Using Use Cases !!!

Raghavendra K S

Jan 2001

Introduction

What is a Use Case ?

- A use case is a ***Story*** ! ;-)
- A use case ***models*** a scenario!
- Aid to make functional Requirements ***readable***, ***reviewable*** and ***tractable*** !

Some Facts about Use cases

- A use case is a powerful tool to settle the functional requirements of your system with the client.
- To be trained to Read a properly written Use case takes a maximum of a couple of hours :-) but to Write one is 1000 times more difficult ! :-(
- Design is never done based on Use cases alone !

Some Facts about Use cases

contd...

- Help in Project estimates
- Help in brainstorming
- No standard (formal) form !!! :-)

Users, executives and developers appreciate seeing requirements in the form of goals.

Users:

"We can understand what these mean"

"You mean we are going to have to ...?"

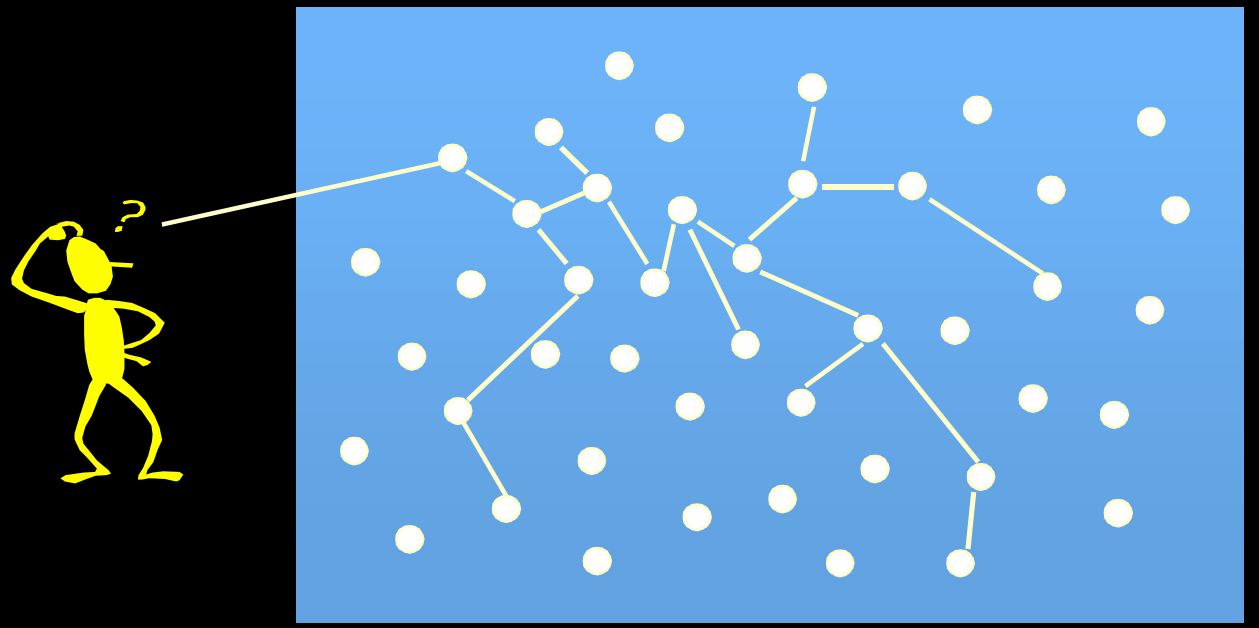
Executives:

"You left out one thing here ..."

Developers:

"These are not just a pile of paragraphs!"

Use Case Analysis is a technique to capture business process from user's perspective



Courtesy : Rational Rose

Agenda

- Cockburn's work (Use cases with **GOALS**)
- We will analyze the present Use cases
- some effective ways to collect the s/w reqts

Requirements Vs Use Cases

Chapter 1. Purpose and scope

1a. What is the overall scope and goal?

1b. Stakeholders (who cares?)

1c. What is in scope, what is out of scope

Chapter 2. The use cases

2a. The primary actors and their general goals

2b. The business use cases

2c. The system use cases

Chapter 3. The terms used / Glossary

Chapter 4. The technology to be used

4a. What technology requirements are there for this system?

4b. What systems will this system interface with, with what requirements?

Courtesy : CockBurn

SRDS *Contd ...*

Chapter 5. Other various requirements

5a. Development process

Q1. Who are the project participants?

