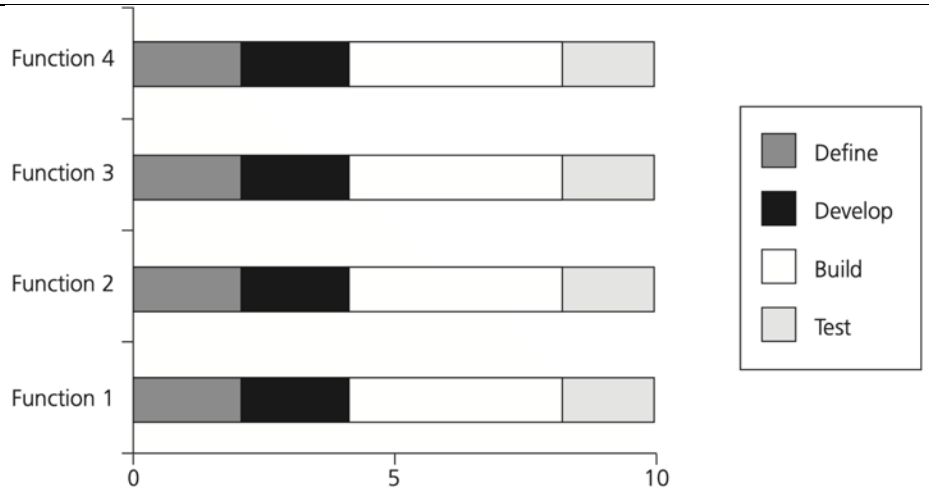
Table of Contents

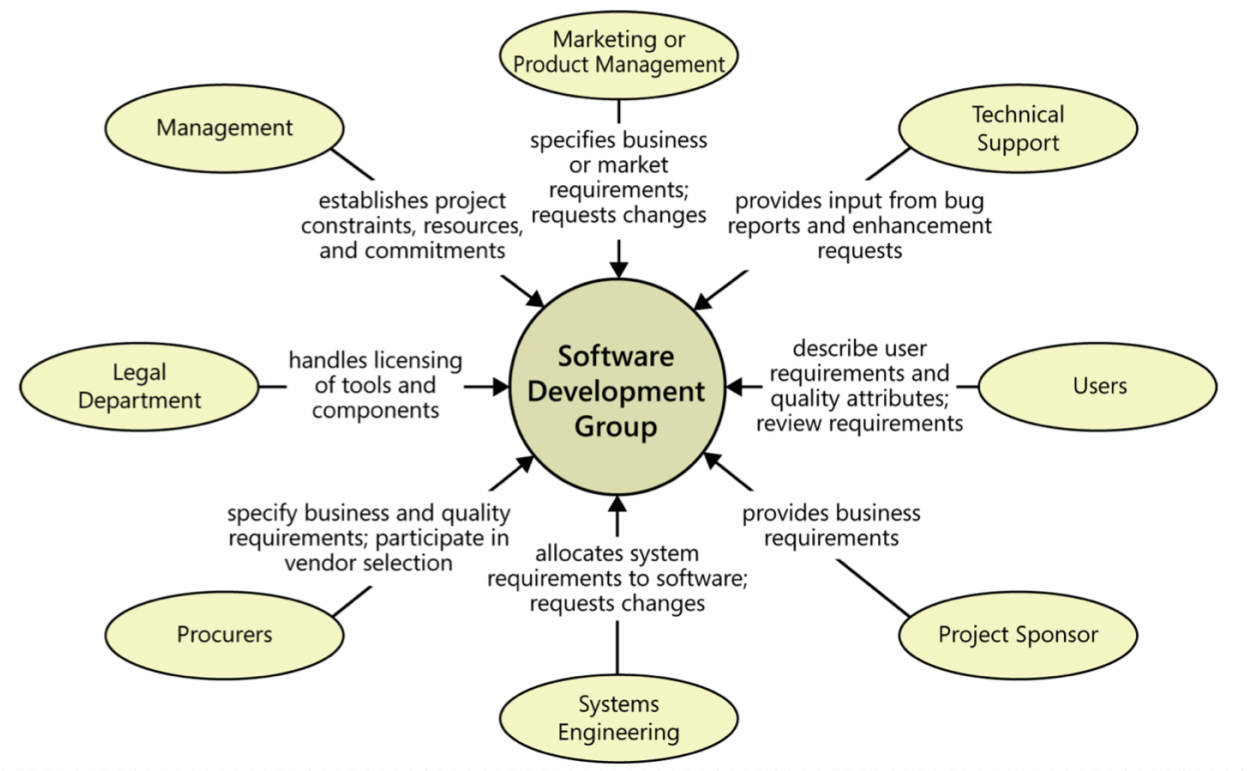
SDLC - Software development life cycle

Software development life cycle is a sequence of activities by which a software product is defined, designed, built, and verified.

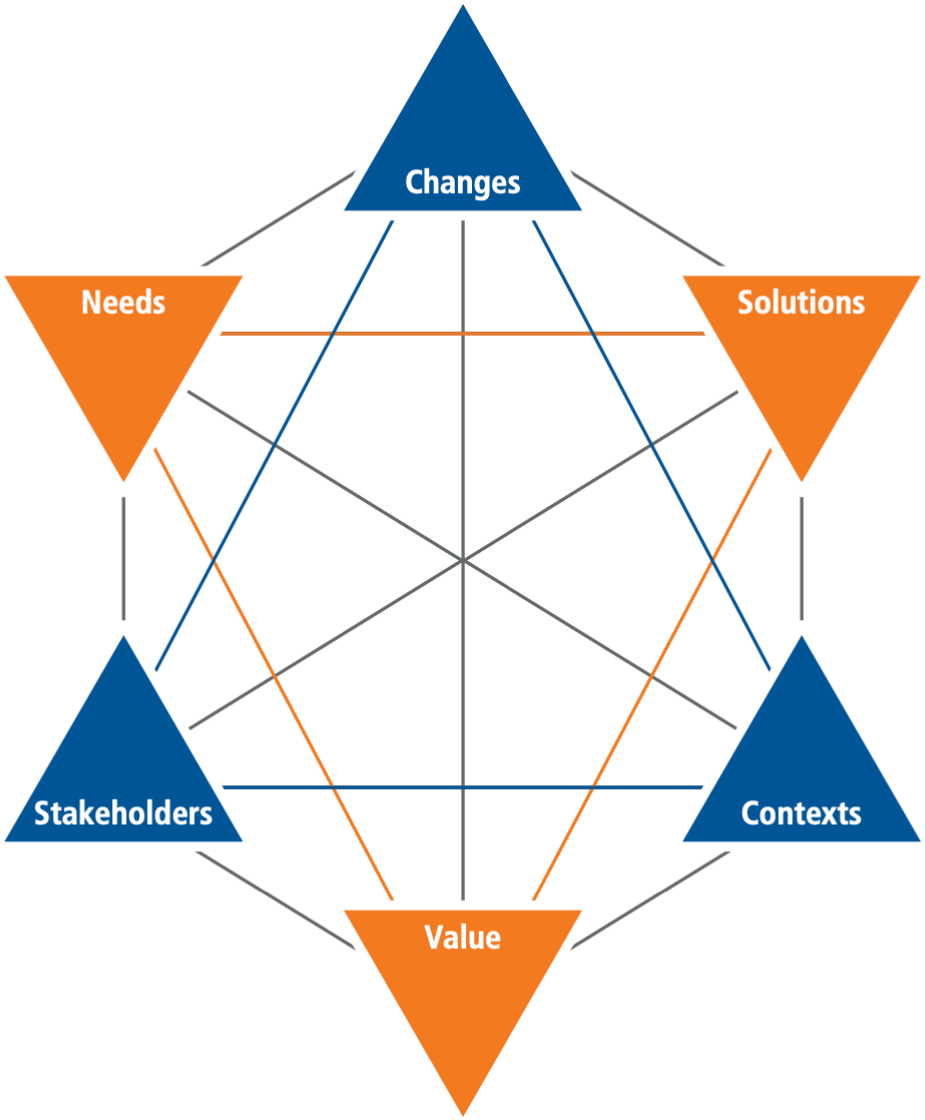
Software Development Models

Iterative & Incremental	
Iterative development model A development lifecycle where a project is broken into a usually large number of iterations. An iteration is a complete development loop resulting in a release (internal or external) of an executable product, a subset of the final product under development, which grows from iteration to iteration to become the final product.	
Waterfall	A model of the software development process in which the various activities of requirements, design, coding, testing, and deployment are performed sequentially with little overlap or iteration.
Incremental development model A development lifecycle where a project is broken into a series of increments, each of which delivers a portion of the functionality in the overall project requirements. The requirements are prioritized and delivered in priority order in the appropriate increment. In some (but not all) versions of this lifecycle model, each subproject follows a 'mini V-model' with its own design, coding and testing phases	
Agile	A group of software development methodologies based on iterative incremental development, where requirements and solutions evolve through collaboration between self-organizing cross- functional teams

<table>
<tr><td>RAD</td><td colspan="2"> 
</td></tr>
<tr><td>SCRUM</td><td colspan="2">a framework for managing and controlling iterative projects where the product owner works with cross-functional teams to create a list of tasks to be done.
Scrum is based on the theory of empirical process control, which relies on transparency, inspection, & adaptation. Scrum Is Iterative & Incremental
- Transparency: Scrum Reviews Provide Transparency.
- Inspection: Scrum Reviews & Retrospectives Offer Inspection Opportunities.
- Adaptation: Scrum Teams Can Adapt the Product at the End of Every Sprint.</td></tr>
<tr><td></td><td>Iteration/ sprint</td><td>An uninterrupted development period, typically one to four weeks in duration, during which a development team implements a defined set of functionality selected from the product backlog or baselined requirements for the product</td></tr>
</table>


Marketing or Product Management
Management
Technical Support
specifies business or market requirements; requests changes
establishes project constraints, resources, and commitments
provides input from bug reports and enhancement requests
Software Development Group
Legal Department
handles licensing of tools and components
describe user requirements and quality attributes; review requirements
Users
specify business and quality requirements; participate in vendor selection
allocates system requirements to software; requests changes
provides business requirements
Procurers
Systems Engineering
Project Sponsor

Core concept



Change	The act of transformation in response to a need.	
	change control board	The group of people responsible for deciding to accept or reject proposed changes on a software project, including changes in requirements.
Need	A problem or opportunity to be addressed.	
Solution	All of the components delivered by a project to achieve a set of business objectives specified by an organization, including software, hardware, business processes, user manuals, and training.	
Stakeholder	An individual, group, or organization that is actively involved in a project, is affected by its process or outcome, or can influence its process or outcome.	
Value	The worth, importance, or usefulness of something to a stakeholder within a context.	

	Value can be tangible or intangible. Tangible value is directly measurable. Tangible value often has a significant monetary component. Intangible value is measured indirectly. Intangible value often has a significant motivational component, such as a company's reputation or employee morale.	
Context	The circumstances that influence, are influenced by, and provide understanding of the change.	
	context diagram	An **analysis model** that depicts a system at a high level of abstraction. The context diagram identifies objects outside the system that exchange data with the system, but it shows nothing about the system's internal structure or behavior.
	external entity	An object in a context diagram or a data flow diagram that represents a user class, actor, software system, or hardware device that is external to the system being described but interfaces to it in some fashion. Also called a *terminator*.

