47006- ANÁLISE E MODELAÇÃO DE SISTEMAS

Práticas da Engenharia de Requisistos

Ilídio Oliveira

V2022/05/03

universidade de aveiro
departamento de eletrónica, telecomunicações e informática
deti

Objetivos de aprendizagem

- Distinguir especificação de requisitos centrados no utilizador e no produto
- Descrever técnicas de documentação dos requisitos
- Compreender a relação entre requisitos e casos de utilizaçãoldentificar as disciplinas e actividades relacionadas com os requisitos no OpenUP
- Explicar o significado de "requisitos evolutivos“
- Descrever os tipos de informação que o Analista recolhe nas actividades de engenharia de requisitos
- Explicar como implementar actividades de rastreio de requisitos

Requisitos *vs* cenários

Wiegers: duas abordagens principais?

Usage-centric or product-centric?

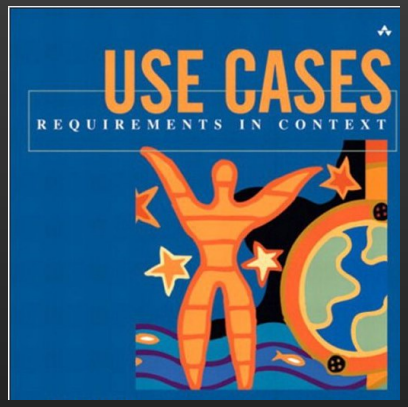
Requirements elicitation typically takes either a usage-centric or a product-centric approach, although other strategies also are possible. The usage-centric strategy emphasizes understanding and exploring user goals to derive the necessary system functionality. The product-centric approach focuses on defining features that you expect will lead to marketplace or business success. A risk with product-centric strategies is that you might implement features that don't get used much, even if they seemed like a good idea at the time. We recommend understanding business objectives and user goals first, then using that insight to determine the appropriate product features and characteristics.

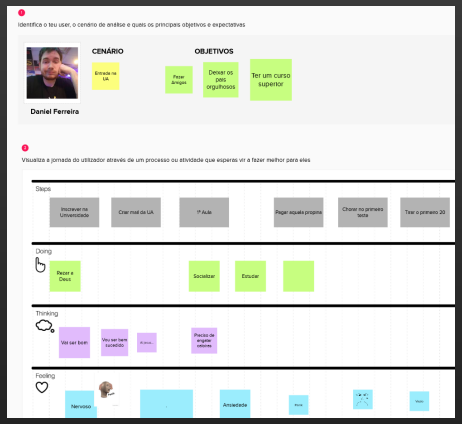
Especificação centrada no produto: “The system shall....”

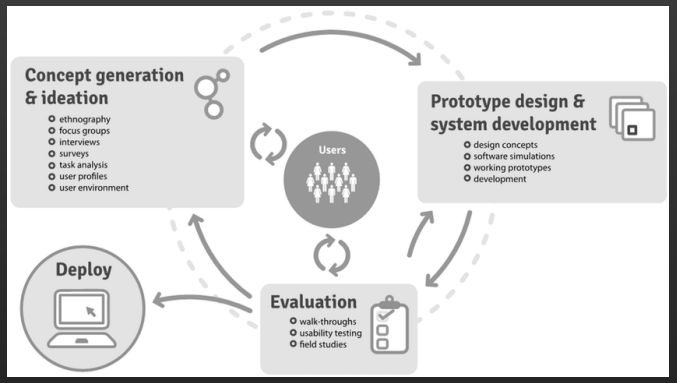
#	Requisito
RF.1	O sistema deve permitir a um profissional criar um novo pedido de adesão , em auto-serviço, na web.
RF.2	O sistema deve enviar credenciais temporárias para os pedidos de adesão e enviá-las, por email, aos solicitantes.
RF.3	O sistema deve permitir a pesquisa de cheques-dentista (emitidos) por número de utente do SNS.
RNF.1	As pesquisas de cheques-dentista têm de retornar resultados em <5 segundos ou um evento de tempo expirado.
...	

“Product centric”: pensamento focado em caraterísticas técnicas do que seria um “bom” produto.

Foco na utilização → captar os requisitos ao contar **histórias** (narrativas)







Use cases
(Suportado na UML)

Jornada do utilizador (CJM)

Desenho centrado no utilizador (UCD)

...e ainda: *User Stories*, *Design Thinking*,...

