 AM	5.2 KB
Review: Run Fat Boy Run	February 25, 2008	9:49 PM	5.5 KB
I won't play Ace Combat 6, but still...	February 25, 2008	12:38 PM	3.1 KB
In A Perfect World: Threevue.com Podcasts	February 25, 2008	5:30 AM	5.1 KB
gamers in the wild: tonight at work	February 25, 2008	2:23 AM	6.3 KB

- Email

–The first widely used digital text communication system. The email format from the early 1970’s looks similar to that of today.
–Alternatives at the time were the telephone, regular mail, or the fax.
–The ability to send text was new. Attachments were added later.
–It provided fast, easy communications for people who are not physically located in the same place. It is less demanding than telephone communications.
–It provided a way to transfer large amounts of information.


How important are ethics in today's society?
THE CHRISTIAN SCIENCE MONITOR Bennett

Personal Ethics

-Ethics presentations normally involve a lot of case studies and discussion around what is ethical.

- Unethical behaviour can be either intentional or unintentional.
- You can be held responsible and even liable for the consequences of your actions in either situation.

A Case Study

- You are going to give a presentation to try and win a contract for an air traffic control system from a potential client.
- You are on an airplane and the person sitting next to you is reading a document about their bid for the same contract.
- You see that their company has found a way to develop the software for 30% less than your company.

-What is the correct thing to do?   -say nothing and read as much as you can
   -start a conversation and try to learn more without revealing who you are
   -introduce yourself and let them know you are bidding for the same contract, and mention your interest in the cheaper process, any information they reveal is then their choice
   -not use any information you learn, is it possible to unlearn something

Variations on the Last Case

- What if the developer is having a discussion with a colleague and you can overhear it without any action on your part?
- What if the developer left the report open on his seat when he went to the bathroom?
- What if the report were left face down? Would you turn it over?

-The outcome could be losing the contract or your job if you are caught. Your company could be sued.
-Are you answering the questions based on the ethics of the situation or on the risk involved in making each choice?
-What if your company was having financial difficulty? Would that affect you?
-Would you act unethically if you thought that you could get away with it?

Another Variation

- You overhear the two developers discussing how they can create the software for 30% less and they mention working with a company that you know about.
- This company is a front for an individual who has been convicted of falsifying safety documents and bribing safety inspectors.

-Your concern is now about public safety, not just economics. Would this change your behaviour?
-Would you tell the developers about the company with which they intend to work?
-Would you use this to discredit their bid and win the contract?

Common Violations

- Failing to protect the public

-Failing to protect the safety, health, welfare, and property of the public by not notifying employers or clients of such dangers.
-This also can occur if one fails to protect the safety, health, welfare, and property of the public by approving documents or work that are in violation of professional standards. This is particularly true for professional engineers.

Failing to Protect the Public Case Study

- Morton-Thiokol (M-T) manufactures the booster rockets for the space shuttle. Management and representatives of NASA were informed that there was an increased probability that the engines (specifically the O-rings) could fail to operate correctly as the temperature increased.

-This is what caused the space shuttle Challenger disaster.

- The M-T director of the solid rocket booster project was Allan J. MacDonald. He arranged a teleconference between M-T and NASA engineers to discuss concerns that a cold weather launch could be dangerous.
- One engineer recommended that no launch be attempted in temperatures of less than 53 degrees Fahrenheit.

-The project director identified the problem and attempted to address it.
-A solution was identified.

- M-T was in the process of renegotiating its contract to supply NASA with the solid rocket boosters.
- A senior executive from M-T told the Vice President of engineering to “take off your engineering hat and put on your management hat.”
- Company executives overruled the concerns of their own engineers by voting that the seals **could not be shown to be unsafe**.

-Management did not want their product to look unsuccessful during a contract negotiation.
-Notice the clever use of wording.

