SOFTWARE REQUIREMENTS ENGINEERING

TEAM SKILL 2: UNDERSTANDING USER AND STAKEHOLDER NEEDS

REQUIREMENT ELICITATION TECHNIQUES-III

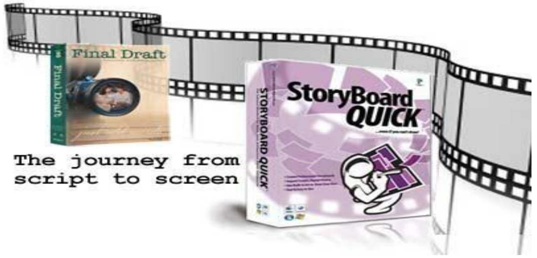
Presentation Outline

- Storyboarding
- Benefits of Storyboarding
- Limits of Storyboarding
- Types of Storyboarding
  - Passive Storyboarding
  - Active Storyboarding
  - Interactive Storyboarding
- What Storyboards Do
- Tools for Storyboarding
- Tips for Storyboarding
- Role Playing
- How to Role Play
- Scripted Walkthroughs
- CRC Cards

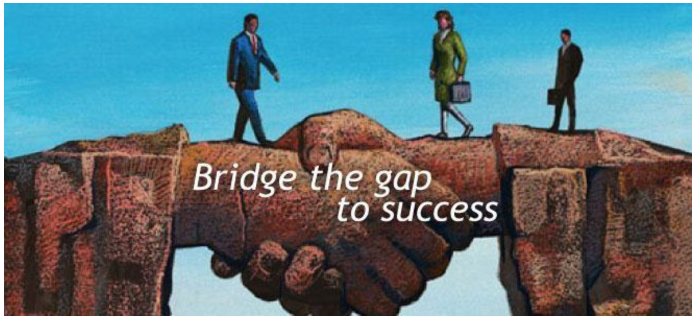
Requirements Elicitation Techniques

- ✓ Interviews
- ✓ Questionnaires
- ✓ Background Reading
- ✓ Introspection
- ✓ Social Analysis
- ✓ Requirements Workshops
- ✓ Brainstorming and Idea Reduction
- ✓ Story Boarding
- ✓ Role Playing
- ✓ Prototyping
- ✓ Requirements Reuse

Storyboard



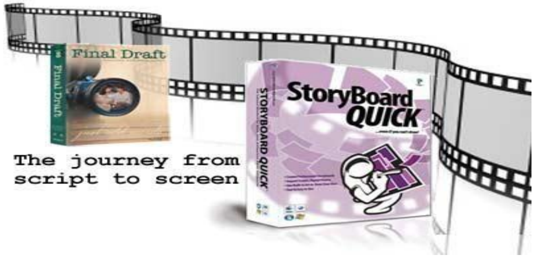
- A Storyboard is a **logical and conceptual description of system functionality for a specific scenario**, including the interaction required between the users and the system



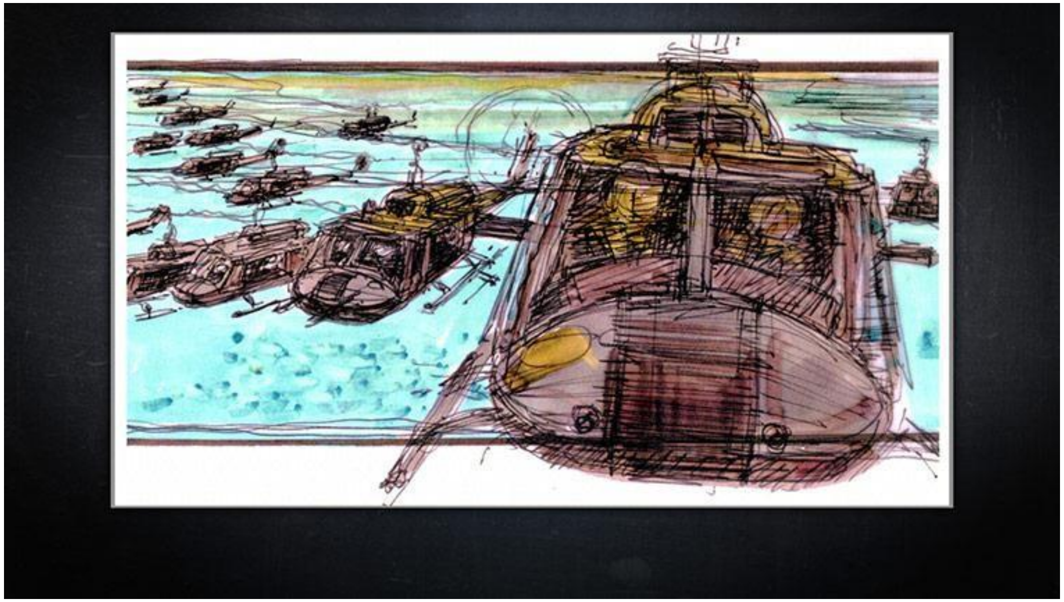
- A Storyboard ***"tells a specific story***"