Q2. What values will be reflected in the project (simple, soon, fast, or flexible)?

Q3. What feedback or project visibility do the users and sponsors wish?

Q4. What can we buy, what must we build, what is our competition to this sys-tem?

Q5. What other process requirements are there (testing, installation, etc.)?

Q6. What dependencies does the project operate under?

5b. Business rules

5c. Operations, security, documentation

5d. Use and usability

5e. Maintenance and portability

5f. Unresolved or deferred

Chapter 6. Human backup, legal, political, organizational issues

Q1. What is the human backup to system operation?

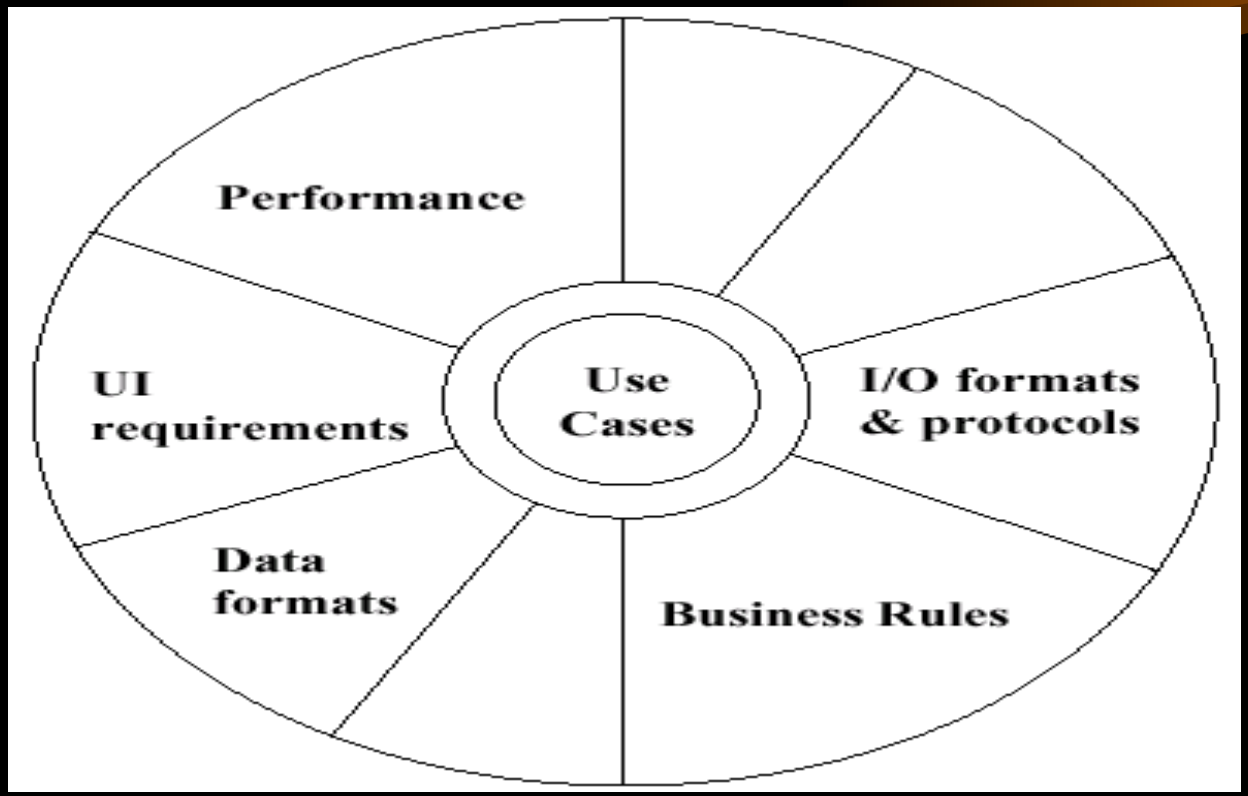
Q2. What legal, what political requirements are there?

Q3. What are the human consequences of completing this system?

Q4. What are the training requirements?

Q5. What assumptions, dependencies are there on the human environment?

"Hub-and-spoke" model of requirements.



Use cases connect many other requirements details.

Courtesy: Cockburn

Time to Rip off !

