CT70A2000
Requirements Engineering

Shola Oyedeji

LUT University

LECTURE 1

RE – What & Why?

Course Introduction

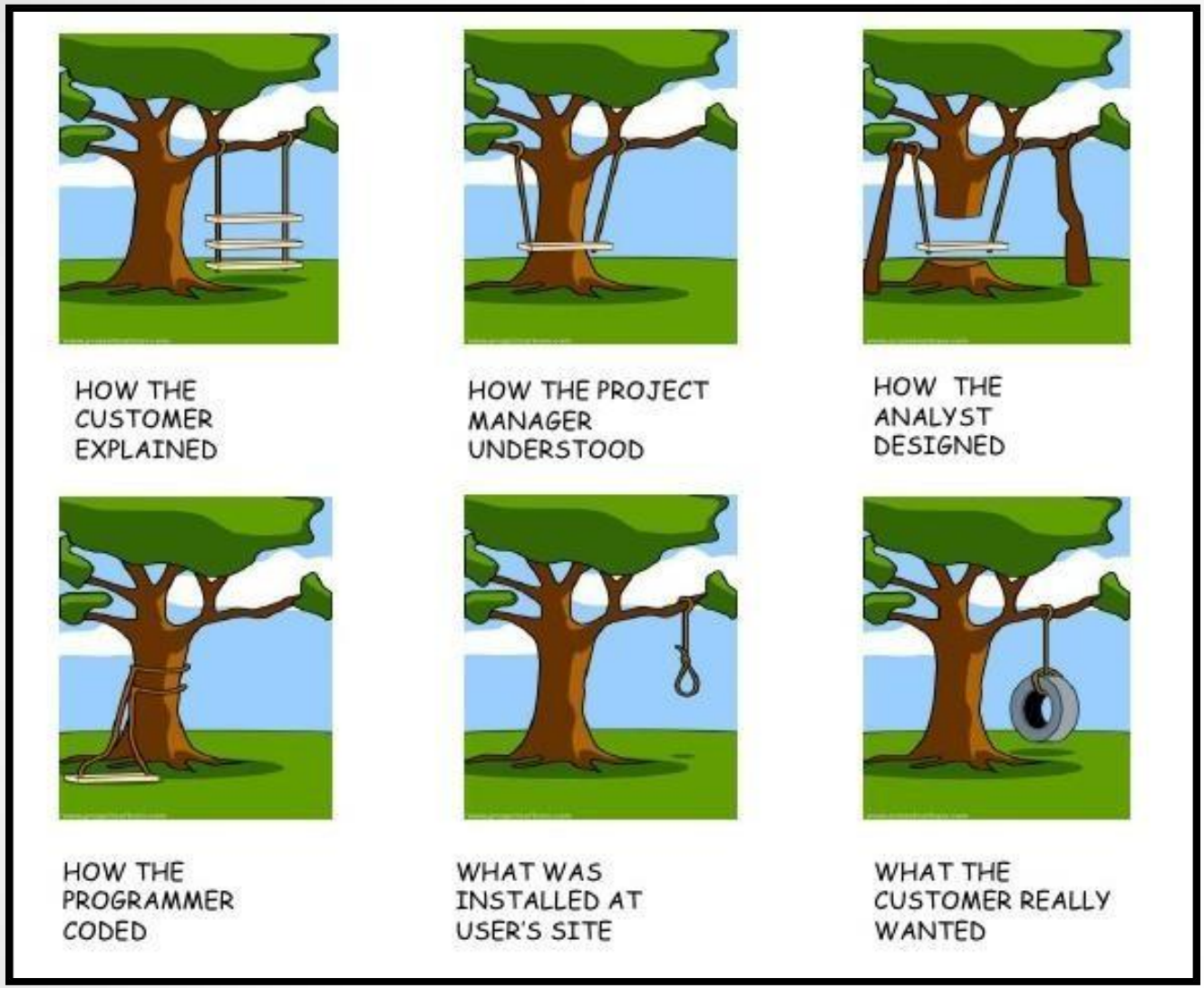
Introduction to Traditional RE

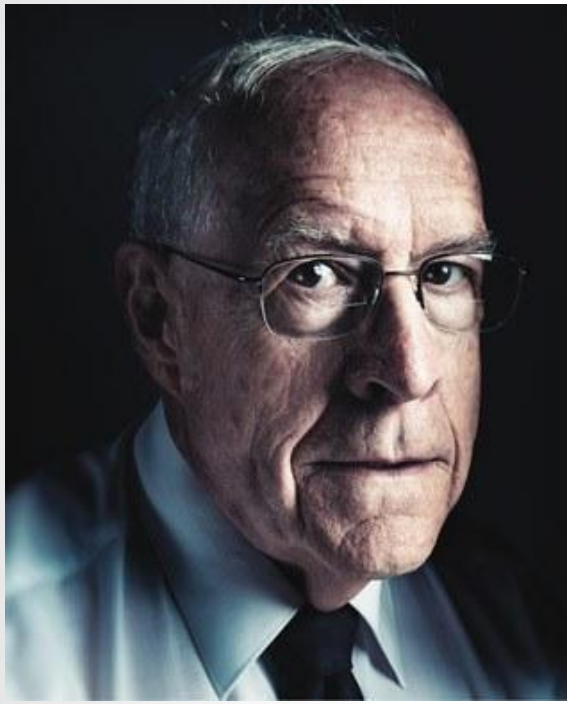
Establishing Product Vision & Scope

What is a requirement?

Requirements specify what the software must do without describing how to do it.

- *A requirement is "anything that drives design choices",* (Lawrence 1997).
- The IEEE Standard Glossary of Software Engineering Terminology (1990) defines a requirement as:
  1. *A condition or capability needed by a user to solve a problem or achieve an objective.*
  2. *A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed document.*
  3. *A documented representation of a condition or capability as in 1 or 2.*
- *"Requirements are…a specification of what should be implemented. They are descriptions of how the system should behave, or of a system property or attribute. They may be a constraint on the development process of the system."* (Sommerville and Sawyer 1997)


HOW THE CUSTOMER EXPLAINED
HOW THE PROJECT MANAGER UNDERSTOOD
HOW THE ANALYST DESIGNED
HOW THE PROGRAMMER CODED
WHAT WAS INSTALLED AT USER'S SITE
WHAT THE CUSTOMER REALLY WANTED

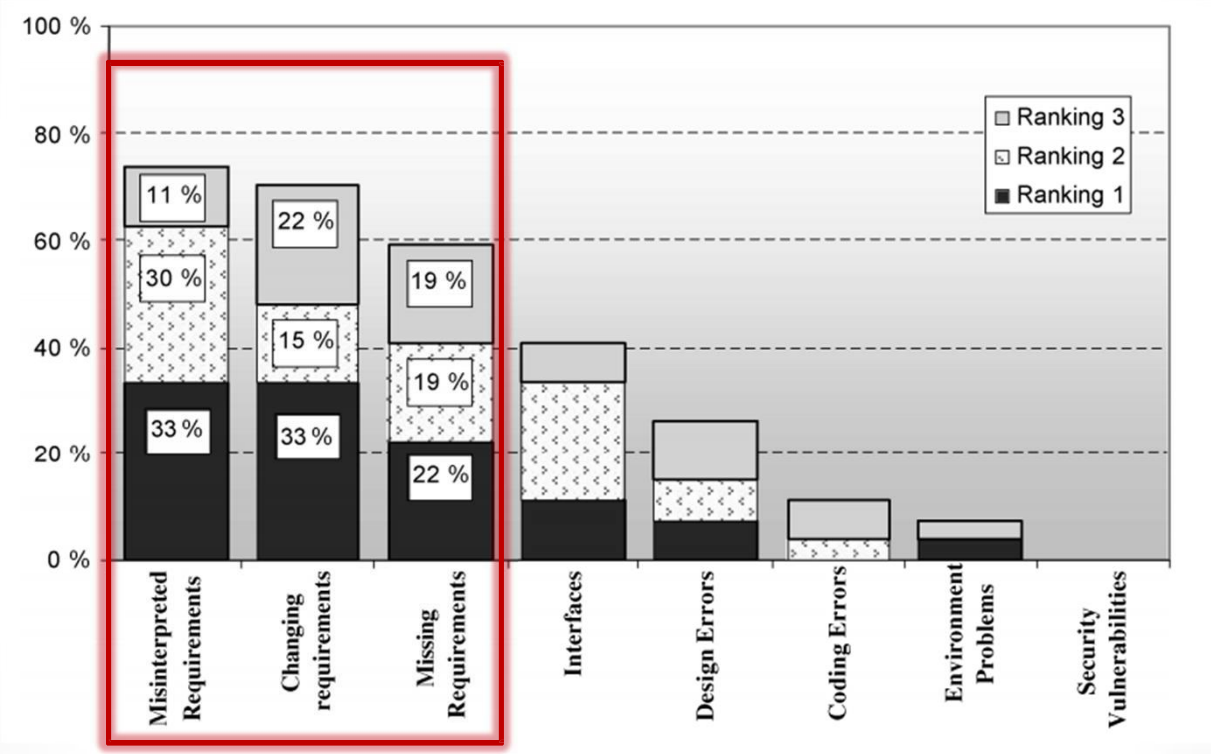


“The hardest single part of building a software system is deciding precisely what to build.

