For instructions on using this template, please see Notes to Author/Template Instructions on page 21. Notes on accessibility: This template has been tested and is best accessible with JAWS 11.0 or higher. For questions about using this template, please contact CMS IT Governance. To request changes to the template, please submit an XLC Process Change Request (CR).

Centers for Medicare & Medicaid Services
CMS eXpedited Life Cycle (XLC)

<Project Name / Acronym>

System Design Document

Version X.X

MM/DD/YYYY

Document Number: <document's configuration item control number>

Contract Number: <current contract number of company maintaining document>

Table of Contents

List of Figures

No table of figures entries found.

List of Tables

1. Introduction

Instructions: Provide identifying information for the existing and/or proposed automated system or situation for which the SDD applies (e.g., the full names and acronyms for the development project, the existing system or situation, and the proposed system or situation, as applicable), and expected evolution of the document. Also describe any security or privacy considerations associated with use of this document.

The System Design Document (SDD) describes how (1) the functional and nonfunctional requirements recorded in the Requirements Document, (2) the preliminary user-oriented functional design recorded in the High Level Technical Design Concept/Alternatives document, and (3) the preliminary data design documented in the Logical Data Model (LDM) are transformed into more technical system design specifications from which the system will be built. The SDD is used to document both high-level system design and low-level detailed design specifications.

The SDD describes design goals and considerations, provides a high-level overview of the system architecture, and describes the data design associated with the system, as well as the human-machine interface and operational scenarios. The high-level system design is further decomposed into low-level detailed design specifications for each of the system's components, including hardware, internal communications, software, system integrity controls, and external interfaces. The high-level system design serves as primary input to the Preliminary Design Review (PDR). The low-level detailed design serves as input to the Detailed Design Review (DDR).

1.1 Purpose of the System Design Document

Instructions: Provide the purpose of the System Design Document. This document should be tailored to fit a particular project's needs.

The System Design document documents and tracks the necessary information required to effectively define architecture and system design in order to give the development team guidance on architecture of the system to be developed. Design documents are incrementally and iteratively produced during the system development life cycle, based on the particular circumstances of the IT project and the system development methodology used for developing the system. Its intended audience is the project manager, project team, and development team. Some portions of this document such as the user interface (UI) may on occasion be shared with the client/user, and other stakeholder whose input/approval into the UI is needed.

2. General Overview and Design Guidelines/Approach

This section describes the principles and strategies to be used as guidelines when designing and implementing the system.

2.1 General Overview

Instructions: Briefly introduce the system context and the basic design approach or

organization. Provide a brief overview of the system and software architectures and the design goals. Include the high-level context diagram(s) for the system and subsystems previously provided in the High-Level Technical Design Concept/Alternatives and/or Requirements Document, updated as necessary to reflect any changes that have been made based on more current information or understanding. If the high-level context diagram has been updated, identify the changes that were made and why.

2.2 Assumptions / Constraints / Risks

2.2.1 Assumptions

Instructions: Describe any assumptions or dependencies regarding the system, software and its use. These may concern such issues as: related software or hardware, operating systems, end-user characteristics, and possible and/or probable changes in functionality.

2.2.2 Constraints

Instructions: Describe any global limitations or constraints that have a significant impact on the design of the system's hardware, software and/or communications, and describe the associated impact. Such constraints may be imposed by any of the following (the list is not exhaustive):

a) Hardware or software environment
b) End-user environment
c) Availability or volatility of resources
d) Standards compliance
e) Interoperability requirements
f) Interface/protocol requirements
g) Licensing requirements
h) Data repository and distribution requirements
i) Security requirements (or other such regulations)
j) Memory or other capacity limitations
k) Performance requirements
l) Network communications
m) Verification and validation requirements (testing)
n) Other means of addressing quality goals
o) Other requirements described in the Requirements Document

2.2.3 Risks

Instructions: Describe any risks associated with the system design and proposed mitigation strategies.

2.3 Alignment with Federal Enterprise Architecture (FEA)

Instructions: Describe alignment with FEA.

3. Design Considerations

Instructions: Describe issues which need to be addressed or resolved before attempting to devise a complete design solution.