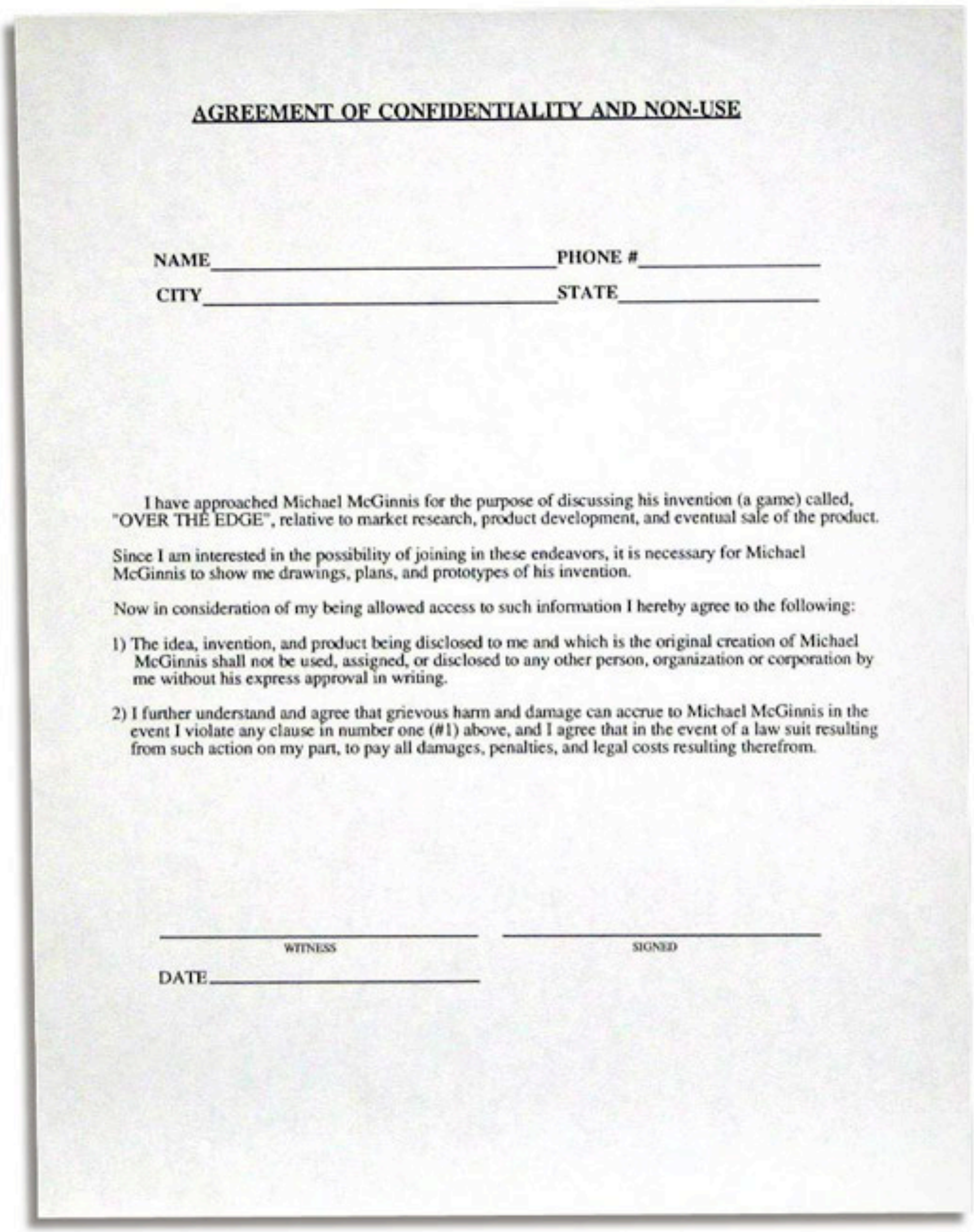
- Morton-Thiokol's Vice President for the booster rocket program had to sign the recommendation to launch document when Allan MacDonald refused to sign it.

-The person in charge refused to acknowledge the engine was safe when there were serious concerns that it was unsafe.

Common Violations

- Unethical disclosure of information

-Revealing information obtained in a professional capacity without prior consent of the employer (who the information was obtained from).
-What happens if you move to a company which competes with your previous employer? Can you forget the trade secrets of the first company? Can you differentiate between the confidential and non-confidential information?
-The developer and the employers need to carefully identify what is confidential.



AGREEMENT OF CONFIDENTIALITY AND NON-USE

NAME____________________ PHONE #____________

CITY____________________ STATE____________

I have approached Michael McGinnis for the purpose of discussing his invention (a game) called, "OVER THE EDGE", relative to market research, product development, and eventual sale of the product.

Since I am interested in the possibility of joining in these endeavors, it is necessary for Michael McGinnis to show me drawings, plans, and prototypes of his invention.

Now in consideration of my being allowed access to such information I hereby agree to the following:

1) The idea, invention, and product being disclosed to me and which is the original creation of Michael McGinnis shall not be used, assigned, or disclosed to any other person, organization or corporation by me without his express approval in writing.

2) I further understand and agree that grievous harm and damage can accrue to Michael McGinnis in the event I violate any clause in number one (#1) above, and I agree that in the event of a law suit resulting from such action on my part, to pay all damages, penalties, and legal costs resulting therefrom.

________________ ________________
WITNESS SIGNED

DATE________________

- The non-disclosure agreement (NDA)

-This acts as a promise to not release some information. They are fairly common.
-It is often used to limit the release of information about new products. The person who signs the NDA can receive the information but they are not allowed to pass the information to anyone else.
-Companies use them so they can tell other companies about their future products but they do not want to release the information to the public. They can also be used during user testing.
-Know what you are signing.