- The goal in all techniques is to interact with the user
  - Helps to Bridge the gap between the user and the developer
  - Nonetheless, most of these interpretations agree that the purpose of storyboarding is to gain an early reaction from the users on the concepts proposed for the application.
  - With storyboarding, the **user's reaction can be observed very early in the lifecycle**. In so doing, storyboards offer an effective technique for addressing the ***"Yes, But"*** syndrome.
  - Storyboarding is also very effective for "Blank page" Syndrome

Storyboarding



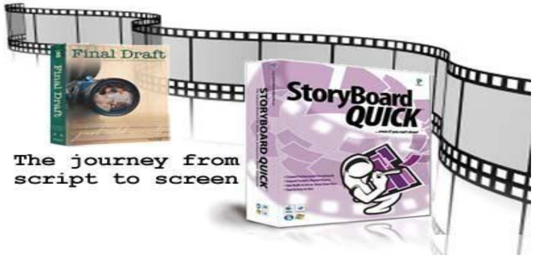
- Often used in *combination with another technique*
  - ✓ ***Capture ideas*** via brainstorming or interviewing
  - ✓ ***Document in a storyboard*** to show user what is understood
- Human factors experts have promoted storyboarding for years
  - ✓ This technique is used heavily in the **movie industry**



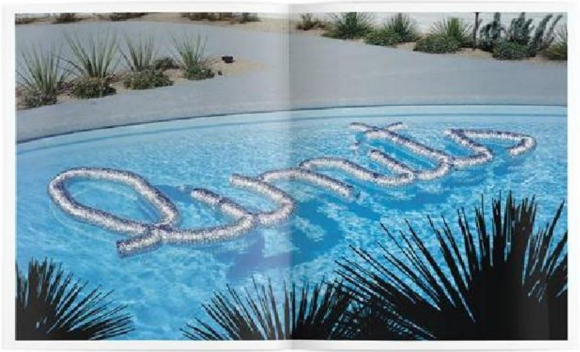
Benefits of Storyboard

- A storyboard serves multiple purposes
  - ✓ Provides an early review of the User Interface of the system
  - ✓ Is extremely inexpensive
  - ✓ Is user friendly, informal, and interactive
  - ✓ Is easy to create and easy to modify