3.1 Goals and Guidelines

Instructions: Describe any goals, guidelines, principles, or priorities which dominate or embody the design of the system and its software. Examples of such goals might be: an emphasis on speed versus memory use; or working, looking, or "feeling" like an existing product. Guidelines include coding guidelines and conventions. For each such goal or guideline, describe the reason for its desirability unless it is implicitly obvious. Describe any design policies and/or tactics that do not have sweeping architectural implications (meaning they would not significantly affect the overall organization of the system and its high-level structures), but which nonetheless affect the details of the interface and/or implementation of various aspects of the system (e.g., choice of which specific product to use).

3.2 Development Methods & Contingencies

Instructions: Briefly describe the method or approach used for the system and software design (e.g., structured, object-oriented, prototyping, J2EE, UML, XML, etc.). If one or more formal/ published methods were adopted or adapted, then include a reference to a more detailed description of these methods. If several methods were seriously considered, then each such method should be mentioned, along with a brief explanation of why all or part of it was used or not used. Describe any contingencies that might arise in the design of the system and software that may change the development direction. Possibilities include lack of interface agreements with outside agencies or unstable architectures at the time the SDD is prepared. Address any possible workarounds or alternative plans.

3.3 Architectural Strategies

Instructions: Describe any design decisions and/or strategies that affect the overall organization of the system and its higher-level structures. These strategies should provide insight into the key abstractions and mechanisms used in the system architecture. Describe the reasoning employed for each decision and/or strategy (possibly referring to previously stated design goals and principles) and how any design goals or priorities were balanced or traded-off.

Describe compliance with CMS Enterprise Architecture (EA) and standards. Specifically identify any deviations that were made from the CMS EA and standards, and provide rationale to support the deviation(s). When describing a design decision, discuss any other significant alternatives that were considered, and the reasons for rejecting them (as well as the reasons for accepting the alternative finally chosen). Sometimes it may be most effective to employ the "pattern format" for describing a strategy.

Examples of design decisions might concern (but are not limited to) things like the following:

a) Use of a particular type of product (programming language, database, library, commercial off-the-shelf (COTS) product, etc.)
b) Reuse of existing software components to implement various parts/features of the system
c) Future plans for extending or enhancing the software
d) User interface paradigms (or system input and output models)
e) Hardware and/or software interface paradigms
f) Error detection and recovery
g) Memory management policies
h) External databases and/or data storage management and persistence
i) Distributed data or control over a network
j) Generalized approaches to control
k) Concurrency and synchronization
l) Communication mechanisms
m) Management of other resources

3.4 Performance Engineering

A contributing factor to System Design will be Performance Requirements.

Performance Requirements are the defined scalability or responsiveness expectations of specific workloads that process on a system.

Instructions:

(a) Using the Performance Requirements defined in the Requirements Document, provide a detailed explanation that describes how the Performance Requirements were incorporated into the system's design. Please refer to Sections 2.0 of the CMS Performance Test Plan and Results Template for guidance on defining Performance Requirements.

(b) Start preparing Production Load Model(s) in preparation for Performance testing. Please refer to Sections 2.1.1 of the CMS Performance Test Plan and Results Template for guidance on Load Model construction.

4. System Architecture and Architecture Design

This section outlines the system and hardware architecture design of the system that is being built.

Instructions: Describe the system architecture, how the application interacts with other applications. Not necessarily how the application itself works but, how the appropriate data is correctly passed between applications. Provide an overview of how the

functionality and responsibilities of the system were partitioned and then assigned to subsystems or components. Don't go into too much detail about the individual components themselves in this section. A subsequent section of the SDD will provide the detailed component descriptions. The main purpose here is to gain a general understanding of how and why the system was decomposed, and how the individual parts work together to provide the desired functionality.

At the top-most level, describe the major responsibilities that the software must undertake and the various roles that the system (or portion of the system) must play. Describe how the system was broken down into its components/subsystems (identifying each top-level component/subsystem and the roles/responsibilities assigned to it). Describe how the higher-level components collaborate with each other in order to achieve the required results. Provide some sort of rationale for choosing this particular decomposition of the system (perhaps discussing other proposed decompositions and why they were rejected).