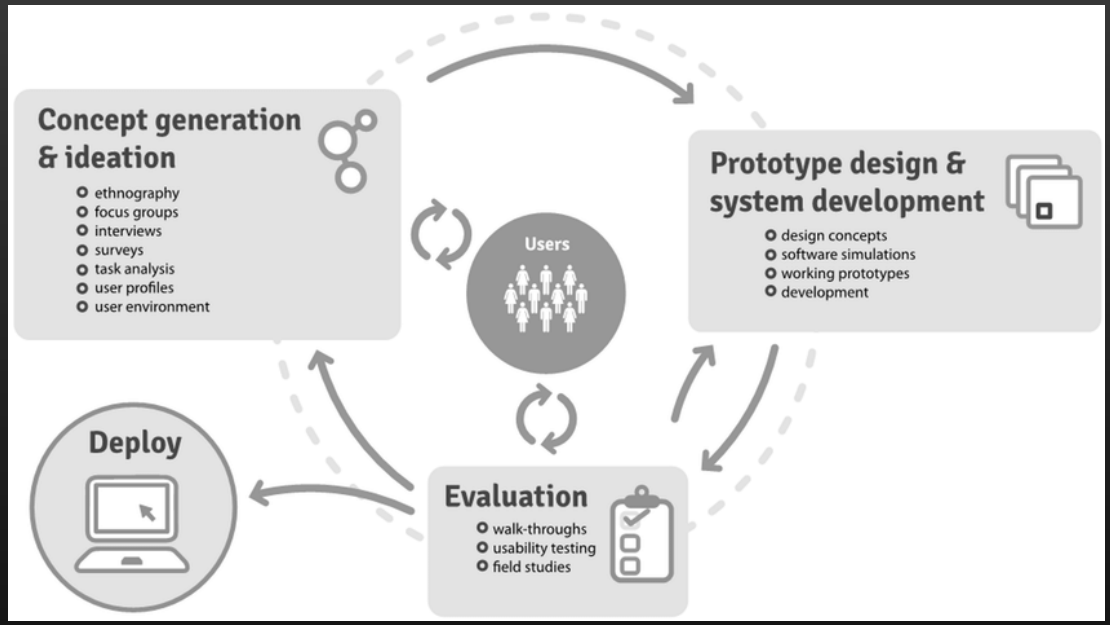
UCD: User Centered Design

Ponto de partida: *personas* & elaboração de cenários

Os requisitos são derivados dos cenários

Evolui através de ciclos de prototipagem e validação

Fortemente orientado para a satisfação do utilizador



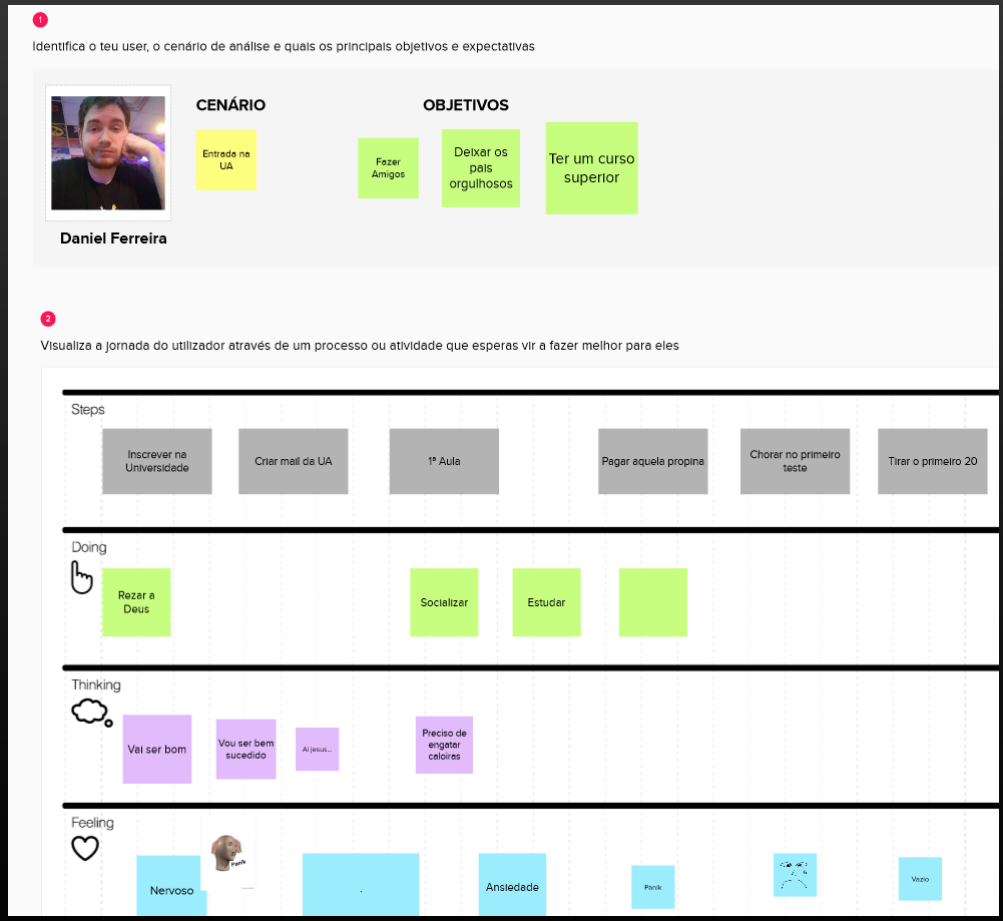
CJM: Custommer Journey Maps

Foco na experiência global do utilizador

Procurar pontos fracos (“pain points”) e oportunidades

Instrumento visual: jornada do utilizador

Muito útil para gerar discussão e descoberta, em equipas pluridisciplinares



CaU: contam histórias que mostram os requisitos funcionais em contexto

Caso Utiliz.:	Consultar informação clínica (referenciação)
Motivação:	O Médico Dentista (MD) acede ao sistema para consulta informação clínica inserida pelo médico assistente.
Sequência típica:	1. Inicia-se quando o Médico Dentista recebe um Utente portador de Cheque-Dentista para consulta. 2. O MD acede à opção de pesquisa na sua página de entrada. 3. O sistema apresenta o formulário de pesquisa, com as hipóteses de pesquisa por nr utente, nome, género e data de Nascimento. 4. O MD insere o número de utente do SNS e confirma a pesquisa. 5. O sistema pesquisa os cheques-dentista existentes para aquele utente e apresenta uma listagem ordenada do mais recente para o mais antigo. 6. O MD seleciona uma entrada na lista para abrir a informação de detalhe. 7. O sistema apresenta para esse CD o cabeçalho com a identificação do cheque e utente, e uma seção com a informação clínica disponível.
Sequências	...

RF.3	O sistema deve permitir a pesquisa de cheques-dentista (emitidos) por número de utente do SNS.

SRS – *Software Requirements Specification*

Documentação de requisitos: *SRS report*

Um relatório especial para documentar os requisitos.

Modelos candidatos:

- "System Proposal" by A.Dennis
- SRS by K.Wiegers
- "Software requirements specification" by ISO/IEC/IEEE 29148:2018 —International Standard — Systems and software engineering — Life cycle processes — Requirements engineering → international standard describing requirements engineering processes for development of software and hardware products.

