System Development and Life-Cycle Management (SDLCM) Methodology

Subject	Type	Standard
As-Built System Documentation	Identifier	S–4151
	Effective Date	July 2002
	Revision No.	1

Approval _____

CISSCO Program Director

A. PURPOSE

This standard specifies the content and format requirements for As-Built System Documentation.

B. APPLICABILITY

This standard applies *only if* NRC or a contractor has developed an application system without the benefits that would have been derived by following the guidance of NRC's SDLCM Methodology and if that system is now being transitioned to a different NRC organization or contractor for life-cycle management. In such a situation, preparing the complete PDAD, LDD, and PDD after the application system has already been developed may not be useful. The As-Built System Documentation consolidates the requirements and design of an existing system in one product to facilitate the maintenance and future enhancement of the application system. This standard may *not* be used as a substitute for the PDAD, LDD, and PDD during the development of a new application system, because some material required for the successful design and development is not included in this standard.

The Technical Project Manager and members of the team responsible for system maintenance use this standard for documenting the requirements and design of a legacy system; key managers and quality assurance personnel use it for reviewing the as-built documentation; and the Executive Sponsor uses it when approving the document. The As-Built System Documentation is made available to all members of the maintenance or enhancement team, preferably in electronic form.

C. REFERENCE PUBLICATIONS

The following publications contain related information:

- SDLCM Methodology Handbook, Appendix E, Transition of Legacy Systems

- SDLCM Methodology Standard S–3051, System Requirements Specification
- SDLCM Methodology Standard S–3053, System Operations Concept
- SDLCM Methodology Procedure P–3111, Process Modeling
- SDLCM Methodology Standard S–3151, Data Models
- SDLCM Methodology Standard S–3161, Process Models
- SDLCM Methodology Standard S–3162, Context Diagrams
- SDLCM Methodology Standard S–3163, Data Flow Diagrams
- SDLCM Methodology Standard S–3164, External Systems Interface Diagrams
- SDLCM Methodology Standard S–3171, Logical Design Document
- SDLCM Methodology Standard S–3172, Physical Design Document
- SDLCM Methodology Standard S–3351, Data Dictionary
- *Systems Development CASE Tool Guidelines*, Systems Development and Integration Branch (SDIB) Office of Information Resources Management (OIRM), September 12, 1995
- *Standards and Conventions*, SDIB OIRM, August 28, 1995

D. STANDARD

The As-Built System Documentation consolidates the requirements and design of an existing system in one product to facilitate the maintenance and future enhancement of the application system. A documentation product built to comply with this standard contains all of the requirements and design information essential for successful maintenance and future enhancement activities. As the application system is modified through maintenance or enhancement, the As-Built System Documentation will also be maintained to reflect the modified requirements and design.

Tailor this standard as needed to be consistent with the size, scope, and complexity of the system. Add sections and subsections for special topics. Sections and subsections that are not applicable should *not* be deleted; they should indicate “not applicable.” Provide a brief statement to explain why a particular section is not applicable. When appropriate, reference other documents rather than repeating material unnecessarily.

The following paragraphs describe the content of the As-Built System Documentation.

1. INTRODUCTION

1.1 Background

Describe the activities that led to the need to create this As-Built System Documentation.

1.2 Objectives

Describe the objectives of creating this documentation and briefly state how the document will be used to support maintenance and enhancement activities.

1.3 Scope

Describe what this product covers (its scope). Describe what it does not cover (constraints are related to scope and are to be addressed in this subsection).

1.4 Assumptions

State what assumptions have been made about the application system, its requirements, and its design. Discuss the risks, if any, associated with those assumptions.

1.5 Applicable Documents

List any other documents that apply.

1.6 Overview

Discuss the organization of the As-Built System Documentation. Describe the basic content of each major section of the document. Describe how this document will be maintained as the application system evolves.

PART 1—REQUIREMENTS

Part 1 of the As-Built System Documentation specifies the requirements of the existing application system. The requirements will be updated as the system evolves through maintenance and enhancement activities. This section is needed to ensure that the design and the system tests can be traced back to requirements.