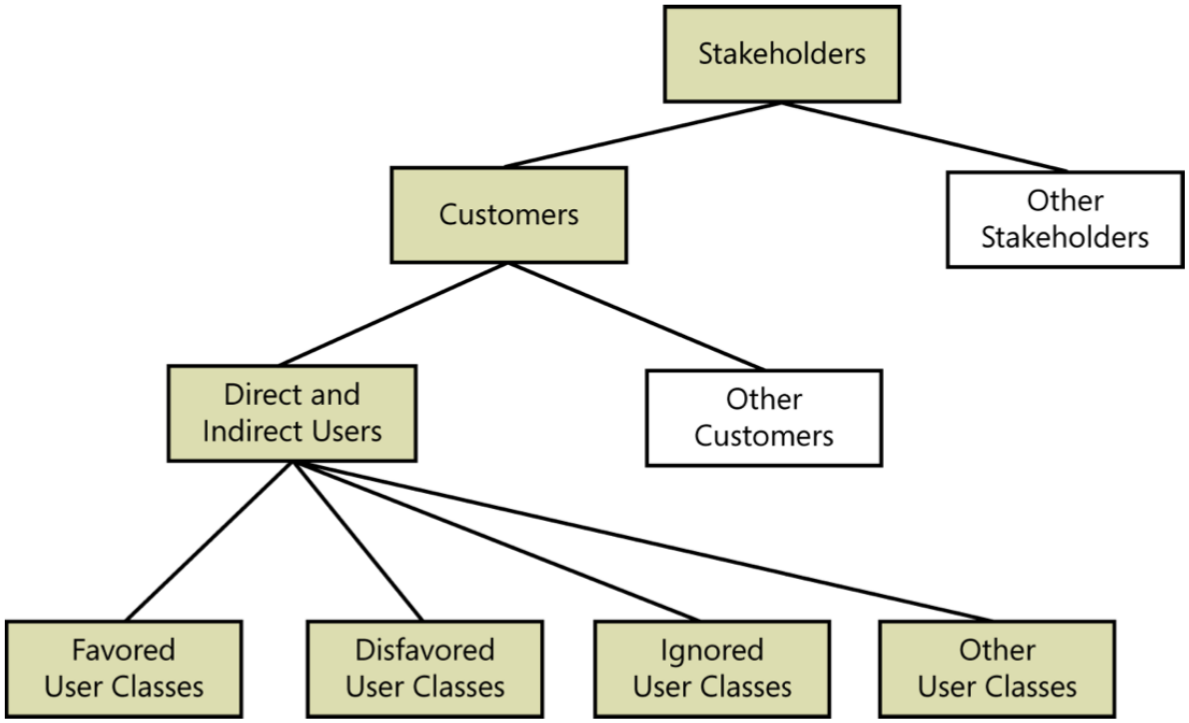
Key Terms

Business Analysis	is as the practice of enabling change in an enterprise by defining needs and recommending solutions that deliver value to stakeholders.	
Design	is a usable representation of a solution	
Project	A Project is a temporary, unique and progressive attempt or endeavor made to produce some kind of a tangible or intangible result (a unique product, service, benefit, competitive advantage, etc.). It usually includes a series of interrelated tasks that are planned for execution over a fixed period of time and within certain requirements and limitations such as cost, quality, performance, others.	
	Green-field project	A project in which new software or a new system is developed.
	Facilitator	A person who is responsible for planning and leading a group activity, such as a requirements elicitation workshop.
Organization	An autonomous group of people under the management of a single individual or board, that works towards common goals and objectives.	
Plan	is a proposal for doing or achieving something. Plans describe a set of events, the dependencies among the events, the expected sequence, the schedule, the results or outcomes, the materials and resources needed, and the stakeholders involved.	
Requirement	A statement of a customer need or objective, or of a condition or capability that a product must possess to satisfy such a need or objective. A property that a product must have to provide value to a stakeholder.	
	requirement attribute	Descriptive information about a requirement that enriches its definition beyond the statement of intended functionality. Example attribute types are origin, rationale, priority, owner, release number, and version number.
	requirement pattern	A systematic approach to specifying a particular type of requirement (includes templates, example, applicability,..)
	requirement specification	The product from the process of documenting a software application's requirements in a structured, shareable, and manageable form

	SRS	A collection of the functional and nonfunctional requirements for a software product.
	Requirement allocation	The process of apportioning system requirements among various architectural subsystems and components.
Risk	A condition that could cause some loss or otherwise threaten the success of a project.	
	mitigation plan	reduce the probability of impact of the identified risk.
	contingency plan	plan to control the impact as risk event looks like occurring
Domain	Sphere of knowledge that defines a common set of requirements or terminology	
Business domain	*Business Domain* refers to real-world aspects of your solution (e.g. Healthcare, Aviation, Finance, Military, Retail, etc). The business domain informs your *requirements* and *acceptance criteria* for the system; it can be suggestive of a very high-level form of segregation for different areas.	
	class	A description of a set of objects having common properties and behaviors, which typically correspond to real-world items (persons, places, or things) in the business or problem domain.
	Entity	An item in the business domain about which data is collected and stored.
	class diagram	An analysis model that shows a set of system or problem domain classes, their interfaces, and their relationships.
Technical domain	*Technical domain* refers to technologies used, including patterns and frameworks (e.g. ASP.NET/Ruby on rails, MVC Pattern, etc). These tend to inform specific design choices and architectures for applications or related groups of applications.	
Enterprise	An enterprise is a system of one or more organizations and the solutions they use to pursue a shared set of common goals.	
Product	Whatever ultimate deliverable a project is developing. In this book, product, application, system, and solution are used interchangeably.	
	product backlog	On an agile project, the prioritized list of work remaining for the project. A backlog can contain user stories, business processes, change requests, infrastructure development, and defect stories. Work items from the backlog are allocated to upcoming iterations based on their priority.
	product champion	A designated representative of a specific user class who supplies the user requirements for the group that he or she represents.
	product owner	A role, typically on an agile project team, that represents the customer and that is responsible for setting the product vision, providing project boundaries and constraints,

		prioritizing the contents of the product backlog, and making product decisions.
	system	A product that contains multiple software and/or hardware subsystems. Colloquially, *system* also is used interchangeably in this book with *application*, *product*, and *solution* to refer to whatever software-containing deliverable a team is building.
Vision	A statement that describes the strategic concept or the ultimate pur…	
Scope	The portion of the ultimate product vision that the current project will address. The scope draws the boundary between what's in and what's out for a project that creates a specific release or for a single development iteration.	
	scope creep	A condition in which the scope of a project continues to increase in an uncontrolled fashion throughout the development process.
	Gold plating	Unnecessary or excessively complex functionality that is specified or built into a product, sometimes without customer approval.
Vision and scope document	A collection of the business requirements for a new system, including business objectives, success criteria, a product vision statement, and a project scope description.	

Stakeholder Classification



Don’t overlook indirect user classes. They won’t use your application themselves, instead accessing its data or services through other applications or through reports. Your customer once removed is still your customer.

User persona	A persona is a description of a hypothetical, generic person who represents a group of users having similar characteristics and needs If we can design the system that meets that one person’s needs, all the expectations and needs of the whole class that this person represents will be satisfied • Create personas that truly represents the user class, based on the market, demographic and ethnographic research. • Use personas to understand requirements and design best user experiences to meet the needs of user communities.
Favored user	Favored user classes are those whose satisfaction is most closely aligned with achieving the project’s business objectives
Disfavored user	Disfavored user classes are groups who aren’t supposed to use the product for legal, security, or safety reasons; such as user impersonators, bots, internet agents etc.
Ignored user	Ignored user classes use the product but not specifically built to suit them
Other user classes	Other user classes are neither favoured, disfavoured or ignored but of equal importance when it comes to defining product requirements
Customer	A customer uses or may use products or services produced by the enterprise and may have contractual or moral rights that the enterprise is obliged to meet.
Business Analyst	The role on a project team that has primary responsibility for working with stakeholder representatives to elicit, analyze, specify, validate, and manage the project's requirements. Also called a *requirements analyst*, *system analyst*, *requirements engineer*, *requirements manager*, *business systems analyst*, and simply *analyst*. **BA’s tasks** • Define business requirements • Plan the requirements approach • Identify project stakeholders and user classes • Elicit requirements • Analyze requirements • Document requirements • Communicate requirements • Lead requirements validation • Facilitate requirements prioritization • Manage requirements

End user	End users are stakeholders who directly interact with the solution. End users can include all participants in a business process, or who use the product or solution. Also called user
domain subject matter expert	is any individual with in-depth knowledge of a topic relevant to the business need or solution scope; such as managers, process owners, legal staff, consultants, and others
implementation subject matter expert	is any stakeholder who has specialized knowledge regarding the implementation of one or more solution components; such as project librarian, change manager, configuration manager, solution architect, developer, database administrator, information architect, usability analyst, trainer, and organizational change consultant.
operational support	is responsible for the day-to-day management and maintenance of a system or product; such as operations analyst, product analyst, help desk, and release manager.
project manager	are responsible for managing the work required to deliver a solution that meets a business need, and for ensuring that the project's objectives are met while balancing the project factors including scope, budget, schedule, resources, quality, and risk
regulator	are responsible for the definition and enforcement of standards; such as government, auditor.
Sponsor	are responsible for initiating the effort to define a business need and develop a solution that meets that need
Tester / QC	are responsible for determining how to verify that the solution meets the requirements defined by the business analyst, as well as conducting the verification process. Also called quality controller
Supplier	is a stakeholder outside the boundary of a given organization or organizational unit. Suppliers provide products or services to the organization and may have contractual or moral rights and obligations that must be considered. Alternate roles are providers, vendors, and consultants.

Product vs. Project Manager

Product manager	**Project manager**
Works with outside stakeholders	Works with internal stakeholders
Helps to define the product vision	Helps teams execute on a shared vision
Outlines what success looks like	Outlines the plan for achieving success
Owns vision, marketing, ROI	Owns team backlog and fulfillment work
Works at a conceptual level	Involded in day-to-day activities

Resolving conflicting requirements

Don’t justify doing whatever any customer demands because “The customer is always right.” We all know the customer is not always right (Wiegers 2011). Sometimes, a customer is unreasonable, uninformed, or in a bad mood. The customer always has a point, though, and the software team must understand and respect that point.

Disagreement between	**How to resolve**

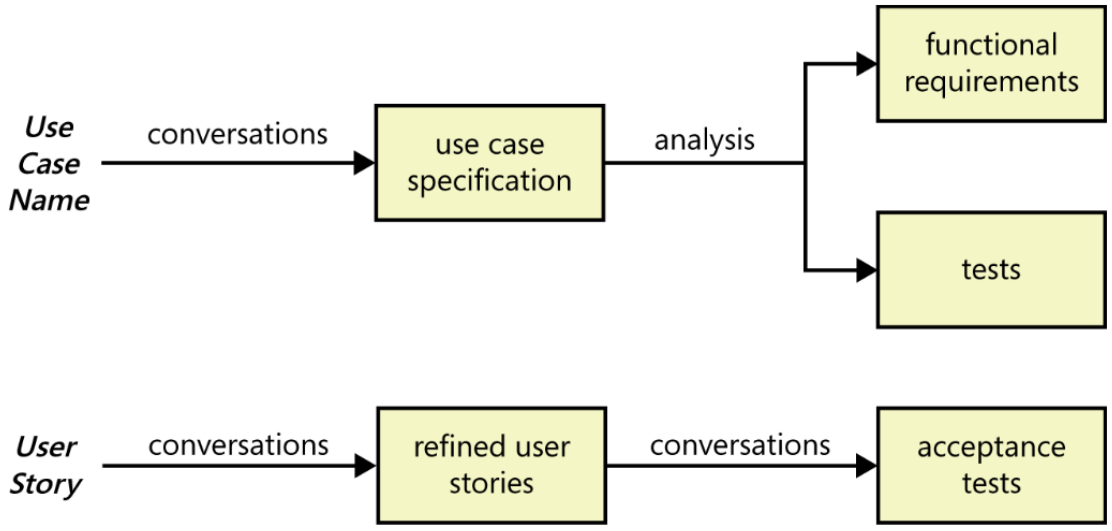
Individual users	Product champion or product owner decides
User classes	Favored user class gets preference
Market segments	Segment with greatest impact on business success gets preference
Corporate customers	Business objectives dictate direction
Users and user managers	Product owner or product champion for the user class decides
Development and customers	Customers get preference, but in alignment with business objectives
Development and marketing	Marketing gets preference