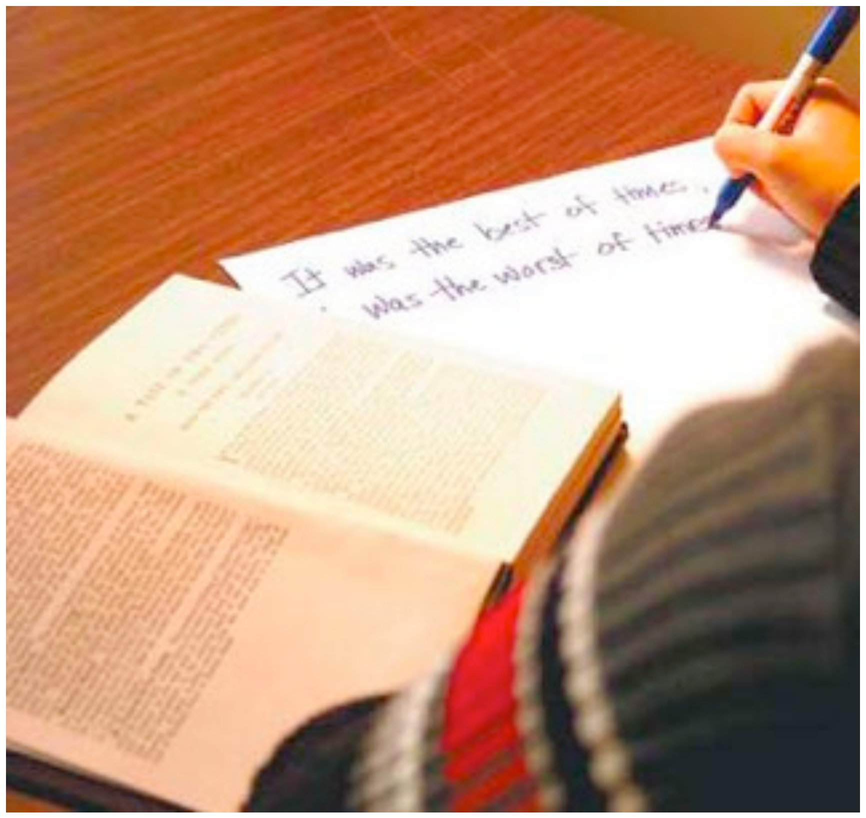
Common Violations

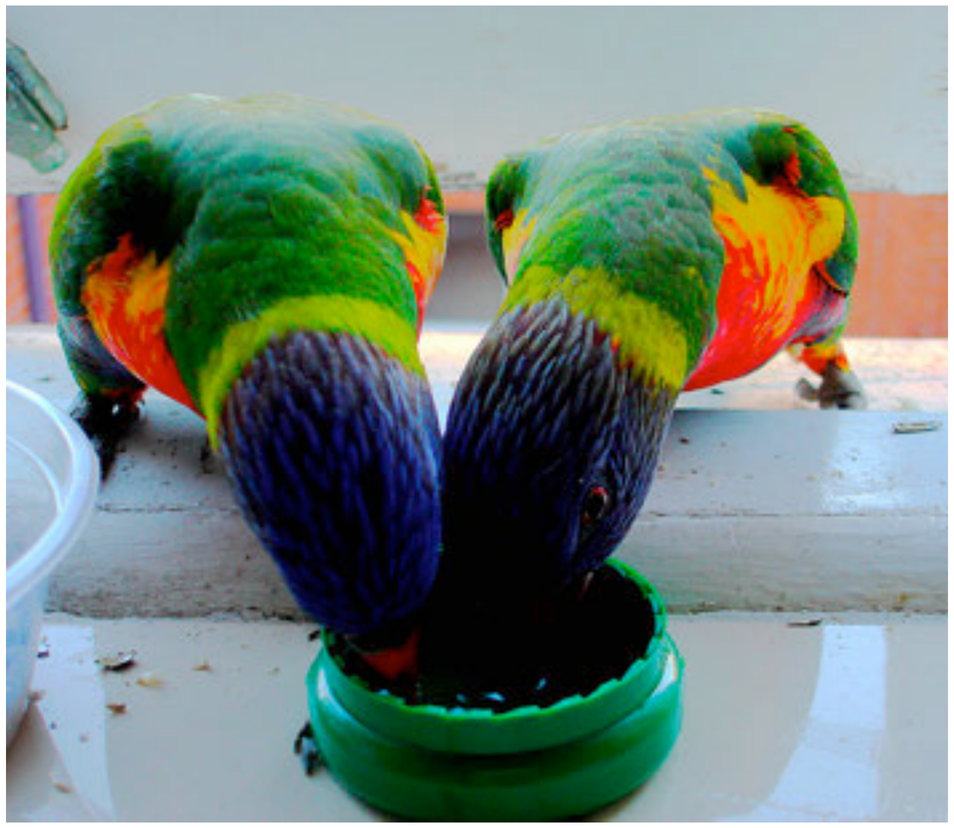
- Failure to include all pertinent information

-This involves not being truthful in reports or statements.

Types of Misconduct

- Plagiarism - misrepresenting other's work as your own.
- Unauthorized Collaboration - working together on work that is intended for individuals.





-Forms of academic misconduct that can apply to professional activities.
-Plagiarism can easily be avoided if you correctly attribute the work to the original author. Using citations and quotation marks will often be sufficient.
-Minor changes to the work is not sufficient to make it your own. An idea can be plagiarized even if the description is changed.
-Claims of "working closely together" are not justification for collaboration.

Types of Misconduct

- Impersonation - having someone impersonate oneself.
- Falsification - presenting false documents.
- Withholding - not releasing documents to gain an advantage.

-Impersonation can be in person or electronically. Both parties are at risk.
-Falsification can be false medical documents, false research findings, false credentials, false or altered grades or time stamps.