Make use of design patterns whenever possible, either in describing parts of the architecture (in pattern format), or for referring to elements of the architecture that employ them. Provide rationale for choosing a particular algorithm or programming idiom (or design pattern) to implement portions of the system's functionality.

4.1 Logical View

Instructions: Insert any related logical views or provide a reference to where they are stored.

4.2 Hardware Architecture

Instructions: Describe the overall system hardware and organization, indicating whether the processing system is distributed or centralized. Identify the type, number, and location of all hardware components including the presentation, application, and data servers and any peripheral devices (e.g., load balancers, SSL accelerator, switches, firewalls, with a brief description of each item and diagrams showing the connectivity between the components along with required firewalls, ports, and network bands utilized (e.g., management band). Include resource estimates for processor capacity, memory, on-line storage, and auxiliary storage.

4.2.1 Security Hardware Architecture

Instructions: Describe the hardware components and configuration supporting the security and privacy of the system. Specify the architecture for (1) authentication to validate user identity before allowing access to the system, including the use of IACS/EUA or other type of Identity Vetting & Authentication system;(2) authorization of users to perform functional activity once logged into the system, (3) encryption protocol to support the business risks and the nature of information, and (4) logging and auditing design, if required. The design should be based on the designated system security level and provide adequate protection against threats and vulnerabilities.

4.2.2 Performance Hardware Architecture

Instructions: Describe the hardware components and configuration supporting the performance and reliability of the system. Identify single points of failure and, if relevant, describe high availability design (e.g., clustering).

4.3 Software Architecture

Instructions: Describe all software that is needed to support the system, the hardware component for which each software component is targeted, and specify the physical location of all software systems. List such things as logical components (e.g., JSP in presentation layer, JNDI in application layer, EJB and JDBC in data layer), database platforms, computer languages, compilers, utilities, operating systems, communications software, programming computer-aided software engineering tools, commercial off-the-shelf (COTS) software, open source frameworks, etc, with a brief description of the function of each item and any identifying information such as manufacturer, version number, number and types of licenses needed, etc., as appropriate. Identify all Computer Software Configuration Items (CSCIs), Computer Software Components (CSCs) and Application Programming Interfaces (APIs) to include name, type, purpose and function for each; the interfaces, messaging, and protocols for those elements; and rationale for the software architectural design. Include software modules that are functions, subroutines, or classes. Use functional hierarchy diagrams, structured organization diagrams (e.g., structure charts), or object-oriented diagrams that show the various segmentation levels down to the lowest level. All features on the diagrams should have reference numbers and names.

Include a narrative that expands on and enhances the understanding of the functional breakdown. If necessary, describe how a component was further divided into subcomponents, and the relationships and interactions between the subcomponents. Proceed into as many levels/subsections of discussion as needed in order to provide a high-level understanding of the entire system or subsystem, leaving the details for inclusion in a later section of the SDD. Include data flow diagrams that conform to appropriate standards (e.g., Yourdon-Demarco conventions) and provide the physical process and data flow related to the logical process and data flow decomposed to the primitive process level (describing how each input is processed/transformed into the resulting output). If there are parts of the system that already existed before this development effort began, then only describe the relationships and interactions between the old parts and the new parts. Pre-existing parts that are modified or enhanced need to be described only to the extent that is necessary to provide a sufficient understanding of the nature of the changes that are being made.

4.3.1 Security Software Architecture

Instructions: Describe the software components and configuration supporting the security and privacy of the system. Specify the architecture for (1) authentication to validate user identity before allowing access to the system, including the use of IACS/EUA or other type of Identity Vetting & Authentication system;(2) authorization of users to perform functional activity once logged into the system, (3) encryption protocol

to support the business risks and the nature of information, and (4) logging and auditing design, if required. The design should be based on the designated system security level and provide adequate protection against threats and vulnerabilities.

4.3.2 Performance Software Architecture

Instructions: Describe the software components and configuration supporting the performance and reliability of the system. Identify single points of failure and, if relevant, describe high availability design (e.g., clustering).

4.4 Information Architecture

Instructions: Describe the information that will be stored in the system (e.g. beneficiary information, claim data, etc.) Identify if any of the information is personally identifiable information (PII), individually identifiable information (IIF), or personal health information (PHI).

