EN 50128 is a standard that provides guidelines for the development of software for railway control and protection systems, emphasizing the importance of Verification and Validation (V&V) in ensuring software reliability and safety. This document outlines the processes, techniques, and documentation necessary for V&V, particularly for critical software applications.

Overview of EN 50128 Chapters

1. **Scope and Normative References**:

   - This section outlines the applicability of the standard and references other relevant documents that support compliance with EN 50128.

2. **Terms and Definitions**:

   - Key concepts related to software development and V&V are defined here, such as "verification," "validation," and "critical software."

3. **Software Lifecycle**:

   - The standard describes the software lifecycle phases, including requirements specification, design, implementation, testing, and maintenance. Each phase has specific V&V activities associated with it.

4. **V&V Process**:

   - This section highlights the necessity of V&V throughout the software lifecycle. It emphasizes that V&V should be planned and executed in conjunction with software development, ensuring that the software meets its requirements and performs its intended functions.

5. **Software Requirements Specification**:

   - This chapter details the requirements that software must fulfill. The Software Requirements Specification (SRS) must be clear, unambiguous, and complete, serving as the foundation for subsequent verification activities.

6. **Software Architecture Specification**:

 - This section outlines how the architecture of the software should be documented. A clear architecture aids in understanding interactions within the software, which is crucial for effective verification and validation.

7. **Software Design Specification**:

 - The Software Design Specification (SDS) serves as a bridge between the requirements and the implementation. This document must detail how the software will be structured to meet the SRS.

8. **Software Implementation**:

 - This chapter focuses on the coding phase, where the software is developed based on the design specifications. Coding standards and practices should be adhered to ensure quality.

9. **Verification and Validation Activities**:

 - This chapter discusses the specific activities required for V&V, including reviews, inspections, and testing. It emphasizes that V&V should be integral to all life-cycle stages, not merely a final step.

10. **Application Test Specification**:

 - The Application Test Specification outlines the tests to be performed to ensure the software meets its intended purpose. It must specify tests for data/algorithm integration and overall functionality.

11. **Validation and Assessment**:

 - This section describes how validation activities should assess the performance of the software throughout its lifecycle, ensuring compliance with all specified requirements.

Verification of Critical Software

Verification is the process of evaluating the software to ensure it meets its specified requirements at various stages of the development lifecycle. It encompasses several activities:

1. **Static Analysis**:

   - This involves analyzing the software without executing it to identify potential errors or vulnerabilities in the code. Tools may be used to perform automated static code analysis.

2. **Reviews and Inspections**:

   - Formal reviews of documents (SRS, SDS, etc.) are conducted to ensure completeness and correctness. Inspections involve peers examining the code or documentation to identify defects before moving to the next phase.

3. **Testing**:

   - Testing is a critical component of verification and includes several techniques:

     - **Unit Testing**: Testing individual components for correctness.

     - **Integration Testing**: Ensuring that combined components function correctly together.

     - **System Testing**: Testing the complete system against the requirements.

     - **Acceptance Testing**: Conducted to determine if the system meets the acceptance criteria.

Techniques for Software Verification

White Box Testing

White box testing is a verification technique that involves testing the internal structures or workings of an application, as opposed to its functionality (black box testing). Here’s how white box testing is typically carried out:

1. **Input**:

   - Gather various documents, including requirements, functional specifications, design documents, source code, and security specifications. This information provides the necessary context for the testing.

2. **Processing**:

   - Perform a risk analysis to guide the testing process. Develop a detailed test plan that outlines the specific tests to be executed and the criteria for success. Test cases should be designed to thoroughly exercise the code, targeting all possible execution paths to uncover potential errors.

   - Execute the test cases and record the results meticulously. The focus is on code coverage, ensuring that every statement, branch, and path has been tested.

3. **Output**:

   - Compile the results of the testing into a final report, which summarizes the testing process, outcomes, and any issues discovered. This report should also include recommendations for addressing any identified defects.

Additional Verification Techniques

- **Code Reviews**: Conduct peer reviews of the code to catch issues early in the development process.

- **Continuous Integration (CI)**: Implement CI practices to automate testing and ensure that new code changes do not introduce defects.

- **Formal Methods**: Utilize mathematical techniques for specification and verification to provide a high level of assurance in safety-critical systems.

Conclusion

The verification and validation of critical software as per EN 50128 is a comprehensive and systematic process that involves multiple stages and techniques. By following the guidelines in the standard and employing robust verification techniques such as white box testing, organizations can ensure that their software is reliable, safe, and compliant with regulatory requirements. The focus on early and continuous verification throughout the lifecycle is vital for the successful deployment of software in railway control and protection systems.