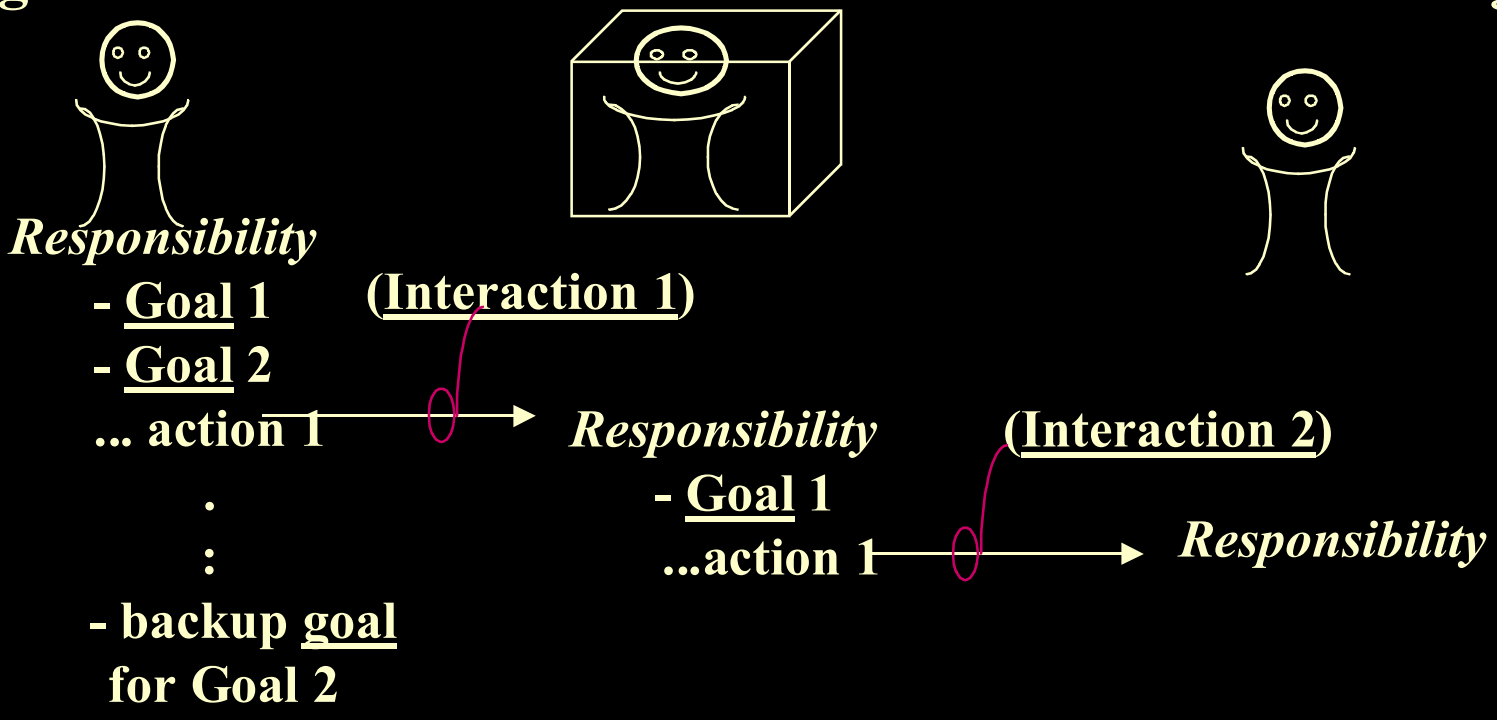
- Actors
  - Primary
  - Secondary
  - Tertiary
- Interactions
  - simple
  - compound
- Scenario
- Sequence
- Goals and Responsibilities

The basic model of use cases is that Actors interact to achieve their goals.

Primary Actor
person or system with goal for s.u.d.