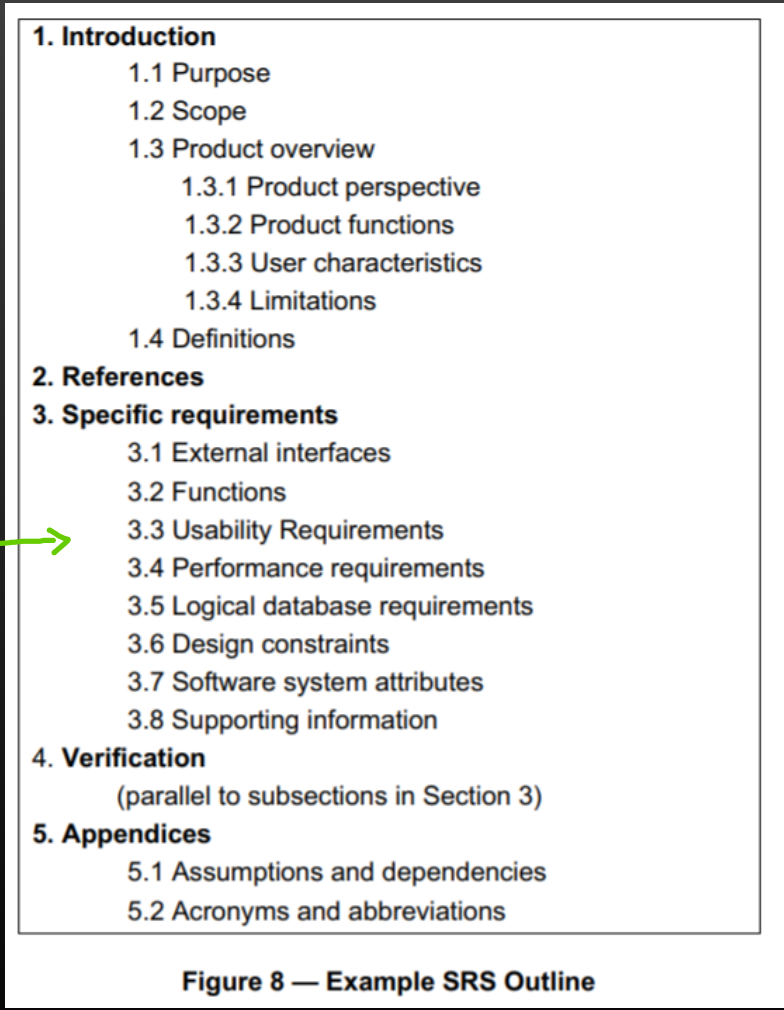


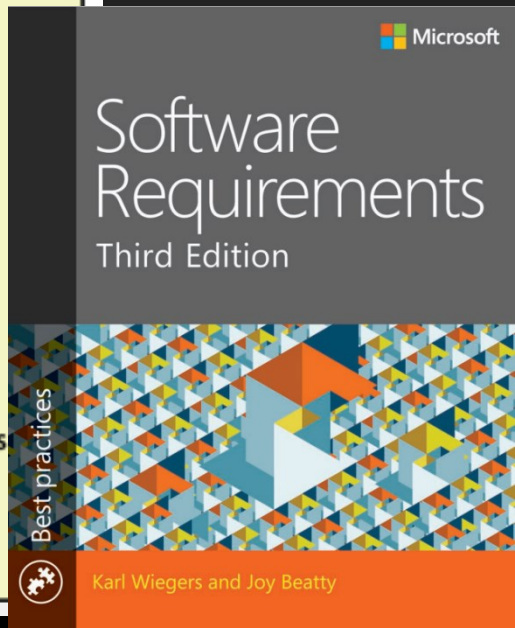
Figure 8 — Example SRS Outline

Wieger's proposal

1. **Introduction**
   - 1.1 Purpose
   - 1.2 Document conventions
   - 1.3 Project scope
   - 1.4 References
2. **Overall description**
   - 2.1 Product perspective
   - 2.2 User classes and characteristics
   - 2.3 Operating environment
   - 2.4 Design and implementation constraints
   - 2.5 Assumptions and dependencies
3. **System features**
   - 3.x System feature X
     - 3.x.1 Description
     - 3.x.2 Functional requirements
4. **Data requirements**
   - 4.1 Logical data model
   - 4.2 Data dictionary
   - 4.3 Reports
   - 4.4 Data acquisition, integrity, retention, and disposal
5. **External interface requirements**
   - 5.1 User interfaces
   - 5.2 Software interfaces
   - 5.3 Hardware interfaces
   - 5.4 Communications interfaces
6. **Quality attributes**
   - 6.1 Usability
   - 6.2 Performance
   - 6.3 Security
   - 6.4 Safety
   - 6.x [others]
7. **Internationalization and localization requirements**
8. **Other requirements**

Appendix A: Glossary

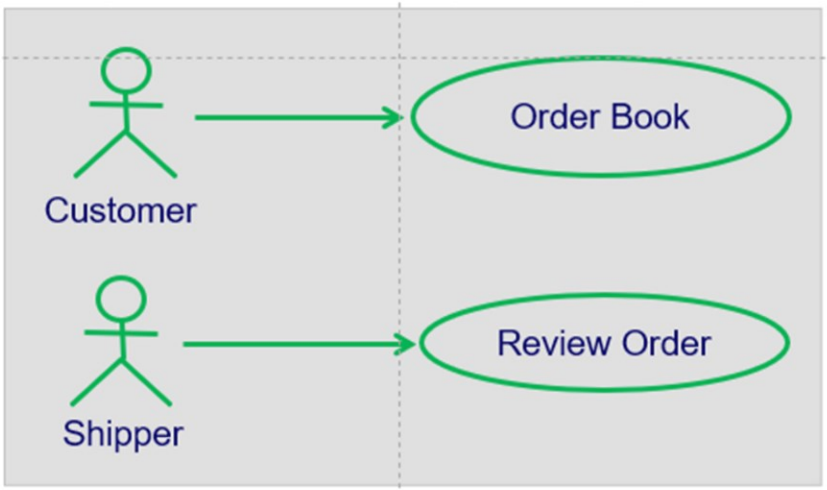
Appendix B: Analysis models



Os requisitos na OpenUP

Use Case Driven Development

- This practice describes how to capture requirements with a combination of use cases and system-wide requirements, and then drive development and testing from those use cases.



Artefactos relacionados com os requisitos no OpenUP?

