



Module 3



Right Product – Client Needs and Software Requirements

Recall Functionality in Topic 2.1

- ▶ What the customer needs
 - ▶ Feature set
 - ▶ Security – safety, exploitability

How can we elicit, express, verify and validate what the customer really needs? See [Module 3 Right Product – Client Needs and Software Requirements](#)



How the customer explained it



How the project leader understood it



How the analyst designed it



How the programmer wrote it



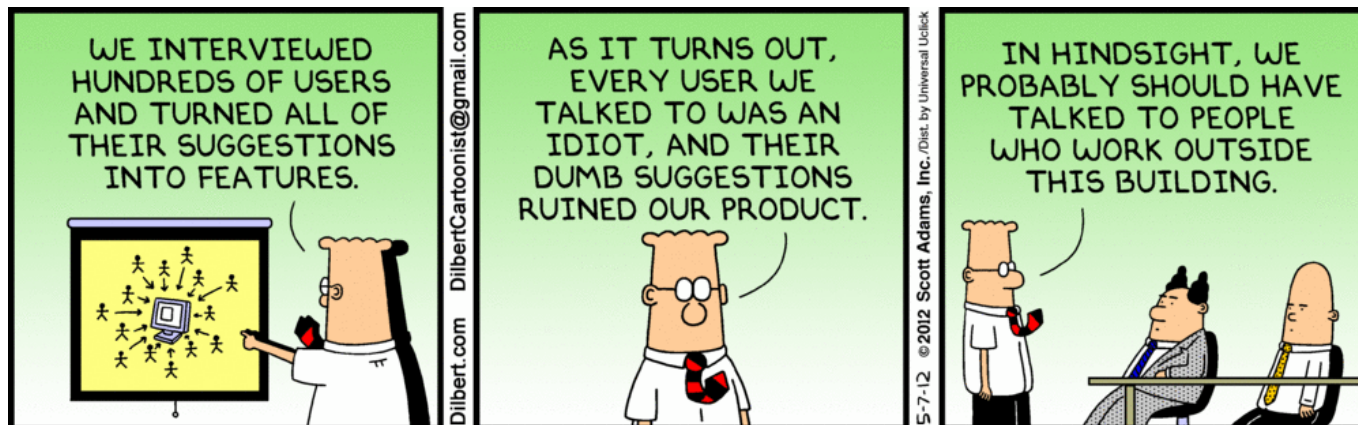
What the customer really needed



Recall Usability in Topic 2.1

- ▶ Human factors, aesthetics, consistency, documentation, responsiveness
- ▶ How effective is the product from the standpoint of the person who must use it?
- ▶ Is it aesthetically acceptable?
- ▶ Is the documentation accurate and complete?

How can we better understand user? See [Module 3 Topic 3.2 User Interaction](#) and [Topic 3.5 Validating Requirements](#)



Module Overview (Week 3 – Week 5)

▶ Topic 3.1 Introduction to Requirements

- ▶ What is a requirement and what are requirement activities?
- ▶ Types of requirements
- ▶ Boundary between requirements and design

▶ Topic 3.2 User Interaction

- ▶ Understand users and elicit requirements from user interactions
- ▶ Techniques that help requirements elicitation – use cases, wireframes, storyboards

▶ Topic 3.3 Writing Requirements

- ▶ Express agile requirements in user stories, acceptance criteria and acceptance tests
- ▶ Prioritize user stories in product backlog and story maps

▶ Topic 3.4 Requirements Quality

- ▶ Summarize the criteria for user stories and clarify ambiguous requirement

▶ Topic 3.5 Validating Requirements


- ▶ Reviews and metrics related to creating the right product



Learning Outcomes

- ▶ Create clear requirements to drive effective software development
- ▶ Visualize client needs using low-fidelity prototypes
- ▶ Maximize the effectiveness of client interactions
- ▶ Adapt to changing product requirements
- ▶ Apply review and testing techniques to validate and verify user stories





Module 3 – Topic 3.1



Introduction to Requirements

Topic Outline

- ▶ What is a requirement?
- ▶ What are requirement activities?
- ▶ Types of requirements
- ▶ Boundary between requirements and design



What is a Requirement?

- Requirements are specific descriptions of your client's needs

How can we get this?



How the customer explained it



How the project leader understood it



How the analyst designed it



How the programmer wrote it



What the customer really needed

Quick Question

Of the following, which would you consider software requirements?