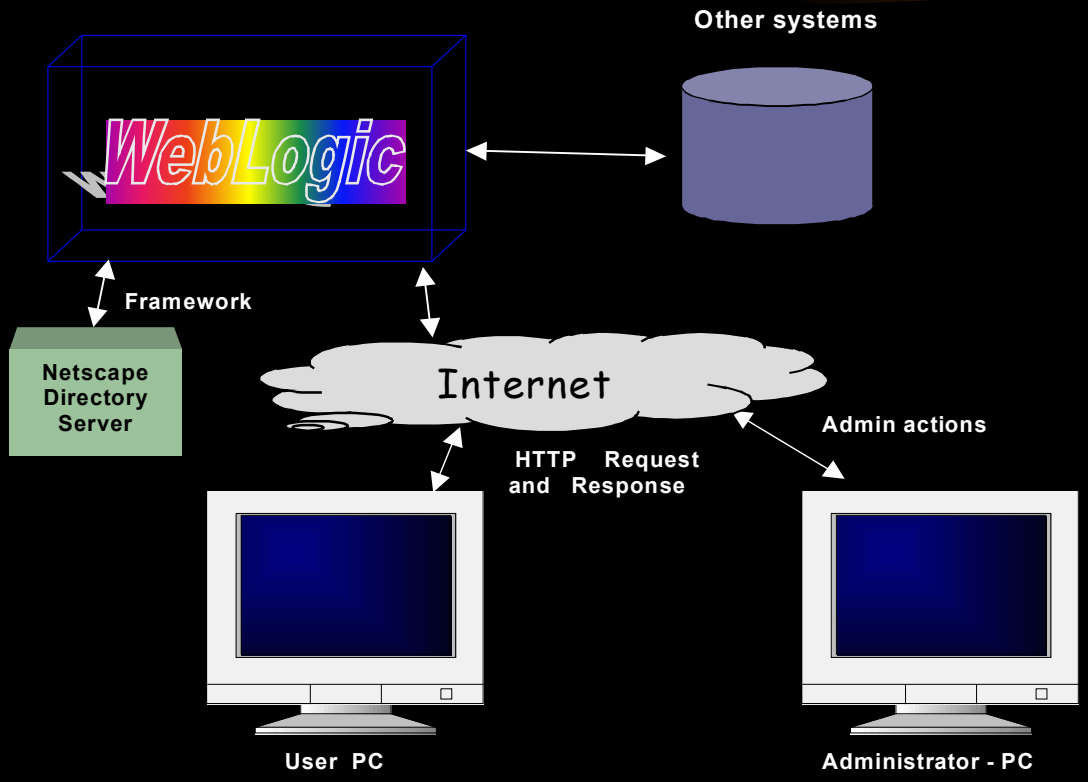
System under design
could be any system

Secondary Actor
other system against which s.u.d. has a goal



An Actor's Action Triggers an interaction, calling upon another actor's responsibility.

Let us work out a good example !



Let us Fix the Scope !

Item	In	Remarks
1) Admin Creation of User	Yes	Admin should be allowed to create users.
2) Admin Modification of User	Yes	Admin can modify a user's details
3) Admin Deletion of User	Yes	Admin can remove a user
4) Back Up by Admin	Ver 2	Time to time back up of the use details – not so important to the client right now.
5) Self Creation, Updation and Deletion	Version 2	The user can control his user details
6) Integration with X	Yes	X will use UA system for validation and user information
7) Integration with Y	Ver 2	Right now client is not intereste in this.
8) Integration with z	Out	Z is too complicated.

Another way to fix the scope

- Step 1:

  Primary Actors :

  1) HNS administrators

  2) HNS service users

  3) Systems X,Y and Z.

  Tertiary Actors :

  1) CEO of HNS

Another way to fix the scope *contd ...*

- Step 2:

Goals :

Primary Actor	Goal	V	Goal Briefs
1) Administrator	To manage (create,delete and modify) user details.	1.0	Admin should be able to manage the details of the users.
2) User	To self manage (create,delete and modify) details.	2.0	User wants to maintain self details.
3) Administrator	To back up user details	2.0	Admin wants to back up user details from time to time.
4) System X	Use the system for authentication and for user profile.	2.0	X needs user profiles. Also X will use the system for authentication.
5) System Y	Use the system for authentication	2.0	Y will use it for authentication
6) System Z	Use the system for authentication and for user Pofile	Out	Z is too complicated to understand and the client is not interested in integrating it – he wants higher security for Z.
….	….	..	….

Courtesy: Cockburn

Primary Actors - Are they Important ?

- Beginning of the Requirements gathering
  - Do not discuss the goals directly - identify the primary actors and brain storm on their goals.
- Just before System delivery Date
  - Partition system into various packages
  - Set security levels
  - Establish Training

Otherwise we do not care !

Creating a slightly more number of primary actors does not hurt !

For heaven's sake do this one thing !!!

Please identify the tertiary actors and understand their '**expectations**' !

Aim : To shift the security level from Application level to the web server level.