Use-Case Model

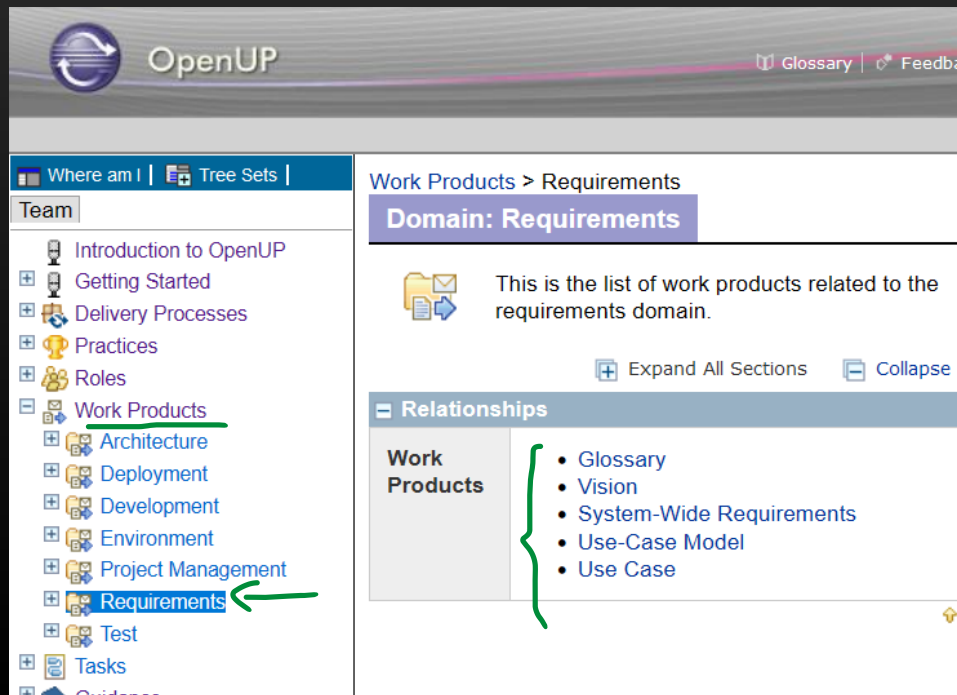
Um conjunto de cenários típicos de utilização de um sistema. São sobretudo para requisitos funcionais (comportamento).

System-wide requirements

Um especificação suplementar com aquilo que não é adequado colocar nos casos de utilização. Este artefacto é sobretudo para todos os requisitos não-funcionais. É também o lugar para registar características funcionais não expressas (ou susceptíveis de ser expressas) como casos de utilização (por exemplo, uma geração de relatórios).

Glossary

O Glossário define termos dignos de nota. Abrange também o conceito de dicionário de dados, que regista requisitos relacionados com os dados, tais como regras de validação, valores aceitáveis, e assim por diante.



Examples of SRS reports

→ SRS example fom Wiegers

→ System Proposal from Dennis (chap. 3)

* Not only requirements

→ OpenUP: use cases + system-wide requirements

Types of information collected in requirements development

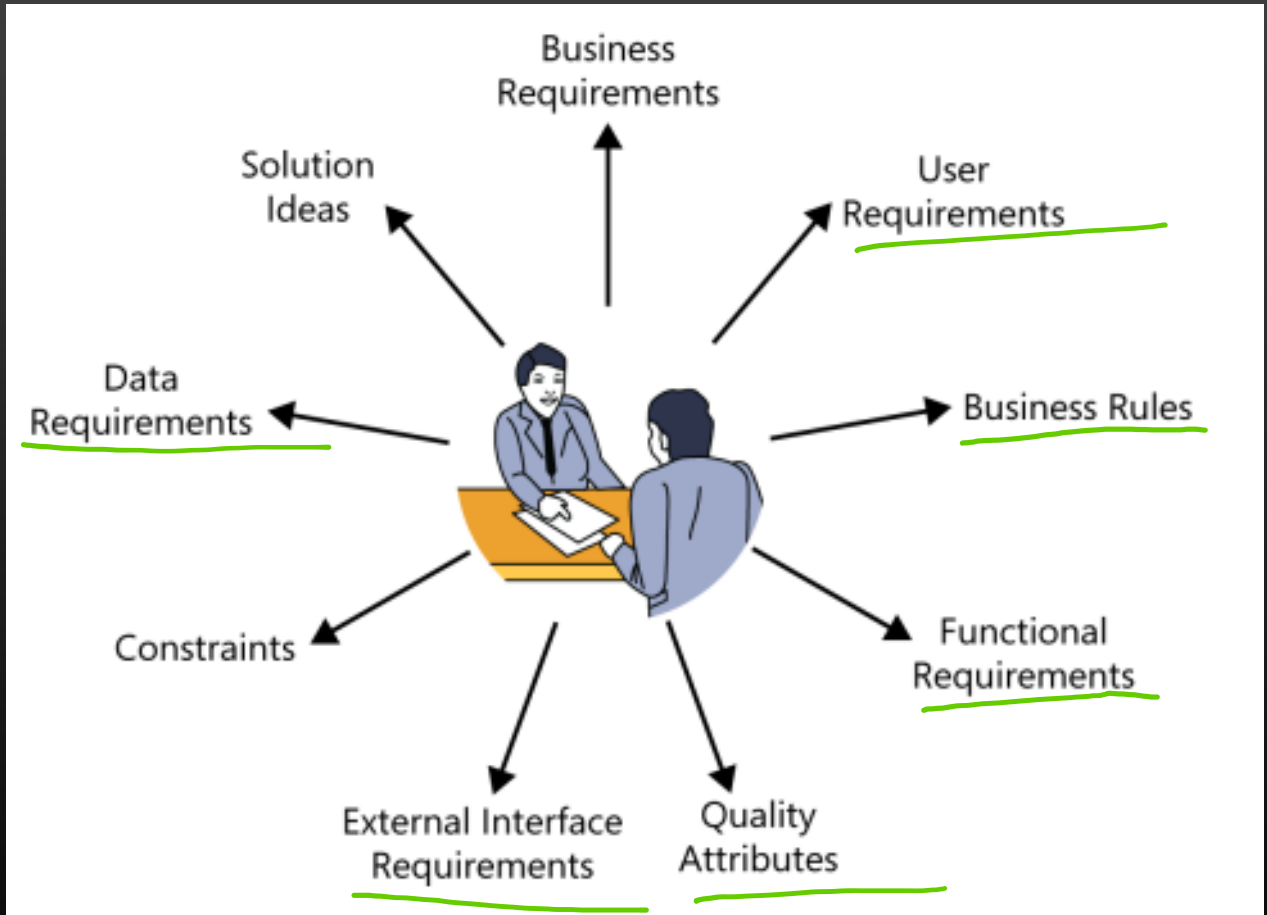


FIGURE 7-7 Classifying customer input.

TABLE 1-1 Some types of requirements information

Term	Definition
Business requirement	A high-level business objective of the organization that builds a product or of a customer who procures it.
Business rule	A policy, guideline, standard, or regulation that defines or constrains some aspect of the business. Not a software requirement in itself, but the origin of several types of software requirements.
Constraint	A restriction that is imposed on the choices available to the developer for the design and construction of a product.
External interface requirement	A description of a connection between a software system and a user, another software system, or a hardware device.
Feature	One or more logically related system capabilities that provide value to a user and are described by a set of functional requirements.
Functional requirement	A description of a behavior that a system will exhibit under specific conditions.
Nonfunctional requirement	A description of a property or characteristic that a system must exhibit or a constraint that it must respect.
Quality attribute	A kind of nonfunctional requirement that describes a service or performance characteristic of a product.
System requirement	A top-level requirement for a product that contains multiple subsystems, which could be all software or software and hardware.
User requirement	A goal or task that specific classes of users must be able to perform with a system, or a desired product attribute.