- No other part of the conceptual work is as difficult as establishing the detailed technical requirements
- No other part of the work so cripples the resulting system if done wrong.
- No other part is as difficult to rectify later.”

Fred Brooks (1987), “No Silver Bullet: Essence and Accidents of Software Engineering”

Major sources of software errors



Seija Komi-Sirviö and Maarit Tihinen: Lessons Learned by Participants of Distributed Software Development Knowledge and Process Management Volume 12 Number 2 pp 108–122 (2005)

Misunderstandings are expensive

COMMON REQUIREMENTS RISKS:

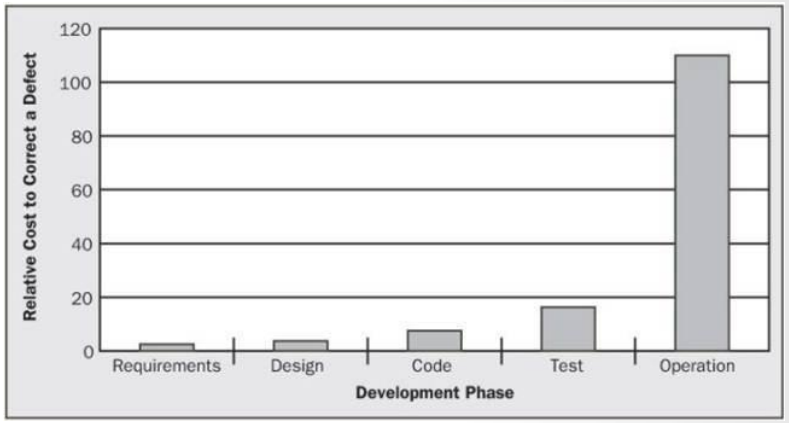
Insufficient User Involvement leads to late-breaking requirements that delay project completion. There's no substitute for having the development team work directly with actual users throughout the project

Creeping User Requirements. As requirements evolve and grow during development, projects often exceed their planned schedules and budgets.

Ambiguous Requirements. One symptom of ambiguity is that a reader can interpret a requirement statement in several ways. Another sign is that multiple readers of a requirement arrive at different understandings of what it means.

Gold Plating takes place when a developer adds functionality that wasn't in the requirements specification but that the developer believes "the users are just going to love."

Minimal Specification. Sometimes marketing staff or managers are tempted to create a limited specification, perhaps just a product concept sketched on a napkin. They expect the developers to flesh out the spec while the project progresses.



Relative cost to correct a requirement defect depending on when it is discovered

K. E. Wiegers, *Software Requirements*. Microsoft Press, 2003.

Overlooked User Classes. Most products have several groups of users who might use different subsets of features, have different frequencies of use, or have varying experience levels. If you don't identify the important user classes for your product early on, some user needs won't be met.

Inaccurate Planning. Vague, poorly understood requirements lead to overly optimistic estimates, which come back to haunt us when the inevitable overruns occur.

Why to pay attention to RE?

- Fewer requirements defects
- Reduced development rework
- Fewer unnecessary features
- Lower enhancement costs
- Faster development
- Fewer miscommunications
- Reduced scope creep
- Reduced project chaos
- More accurate system-testing estimates
- Higher customer and team member satisfaction

LECTURE 1

RE – What & Why?

Course Introduction

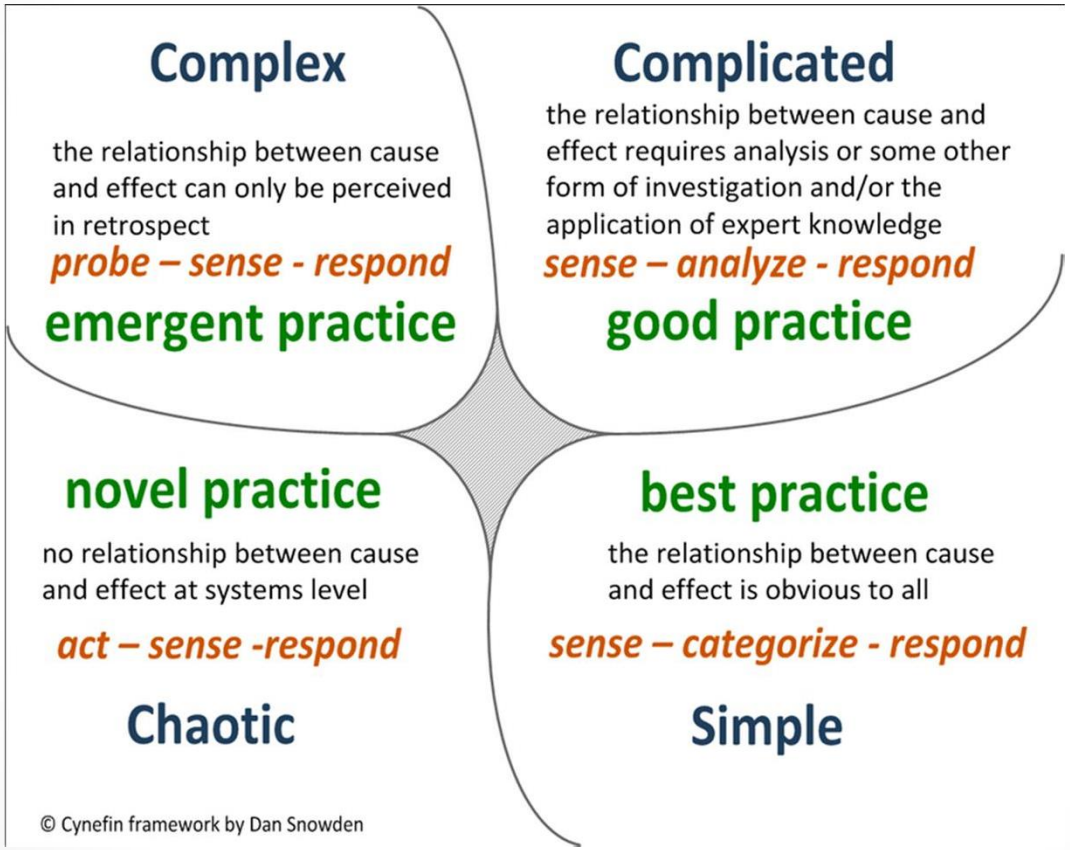
Introduction to Traditional RE

Establishing Product Vision & Scope

CYNEFIN FRAMEWORK

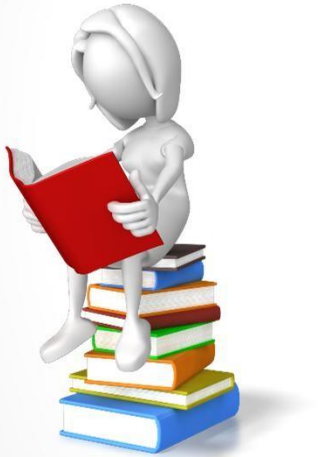
UNORDERED

ORDERED



D. J. Snowden and M. E. Boone, "A Leader's Framework for Decision Making", *Harvard Business Review, vol. 85, 2007*
https://hbr.org/2007/11/a-leaders-framework-for-decision-making

COURSE STRUCTURE



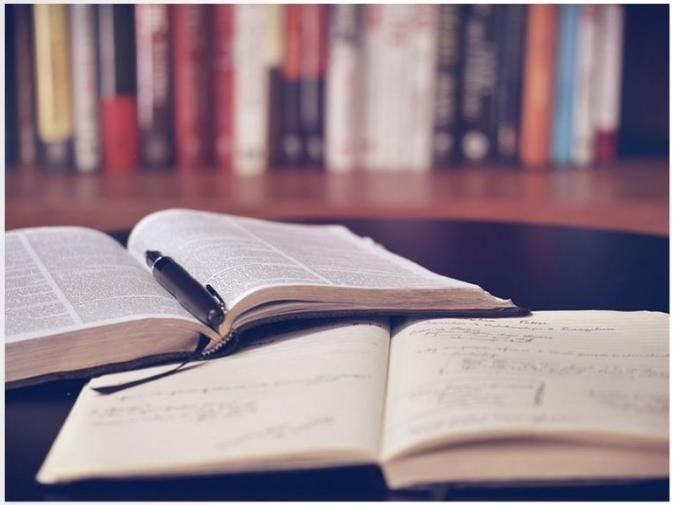
MODULE 1:
Learn Traditional RE practices