4.4.1 Records Management

Federal regulations issued by NARA are outlined at 36 CFR – Subchapter B –Records Management. Business Owners must contact OSORA to initiate the records management process.

4.4.1.1 Identify all data (as well as the format of the data—paper, manual input, electronic data) supplied to the system as well as who/what is supplying the data.

4.4.1.2 Provide instructions on what happens to the manual/electronic inputs after they are entered into the master file/database and are verified.

4.4.1.3 Master Files – Provide a detailed description of the data maintained in the system/database.

4.5 Internal Communications Architecture

Instructions: Provide a detailed description of the system's communications network, denoting the communications architecture(s) being implemented (e.g., Token Ring, Ethernet, etc.) and how system components are linked. Describe any Local or Wide Area Networks and buses. Include descriptions of necessary equipment (e.g., hubs, routers, cabling, transmitters, firewalls, ports, etc.). Provide a diagram depicting the communications flow between the system and subsystem components. Include resource estimates for communications network capacity (LAN and WAN) needed to install and execute each application on each platform.

4.6 Security Architecture

Instructions: Insert any related security architecture documents, including integrity controls, or provide a reference to where they are stored.

4.7 Performance

Instructions: Insert any performance documents or provide a reference to where they are stored.

4.8 System Architecture Diagram

Instructions: Using the hardware, software, communications, and information designs described above, depict the overall, integrated structure of the system in terms of presentation, application, and data regions.

5. System Design

5.1 Business Requirements

Instructions: Insert any related project business requirements or provide a reference to where they are stored.

5.2 Database Design

Instructions: Describe the design of all database management system (DBMS) files and non-DBMS files associated with the system. Provide a comprehensive data dictionary showing data element name, type, length, source, validation rules, maintenance (create, read, update, delete (CRUD) capability), data stores, outputs, aliases, and description. The Data Design information can be included as an appendix or recorded in a separate Database Design Document (DDD), as appropriate, which would be referenced here. A template for a DDD is available from the CMS Integrated IT Investment & System Life Cycle Framework website located at https://www.cms.gov/Research-Statistics-Data-and-Systems/CMS-Information-Technology/SystemLifecycleFramework/index.html

5.2.1 Data Objects and Resultant Data Structures

Instructions: For each functional data object, specify the data structure(s) which will be used to store and process the data. Describe any data structures that are a major part of the system, including major data structures that are passed between components. List all functions and function parameters. For functions, give function input and output names in the description. Refer as appropriate to the decomposition diagrams.

5.2.2 File and Database Structures

Instructions: Using the Logical Data Model (LDM), create a physical data model that describes data storage and manipulation in the systems architectural setting. Describe file structures and their locations. Explain how data may be structured in the selected DBMS, if applicable. Refer to a separate DDD, as appropriate. For networks, detail the specific distribution of data. Note any changes to the LDM that occur because of software or hardware requirements.

5.2.2.1 Database Management System Files

Instructions: Provide the detailed design of the DBMS files. Generally, this information should be documented in a separate DDD that should be referenced within this section. A template for a DDD is available from the CMS Integrated IT Investment & System Life Cycle Framework website located at https://www.cms.gov/Research-Statistics-Data-and-Systems/CMS-Information-Technology/SystemLifecycleFramework/index.html .

5.2.2.2 Non-Database Management System Files

Instructions: Provide the detailed description of all non-DBMS files and include a narrative description of the usage of each file that identifies if the file is used for input, output, or both, and if the file is a temporary file. Also provide an indication of which modules read and write the file and include file structures (refer to the data dictionary). As appropriate, the file structure information should include the following:

a) Record structures, record keys or indexes, and data elements referenced within the records
b) Record length (fixed or maximum variable length) and blocking factors
c) Access method (e.g., index sequential, virtual sequential, random access, etc.)
d) Estimate of the file size or volume of data within the file, including overhead resulting from file access methods
e) Definition of the update frequency of the file (If the file is part of an online transaction-based system, provide the estimated number of transactions per unit of time, and the statistical mean, mode, and distribution of those transactions.)
f) Backup and recovery specifications

5.3 Data Conversion

Instructions: Insert any documents describing any necessary data conversions or provide a reference to where they are stored.