How much do we need to dress them up ?

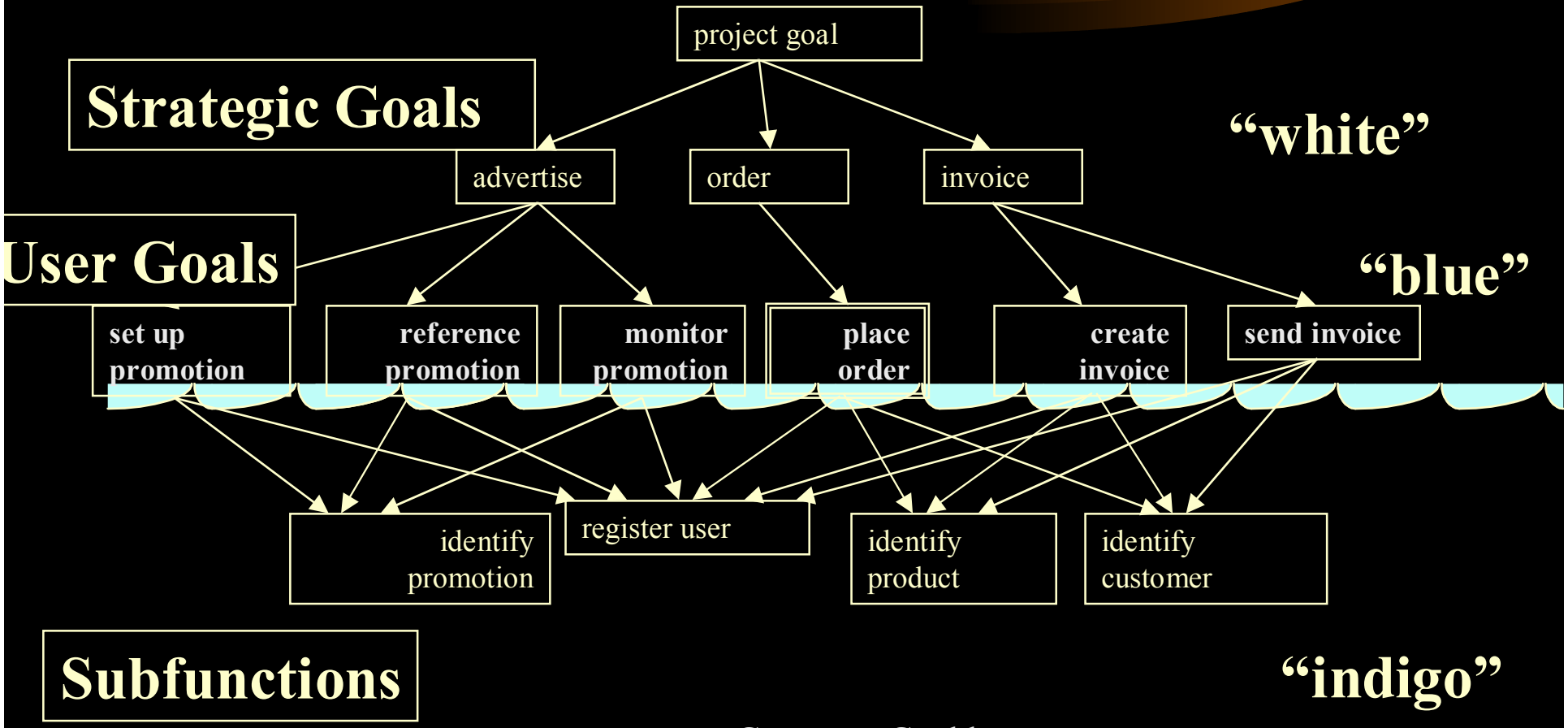
- Business Use cases
- System Use Cases

What is the Scope of your Use Case ?

- Organization
- System Scope

Summary use cases, tasks, and subfunctions form a graph

- Sailboat image: User goals are at sea level.



Courtesy:Cockburn

*User Goal Level(*Blue*)*

- It is done by one person,in one place,at one time. (2- 20 mins)
- The actor can go away happily as soon as this goal is completed.
- The actor can ask for a raise after doing many of these.

Note : The shortest summary of the function of a system is the list of blue goals it supports. This list will be the basis for prioritization, delivery,team division,estimation and development.