MODULE 2:
Explore for present and future RE practices

LEARNING BY DOING

1. Form RE-teams up to 6 persons
2. Customer team will be assign to your team
3. Select Vision & Scope for your customer project. Discuss with the customer team:
   - Identify key stakeholders
   - Understand customer needs and problems to develop requirements
4. Plan your RE activities, including
   - Agreeing on roles&responsibilities
   - How to elicit requirements?
   - What models to develop?
   - How to prioritize requirements?
   - How to manage requirements?
5. Elicit, develop, model, prioritize and document your requirements. Documents to be produced:
   - Vision & Scope document
   - Use Case Document
   - Requirement analysis with sustainability dimensions
   - Software Requirements Specfication (SRS)
   - Requirements Management Plan
6. Conduct requirements review meetings of RE-documentation by 2 other teams

STUDY MATERIAL



- No official book for the course
- All necessary information shall be given by:
  - Lecture material
  - Digital lessons (Moodle)
  - E-books and internet articles (listed in Moodle)

GRADING

- 30% Module 1 teamwork
- 20% Module 2 teamwork
- 20% Weekly Lecture Task
- 10% Individual Learning Diary
- 20% Examination

There are no opportunities to improve grade afterwards!

Teams are partly responsible themselves for achieving fair grading!

- Use Moodle to report uneven work within a team
- Use Moodle to report **clearly** unfair peer-assessment of your work
- Do not miss deadlines under any circumstances!
- Always check that your submission was uploaded to Moodle correctly!

LECTURE 1

RE – What & Why?

Course Introduction

Introduction to Traditional RE

Establishing Product Vision & Scope

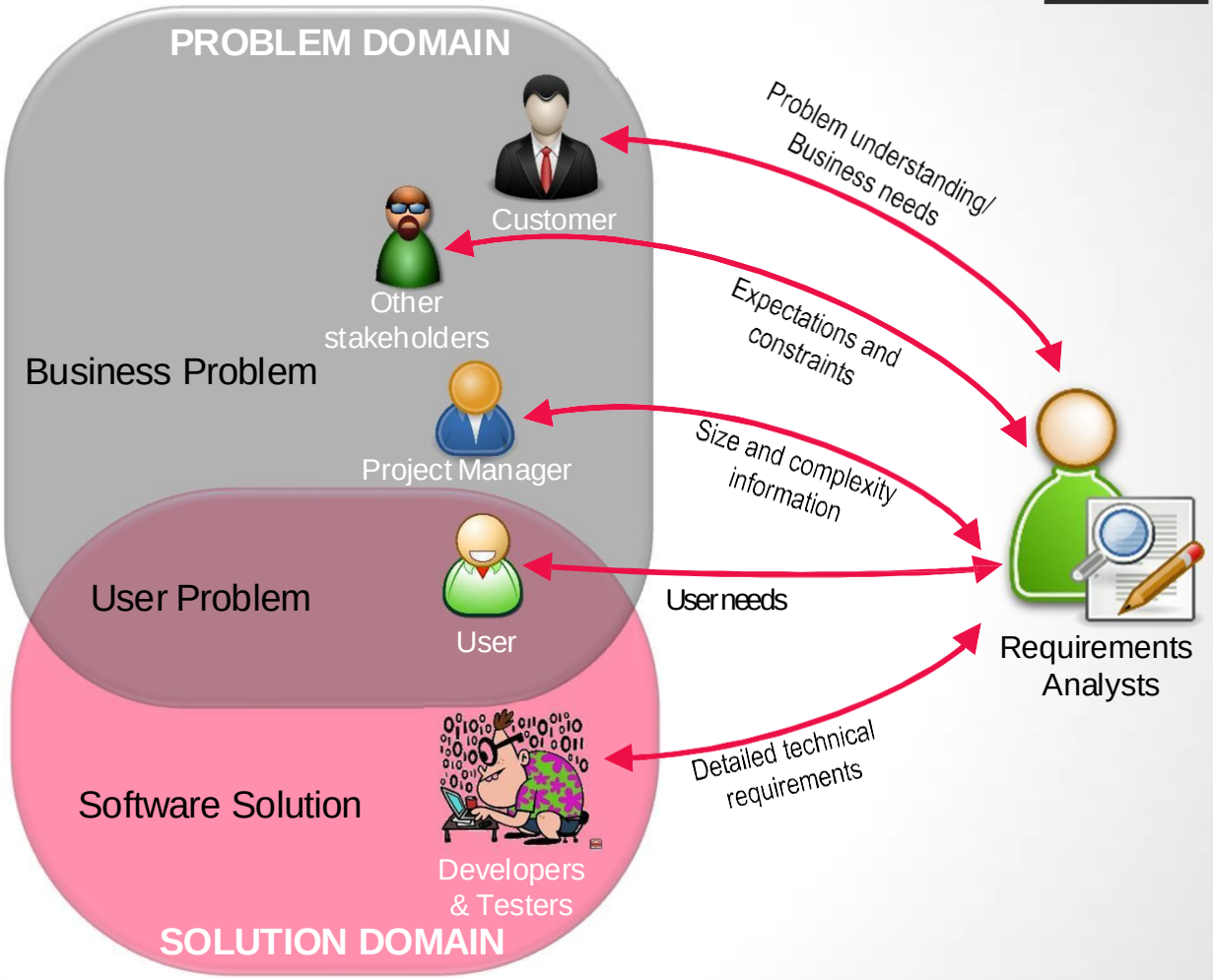
THE BIG PICTURE

DOCUMENTATION:

Project Charter/
Vision and scope document

User Requirements

Software Requirements
Specification



REQUIREMENTS ANALYSTS

Requirements Analysts

- Individual who has the primary responsibility to gather, analyze, document, and validate the needs of the project stakeholders.
- Serves as the principal conduit through which requirements flow between the customer community and the software development team
- Project role, not necessarily a job title
- Responsible in collecting and disseminating product information, whereas the project manager takes the lead in communicating project information
- Steers the project participants toward a requirements agreement that leads to a win/win/win outcome so that:
  - Customers are delighted with the product
  - The developing organization is happy with the business outcomes
  - The development team members are proud of the good work they did on a challenging and rewarding project.

K. E. Wiegers, *Software Requirements*. Microsoft Press, 2009.

REQUIREMENTS ANALYSTS SKILLS AND TASKS

Essential skills needed:

- Listening
- Interviewing and questioning
- Analytical
- Facilitation
- Observation
- Writing
- Organizational
- Modeling
- Interpersonal
- Creativity

