Software Requirements Introduction

Lecturer: Ngo Huy Bien
Software Engineering Department
Faculty of Information Technology
VNUHCM - University of Science
Ho Chi Minh City, Vietnam
nhbien@fit.hcmus.edu.vn

Objectives

- To present *why* use requirements
- To present *what* is a requirement
- To *categorize* requirements
- To create an *executive summary*
- To create a *project vision*
- To create a *project charter*
- To *elicit* software requirements
- To write a *software requirement specification*
- To *validate* a requirement

References

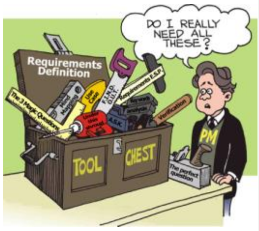
1. Suzanne Robertson and James Robertson. Mastering the Requirements Process. 2nd Edition. Addison Wesley Professional. 2006.
2. Klaus Pohl et al. Software Product Line Engineering: Foundations, Principles, and Techniques. 2005.
3. Craig Larman. Applying UML And Patterns. Addison Wesley Professional. 2004.
4. Alan R. Hevner et al. Design Science in Information Systems Research. MIS Quarterly. 2004.
5. Jennifer Greene and Andrew Stellman, Applied Software Project Management. 2005.
6. Peter Pin-Shan Chen. Entity-Relationship Model. 1976.
7. Till Schummer and Stephan Lukosch. Patterns for Computer-Mediated Interaction. John Wiley & Sons. 2007.
8. Don Batory. Feature Models, Grammars, and Propositional Formulas. 2005.
9. Project Management Institute. A Guide to the Project Management Body of Knowledge. 5th Edition. 2013.

Useless Software! Wrong Feature!

what stakeholders need
what is built

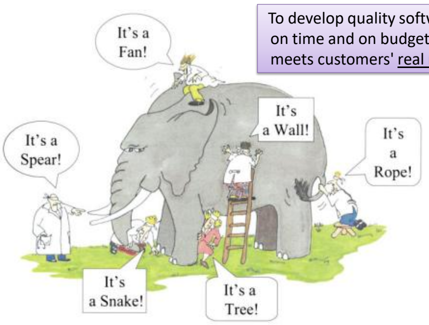
What is a Requirement? [1, 2]

- *Requirements* are capabilities and conditions to which the system must conform.
- A requirement is something the product *must do* or a *quality* it must have.
- *Functional requirements* are things the product must do.
- *Nonfunctional requirements* are qualities the product must have.
- *Constraints* are global requirements.



Why Do I Need Requirements?

To develop quality software—on time and on budget—that meets customers' real needs.



Where to Start? – Begin with the *Business* Requirements

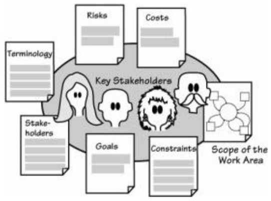
Clients

Users

Stakeholders

Project Blastoff Meeting [1]

- *Blastoff* is also known as "project initiation," "*kickoff*," "charter," "project launch," and many other things.
- The blastoff meeting prepares the project and ensures its *feasibility* before launching the detailed requirements effort.



Domains of Interest

- "*Roads* freeze in winter, and *icy conditions* cause road accidents that kill people.

  We need to be able to predict when ice will form on a road so we can *schedule* a *de-icing truck* to treat the road in time.

  We expect a new system to provide more accurate *predictions* of icy conditions.

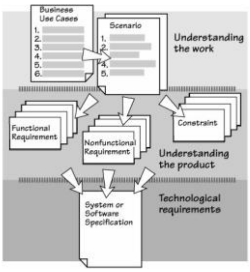
  This will lead to more timely de-icing treatment than at present, which will reduce road accidents.

  We also want to *eliminate* indiscriminate treatment of roads, which wastes de-icing compounds and causes environmental damage."
- Pay attention to the *subjects*.
- Four *domains*: Roads, Weather, Scheduling, Trucking.

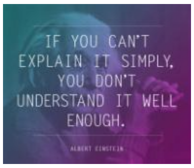
UNCLEAR

Evolution of Requirements

- They start out as fairly *vague ideas* as the analysts and stakeholders explore the work area.
- As the ideas for the product emerge over time, the requirements become *precise* and *testable*.
- They remain *technologically neutral* until the designer becomes involved and adds those requirements needed to make the product work in its technological environment.

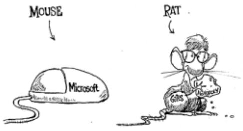


Terminology

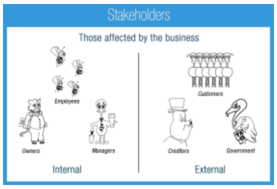


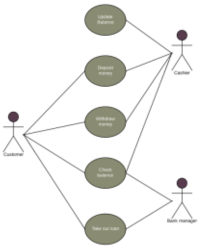
IF YOU CAN'T EXPLAIN IT SIMPLY, YOU DON'T UNDERSTAND IT WELL ENOUGH.