- A. As a hotel manager, I want to be able to check guests in
- B. I'm required to make you sign a contract
- C. Account holders will enter their banking pin into a numeric keypad
- D. Users are male



Wants versus Needs

- ▶ Tease apart the client's wants from their needs

Wants:

“are desired functions that they’d like to see in the product”

Needs:

“are the core functions required in order to address the specific problem that the product is intended to solve”



Requirement Activities

PROJECT MANAGEMENT PHASE

Creating a Process
Setting Standards
Managing Risks
Performing Estimations
Allocating Resources
Making Measurements
Improving Process

SPECIFICATION PHASE

Identifying Needs
Eliciting Requirements
Expressing Requirements
Prioritizing Require'ts
Analysing Require'ts
Managing Require'ts
Formulating Potential Approaches

DESIGN & IMPLEMENTATION PHASE

Designing Architecture
Designing Databases
Designing Interfaces
Creating Executable Code
Integrating Functionality
Documenting

expressing requiriments and testing can be done together

VERIFICATION & VALIDATION PHASE

Developing Test Procedures
Creating Tests
Executing Tests
Reporting Evaluation Results
Reviewing & Auditing
Client Demonstrations
Conducting Retrospectives



Requirement Activities

Focus on:

Eliciting requirements

Expressing requirements

Analysing requirements

&

**Related V&V and
Monitoring activities**

Briefly cover:

Prioritizing requirements

Managing requirements

**(covered in detail in
COMP3120 Managing
Software Development)**



Eliciting versus Gathering Requirements

Requirement gathering
gets you this



How the
customer
explained it



How the
project leader
understood it



How the
analyst
designed it



How the
programmer
wrote it



What the
customer
really needed

Requirement elicitation
gets you this



Types of Requirements

Business Requirements

- ▶ Why the product is needed
 - ▶ Define product needs that provide tangible and quantifiable business value

An example:

I need this product to appeal to clothing designers in India to raise our revenue by \$5,000,000 per year

An example:

FactFinding organization needs this product to support the International Travel Watchdogs objective to ensure air passengers can openly compare airlines



Business Rules

► Constrain how the product functions

Privacy policy:

Data must be stored securely and not shared with non essential personnel

Brand uniformity:

The products to be developed must be visually consistent with other products owned by the client

Government regulation:

Maintain transaction data for a specific period of time



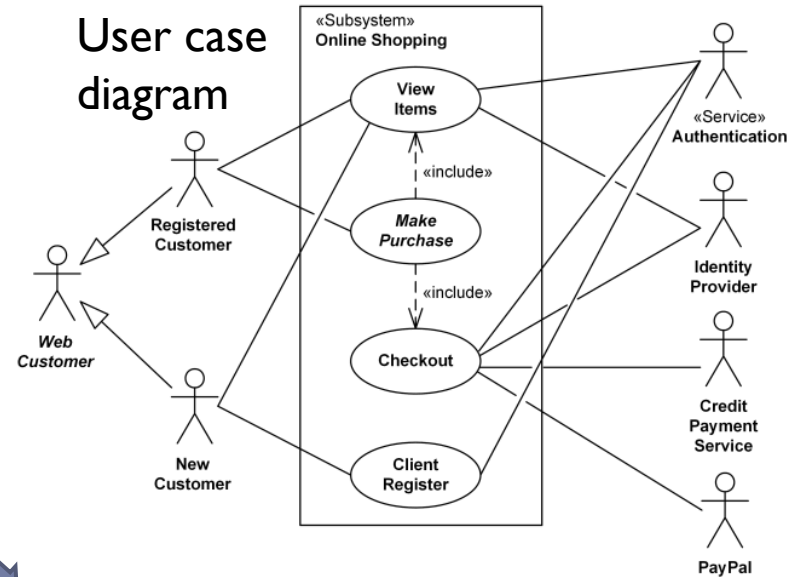
the first functional requirement
related to the functionality.

User Requirements

user requirements let us focus on the
perspective of users.

- Outline exactly what tasks the **end users** can do with the product

Typically, client needs come in this form: “Customers should be able to make purchases within the system. I think that's it.”



Elicit user requirements and develop them into specific, actionable developer tasks

User story:

As a new customer, I want to register an online shopping account, so that I can make an online purchase

Quick Question

Which of these represents the basic form of a user story?