Limitations of Storyboard

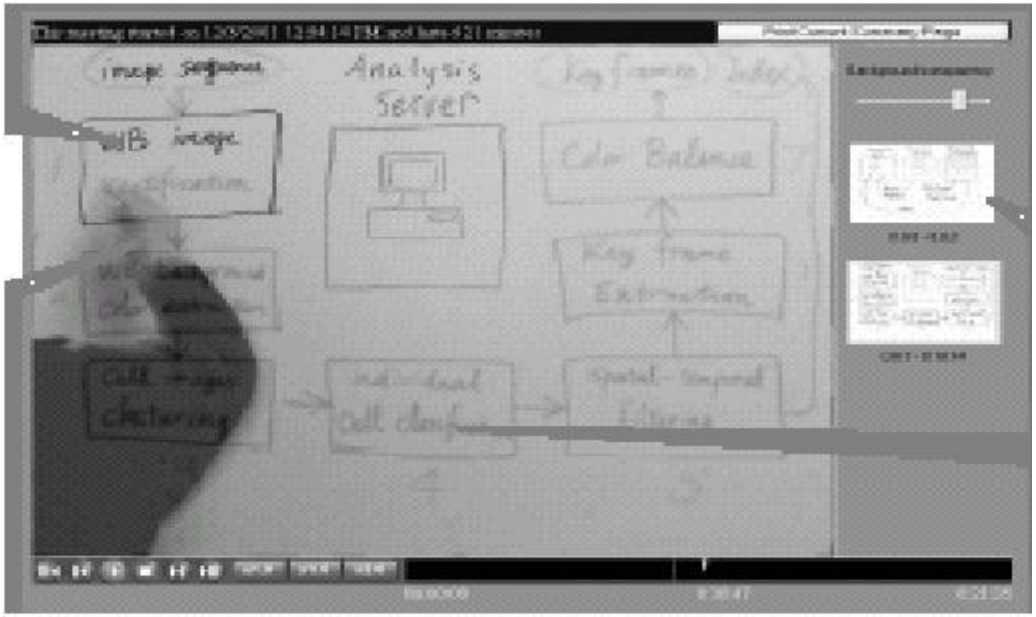


- One of the biggest problems with storyboards is that they can become obsolete very quickly. User ***interfaces originally defined often change over time***, and that creates a maintenance burden.



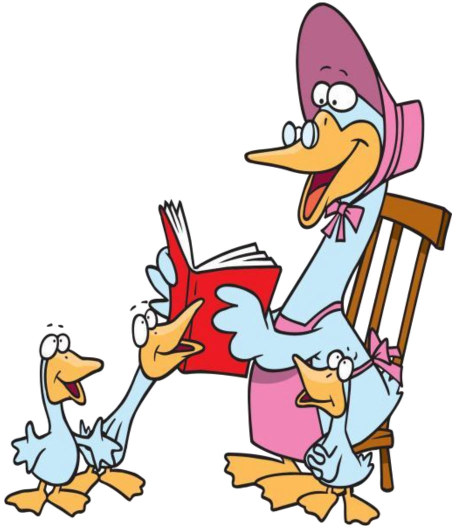
Types of Storyboards

- Passive Storyboards
- Active Storyboards
- Interactive Storyboards



Passive Storyboards

- Passive storyboards tell a story to the user.
- They can consist of ***sketches, pictures, screen shots, presentations, or sample application outputs***.
- In a passive storyboard, the analyst **plays the role of the system** and simply walks the user through the storyboard, with a "When you do this, this happens" explanation.



Active Storyboards

- Try to make the user see “a movie that hasn’t been produced yet”.

- Active storyboards are ***animated or automated***, perhaps by an automatically sequencing slide presentation, an animation tool, a recorded computer script or simulation, or even a homemade movie.

- Active storyboards provide an **automated description of the way the system behaves in a typical usage** or operational scenario.