ALBERT EINSTEIN



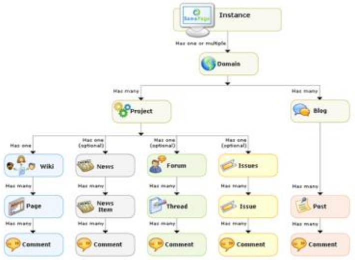
Stakeholders





Conceptual Structure

Aspect: The system's *concepts*.



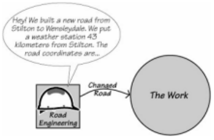
Setting the Scope

Tầm ngắm

- *The scope* you are interested in is the scope of the work for which the product is to be used.
- *Work* here means the business activity for which the user needs the product.
- Setting *the scope* of the work means you determine what work you are about to study, what other pieces of work surround it, and what flows of information make up the connections.
- When you set the scope, you are deciding *how much* of the work you will study and what you will not study.

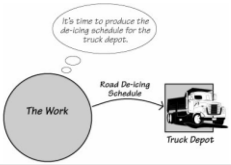
Business Events

- Any system or piece of work responds to *things* that happen outside it. We call these happenings *business events*.
- A business event takes place *outside* the scope of the work.
- The work learns that it has happened through the arrival of an *incoming flow of information*.
- Business events are *determined using* the flows from the adjacent systems on the context diagram.


Hey! We built a new road from Stilton to Wensleydale. We put a weather station 43 kilometers from Stilton. The road coordinates are...
Changed Road
The Work
Road Engineering

Time-Triggered Business Events

- A *time-triggered* business event happens when a prearranged time is reached.
- This is based on either
  - a *periodic occurrence* (for example, the end of the month, or 5 P.M. each day),
  - a *fixed time interval* (three hours since the last occurrence), or
  - a certain *amount of time elapsing* since another business event (30 days after sending out an invoice).

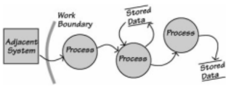

It's time to produce the de-icing schedule for the truck depot.
The Work
Road De-icing Schedule
Truck Depot

Finding the Business Events

- Each of the flows that *enters* or *leaves* the work is the result of a business event.
- You need some ***knowledge of the work*** to figure out the business events.
- Start the process of determining business events *during blastoff*, when the key stakeholders are present.

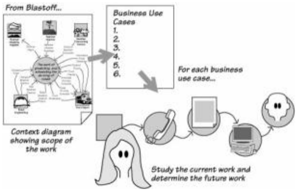
Business Use Cases

- When a business event happens, the work responds by initiating a *business use case*.
- The business use cases are the work's responses to the business events.
- The work's response to the business event is to continue processing until all active tasks (the processes) have been completed and all data retrieved or stored.
- The business use case is a collection of *identifiable* processes, data that is retrieved and/or stored, output generated, messages sent, or some combination of these.



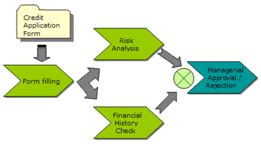
Studying Business Use Cases

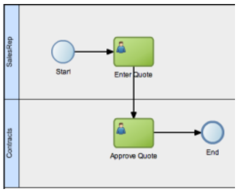
- From the work context diagram, you *determine* the business events and the resulting business use cases.
- The business use cases are studied *until* the analyst understands the desired functionality of the work and the part of that functionality to be performed by the product.


From Blastoff...
Business Use Cases
1.
2.
3.
4.
5.
6.
For each business use case...
Context diagram showing scope of the work
Study the current work and determine the future work

Business Process Model

Business process is a collection of related, structured activities or tasks that produce a specific service or product (serve a particular goal) for a particular customer or customers.





Storyboards [1]

- *Storyboards* are a prototyping technique borrowed from the film and cartoon industries. When a cartoonist is planning a cartoon, he sketches a number of linked pictures. These pictures identify the story line and guide the cartoonist in how many detailed pictures he needs to draw.
- *Building a storyboard* means thinking of the proposed functionality as a story and breaking it into a series of steps, or discrete actions. Draw each action as a panel of the story.

Brad and Janet take the family away for the weekend.

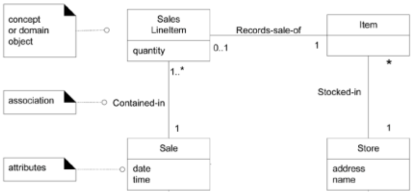
HomeSafe senses the car leaving, locks the doors, and turns on the burglar alarm. It also activates the random lights program. HomeClimate turns down the heating.

Before they return, Janet phones HomePlate to check the food supplies at home. She asks HomePlate to thaw a frozen chicken.

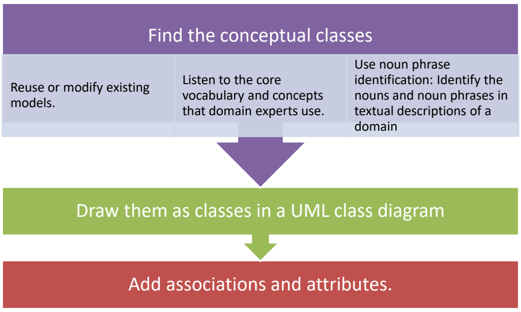
From the driveway, Brad phones HomeSafe to open the garage door and unset the burglar alarm. HomeMusic plays their favorite homecoming music. HomePlate starts cooking the chicken.

