

Design Notebook (DNB) Guidelines and Standards¹

These guidelines outline the design process and how each step should be documented. The required document organization and paragraph numbering is provided. *Remember to use the document template.* Reference and explain all figures. Use (required) this guideline together with the IEEE Std. 1016-1998 Recommended Practice for Software Design Descriptions and the Software Design handout which covers Chapters 6 and, 10 of the Budgen text.

- **Title page**
- **Abstract²**
- **Front-matter (table of contents for sections, figures and tables)**
- **Introduction**
 - ✓ Project purpose and goals
 - ✓ Design approach (SSA/SD is required)
 - ✓ Traceability approach
 - ✓ Background
 - ✓ Organization of this document
- **Requirements analysis (summarize major requirements)**
- **Design Representation**
 - ✓ Context diagram
 - ✓ Steps 1-2: Data flow diagrams (augmented with 1-3 P-specs)
 - ✓ Step 3: Transaction analysis
 - Divides a complete system DFD from Steps 1 and 2 into clusters or parts.
 - Purpose: separate a large system into a network of cooperating subsystems.
 - Output is to identify the event-stimulus/activity/response-effect transactions.
 - Transactions should correlate with the parts (e.g., withdrawal at an ATM).
 - This activity identifies important test cases that are used to verify a transaction is working correctly. Place a description of any such test cases in Appendix D.
 - ✓ Transform analysis
 - Non-hierarchical diagram with central transform (optional) called NHDFD
 - Structure chart
 - ✓ Design decision log (Optional [may contain summary remarks])³
 - ✓ References (all references must be cited in the main body of the test)
- **Glossary (may be part of the introduction subsection [but is preferred to be here])**
- **Appendix A: Data dictionary**
- **Appendix B: Project schedule⁴**
- **Appendix C: Requirements traceability (with DFD column completed)**
- **Appendix D: Identified test cases (optional)²**

¹ Standard (IEEE Std 1016-1987): This is the older standard, but its simpler and more applicable in an academic setting. Read ¶4 and ¶5 for information on content for the DNB. Read ¶6 for further ideas including the appendix to see an example of structure. Refer to IEEE Std 1016.1-1993 for more information on (¶7) on Design Methods and notations. A good example is contained there (¶7.1 and 7.2).

² Include in the abstract a complete overview of the product (purpose, goals, and design constraints) including your chosen design methodology, any exceptions to the stated requirements.

³ The *design decision log* and the *identified test cases* can gain your team a head start on the test report.

⁴ Give a key describing the symbology. Define milestones and deliverables. Describe the period of performance, and the items shown in each row of the PERT/GNAT chart. Enumerate all team members and their individual responsibilities.

Here is an example showing the required **numbering** sequence (based on SSA/SD [see Budgen text Chapter 10 and pages 97 - 111, and/or see Griffiths book for information about SSA/SD]):

1 Introduction

- 1.1 Project Purpose and Goals (problem description)
- 1.2 Design Approach (include brief outline of methodology and any tailoring)
- 1.3 Traceability Approach
- 1.4 Background
- 1.5 Organization of this document

2 Requirements Analysis (break out the functional part(s) and non-functional part)

- 2.1 Major Top-level Requirement A
 - 2.1.1 Associated Low-level requirement(s) (**optional**)
- 2.2 Major Top-level Requirement B
 - 2.2.1 Associated Low-level requirement(s) (**optional**)
- 2.3 Major Top-level Requirement C (and possibly others ...)
 - 2.3.1 Associated Low-level requirement(s) (**optional**)

3 Design Representation

- 3.1 Context Diagram (with description)
- 3.2 Level 1 Data Flow Diagram and P-specs (with description)

The P-Specs are difficult to organize. Use the "code" style that is defined in the document template. Use paragraph numbering with a title that is exactly the same as the process bubble name.
- 3.3 Level 2/3 Data Flow Diagrams (with P-specs as necessary)
- 3.4 Transaction Analysis (with description)
 - 3.4.1 Partitions of the System DFD