2. SYSTEM REQUIREMENTS SPECIFICATION (SRS)

The purpose of this section is to document the System Requirements Specification (SRS) on which the design of the system was based. An SRS specifies the functional, performance, operational, and programmatic requirements for a system as an entity.

State individual requirements in quantitative terms. Avoid the use of negative requirements. Consider testability, completeness, consistency, and conciseness when writing each requirement.

Number all requirements uniquely to facilitate traceability.

2.1 Functional Requirements

Identify all system function requirements ensuring that all requirements derived from the Systems and Operational Concept Document (SOC) and other user need and client requirements are captured**.**

Organize the requirements into functional areas. For each functional area, define, as appropriate:

- Input data
- Process(es) (algorithms, transformations, manipulations, calculations)
- Output data
- External interfaces
- Communications
- Special management information needs
- User Training and knowledge level (business and platform)

Use data requirements and system control as section headings, if appropriate.

Include a separate subsection for each functional area.

2.2 Performance Requirements

Decompose the performance requirements section into the following subsections, as appropriate:

- External workloads (for example, total size of data input per day)
- Internal function workloads (for example, function X must compute responses within 18 milliseconds)

- Throughput and response times (for example, complete simple query in the designated time period)
- Data quality, integrity, accuracy (for example, some elements must be accurate to a certain number of significant figures)
- Data retention (for example, number of years to retain data on-line versus long-term storage)
- System and communications capacity (for example, maximum number of communications lines connected to the system)
- Reliability, maintainability, and availability (for example, mean time between failures, mean time to repair)
- Periodicity, precision, simultaneity, frequency of transactions, processing, operations (for example, the system must process 10,000 transactions per second)
- Human workload and performance (for example, generation of a designated number of reports per day)
- Growth, flexibility, and expandability (for example, the system shall be capable of supporting four additional interfaces)
- Fault detection and isolation
- Backup and Recovery (period of retention, etc.)

2.3 Operational Requirements

Decompose the operational requirements section into the following subsections, as appropriate:

- Human factors (for example, user interface, keyboard layouts, display panel design, display formats).
- Environment (for example, temperature, humidity, air conditioning, shock and motion, shelf life)
- System monitoring
- Configuration control (for example, automated configuration control software)
- Training (for example, operator training)
- Support capabilities (for example, checkpoint and restart)
- Maintenance (for example, preventive maintenance)

- Logistics (for example, tracking of magnetic tapes)
- Facilities (for example, government-furnished equipment, test equipment, space, environmental such as heating and air conditioning)
- Safety (for example, fire, grounding of electrical equipment, requirements intended to prevent personnel injury)
- Security (for example, facility access, password protection, software, and data security requirements)
- Geographic location (for example, multiple sites)
- Documentation (for example, users or operations guides)

2.4 Programmatic Requirements

Decompose the programmatic requirements section into the following subsections, as appropriate:

- Development facility
- Maintenance support requirements or constraints
- Special test requirements
- Installation, phase-in, and turnover
- Schedule and budget constraints
- Growth, flexibility, and expandability
- Development standards
- Procurement activities
- Shipping and storage requirements

2.5 Special Requirements

Specify any system requirements that are not appropriate for inclusion in any of the previous sections.

2.5.1 ACCESSIBILITY REQUIREMENTS FOR INDIVIDUALS WITH DISABILITIES.

2.5.2 RECORDS MANAGEMENT REQUIREMENTS.

3. DATA REQUIREMENTS

Define the data requirements of the Project, including

- A list of entities, that is, things that you want to keep information about
- The entity definitions, that is, attributes of the entities
- A context diagram to explain the relationships among the entities.

3.1 Entity List and Definitions

Identify those top-level entities required by the Project. Include only those entities shown in the context diagram and known at the functional level. (Note that the SRS Document defines initial Project requirements and analyses of alternative solutions. It is not a system requirements analysis and design document. A complete list of entities will be created and documented in the Logical Design Document, S–3171, as a product within Component 3, Design the Solution.)

