Business Analysis Training

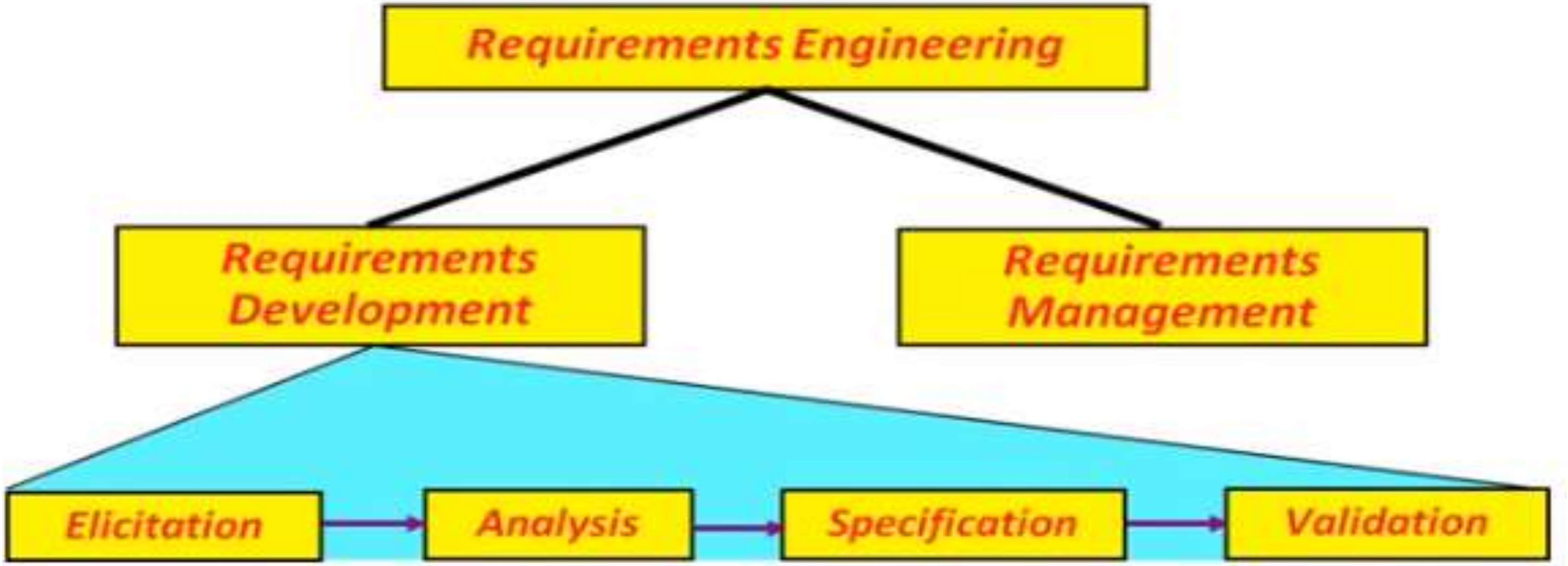
Requirement Management

mindsmapped

Agenda

- Requirements Management
  - Requirement Prioritization
  - MoSCoW Analysis
  - Time Boxing
  - Voting Technique
  - Verifying and Validating Requirements
    - Verifying Requirements
    - Validate Requirements
    - Key Requirements Management Practices
    - The Requirements Baseline
    - Requirements Version Management
    - Requirements Change Control
    - Impact Analysis of Requirements
    - Requirements Attributes
    - Requirements status tracking
    - Requirements Traceability
    - Requirements Traceability Matrix

Components of Requirements Engineering



Requirement Prioritization

- Prioritization plays a key role in our daily lives since we have a number of tasks to be performed in the various roles we play. Likewise as a business analyst we should be able to prioritize the requirements we gather from the client. There are various techniques to do so such as MoSCoW Analysis, Timeboxing or Budgeting, Voting, Decision Analysis and Risk Analysis.

MoSCoW Analysis

The requirements can be categorized into the following:

- **Must**: These are the most crucial requirements Without these the final solution will be incomplete.
- **Should**: These are also crucial requirements but there could be workarounds to satisfy these.
- **Could**: These are desirable or "nice to have" requirements. If time permits these will be included in the final solution.
- **Would/Won't**: These will be included in the next release or will be omitted completely.

Time Boxing

Time boxing which is also known as budgeting. There are three approaches in this.

All in: The group assigns a duration of time or cost to implement each requirement. We start with all requirements in the box and remove one-by-one based on the deadlines and budget.

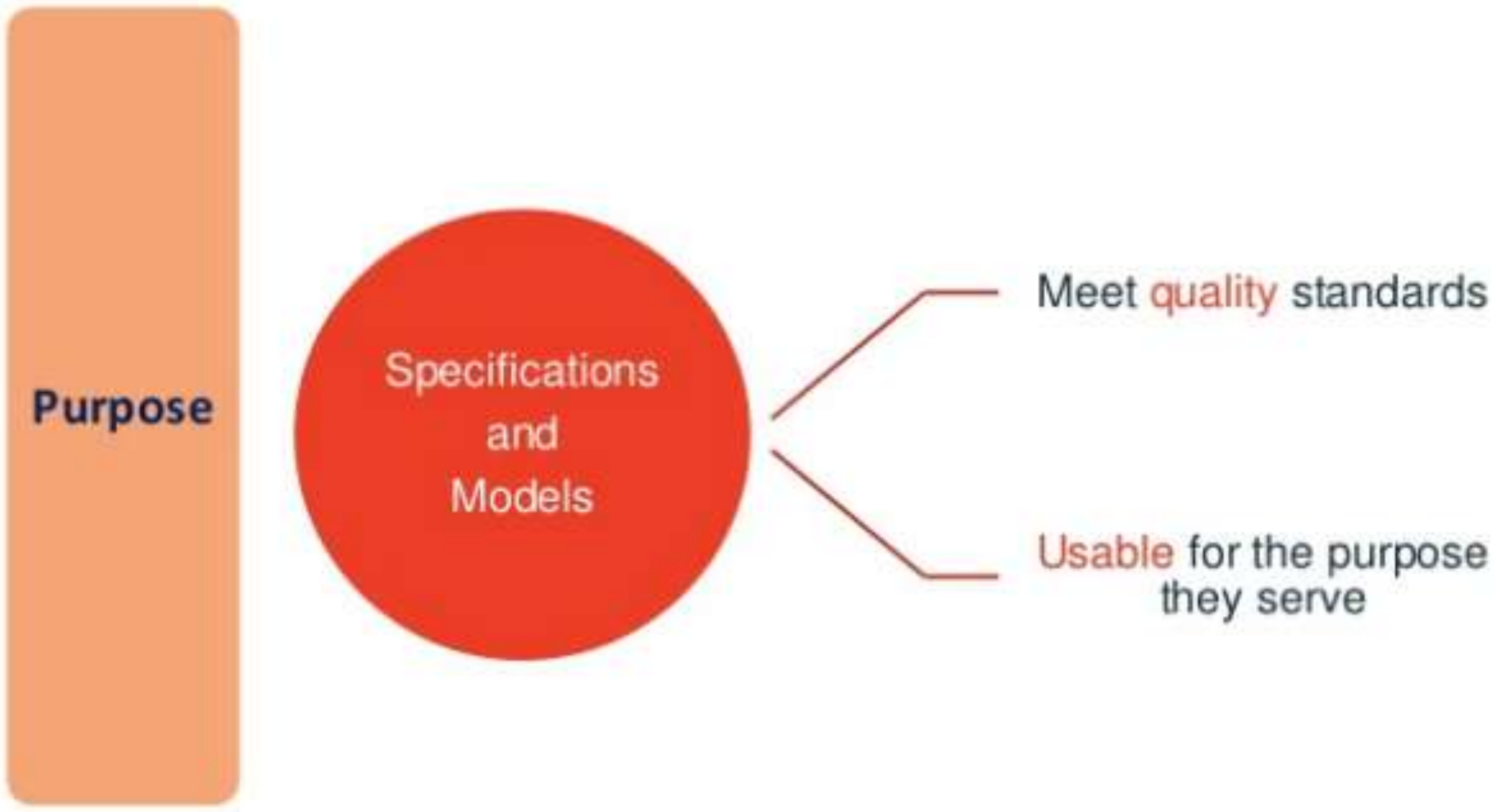
- **All out:** The group assigns the duration of time or cost to implement each requirement. All requirements are out of the box and we add these one-by-one until the cost/time limit is reached.
- **Selective:** This is a balanced approach. Here we identify the high priority requirements and then add/remove those one-by-one to meet the scheduled time/budget limits.

Voting Technique

- A set requirements are distributed to a group of stakeholders and each of them gives a vote to prioritize the distributed requirements.
- We can conclude by stating that one of the key skills a good business analyst should have is prioritization of tasks as it would help in minimizing overheads and maximizing the resource utilization.

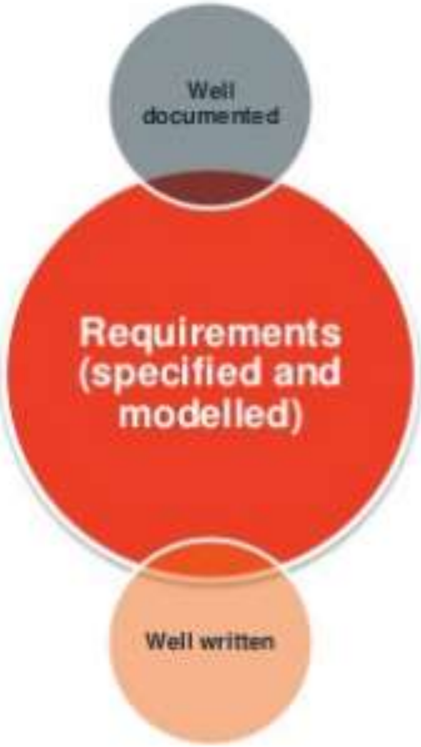
Verifying and Validating Requirements

Verifying Requirements



The purpose of Verifying Requirements is to ensure that requirements and designs specifications and models meet quality standards and are usable for the purpose they serve.

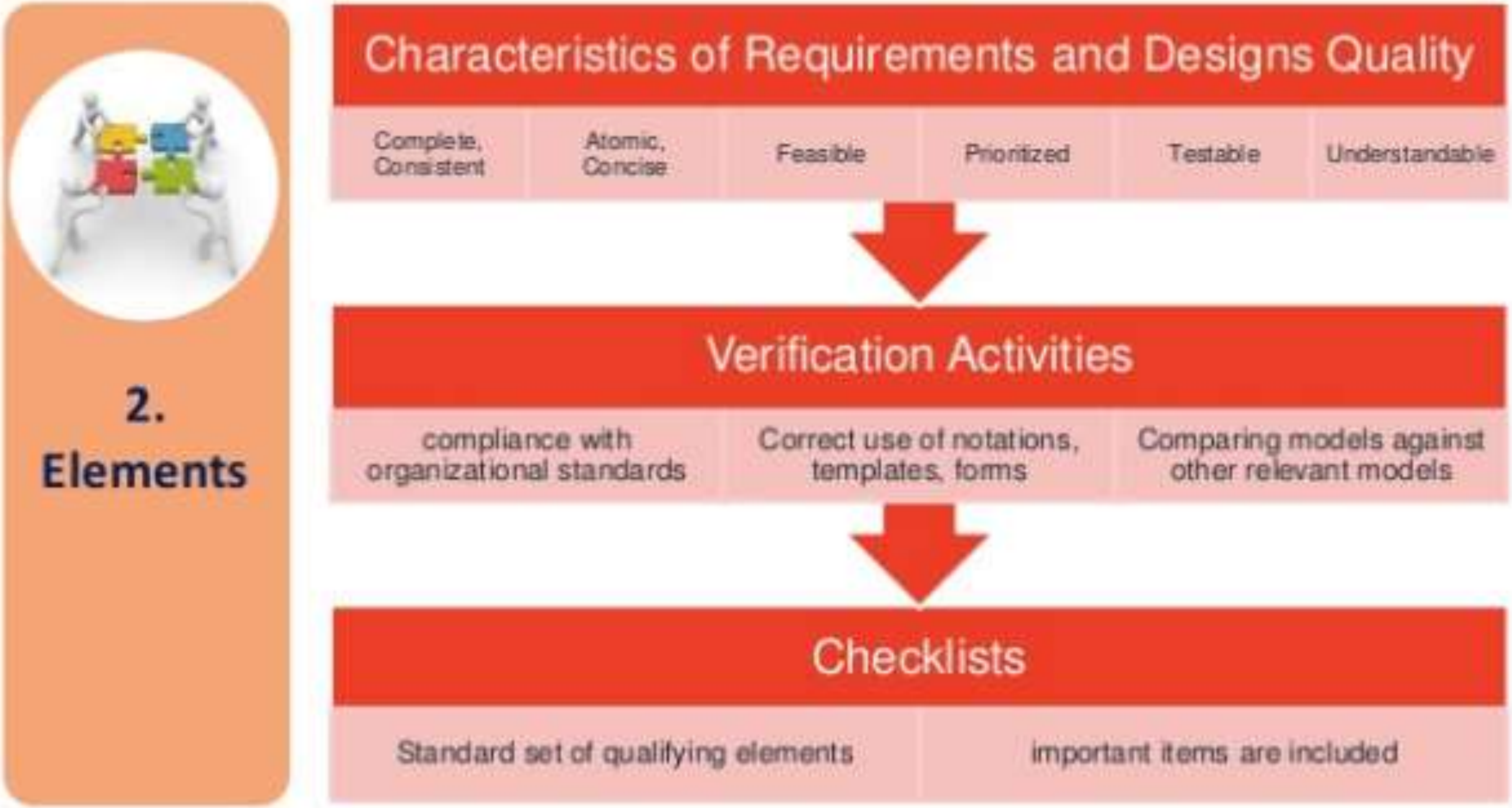
1.
Inputs



Well documented

Requirements (specified and modelled)