Domain Model

- Informally, a *conceptual class* is an idea, thing, or object that has symbol, intention and examples.
- A *domain model* is a visual representation of conceptual classes or real-situation objects in a domain.
- *Elements* of a domain model are domain object classes, and the relationships between them.



How to Create a Domain Model?



Find the conceptual classes

Reuse or modify existing models.

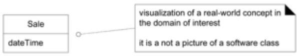
Listen to the core vocabulary and concepts that domain experts use.

Use noun phrase identification: Identify the nouns and noun phrases in textual descriptions of a domain

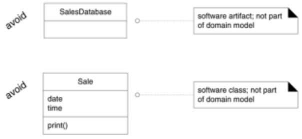
Draw them as classes in a UML class diagram

Add associations and attributes.

Domain Model vs. Software Business Objects


Sale
dateTime
visualization of a real-world concept in the domain of interest
it is a not a picture of a software class

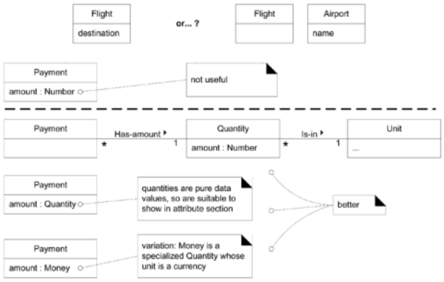
A DOMAIN MODEL SHOWS REAL-SITUATION CONCEPTUAL CLASSES, NOT SOFTWARE CLASSES.


avoid
SalesDatabase
software artifact; not part of domain model
avoid
Sale
date
time
print()
software class; not part of domain model

A DOMAIN MODEL DOES NOT SHOW SOFTWARE ARTIFACTS OR CLASSES.

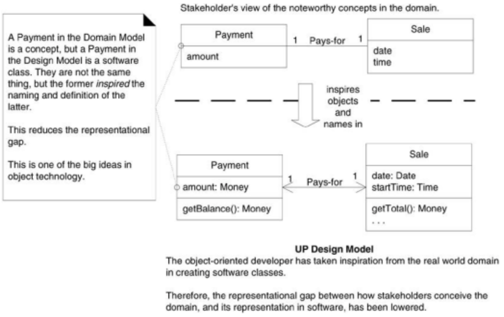
Attributes vs. Classes

If we do not think of some conceptual class X *as* a number or text or date/time in the real world, X is probably a conceptual class, not an attribute.



Why Domain Model?

This supports a *low representational gap* between our mental and software models.



A Payment in the Domain Model is a concept, but a Payment in the Design Model is a software class. They are not the same thing, but the former *inspired* the naming and definition of the latter.

This reduces the representational gap.

This is one of the big ideas in object technology.

UP Design Model

The object-oriented developer has taken inspiration from the real world domain in creating software classes.

Therefore, the representational gap between how stakeholders conceive the domain, and its representation in software, has been lowered.

DON'T PANIC

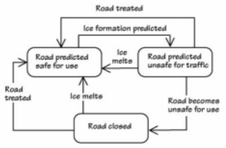
Business Model

- A *domain model* is defined as an abstract model that captures the most important types of objects in the context of the system.
- The *domain objects* represent the "things" that exist or events that transpire in the environment in which the system works.
- A *business model* comprises two models:
  - a *business use-case model* to describe the business actors and the business processes,
  - and a *business object model* to describe business entities used by the business use cases.

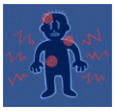
DON'T PANIC

State Model

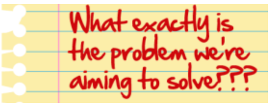
- A *state model* (also known as a state transition diagram) is a model that shows which states the objects for a class (sometimes called entities) may take, and illustrates the transitions between those states.
- A *state* is a steady condition for the object, and each rectangle in the model identifies a different state that this object will be in at one time or another.
- The story is told not so much by the states, as by the *transitions between states*.


Road treated
Ice formation predicted
Road predicted safe for use
Ice melts
Road predicted unsafe for traffic
Road treated
Ice melts
Road becomes unsafe for use
Road closed

Problem-Definition [4]



- Formally, a *problem* can be defined as the differences between a goal state and the current state of a system.
- *Problem solving* can be defined as a search process using actions to reduce or eliminate the differences.



What exactly is the problem we're aiming to solve???

- ❖ PAIN
- ❖ FEAR
- ❖ HAPPINESS

1. WE CANNOT…
2. IT'S DIFFICULT…
3. IT COSTS TOO MUCH…

Problem Relevance

- *The relevance* of any design-science research effort is with respect to a constituent community.
- For IS researchers, that *constituent community* is
  - the practitioners who plan, manage, design, implement, operate, and evaluate information systems and
  - those who plan, manage, design, implement, operate, and evaluate the technologies that enable their development and implementation.
- To be relevant to this community, research *must* address
  - the problems faced and
  - the opportunities afforded by the interaction of people, organizations, and information technology.

