

# System Verification and Validation Plan Checklist

Spencer Smith

December 25, 2025

- Follows writing checklist (full checklist provided in a separate document)
  - L<sup>A</sup>T<sub>E</sub>X points
  - Structure
  - Spelling, grammar, attention to detail
  - Avoid low information content phrases
  - Writing style
- Follows the template, all parts present
  - Table of contents
  - Pages are numbered
  - Revision history included for major revisions
  - Sections from template are all present
  - Values of auxiliary constants are given (constants are used to improve maintainability and to increase understandability)
- Grammar, spelling, presentation
  - No spelling mistakes (use a spell checker!)
  - No grammar mistakes (review, ask someone else to review (at least a few sections))
  - Paragraphs are structured well (clear topic sentence, cohesive)

- Paragraphs are concise (not wordy)
- No Low Information Content (LIC) phrases ([List of LIC phrases](#))
- All hyperlinks work
- Every figure has a caption
- Every table has a heading
- Symbolic names are used for quantities, rather than literal values
- LaTeX
  - Template comments do not show in the pdf version, either by removing them, or by turning them off.
  - References and labels are used so that maintenance is feasible
- Overall qualities of documentation
  - Test cases include SPECIFIC input
  - Test cases include EXPLICIT output
  - Description over specification, when appropriate
  - Plans for what to do with description data (performance, usability, etc). This may involve saying what plots will be generated.
  - Plans to quantify error for scalar values using relative error
  - Plans to quantify error for vector and matrix values using a norm of an error vector (matrix)
  - Plans are feasible (can be accomplished with resources available)
  - Plans are ambitious enough for an A+ effort
  - Survey questions for usability survey are in an Appendix (if appropriate)
  - Plans for task based inspection, if appropriate
  - Very careful use of random testing
  - Specific programming language is listed
  - Specific linter tool is listed (if appropriate)
  - Specific coding standard is given
  - Specific unit testing framework is given

- Investigation of code coverage measuring tools
- Specific plans for Continuous Integration (CI), or an explanation that CI is not being done
- Specific performance measuring tools listed (like Valgrind), if appropriate
- Traceability between test cases and requirements is summarized (likely in a table)