Well written



2. Elements

Characteristics of Requirements and Designs Quality

- Complete, Consistent
- Atomic, Concise
- Feasible
- Prioritized
- Testable
- Understandable

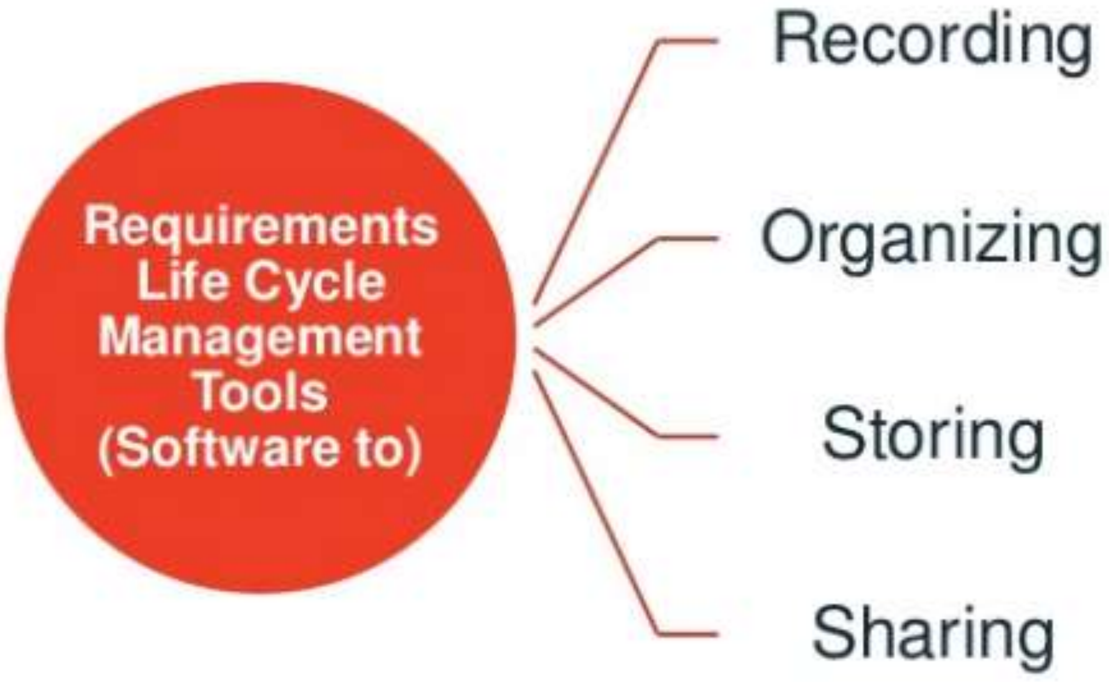
Verification Activities

- compliance with organizational standards
- Correct use of notations, templates, forms
- Comparing models against other relevant models

Checklists

- Standard set of qualifying elements
- important items are included

3. Guidelines and Tools



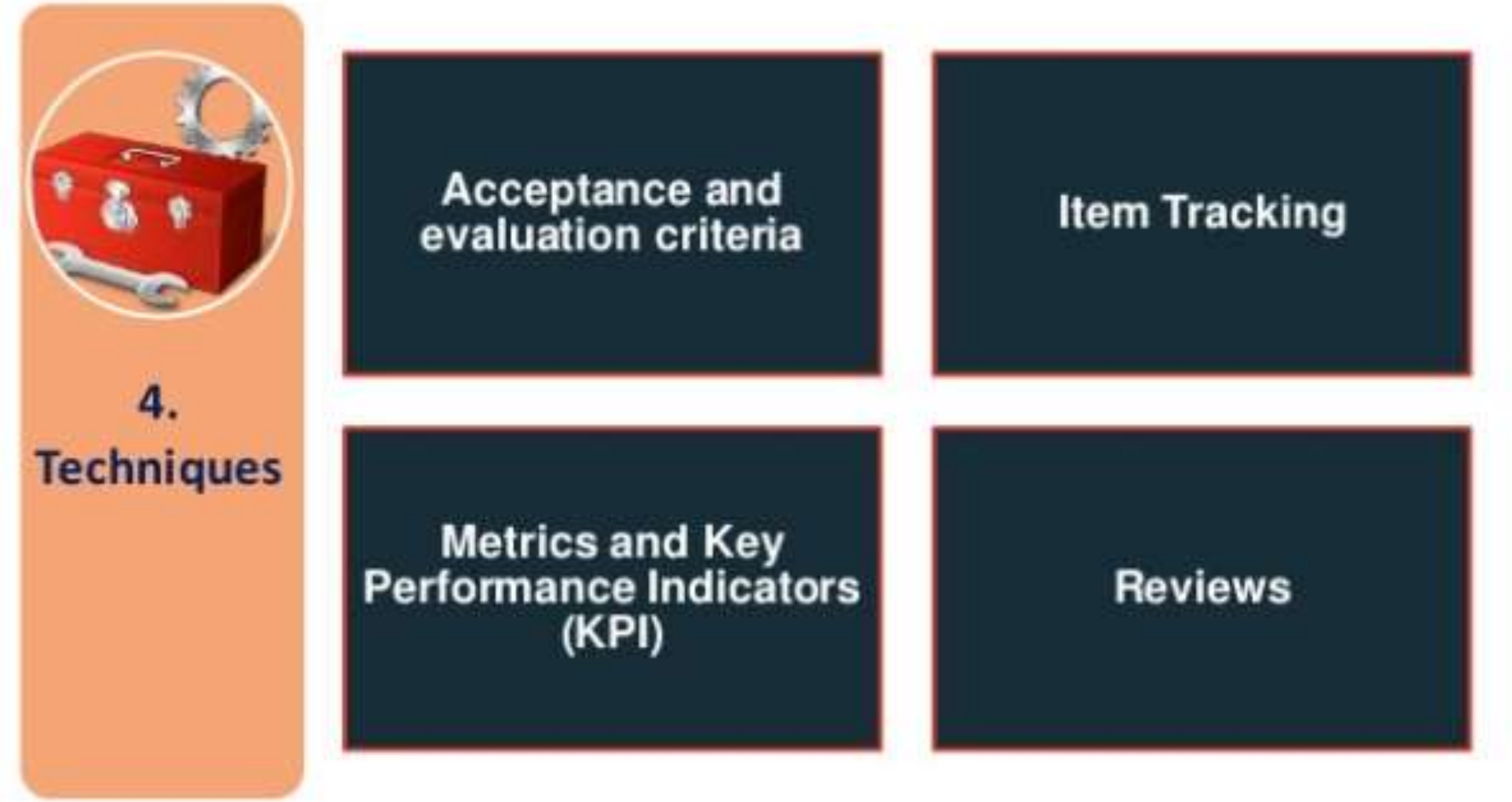
Requirements Life Cycle Management Tools (Software to)

Recording

Organizing

Storing

Sharing



4. Techniques

Acceptance and evaluation criteria

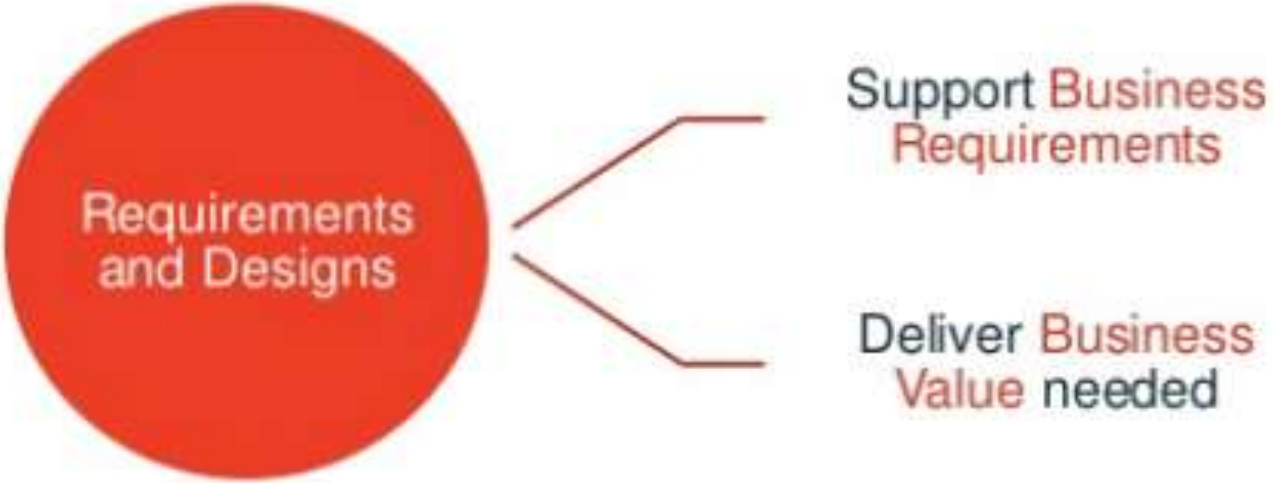
Item Tracking

Metrics and Key Performance Indicators (KPI)