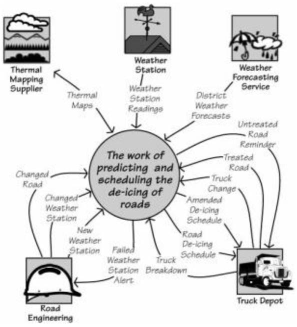
Business Value (Benefits)

- Reduce *costs (time)*
- Improve *customer service*
- Improve *communication*
- Improve *decision making*
- Create or *strengthen relationships* with customers, or partners
- Improve *processes*
- Improve *reporting* capabilities
- Support new *legal* requirements
- Bring more *pleasure*
- Create a *new* product or service. Is this valuable?

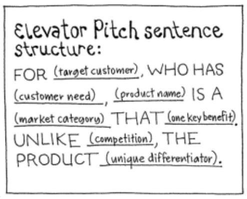
Be measurable

First-Cut Work Context

- The *work context diagram* identifies the scope of the work that we intend to study.
- It shows the work as a *single*, as-yet uninvestigated process, surrounded by the adjacent systems.
- The work context shows where the *responsibilities* of the work and the responsibilities of the adjacent systems start and end.
- First understand the work, then decide which *product* best supports that work.



Evaluator Pitch



Elevator Pitch sentence structure:

FOR (target customer), WHO HAS (customer need), (product name) IS A (market category) THAT (one key benefit). UNLIKE (competition), THE PRODUCT (unique differentiator).

Example

- For [construction managers]
- who [need to track what type of work is being done on the construction site],
- the [CSWP*]
- is a [safety work permit system],
- that [creates, tracks, and audits safety work permits].
- Unlike [the current paper-based system]
- our product [is web based and can be accessed any time from anywhere].

*CSWP: Construction Safety Work Permit

Executive Summary

- Elevator pitch
- Market/Target audience (*customers* , users)
- Pain points/needs/happiness
- Customer *discovery*
  - Where? – universities, hospitals, streets
  - How? – survey, demo, trial version
  - What to survey? – personal information, pains/needs/happiness, reasons, comments
- Competitors/weakness
- Differentiators
- Risks/Strengths/Opportunities
- Predictable *incomes*
- Grand vision

Goals [1]

- Goals: What Do You *Want* to Achieve?
- [Oxford Dictionary] Something that you *hope* to achieve.
- The *project goal* is the highest-level requirement.
- All of the detailed requirements must make a positive *contribution* toward reaching that goal.

Goal Settings

- *Purpose*: To accurately forecast road freezing times and schedule de-icing treatment.
- *Advantage*: To reduce road accidents by eliminating icy road conditions.
- *Measurement*: Accidents attributed to ice shall be no more than 15 percent of the total number of accidents during winter.
- *Vague purpose*: To improve the way we do business.

SMART Goals

A SMART goal is a well-defined target that gives you clarity, direction, motivation, and focus.

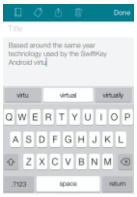
- What? Why? Who? Where? Which?
- How much? How many? How will I know when it is accomplished?
- Which steps? How can the goal be accomplished?
- Does this seem worthwhile?
- When? What can I do 6 months from now? What can I do 6 weeks from now? What can I do today?

Features

- A *feature* is an end-user visible characteristic of a system.
- A *feature tree* hierarchically structures the set of features of a system.
- A feature can be decomposed into several sub-features that are *mandatory*, *optional*, or *alternative*.

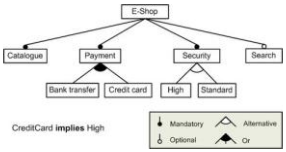
How to Indentify a Correct Feature?

- Spot the problem
  - Talk to a client
- Talk to two
  - Find *two other clients* in that same industry
- Understand it
  - Just *listen* and understand their problem
- Prototype it
  - Walk them through how you *imagine* it working.
  - Are they confused? Are they excited?
- Inspect them
  - Build a functioning prototype and *watch them use it*
- Build it

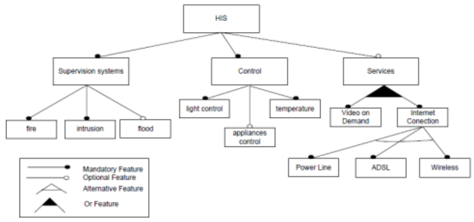


Feature Model [8]

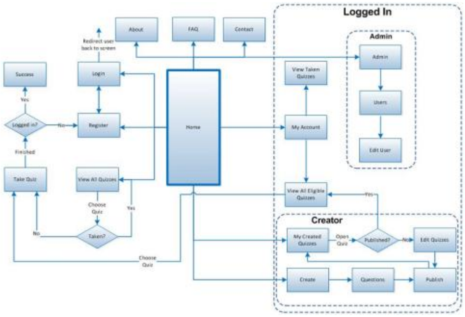
- A *feature model* is a hierarchically arranged set of features.
- *Relationships* between a parent (or compound) feature and its child features (or sub-features) are categorized as:
  - *And* — all sub-features must be selected,
  - *Alternative* — only one sub-feature can be selected,
  - *Or* — one or more can be selected,
  - *Mandatory* — features that required, and
  - *Optional* — features that are optional.
- A *feature diagram* is a graphical representation of a feature model.