Abordagem para Análise de Sistemas?

https://youtu.be/p5gDbf0je8k

Domínio:

- Novos processos de trabalho
- Modelo dos conceitos do domínio

Requisitos:

Ponto de partida: use cases

...com a flexibilidade das *user stories/use case slices*

Requisitos evolutivos

Requisitos podem ser evolutivos?

Plan-driven, "waterfall" approach

Tentativa de definir e estabilizar os requisitos na primeira fase de um projeto (antes de qualquer programação)

Requisitos definidos à cabeça para o conjunto do sistema

CONTROLO
Estabilizar os requisitos à cabeça como estratégia para ↓ risco

Agile, evolutionary approach

Aceitar e preparar-se para pedidos/requisitos inevitavelmente mutáveis e/ou pouco claros dos stakeholders.

Com o suporte de algum tipo de abordagem sistemática para encontrar, documentar, organizar e acompanhar a evolução dos requisitos de um sistema.

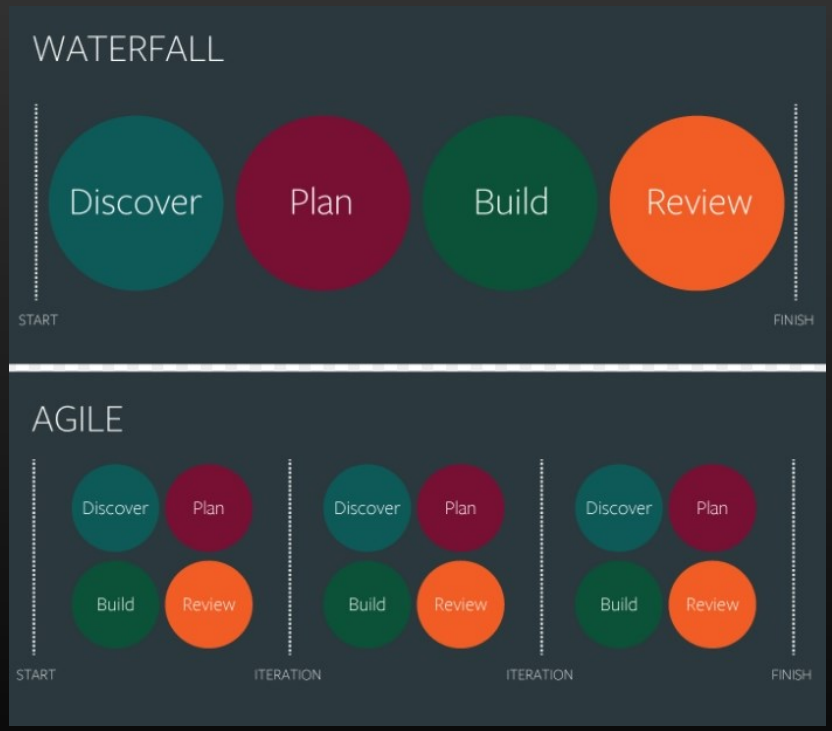
ADAPTAÇÃO
Prever formas de acomodar a mudança como estratégia para ↓ risco

Alto nível, em tempo útil; detalhado quando próximo da construção

Wiegers:

Não se tem de escrever o SRS para o produto inteiro antes de iniciar o desenvolvimento;

necessário definir os requisitos para cada incremento antes de o construir.



A OpenUP propõe requisitos evolutivos?

Practices > Technical Practices > Use Case Driven Development > Tasks > Identify and Outline Requirements

Task: Identify and Outline Requirements

This task describes how to identify and outline the requirements for the system so that the scope of work can be determined.

Disciplines: Requirements

Expand All Sections Collapse All Sections

Purpose

The purpose of this task is to identify and capture functional and non-functional requirements for the system. These requirements form the basis of communication and agreement between the stakeholders and the development team on what the system must do to satisfy stakeholder needs. The goal is to understand the requirements at a high-level so that the initial scope of work can be determined. Further analysis will be performed to detail these requirements prior to implementation.

Práticas de eng.a de requisitos (fonte: K. Wiegers)

Requirements engineering activities

Discover/develop

Requirements gathering and elicitation

Critical aspect in the overall lifecycle

Manage

Document, track and evolve

Apply a change process (assess the risk,...)

The approach to requirements development should be adapted to the kind of project in hands.

E.g.: how formal should be the requirements change request?