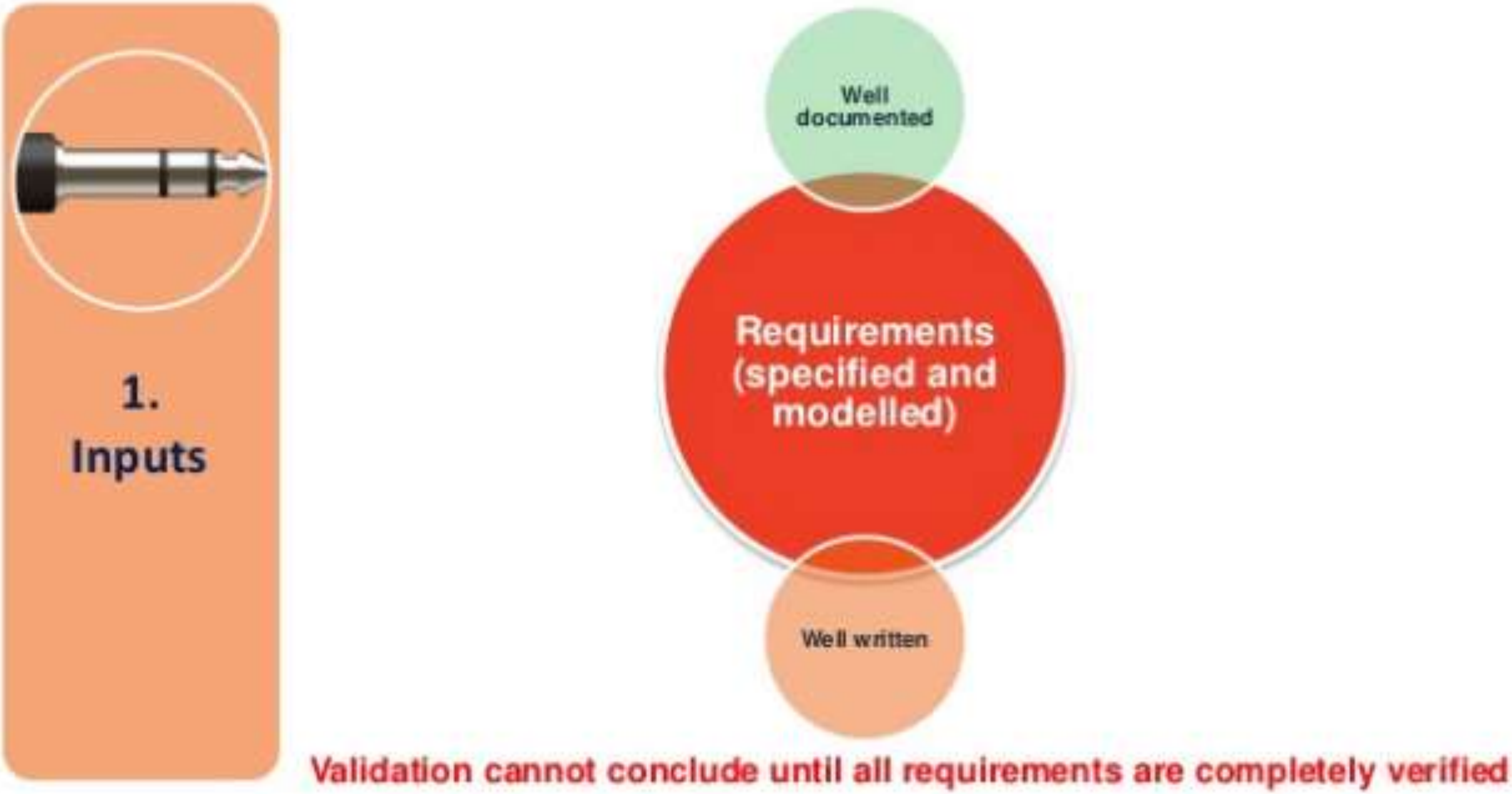
Reviews

Validate Requirements

Purpose



The purpose of Validate Requirements is to ensure that all requirements and designs align to the business requirements and support the delivery of needed value.



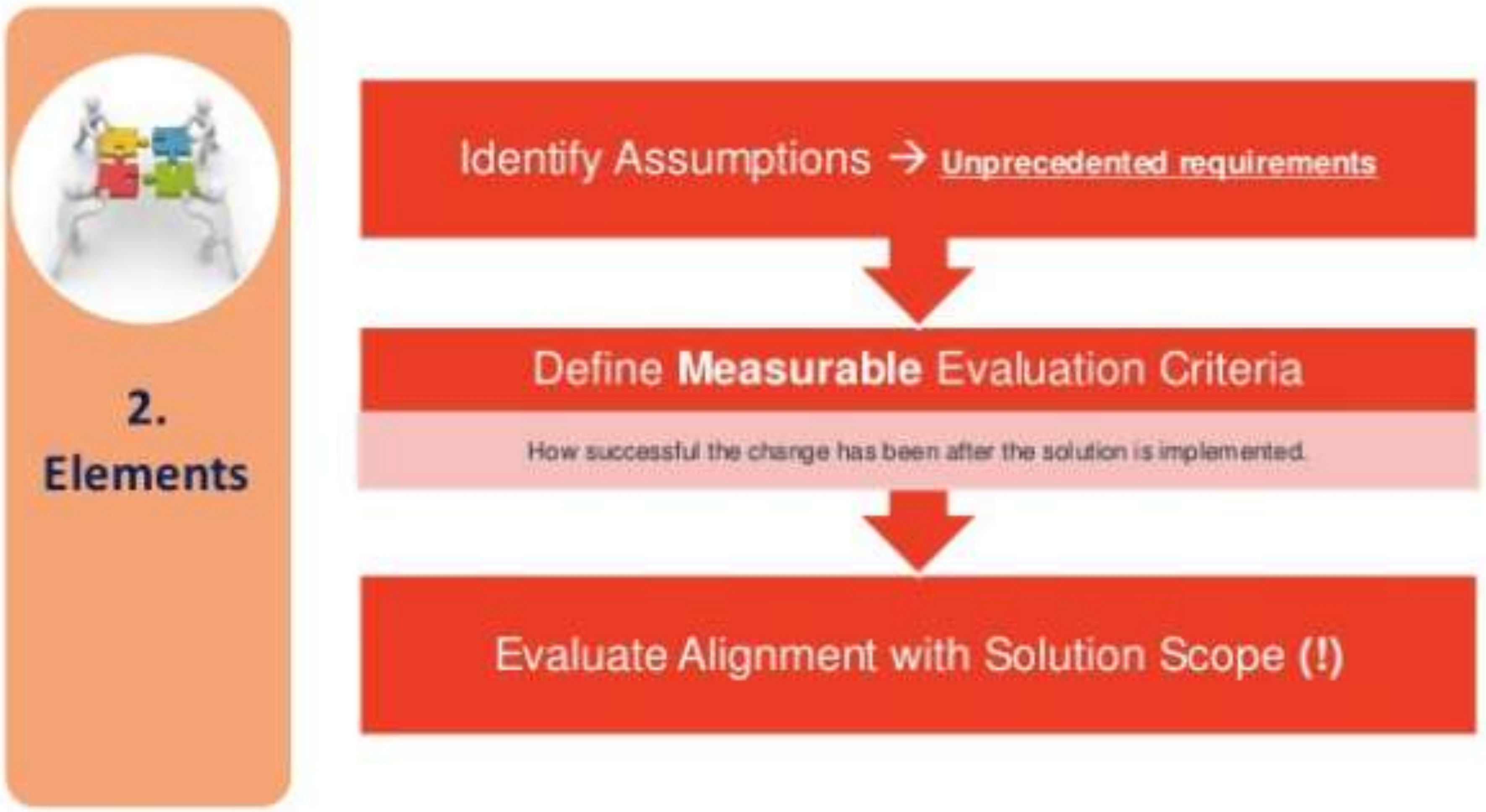
1. Inputs

Well documented

Requirements (specified and modelled)

Well written

Validation cannot conclude until all requirements are completely verified



2. Elements

Identify Assumptions → **Unprecedented requirements**

Define **Measurable** Evaluation Criteria

How successful the change has been after the solution is implemented.

Evaluate Alignment with Solution Scope (!)

mindsmapped

3. Guidelines and Tools

Business Objectives

Future State Description

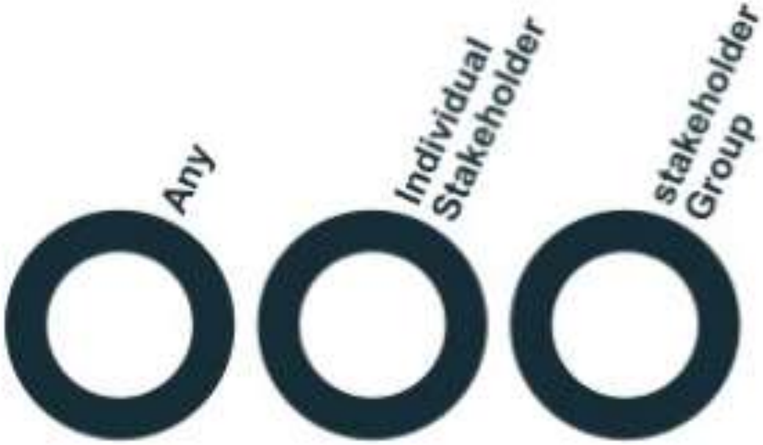
Potential Value

Solution Scope

4. Techniques

- Acceptance and Evaluation Criteria
- Document Analysis
- Financial Analysis
- Item Tracking
- Reviews
- Metrics and Key Performance Indicators (KPI)
- Risk Analysis Management

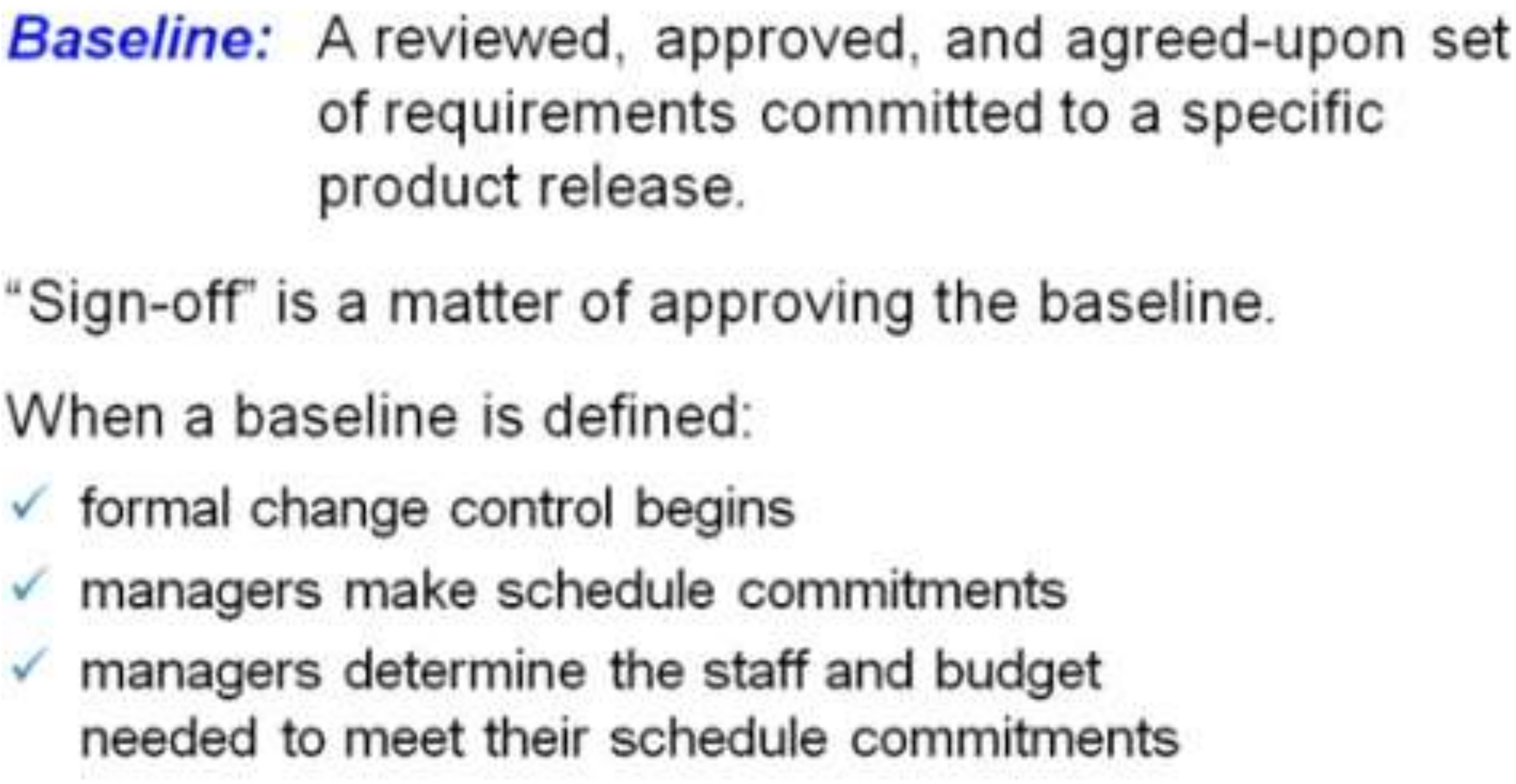
5. Stakeholders



Key Requirements Management Practices

- Create a requirements **baseline.**
- **Manage version** of requirements documents.
- Adopt and enforce a change control process.
- Perform requirements change **impact analysis.**
- Store requirement **attributes.**
- **Track** the status of each requirement.
- Trace requirements into **designs, code, and tests.**
- Store requirements in a **requirements management tool.**