Feature Model Example



User Interface Flow Diagram



Project Deliverables

Item	Assignee	Approved By	Due Date
Requirement specification			
Test specification			
Design specification			
Schedule			
Source code			
Test data and result			
User guide			

Project Vision [5]

A *project vision* is the picturing of the project's deliverable as the solution to the stated need or problem.

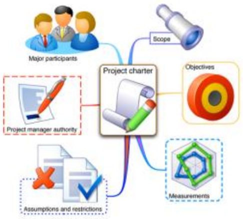
- Background, context, overview
- *The problem* that the project will solve, the stakeholders, the users
- *The solution* (vision statement), business cases
- The work context diagram
- The reasons to solve that problem (*objectives*, *benefits*)
- The related work
- The budget, the timing
- *The features* that will be developed to obtain the objectives
- The features that will *be excluded*
- Deliverables
- The risks, the assumptions
- Conclusion

Vision and Scope Document

Project Charter [9]

A document that formally *authorizes* a project or a phase and documenting initial requirements that satisfy the stakeholder's needs and expectations.

- Executive Summary
- Vision and Scope Summary
  - Business Need and Background
  - Project Description and Scope (Solution)
  - Project Goals
  - Project Deliverables
  - Constraints
- Project Management and Governance
  - Role/Name/Responsibilities
- Project Facilities and Resources
- Project Schedule and Cost
- Impact Analysis
  - Who? What?
- Assumptions
- Signatures



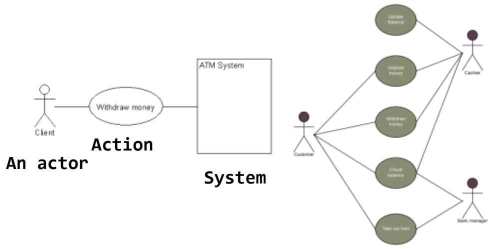
Software Requirements Elicitation

Requirements Analysis and Specification

SOLUTION-INDEPENDENT MODELS

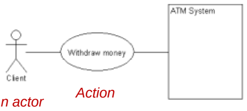
Functional Requirements [3]

- An *actor* is something with behavior, such as a person (identified by role), computer system, or organization.



Scenario and Use Case

- A *scenario* is a specific sequence of actions and interactions between actors and the system.
- A *use case* is a collection of related success and failure scenarios that describe an actor using a system to support a goal.



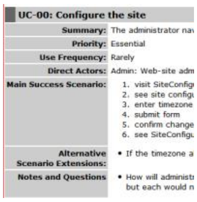
An actor

Action

System

Use Case Elements

- *Name* – Unique identifier that describes an achievable goal.
- *Goal* – Briefly describes what the user intends to achieve with this use case.
- *Summary* – Provides a quick overview.
- *Actor* – anything with behavior.
- *Trigger* states what event gets the use case started.
- *Preconditions* state what must always be true before beginning a scenario in the use case.
- *Steps* – Sequence of actions and interactions between actors and the system.
- *Post-conditions* state what must be true on successful completion of the use case.



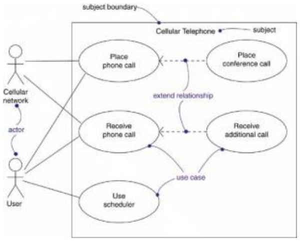
UC-00: Configure the site

Summary:	The administrator nav
Priority:	Essential
Use Frequency:	Rarely
Direct Actors:	Admin: Web-site adm
Main Success Scenario:	1. visit SiteConfig 2. see site configu 3. enter timezone 4. submit form 5. confirm change 6. see SiteConfigu
Alternative Scenario Extensions:	• If the timezone a
Notes and Questions	• How will administr but each would n

http://readyset.tigris.org/nonav/templates/use-case-suite.html

Use Case Diagram [3, 7]

A *use case* diagram is a diagram that shows a set of use cases and actors and their relationships.



USE CASE DIAGRAM

Use cases are <u>text documents</u>, not diagrams, and use-case modeling is primarily an act of <u>writing text</u>, not drawing diagrams.

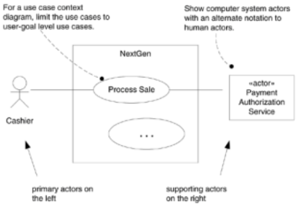
Black-box Use Cases

Black-box use cases do *NOT* describe the <u>internal workings</u> of the system, its components, or design.

Black-box style	Not
The system records the sale.	The system writes the sale to a database. …or (even worse): The system generates a SQL INSERT statement for the sale…

How to Find Use Cases?

1. Choose the *system boundary*.
2. Identify the *primary actors*.
3. Identify *actor's goal*.
4. Define use case that *satisfies user goal* (a verb).


For a use case context diagram, limit the use cases to user-goal level use cases.
Show computer system actors with an alternate notation to human actors.
NextGen
Process Sale
. . .
«actor» Payment Authorization Service
Cashier
primary actors on the left
supporting actors on the right

Use Case Diagram

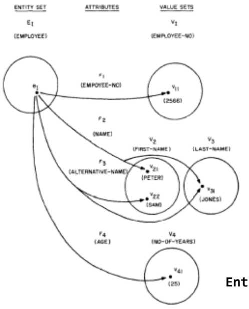
How To Write a *Good* Use Case?