Provide basic information for each top-level entity as an entity definition:

- Name
- Identifier (primary key)
- List of included subtypes, if any
- Description (several sentences)
- Average volume at conversion
- Annual growth percentage
- Active life
- Estimated size in characters (preliminary)
- Source of the data for the entity (existing or to be created) or method of derivation from existing or to-be-created source(s)

Compare the entity list with the Enterprise Model to ensure consistency and to avoid conflict.

3.2 Context Diagram

Provide a pictorial view of the entities and the relationships among the entities. (See SDLCM Methodology Standard S–3162, Context Diagrams.)

For each entity in the context diagram, provide:

- Name
- Description (one or two sentences)
- Cardinality

PART 2--LOGICAL DESIGN

Part 2 of the As-Built System Documentation specifies the logical design of the existing application system. The logical design will be updated as the system evolves through maintenance and enhancement activities.

4. APPROACH

4.1 Design Methods

Provide a high-level summary of the method(s) used to design the system. Describe any constraints on the design.

Discuss how the design method accounts for the use or integration of commercial off-the-shelf (COTS) products. Specifically, identify and discuss any directives or strategies for using COTS products. Specify COTS packages documented, analyzed, enhanced, or modified.

4.2 Design Alternatives

If applicable, provide a brief description of alternative design architectures and decomposition considered, including the one selected. Discuss the results of the analyses of candidate designs and the criteria used for selection.

4.3 Design Studies

If applicable and known, provide information on analysis, modeling, and trade studies performed during the original logical design effort or as a part of maintenance or enhancement redesigns. Show how the results of the studies (such as workload; performance; reliability, maintainability, and availability; and database management system) demonstrated that the logical design would meet requirements. List any assumptions used and present the results.

4.4 Design Issues

This section is applicable only when prior As-Built System Documentation is being updated for a maintenance change.

Provide an overview of unresolved issues in the logical design. Discuss any to-be-determined (TBD) issues in the requirements and interfaces, their status, and what steps are planned or need to be taken to resolve them.

Discuss any risks or uncertainties in the hardware and software design. Describe the studies and prototyping efforts planned to resolve them during physical design.

5. LOGICAL MODEL

Provide an overview of the design of the software and hardware elements of the solution. Discuss the external interfaces to the system. Refer to any agreements concerning external interfaces. Provide a brief description of the high-level processes or subsystems that make up the logical design of the solution. Note: These are the processes that are identified in the Level 1 data flow diagram.

5.1 System Architecture

Provide a high-level description of the architecture, an end-to-end data flow diagram, and a diagram(s) that identifies the internal and external interfaces for the logical design. Include a context diagram. Include any applicable interfaces to corporate data, legacy systems, and users in other organizations or locations.

If applicable, describe the characteristics of the computer network, network components, and operating systems to be used.

5.2 Logical Data Model

Provide a logical data model of the existing system, including the identification of corporate data that must be accessed or manipulated by the system, any data that must be added to the corporate databases, capacity and archiving requirements for the data, and data security requirements.

5.2.1 ENTITY RELATIONSHIP DIAGRAM

Include a copy of the system's entity relationship diagram (ERD) and the entity descriptions contained in the data dictionary. If the system depends on a computer aided software engineering (CASE) tool, refer to the on-line repository or encyclopedia where the logical data model and dictionary are stored and updated as needed.

If the CASE tool uses any conventions different from those described in Section 3.2, Data Model, of the *Systems Development CASE Tool Guidelines*, or Section 3, Logical Model Object Names, of the *Standards and Conventions* document, identify the conventions used and document the approval to use the tool.

5.2.2 DATA MAPPING AND NAVIGATION TO LEGACY SYSTEMS AND SCREENS

Use a table to document the mapping between system data entities and corporate or legacy system data entities. Identify how the system acts on (reads, manipulates, updates) these data entities.

Specify how the system navigates to legacy systems to obtain needed data. If the system uses special screens to access the legacy data, refer to the as-built screens or provide a first-cut design of the screens for a maintenance change.

Identify data entry and display requirements and provide the design for these items. Discuss screen and dialog performance requirements that are satisfied by the design.