The Requirements Baseline



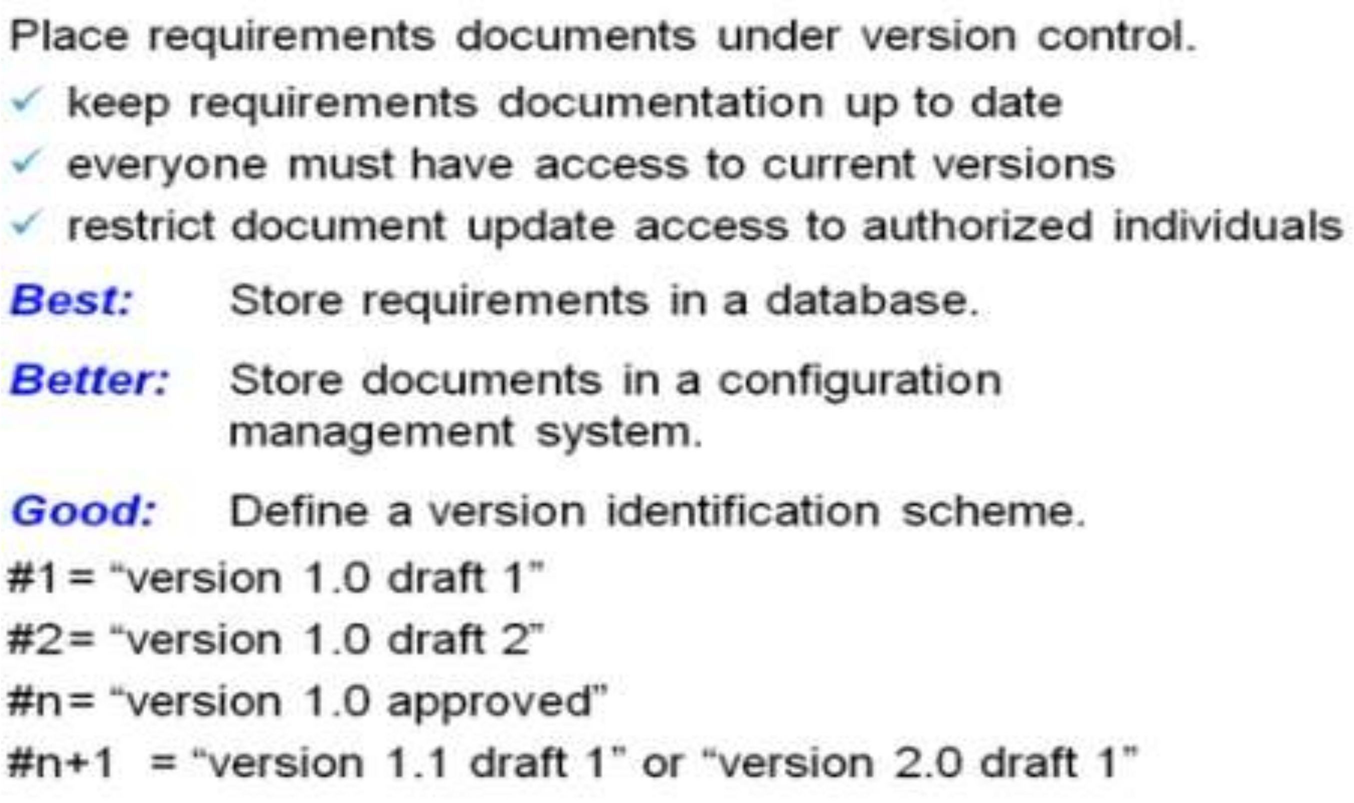
Baseline: A reviewed, approved, and agreed-upon set of requirements committed to a specific product release.

"Sign-off" is a matter of approving the baseline.

When a baseline is defined:

- ✓ formal change control begins
- ✓ managers make schedule commitments
- ✓ managers determine the staff and budget needed to meet their schedule commitments

Requirements Version Management



Place requirements documents under version control.

- ✓ keep requirements documentation up to date
- ✓ everyone must have access to current versions
- ✓ restrict document update access to authorized individuals

Best: Store requirements in a database.

Better: Store documents in a configuration management system.

Good: Define a version identification scheme.

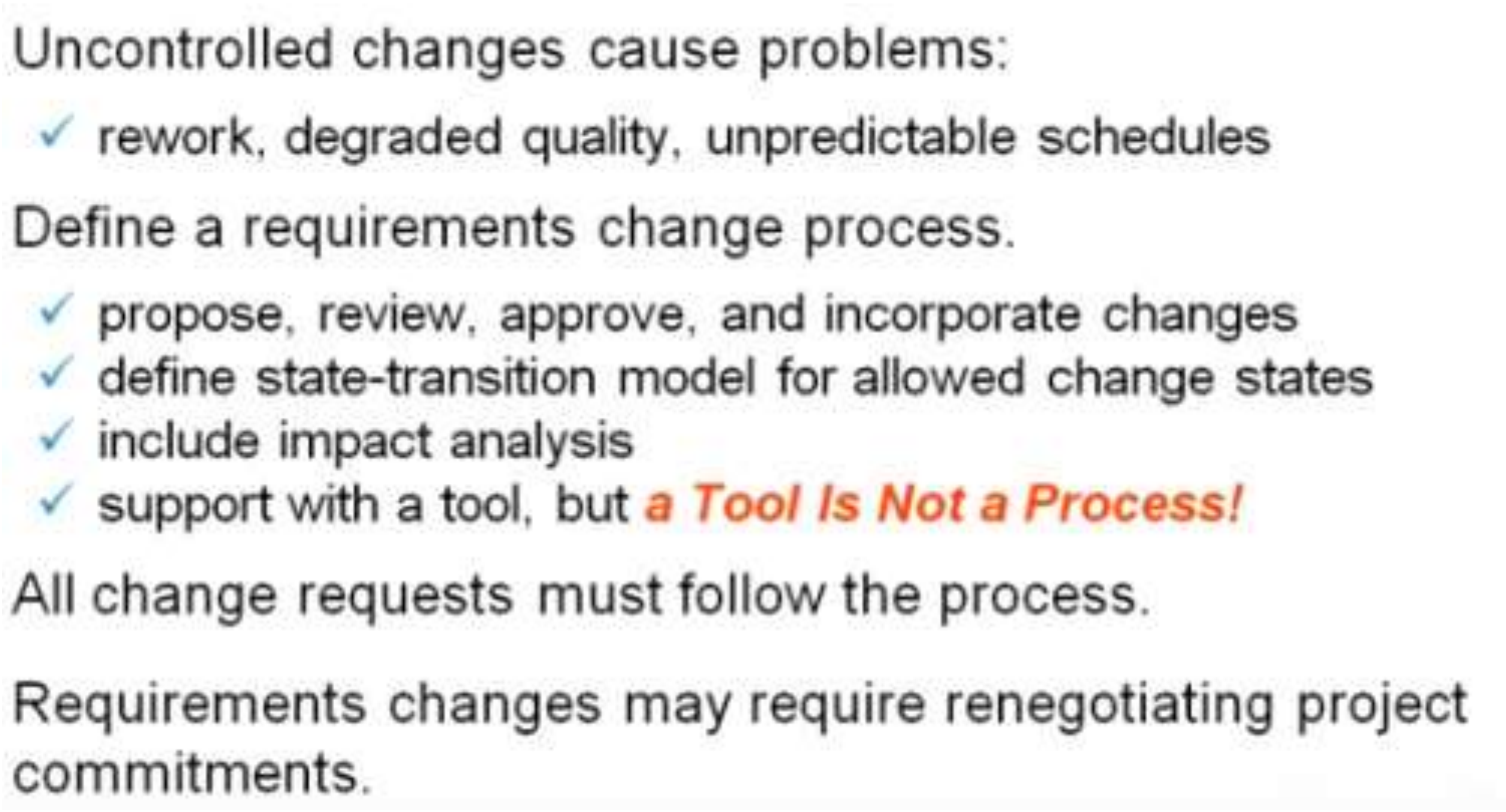
#1= "version 1.0 draft 1"

#2= "version 1.0 draft 2"

#n= "version 1.0 approved"

#n+1 = "version 1.1 draft 1" or "version 2.0 draft 1"

Requirements Change Control



Uncontrolled changes cause problems:

- ✓ rework, degraded quality, unpredictable schedules

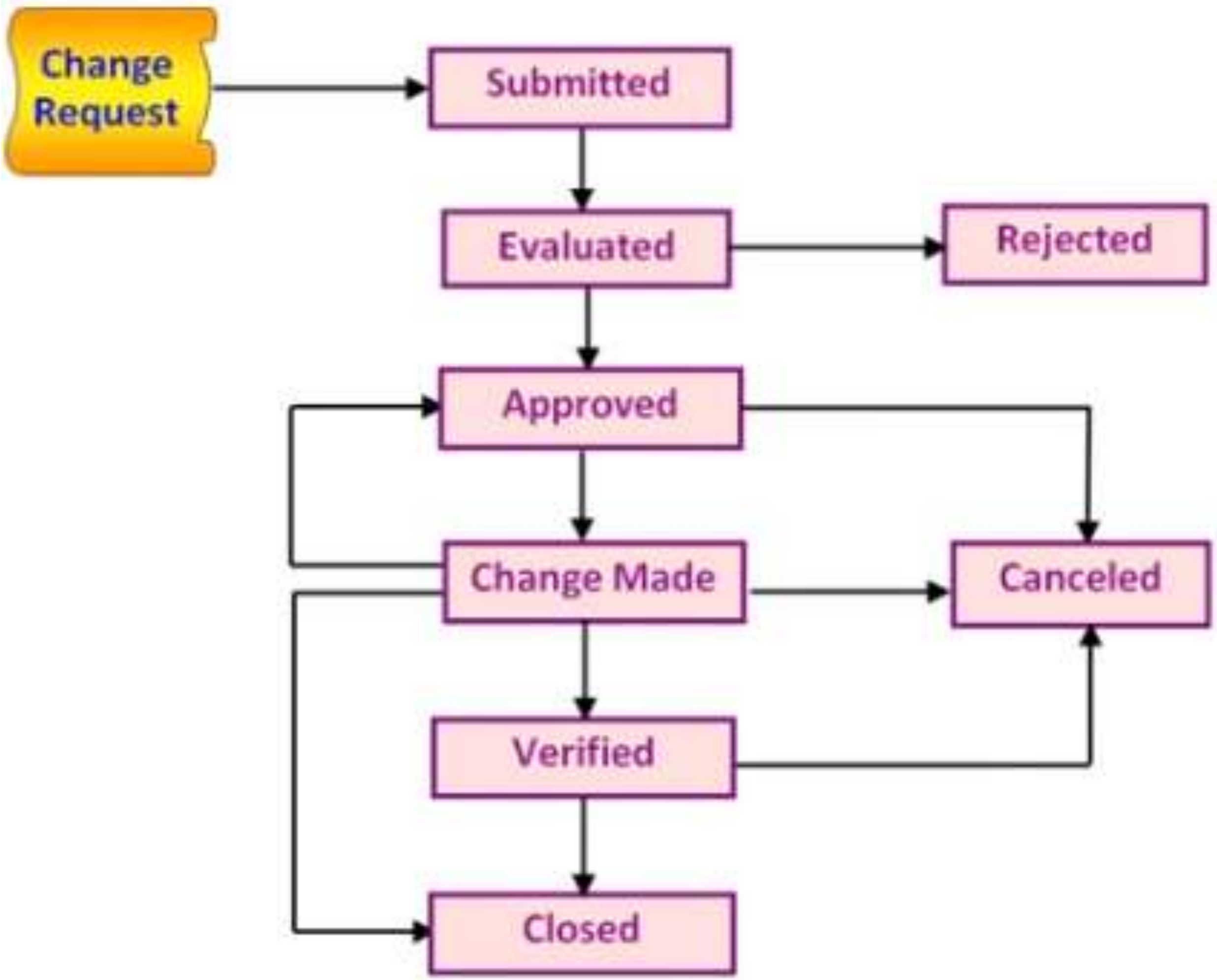
Define a requirements change process.