- *Who* is the actor?
- *What* does the actor do?
- How does the actor interact with the system?
  - A SCREEN. How does it <u>look like</u>?
- What are the *inputs* from the actor in each step?
  - NUMBER
  - STRING
  - DATE/TIME
  - TRUE/FALSE
- What will the actor receive for each step (*outputs*)?
  - A SCREEN. <u>What</u> is in the screen (NUMBER, STRING, DATE/TIME, TRUE/FALSE, LIST)? How does it <u>look like</u>?

Why Use Cases?

- *Simple and possible* for domain experts or requirement donors to themselves write (or participate in writing)
- Emphasize the *user goals* and perspective
- The ability to *scale both up and down* in terms of sophistication and formality.

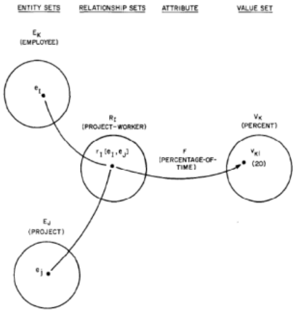
Data Requirements [6]



ATTRIBUTE ENTITY SET AND VALUE SET	f_1 (EMPLOYEE-NO)	f_2 (NAME)		f_3 (ALTERNATIVE-NAME)		f_4 (AGE)
E_1 (EMPLOYEE)	V_1 (EMPLOYEE-NO)	V_2 (FIRST-NAME)	V_3 (LAST-NAME)	V_2 (FIRST-NAME)	V_3 (LAST-NAME)	V_4 (NO-OF-YEARS)
e_1	v_{11} (2566)	v_{21} (PETER)	v_{31} (JONES)	v_{22} (SAM)	v_{31} (JONES)	v_{41} (25)
e_2	v_{12} (3378)	v_{23} (MARY)	v_{32} (CHEN)	v_{24} (BARB)	v_{33} (CHEN)	v_{42} (23)
⋮	⋮	⋮		⋮		⋮

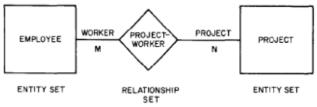
Entity Set, Attributes and Value Sets

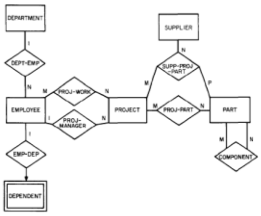
Relationship Set



ROLE	WORKER	PROJECT		
			F (PERCENTAGE-OF-TIME)	RELATIONSHIP ATTRIBUTE
ENTITY SET	E_I (EMPLOYEE)	E_J (PROJECT)	V_K (PERCENTAGE)	VALUE SET
	e_{I1}	e_{J1}	v_{K1} (20)	
	⋮	⋮	⋮	

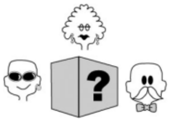
Entity Relationship Diagram





Look and Feel Requirements [1]

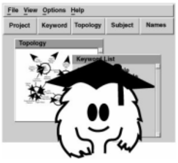
- The product shall comply with *corporate branding standards*.
- The product shall be *attractive to an older audience*.
- The product shall appear *simple to use*.
- The product shall appear *state of the art*.
- The product shall comply with the [Aqua/Vista/Motif/etc.] *guidelines*.
- The product shall conform to the *established look and feel of the organization's products*.



- ○ Be careful not to design the interface, because you do not yet know the complete requirements of the product.
- ○ Designing is the task of the product's designers, once they know the requirements.

Usability and Humanity Requirements

- The product shall be easy to use.
- The product shall be easy to use by members of the public who might not read English.
- The product shall be *easy to learn*.
- The product shall be *easy to use on the first attempt by a member of the public without training*.
- *Ninety percent of a panel* that is representative of the general public shall successfully purchase a ticket from the product within *45 seconds of their first encounter*.



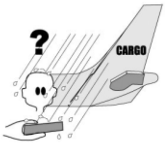
Performance Requirements

- The product shall identify whether an aircraft is hostile or friendly within 0.25 second.
- The product shall have the capacity for 5,000 roads.

- *Speed* to complete a task
- *Accuracy* of the results
- *Safety* to the operator
- *Volumes to be held* by the product
- *Ranges* of allowable values
- *Throughput*, such as the rate of transactions

Operational and Environmental Requirements

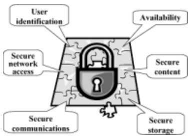
- The product shall be used in and around trucks at night and during rainstorms, snow, and freezing conditions.
- The product shall interface with the thermal mapping database.
- The product shall conserve battery life.



- *The operating environment*
- *The condition of the users* (Are they in the dark, in a hurry, and so on?)
- Partner or *collaborating systems*

Security Requirements

- The product shall ensure that *only authorized users have access* to the [name of] data (or function).
- The product shall deliver data in a manner that prevents further or *second-hand use by unauthorized people*.
- The product shall *retain a journal of all transactions* for the statutory period.