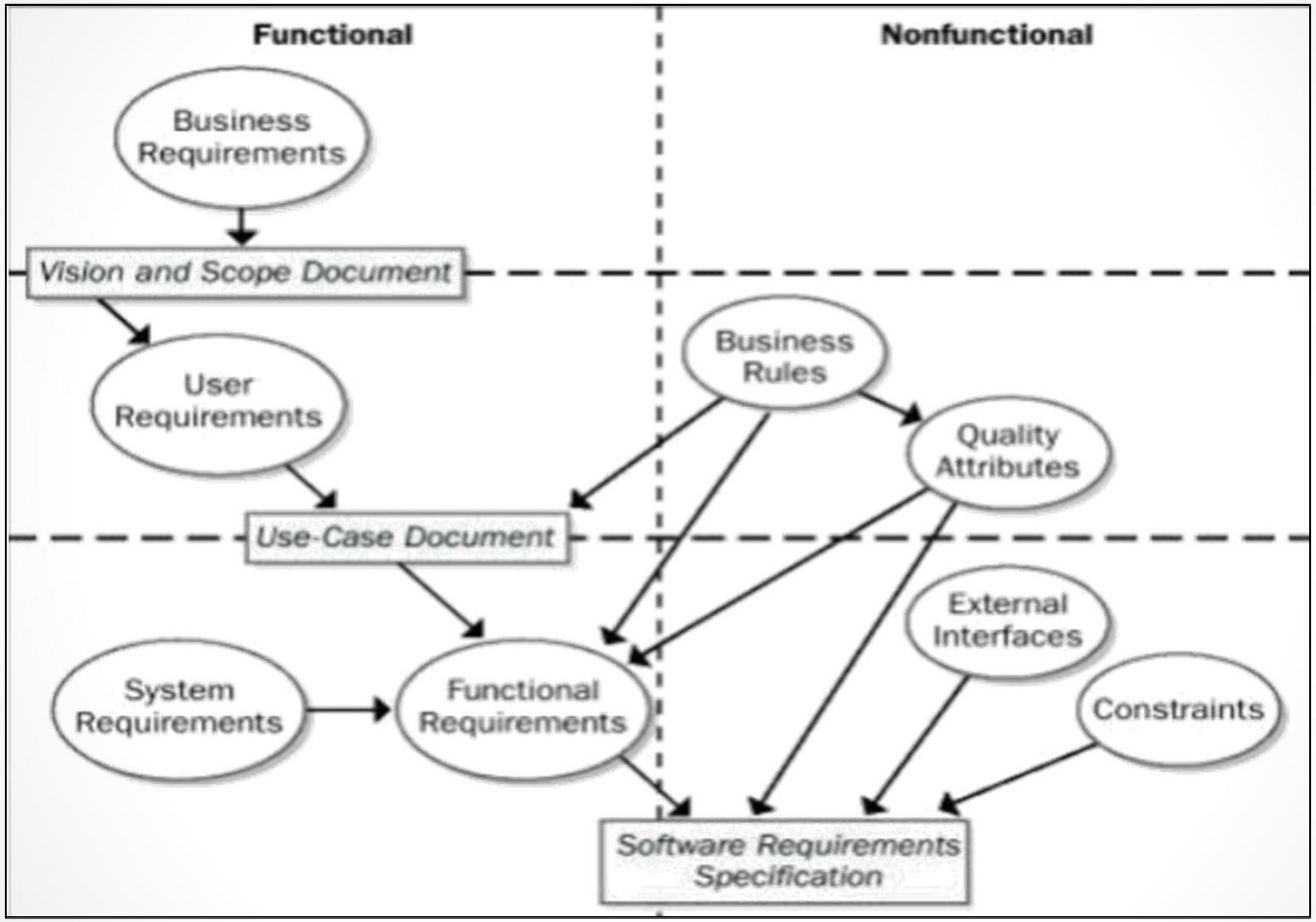
Requirements Analysts

Tasks to be done:

- Define business requirements
- Identify project stakeholders and user classes
- Elicit requirements
- Analyze requirements
- Write requirements specifications
- Model the requirements
- Lead requirements validation
- Facilitate requirements prioritization
- Manage requirements

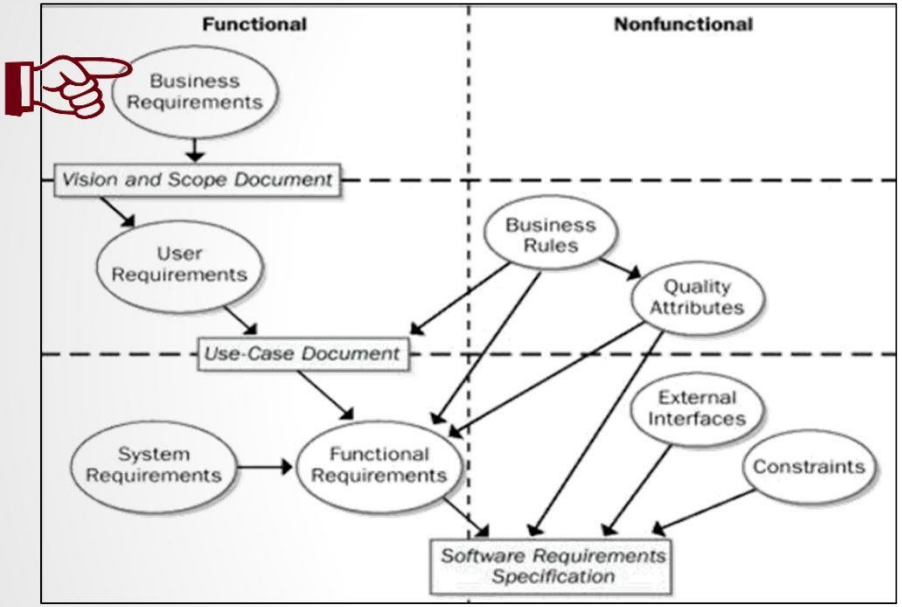
K. E. Wiegers, *Software Requirements*. Microsoft Press, 2009.

LEVELS OF REQUIREMENTS



K. E. Wiegers, *Software Requirements*. Microsoft Press, 2009.

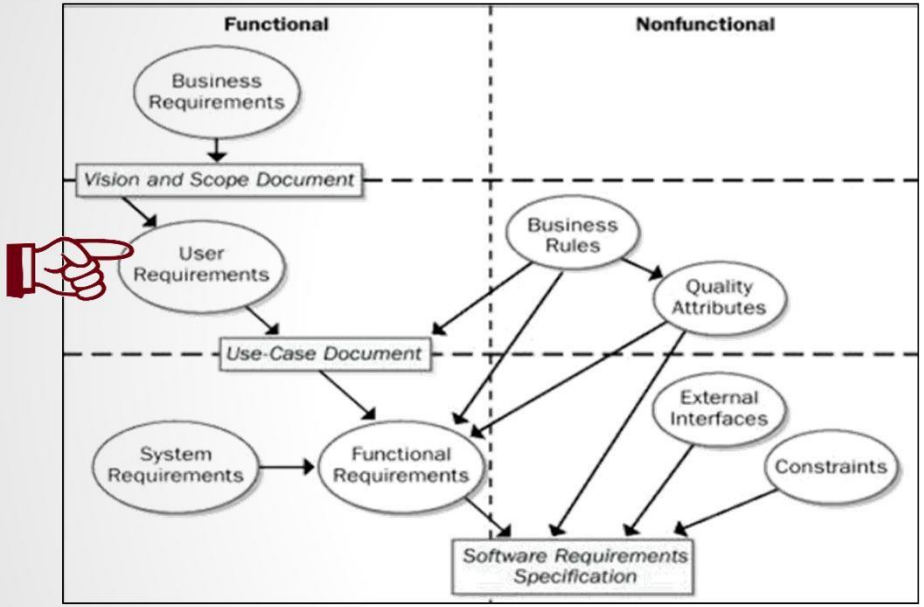
Business Requirements



K. E. Wiegers, *Software Requirements*. Microsoft Press, 2009.

- Represent high-level objectives of the organization or customer who requests the system.
- Typically come from the funding sponsor for a project, the acquiring customer, the manager of the actual users, the marketing department, or a product visionary.
- Describe why the organization is implementing the system—the objectives the organization hopes to achieve.
- It called sometimes *a vision and scope document*, or a *project charter* or a *market requirements document*.
- Defining the project scope is the first step in controlling the common problem of scope creep.

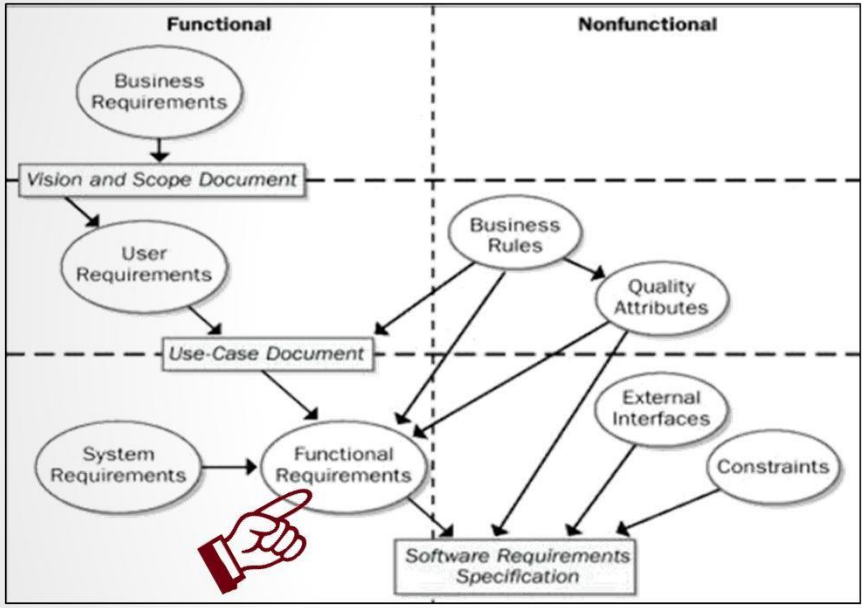
User Requirements



K. E. Wiegers, *Software Requirements*. Microsoft Press, 2009.

- Bridge the business requirements and software requirements.
- Describe user goals or tasks that the users must be able to perform with the product.
- Valuable ways to represent user requirements include use cases, scenario descriptions, and event-response tables.
- User requirements therefore describe what the user will be able to do with the system.
- An example of a use case is "Make a Reservation" using an airline, a rental car, or a hotel Web site.