Types of Misconduct

- Unauthorized aids and assistance - to use or **obtain** prohibited material. It includes writing, research, and software services which are prohibited unless specifically allowed by the instructor.
- The Rent-A-Coder case was instrumental in creating this form of misconduct.

-This is a large category. It is directed towards services which do the work for their clients who can then benefit from it.
-The claim that the work itself is not submitted and that it is used for studying purposes is not accepted.
-In the Rent-A-Coder case, a student purchased assignments and then submitted them as his own work. He claimed the work was used to study from and was not submitted directly. Obtaining completed solutions became a new for of misconduct because of this case.

Types of Misconduct

- Improper Access and Obstruction - includes making it difficult or impossible to access material, improper access to materials, and improper dissemination.

-Access includes destroying or hiding items such as library books or data files.
-Improper access includes copies of exams, tests, or assignments,
-Improper dissemination involves making confidential information available to the public.

Social Networking and Ethics

- Is it acceptable for potential employers to search their applicant's history on the web?
- Is it acceptable to fire someone for posts to a social networking page?

-Everything on social networking pages is public. Access restrictions can limit who sees the posts but they are not infallible and the people who are allowed access may not keep the contents of the page a secret.
-Nothing on the internet goes away.
-According to a 2009 study by internet security firm Proofpoint, 8% of companies with more than 1000 employees have fired someone for social media actions. This is double what was reported in 2008.

Fired by Facebook

- Feb. 26, 2009: A U.K. teenager was fired for calling her job "boring." According to The Daily Mail, Kimberley Swann posted comments such as, "First day at work. Omg (oh my god)!! So dull!!" and "All I do is shred holepunch and scan paper!!!" [sic]. Swann was canned after her boss discovered the comments.

Fired by Facebook

- April 27, 2009: A Swiss woman was fired after calling in sick and then logging into Facebook on her “sick day.” Apparently the woman had a migraine and called out of work because she thought the light from a computer would bother her and she needed to lie in a dark room. When her employer caught her surfing Facebook, it was presumed that she was indeed well enough to sit in front of a computer, and she was let go.

- August 27, 2009: Ashley Payne, a Georgia high school teacher, was forced to resign after the local school board came across pictures of her sipping beer and wine. The pictures, which appeared on Payne's Facebook page, were from a vacation she had taken that summer, which included a trip to the Guinness Brewery in Ireland. Payne was quoted as saying "I did not think that any of this could jeopardize my job because I was just doing what adults do and have drinks on vacation and being responsible about it." She sued the school district last November.

References

- Engineering by Design, second edition, by Gerard Voland, Pearson Education, Inc., 2004.
- University of Guelph, 2010-2011 Academic Calendar.
- Cytalk Web Site - http://internet.cytalk.com/2010/09/how-to-get-stupidly-fired-because-of-facebook/

Professional Communications

-A lot of this sounds obvious but it is easy to forget in heated discussion.

- How to deal with professional communications.

-You will be required to deal with people who have very different experiences than your own.
-They will have different expertise and experience than your own.
-You will need a range of presentation and listening strategies.

- Be Mindful

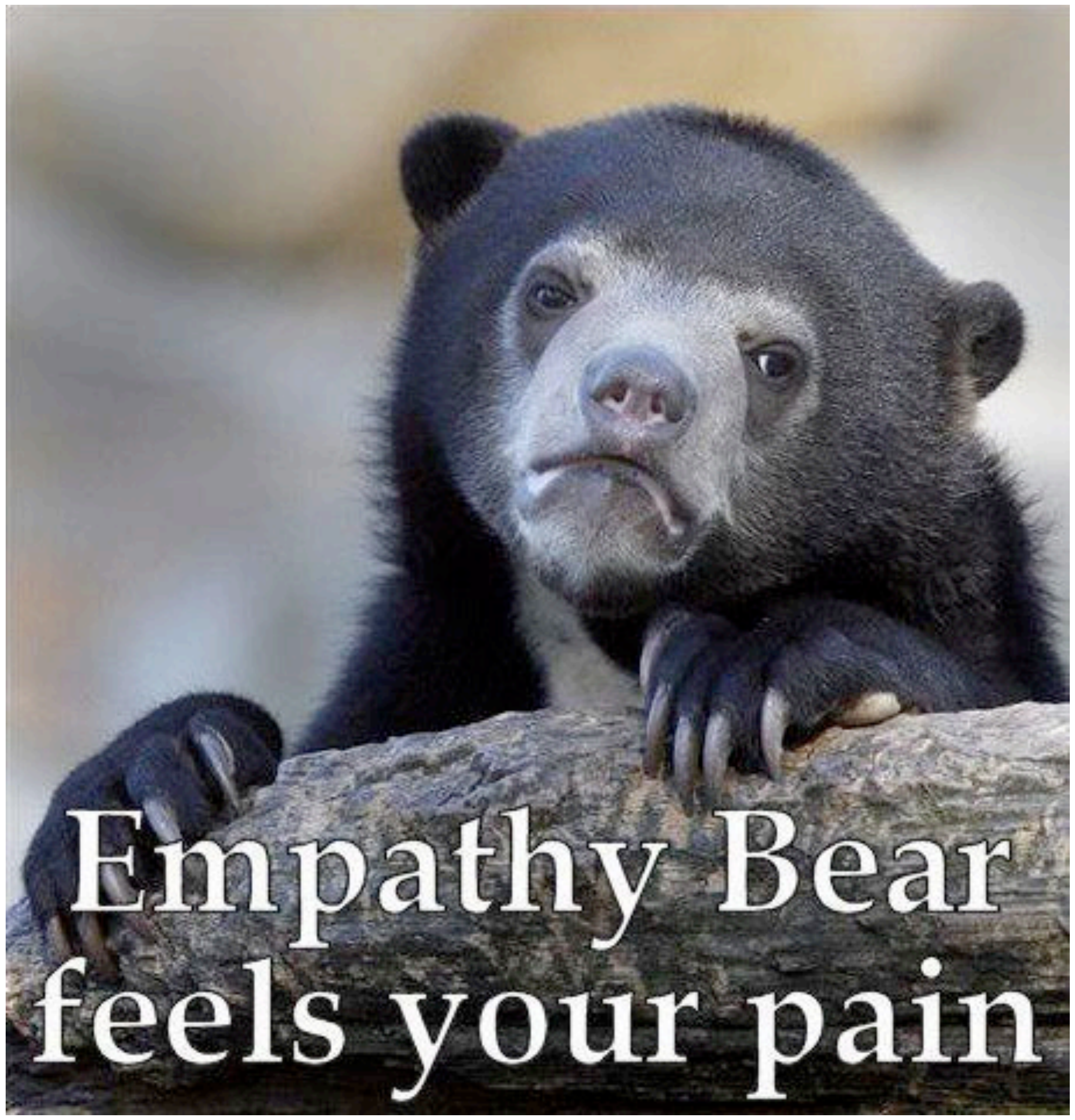
-Identify the role of the participants, the desired outcomes, and the options for communications available to you.
-Is it best to use email, letters, memos, the telephone, meetings, etc.?
-Determine the best method of communications to achieve the desired results, not the method which is easiest for you.
-Be aware that some people may be hesitant to provide information if they believe their supervisor will learn about it.