Note: The navigation process and any new screens or displays will be designed in greater detail as a part of physical design.

5.3 Logical Process Model

Discuss the process modeling activities, including the business processes and practices or decision-making processes that are automated and any transaction performance requirements, such as volume, speed, and frequency of transactions.

Copies of the system's data flow diagrams, also known as logical process models, and the process descriptions are included in the subsystem-by-subsystem description contained in Section 5 of this document, so this section should refer the reader to Section 5 and to the on-line repository or encyclopedia where this information is stored and updated.

If the CASE tool used by the project uses any conventions different from those described in Section 3.3, Process Model, of the *Systems Development CASE Tool*

Guidelines, or Section 3, Logical Model Object Names, of the *Standards and Conventions* document, identify the conventions used and the project's approval to use the tool.

6. SYSTEM DESCRIPTION

Include a copy of the Level 1 Data Flow Diagram (DFD) of the logical process model. For each process described in the Level 1 DFD, identify subsystems and configuration items (CIs), both those that are implemented in software (including COTS products) and those that are implemented in hardware or firmware. Provide high-level architectural hierarchy or other block diagrams that show the relationships among the subsystems. Provide a brief, high-level narrative description of these diagrams that identifies the hardware and software subsystems and CIs, the functions they perform, and how they interact. For a software subsystem based on a COTS product, discuss the subsystem in terms of the COTS product as a whole, rather than in terms of the processes provided within the COTS product. The narrative information may be provided in bulleted lists or a table as appropriate.

For each process, provide a subsystem description in the format shown below.

6.1 Subsystem Descriptions

6.1.1 NAME OF SUBSYSTEM 1

Provide a functional description of the subsystem and a high-level review of the requirements met by the subsystem. Use the subsystem's process description. Include a copy of the DFD(s) that decompose this subsystem to show the software CIs contained in the subsystem. If applicable, provide a block diagram showing the hardware CIs contained in the subsystem. Discuss the interfaces between this subsystem and each of the other subsystems in the system. Discuss the interfaces among the CIs in the subsystem.

6.1.1.1 SOFTWARE Configuration Items

Provide a summary of the software CIs.

6.1.1.1.1 Software CI 1 Description

Provide a functional description of the software CI and a high-level review of the requirements met. Use the CI process description. Identify the source of the CI and whether it is developed, purchased, or acquired from a combination of sources. List files and databases accessed by the CI.

5.1.1.1.*n* Software CI *n* Description

Provide the same information for each additional software CI in subsystem 1 as for software CI 1.

6.1.1.2 HARDWARE Configuration Items

Provide a summary of the hardware CIs.

6.1.1.2.1 Hardware CI 1 Description

Provide a functional description of the hardware CI and a high-level review of the requirements met. Identify the source of the CI and whether it is to be developed, purchased, or acquired from a combination of sources.

5.1.1.2.*n* Hardware CI *n* Description

Provide the same information for each additional hardware CI in the subsystem as for hardware CI 1.

5.1.*N* SUBSYSTEM *N* DESCRIPTION

Provide the same information for each additional subsystem as for subsystem 1.

6.2 External Interfaces

Provide a logical description of each external interface (for example, interfaces to the network topology, another system, subsystem, or CI). Identify the name of the interface, its type (for example, hardware interface, data file), purpose, and frequency of interchange. Describe the structure and organization of the data. Reference the system's data model to provide the format of the data components, including such items as data element type, description, representation, range of values, and units. Estimate the amount of data crossing the interface and estimate the imposed data storage requirements.

If applicable, include an external systems interface diagram (see standard S–3164).

If the logical description of the external interfaces will not be useful for system maintenance, this section is not applicable. In that case, provide only the physical description in Section 7.1.4.

6.3 Internal Interfaces

Provide a logical description of each interface between and among the components of a CI. Identify the name of the interface and its type, purpose, and frequency of interchange. Describe the organization of the data (for example, record and block structure). Define the preliminary format of the data components, including data element type, description, representation, range of values, units, etc.

