List of Requirements

Format:

- Can be free or structured text that describes system functions and properties.

Advantages:

- Simple to **draft** and **distribute**.
- Can be **versioned** to track changes over time.

Disadvantages:

- Lacks focus on **user interaction**, making it hard for customers to comprehend.
- Can introduce **ambiguities** and **inconsistencies**, especially in interactions between requirements.

Use Case Diagrams

Format:

- **Diagrams** depict user interactions with the system.
- **Textual** description of these interactions follows a sequence of steps.

Advantages:

- **Intuitive** and simpler for customers to grasp.
- Focuses on **user functions** (what the system does).

Disadvantages:

- Hard to represent and track **non-functional requirements**.
- Managing diagrams can be more **work-intensive** compared to text.

User Stories

Format:

- Structured as: "As a [user], I want to do [this] because [of that]."

Advantages:

- Compact, **intuitive**, and simple for customers to understand.
- Emphasizes **user functions**.

Disadvantages:

- Difficult to handle **non-functional requirements**.
- **Partial specification**: many details need refinement during implementation (common in Agile).

Requirements Engineering

Goal:

- **Define and maintain requirements** throughout the project's life cycle.

Activities:

- **Elicitation**: Using methods like workshops, brainstorms, and focus groups.
- **Structuring**: Organizing requirements for clarity and ease of maintenance.
- **User Experience Design**: Designing interactions to enhance user satisfaction.
- **Validation**: Ensuring completeness and consistency of requirements.

Requirements Structuring

Goal:

- Enhance the **maintenance** and clarity of requirements over time.

Tools:

- Requirements should be **isolated** and easily **identifiable**.
- Organized and classified using frameworks like **FURPS** (Functionality, Usability, Reliability, Performance, and Supportability).
- **Annotated** for priority, importance, and traceability.

User Experience Design

Goal:

- Ensure a **coherent** and satisfying user experience across all software artifacts (e.g., design, interface, manuals).

Tools:

- **User-centered analysis**: Focus on how users will interact with the system through focus groups and experiments.
- **User-centered design**: Specify interactions via **mock-ups**.

Requirements Validation

Key Objectives:

- Address **inconsistencies**: Conflicts between requirements (e.g., one requirement contradicting another).
- Resolve **incompleteness**: Ensure all cases and scenarios are accounted for, including non-nominal situations.
- Eliminate **duplicates**: Avoid multiple descriptions of the same requirement in different forms.

Project Management (PM)-Relevant Activities:

- **Productivity and Size Metrics**: Measure coding efficiency and project scope.
- **Quality Metrics**: Assess code quality via bugs, defects, and improvements.
- Use of **coding and documentation standards** for consistency.
- **Code management practices** like version control and release standards.

Verification and Validation (V&V)

- **Validation**: Are we building the **right** system?
- **Verification**: Did we build the system **correctly**?
- V&V is a major aspect of **quality management**.
- **Testing** is the primary means of performing V&V in software systems.

Types of Testing

- **Unit Testing**:
    - Tests a small piece of code, such as a class.
- **Integration Testing**:
    - Examines interactions between components.

    - Example: The Mars Climate Orbiter bug resulted from different components using metric and imperial units, causing a $400M loss.
- **System Testing**:
    - Ensures that the system meets all requirements and behaves as expected.
    - Involves executing **test cases**.
- **Usability Testing**:
    - Verifies that the user experience is intuitive, effective, and satisfying.
    - Essential for **safety-critical systems** to reduce human error.

The System Testing Process

1. **Test Plan Definition**:
    - Based on the system's **requirements**.
2. **Test Case Creation**:
    - Specific scenarios to verify functionality.
3. **Test Execution**:
    - Actual testing phase to detect errors.
4. **Fixing**:
    - Address and resolve detected issues.
5. **Test Report**:
    - Document the outcomes of the tests, noting if any errors persist.

Deployment

Goal:

- **Installing** and making the new system **operational**.

Key Concerns:

- **Continuity of Business Operations**: Ensure minimal disruption during deployment.
- **Data Migration**: Transfer all relevant data to the new system.
- **Transition to Maintenance**: Move from development to operational support.

Factors to Consider:

- **Human Factor**: Are users ready and trained to use the system?
- **Data Factor**: Is all necessary data for the system available?
- **Hardware Factor**: Are interfaces functioning and ready for the new software?