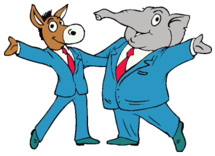
- *Confidentiality*: Data stored by the product is protected from unauthorized access and disclosure.
- *Integrity*: The product's data is the same as the source, or authority, of the data.
- *Availability*: The product's data and functionality are accessible to authorized users and can be produced in a timely manner.

Maintainability and Support Requirements

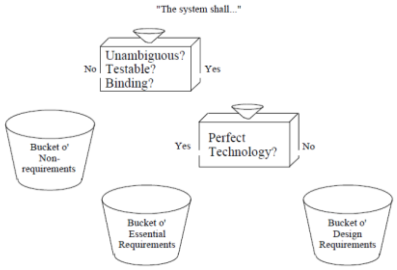
- The product shall be readily *portable to Linux*.
- The product shall be *translated into various foreign languages*. As yet, the languages are unknown.

Cultural, Political and Legal Requirements

- The product shall not use any terms or icons that might possibly *offend anyone* on the planet.
- The product shall use *American spelling.*
- The product shall comply with the Americans with *Disabilities Act*.
- The product shall comply with our *ISO 9001 certification*.



Implications of Perfect Technology



"The system shall..."

Unambiguous?
Testable?
Binding?

No

Yes

Bucket o' Non-requirements

Perfect Technology?

Yes

No

Bucket o' Essential Requirements

Bucket o' Design Requirements

Design Requirements

- Requirements about *speed, cost, and capacity* go into the design bucket
- Requirements about *reliability (MTBF, MTTR)* go into the design bucket
- Requirements about *I/O mechanisms and presentations* go into the design bucket
- Requirements about *computer languages* go into the design bucket
- Requirements about *archiving* go into the design bucket
- Requirements about the *customer's business policy, business process* go into the essential bucket

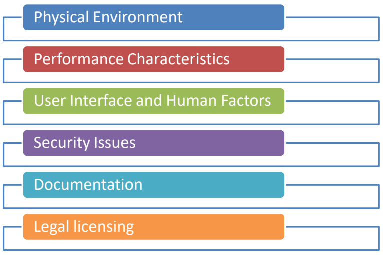
Requirements Specification

- *Users* of the product
- Naming conventions and *definitions*
- *Functional* requirements
  - *Functional and data* requirements
- *Nonfunctional* requirements
  - *Look and feel* requirements
  - *Usability* and *humanity* requirements
  - *Operational* and *environmental* requirements
  - *Security* requirements
- Project *constraints*
- Domain model, UIs, storyboards, state models

1. HTTP://READYSET.TIGRIS.ORG/NONAV/TEMPLATES/SRS.HTML
2. HTTP://WWW.VOLERE.CO.UK/TEMPLATE.HTM

Supplementary Specification [3]

Supplementary specification is primarily for all non-functional requirements, such as performance or licensing. It is also the place to record functional features not expressed (or expressible) as use cases.



Physical Environment

Performance Characteristics

User Interface and Human Factors

Security Issues

Documentation

Legal licensing

Glossary

Glossary defines noteworthy terms, records requirements related to data, such as validation rules, acceptable values, and so forth.

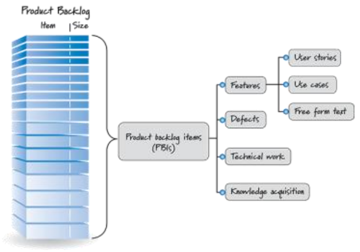
Term	Definition and Information	Format	Validation Rules	Aliases
item	A product or service for sale			
payment authorization	Validation by an external payment authorization service that they will make or guarantee the payment to the seller.			
payment authorization request	A composite of elements electronically sent to an authorization service, usually as a char array. Elements include: store ID, customer account number, amount, and timestamp.			
UPC	Numeric code that identifies a product. Usually symbolized with a bar code placed on products. See www.gs1us.org for details of format and validation.	12-digit code of several subparts.	Digit 12 is a check digit.	Universal Product Code

Business Rules

Business rules dictate how a domain or business may operate.

ID	Rule	Changeability	Source
RULE1	Signature required for credit payments.	Buyer "signature" will continue to be required, but within 2 years most of our customers want signature capture on a digital capture device, and within 5 years we expect there to be demand for support of the new unique digital code "signature" now supported by USA law.	The policy of virtually all credit authorization companies.
RULE2	Tax rules. Sales require added taxes. See government statutes for current details.	High. Tax laws change annually, at all government levels.	law

Product Backlog



Requirements Verification and Validation

- Review and inspection
- Prototypes
- Test cases
- Model validation

Fit Criteria

- The *measurement* of the requirement is the fit criterion.
- It *quantifies* the behavior, the performance, or some other quality of the requirement.
- The *scale of measurement* is the unit you use to test conformance of the product.
- If you *CAN'T* measure a requirement, it is NOT really a requirement.

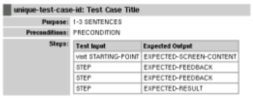
Fit Criteria for Nonfunctional Requirements

- The product shall be *user-friendly*.
- New users shall be able to add a road, change a road, and delete a road within *30 minutes of their first attempt* at using the product.
- The product shall be certified as complying with this year's *corporate branding standards* by the head of marketing.
- *Sixty percent of the target audience* will recognize the product as belonging to the corporation within *five seconds of encountering* it for the *first time*.