- Be Flexible

-The development team can change which can prevent communications from being consistent.
-Team membership can change, email may be down, management may redirect people to other tasks, a new deadline may be imposed, and so on.
-Communication strategies must adapt to the situation. A method which worked previously may not be effective again. Be aware of changes and evaluate the situation.

- Be culturally sensitive

-Be aware of acknowledge cultural differences.
-Do not be judgemental of different behaviours. They are simply differences and there is nothing inherently good or bad about them.
-In a multicultural team make an effort to learn culturally appropriate behaviour.
-For example, is eye contact considered a sign of engagement or rudeness.
-There is a difference between asking, “can you do this” and “do you see a problem with this approach.”
-Is using the person’s first name rude or acceptable?



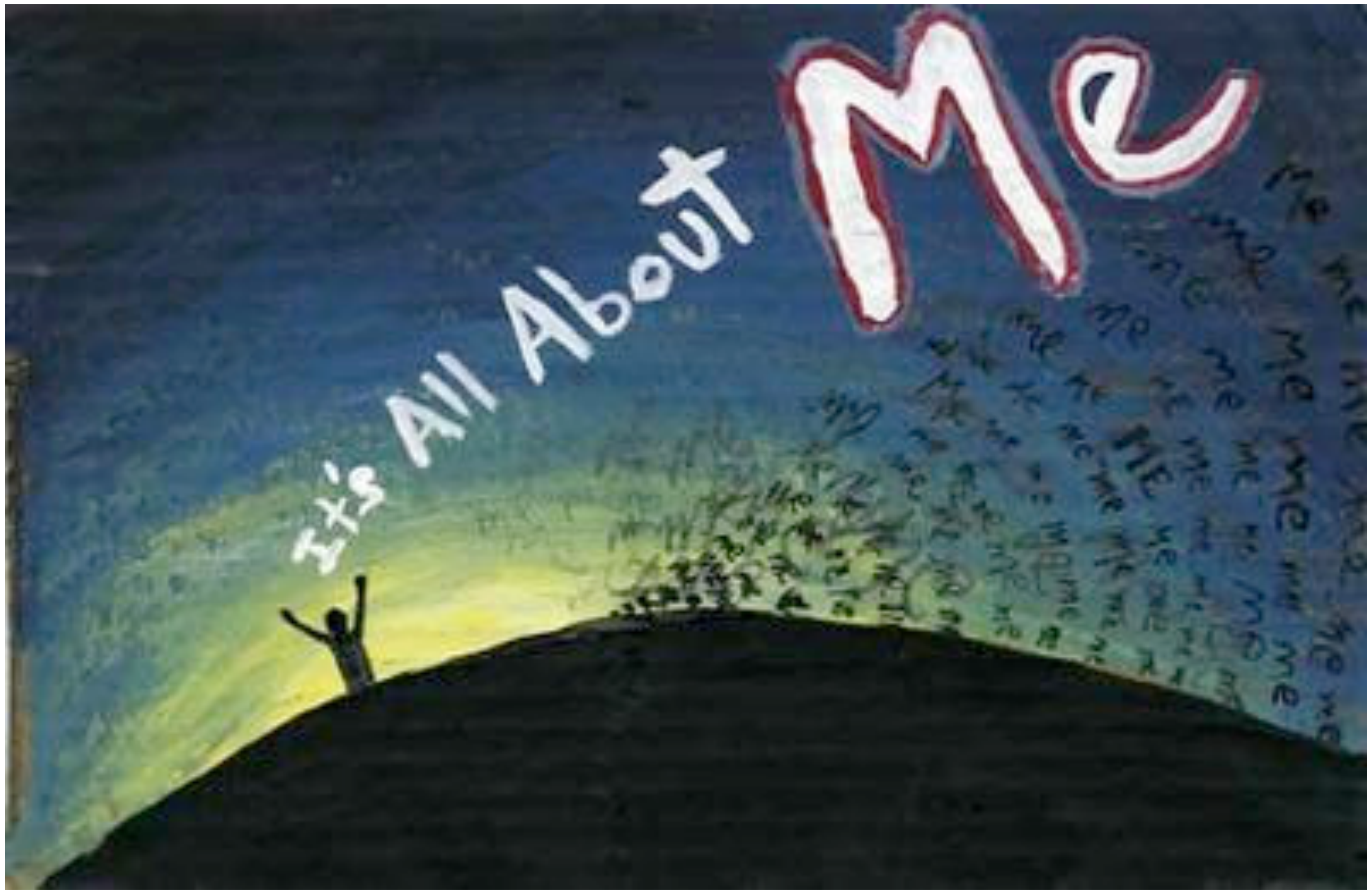
- Be open and empathetic

-An open communication strategy requires that you be open to communicating. Pay attention to the person you are communicating with. Do not look distracted or disinterested.
-Allow the user to speak. Listen carefully and do not interrupt. Listening can be more effective at engaging the other person than speaking.
-You can summarize what they have just said before making your point. This lets them know you are listening to them and that you respect their point of view.
-Using their name is can help but it can seem artificial if overused.

A Listening Strategy

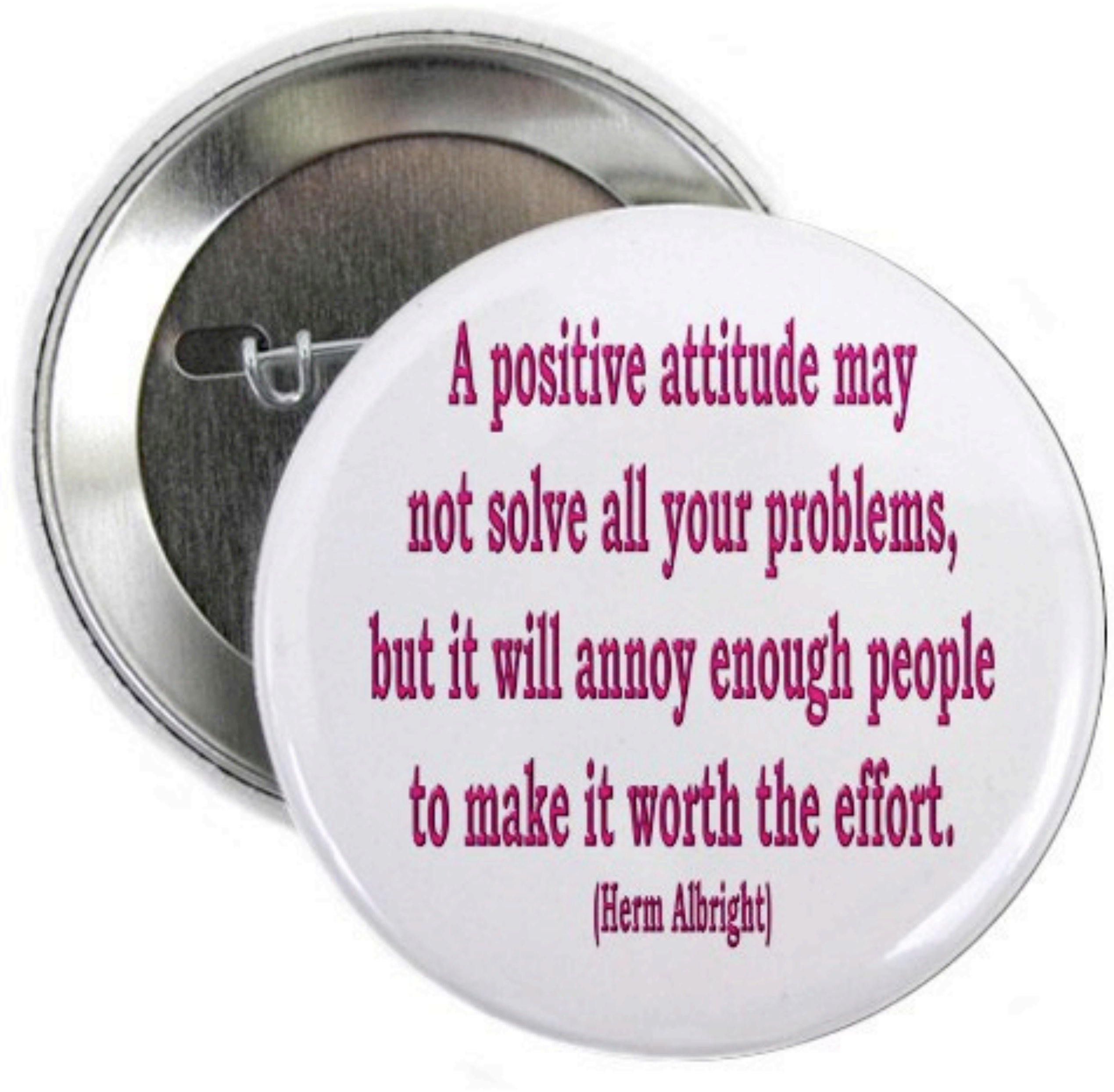
- Initially let them do most of the talking.
- Maintain eye contact.
- Appear willing to listen.
- Use non-verbal acknowledgements to emphasize your attention.