- A. As a __role__, I want to __function__
- B. As a __role__, I want to __function__, since __benefits__
- C. I want to __function__ so that __benefit__
- D. As a __role__, I want to __function__, so that __benefits__



Functional Requirements

- ▶ Describe a behaviour that the **product** should do or support
 - ▶ Inputs → a description of the behaviour → outputs

Consider a mobile Point-of-Sale product

Developer Perspective:
Input → Behaviour →
Output

User Requirements:
As a customer, I want to pay
using a credit card, so that I
can complete a purchase

User perspective:
Who, what, why

Functional requirements:

- A user must be able to insert a Visa or Master card into a credit card reader
- The system must read credit card data from the credit card reader once a card is inserted
- The system must print a transaction receipt once the payment is approved



Non-Functional Requirements

- ▶ How well a product must perform
 - ▶ Address the quality attributes of the product (recall FURPS)

Security:

The system must store user data using AES-256 encryption standard

Reliability/Dependability:

After a system reboot, the full system functionality must be restored within 5 minutes

Maintainability:

The database must be replaceable with any commercial product supporting standard SQL queries

Usability:

80% of first-time users must be able to register an user account within 5 minutes of starting to use the system

Help messages must be displayed in the local language according to the user's locale

Performance:

When a credit card is inserted in the chip reader, the system must detect it within 2 seconds

The system must be able to handle up to 1,000 transactions per second



Additional Requirement Types

Recall plus in
FURPS+

- ▶ External interface – describe the way in which a product interact with other entities outside the product
 - ▶ Media, protocols, formats, levels of a compatibility
- ▶ Physical setting – describe how the product should be designed around its physical environment
 - ▶ GPS tracker in Sahara versus Alaska
- ▶ Development constraints – is your development team capable of creating the product you designed?
 - ▶ Technology, conventions, documentation, process, devices, platforms, resources the development constraints should be done quit early.



Types of Requirements – Sum Up

- ▶ The product as a whole
 - ▶ Business requirements – why the product should exist
 - ▶ Business rules – constrain how the product functions
- ▶ Core requirements
 - ▶ User requirements – what tasks the **end users** can do with the product
 - ▶ Functional requirements – a behaviour that the **product** should do or support
 - ▶ Non-functional requirements - How well a product must perform
- ▶ Additional context
 - ▶ External interface – how the product interact with other entities outside the product
 - ▶ Physical setting - the physical environment in which the product must operate
 - ▶ Development constraints – technology, resources, devices, ...

Also check **Reading 3.1 Business Requirements vs. Functional Requirements**



Quick Question

Airplanes fly from the Canberra International Airport, to the Changi International Airport in Singapore daily. These airplanes have schedules to which they must adhere, and their range is limited by the amount of fuel that they can carry.

Which of the following are non-functional requirements of airplanes flying from Canberra to Singapore?

- A. Must consume jet fuel is the input for the airplane
- B. Flight schedule must not be delayed option B is the non-functionality for not only the airplane
- C. Must be able to travel between Canberra and Singapore on a single tank of fuel
- D. A full load of passengers must be evacuated in no more than 90 seconds



Quick Question

Which of the following would be considered an external interface requirement?

- A. The product must be able to communicate to a customer database and an ad server
- B. The product must be able to remain on battery power for six hours
- C. The product must be able to withstand impact from a drop of no less than fifty meters
- D. The product must withstand full sunlight for ten hours per day



Quick Question

As a user, I want to enter my personal information, so that the product can create my account.

Which type of requirement does this user story represent?
(Check two that apply)