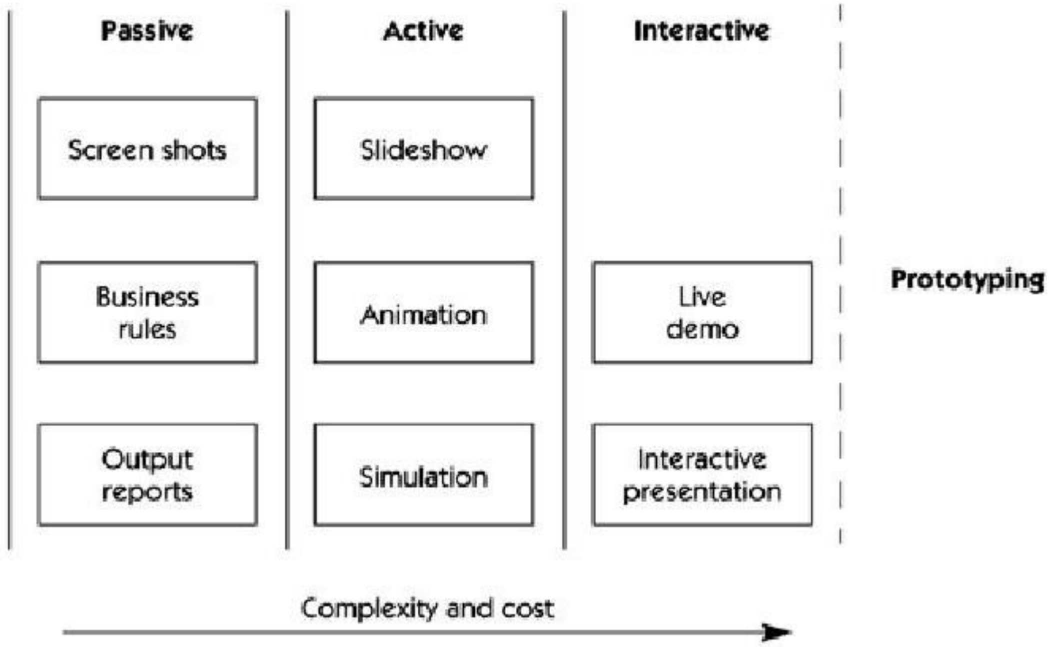
Interactive Storyboards

- Interactive storyboards let the user **experience the system in as realistic a manner as practical.**
- They require participation by the user.
- Interactive storyboards can be simulations or mock-ups [2] or can be advanced to the point of throwaway code.
- An advanced, interactive storyboard built out of throwaway code can be very close to a **throwaway prototype**.

Types of Storyboards

- As Figure 1 shows, these three storyboarding techniques offer a range of possibilities ranging from sample outputs to live interactive demos.

- Indeed, the boundary between advanced storyboards and early product prototypes is a fuzzy one.



What Storyboards Do?

- In software, storyboards are ***used most often to work through the details of the human-to-machine interface.***

- In this area, each user is likely to have a ***different opinion*** of how the interface should work.

- Storyboards for user-based systems deal with the three essential elements of any activity:
  - ✓ Who the players are
  - ✓ What happens to them
  - ✓ How it happens

If history were taught in the form of stories, it would never be forgotten.
- Rudyard Kipling

What Storyboards Do?

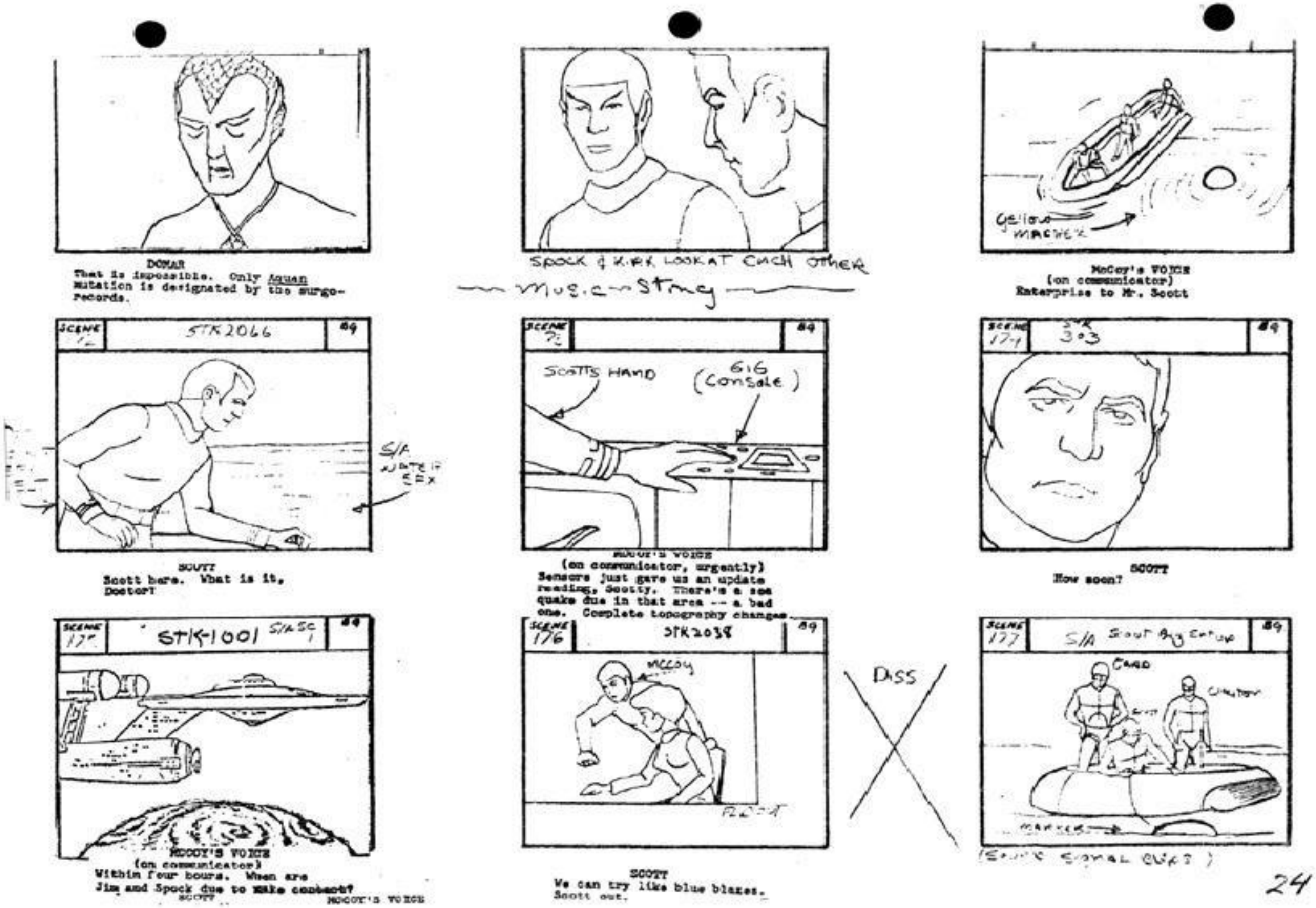
- The ***who element defines*** the players, or the ***users of the system***.
  - In a software system, the who are such players as users, other systems, or devices—in other words they are the actors that interact with the solution we are defining.
  - For users, the interaction is typically described via user input screens or data entry forms, outputs such as data or reports, or other types of input and output devices, such as buttons, switches, displays, and monitors.
- The ***what element*** represents the ***behavior of the users*** as they interact with the system as well as the ***behavior of the system*** as it interacts with the user.
- The ***how element*** provides descriptions of ***how this interaction happens***, showing *events, states, and state transitions*.