Functional Requirements



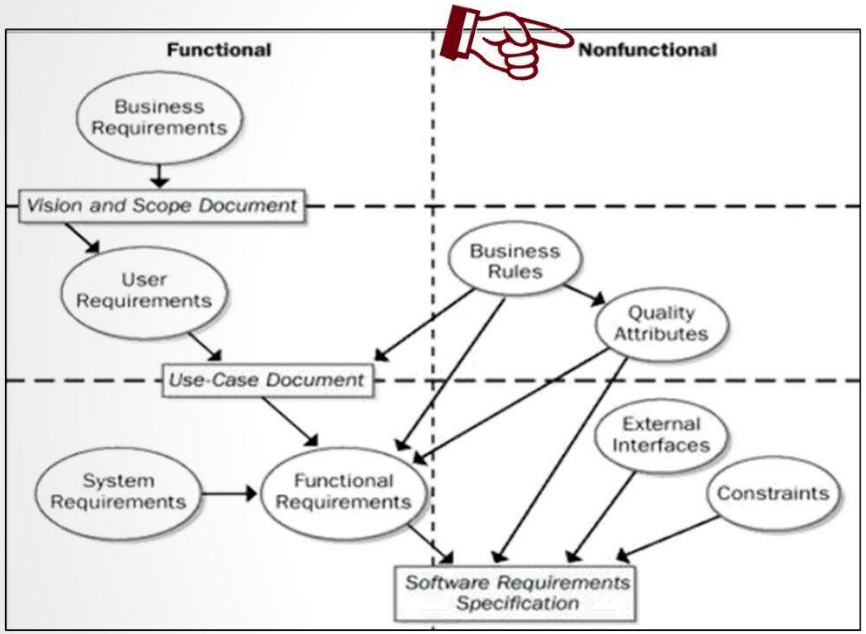
K. E. Wiegers, *Software Requirements*. Microsoft Press, 2009.

- Specify the software functionality that the developers must build into the product to enable users to accomplish their tasks, thereby satisfying the business requirements.
- Sometimes called *behavioral requirements*, these are the traditional *"shall"* statements:

  "The system shall e-mail a reservation confirmation tothe user."
- Functional requirements are documented in a *software requirements specification (SRS)*, which describes as fully as necessary the expected behavior of the software system.
- The SRS is used in development, testing, quality assurance, project management, and related project functions.

System requirements: Top-level requirements for a product that contains multiple subsystems.

Non-Functional Requirements

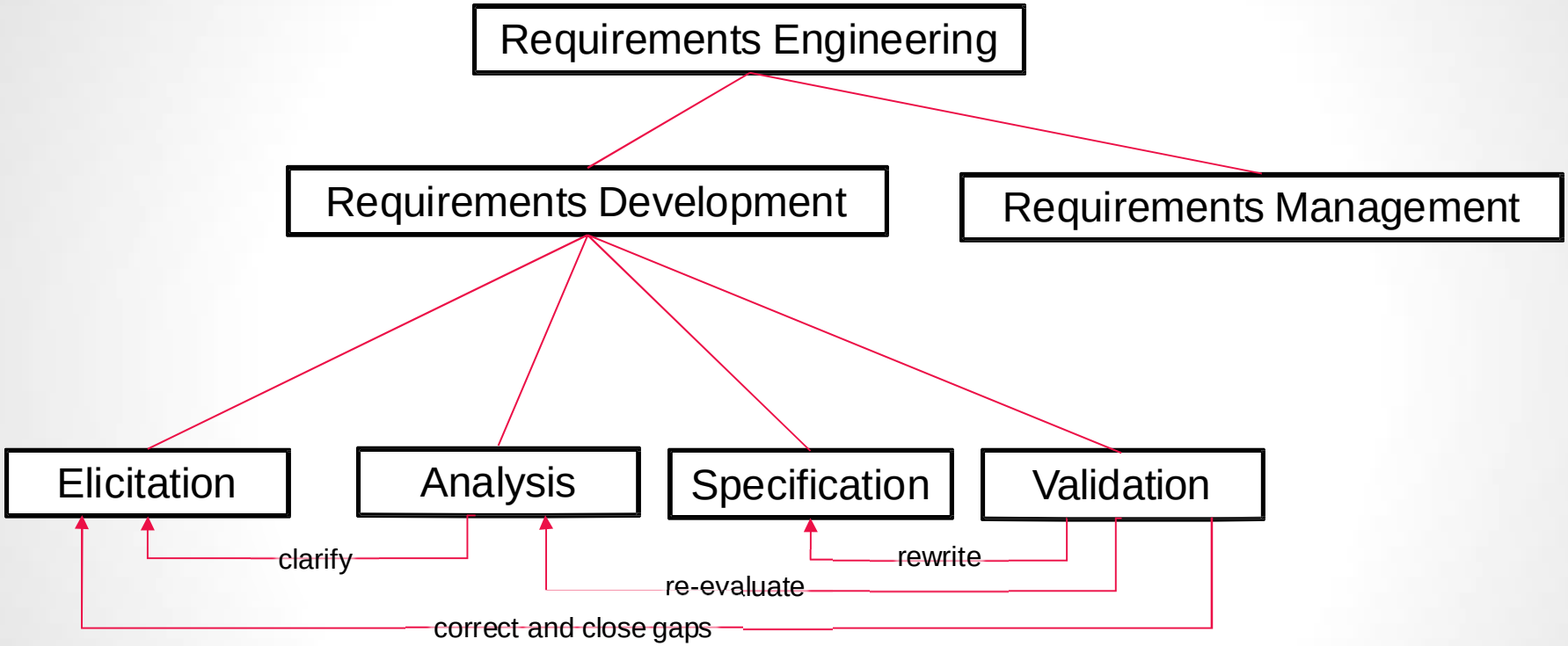


K. E. Wiegers, *Software Requirements*. Microsoft Press, 2009.

In addition to the functional requirements, the SRS contains nonfunctional requirements. These include: performance goals and descriptions of quality attributes.

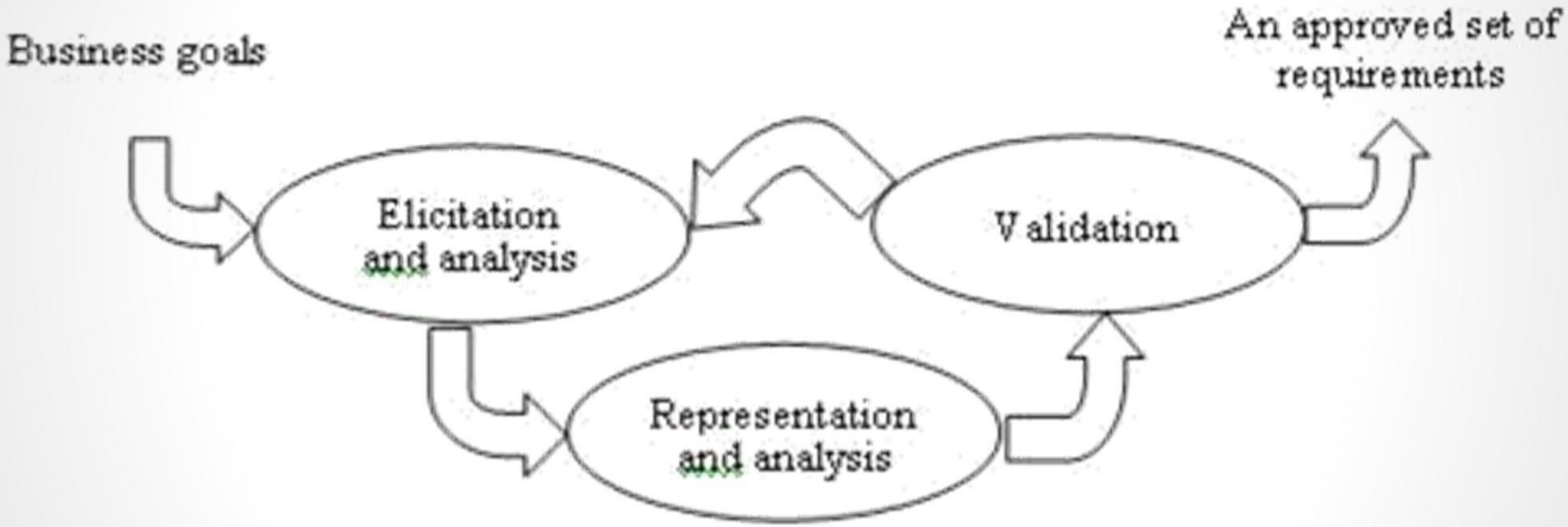
- *Business Rules* including corporate policies, government regulations, industry standards, accounding practices, and computational algorithms.
- *Quality attributes* that augment the description of the product's functionality by describing the product's characteristics in various dimensions that are important either to users or to developers. For example: *usability, portability, integrity, efficiency, and robustness*.
- *External interfaces* between the system and the outside world.
- *Constraints* impose restrictions on the choices available to the developer for design and construction of the product

WHAT DOES RE CONSIST OF?