Requirements Classification

Business requirements	A set of information that describes a business need that leads to one or more projects to deliver a solution and the desired ultimate business outcomes. The business requirements include business opportunities, business objectives, success metrics, a vision statement, and scope and limitations.	
	business objective	A financial or nonfinancial business benefit that an organization expects to receive as a result of a project or some other initiative.
	business rule	A policy, guideline, standard, regulation, or computational formula that defines or constrains some aspect of the business.
Stakeholder Requirement	describe the needs of stakeholders that must be met in order to achieve the business requirements. They may serve as a bridge between business and solution requirements.	
Solution requirements	describe the capabilities and qualities of a solution that meets the stakeholder requirements. They provide the appropriate level of detail to allow for the development and implementation of the solution.	
	Functional	A description of a behavior that a software system will exhibit under specific conditions
	Non-Functional	do not relate directly to the behaviour of functionality of the solution, but rather describe conditions under which a solution must remain effective or **qualities** that a solution must have.
Transition requirements	describe the capabilities that the solution must have and the conditions the solution must meet to facilitate transition from the current state to the future state, but which are not needed once the change is complete. They are differentiated from other requirements types because they are of a temporary nature. Transition requirements address topics such as data conversion, training, and business continuity.	
User requirement	A goal or task that specific classes of users must be able to perform with a system, or a desired product attribute	

	represent user requirements	Use cases, user stories, and scenarios are common ways
	use case	A description of a set of logically related possible interactions between an actor and a system that results in an outcome that provides value to the actor
	use case diagram	An **analysis model** that identifies the actors who can interact with a system to accomplish valuable goals and the various use cases that each actor might be involved with
	user class	A group of users for a system who have similar characteristics and requirements for the system. Members of a user class function as actors when interacting with the system through use cases.
	User role / actor	A person performing a specific role, a software system, or a hardware device that interacts with a system to achieve a useful goal
	user story	A format to capture user requirements on agile projects in the form of one or two sentences that articulate a user need or describe a unit of desired functionality, as well as stating the benefit of the functionality to the user
	scenario	A description of a specific interaction between a user and a system to accomplish some goal. Alternatively, an instance of usage of the system, or a specific path through a use case.
system requirement	A high-level requirement for a product that contains multiple subsystems, which could be all software or software and hardware.	
External interface	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.	
	Feature list	The list of features that together makes a product.
Quality attributes	A nonfunctional requirement that describes a service or performance characteristic of a product. Types of quality attributes include usability, portability, maintainability, integrity, efficiency, reliability, and robustness. Quality attribute requirements describe the extent to which a software product must demonstrate desired characteristics.	
Product requirement	properties / attributes of a software system to be built	
Project requirement	expectations and deliverables that are not a part of the product the team implements, but that are necessary to the successful completion of the project as a whole	

User case vs. user story



User case approach

- Who (or what) is notified when something occurs within the system?
- Who (or what) provides information or services to the system?
- Who (or what) helps the system respond to and complete a task?

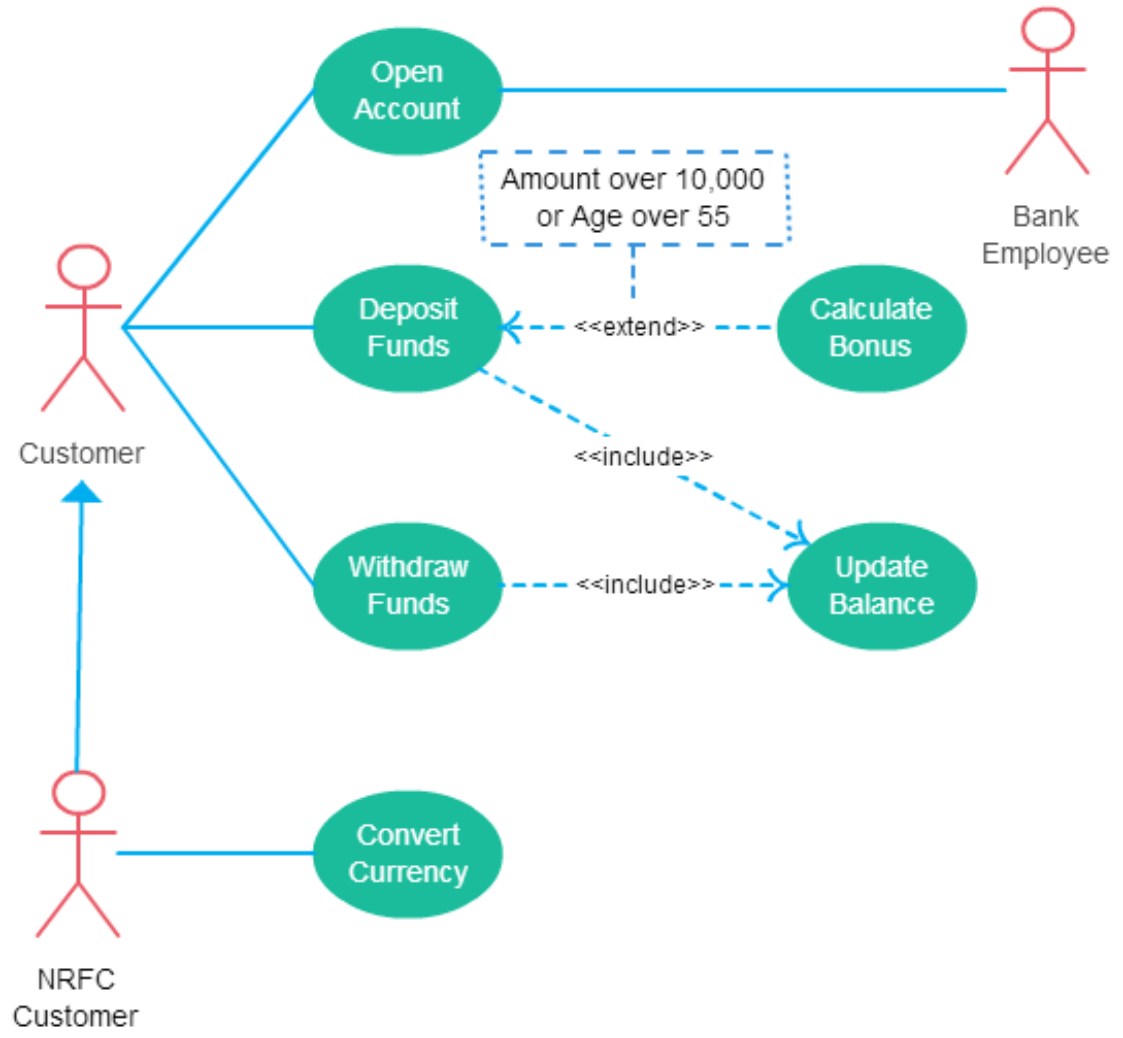
Note: Use cases and business rules are intertwined

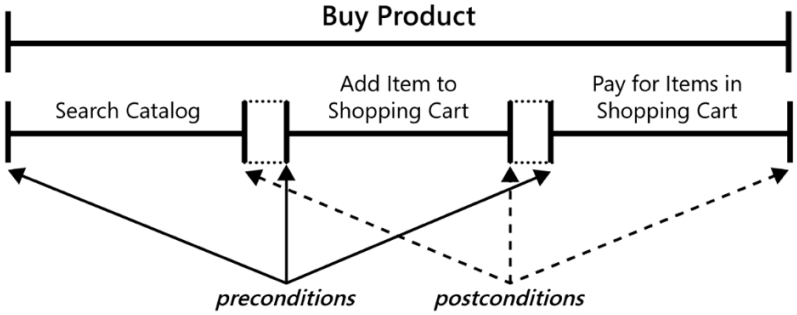
Identifying use cases

- Identify the actors first
- Create a specific scenario to illustrate each business process
- Using a business process description
- Identify the external events
- Use a CRUD analysis to identify data entities that require use cases to create, read, update, delete, or otherwise manipulate them
- Examine the context diagram

To void

- Too many use cases
- Highly complex use cases
- Including design in the use cases
- Including data definitions in the use cases
- Use cases that users don't understand



Precondition	A condition that must be satisfied or a state the system must be in before a use case can begin
postcondition	A condition that describes the state of a system after a use case is successfully completed. 
normal flow	The default sequence of steps in a use case, which leads to satisfying the use case's postconditions and letting the user achieve his goal. Also known

	as the normal course, main course, normal sequence, and main success scenario
alternative flow	A path through a use case that leads to success but that involves a variation from the normal flow in the specifics of the task or in the actor's interaction with the system.
Exception	A condition that can prevent a use case from concluding successfully. Unless some recovery mechanism is possible, the use case's postconditions are not reached and the actor's goal is not achieved.
extend relationship	A construct in which an alternative course in a use case interrupts the normal sequence of steps. The steps that the actor follows when executing the alternative course can be packaged into an extension use case that is invoked to complete the alternative.
include relationship	A construct in which several steps that recur in multiple use cases are factored out into a separate sub-use case, which the other use cases then invoke when needed.

User story approach

User stories are part of an agile approach that helps shift the focus from writing about requirements to talking about them. All agile user stories include a written sentence or two and, more importantly, a series of conversations about the desired functionality.

Format: As a < type of user >, I want < some goal > so that < some reason >

Note: Remember that the written part of an agile user story ("As a user, I want …") is incomplete until the discussions about that story occur

Detail can be added to user stories in two ways

- By splitting a user story into multiple, smaller user stories.
- By adding "conditions of satisfaction."

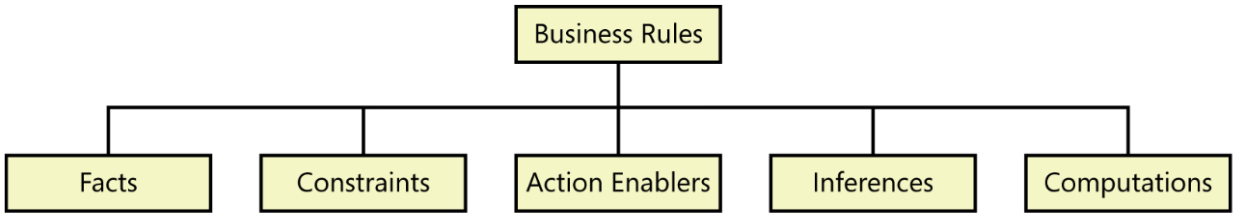
Product Backlog	On an agile project, the prioritized list of work remaining for the project. A backlog can contain user stories, business processes, change requests, infrastructure development, and defect stories. Work items from the backlog are allocated to upcoming iterations based on their priority.
Sprint	An uninterrupted development period, typically one to four weeks in duration, during which a development team implements a defined set of functionality selected from the product backlog or baselined requirements for the product.
Sprint Backlog	the subset of product backlog that a team targets to deliver during a sprint in order to accomplish the sprint goal and make progress toward a desired outcome.
Product Owner	A role, typically on an agile project team, that represents the customer and that is responsible for setting the product vision, providing project boundaries and constraints, prioritizing the contents of the product backlog, and making product decisions.

Epic	A user story on an agile project that is too large to implement in one development iteration. It is subdivided into smaller stories that each can be fully implemented in a single iteration.