5.4 User Machine-Readable Interface

Instructions: Provide a description of each user class or role associated with the system. A user class is distinguished by the ways in which users interact with the proposed system or situation. Factors that distinguish a user class include common responsibilities, skill levels, work activities, and modes of interaction with the system. In this context, a user is anyone who interacts with the proposed system, including operational users, data entry personnel, system operators, operational support personnel, system maintainers, and trainers. For each user class, provide estimates of the total number of users anticipated, a maximum number of concurrent users, and the number of external users.

5.4.1 Inputs

Instructions: Provide a description of the input media used by the user/operator for providing information to the system. Show a mapping to the high-level data flows (e.g., data entry screens, optical character readers, bar scanners, etc.). If appropriate, the

input record types, file structures, and database structures provided in the section for Data Design, may be referenced. Include data element definitions, or refer to the data dictionary. Provide the layout of all input data screens or graphical user interfaces (GUIs) (e.g., windows). Define all data elements associated with each screen or GUI, or reference the data dictionary. Provide edit criteria for the data elements, including specific values, range of values, mandatory/optional, alphanumeric values, and length. Also address data entry controls to prevent edit bypassing. Discuss the miscellaneous messages associated with user/operator inputs, including (but not limited to) the following:

a) Copies of form(s) if the input data are keyed or scanned for data entry from printed forms
b) Description of any access restrictions or security considerations
c) Each transaction name, code, and definition, if the system is a transaction-based processing system
d) Incorporation of the Privacy Act statement into the screen flow, if the system is covered by the Privacy Act
e) Description of accessibility provisions to comply with Section 508 of the Rehabilitation Act

5.4.2 Outputs

Instructions: Describe the system output design relative to the user/operator. Show a mapping to the high-level data flows. System outputs include reports, data display screens and GUIs, query results, etc. The output files described in the section for Data Design may be referenced. The following should be provided, if appropriate:

a) Identification of codes and names for reports and data display screens
b) Description of report and screen contents (provide a graphical representation of each layout and define all data elements associated with the layout or reference the data dictionary)
c) Description of the purpose of the output, including identification of the primary users
d) Report distribution requirements, if any (include frequency for periodic reports)
e) Description of any access restrictions or security considerations

5.5 User Interface Design

Instructions: Insert any user interface design documents or provide a reference to where they are stored.

5.5.1 Section 508 Compliance

Instructions: Describe accessibility considerations in your user interface design and insert your section 508 compliance related documents or provide a reference to where they are stored.

6. Operational Scenarios

Instructions: Describe the general functionality of the system from the users' perspectives and provide an execution or operational flow of the system via operational scenarios that provide step-by-step descriptions of how the proposed system should operate and interact with its users and its external interfaces under a given set of circumstances. The scenarios tie together all parts of the system, the users, and other entities by describing how they interact, and may also be used to describe what the system should not do.

Operational scenarios should be described for all operational modes, transactions, and all classes of users identified for the proposed system. For each transaction, provide an estimate of the size (use maximum, if variable) and frequency (e.g., average number per session). Identify if there any transactional peak periods and include an estimate of frequency during those periods. Each scenario should include events, actions, stimuli, information, and interactions as appropriate to provide a comprehensive understanding of the operational aspects of the proposed system.

The scenarios can be presented in several different ways: 1) for each major processing function of the proposed system, or 2) thread-based, where each scenario follows one type of transaction type through the proposed system, or 3) following the information flow through the system for each user capability, following the control flows, or focusing on the objects and events in the system. The number of scenarios and level of detail specified will be proportional to the perceived risk and the criticality of the project.

7. Detailed Design

Instructions: Provide the information needed for a system development team to actually build and integrate the hardware components, code and integrate the software components, and interconnect the hardware and software segments into a functional product. Additionally, address the detailed procedures for combining separate COTS packages into a single system.