This is not a structure chart but a diagram showing clustered DFD bubbles that are related by their dependency on a particular transaction that will cause their functionality to be executed (i.e., a trace). This information provides the basis for identifying the transactions.
 - 3.4.2 Identified Transactions
- 3.5 Transform Analysis
 - 3.5.1 Non-hierarchical Data-Flow Diagram with Central Transform (**optional**)
 - 3.5.2 Structure Chart (**not optional**)
- 3.6 Design Decision Log (**contains major design decisions and rationale for each**)

4 References

5 Glossary

6 Appendix A: Data Dictionary (see the last page for the required format)

7 Appendix B: Project Schedule

8 Appendix C Requirements Traceability Matrix

9 Appendix D: Identified Test Cases

An Example Introduction:

This project primarily involved modifications to an existing program. Therefore, a tailored Structured System Analysis/Structured Design (SSA/SD) process was followed which included reverse engineering, as well as forward engineering activities. The reverse engineering activities involved analysis of the existing CSPN code and the development of a level 1 data flow diagram (DFD) for the existing system. A top-down SSA/SD process was then used to forward engineer only those parts of the existing CSPN system that required modification.

A context diagram was produced for the overall CSPN system, and additional level 1 and level 2 logical data flow diagrams (DFDs) were generated for any new functions, or functions that were significantly modified.

The existing legacy transactions were analyzed in terms of events, stimulus, activities, and response. Based on this analysis, a top-level structure chart was then generated for the CSPN system, and the modified functions were identified. The top-level structure chart was refined to identify the specific changes required to the modified functions using pseudocode and/or C code constructs. These code modifications were then integrated and tested with the existing legacy code.

SSA/SD Structured System Analysis and Structured Design Method Overview

Steps 1-2: Structured systems analysis (see Budgen Figure 10.5)

- **Level 0 = context diagram.**
- **Level 1 = top level DFD.**
- **Level 2 = explosion of level 1 DFD bubbles.**
- **Level 3 = use this level as appropriate.**

Step 3: Transaction analyses step and has five basic components:

- **The event in the systems environment that causes the transaction to occur;**
- **The stimulus that is applied to the system to inform it about the event;**
- **The activity that is performed by the system as a result of the stimulus;**
- **The response that this generates in terms of output from the system;**
- **The effect that this has upon the environment;**

You will need to identify a simple but comprehensive example of a transaction (see page 218 and Figure 10.8 of Budgen) that accounts for all of the five items described above. This will help you to define a good set of test cases. There should be 4 transactions identified. These will be your main starting points for the demonstration.

Step 4: Identify the central transform in the DFD:

You do not have to redraw the DFDs but if you add a “boss” bubble redraw showing where the boss fits. Which will allow you to develop a hierarchical structure chart. **Develop structure charts for all of your level 1-2 DFDs.**

Step 5: Merge the Structure Charts

The objective of this step is to produce a single structure chart (see Figures 10.23, 24 of Budgen).

The following template has been constructed for defining the data elements as shown in the following table:

Name:
Description:
Used in:
Units:
Range:
Data type:
Attribute:
Data store location:
Accuracy:

Name	This field gives the name of the variable used in the specification. The variable name used during coding must be the same as specified.
Description	This field gives a brief description of the variable.
Used in	This field gives a reference to the functional units using this variable.
Units	This field indicates the unit of measure for the data contained in the variable being defined.
Range	This field specifies the acceptable range of data values for the variable.
Data type	The data type field specifies the data type to be used when declaring the variable during coding.
Attribute	This field indicates whether or not the variable contains data, control information, or a data condition.
Data store location	This field references the common region where the variable must be stored.
Accuracy	This field dictates the degree of accuracy required for output comparisons to be made between implementations . In the data dictionary, accuracy is listed as N/A where accuracy is not applicable, or TBD where accuracy is <u>To Be Determined</u> later. A formal modification will be released when the values of the accuracy requirements have been approved.