What Storyboards Do?

For example, a storyboard for an automated-vehicle amusement park ride includes the following things.

- The who represented the guests who ride on the vehicle.
- The what represented the behavior of the vehicle as it provided various events for the guests.
- The how provided further descriptions of how this interaction happens—events, state transitions—and described both the guest states (surprised, scared) and the vehicle states (accelerating, braking, unloading).

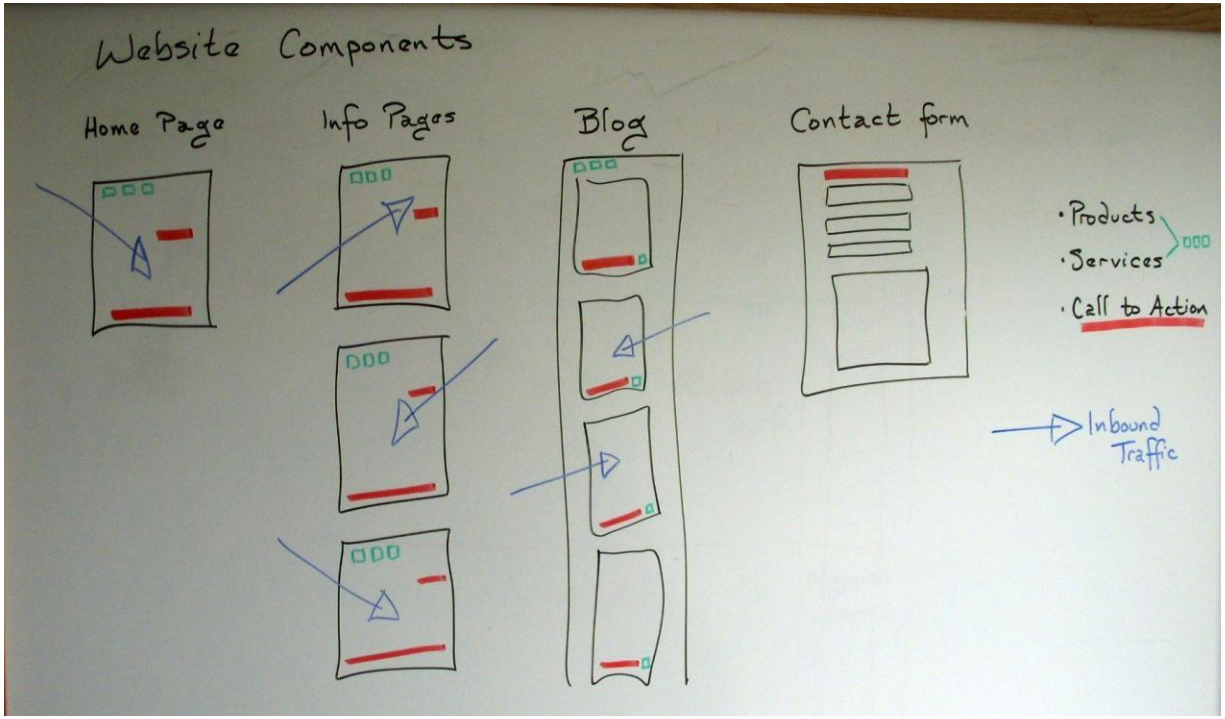
What Storyboards Do?