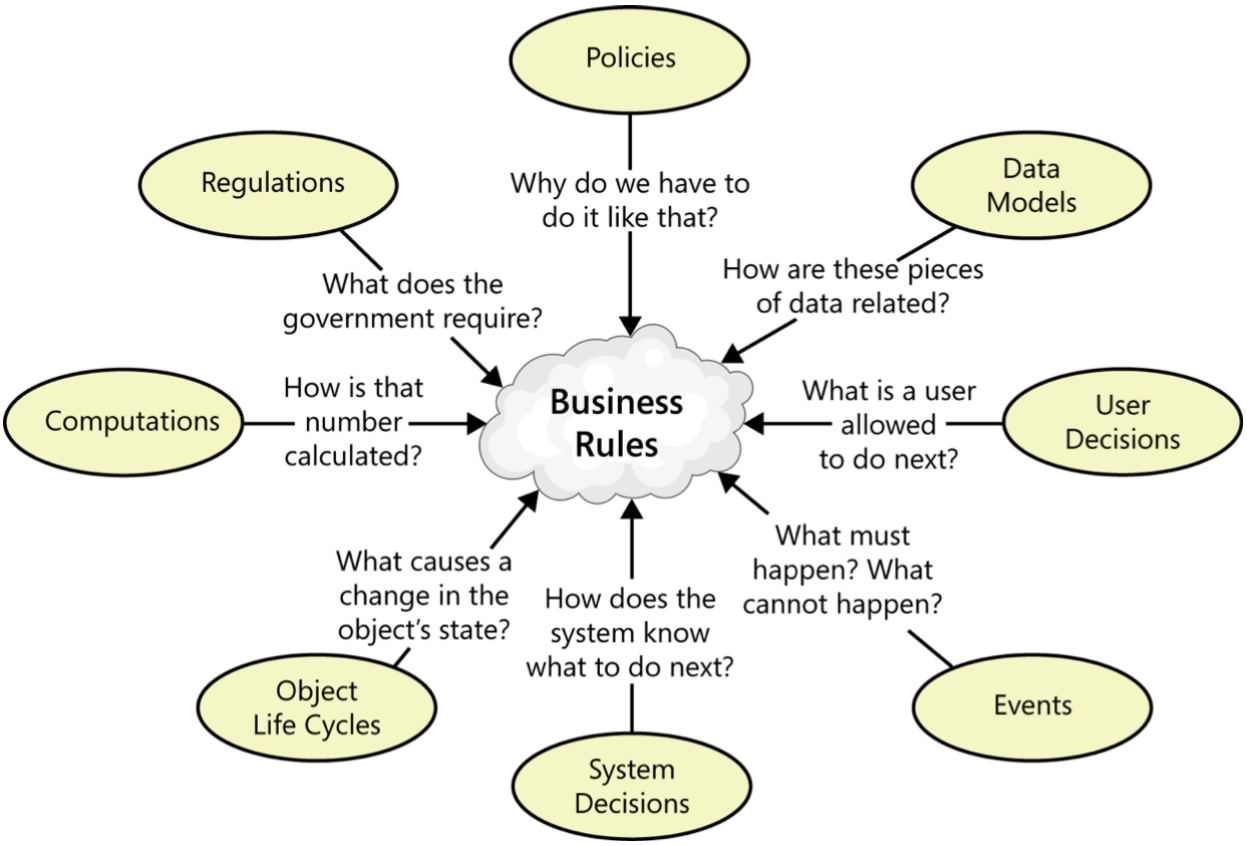
Business Rules

Business rules and their corresponding functional requirements sometimes look a lot alike. However, the rules are external statements of policy that must be enforced in software, thereby driving system functionality.

Business rule taxonomy



Discovering business rules by asking questions from different perspectives.



Term	Definition

business perspective	A business rule is guidance that there is an obligation concerning conduct, action, practice, or procedure within a particular activity or sphere.
information system perspective	A business rule is a statement that defines or constrains some aspect of the business. It is intended to assert business structure or to control or influence the behavior of the business.
Fact	Facts are simply statements that are true about the business at a specified point in time. A fact describes associations or relationships between important business terms.
Constrain	A restriction that is imposed on the choices available to the developer for the design and construction of a product. Other types of constraints can restrict the options available to project managers. Business rules often impose constraints on business operations and hence on software systems. E.g. Organizational policies, Government regulations, Industry standards
Action enablers	A rule that triggers some activity if specific conditions are true is an action enabler.
Inferences	Sometimes called inferred knowledge or a derived fact, an inference creates a new fact from other facts. Inferences are often written in the "if/then" pattern also found in action-enabling business rules, but the "then" clause of an inference simply provides a piece of knowledge, not an action to be taken.
Computations	that transform existing data into new data by using specific mathematical formulas or algorithms.
Atomic business rule	an atomic business rule is a statement that defines or constrains some aspect of the business, but it cannot be broken down or decomposed further into more detailed business rules. Each atomic business rule may be based on one or more business rule statements.

Non-functional Requirement

External quality	Definition
Usability	How well the system protects against injury or damage Usability requirements deal with ease of learning, ease of use, error avoidance and recovery, efficiency of interactions, and accessibility. The usability requirements specified here will help the user interface designer create the optimum user experience.
Performance	How quickly and predictably the system responds to user inputs or other events State specific performance requirements for various system operations.
Security	How well the system protects against unauthorized access to the application and its data Specify any requirements regarding security or privacy issues that restrict access to or use of the product. These could refer to physical, data, or software security.
Safety	Specify requirements that are concerned with possible loss, damage, or harm that could result from use of the product

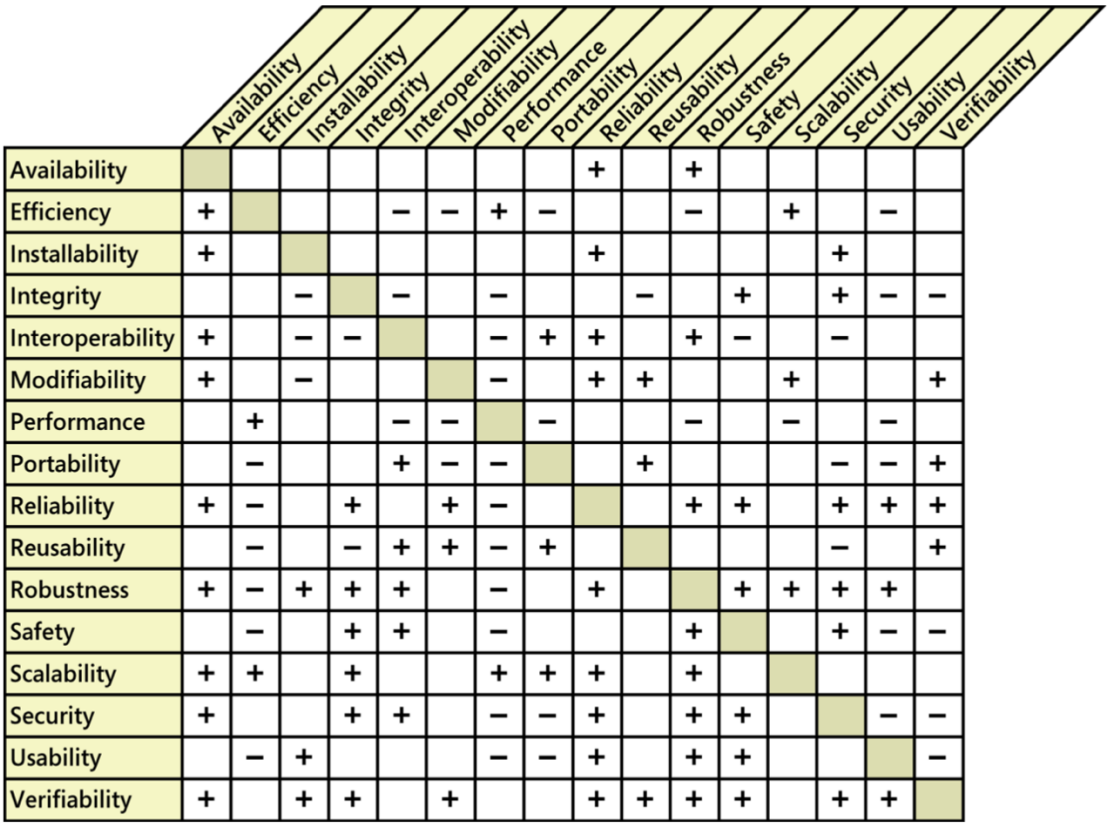
Availability	The extent to which the system's services are available when and where they are needed
Installability	How easy it is to correctly install, uninstall, and reinstall the application
Integrity	The extent to which the system protects against data inaccuracy and loss
Interoperability	How easily the system can interconnect and exchange data with other systems or components
Reliability	How long the system runs before experiencing a failure
Robustness	How well the system responds to unexpected operating conditions

Internal quality	**Definition**
Efficiency	How efficiently the system uses computer resources
Modifiability	How easy it is to maintain, change, enhance, and restructure the system
Portability	How easily the system can be made to work in other operating environments
Reusability	To what extent components can be used in other systems
Scalability	How easily the system can grow to handle more users, transactions, servers, or other extensions
Verifiability	How readily developers and testers can confirm that the software was implemented correctly

Specifying quality requirements with Planguage
A keyword-oriented language developed by Tom Gilb that enables precise and quantitative specification of requirements, particularly nonfunctional requirements.

- **TAG** Performance. Report. Response Time
- **AMBITION** Fast response time to generate accounting reports on the base user platform.
- **SCALE** Seconds of elapsed time between pressing the Enter key or clicking OK to request a report and the beginning of the display of the report.
- **METER** Stopwatch testing performed on 30 test reports that represent a defined usage operational profile for a field office accountant.
- **GOAL** No more than 8 seconds for 95 percent of reports.
- **STRETCH** No more than 2 seconds for predefined reports, 5 seconds for all reports.
- **WISH** No more than 1.5 seconds for all reports.
- **base user platform DEFINED** Quad-core processor, 8GB RAM, Windows 8, single user, at least 50 percent of system RAM and 70 percent of system CPU capacity free, network connection speed of at least 30 Mbps.

Quality attribute trade-offs



	Availability	Efficiency	Installability	Integrity	Interoperability	Modifiability	Performance	Portability	Reliability	Reusability	Robustness	Safety	Scalability	Security	Usability	Verifiability
Availability									+		+					
Efficiency	+				−	−	+	−			−		+		−	
Installability	+								+					+		
Integrity			−		−		−			−		+		+	−	−
Interoperability	+		−	−			−	+	+		+	−		−		
Modifiability	+		−				−		+	+			+			+
Performance		+			−	−		−			−		−		−	
Portability		−			+	−	−			+				−	−	+
Reliability	+	−		+		+	−				+	+		+	+	+
Reusability		−		−	+	+	−	+						−		+
Robustness	+	−	+	+	+		−		+			+	+	+	+	
Safety		−		+	+		−			+				+	−	−
Scalability	+	+		+			+	+	+		+					
Security	+			+	+		−	−	+		+	+			−	−
Usability		−	+				−	−	+		+	+				−
Verifiability	+		+	+		+			+	+	+	+		+	+	

Change Request

A change request is a proposal to alter a product or system, often brought up by the client or another team member. During a project, this can happen when a client wants to change or alter the agreed upon deliverables.