7.1 Hardware Detailed Design

Instructions: Provide enough detailed information about each of the individual hardware components to correctly build and/or procure all the hardware for the system (or integrate COTS items). If there are many components or if the component documentation is extensive, place it in an appendix. Add additional diagrams and information, if necessary, to describe each component and its functions adequately. Industry-standard component specification practices should be followed. For COTS components, identify specific vendor and appropriate item names and model numbers. Include the following information in the detailed component designs, as applicable:

a) Power input requirements for each component

b) Signal impedances and logic states
c) Connector specifications (serial/parallel, 11-pin, male/female, etc.)
d) Memory and/or storage space specifications
e) Processor requirements (speed and functionality)
f) Graphical representation depicting the number of hardware items (e.g., servers, I/O devices, monitors, printers, etc.), and the relative positioning of the components to each other
g) Cable type(s) and length(s)
h) User interfaces (buttons, toggle switches, etc.)
i) Hard drive/floppy drive/CD-ROM specifications
j) Monitor resolution

7.2 Software Detailed Design

Provide a detailed description for each system software Service that addresses the following software Service attributes. Much of the information that appears in this section should be contained in the headers/prologues and comment sections of the source code for each component, subsystem, module, and subroutine. If so, this section may largely consist of references to or excerpts of annotated diagrams and source code. Any referenced diagrams or source code excerpts should be provided at any design reviews.

Service Identifier – the unique identifier and/or name of the software Service.

Classification – the kind of Service (e.g., application, data service, etc.)

Definition – the specific purpose and semantic meaning of the Service.

Requirements – the specific functional or nonfunctional requirements that the Service satisfies.

Internal Data Structures – the internal data structures for the Service.

Constraints – any relevant, assumptions, limitations, or constraints for the Service. This should include constraints on timing, storage, or Service state, and might include rules for interacting with the Service (encompassing pre-conditions, post-conditions, invariants, other constraints on input or output values and local or global values, data formats and data access, synchronization, exceptions, etc.)

Composition – a description of the use and meaning of the subServices that are a part of the Service.

Users/Interactions – a description of the Service's collaborations with other Services. What other Services is this entity used by? What other Services do this entity use (including any side-effects this Service might have on other parts of the system)? This includes the method of interaction, as well as the interaction itself. Object-oriented designs should include a description of any known or anticipated sub-classes, super-classes, and meta-classes.

Processing – a description of precisely how the Service goes about performing the duties necessary to fulfill its responsibilities. This should encompass a description of

any algorithms used; changes or state; relevant time or space complexity; concurrency; methods of creation, initialization, and cleanup; and handling of exceptional conditions.

Interfaces/Exports – the set of services (resources, data types, constants, subroutines, and exceptions) that are provided by the Service. The precise definition or declaration of each such element should be present, along with comments or annotations describing the meanings of values, parameters, etc. For each service element described, include or provide a reference in its discussion to a description of its important software Service attributes (Component Identifier, Classification, Language, SLOC Estimate, Definition, Responsibilities, Requirements, Internal Data Structures, Constraints, Composition, Uses/Interactions, Resources, Processing, and Interfaces/Exports).

Reporting Design and Integration – if built in, provide details on data traffic and volumes.

7.3 Security Detailed Design

Instructions: Provide a graphical representation with detailed information for each of the individual security hardware components. Specify the design for the below items as required.

a) *Authentication*
b) *Authorization*
c) *Logging and Auditing*
d) *Encryption*
e) *Network ports usage*
f) *Intrusion Detection and Prevention (especially if hosted in a non-CMS data center*

The design should be based on the designated system security level and provide adequate protection against threats and vulnerabilities.

7.4 Performance Detailed Design

Instructions: Provide a graphical representation with detailed information for each of the individual performance and reliability hardware components to include the below items:

a) *Capacity and volume requirements/estimates*
b) *Performance expectations*
c) *Availability requirements*
d) *Performance design to meet capacity requirements*
e) *Reliability design to meet availability requirements*
f) *Backup, recovery, and archive design*

Identify single points of failure and, if relevant, describe high availability design (e.g., clustering).