- ✓ propose, review, approve, and incorporate changes
- ✓ define state-transition model for allowed change states
- ✓ include impact analysis
- ✓ support with a tool, but ***a Tool Is Not a Process!***

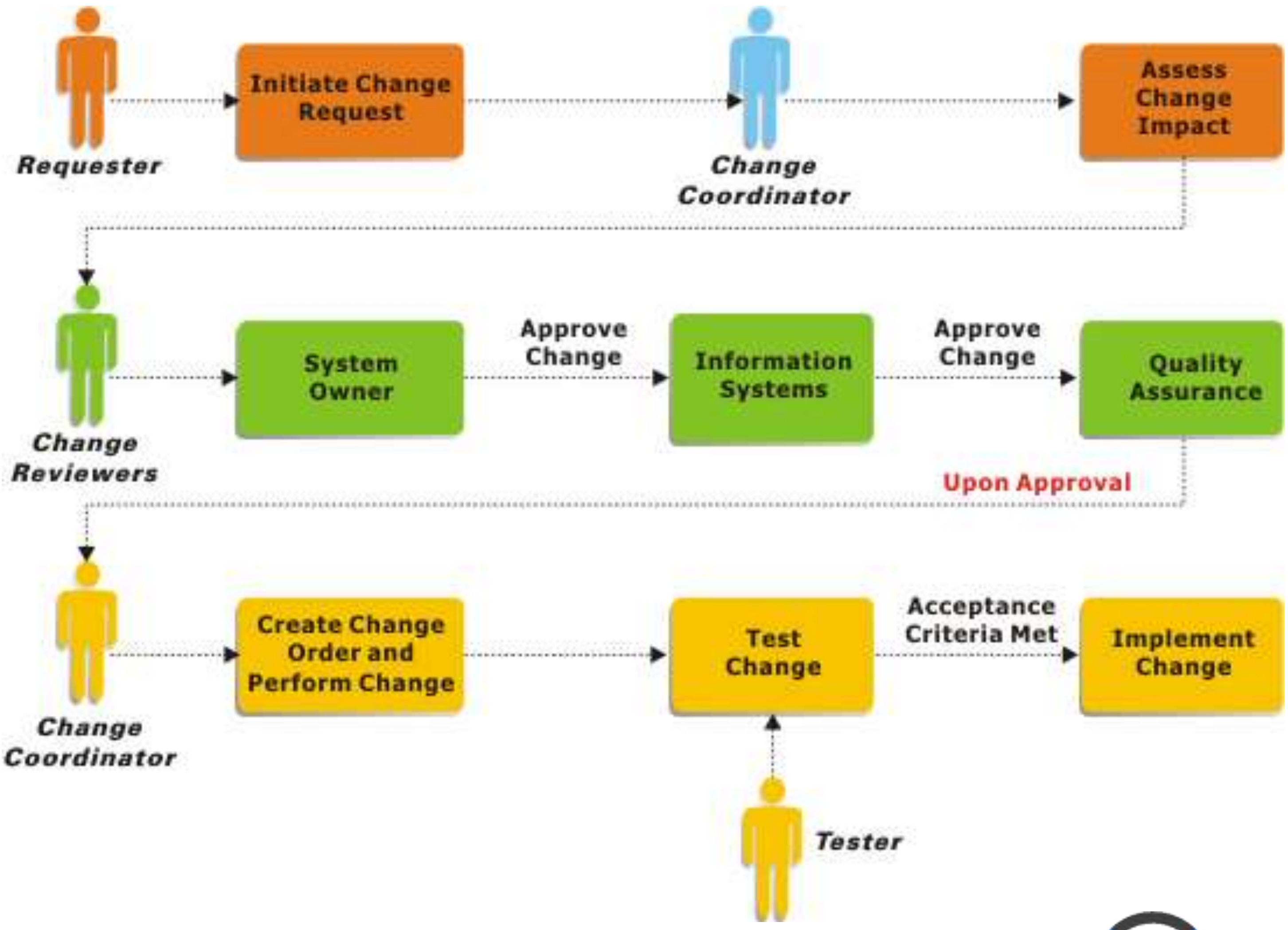
All change requests must follow the process.

Requirements changes may require renegotiating project commitments.

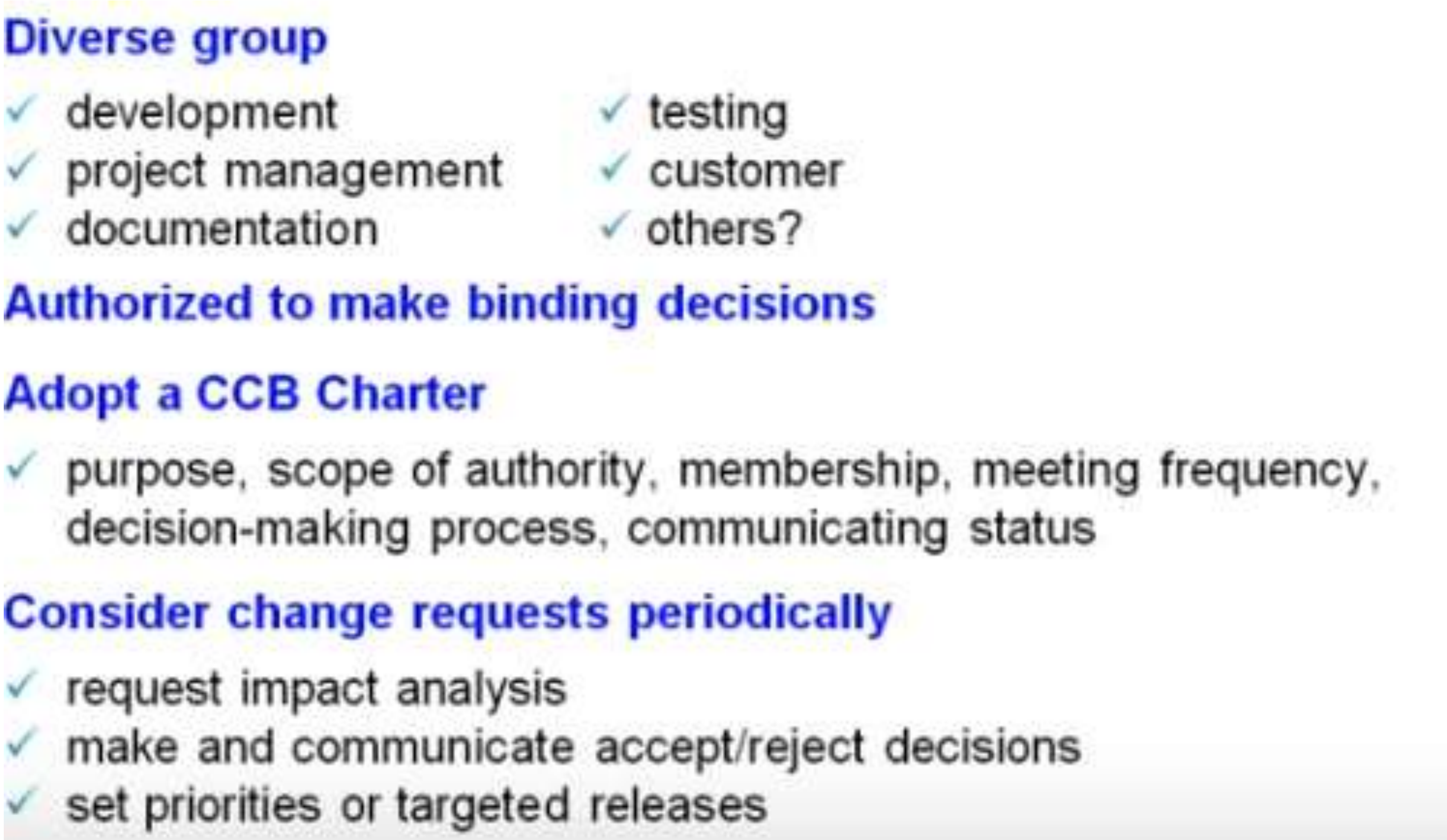
Possible Change Request Statuses



A Change Control System



Change Control Board



Diverse group

- ✓ development
- ✓ project management
- ✓ documentation
- ✓ testing
- ✓ customer
- ✓ others?

Authorized to make binding decisions

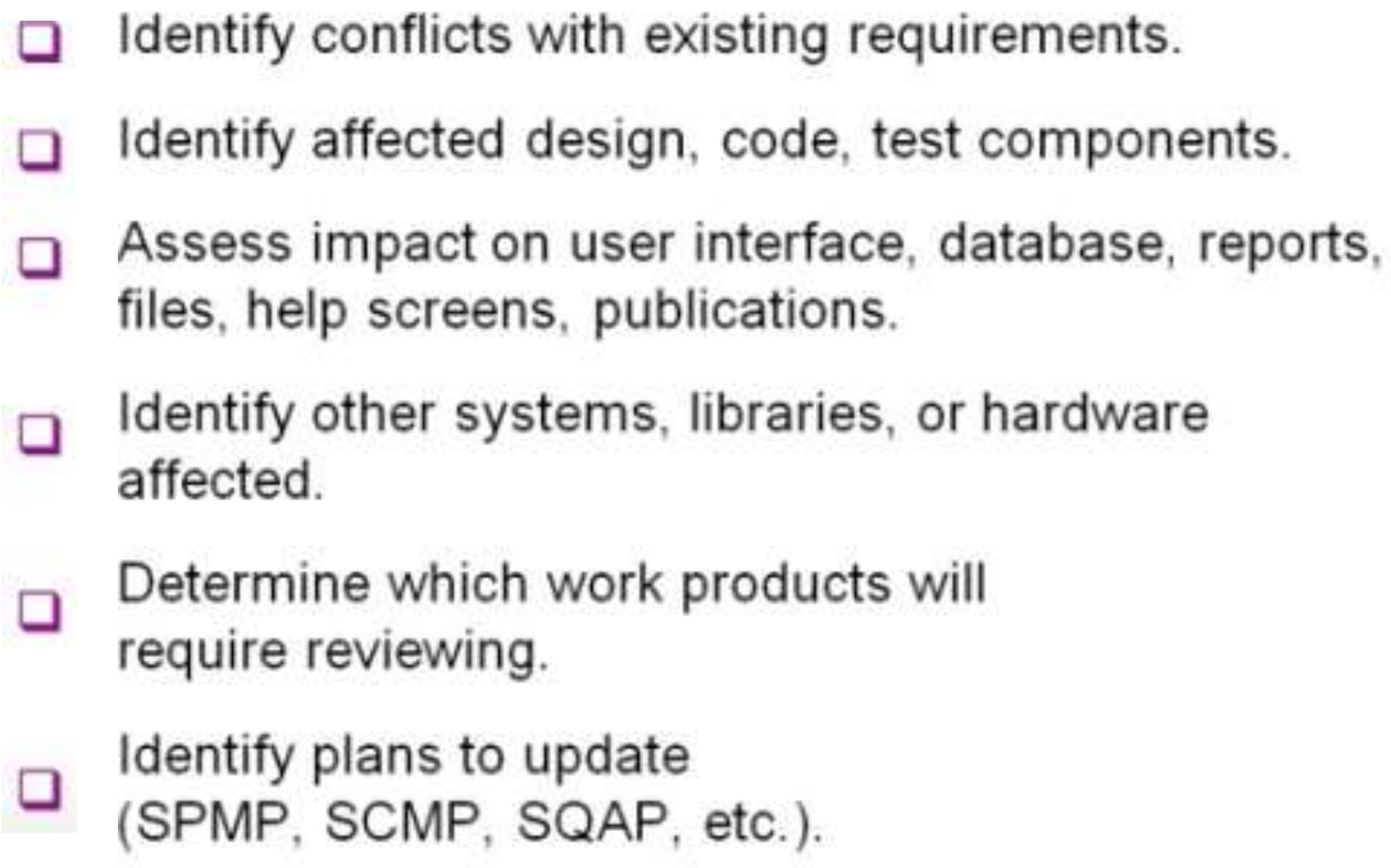
Adopt a CCB Charter

- ✓ purpose, scope of authority, membership, meeting frequency, decision-making process, communicating status

