**Chapter 28   
Change happens**

1. **Why is it essential to manage changes in a project?**

* Explore the importance and benefits of controlling changes in maintaining project stability and meeting objectives.

Managing changes is essential to:

* Adapt to evolving business needs and market changes
* Avoid miscommunication and expectation gaps
* Prevent code inconsistency and rework
* Support informed, cost-effective decisions
* Ensure that all changes are communicated and traceable
* Maintain project stability and deliver expected value

1. **What key elements should be included in a change control policy?**

* Investigate what guidelines and rules should be established to effectively manage changes within a project.

A good change control policy should include:

* **Clear process** for submitting and evaluating change requests
* **Defined roles** and responsibilities for decision-making
* **Impact assessment** on cost, schedule, and quality
* **Formal approval** before implementing changes
* **Transparent communication** to all affected stakeholders
* **Documentation and traceability** of all changes
* **Version control** to track requirement updates

1. **What are the basic steps involved in a change control process?**

* Detail the key stages of identifying, evaluating, and implementing changes within a formalized process.

A well-structured change control process typically includes the following key steps:

* **Submit the change request**  
  – Stakeholders formally propose a change using a simple, standardized method.
* **Log and track the request**  
  – Record the request and ensure it's not lost or overlooked.
* **Initial review and triage**  
  – Verify if the request is valid and clear enough for further analysis.
* **Impact analysis**  
  – Assess the effect on cost, schedule, system components, and customer value.
* **Decision-making**  
  – Authorized roles decide whether to approve, reject, or defer the change.
* **Communicate the decision**  
  – Notify all affected stakeholders of the outcome.
* **Implement the approved change**  
  – Start at the highest abstraction level affected and cascade through related components.
* **Update documentation and traceability**  
  – Ensure requirements, designs, tests, and user documents reflect the change.
* **Review and verify the change**  
  – Confirm that the change is correctly implemented and tested.

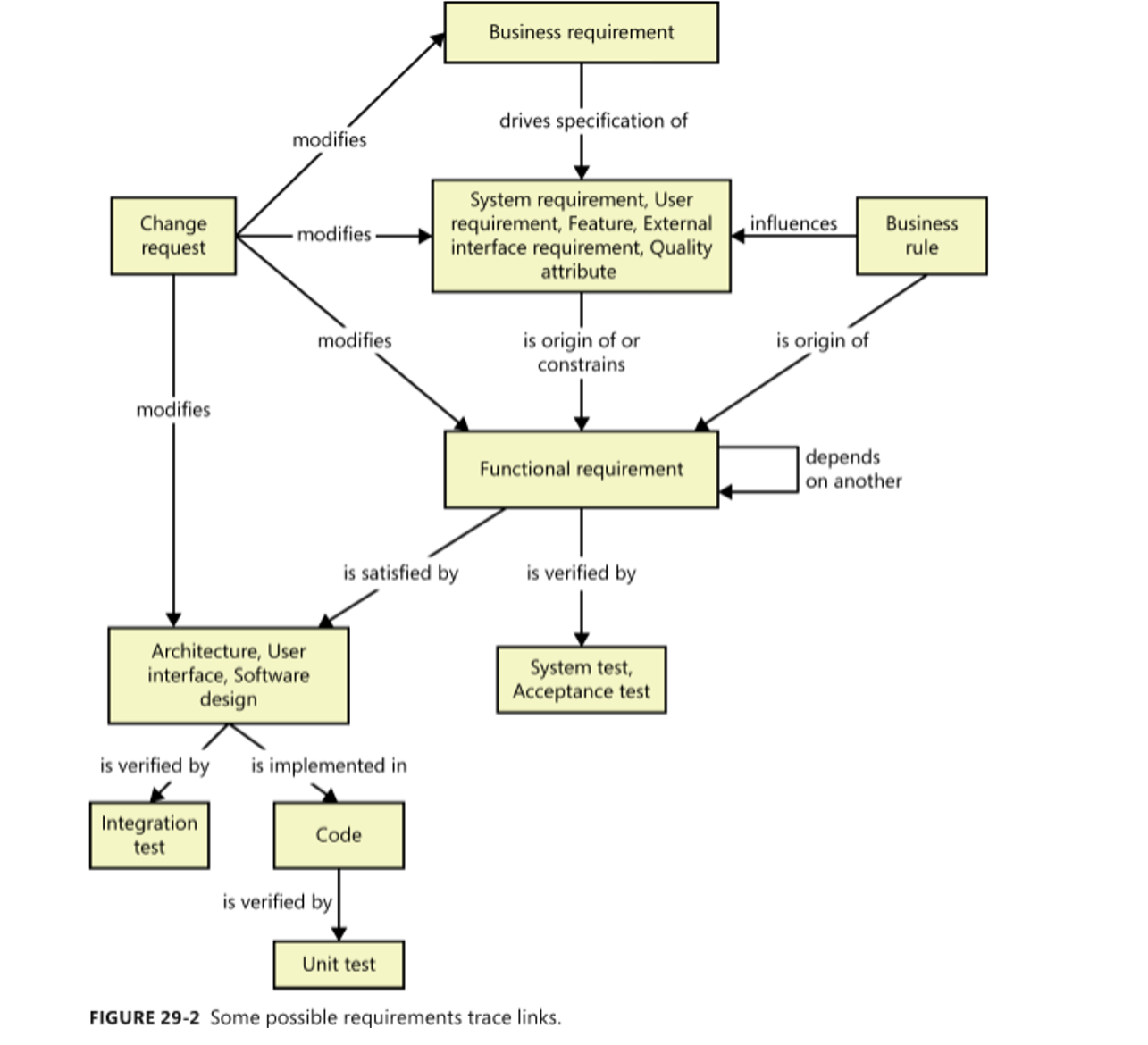
1. **What role does the Change Control Board (CCB) play in change management?**