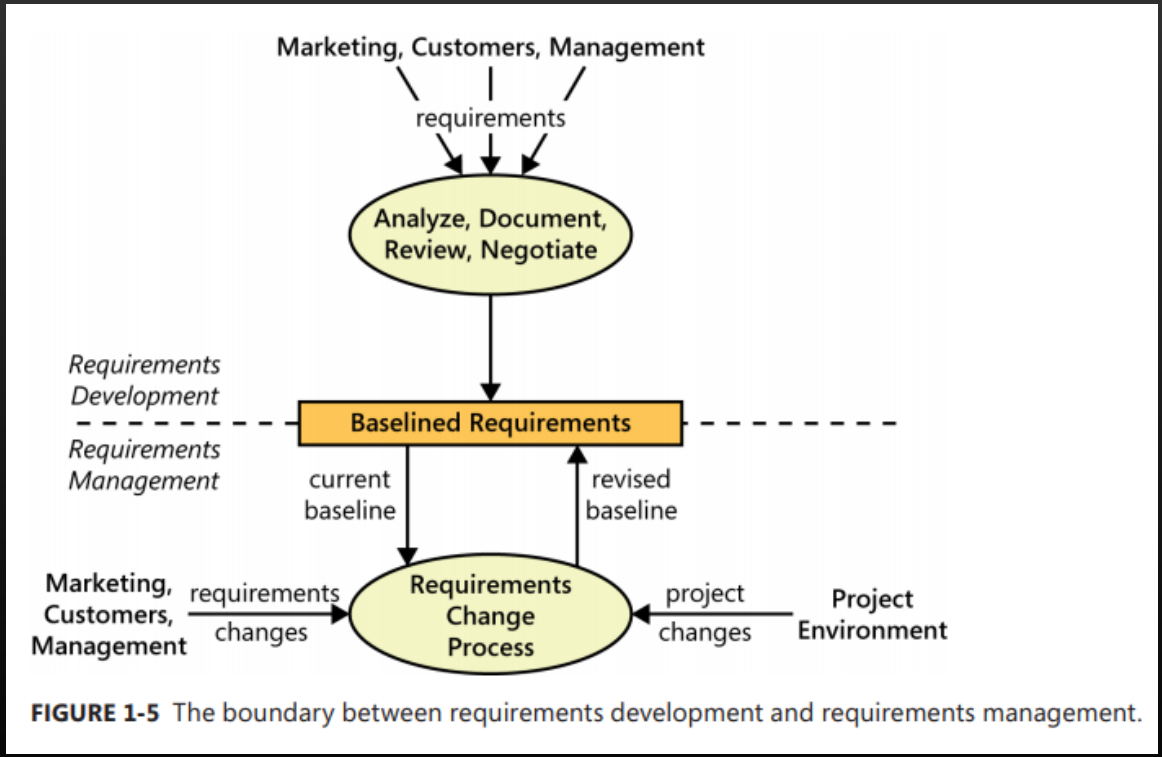


FIGURE 1-5 The boundary between requirements development and requirements management.

Requirements development activities

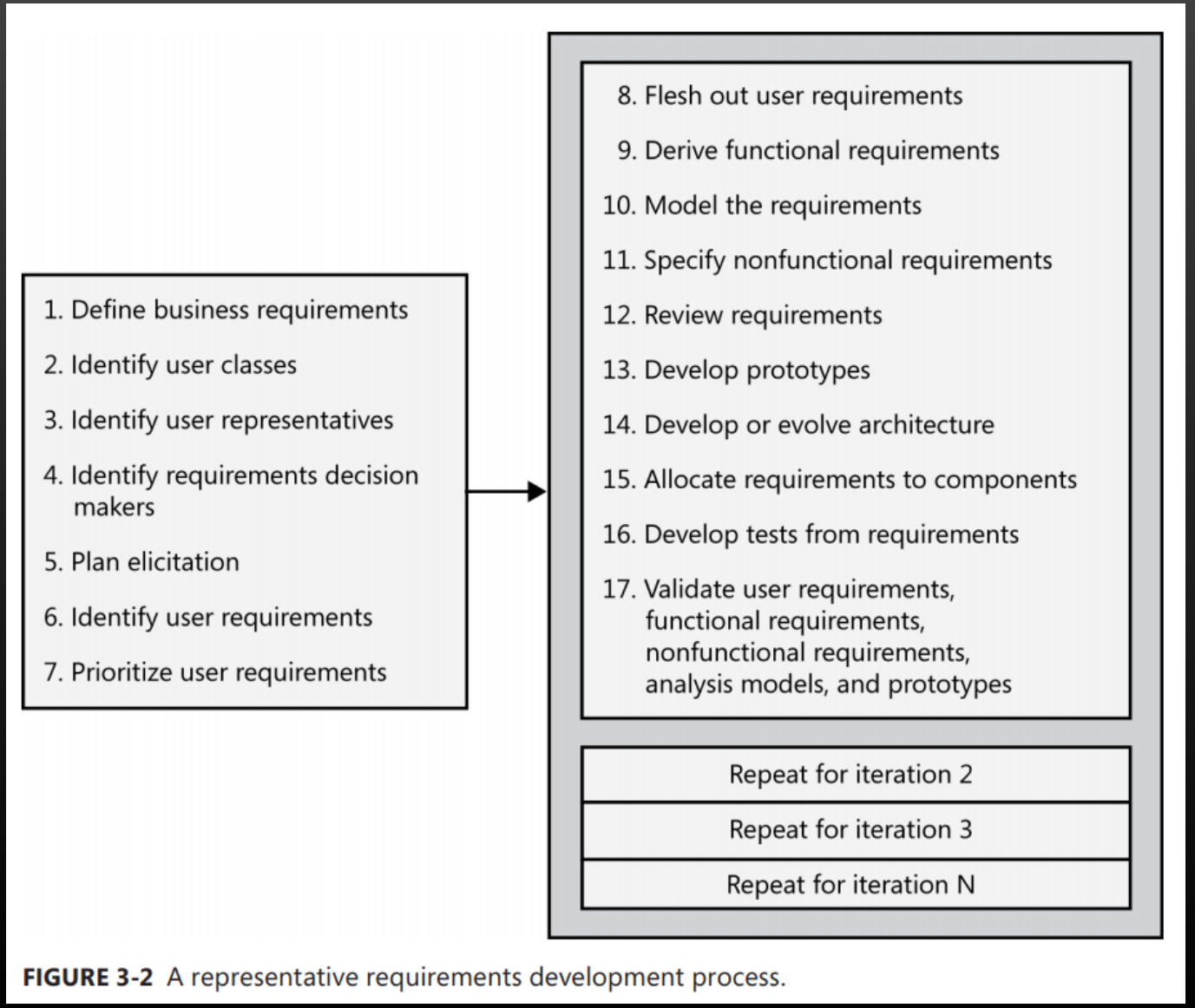


FIGURE 3-2 A representative requirements development process.

TABLE 3-1 Requirements engineering good practices

Elicitation	Analysis	Specification	Validation
■ Define vision and scope	■ Model the application environment	■ Adopt requirement document templates	■ Review the requirements
■ Identify user classes	■ Create prototypes	■ Identify requirement origins	■ Test the requirements
■ Select product champions	■ Analyze feasibility	■ Uniquely label each requirement	■ Define acceptance criteria
■ Conduct focus groups	■ Prioritize requirements	■ Record business rules	■ Simulate the requirements
■ Identify user requirements	■ Create a data dictionary	■ Specify nonfunctional requirements	
■ Identify system events and responses	■ Model the requirements		
■ Hold elicitation interviews	■ Analyze interfaces		
■ Hold facilitated elicitation workshops	■ Allocate requirements to subsystems		
■ Observe users performing their jobs			
■ Distribute questionnaires			
■ Perform document analysis			
■ Examine problem reports			
■ Reuse existing requirements			