Consider change requests periodically

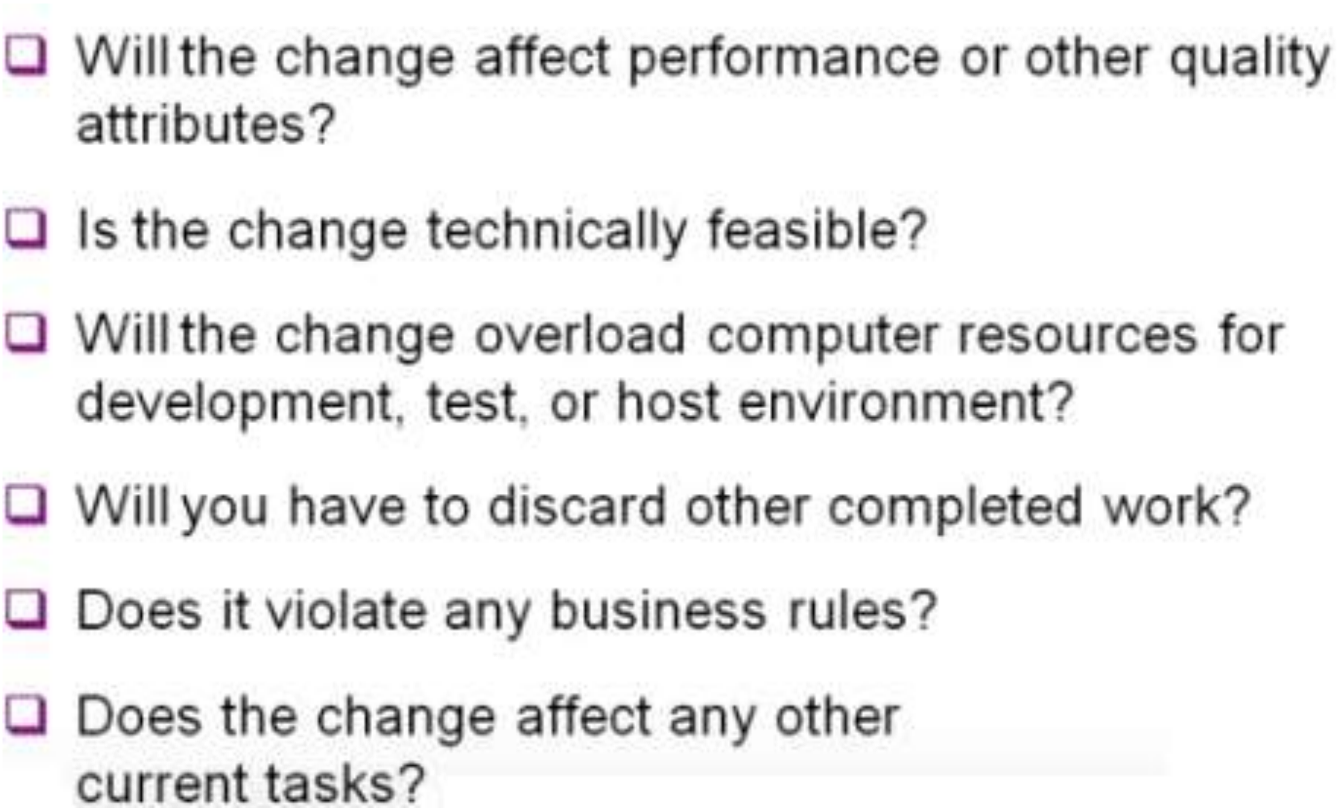
- ✓ request impact analysis
- ✓ make and communicate accept/reject decisions
- ✓ set priorities or targeted releases

Impact Analysis for Requirements Changes-1



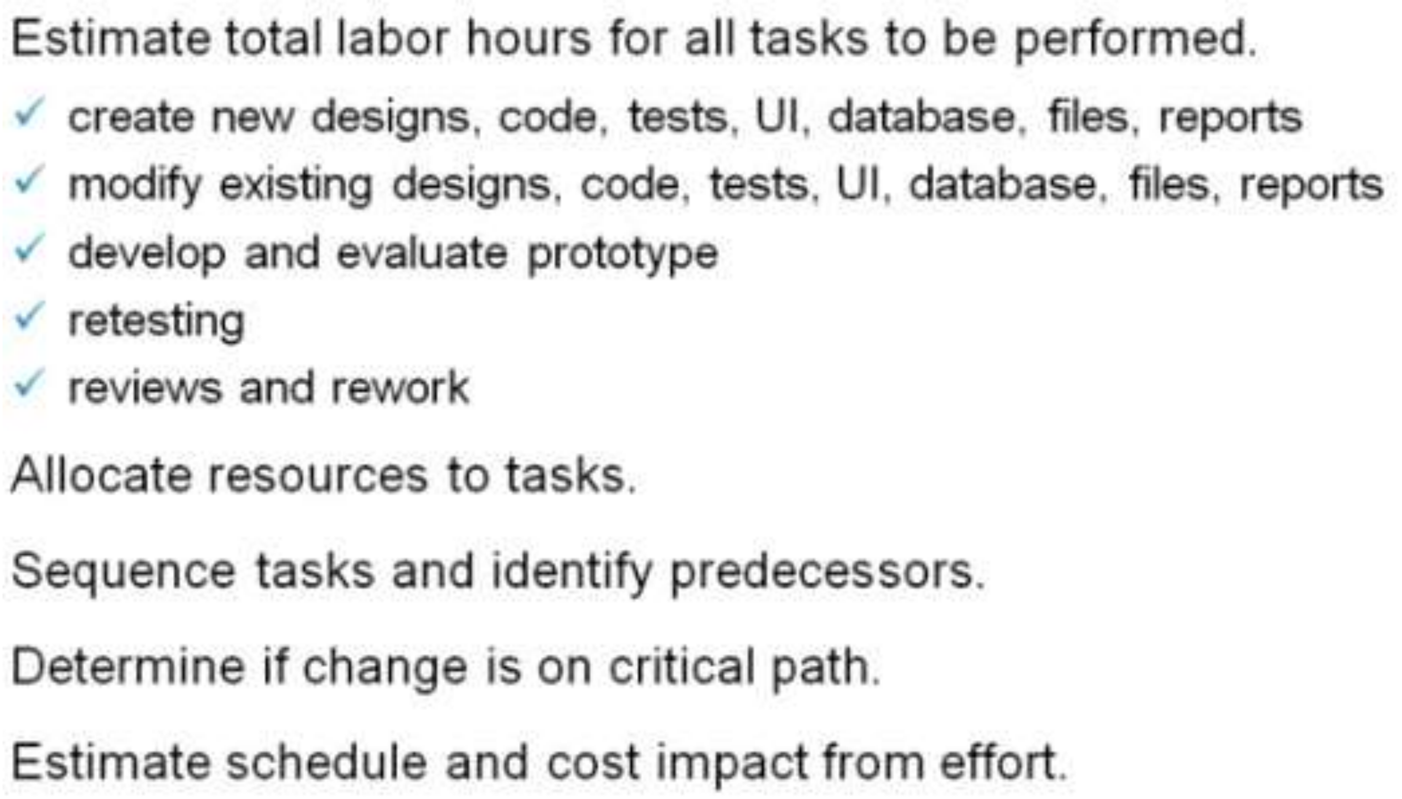
- Identify conflicts with existing requirements.
- Identify affected design, code, test components.
- Assess impact on user interface, database, reports, files, help screens, publications.
- Identify other systems, libraries, or hardware affected.
- Determine which work products will require reviewing.
- Identify plans to update (SPMP, SCMP, SQAP, etc.).

Impact Analysis for Requirements Changes-2



- ❑ Will the change affect performance or other quality attributes?
- ❑ Is the change technically feasible?
- ❑ Will the change overload computer resources for development, test, or host environment?
- ❑ Will you have to discard other completed work?
- ❑ Does it violate any business rules?
- ❑ Does the change affect any other current tasks?

Impact Analysis for Requirements Changes-3



Estimate total labor hours for all tasks to be performed.

- ✓ create new designs, code, tests, UI, database, files, reports
- ✓ modify existing designs, code, tests, UI, database, files, reports
- ✓ develop and evaluate prototype
- ✓ retesting
- ✓ reviews and rework

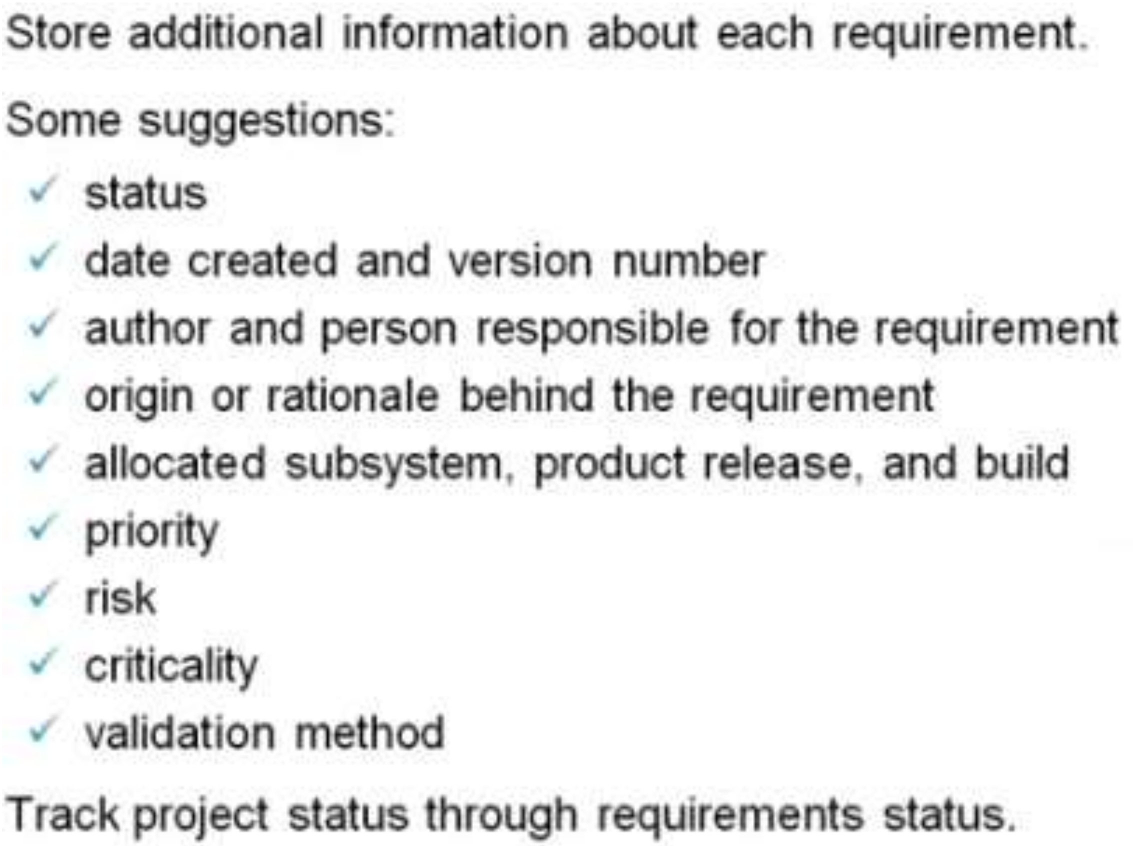
Allocate resources to tasks.

Sequence tasks and identify predecessors.

Determine if change is on critical path.

Estimate schedule and cost impact from effort.