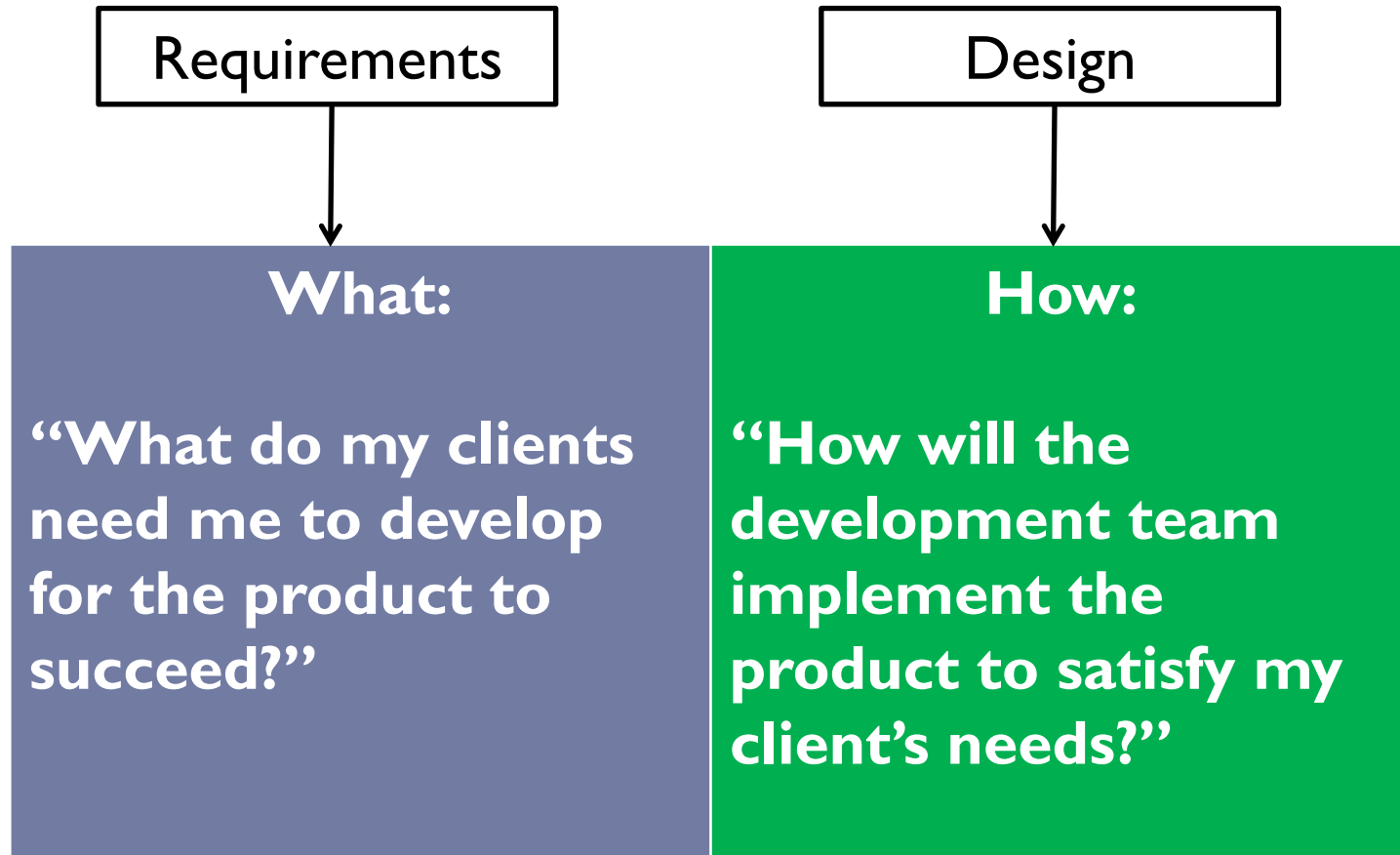
- ▶ Business rule
- ▶ Business requirement
- ▶ User requirement
- ▶ Development constraint
- ▶ External interface
- ▶ Physical setting
- ▶ Functional requirement
- ▶ Non-functional requirement



Boundary between Requirements and Design

The Boundary is Fuzzy!

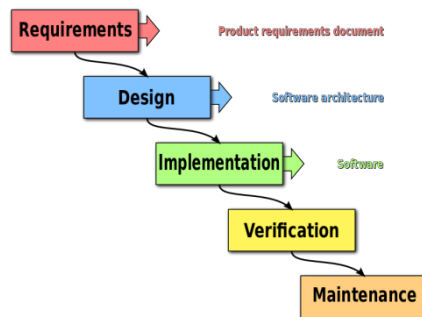
The Boundary is Fuzzy



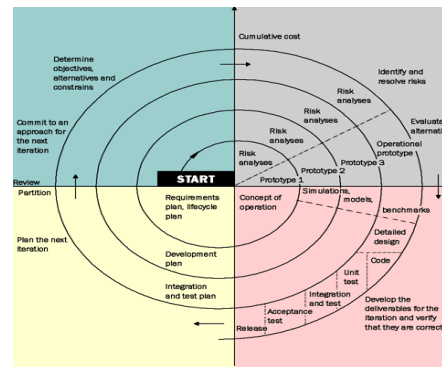
Recall Process Models

In which software process model would requirements and design be two completely separate phases?