Requirements management	Knowledge	Project management
■ Establish a change control process	■ Train business analysts	■ Select an appropriate life cycle
■ Perform change impact analysis	■ Educate stakeholders about requirements	■ Plan requirements approach
■ Establish baselines and control versions of requirements sets	■ Educate developers about application domain	■ Estimate requirements effort
■ Maintain change history	■ Define a requirements engineering process	■ Base plans on requirements
■ Track requirements status	■ Create a glossary	■ Identify requirements decision makers
■ Track requirements issues		■ Renegotiate commitments
■ Maintain a requirements traceability matrix		■ Manage requirements risks
■ Use a requirements management tool		■ Track requirements effort
		■ Review past lessons learned

Prototyping as a requirements validation practice

TABLE 15-1 Typical applications of software prototypes

	Throwaway	Evolutionary
Mock-up	■ Clarify and refine user and functional requirements. ■ Identify missing functionality. ■ Explore user interface approaches.	■ Implement core user requirements. ■ Implement additional user requirements based on priority. ■ Implement and refine websites. ■ Adapt system to rapidly changing business needs.
Proof of concept	■ Demonstrate technical feasibility. ■ Evaluate performance. ■ Acquire knowledge to improve estimates for construction.	■ Implement and grow core multi-tier functionality and communication layers. ■ Implement and optimize core algorithms. ■ Test and tune performance.

Evolving requirements: life after the baseline

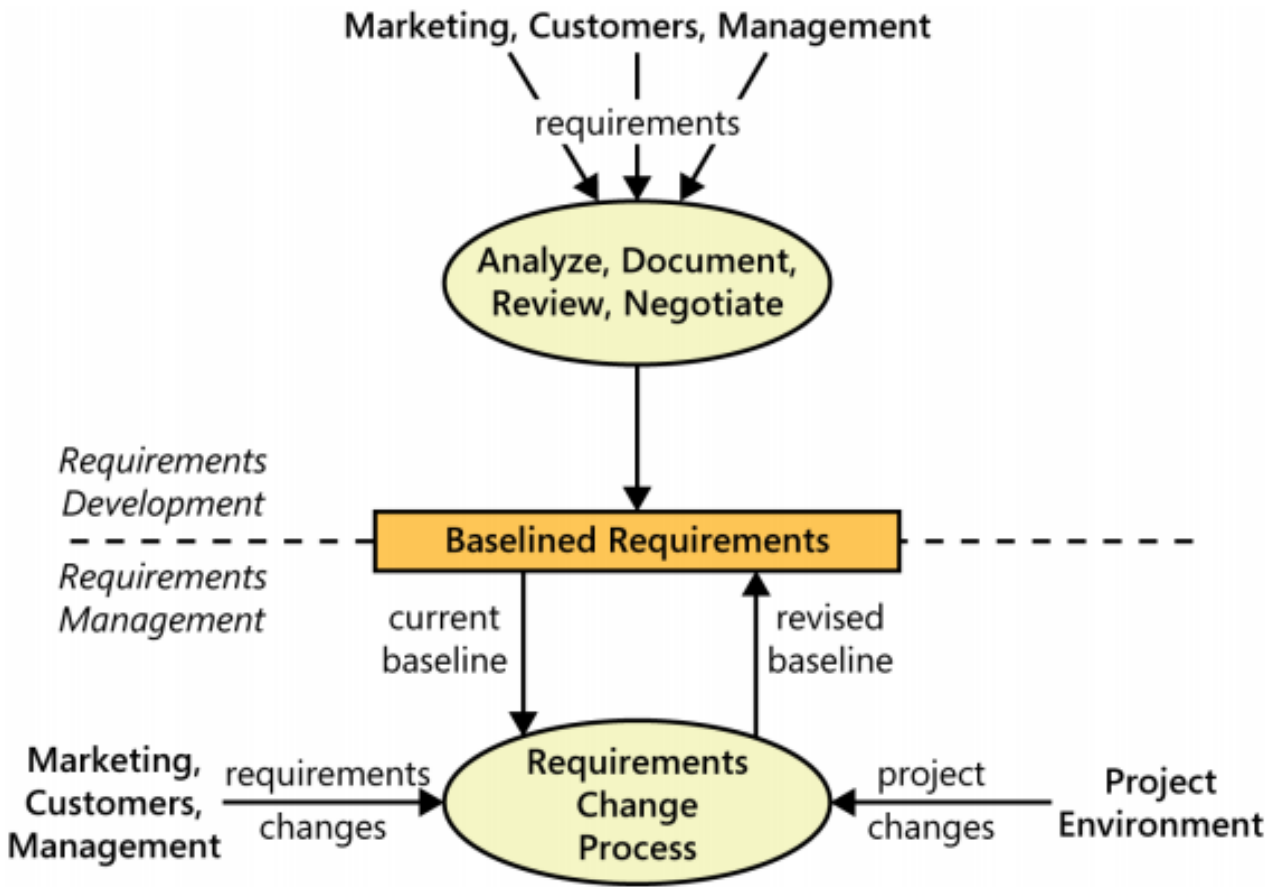


FIGURE 1-5 The boundary between requirements development and requirements management.

TABLE 3-1 Requirements engineering good practices

Elicitation	Analysis	Specification	Validation
■ Define vision and scope ■ Identify user classes ■ Select product champions ■ Conduct focus groups ■ Identify user requirements ■ Identify system events and responses ■ Hold elicitation interviews ■ Hold facilitated elicitation workshops ■ Observe users performing their jobs ■ Distribute questionnaires ■ Perform document analysis ■ Examine problem reports ■ Reuse existing requirements	■ Model the application environment ■ Create prototypes ■ Analyze feasibility ■ Prioritize requirements ■ Create a data dictionary ■ Model the requirements ■ Analyze interfaces ■ Allocate requirements to subsystems	■ Adopt requirement document templates ■ Identify requirement origins ■ Uniquely label each requirement ■ Record business rules ■ Specify nonfunctional requirements	■ Review the requirements ■ Test the requirements ■ Define acceptance criteria ■ Simulate the requirements

Requirements management	Knowledge	Project management
■ Establish a change control process ■ Perform change impact analysis ■ Establish baselines and control versions of requirements sets ■ Maintain change history ■ Track requirements status ■ Track requirements issues ■ Maintain a requirements traceability matrix ■ Use a requirements management tool	■ Train business analysts ■ Educate stakeholders about requirements ■ Educate developers about application domain ■ Define a requirements engineering process ■ Create a glossary	■ Select an appropriate life cycle ■ Plan requirements approach ■ Estimate requirements effort ■ Base plans on requirements ■ Identify requirements decision makers ■ Renegotiate commitments ■ Manage requirements risks ■ Track requirements effort ■ Review past lessons learned