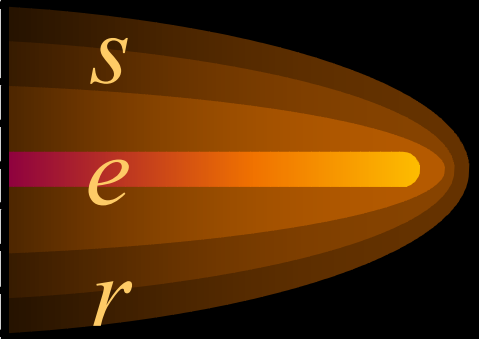
Summary Level (White)

- They show the context in which the system services will function.
- They show life-cycle sequencing of related goals
- They provide a TOC for the lower-level use cases
- They typically execute over many hours, days,weeks,months or years.

Sub Function (Indigo)

- Needed on occasion for readability
- Needed on occasion because many other goals use them.
- Examples : “Find a whatever” !

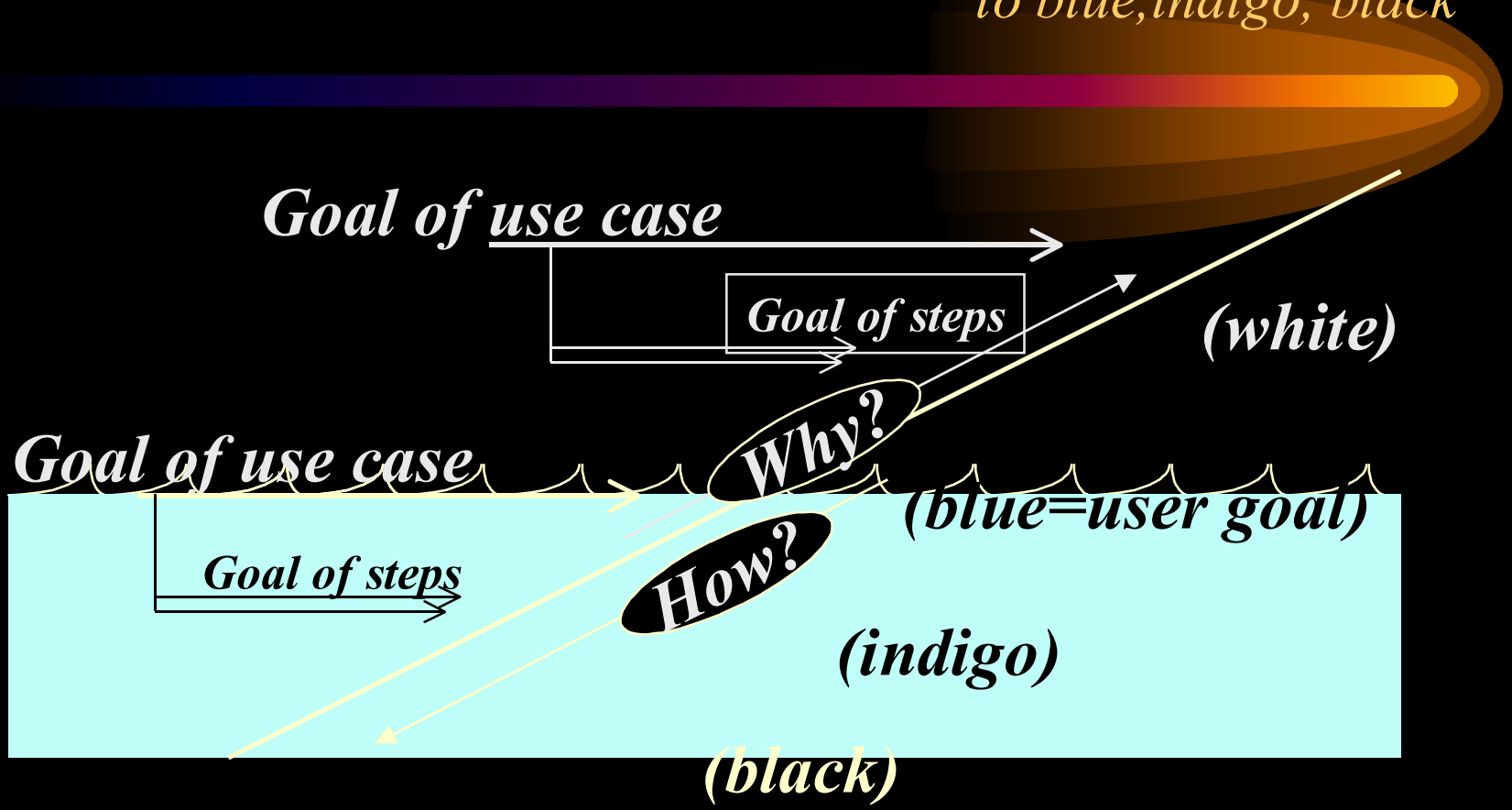
Use Case #	2.00		
Conext of Use:	A new user entry needs to be made		
Scope :	UA system		
Level	User Goal		
Primary Actors	The Admin		
Stake Holders and Interests	Stake Holder	Interests	
	The Admin	Create a new user	
Preconditions	-		
Successful End Condition	The user is registered.		
Failed End condition	The user registration fails		
Failed End Protection	-----		
Trigger	The admin wants to create a new user		
Description	Step	Action	
	1	Admin enters user details	
	2	The UA system validates the user details	
	3	The UA system makes a new user entry.	
Extensions			
	2a	The UA system finds user already exists	
		Give another chance to the admin	
	2a	The UA system finds some inconsistency in user details	
		Give another chance to the admin	
Variants			
	1	User Details	Profile info
			Access privileges