7.5 Internal Communications Detailed Design

Instructions: If the system includes more than one component, there may be a requirement for internal communications to exchange information, provide commands, or support input/output functions. Provide enough detailed information about the communication design to correctly build and/or procure the communications components for the system. Include the following information in the detailed component designs, as appropriate:

a) *Number of servers and clients to be included on each area network*
b) *Specifications for bus timing requirements and bus control*
c) *Format(s) for data being exchanged between components*
d) *Graphical representation of the connectivity between components, showing the direction of data flow (if applicable), and approximate distances between components (information should provide enough detail to support the procurement of hardware to complete the installation at a given location)*
e) *LAN topology*

8. System Integrity Controls

Instructions: Provide design specifications for the following minimum levels of control and any additional controls as appropriate or necessary:

a) *Internal security to restrict access of critical data items to only those access types required by users/operators*
b) *Audit procedures to meet control, reporting, and retention period requirements for operational and management reports*
c) *Application audit trails to dynamically audit retrieval access to designated critical data*
d) *Standard tables to be used or requested for validating data fields*
e) *Verification processes for additions, deletions, or updates of critical data*
f) *Ability to identify all audit information by user identification, network terminal identification, date, time, and data accessed or changed.*

9. External Interfaces

Instructions: Describe any interfaces that exist with external systems that are not within the scope of the system being designed, regardless whether the other systems are managed by CMS or another entity. Describe the electronic interface(s) between the system being designed and each of the other systems and/or subsystem(s), emphasizing the point of view of the system being designed. If there is more than one or two external systems, or if the interfaces are not simplistic, one or more separate Interface Control Documents (ICDs) should be prepared and referenced here. If applicable, identify how many ICDs exist and what they are. A template for an ICD is available from the CMS Integrated IT Investment & System Life Cycle Framework

website located at https://www.cms.gov/Research-Statistics-Data-and-Systems/CMS-Information-Technology/SystemLifecycleFramework/index.html

9.1 Interface Architecture

Instructions: Describe the interface(s) between the system being developed and other systems (e.g., batch transfers, queries, etc.), indicating the location of the interfacing system. Include the interface architecture(s) being implemented (e.g., wide area networks, gateways, etc.) and the interfacing mechanisms (e.g., MQ, Gentran, etc.) If remote connectivity is required, identify the method of access. Provide a diagram depicting the communications path(s) between this system and each of the other systems, which should map to the context diagram(s) provided in the Section for System Overview. The graphical representation should depict the connectivity between systems, showing the direction of data flow. Use subsections or a separate ICD(s) to address each interface independently.

9.2 Interface Detailed Design

Instructions: For each external system with which the system being designed interfaces, describe the information exchange and rules governing the interface. Provide enough detailed information about the interface to correctly format, transmit, and/or receive data across the interface. Generally, this information should be documented in a separate ICD(s) that should be referenced within this section. A template for an ICD is available from the CMS Integrated IT Investment & System Life Cycle Framework website located at https://www.cms.gov/Research-Statistics-Data-and-Systems/CMS-Information-Technology/SystemLifecycleFramework/index.html

Appendix A: Record of Changes

Instructions: Provide information on how the development and distribution of the System Design Document will be controlled and tracked. Use the table below to provide the version number, the date of the version, the author/owner of the version, and a brief description of the reason for creating the revised version.

Table 1: Record of Changes

Version Number	Date	Author/Owner	Description of Change

Appendix B: Acronyms

Instructions: Provide a list of acronyms and associated literal translations used within the document. List the acronyms in alphabetical order using a tabular format as depicted below.

Table 2: Acronyms

Acronym	Literal Translation
CMS	Centers for Medicare & Medicaid Services
COTS	Commercial Off-the-Shelf
DBMS	Database Management System
DDD	Database Design Document
EA	Enterprise Architecture
GUI	Graphical User Interface
ICD	Interface Control Document
LAN	Local Area Network
LDM	Logical Data Model
SDD	System Design Document
SLOC	Source Lines of Code
WAN	Wide Area Network

Appendix C: Glossary

Instructions: Provide clear and concise definitions for terms used in this document that may be unfamiliar to readers of the document. Terms are to be listed in alphabetical order.

Table 3: Glossary

Term	Definition

Appendix D: Referenced Documents

Instructions: Summarize the relationship of this document to other relevant documents. Provide identifying information for all documents used to arrive at and/or referenced within this document (e.g., related and/or companion documents, prerequisite documents, relevant technical documentation, etc.).