DOMAR
That is impossible. Only Aquan mutation is designated by the surgo-records.
SPOCK & KIRK LOOK AT EACH OTHER
MUSIC STING
Gellow Marker
McCoy's VOICE
(on communicator)
Enterprise to Mr. Scott
SCENE
STK 2066
89
S/A WATER EFX
SCOTT
Scott here. What is it, Doctor?
SCENE
89
SCOTT'S HAND
616 (CONSOLE)
MCCOY'S VOICE
(on communicator, urgently)
Sensors just gave us an update reading, Scotty. There's a sea quake due in that area -- a bad one. Complete topography changes.
SCENE
174
STK 303
89
SCOTT
How soon?
SCENE
175
STK 1001
S/A SC 1
89
MCCOY'S VOICE
(on communicator)
Within four hours. When are Jim and Spock due to make contact?
SCOTT
MCCOY'S VOICE
SCENE
176
STK 2038
89
MCCOY
ROCK
SCOTT
We can try like blue blazes. Scott out.
DISS
SCENE
177
S/A Scout Bug Set up
89
CARD
SCOTT
(SCOUT SIGNAL QUAT)
MARKER
24

Tools for Storyboarding

- The tools and techniques for storyboarding can be as varied as the team members' and users' imaginations will allow.

- ***Passive***-storyboarding constructs have been made out of tools as simple as *paper and pencil or Post-it notes*.



Tools for Storyboarding

- More advanced storyboards can be build with presentation managers such as PowerPoint.

- Passive, active, and user-interactive storyboards have been built with various packages that allow fast development of user screens and output reports.

- Interactive storyboards can be built with a variety of specialty software packages for interactive prototyping, and tools such as Macromedia's ***Director and Cinemation*** from Vividus Corporation and Xara's ***Graphics maker*** can be used to create more complex animations and simulations.

Tips for Storyboarding

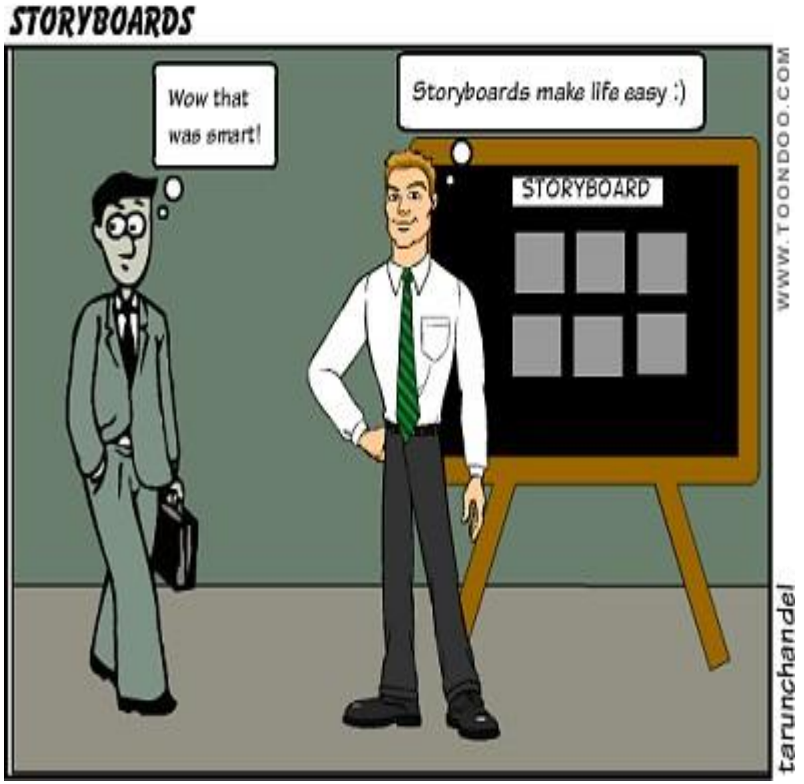
- Here are some tips to keep in mind as you practice your storyboarding technique.