K. E. Wiegers, *More about Software Requirements*. Microsoft Press, 2006.

Requirements Development

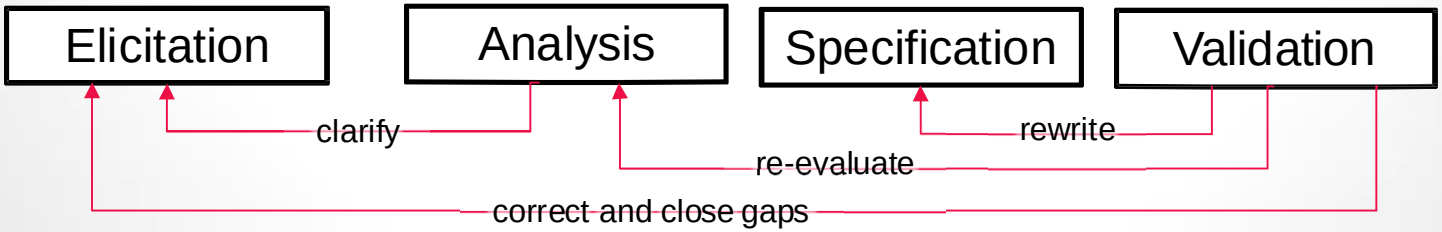


http://www.soberit.hut.fi/T-76.4115/13-14/instructions/requirements_definition.html

Requirements Development

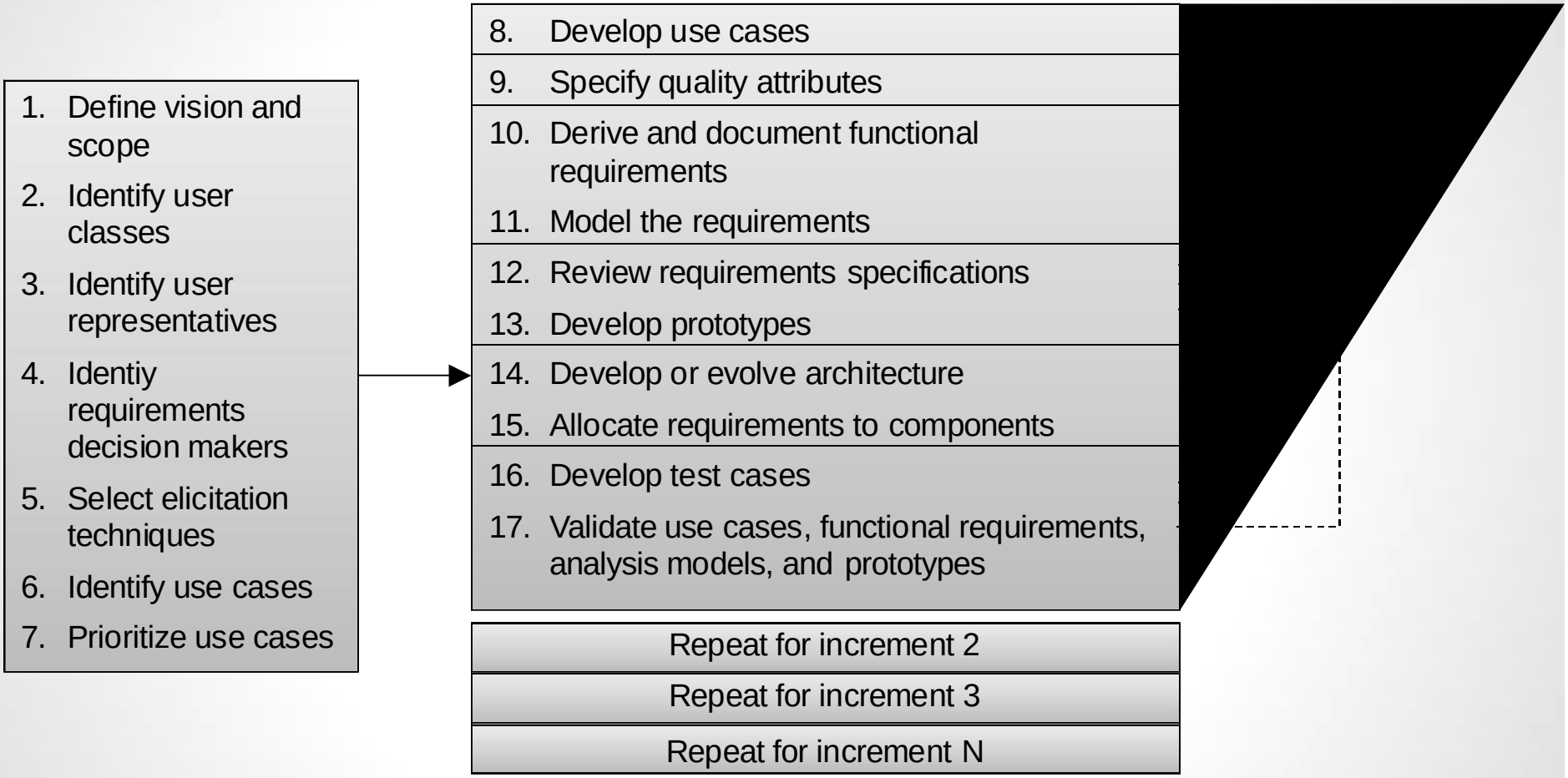
Requirements development are subdivided into elicitation, analysis, specification, and validation. They are done iteratively and including the following:

- Identifying the product's expected user classes
- Eliciting needs from individuals who represent each user class
- Understanding user tasks and goals and the business objectives with which those tasks align
- Analyzing the information received from users to distinguish their task goals from functional requirements, nonfunctional requirements, business rules, suggested solutions, and extraneous information
- Allocating portions of the top-level requirements to software components defined in the system architecture
- Understanding the relative importance of quality attributes
- Negotiating implementation priorities
- Translating the collected user needs into written requirements specifications and models
- Reviewing the documented requirements to ensure a common understanding of the users' stated requirements and to correct any problems before the development group accepts them.



K. E. Wiegers, *Software Requirements*. Microsoft Press, 2003.

One suggested requirements development process



K. E. Wiegers, *Software Requirements*. Microsoft Press, 2003.

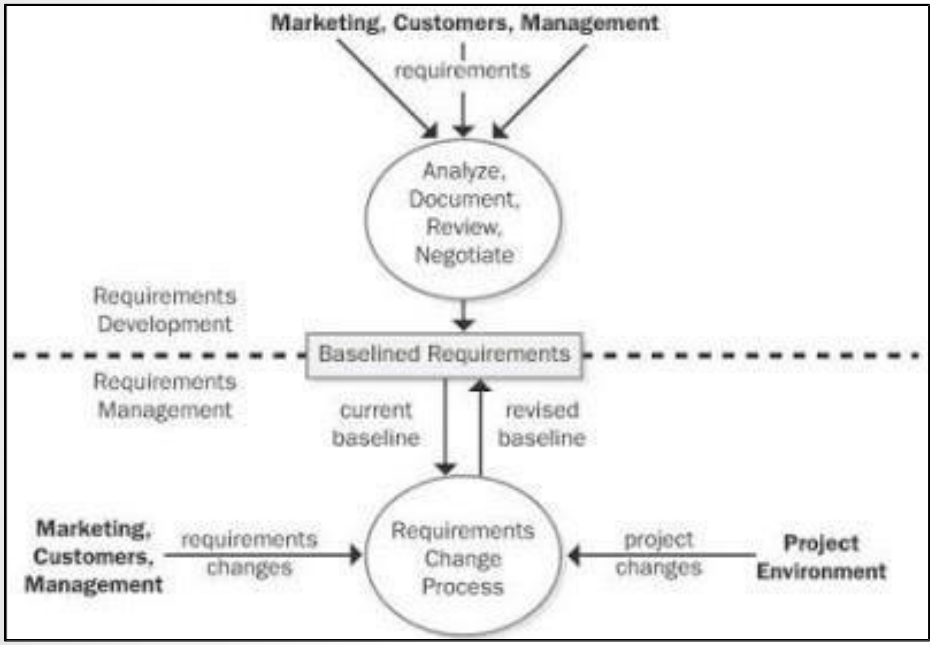
RE-related differences between bespoke development and product development