* Understand the responsibilities and authority of the Change Control Board in approving or rejecting changes.
* The Change Control Board (CCB) is responsible for evaluating, approving, modifying, or rejecting proposed changes to a project’s requirements, design, or scope. It ensures that only beneficial and feasible changes are implemented. The CCB helps maintain control over the project's direction, cost, and schedule by making structured decisions based on input from relevant stakeholders. It also ensures transparency, traceability, and that all impacted teams are informed of decisions.

1. **How can change impact analysis improve decision-making in a project?**

* Discuss the importance of assessing the potential effects of a change on various aspects of a project before implementation.
* Change impact analysis helps project teams understand how a proposed change will affect cost, timeline, resources, system components, and customer value. By identifying potential risks and dependencies before implementation, teams can avoid unexpected side effects, rework, or conflicts. This leads to more informed, balanced, and strategic decision-making, ensuring that accepted changes truly align with business goals and project constraints.

**Chapter 29   
Links in the requirements chain**

**Explain diagram below**

1. Change Request – Starting point of change

A Change Request may affect:

* Business requirements – business goals or objectives
* System/User/Feature/Quality requirements – detailed technical and non-functional needs
* Functional requirements – specific system functionalities
* Design – including architecture, user interface, and software structure

2. Business Requirements – Strategic foundation

Business requirements drive the creation of lower-level requirements and are also influenced by business rules.

3. Functional Requirements – Testing foundation

Functional requirements:

* Guide software and UI design
* Serve as the basis for system and acceptance testing
* May have dependencies on or from other functional requirements