A Recap

- Main case scenario - The rosy path !!! :-)
- Extension cases - The villains come in !!! :-(
- Variants List - One shot solution ! ;-)

The use case goal level is higher than the steps'. They white to blue,indigo, black



Precision can be varied but not accuracy !!!

Finding the right goal level

Find the user goal level :

- What does the user want from the system **NOW** ?
- After getting it, can he go and do some other work ?

Keep asking WHY to summarize !

Keep asking HOW to decompose !

PLEASE do not do the following!

- Please realize that to describe a system in terms of screen is a very **naïve** approach (That will be the last thing I would ever do !!!)
- UI design is NOT my job as a use case writer !
- UI design is not even a part of SRDS (but definitely I suggest that you append it to SRDS).

Let us digress a bit … How to specify UI ?

The UI design document can be a aid to understand the system - as it **"evolves"** please feel free to attach it to the SRDS.

Two levels of Precision :

- Low Precision view : a screen-flow diagram drawn as a finite state machine or state chart.
- Medium Precision view : a bitmap, reduced size snapshot of the designed UI.

Back into Use Cases ... when are we done ?

- You have captured all the primary actors and their user goals wrt to the system
- The use cases handle every trigger condition that can touch the system.
- Have written all the summary and subfunction use cases
- The sponsors read them and agree that is all that they want
- each use case is clear enough written that
  - The sponsors agree they will be able to tell whether or not it is actually delivered.
  - The users read,understand and agree
  - The developers read,understand and agree that they can actually develop that functionality.

Courtesy : Alaistar Cockburn

Mapping to Tasks, Design, UI and Testing.

Use cases to design :

- Design doesn't cluster by use case
- Blindly following the use case structure leads to functional decomposition designs ! :-(

Silver Lining :-)

Functional Decomposition is good for requirements does NOT in any way mean that it is good for s/w designing.

Encapsulation of data and behavior definitely lends the design to be maintainable and easily evolvable but it is not proven that it is a good technique/paradigm to capture requirements.

PLEASE be mature enough NOT to translate a use case directly to an object !!! Please exercise your gray cells while you design !!!

How do use cases help in Testing ?

- Acceptance test plans for Main case scenario
- Acceptance test plans for Extensions
- Please prepare Acceptance test plans for Variations too !

How do you plan ?

Most Use Cases show an easy way to plan resources.

However there is a small problem.

- The problem is that different versions need different levels of ‘maturity/robustness’ of different use cases !
  - Write different versions of ‘the’ use case
  - Write in different fonts/colors in one use case