Process to deal with change request

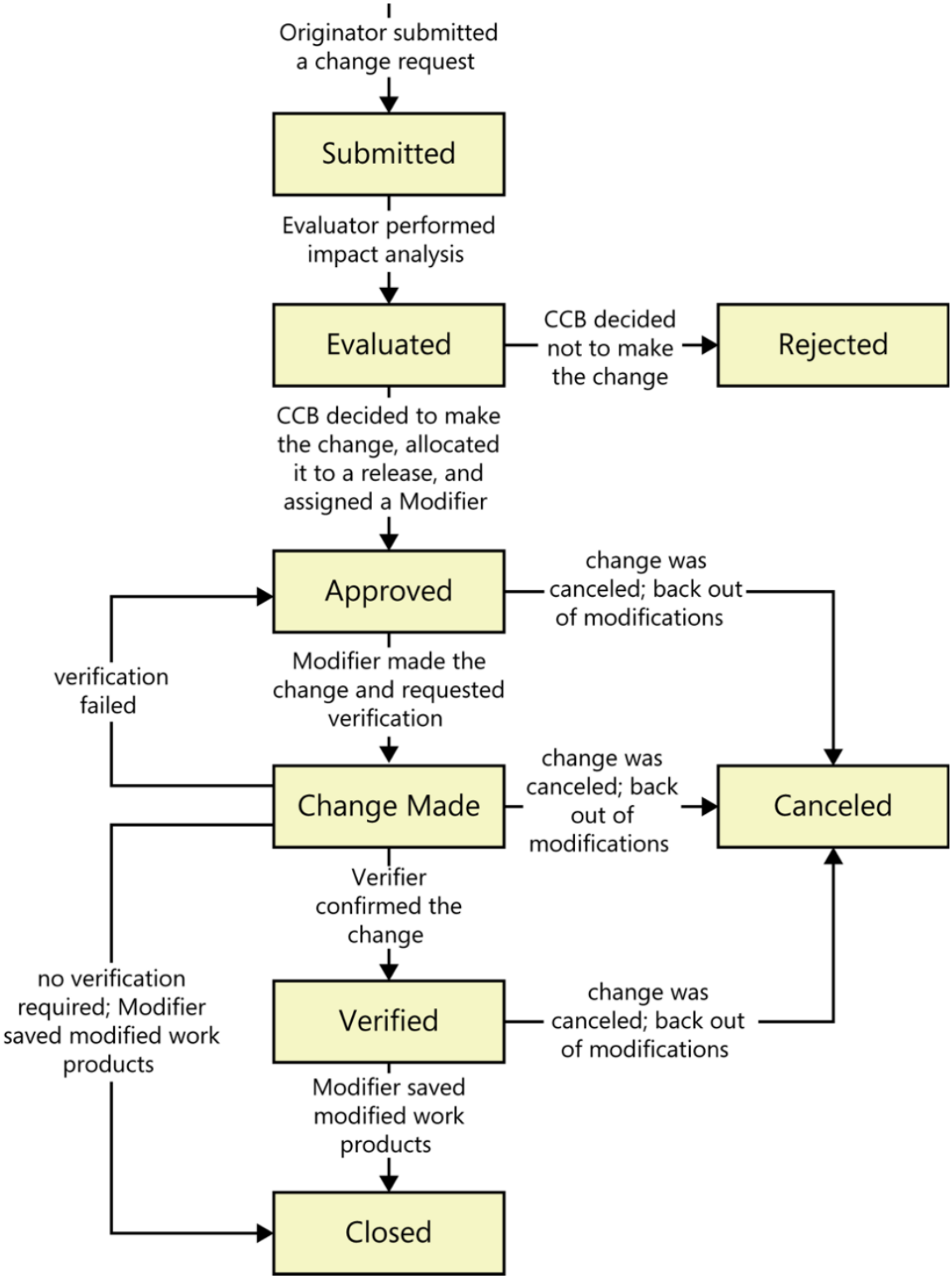
Change control process	Definition
Entry criteria	The basic entry criterion for your change control process is that a change request with all the necessary information has been received through an approved channel
Exit criteria	Satisfying the following exit criteria indicates that an execution of your change control process was properly completed: • The status of the request is Rejected, Closed, or Canceled. • All modified work products are updated and stored in the correct locations. • The relevant stakeholders have been notified of the change details and the status of the change request.

- **Request any supporting materials**: You want the person who is making the change to be as specific as possible. Ask that person to put their request in writing and provide any supporting materials that might be helpful.
- **Evaluate change request**:
    - Determine whether the change request is in inside or outside the scope
    - Have team assess the priority of the change request
- **Make change decision**: Approve or reject the change request
- **Implement the change**: Decide on a course of action going forward
- **Verify the change**: Requirements changes typically are verified through a peer review to ensure that modified deliverables correctly address all aspects of the change.
- **Change control status reporting:** Identify the charts and reports you'll use to summarize the contents of the change database. These charts might show the number of change requests in each state as a function of time, or trends in the average time that a change request is unresolved.

Suggested change request attributes

Item	Description
Change origin	Functional area that requested the change; possible groups include marketing, management, customer, development, and testing
Change request ID	Unique identifier assigned to the request
Change type	Type of change request, such as requirement change, proposed enhancement, or defect report
Date submitted	Date the Originator submitted the change request
Date updated	Date the change request was most recently modified
Description	Free-form text description of the change being requested
Implementation priority	The relative importance of making the change as determined by the CCB: low, medium, or high
Modifier	Person who is primarily responsible for implementing the change
Originator	Person who submitted this change request
Originator priority	The relative importance of making the change from the Originator's point of view: low, medium, or high
Planned release	Product release or iteration for which an approved change is scheduled
Project	Name of the project in which a change is being requested
Response	Free-form text of responses made to the change request; multiple responses can be made over time; do not change existing responses when entering a new one
Status	The current status of the change request, selected from the options in Figure 28-2
Title	One-line summary of the proposed change
Verifier	Person who is responsible for determining whether the change was made correctly

Change request status



Possible project roles in change-management activities

Role	Description and responsibilities
CCB Chair	Chairperson of the change control board; generally has final decision-making authority if the CCB does not reach agreement; identifies the Evaluator and the Modifier for each change request
CCB	The group that decides to approve or reject proposed changes for a specific project
Evaluator	Person whom the CCB Chair asks to analyze the impact of a proposed change
Modifier	Person who is responsible for making changes in a work product in response to an approved change request
Originator	Person who submits a new change request
Request Receiver	Person who initially receives newly submitted change requests
Verifier	Person who determines whether the change was made correctly

Defects, Incidents and Issues

Term	Definition
Issue	A defect, open question, or decision regarding a requirement. Examples include items flagged as TBD, pending decisions, information that is needed, and conflicts awaiting resolution.
Defect	
Defect checklist	To help reviewers look for typical kinds of errors in the products they review, develop a defect checklist for each type of requirements document your projects create.
Issue list	The issue log, sometimes also known as an issue register, is a project document where all issues that are negatively affecting the project are recorded and tracked.
Defect log	A log or database of all defects that were uncovered during the testing and maintenance phase of development.
Defect report	Defect report, also known as Bug Report, is a document that identifies and describes a defect detected by a tester.
Error (mistake)	A human action that produces an incorrect result
Defect (bug, fault)	A flaw in a component or system that can cause the component or system to fail to perform its required function, e.g. an incorrect statement or data definition. A defect, if encountered during execution, may cause a failure of the component or system.
Failure	Deviation of the component or system from its expected delivery, service or result. Failure is an event; fault is a state of the software, caused by an error
Incident	A condition that is different from what is expected, such a deviation from requirements or test cases

Issue types

Issue type	Description
Requirement question	Something isn't understood or decided about a requirement.
Missing requirement	Developers uncovered a missed requirement during design or implementation.
Incorrect requirement	A requirement was wrong. It should be corrected or removed.
Implementation question	As developers implement requirements, they have questions about how something should work or about design alternatives.
Duplicate requirement	Two or more equivalent requirements are discovered. Delete all but one of them.
Unneeded requirement	A requirement simply isn't needed anymore.

Defect checklist sample

Completeness

- ❑ Do the requirements address all known customer or system needs?
- ❑ Is any needed information missing? If so, is it identified as TBD?
- ❑ Have algorithms intrinsic to the functional requirements been defined?
- ❑ Are all external hardware, software, and communication interfaces defined?
- ❑ Is the expected behavior documented for all anticipated error conditions?
- ❑ Do the requirements provide an adequate basis for design and test?
- ❑ Is the implementation priority of each requirement included?
- ❑ Is each requirement in scope for the project, release, or iteration?

Correctness

- ❑ Do any requirements conflict with or duplicate other requirements?
- ❑ Is each requirement written in clear, concise, unambiguous, grammatically correct language?
- ❑ Is each requirement verifiable by testing, demonstration, review, or analysis?
- ❑ Are any specified error messages clear and meaningful?
- ❑ Are all requirements actually requirements, not solutions or constraints?
- ❑ Are the requirements technically feasible and implementable within known constraints?

Quality Attributes

- ❑ Are all usability, performance, security, and safety objectives properly specified?
- ❑ Are other quality attributes documented and quantified, with the acceptable trade-offs specified?
- ❑ Are the time-critical functions identified and timing criteria specified for them?
- ❑ Have internationalization and localization issues been adequately addressed?
- ❑ Are all of the quality requirements measurable?

Organization and Traceability

- ❑ Are the requirements organized in a logical and accessible way?
- ❑ Are all cross-references to other requirements and documents correct?
- ❑ Are all requirements written at a consistent and appropriate level of detail?
- ❑ Is each requirement uniquely and correctly labeled?
- ❑ Is each functional requirement traced back to its origin (e.g., system requirement, business rule)?

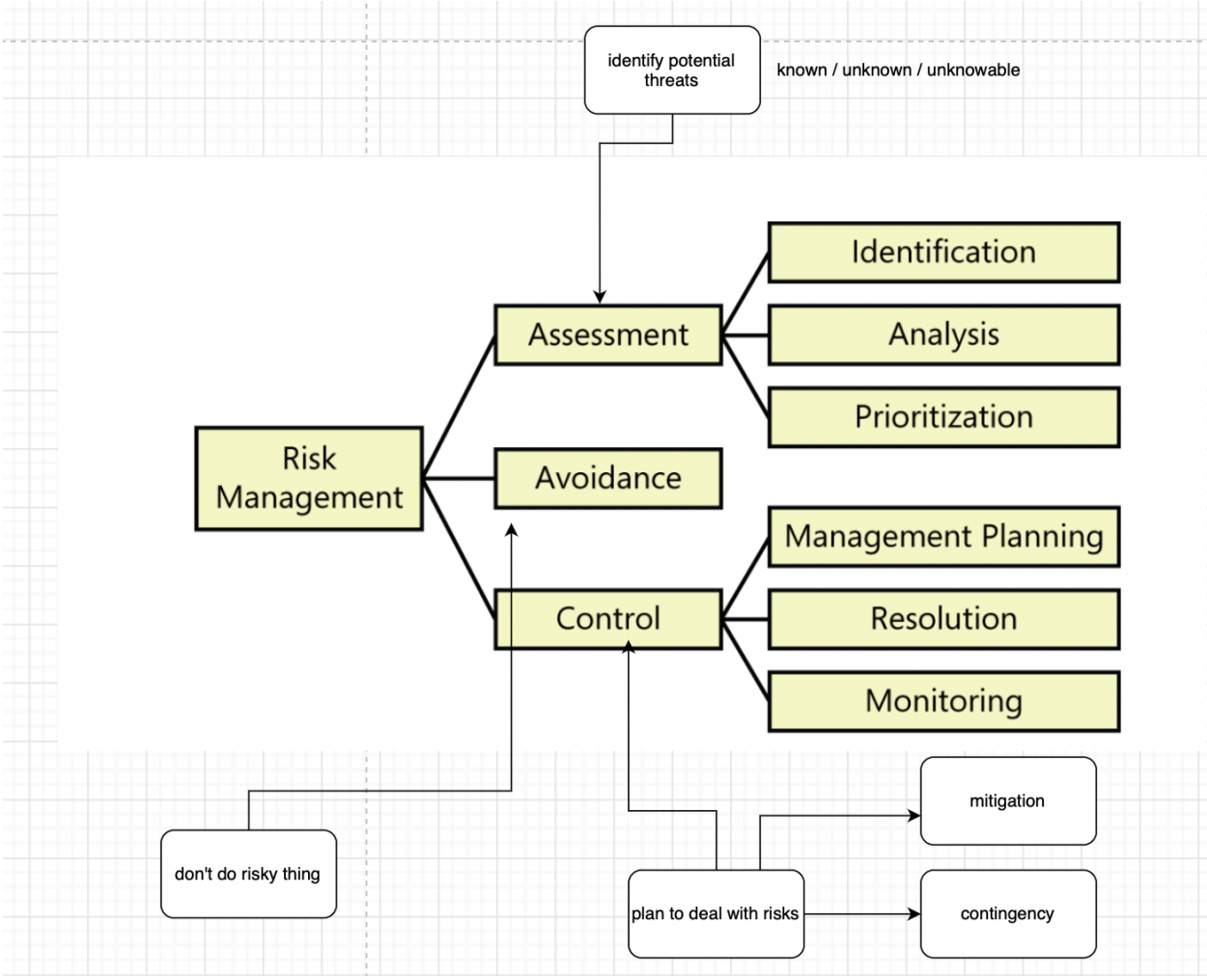
Other Issues

- ❑ Are any use cases or process flows missing?
- ❑ Are any alternative flows, exceptions, or other information missing from use cases?
- ❑ Are all of the business rules identified?
- ❑ Are there any missing visual models that would provide clarity or completeness?
- ❑ Are all necessary report specifications present and complete?