Aspect	BESPOKE DEVELOPMENT	PRODUCT DEVELOPMENT
Primary goal	Compliance to the specification.	Time-to-market. Requirements are jettisoned rather than allowing delay of release.
Measure of success	Satisfaction, acceptance.	Sales, market share, product reviews.
Life-cycle	One release, then maintenance.	Several releases, as long as there is a market for the product.
Requirements conception	Elicitated, analyzed, validated.	Invented. Either the market (marketing department) permits a feature, or technology does.
Requirements specification	Used as a contract between customer and supplier.	Rarely exists in orthodox RE terms, if so, they are much less formal. Requirements are communicated verbally.
Users	Known or identifiable. Termed user, end user.	Unknown, may not exist until the first product is on the market. Termed customer.
Physical distance to users	Usually small.	Usually large.
Main stakeholder	Customer organization.	Developing organization.
Specific RE issues	Elicitation, modeling, validation, conflict resolution.	Managing a steady stream of new requirements. Prioritizing, cost-estimating, release planning.
Developer's association with the software	Short-term (until end of project).	Long-term, promoting e.g. investment in maintainability.
Validation	Ongoing process.	Very late, e.g. at trade fairs.
Use of RE standards and explicit methods	More common.	Rare.
Use of iterative techniques	Less common.	More common.
Domain expertise available on the development team	More common.	Less common (product development often breaks new ground).

(Compiled by Carlshamre (2001, p. 59))

Requirements Management

Establishing and maintaining an agreement with the customer on the requirements for the software project.



Boundary between requirements development and requirements management
K. E. Wiegers, *Software Requirements*. Microsoft Press, 2003.

Requirements management activities include the following:

- Defining the requirements baseline (a snapshot in time representing the currently agreed-upon body of requirements for a specific release)
- Reviewing proposed requirements changes and evaluating the likely impact of each change before approving it
- Incorporating approved requirements changes into the project in a controlled way
- Keeping project plans current with the requirements
- Negotiating new commitments based on the estimated impact of requirements changes
- Tracing individual requirements to their corresponding designs, source code, and test cases
- Tracking requirements status and change activity throughout the project

LECTURE 1

RE – What & Why?

Course Introduction

Introduction to Traditional RE

Establishing Product Vision & Scope

Vision & Scope

Product vision:

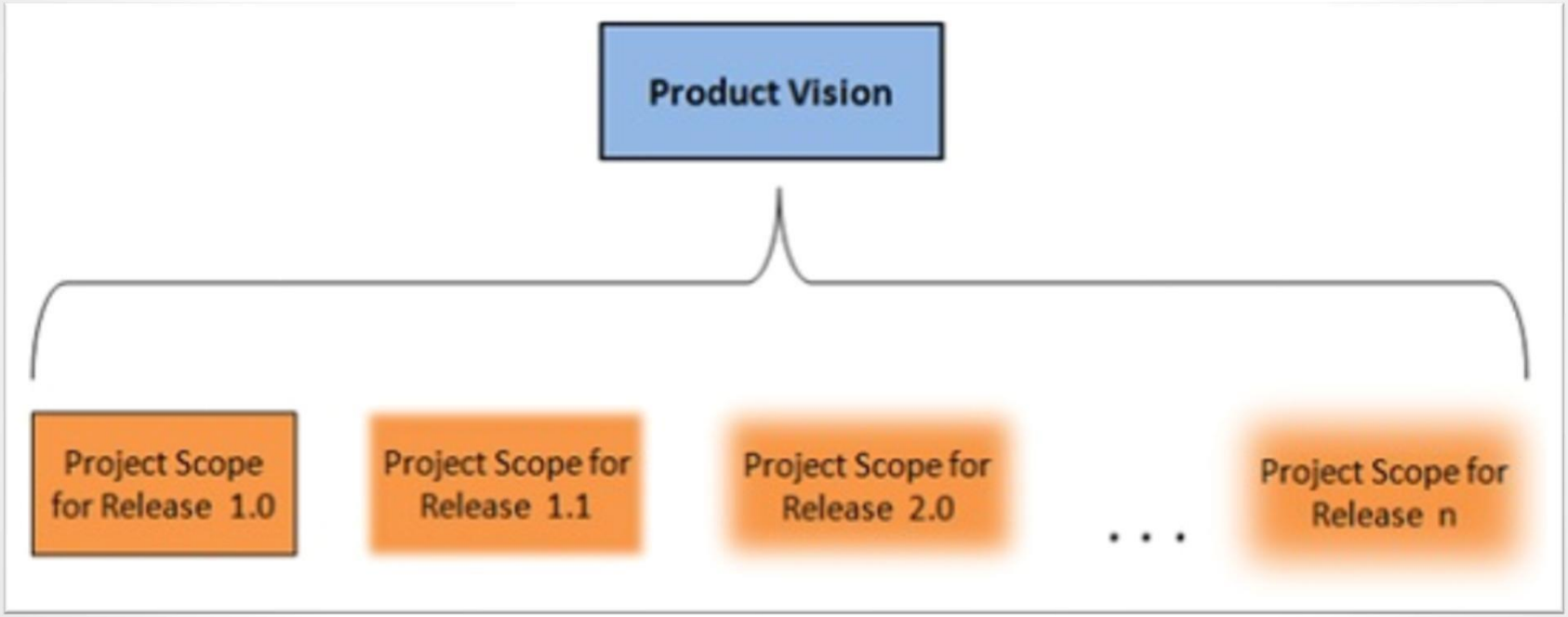
- A long-term strategic concept of the ultimate purpose and form of a new system.
  - Aligns all stakeholders in a common direction
  - Describes what the product is about and what it eventually could become

Project scope:

- Portion of the ultimate long-term product vision the current project will address
  - Draws the boundary between what's in and what's out.
  - Defines the project's limitations.
- Scope details are represented by the requirements baseline that the team defines for the project

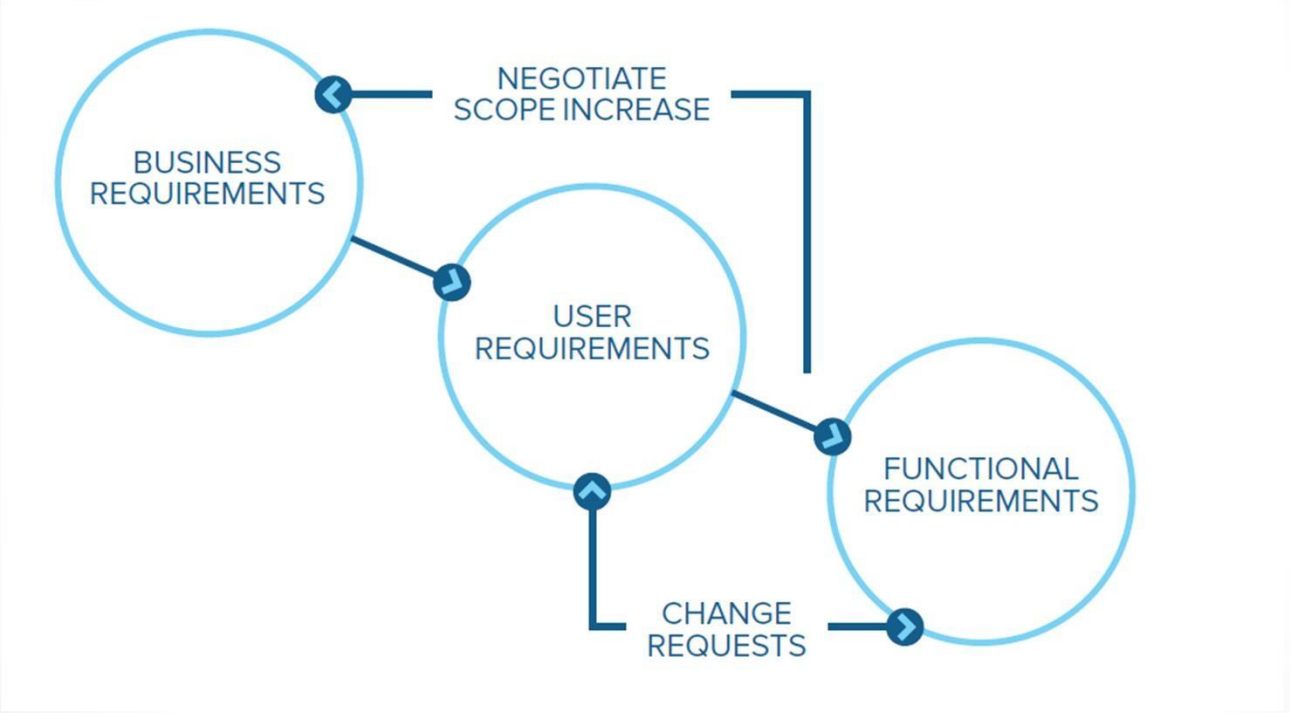
K. E. Wiegers, *Software Requirements*. Microsoft Press, 2003.