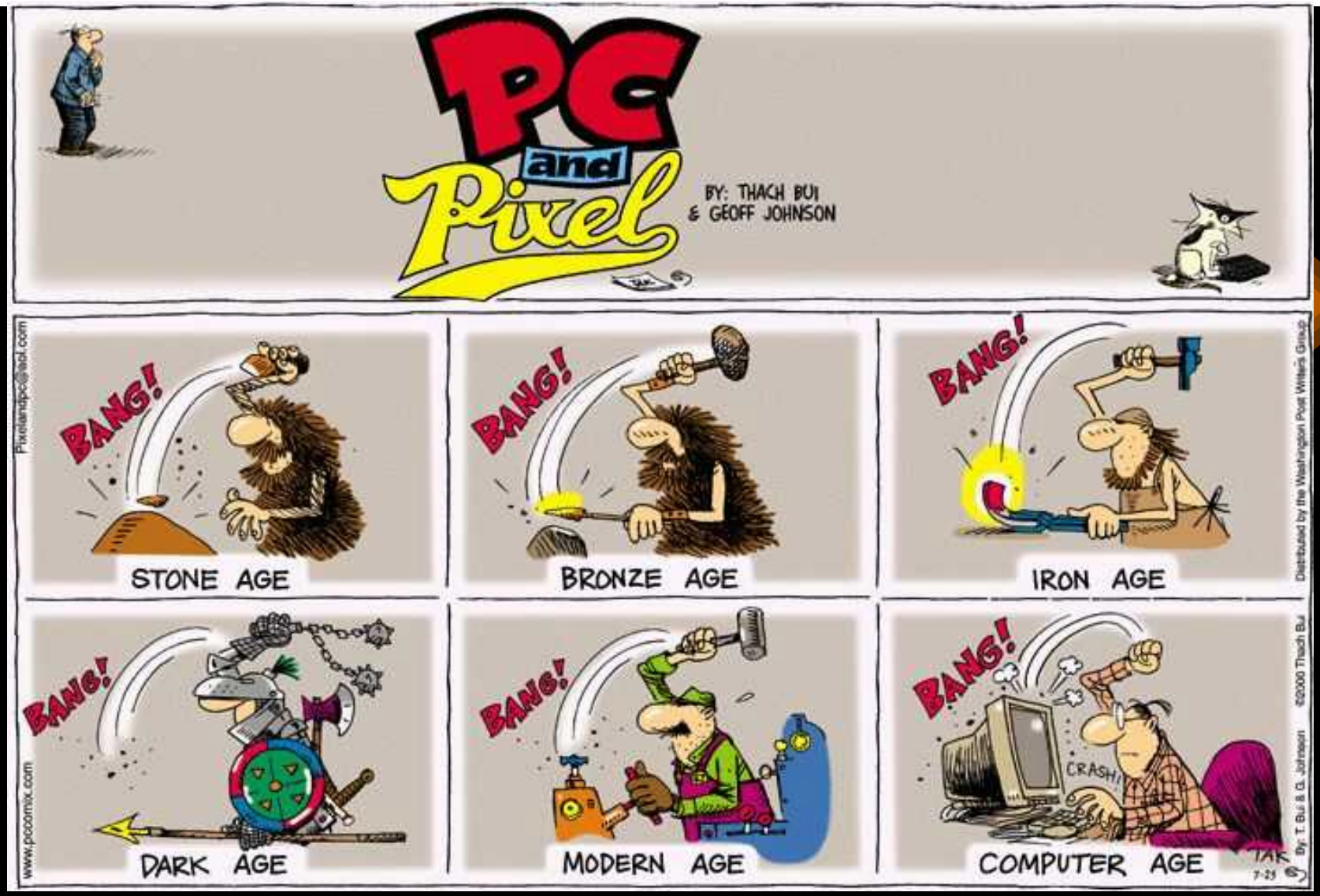
A suggested Framework

Use Case Number				
Actors	<Actors involved>			
Goal	<Goal of the primary Actor>			
Business Need	<Top|High|Medium|Low>			
Technical Difficulty	<High|Medium|Low>			
Priority	<ordering in use cases>			
Remarks	<what the customer feels>			
Developers	<people working on the use case>			
Planned Start Date				
Planned End Date				
Actual Start Date				
Actual end Date				
Development Status	<record the status of development>			
Development Notes	<remarks about development>			

The simpler .. The Sweeter !!! ;-)

- Move up to summary goals
- ***KICK*** UI from use cases
  - Rebecca differs here - she introduces a two columns dialog (conversations)
- Use Variants

The AK47 Round !!!



Hope this is not the effect of the presentation !!! *;-)*

Time to say Adios !!!!

All the Best with your projects !

Think it Over : S/w engineering is more of a game than engineering ! It is a game where you need to deliver the ‘system’ with limited people,time and money ! And a smart methodology will always have a greater probability to win !