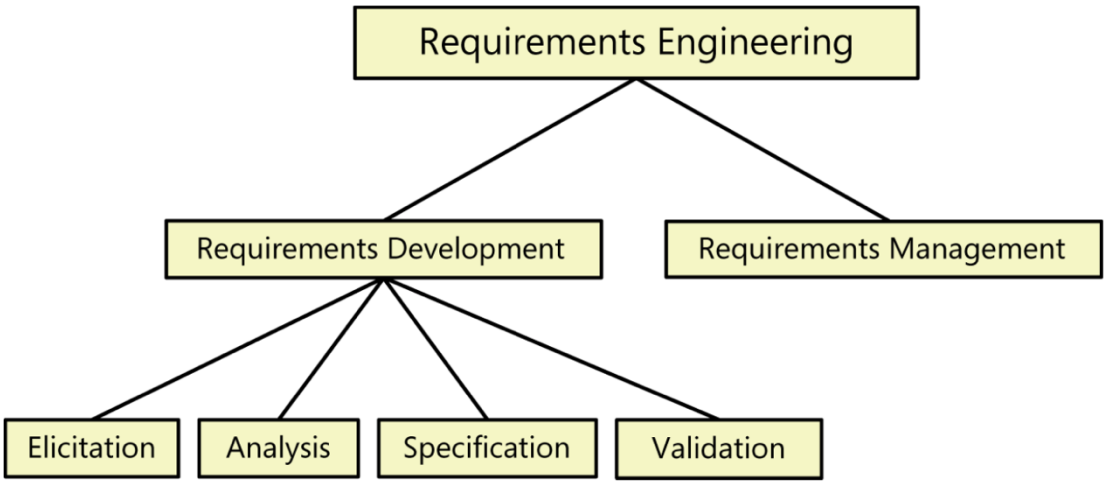
FIGURE 17-4 A defect checklist for reviewing requirements documents.

Risks

Risk is a factor that could result in future negative consequences; usually expressed as impact and likelihood.

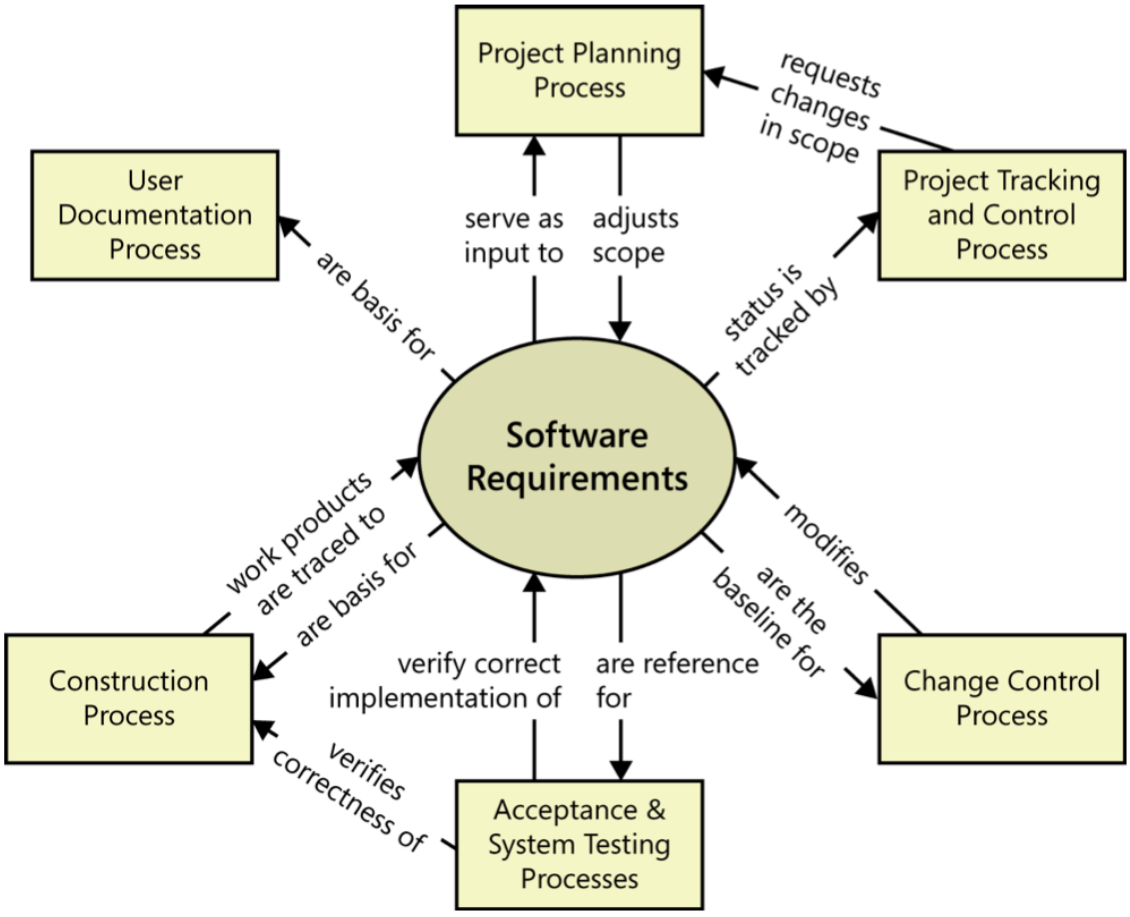


Requirement Engineering



Process

Relationship of requirements to other project processes.



Term	Definition
Procedure	A step-by-step description of a course of action to be taken to perform a specified activity, describing how the activity is to be accomplished.
Process	A sequence of activities performed for a particular purpose. A process description is a documented definition of those activities.
Process assets	Items such as templates, forms, checklists, policies, procedures, process descriptions, and sample work products that are collected to assist an organization's effective application of software development practices.
Process flow	The sequential steps of a business process or the operations of a proposed software system. Often represented by using an activity diagram, flowchart, swim-lane diagram, or other modeling notation.
Checklist	A list that enumerates activities, deliverables, or other items to be noted or verified. Checklists are memory joggers. They help ensure that busy people don't overlook important details.
Example	A representative of a specific type of work product. Accumulate and share good examples as your project teams create them.
Policy	A guiding principle that sets a management expectation of behaviors, actions, and deliverables. Processes should enable satisfaction of the policies.

Investing in requirements accelerates development

	Effort devoted to requirements	Schedule devoted to requirements
Faster projects	14%	17%
Slower projects	7%	9%

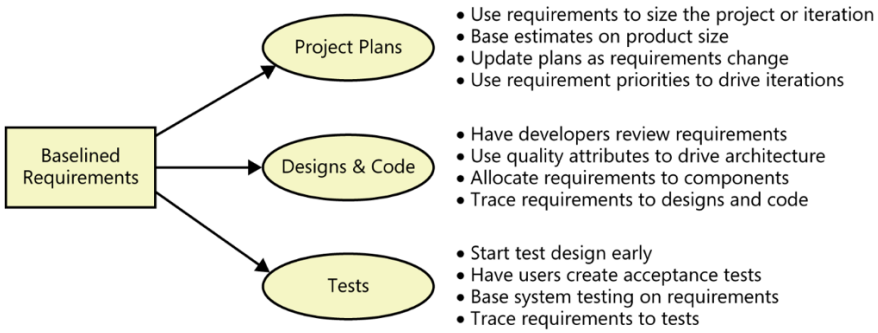
Good requirement

A good requirement should be

- **Complete**: Each requirement must contain all the information necessary for the reader to understand it
- **Correct**: Each requirement must accurately describe a capability that will meet some stakeholder's need and must clearly describe the functionality to be built
- **Feasible**: It must be possible to implement each requirement within the known capabilities and limitations of the system and its operating environment, as well as within project constraints of time, budget, and staff.
- **Necessary**: Each requirement should describe a capability that provides stakeholders with the anticipated business value, differentiates the product in the marketplace, or is required for conformance to an external standard, policy, or regulation. Every requirement should originate from a source that has the authority to provide requirements.
- **Prioritized**: Prioritize business requirements according to which are most important to achieving the desired value.
- **Verifiable:** testable
- **Unambiguous:** *"Comprehensible" is related to "unambiguous": readers must understand what each requirement is saying.*

- **Consistent**: don't conflict with other requirements of the same type or with higher-level business, user, or system requirements. If you don't resolve contradictions between requirements before diving into construction, the developers will have to deal with them
- **Traceable**
- **Modifiable**

Notes:
- Never going to get **perfect** requirement
- Only "**good enough**" (depend on context) – need to accept risk

Term	Definition
Ambiguous requirement	When multiple readers of a requirement arrive at different understandings of what it means
Gold plating	When a developer adds functionality that wasn't in the requirements specification (or was deemed out of scope) but which the developer believes "the users are just going to love."
Scope creep	A condition in which the scope of a project continues to increase in an uncontrolled fashion throughout the development process.
Baseline	a snapshot in time that represents the current agreed-upon, revired, approved set of requirements. 
Prioritization	The act of determining which requirements for a software product are the most important for achieving business success and the sequence in which requirements should be implemented.
Tracing	The process of defining logical links between one system element (user requirement, functional requirement, business rule, design component, code module, test, and the like) and another. Also called traceability.
TBD	Abbreviation for to be determined. TBD serves as a placeholder when you know you are missing some requirements information
Template	A pattern to be used as a guide for producing a complete document or other item.

Benefits from a high-quality requirements process
- Fewer defects in requirements and in the delivered product.

- Reduced development rework.
- Faster development and delivery.
- Fewer unnecessary and unused features
- Lower enhancement costs
- Fewer miscommunications.
- Reduced scope creep (uncontrolled growth in changes of scope)
- Reduced project chaos.
- Higher customer and team member satisfaction.
- Products that do what they're supposed to do.

Rights that customers can expect when it comes to requirements issues

- Mutual respect
- To expect BAs to speak your language
- To expect BAs to learn about your business and your objectives
- To expect BAs to record requirements in an appropriate form
- To receive explanations of requirements practices and deliverables
- To hear alternatives / ideas from BA
- Describe characteristics that will make the product easy to use
- To receive a system that meets your functional needs and quality expectations
- To hear about ways to adjust requirements to accelerate development through reuse
- You have the right to make changes in the requirements as the business evolves, as the team gathers more input from stakeholders, or as you think more carefully about what you need.

Reuse requirement

The act of using existing requirements knowledge in multiple systems that share some similar functionality.

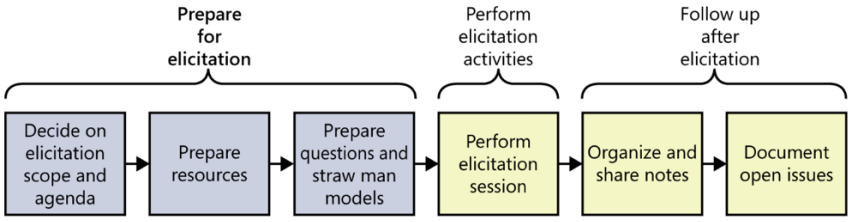
Following are several barriers you might encounter when it comes to reusing requirements.