  - ✓ Don't invest too much in a storyboard.
  - ✓ If you don't change anything, you don't learn anything.
  - ✓ Don't make the storyboard too functional.
  - ✓ Whenever possible, make the storyboard interactive.

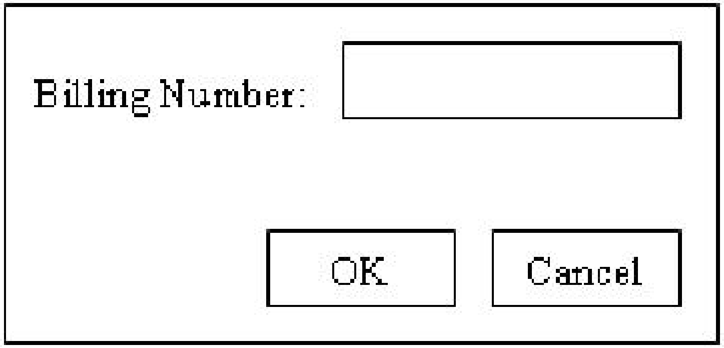
Who uses Storyboards?

The following people use the Storyboards:

- System analysts, to explore, clarify, and capture the behavioral interaction envisioned by the user as part of requirements elicitation.
- user-interface designers, to design the user interface and to build a prototype of the user interface;
- designers of the classes that provide the user interface functionality. They use this information to understand the system's interactions with the user, so they can properly design the classes.
- those who test to test the system's features.
- the manager to plan and follow up the analysis & design work.



Examples


Billing Number:

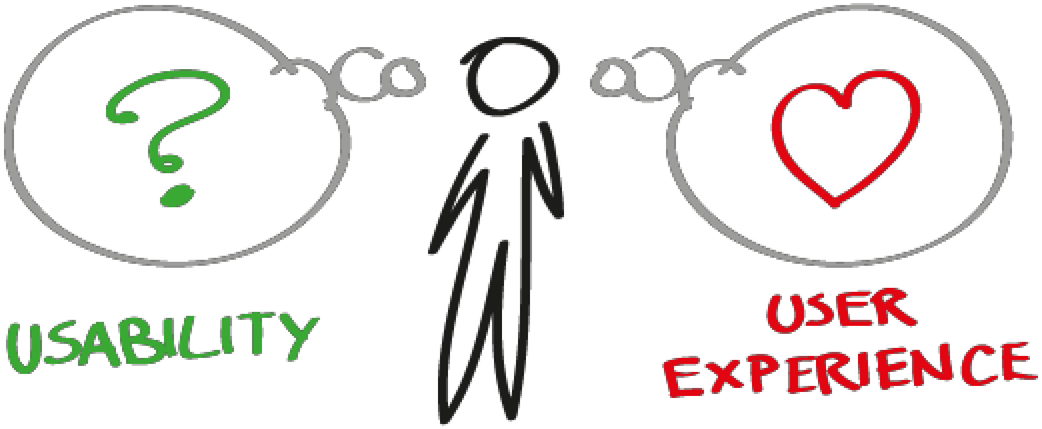
- Imagine that in reviewing this visual storyboard, our end-user accountant says, "Well, actually, billing numbers are divided into two parts, the year and a unique number.

- This drawing shows only one field for the account number." Then he adds: "And by the way, I don't want to enter the year all the time, so please initialize this value with the current year, which I can overwrite if necessary."

Role Playing

- Role playing allows the development team to experience the user world from the user's perspective by ***playing the role of the user***

- Role playing can be effective to address the following causes:
  - ✓ Many *users cannot communicate the procedures they follow* or the needs that must be addressed
  - ✓ Many users *do not have the freedom to admit that they do not follow* prescribed procedures; therefore what they tell you may or may not be what they actually do
  - ✓ Individual users may have *patterns of work activity* that requires workarounds or unique paths of implementation that may hide real problems
  - ✓ It is impossible for any developer *to anticipate every question that must be asked* or for users to know **"what questions the developer should be asking"**