-Listen and nod your head.



- The I-Message

-Take ownership of your part of the discussion buy when stating your ideas by using sentences beginning with, “I ....”. For example, “I didn’t understand that,” instead of “you were unclear.” Statements that contain “you” sound more like criticism.
-This prevents criticism of the other person.
-Another example, “Your performance at the meeting was poor,” versus “I was disappointed with your presentation at the meeting.”


A positive attitude may
not solve all your problems,
but it will annoy enough people
to make it worth the effort.
(Herm Albright)

- Be Positive

-Make statements that have a positive tone instead of a negative one.
-Instead of saying, "Your solution is stupid," you could say "The other option looks like it will be more effective."
-Counterproposals should sound like improvements rather than replacements for ideas. This will make them more acceptable.

- Be Honest

-Speaking your mind clearly and honestly may cause some initial discomfort but it is worth the trouble because the issue being discussed can be fully resolved.
-In situations where there is a conflict involve there is less likelihood that it will reappear if the issue is fully resolved.

- Be Fair

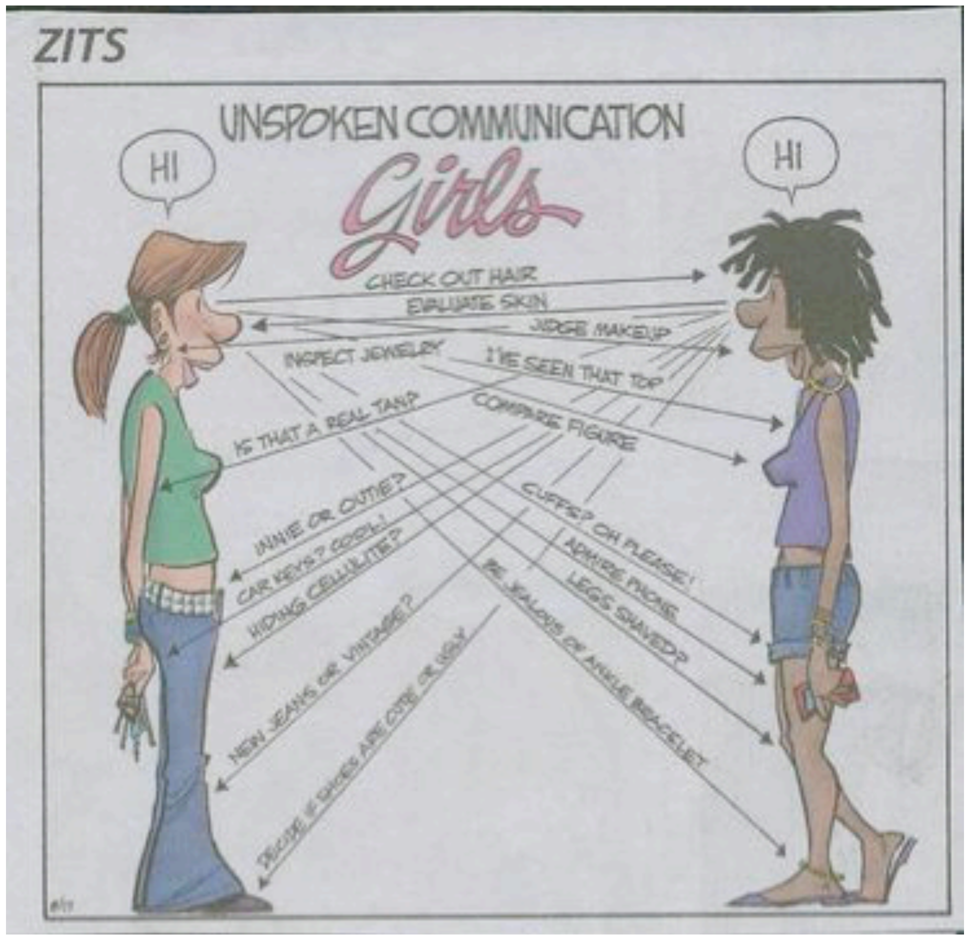
-Keep focussed on the current discussion.
-Reminding people of past offences, using unrelated examples, or changing the subject only serve to add resentment to the relationship.
-Trying to one-up or browbeat someone will not lead to the destruction of the relationship and will not create useful communications.
-Listen to the other person’s concerns and acknowledge when you are at fault.

