SOFTWARE DESIGN SPECIFICATION
Outline and Requirements

General Comments: *The following guidelines are given in italics and reporesent the base minimim required to complete this plan. Non-italicized text is to be included in your plan verbatim. All sections must be included in your plan, even if you do not feel it applies.*

1. Scope

The general scope of the project is given in this section. Give one or two introductory paragraphs here.

1.1. System Objectives

Clearly state the overall objectives and subobjectives of the system of which this software project is a part. These objectives must be consistent with the objectives given in the Systems Requirements Document and the Software Requirements Document.

1.2. Hardware, Software and Human Interfaces

Clearly identify the scope of all interfaces between the software project described by this document and the external software programs, hardware ports, and human functions such as display screens and mouse inputs. Include a context diagram or figure to place your descriptions in context. Note that this section describes the scope of interfaces, not the interfaces themselves.

1.3. Major Software Functions

Using applicable diagrams and text, describe all major software functions that are a part of the software project described in this document. These functions should be described only to the extent that the functional scope of the project is made clear.

1.4. Externally Defined Database

Define and characterize any significant files, external databases or database systems used by the program. Level of description should be enough to establish the scope of interaction between the program and the database or database system.

1.5. Major Design Constraints and Limitations

Identify all design constraints and limitations against which the software will be defined. Repeat the constraints and limitations from the SRS and further refine their description at a design level.

2. Reference Documents

Identify all documents used throughout the design process. Reference documents include:

- *Mandatory compliance and guidance standards*
- *Documentation on all prior software, components, and libraries used in this project*
- *Significant CASE tools documents*
- *Sources of algorithms, methods, or significant processes used in developing or within the software itself*

Use a standard bibliography style of reference. Separate the listings of documents into the following categories:

2.1. Existing Software Documentation
2.2. System Documentation
2.3. Vendor Documents
2.4. Technical Reference

3. Design Description

This section contains the actual design descriptions of the program, subsystems, and each module. Use graphics and tabular representations augmented by clear and consise text. Cross-reference liberally.

3.1. Data Description

3.1.1. Review of Data Flow

Copy significant data flow diagrams from the SRS to this section. Explain the diagrams with text to ensure section stands alone (i.e., do not make references back to the SRS). At a minimum the level 0 and level 1 diagram must be shown and described.

3.1.2. Review of Data Structure

Copy from the SRS the data descriptions of all data id's shown in the figures, table, or text given in Section 3.1.1 above. Explain all data as necessary to ensure section stands alone without reference to the SRS.

3.2. Derived Program Structure

Give a sequence of heirarchical structure chart figures in decomposeable order each augmented with tables and text for added clarity. Note that the hierachical charts do not necessary reflect the same organization as the data flow diagrams in the SRS, but rather reflect a module-level structure of the program.

3.3. Interfaces within Structure

Completely identify all aspects of the major interfaces between structure elements in the program. Such interfaces may include data structures, timing diagrams, lists of data item id's, or other form of explanation. Note that each interface will include explanatory text, and should include graphical representations wherever possible. The depth of description should be enough that a complete module design can be given in the next section where the level of abstraction of the interface is no higher than that of the module design.

4. Module Design

This section should be organized by module. For each module present in detail the design of the module. The sections to be given for each module are:

Processing Narrative	*Describe the behavior of the module in text.*
Interface Description	*First give the calling syntax as **proc (arg, arg, ...)** then define each arg, proc type, and return type.*
Design Language Description	*Code the module using pseudo-code or formal design language.*
Modules Used	*Give a list of modules called by this module. Give a short phrase description of each module name on the list.*
Data Organization	*Completely describe each internal data item in the module using a formal data method (e.g., C types, ADA types, etc.).*
Comments	*Give in text all special notes or other items not included in the Processing Narrative section.*

This section can be quite large. Make up each section in a form-like format to make the task of preparing this section more structured.

5. File Structures and Global Data

Give a brief textual description of each file and global data items as a means of introducing this section.

5.1. External File Structures

For each file, give the following:

Logical Structure	*Describe the logical structure of the entire file graphically or tabularly. Use text as necessary to clarify the structure.*
Logical Record Description	*For each record type in the file, describe its strucuture and meanings of each field.*
Access Method	*Identify the method of access to the file (e.g., hash, indexed. pile, sequential, etc.).*

The use of formal data modelling methods is helpful, but not strictly necessary.

5.3 Global Data

Describe each global data item in turn, giving sufficient information for direct coding into data types in the implementation phase. The use of formal data modelling methods is helpful, but not required.

5.3. File and Data Cross Reference

In this section, give in tabular and/or graphical form the association between data items in the program and data items in each file. For instance if you have an input file where each line is read in turn and the data is parsed into a data structure called inline, then show how the line structure and fields maps into the data structure items. Also give any enumeration or range limits.

6. Requirements Cross Reference

This section of several tabular lists such as

- *Module versus requirement in the SRS*
- *Module data item versus data dictionary entry in the SRS*
- *File name versus file name in the SRS*
- *... and any other major association between the contents of the SRS and the design*

7. Test Provisions

This section refines the test plan information given in the SRS. Note that the tests themselves are not specified here (they are given in the Software Test Plan document).

7.1. Test Guidelines

Identify all guidelines and suggestions for test setup, environment, and application. This section applies primarily to the unit testing of each module or collection of units performing a function.

7.2. Integration Strategy

Give an outline of the procedure or sequence of tasks (including testing tasks) for integrating the modules into a working program. Ususally, each unit (i.e., module) is unit tested, then the modules are integrated into functional collections of modules and each is tested. Then these units combined into subsystems each of which are tested followed by a complete all-up program test. This section should give a "cookbook" step-by-step presentation of this process.

7.3. Special Considerations

This section includes further clarification on test development or application. For instance, if some special stubs are needed they can be further described here.

8. Packaging

This section describes how the executing segments of the program go together. If the program is segmented by overlays than section 8.1 applies. Other forms of program segmentation are allocation of modules or objects to nodes in a parallel processor, assignment of program sections to parts of memory (such as the TPA, high memory, or extended memory on a PC). If there are no special packaging requirements, so state.

8.1. Special Program Overlay Provisions

Graphically show how the program is partitioned for memory or processor allocation purposes. Use text for clarification.

8.2. Transfer Considerations

If the program is dynamically allocated during operation (e.g., VROOM by Borland Int.) describe its operation here. This section generally applies to programs operating over a suite of processors on a network.

9. Special Notes

Anything that doesn't fit above goes here.

10. Appendices

The Appendix contains copies of vendor or internal documents, large graphics that do not fit well in the body of the document, or other information gemain to the document but to unwieldy or unimportant to put in the body of the document.