4. Design → Code → Testing

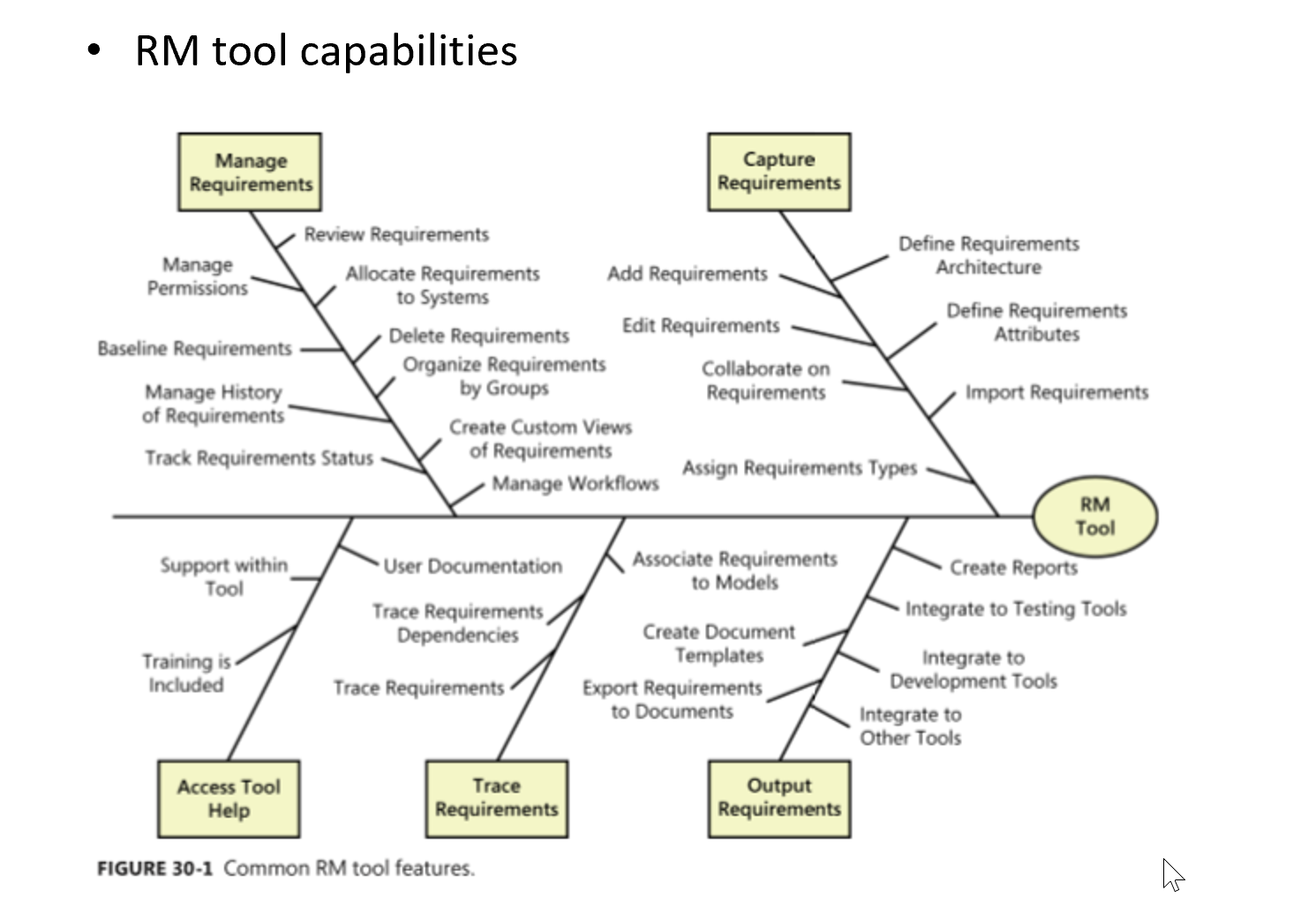
From functional requirements:

* Software design is created
* Then implemented in code
* And verified through:
  + Unit tests
  + Integration tests
  + System and acceptance tests

5. Key takeaway

The diagram shows that even a small change request can cascade through all stages of the development life cycle.

Chapter 30  
Tools for requirements engineering



### **1. Manage Requirements**

An RM tool allows teams to:

* Review, delete, and organize requirements
* Set permissions for users
* Allocate requirements to systems or components
* Group requirements for easier tracking
* Create custom views and manage workflows
* Baseline requirements and track their history and status

### **2. Capture Requirements**

The tool also helps with capturing and entering new requirements:

* Add, edit, and collaborate on requirements
* Import requirements from external sources
* Define requirement types, attributes, and architecture
* Assign requirement types for better categorization

### **3. Access Tool Help**

Support features included in the RM tool:

* Built-in help documentation
* Training resources for users
* Guides and instructions within the tool

### **4. Trace Requirements**

One of the most important capabilities:

* Trace relationships and dependencies between requirements
* Link requirements to design elements, code, or test cases
* Maintain full traceability across the project lifecycle

### **5. Output Requirements**

The tool enables:

* Exporting requirements to documents and reports
* Creating document templates
* Integrating with:
  + **Testing tools** (for test planning)
  + **Development tools** (for implementation)
  + **Other tools** (e.g., project management systems)

## **Conclusion**

This diagram shows that an RM Tool is **not just for storing requirements** — it acts as a **central hub** for:

* Capturing and organizing requirements
* Managing change
* Ensuring traceability
* Supporting collaboration
* Connecting with development and testing tools

An effective RM tool helps ensure that requirements are always **clear, traceable, and under control** throughout the project lifecycle.