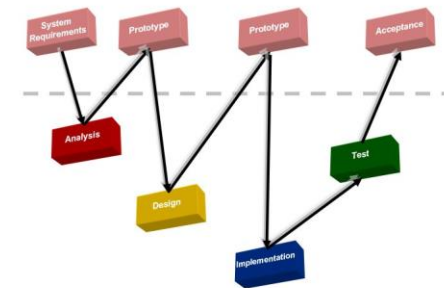
A. Waterfall



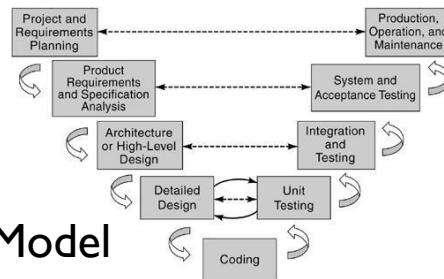
B. Spiral



C. Sawtooth



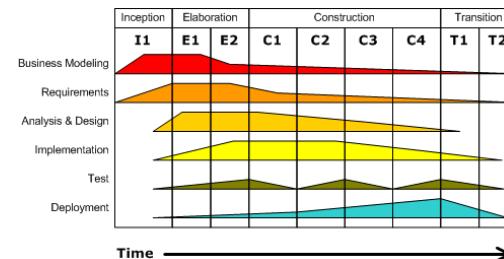
D. V-Model



The V-Shaped Software Development Life Cycle Model

Iterative Development

Business value is delivered incrementally in time-boxed cross-discipline iterations.



E. Unified Process

Recall Linear Models

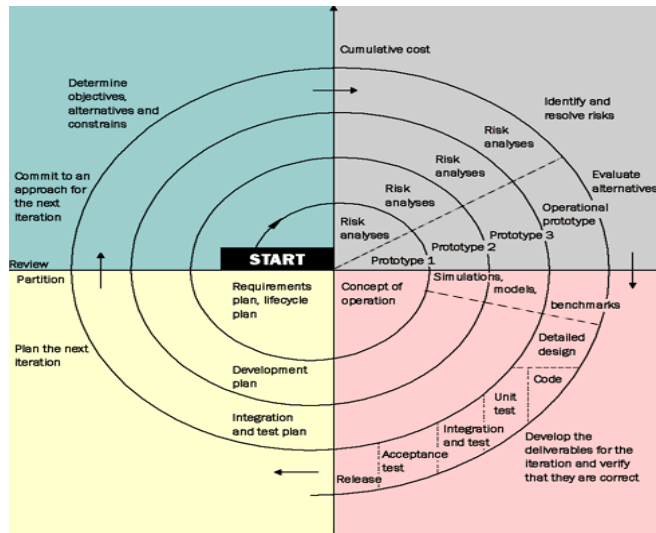
- ▶ They include phases, which happen **sequentially**, one after another
- ▶ They originate in the manufacturing and construction industries
 - ▶ The emphasis is on getting the requirements right, upfront, and **not changing** them afterwards
 - ▶ They did not favour having developers try small programming experiments to quickly test out their ideas



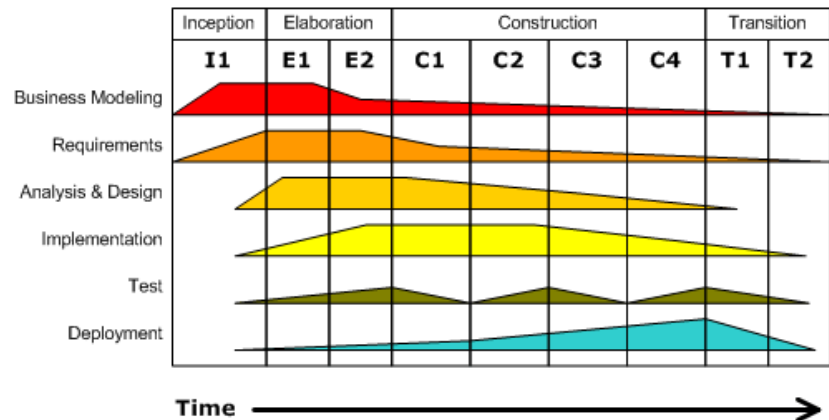
Reality check: developing a software product is a creative endeavour, which necessitates experimentation and constant rework

Recall Iterative and Parallel Process Models

- Form requirements with elements of design in mind



Iterative Development
Business value is delivered incrementally in time-boxed cross-discipline iterations.



But Still the Emphasis is Different!

- ▶ Blurring the boundary is fine, but avoid imposing certain design solutions upon your product too early

In the following examples, does the design sneak into requirements in bad ways?

The product shall be written in the Java programming language

The user clicks okay to submit a request



Avoid Too Much Design in Requirements

- ▶ As you develop requirements, keep these questions in mind
 - ▶ Is a solution just a possible solution?
 - ▶ Is the solution the only one possible?
 - ▶ Is the solution addressing the wrong problem?
 - ▶ Is the solution just to attract developer interest?
 - ▶ Is the client more solution focused?