If the logical description of the internal interfaces will not be useful for system maintenance, this section is not applicable. In that case, provide only the physical description in Section 7.1.5.

7. SYSTEM OPERATIONS DESIGN

Provide an operational overview of the system.

7.1 Operations Scenarios

Discuss the design in terms of the operational scenarios for each major product produced. Show how the elements of the design work together to facilitate the production of the product. Discuss how the design meets operational requirements such as performance and data quality.

7.2 User-System Interface

Provide a description of the look and feel of the user or system interface. Discuss how the users will interact with the system to perform various functions.

PART 3—PHYSICAL DESIGN

8. PHYSICAL MODEL

8.1 System Architecture

8.1.1 SYSTEM COMPONENTS

Describe the physical characteristics of the network, client-server, or desktop environment on which the application system has been deployed. Include discussion of environmental constraints in terms of external communications, product compatibility or limitations, and any other technical issues.

For network architectures, characterize the network with respect to single- or multi-user capabilities and client-server architecture. Describe the network topology, connectivity, and protocols if they are significant considerations for the physical design. If applicable, describe physical security measures that are provided.

For client-server architectures, describe the physical characteristics of the file server system. Describe the different types of clients, and provide the configurations for each in terms of hardware and software. Identify any modifications or enhancements that were made.

Describe any additional servers such as those used for application and database services. For each additional server, describe its interface to the network and its security processes.

Describe printers, scanners, and other peripheral or physical components that are used with the application system.

8.1.2 SOFTWARE COMPONENTS

Describe the software components of the system and the functions that each provides.

Use structure charts or object-oriented diagrams expanded to the unit level, showing data flow, control, input, and output.

Describe the external interfaces of the software components; that is, the application interfaces for interprocess communication or data sharing that must be considered when replacing or upgrading a component.

For each COTS product, indicate if the product is used as-is to satisfy part or all of a requirement (or requirements), or describe how the product was customized or modified. Customization or modification methods include:

- Use configuration capabilities of the COTS product
- Use the Application Programming Interface (API) kits of the COTS products
- Create new functionality through the use of COTS programming tools
- Add and integrate additional COTS products

8.1.3 USER INTERFACE

Briefly describe how a user accesses and interfaces with the application system. For example, discuss the use of a common user interface, if applicable, to integrate desktop applications for word processing, spreadsheets, electronic mail, and database applications. Describe the types of security restrictions that are in place at the user interface.

Provide detailed descriptions of processing initiated by operator-specified input and actions in terms of control points, functions performed, and results obtained. Provide a listing of numbered error messages with descriptions of system and user actions.

Describe critical components of the interfaces such as the need for synchronization of operations (for example, data read and write) by either user data entry or automatic means.

8.1.4 EXTERNAL INTERFACES

Provide a description of each external interface (for example, interfaces to the network topology, another system, subsystem, or configuration item). Identify the name of the interface, its type (for example, hardware interface, data file), purpose, and frequency of interchange. Describe the structure and organization of the data. Reference the system's data model to provide the format of the data components, including such items as data element type, description, representation, range of values, and units. Estimate the amount of data crossing the interface and estimate the imposed data storage requirements.

If applicable, refer to or include an external interface diagram for each interface. (See

SDLCM Methodology Standard S–3164, External Systems Interface Diagrams.)

8.1.5 INTERNAL INTERFACES

Provide a description of each interface between and among the components of a configuration item. Identify the name of the interface and its type, purpose, and frequency of interchange. Describe the organization of the data (for example, record and block structure). Define the format of the data components, including data element type, description, representation, range of values, units, etc.

8.2 Physical Data Model

7.2.1 GENERAL MAPPING OF THE LOGICAL MODEL OBJECTS TO THE PHYSICAL MODEL OBJECTS

Describe how the logical model was translated to the physical model. Describe any intermediate data structures that were generated and how they were generated. For example, if a tool was used, specify what tool and version of that tool was used.

For example, Table 4151–1 shows how the Logical Model Objects were translated to Physical Model Objects—the relational schema.