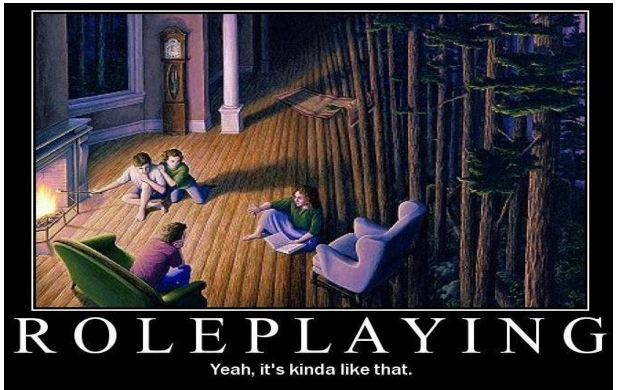


How to Role Play

- In the simplest form of Role playing, the *developer, the analyst, and potentially every member* of the development team simply **take the place of the user and execute the customers work activity**

- There are two ways to get at the root causes:
  - ✓ Use the Fish bone diagram, together with user’s input to analyze the existing system and finds the root cause
  - ✓ The developer/analyst can experience the problem and inaccuracies inherent in the existing system by simply sitting down and execute the system by himself

- By Playing the role of the user and using the system, the developer or analyst can gain a *better understanding of the problem of existing system*



Scripted Walkthroughs

- In scripted walkthrough *each participant follows a script that defines a specific role in the "play"*
- The walkthrough will demonstrates any *misunderstanding in the roles*, lack of *information available* to an actor or *lack of specific behavior needed* for the actors to succeed
  - ✓ For example consider the case of an **automated digital whiteboard** connected with several computers and we are interested to view the interaction of students and teacher with digital whiteboard
  - ✓ A prototype of the device can be used and had team members and customer representatives **play the roles of student and teacher**
  - ✓ The walkthrough contained ***multiple scenes***, such as "students accessing digital whiteboard through their computers and performing the tasks assigned by the teacher to them and "teacher using the white board for lectures"
  - ✓ The scripted walkthrough was very useful way to get the feel for the class room environment and the team learned a few new things in the process

Scripted Walkthroughs

- *One of the advantages of the scripted walkthrough is that a* ***script can be modified and rerun as many*** *times as necessary until the actors get it right*

- The script can also be ***reused to educate*** new team members

- The script may be modified and reused when the ***behavior of the system needs to be changed***

CRC (Class-Responsibility-Collaboration) Cards

- A ***derivative of Role playing*** is often applied as part of an object oriented analysis effort

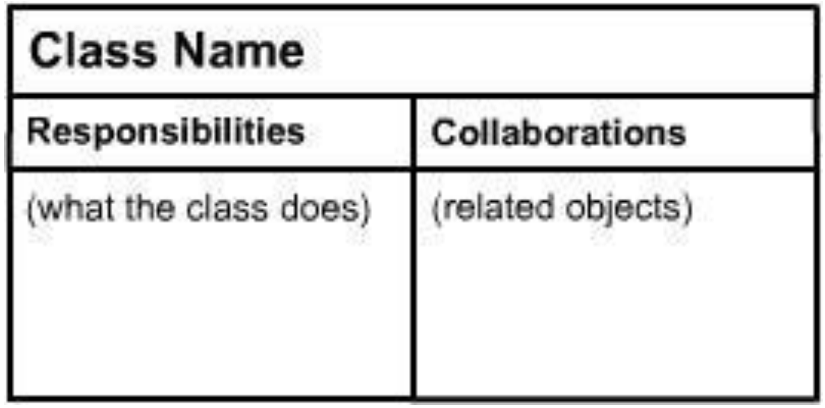
- In a special case of a Role play, each participant is given a ***set of index cards*** describing the class or object; the responsibilities, or behavior; and collaborations, or who the object communicates with, of each entity being modeled

CRC (Class-Responsibility-Collaboration) Cards

- These collaborations may represent entities from the problem domain, or objects that live in the solution domain

- When an ***indicator actor starts a specific behavior, all participants follow the behaviors defined on their cards***

- When the process breaks down due to a lack of information or when one entity talks to another and the collaboration is not defined, the cards can be modified and rerun again



Class Name	
Responsibilities	Collaborations
(what the class does)	(related objects)

References

1. Managing Software Requirements: A Use Case Approach, Second Edition By Dean Leffingwell, Don Widrig, Addison-Wesley
2. http://en.wikipedia.org/wiki/Mockup

For any query Feel Free to ask