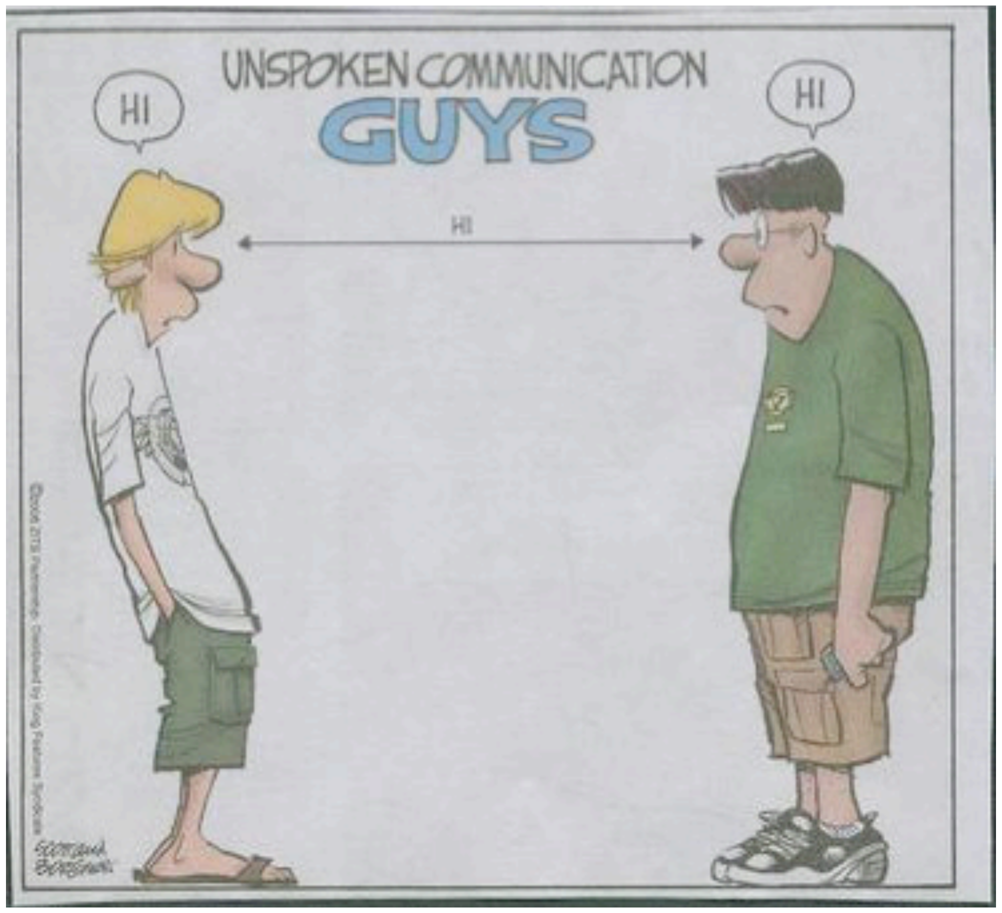
- Avoid Pressure

-People don’t like to be emotionally pressured or verbally bullied.
-If you try to force them to cooperate then they probably will not cooperate at all.

- Avoid Anger

-Anger destabilizes rational thought and can lead to the loss of control of the situation.
-Keeping calm can lead to a discussion which will hopefully solve the problem.
-Anger will cause people to stop participating even if it is their fault.





- Non-verbal Communications

-This includes body language and tone of voice.
-When these don't match the message the listener can be confused.
-If your body language says that you are not interested then the listener may not be want to talk to you.
-The following statements are different depending on which words are emphasized, "I don't want to go out **tonight**" and "I didn't **say** she stole the money."
-If someone else is sending mixed messages the solution is to ask questions which clarify their intent.

- When Speaking

-Recognize how you feel about the topic. If you understand your feelings then you can state your concerns clearly.
-Avoid making emotionally charged statements about others. They will not create a desirable situation. Avoid using statements about others using the work “you” as this sounds like an accusation. It may make you feel better but it wont resolve the situation.

Excuse me,.. on a scale from 1 to 5, with 1 being "always" and 5 being "never," how often do you answer random questions from people on the street?
Greeter
© Original Artist Reproduction rights obtainable from www.CartoonStock.com

- Avoid Leading Questions

-These are intended to pressure the listener into agreeing with you.
-They reflect the values of the questioner more than the listener. They are not useful for gathering information.

- Improving Communication Strategies

-Identify if you have any communications barriers that prevent you from saying what you want to say.
-Common problems include the fear of what others' reactions, lack of confidence, or attempting to avoid failure.
-Try to develop strategies to overcome these problems. Acknowledge them if they exist and try to address them.

References

- Guide to Interpersonal Communication, http://cnx.org/content/m17115/latest/
- Interpersonal Communications: 6 Strategies For Best Results, http://ezinearticles.com/?Interpersonal-Communication---6-Strategies-For-Best-Results&id=4458710
- Interpersonal Communication, http://www.authorstream.com/Presentation/aSGuest51297-425311-Interpersonal-