Table 4151–1. Logical to Physical Model Object Comparison

Logical Design Logical Model Objects	Physical Design Relational Schema
Entity Types	Relational Tables
Attribute Types	Columns
Relationship Types	Relational Schema Relationships
Information Types	Data Types
Subtype Set	Data Subtypes

If the general mapping of the logical to the physical model will not be useful for system maintenance, this subsection is not applicable.

8.2.1 SPECIFIC MAPPING OF THE LOGICAL MODEL OBJECTS TO THE RELATIONAL SCHEMA

List the relational schemas and their components, including:

- Relational tables that represent each entity
- Columns that represent attributes
- Relational schema relationships
- Database primary, foreign, and secondary keys
- Data types

Show how the relational schema components map back to the logical design using tables such as those shown in Tables 4151–2 through Table 4151–4.

Table 4151–2. Mapping of Entity and Attribute Types to Relational Tables and Columns

Logical Design Entity and Attribute Types	Physical Design Relational Tables and Columns
Entity Type 1	Relational Table 1
Attribute Type A	Column A, Primary Key
Attribute Type B	Column B
Entity Type 2	Relational Table 2
Attribute Type C	Column C, Primary Key
Attribute Type D	Column D
Attribute Type E	Column E
Entity Type *n*	Relational Table *n*

Table 4151–3. Mapping of Relationship Types to Relational Schema Relationships

Logical Design Relationship Types	Physical Design Relational Schema Relationships
Relationship Type 1	Relational Schema Relationship 1
Relationship Type 2	Relational Schema Relationship 2
Relationship Type *n*	Relational Schema Relationship *n*

Table 4151–4. Mapping of Information Types to Data Types

Logical Design Information Types and Subtype Sets	Physical Design Data Types and Data Subtypes
Information Type 1	Data Type 1

Subtype Set A	Data Subtype A
Subtype Set B	Data Subtype B
Information Type 2	Data Type 2
Subtype Set C	Data Subtype C
Subtype Set D	Data Subtype D
Subtype Set E	Data Subtype E
Information Type *n*	Data Type *n*

8.2.2 COMPLETING THE PHYSICAL DESIGN

Describe how the physical design was completed after the logical model was translated into the physical model. Include:

- Views specified
- Indexes identified
- Data structures that have been redefined for efficiency
- Column names that have been changed for clarity

8.2.3 DATA DEFINITION LANGUAGE

List the Data Definition Language (DDL) that was generated from the relational schema. Include data dictionary information, such as:

- Tables or view of tables and their definitions
- Columns for the tables or views showing:
  - ♢ Column names
  - ♢ The column's data type format and length
  - ♢ The column's definition
  - ♢ The column's data type set definition
  - ♢ The column's data type set comments

8.3 Physical Process Model

Document the process models that evolved from the logical design, including:

- Process Flow Diagrams
- Screen prototypes

8.3.1 PROCESS FLOW DIAGRAMS

Use process flow diagrams (See in SDLCM Methodology Procedure P–3111, Process Modeling) to document all automated processes. Expound on ideas from the logical model's data flow diagrams and process definitions.

8.3.2 SCREEN PROTOTYPES

Document the screen prototypes that evolved from the logical model's:

- Data flow diagrams
- Process Definitions
- Data flow view

For each screen, specify:

- Input fields
- Processing
- Output fields
- Dialog boxes
- Triggers

8.3.2.1 Screen Prototype 1

8.3.2.2 Screen Prototype 2

7.3.2.*N* Screen Prototype *N*

APPENDIX A. REQUIREMENTS-TO-DESIGN TRACEABILITY

Include a copy of the requirements-to-design traceability matrix, or refer to the on-line traceability matrix.

APPENDIX B. DATA DICTIONARY

Provide a copy of the system's data dictionary, or refer to the on-line data model and data dictionary.

APPENDICES. (IF NEEDED)

If design analysis, modeling, trade studies results, or other data are too detailed to include in the body of this document, include the information as appendices.

ACRONYMS

List and define all acronyms used in the As-Built System Documentation.

REFERENCES

List all cited references.