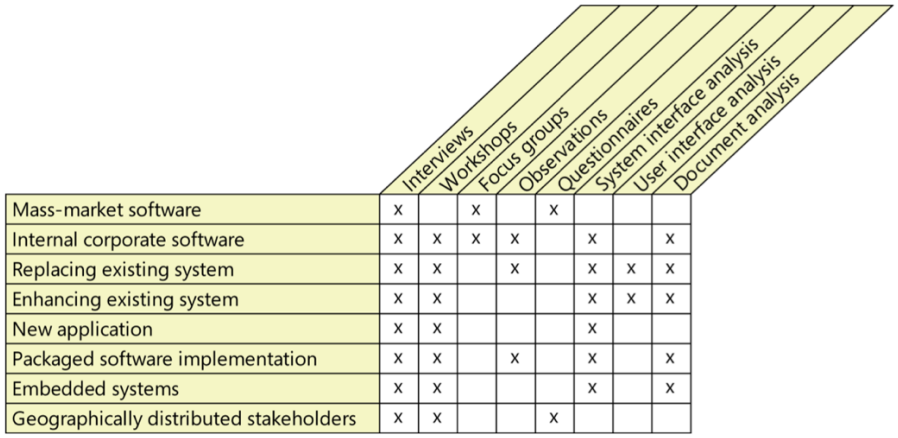
- **Missing or poor requirements**
- **NIH and NAH**: NIH means "not invented here.", NAH is "not applicable here,"
- **Inconsistent organization**
- **Project type**: Requirements that are tightly coupled to specific implementation environments or platforms are less likely to generate reusable requirements or to benefit from an existing pool of requirements knowledge.
- **Ownership**: You might not have the legal right to reuse

Requirement Elicitation

Requirement elicitation are all of the activities involved with discovering requirements
Notes: No assumed requirements and finding missing requirements. You'll likely never discover all of the requirements for your product, but nearly every software team can do a better job of requirements elicitation by applying the practices.

Term	Definition
Key actions	- Identify stakeholders, product's domain

<table>
<tr><td></td><td>
- understand user's tasks and goal, business objective
- research product's domain to understand product/project environment
- classify stakeholder
- Working with individuals who represent each user class to understand their functionality needs and their quality expectations


</td></tr>
<tr><td>Approaches (strategy)</td><td>
- usage-centric: emphasizes understanding and exploring user goals to derive the necessary system functionality
- Product-centric: focuses on defining features that you expect will lead to marketplace or business success
</td></tr>
<tr><td>Techniques</td><td>
- Interview
- Workshop
- Focus group
- Observation
- Questionnaire
- System interface analysis
- User interface analysis
- Document analysis


</td></tr>
<tr><td></td><td>-</td></tr>
</table>

Requirement Analysis

Major activities

- Decomposing high-level requirement into appropriate level requirements
- Classify requirement / mapping function – nonfunctional
- Understanding relative importance of quality attributes

- Prioritizing (discuss / negotiate with customer)
- Identify unnecessary requirement, gaps (discusses / negotiate with customer)

Gap analysis
A comparison of the current state to an alternative or potential state for a system, process, or other aspect of a business situation, to identify significant differences between them.

Root cause analysis
An activity that seeks to understand the underlying factors that contribute to an observed problem.

Requirement Specification
Storing / representing / modelling collected requirements – by writing or drawing (diagrams: ERD, use case diagrams, state, …)

Term	Definition
Assumption	A statement that is believed to be true in the absence of proof or definitive knowledge.
Dependency	As used in requirements specification, a reliance that a project has on a factor, event, or group outside its control.
Architecture	The structure of a system, including any software, hardware, and human components that make up the system, the interfaces and relationships between those components, and the component behaviors that are visible to other components.
External interface requirement	A description of a connection between a software system and a user, another software system, or a hardware device.
User interface (UI)	is the point of human-computer interaction and communication in a device.
Graphical user interface (GUI)	is a form of user interface that allows users to interact with electronic devices through graphical icons and audio indicator such as primary notation, instead of text-based user interfaces, typed command labels or text navigation.
Software interfaces	Describe the connections between this product and other software components (identified by name and version), including other applications, databases, operating systems, tools, libraries, websites, and integrated commercial components.
Hardware interfaces	Describe the characteristics of each interface between the software components and hardware components, if any, of the system.
Communications interfaces	State the requirements for any communication functions the product will use, including email, web browser, network protocols, and electronic forms.

Prototype

Term	Definition

Prototype	A partial, preliminary, or possible implementation of a software system. Used to explore and validate requirements and design approaches. Types of prototypes are evolutionary and throwaway; paper and electronic; and mock-up and proof-of-concept.	
	paper prototype	A nonexecutable mock-up of a software system's user interface using inexpensive, low-tech screen sketches.
	evolutionary prototype	A fully functional prototype created as a skeleton or an initial increment of the final product, which is fleshed out and extended incrementally as requirements become clear and ready for implementation.
	mock-up	A partial or possible representation of a user interface for a software system. Used to evaluate usability and to assess the completeness and correctness of requirements. Could be executable or could be in the form of a paper prototype. Also called a **horizontal prototype.**
	proof of concept	A prototype that implements a portion of a software-containing system that slices through multiple layers of the architecture. Used to evaluate technical feasibility and performance. Also called a **vertical prototype**.
	throwaway prototype	A prototype that is created with the intent of discarding it after it has served its purpose of clarifying and validating requirements and/or design alternatives

UML
An abbreviation for the Unified Modeling Language, which describes a set of standard notations for creating various visual models of systems, particularly for object-oriented software development.

Ecosystem map
An analysis model that shows a set of systems that interact with each other and the nature of their relationships. Unlike a context diagram, an ecosystem map shows systems that have a relationship even if there is no direct interface between them.

Context diagram
An analysis model that depicts a system at a high level of abstraction. The context diagram identifies objects outside the system that exchange data with the system, but it shows nothing about the system's internal structure or behavior.

Dialog map
An analysis model that depicts a user interface architecture, showing the display elements and the navigations permitted between them.

Notes: Dialog maps look a bit like flowcharts, but they serve a different purpose. A flowchart shows the processing steps and decision points involved in a process sequence, but not the UI screens you encounter along the way

Decision table and Decision tree

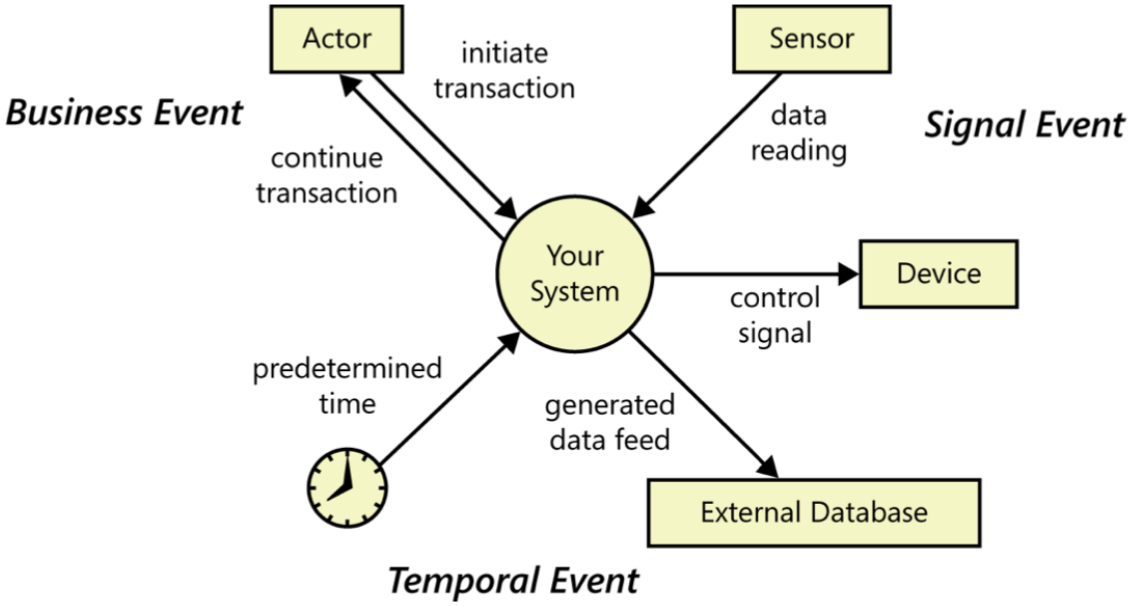
Decision tables and decision trees are two alternative techniques for representing what the system should do when complex logic and decisions come into play.

Decision rule	An agreed-upon way by which a body of people arrives at a decision.
Decision table	An analysis model in the form of a matrix that shows all combinations of values for a set of conditions and indicates the expected system action in response to each combination.
Decision tree	An analysis model that visually depicts the actions a system takes in response to specific combinations of a set of conditions.

Event-response table

A list of the external or time-triggered events that could affect the system and a description of how the system is to respond to each event.

Event	A trigger or stimulus that takes place in a system's environment that leads to a system response, such as a functional behavior or a change in state
Business event	A business event is an action by a human user that stimulates a dialog with the software, as when the user initiates a use case. The event-response sequences correspond to the steps in a use case or swim-lane diagram.
Signal event	A signal event is registered when the system receives a control signal, data reading, or interrupt from an external hardware device or another software system, such as when a switch closes, a voltage changes, another application requests a service, or a user swipes his finger on a tablet’s screen.
Temporal event	A temporal event is time-triggered, as when the computer’s clock reaches a specified time (say, to launch an automatic data export operation at midnight) or when a preset duration has passed since a previous event (as in a system that logs the temperature read by a sensor every 10 seconds).



Activity diagram
An analysis model that depicts a process flow proceeding from one activity to another. Similar to a flowchart.

Class diagram
An analysis model that shows a set of system or problem domain classes, their interfaces, and their relationships.

Data dictionary
A collection of definitions for the data elements and data structures that are relevant to the problem domain.

Dashboard report
A screen display or printed report that uses multiple textual and/or graphical representations of data to provide a consolidated, multidimensional view of what is going on in an organization or a process.

Feature tree
An analysis model that depicts the features planned for a product in a hierarchical tree, showing up to two levels of subfeatures beneath each main feature.

Flowchart
An analysis model that shows the processing steps and decision points in the logic of a process. Similar to an activity diagram.

Data flow diagram
An analysis model that depicts the processes, data stores, external entities, and flows among them that characterize the behavior of data flowing through business processes or software systems.

Swimlane diagram
An analysis model that shows the sequential steps of a business process flow or the operations of a proposed software system. The process is subdivided into visual components called lanes, which show the systems or actors that execute the steps.