SURVEYING

Fit Criteria for Functional Requirements: Test Case

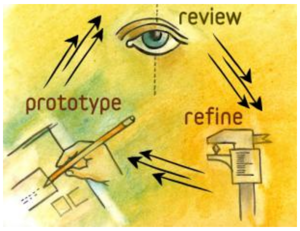
- *unique-test-case-id: Test Case Title*
- *Purpose* – Short sentence or two about the aspect of the system is being tested. If this gets too long, break the test case up or put more information into the feature descriptions.
- *Preconditions* – Assumptions that must be met before the test case can be run.
- *Steps* – Sequence of actions and interactions between actors and the system.
  - Input
  - Expected output
- *Notes and Questions*

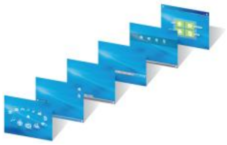

unique-test-case-id: Test Case Title

Purpose:	1-3 SENTENCES
Preconditions:	PRECONDITION
Steps:	

Test Input	Expected Output
visit STARTING-POINT	EXPECTED-SCREEN-CONTENT
STEP	EXPECTED-FEEDBACK
STEP	EXPECTED-FEEDBACK
STEP	EXPECTED-RESULT

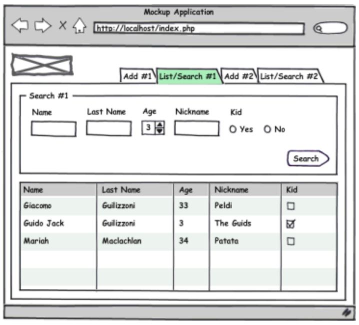
User-Interface Prototyping





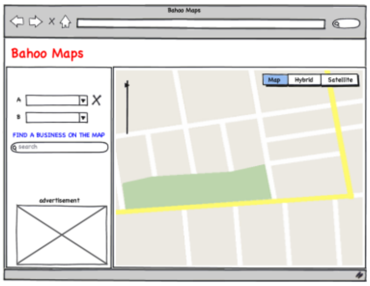
Basic Controls

- Label
- Text Field
- Text Area
- Date Field
- Menus
- Drop Down List
- Radio Buttons
- Check Boxes
- File Uploads
- Slider
- Flip Toggle Switch
- Grid



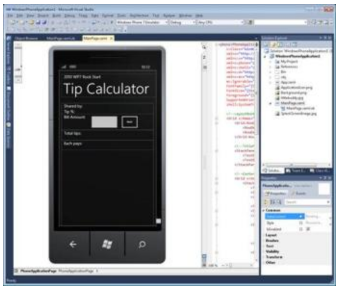
Name	Last Name	Age	Nickname	Kid
Giacomo	Guilizzoni	33	Peldi	☐
Guido Jack	Guilizzoni	3	The Guids	☑
Mariah	Maclachlan	34	Patata	☐

UI Prototyping Using Tool

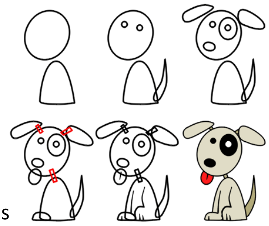


http://balsamiq.com/products/mockups

UI Prototyping Using Code

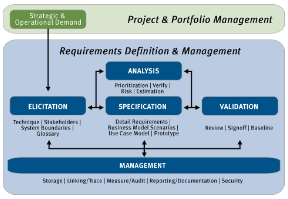


How to Create UIs? [7]



- Searching for templates
- Brainstorming

Requirements Management

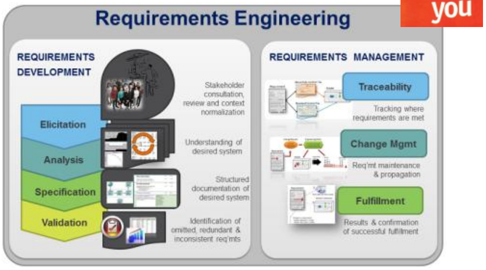

Strategic & Operational Demand
Project & Portfolio Management
Requirements Definition & Management
ANALYSIS
Prioritization | Verify | Risk | Estimation
ELICITATION
Technique | Stakeholders | System Boundaries | Glossary
SPECIFICATION
Detail Requirements | Business Model Scenarios | Use Case Model | Prototype
VALIDATION
Review | Signoff | Baseline
MANAGEMENT
Storage | Linking/Trace | Measure/Audit | Reporting/Documentation | Security

i Dare you

- Tools for modeling requirements
- Tools for managing requirements

Requirements Engineering

i Dare you


Requirements Engineering
REQUIREMENTS DEVELOPMENT
Elicitation
Analysis
Specification
Validation
Stakeholder consultation, review and context normalization
Understanding of desired system
Structured documentation of desired system
Identification of omitted, redundant & inconsistent req'mts
REQUIREMENTS MANAGEMENT
Traceability
Tracking where requirements are met
Change Mgmt
Req'mt maintenance & propagation
Fulfillment
Results & confirmation of successful fulfillment

Thank You for Your Time