Table 4: Referenced Documents

Document Name	Document Location and/or URL	Issuance Date

Appendix E: Approvals

The undersigned acknowledge that they have reviewed the System Design Document and agree with the information presented within this document. Changes to this System Design Document will be coordinated with, and approved by, the undersigned, or their designated representatives.

Instructions: List the individuals whose signatures are desired. Examples of such individuals are Business Owner, Project Manager (if identified), and any appropriate stakeholders. Add additional lines for signature as necessary.

Signature: _____ Date: _____

Print Name: _____

Title: _____

Role: _____

Signature: _____ Date: _____

Print Name: _____

Title: _____

Role: _____

Signature: _____ Date: _____

Print Name: _____

Title: _____

Role: _____

Appendix F: Notes to the Author / Template Instructions

This document is a template for creating a System Design Document for a given investment or project. The final document should be delivered in an electronically searchable format. The System Design Document should stand on its own with all elements explained and acronyms spelled out for reader/reviewers, including reviewers outside CMS who may not be familiar with CMS projects and investments.

This template includes instructions, boilerplate text, and fields. The developer should note that:

- *Each section provides instructions or describes the intent, assumptions, and context for content included in that section. Instructional text appears in blue italicized font throughout this template.*
- *Instructional text in each section should be replaced with information specific to the particular investment.*
- *Some text and tables are provided as boilerplate examples of wording and formats that may be used or modified as appropriate.*

When using this template, follow these steps:

1. *Table captions and descriptions are to be placed centered, above the table.*
2. *Modify any boilerplate text, as appropriate, to your specific investment.*
3. *Do not delete any headings. If the heading is not applicable to the investment, enter "Not Applicable" under the heading.*
4. *All documents must be compliant with Section 508 requirements.*
5. *Figure captions and descriptions are to be placed centered, below the figure. All figures must have an associated tag providing appropriate alternative text for Section 508 compliance.*
6. *Delete this "Notes to the Author / Template Instructions" page and all instructions to the author before finalizing the initial draft of the document.*

Appendix G: XLC Template Revision History

The following table records information regarding changes made to the XLC template over time. This table is for use by the XLC Steering Committee only. To provide information about the controlling and tracking of this artifact, please refer to the Record of Changes section of this document.

Table 5: XLC Template Revision History

Version Number	Date	Author/Owner	Description of Change
1.0	11/9/2007	Julie Shadoan, OIS/EASG/DITPPA	Baseline Version
1.1	8/12/2008	Julie Shadoan, OIS/EASG/DITPPA	Approved by ESD Deliverables Workgroup
1.2	9/23/2009	Kristine Maenner, OIS/EASG/DITPPA	Added sections to call out Security and Performance Hardware and Software
1.3	2/11/2010	Kristine Maenner, OIS/EASG/DITPPA	In the Detailed Design Sections, replaced reference to “component” to “service.”
2.0	11/4/2010	Celia Shaunessy, OIS/EASG/DITG	Added Records Management information.
3.0	08/14/2014	Celia Shaunessy, XLC Steering Committee	Changes made per CR 14-012.
3.1	10/21/2014	Enterprise Test Center	Added Section 3.4 – Performance Engineering- per CR 14-008.
3.2	02/02/2015	Surya Potu, CMS/OEI/DPPIG	Updated CMS logo.

Appendix H: Additional Appendices

Instructions: Utilize additional appendices to facilitate ease of use and maintenance of the document. Suggested appendices include (but are not limited to):

a) *Software Architecture Diagrams – provide the functional hierarchy diagrams, structured organization diagrams, or object-oriented diagrams that show the various segmentation levels of the software architecture down to the lowest level.*

b) *Data Dictionary – provide definitions of all processes, data flows, data elements, and data stores.*

c) *Requirements Traceability Matrix – demonstrate backward traceability of the system and software architectural designs to the functional and nonfunctional requirements documented in the Requirements Document.*

d) *CMS Section 508 Product Assessment – demonstrate compliance or non-compliance with accessibility standards provided in Section 508 of the Rehabilitation Act of 1973, as amended effective June 20, 2001 The template for this appendix is available at https://www.cms.gov/Research-Statistics-Data-and-Systems/CMS-Information-Technology/XLC/Downloads/Sect508ProdAssessment.docx*