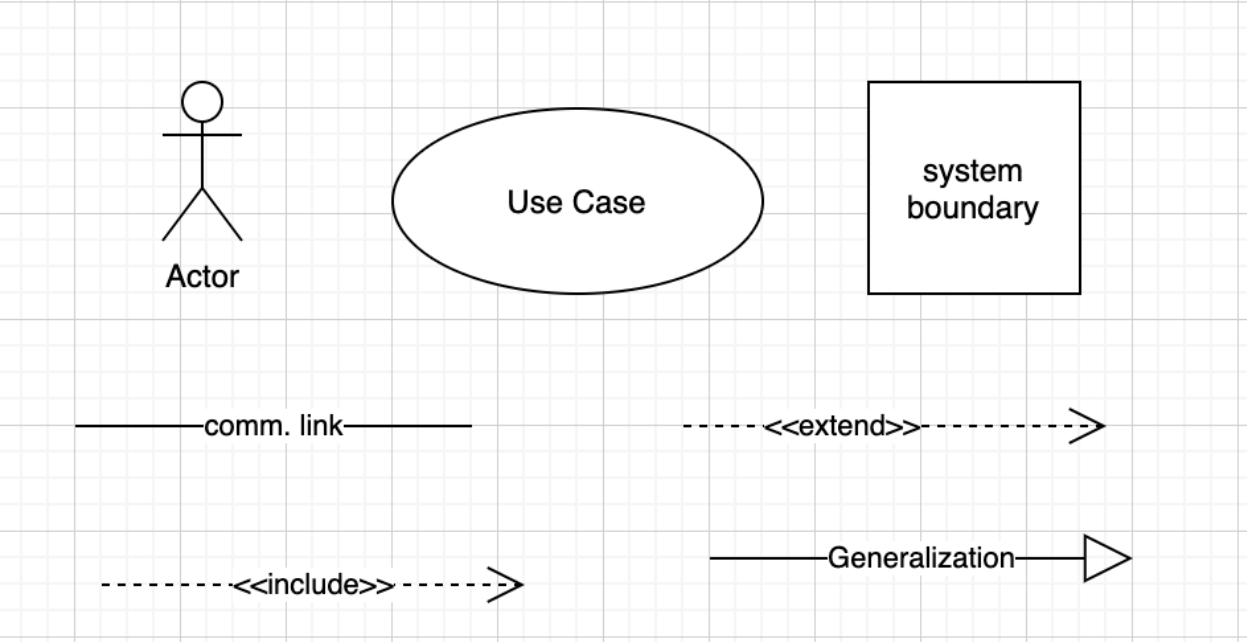
Entity-relationship diagram
An analysis model that identifies the logical relationships between pairs of entities. Used for modeling data.

State machine diagram
An analysis model that shows the sequence of states that an object in a system goes through during its lifetime in response to specific events that take place, or that shows the possible states of the system as a whole. Similar to a state-transition diagram.

State table
An analysis model that shows in matrix form the various states that a system, or an object in the system, can be in, and which of the possible transitions between states are allowed.

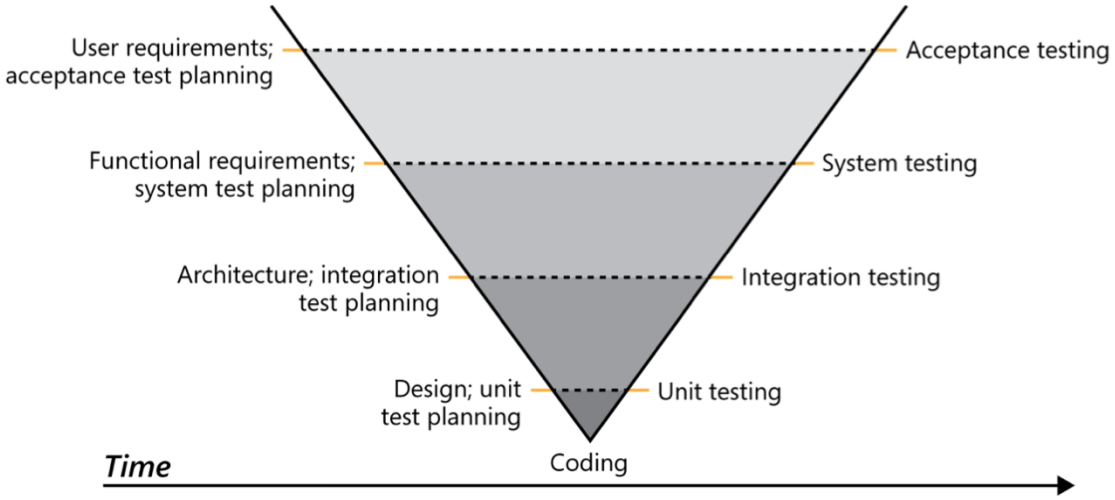
State-transition diagram
An analysis model that visually depicts the various states in which a system or an object in the system can exist, the permitted transitions that can take place between states, and the conditions and/or events that trigger each transition. Similar to a state machine or state-chart diagram.

Use case diagram
An analysis model that identifies the actors who can interact with a system to accomplish valuable goals and the various use cases that each actor might be involved with.



extend relationship	A construct in which an alternative course in a use case interrupts the normal sequence of steps. The steps that the actor follows when executing the alternative course can be packaged into an extension use case that is invoked to complete the alternative.
include relationship	A construct in which several steps that recur in multiple use cases are factored out into a separate sub-use case, which the other use cases then invoke when needed.
Generalization	refers to the relationship which can exist between two use cases and which shows that one use case (child) inherits the structure, behavior, and relationships of another actor (parent).

Requirement Validation



Main activities

- Define / develop acceptance test / acceptance criteria
- Test & review requirement's specification / user's requirement

Term	Definition
Acceptance criteria	Conditions that a software product must satisfy to be accepted by a user, customer, or other stakeholder.
Acceptance test	A test that evaluates anticipated usage scenarios to determine the software's acceptability. Used in agile development both to express details about a user story and to determine whether a user story is fully and correctly implemented.
validation	The process of evaluating a project deliverable to determine whether it satisfies customer needs. Often stated as "Are we building the right product?"
verification	The process of evaluating a project deliverable to determine whether it satisfies the specifications on which it was based. Often stated as "Are we building the product right?"
Review	Reviewing requirements is a powerful technique for identifying ambiguous or unverifiable requirements, requirements that aren't defined clearly enough for design to begin, and other problems.
Informal review	Informal reviews are useful for educating other people about the product and collecting unstructured feedback
Peer review	An activity in which one or more persons other than the author of a work product examine that product with the intent of finding defects and improvement opportunities.
Peer desk-check	in which you ask one colleague to look over your work product.
Pass-around	in which you invite several colleagues to examine a deliverable concurrently.
Walkthrough	during which the author describes a deliverable and solicits comments on it.
Formal review	A review characterized by documented procedures and requirements, e.g. inspection. Typical formal review steps 1. Planning 2. Kick-off 3. Preparation 4. Review meeting 5. Rework 6. Follow-up
Inspection	A type of formal peer review that involves a trained team of individuals who follow a well-defined process to examine a work product carefully for defects.
Planning	1. Defining the review criteria. 2. Selecting the personnel. 3. Allocating roles.

	4. Defining the **entry** and exit criteria for more formal review types (e.g. inspections). 5. Selecting which parts of documents to review. 6. Checking entry criteria (for more formal review types).
Kick-off	2 elements 1. Distributing documents. 2. Explaining the objectives, process and documents to the participants.
Preparation	2 elements 1. Preparing for the review meeting by reviewing the document(s). 2. Noting potential defects, questions and comments.
Review meeting	Examination, evaluation, and recording of results: 1. Discussing or logging, with documented results or minutes (for more formal review types). 2. Noting defects, making recommendations regarding handling the defects, making decisions about the defects. 3. Examining, evaluating and recording issues during any physical meetings or tracking any group electronic communications.
Rework	1. Fixing defects found (typically done by the author). 2. Recording updated status of defects (in formal reviews).
Follow-up	1. Checking that defects have been addressed. 2. Gathering metrics. 3. Checking exit criteria (for more formal review types)
Moderator (inspection leader)	The leader and main person responsible for an inspection or other review process.
Reviewer (inspector)	The person involved in the review that identifies and describes anomalies in the product or project under review. Reviewers can be chosen to represent different viewpoints and roles in the review process.
Author	As the writer of the document under review. The author's task is to illuminate unclear areas and to understand the defects found.
Manager	The manager decides on the execution of reviews, allocates time in project schedules and determines whether review process objectives have been met
Scribe (recorder)	The person who records each defect mentioned and any suggestions for process improvement during a review meeting, on a logging form. The scribe should ensure that the logging form is readable and understandable.
Entry criteria	The set of generic and specific conditions for permitting a process to go forward with a defined task, e.g. test phase. The purpose of entry criteria is to prevent a task from starting which would entail more (wasted) effort compared to the effort needed to remove the failed entry criteria.
Exit criteria	Your inspection process should define the exit criteria that must be satisfied before the moderator declares the full inspection process

	(not just the meeting) complete. Here are some possible exit criteria for requirements inspections: • All issues raised during the inspection have been addressed. • Any changes made in the requirements and related work products were made correctly. • All open issues have been resolved, or each open issue's resolution process, target date, and owner have been documented.
Retrospective	A review in which project participants reflect on the project's activities and outcomes with the intent of identifying ways to make the next project be even more successful.

Requirement Management

The process of working with a defined set of requirements throughout the product's development process and its operational life. Includes tracking requirements status, managing changes to requirements, controlling versions of requirements specifications, and tracing individual requirements to other requirements and system elements.

Traceability

Term	Definition
Requirement traceability matrix	A table that depicts logical links between individual functional requirements and other system artifacts, including other functional requirements, user requirements, business requirements, architecture and design elements, code modules, tests, and business rules.
Tracing	The process of defining logical links between one system element (user requirement, functional requirement, business rule, design component, code module, test, and the like) and another.

Measurement

Term	Definition
Function point	A measure of software size, based on the number and complexity of internal logical files, external interface files, external inputs, outputs, and queries.
Work product	Any interim or final deliverable created for a software project.
WBS	Work breakdown structure • **Step 1** − Create WBS by breaking down the test project into small pieces. • **Step 2** − Divide modules into sub-modules. • **Step 3** − Divide sub-modules further into functionalities. • **Step 4** − Divide functionalities into sub-functionalities. • **Step 5** − Review all the testing requirements to make sure they are added in WBS. • **Step 6** − Figure out the number of tasks your team needs to complete. • **Step 7** − Estimate the effort for each task. • **Step 8** − Estimate the duration of each task.

Analogous Estimating	is a technique which uses the values of parameters from historical data as the basis for estimating similar parameter for a future activity. • Parameters examples: Scope, cost, and duration. • Measures of scale examples − Size, weight, and complexity.
Three-point Estimating	**Step 1** − Arrive at the WBS. **Step 2** − For each task, find three values − most optimistic estimate (O), a most likely estimate (M), and a pessimistic estimate (L). **Step 3** - Calculate estimate, standard deviation Test Estimate = (O + M + L)/3 Where, • O = best case scenario in which nothing goes wrong and all conditions are optimal • M = most likely duration and there may be some problem but most of the things will go right. • L = worst case scenario where everything goes wrong
Program Evaluation and Review Technique (PERT)	**Step 1** − Arrive at the WBS. **Step 2** − For each task, find three values − most optimistic estimate (O), a most likely estimate (M), and a pessimistic estimate (L). **Step 3** - Calculate estimate, standard deviation Test Estimate = (O + (4 × M) + L)/6 Where, • O = best case scenario in which nothing goes wrong and all conditions are optimal • M = most likely duration and there may be some problem but most of the things will go right. • L = worst case scenario where everything goes wrong
FPA	Function Point Analysis
UCP	Use-Case Points (UCP) is a software estimation technique used to measure the software size with use cases.

Requirement status

Status	Definition
Proposed	The requirement has been requested by an authorized source.
In Progress	A business analyst is actively working on crafting the requirement.
Drafted	The initial version of the requirement has been written.
Approved	The requirement has been analyzed, its impact on the project has been estimated, and it has been allocated to the baseline for a specific release. The key stakeholders have agreed to incorporate the requirement, and the software development group has committed to implement it.
Implemented	The code that implements the requirement has been designed, written, and unit tested. The requirement has been traced to the pertinent design and code elements. The software that implemented the requirement is now ready for testing, review, or other verification.
Verified	The requirement has satisfied its acceptance criteria, meaning that the correct functioning of the implemented requirement has been confirmed. The requirement has been traced to pertinent tests. It is now considered complete.
Deferred	An approved requirement is now planned for implementation in a later release.
Deleted	An approved requirement has been removed from the baseline. Include an explanation of why and by whom the decision was made to delete it.
Rejected	The requirement was proposed but was never approved and is not planned for implementation in any upcoming release. Include an explanation of why and by whom the decision was made to reject it.

The process improvement cycle