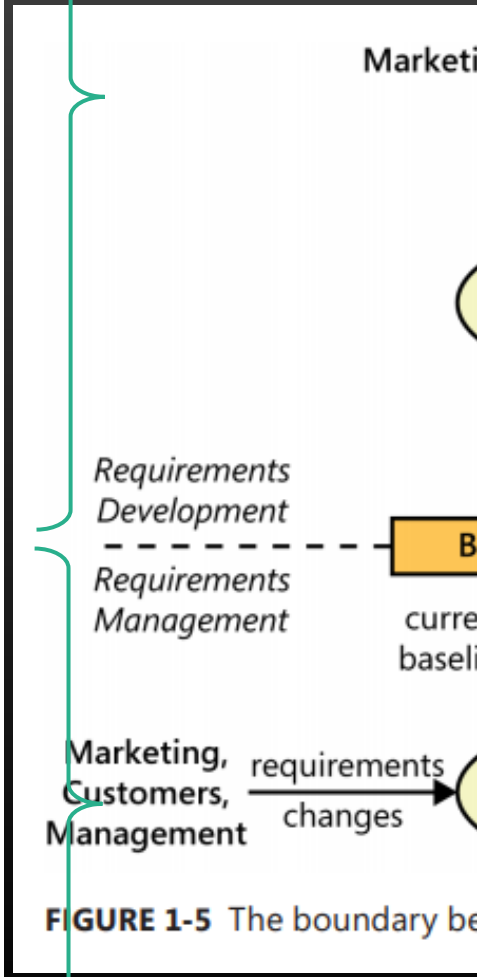


FIGURE 1-5 The boundary be

What is a requirements traceability matrix?

The most common way to represent the links between requirements and other elements in a system is in a **requirements traceability matrix**.

TABLE 29-2 Requirements traceability matrix showing links between use cases and functional requirements

	Use case			
Functional requirement	**UC-1**	**UC-2**	**UC-3**	**UC-4**
FR-1	↵			
FR-2	↵			
FR-3			↵	
FR-4			↵	
FR-5		↵		↵
FR-6			↵	

TABLE 3-1 Requirements engineering good practices

Elicitation	Analysis	Specification	Validation
■ Define vision and scope ■ Identify user classes ■ Select product champions ■ Conduct focus groups ■ Identify user requirements ■ Identify system events and responses ■ Hold elicitation interviews ■ Hold facilitated elicitation workshops ■ Observe users performing their jobs ■ Distribute questionnaires ■ Perform document analysis ■ Examine problem reports ■ Reuse existing requirements	■ Model the application environment ■ Create prototypes ■ Analyze feasibility ■ Prioritize requirements ■ Create a data dictionary ■ Model the requirements ■ Analyze interfaces ■ Allocate requirements to subsystems	■ Adopt requirement document templates ■ Identify requirement origins ■ Uniquely label each requirement ■ Record business rules ■ Specify nonfunctional requirements	■ Review the requirements ■ Test the requirements ■ Define acceptance criteria ■ Simulate the requirements

Requirements management	Knowledge	Project management
■ Establish a change control process ■ Perform change impact analysis ■ Establish baselines and control versions of requirements sets ■ Maintain change history ■ Track requirements status ■ Track requirements issues ■ Maintain a requirements traceability matrix ■ Use a requirements management tool	■ Train business analysts ■ Educate stakeholders about requirements ■ Educate developers about application domain ■ Define a requirements engineering process ■ Create a glossary	■ Select an appropriate life cycle ■ Plan requirements approach ■ Estimate requirements effort ■ Base plans on requirements ■ Identify requirements decision makers ■ Renegotiate commitments ■ Manage requirements risks ■ Track requirements effort ■ Review past lessons learned

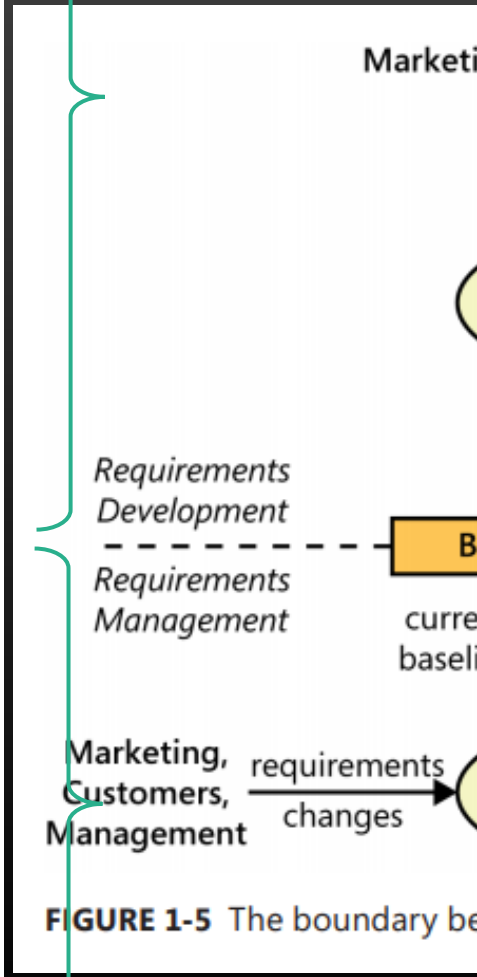


FIGURE 1-5 The boundary be

Requirement attributes to consider:

- Date the requirement was created
- Current version number of the requirement
- Author who wrote the requirement
- Priority
- Status
- Origin or source of the requirement
- Rationale behind the requirement
- Release number or iteration to which the requirement is allocated
- Stakeholder to contact with questions or to make decisions about proposed changes
- Validation method to be used or acceptance criteria

Requirements Management Tool

- Larger project teams will benefit from letting users import requirements from source documents, define attribute values, filter and display the database contents, export requirements in various formats, define traceability links, and connect requirements to items stored in other software development tools.

E.g.: IBM Rational DOORS

Readings & references

Core readings	Suggested readings
• [Wiegers13] – Multiple chapters used.	• [Dennis15] – Chap. 3 – Requirements Determination • [Pressman15] – Chap. 8 – Understanding Requirements