Requirements Attributes



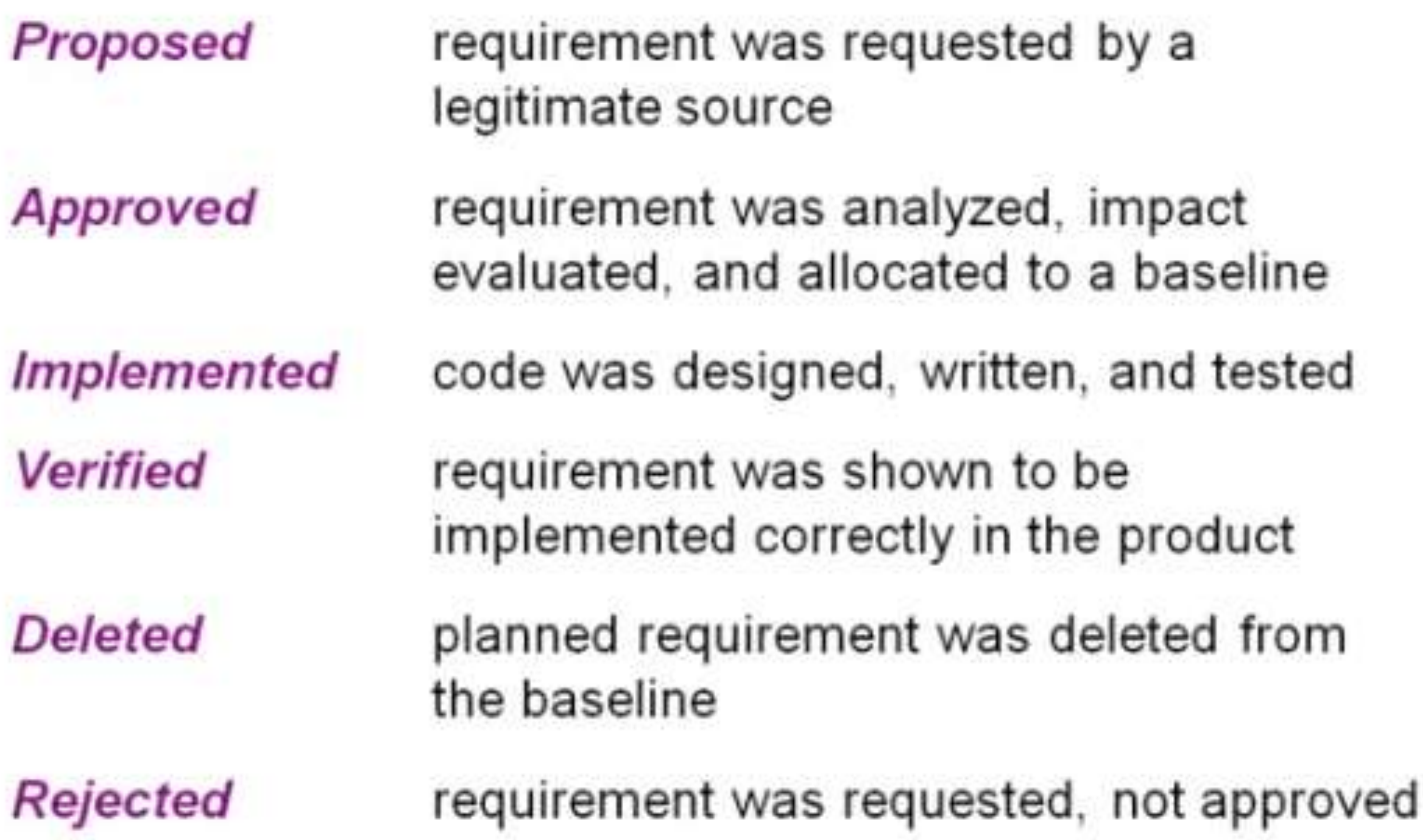
Store additional information about each requirement.

Some suggestions:

- ✓ status
- ✓ date created and version number
- ✓ author and person responsible for the requirement
- ✓ origin or rationale behind the requirement
- ✓ allocated subsystem, product release, and build
- ✓ priority
- ✓ risk
- ✓ criticality
- ✓ validation method

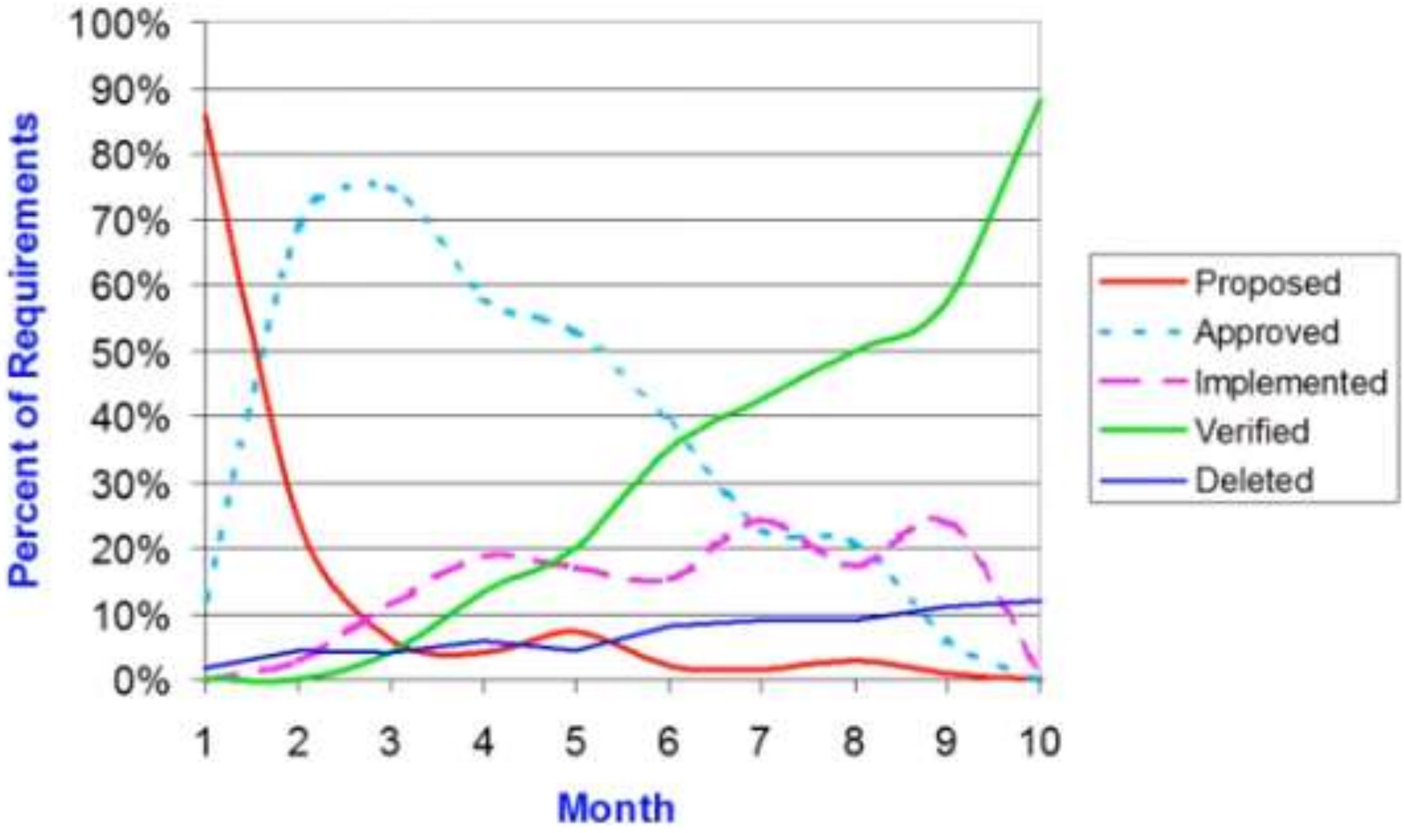
Track project status through requirements status.

Requirements Status Tracking-1

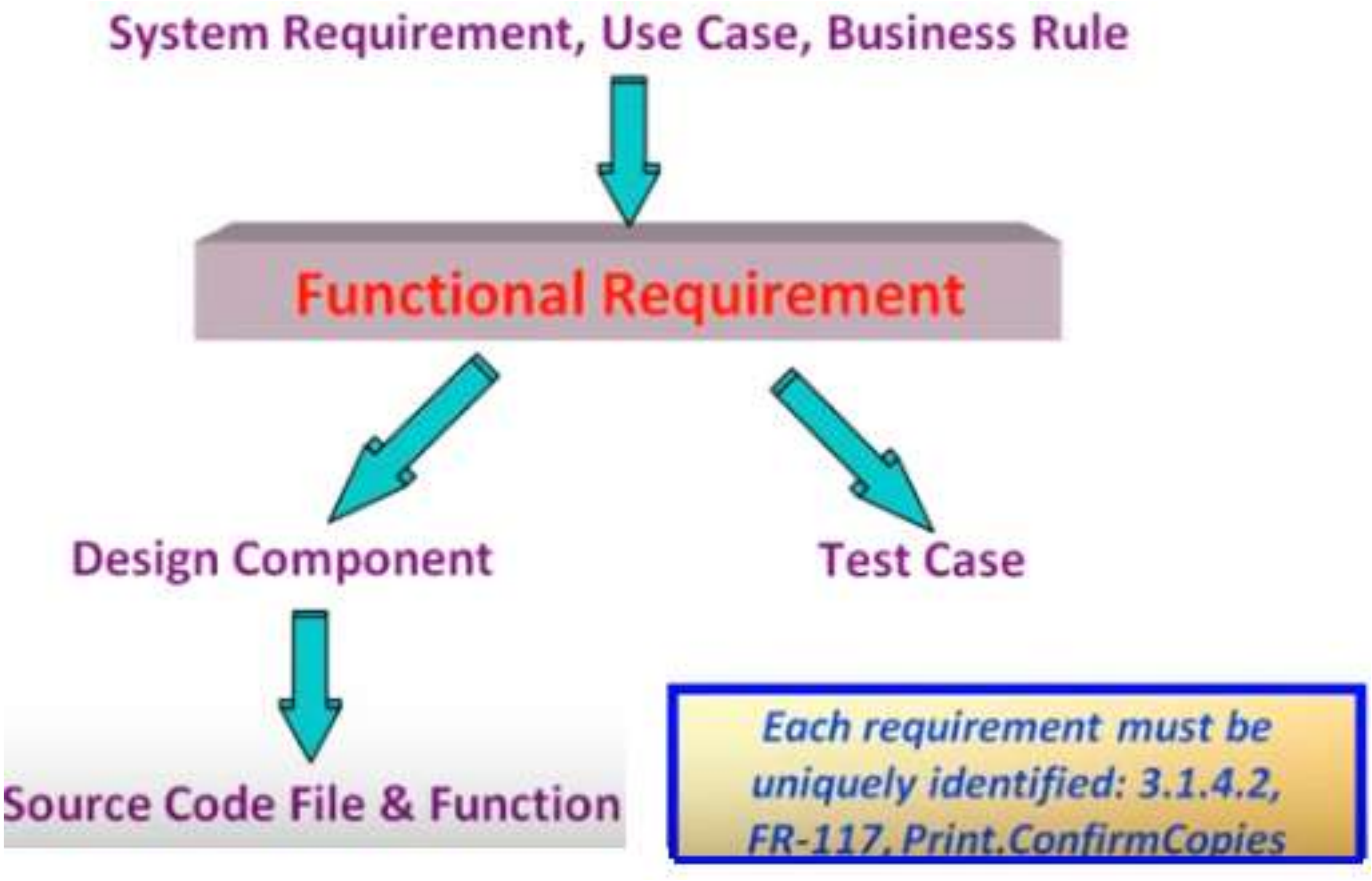


Proposed	requirement was requested by a legitimate source
Approved	requirement was analyzed, impact evaluated, and allocated to a baseline
Implemented	code was designed, written, and tested
Verified	requirement was shown to be implemented correctly in the product
Deleted	planned requirement was deleted from the baseline
Rejected	requirement was requested, not approved