How are Product Vision and Project Scope related?



K. E. Wiegers, *Software Requirements*. Microsoft Press, 2003.

Managing Scope Creep



Karl Wiegers, "Four Techniques for Defining Project Scope".

Vision and Scope Document

- Collects the busines requirements into a single document that sets the stage for subsequent development work.
- The owner of the vision and scope document is the project's executive sponsor, funding authority, or someone in a similiar role

Suggested template:

1. **Business requirements**
   - 1.1 Background
   - 1.2 Business opportunity
   - 1.3 Business objectives
   - 1.4 Success metrics
   - 1.5 Vision statement
   - 1.6 Business risks
   - 1.7 Business assumptions and dependencies
2. **Scope and limitations**
   - 2.1 Major features
   - 2.2 Scope of initial release
   - 2.3 Scope of subsequent releases
   - 2.4 Limitations and exclusions
3. **Business context**
   - 3.1 Stakeholder profiles
   - 3.2 Project priorities
   - 3.3 Deployment considerations

K. E. Wiegers, *Software Requirements*. Microsoft Press, 2003.

V&S Document: 1. Business Requirements

1. **Business requirements**
   - 1.1 Background
   - 1.2 Business opportunity
   - 1.3 Business objectives
   - 1.4 Success metrics
   - 1.5 Vision statement
   - 1.6 Business risks
   - 1.7 Business assumptions and dependencies
2. **Scope and limitations**
   - 2.1 Major features
   - 2.2 Scope of initial release
   - 2.3 Scope of subsequent releases
   - 2.4 Limitations and exclusions
3. **Business context**
   - 3.1 Stakeholder profiles
   - 3.2 Project priorities
   - 3.3 Deployment considerations

1. Background

Summarize the rationale and context for the new product.

2. Business opportunity

- *For a commercial product:* describe the market opportunity
- *For a corporate information system*: describe the business problem to be improved
- Include a comparative evaluation of existing products and potential solutions, indicating why the proposed product is attractive and the advantages it provides.
- Describe the problems that cannot currently be solved without the product.
- Show how it aligns with market trends, technology evolution, or corporate strategic directions.

3. Business objectives

Summarize the important business benefits the product will provide in a quantitative and measurable way

4. Success metrics

State the factors that have the greatest impact on achieving defined success

5. Vision statement

Summarize the long-term purpose and intent, satisfying the needs of diverse stakeholders

6. Business risks

Summarize the major business risks associated with developing-or not developing-this product

7. Business assumptions and dependencies

- Record any assumptions that the stakeholders made when conceiving the project and writing this vision and scope document
- Also, record major dependencies the project has on external factors outside its control. These could include pending industry standards, or government regulations, other projects, third-party suppliers, or development partners

K. E. Wiegers, *Software Requirements*. Microsoft Press, 2003.

Vision Statement

1. **Business requirements**
   - 1.1 Background
   - 1.2 Business opportunity
   - 1.3 Business objectives
   - 1.4 Success metrics
   - 1.5 Vision statement
   - 1.6 Business risks
   - 1.7 Business assumptions and dependencies
2. **Scope and limitations**
   - 2.1 Major features
   - 2.2 Scope of initial release
   - 2.3 Scope of subsequent releases
   - 2.4 Limitations and exclusions
3. **Business context**
   - 3.1 Stakeholder profiles
   - 3.2 Project priorities
   - 3.3 Deployment considerations

Suggested format:

- **For** [*target customer*]
- **Who** [*statement of the need or opportunity*]
- **The** [*product name*]
- **Is** [*a product category*]
- **That** [*key benefit, compelling reason to buy or use*]
- **Unlike** [*primary competitive alternative, current system, or current business process*],
- **Our product** [*statement of primary differentation and advantages of new product*].

EXAMPLE:

For scientists **who** need to request containers of chemicals, **the** Chemical Tracking System **is** an information system **that** will provide a single point of access to the chemical stockroom and to vendors. The system will store the location of every chemical container within the company, the quantity of material remaining in it, and the complete history of each container's locations and usage. **This** system will save the company 25% on chemical costs in the first year of use by allowing the company to fully exploint chemicals that are already available within the company, dispose of fewer partially used or expired containers, and use a single standard chemical purchasing process. **Unlike** the current manual ordering processes, **our product** will generate all reports required to comply with federal and state government regulations that require the reporting of chemical usage, storage, and disposal.

K. E. Wiegers, *Software Requirements*. Microsoft Press, 2003.

V&S Document: 2. Scope and limitations

1. **Business requirements**
   - 1.1 Background
   - 1.2 Business opportunity
   - 1.3 Business objectives
   - 1.4 Success metrics
   - 1.5 Vision statement
   - 1.6 Business risks
   - 1.7 Business assumptions and dependencies
2. **Scope and limitations**
   - 2.1 Major features
   - 2.2 Scope of initial release
   - 2.3 Scope of subsequent releases
   - 2.4 Limitations and exclusions
3. **Business context**
   - 3.1 Stakeholder profiles
   - 3.2 Project priorities
   - 3.3 Deployment considerations

1. Major features
Name or number each of the new product's major features or user capabilities in a unique, persisten way, emphasizing those features that distinguish it from previous or competing products. Giving features a unique label permits tracing it to individual user requirements, functional requirements, and other system elements.

2. Scope of initial release
Summarize the major features that are planned for inclusion in the initial relase of the product. Describe the quality characteristics that will let the product provide the intended benefits to its various user classes.

3. Scope of Subsequent Releases
Indicate which features will be deferred and the desired timing of later releases. You can also improve system performance, reliability, and other quality characteristics as the product matures.

4. Limitations and exclusions
Define the boundary between what's in and what's out in order to manage scope creep and customer expectations.

K. E. Wiegers, *Software Requirements*. Microsoft Press, 2003.

V&S Document: 3. Business context

1. **Business requirements**
   - 1.1 Background
   - 1.2 Business opportunity
   - 1.3 Business objectives
   - 1.4 Success metrics
   - 1.5 Vision statement
   - 1.6 Business risks
   - 1.7 Business assumptions and dependencies
2. **Scope and limitations**
   - 2.1 Major features
   - 2.2 Scope of initial release
   - 2.3 Scope of subsequent releases
   - 2.4 Limitations and exclusions
3. **Business context**
   - 3.1 Stakeholder profiles
   - 3.2 Project priorities
   - 3.3 Deployment considerations

3.1. Stakeholder Profiles

Describe different categories of customers and other key stakeholders for this project. Focus on different types of customers, target market segments, and the different user classes within those segments. Each stakeholder profile should include the following information:

- The major value or benefit that the stakeholder will receive from the product and how the product will generate high customer satisfaction
- Their likely attitudes toward the product
- Major features and characteristics of interest
- Any known constraints that must be accomondated

3.2. Project Priorities

To enable effective decision making, the stakeholders must agree on the project's priorities. One way to approach this is to consider the five dimensions of software project: *features, quality, schedule, cost, and staff*. Each dimension fits in one of the following three categories on any given project:

- **A constraint** A limiting factor within which the project manager must operate
- **A driver** A significant success objective with limited flexibility for adjustment
- **A degree of freedom** A factor that the project manager has some latitude to adjust and balance against the other dimensions.

3.3. Deployment considerations

Describe the environment in which the system will be used and define the vital availability, reliability, performance, and integrity requirements.

K. E. Wiegers, *Software Requirements*. Microsoft Press, 2003.

Context diagram

- A high-level diagram that defines the boundary between the system, or part of a system, and its environment, showing the entities that interact with it.
- It only contains one process node that generalizes the function of the entire system in relationship to external entities
- Used early in a project to get agreement on the scope under investigation
- Fosters clear and accurate communication among the project stakeholders

How to do it:

1. Draw a circle to represent the system and label it with the system name
2. Identify the external entities
3. Add information flows
4. After you draft other user requirements, verify these models against the context diagram, revising them as necessary

Notation:

- Circles are processes (product to be developed).
- Lines are data flows.
- Rectangles are external entities.