Requirements Status Tracking-2



Requirements Traceability



System Requirement, Use Case, Business Rule

Functional Requirement

Design Component

Test Case

Source Code File & Function

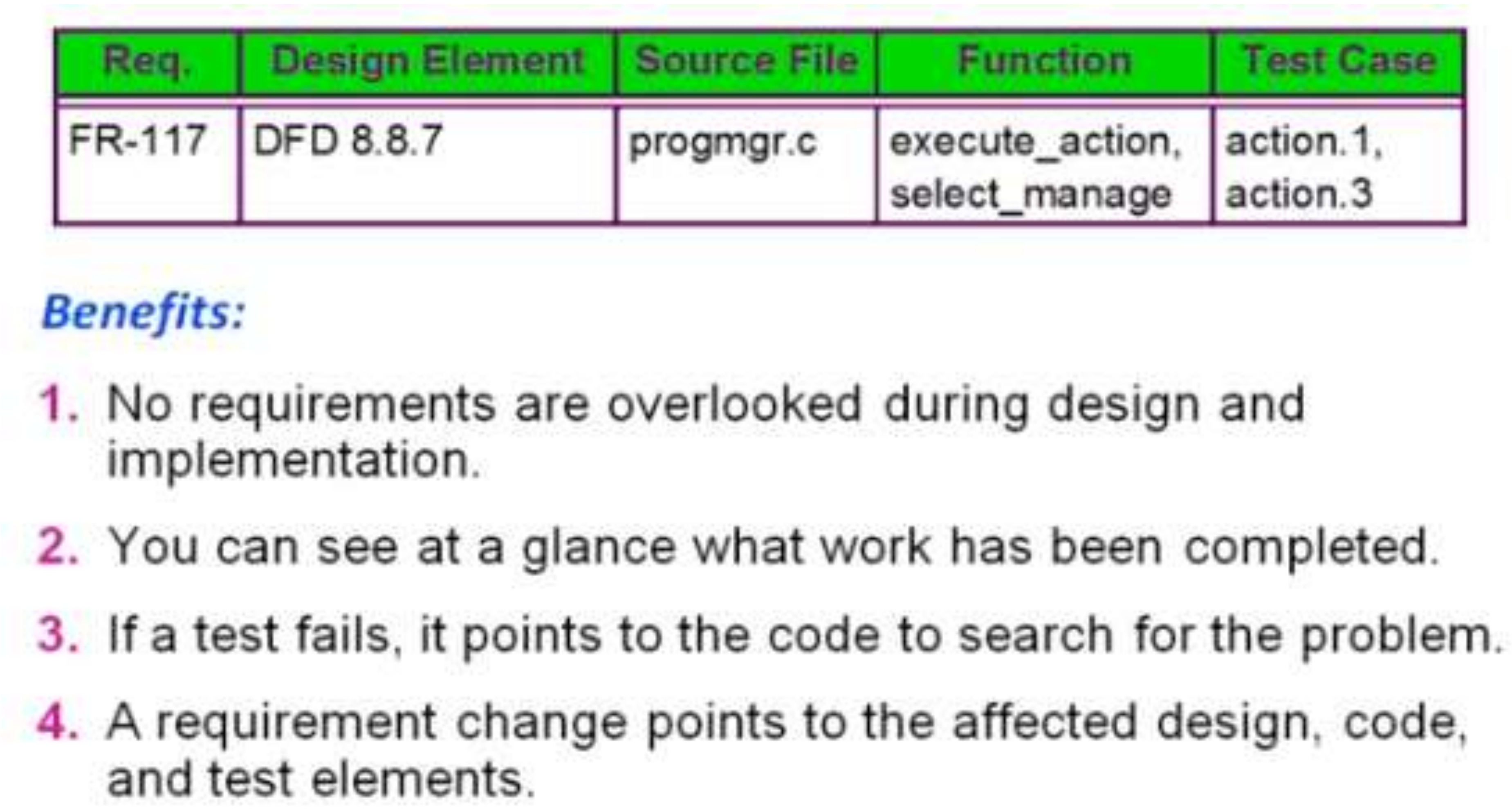
Each requirement must be uniquely identified: 3.1.4.2, FR-117, Print.ConfirmCopies

Requirement Traceability matrix

- The purpose of the Requirements Traceability Matrix is to ensure that all requirements defined for a system are tested.

- The requirements traceability matrix is usually developed in concurrence with the initial list of requirements (functional requirements documents)

RTM Example

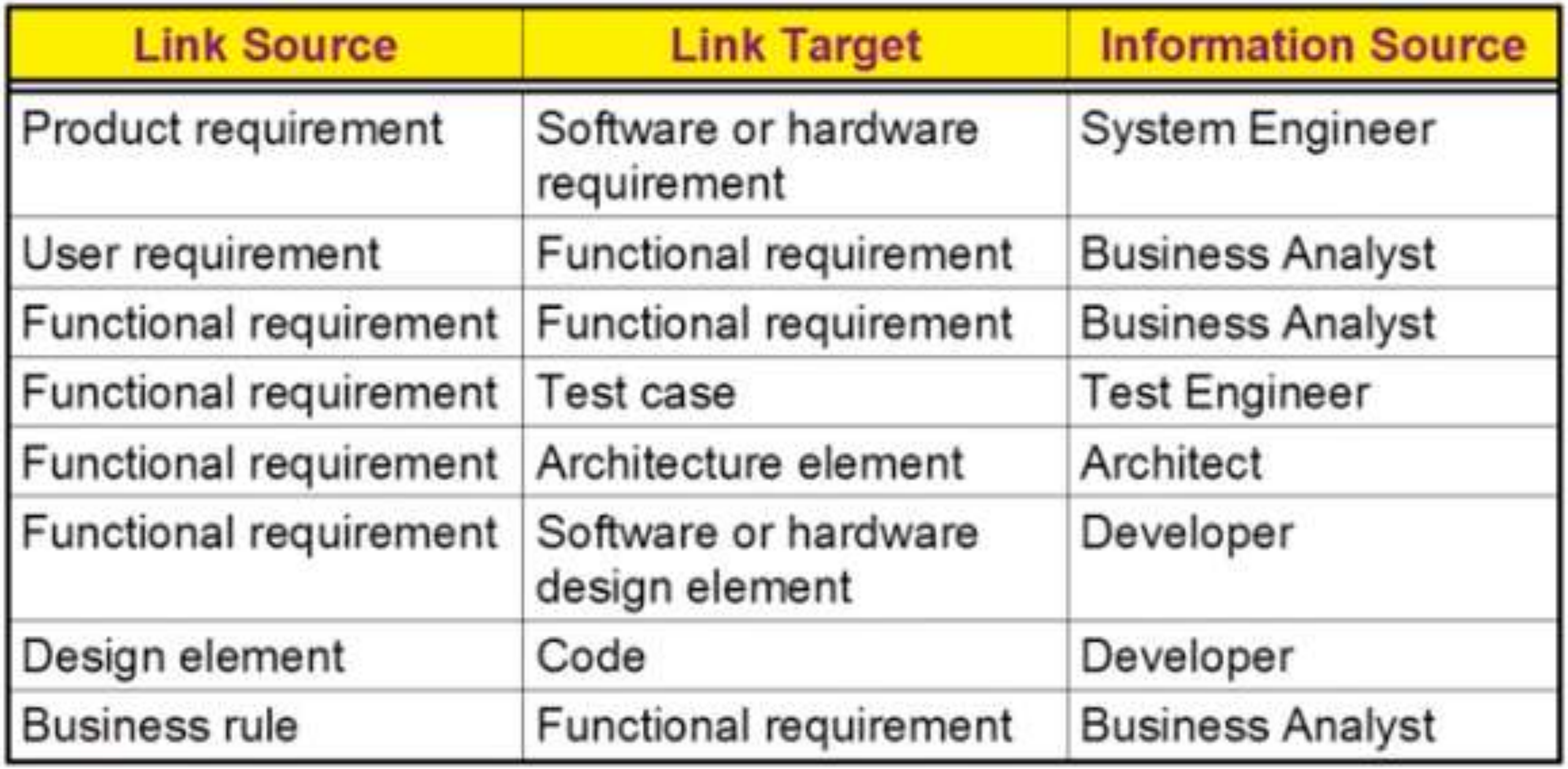


Req.	Design Element	Source File	Function	Test Case
FR-117	DFD 8.8.7	progmgr.c	execute_action, select_manage	action.1, action.3

Benefits:

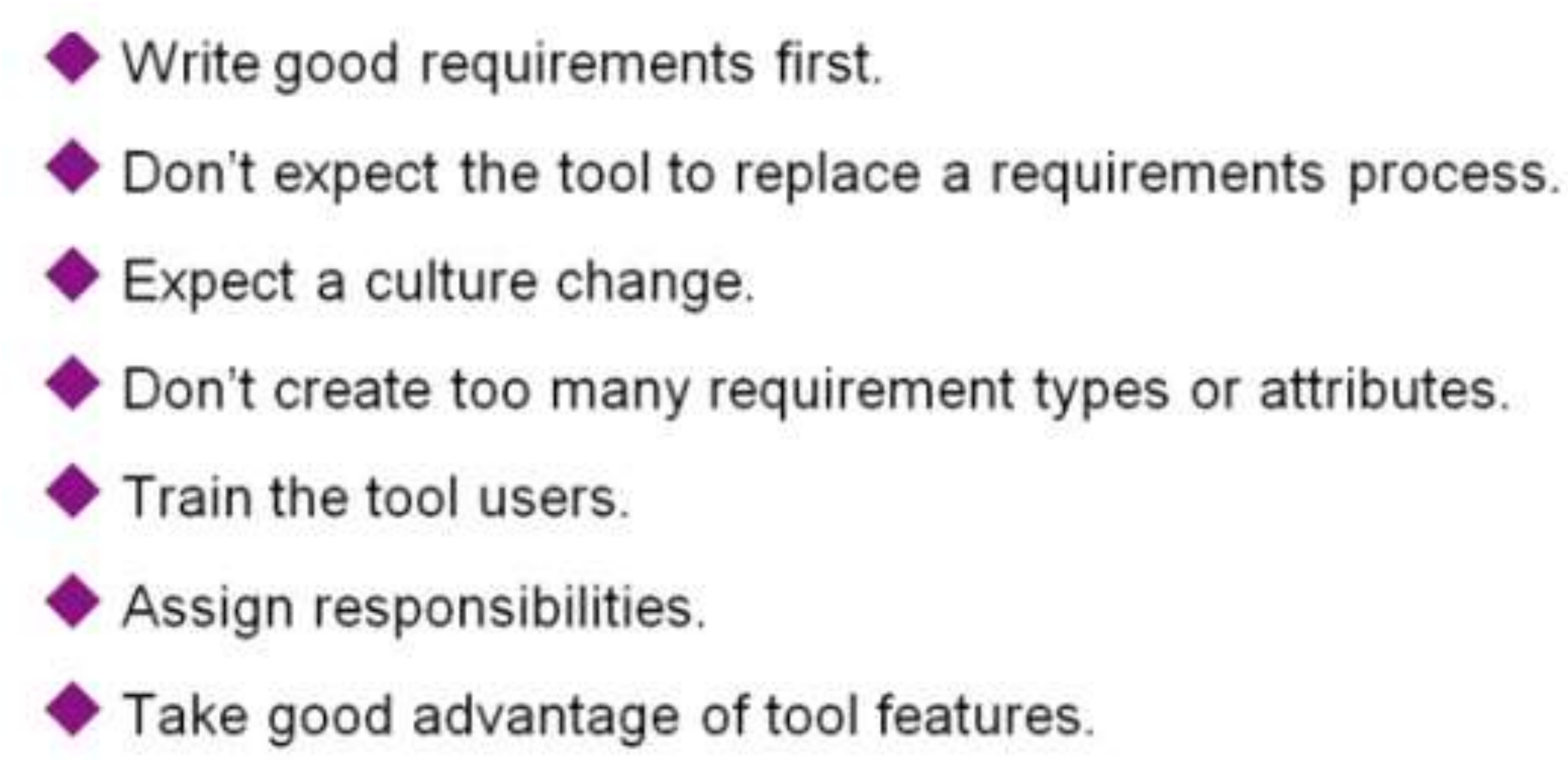
1. No requirements are overlooked during design and implementation.
2. You can see at a glance what work has been completed.
3. If a test fails, it points to the code to search for the problem.
4. A requirement change points to the affected design, code, and test elements.

Where Traceability Links Might Come From?



Link Source	Link Target	Information Source
Product requirement	Software or hardware requirement	System Engineer
User requirement	Functional requirement	Business Analyst
Functional requirement	Functional requirement	Business Analyst
Functional requirement	Test case	Test Engineer
Functional requirement	Architecture element	Architect
Functional requirement	Software or hardware design element	Developer
Design element	Code	Developer
Business rule	Functional requirement	Business Analyst

Requirements Management Best Practices



- Write good requirements first.
- Don't expect the tool to replace a requirements process.
- Expect a culture change.
- Don't create too many requirement types or attributes.
- Train the tool users.
- Assign responsibilities.
- Take good advantage of tool features.

Thank you