Department of Health and Human Services

Centers for Medicare & Medicaid Services
Integrated IT Investment & System Life Cycle Framework

CMS Requirements Writer's Guide

Version 4.11

August 31, 2009

(Approved for use by the ILC Steering Committee)

Record of Changes

Number	Date	Reference	A = Add, M = Modify, D= Delete	Description of Change	CR #
1	October 6, 2008	All	A	Initial Release, v4.0	NA
2	April 20, 2009	Pages 13-14	M	Updated Sections 3.1.5, 3.1.6 and 3.3	NA
3	August 31, 2009	Page 21	M	Updated example scenario, inserted “user requirements” where applicable, and corrected metamorphosis of requirements document	57

CR: Change Request

Table of Contents

List of Figures

List of Tables

1.0 INTRODUCTION

1.1 Background & Vision

The Centers for Medicare & Medicaid Services (CMS) desire to apply a more uniform methodology to the documentation of proposed information systems that service the Agency's Information Technology (IT) needs. This initiative spans the CMS enterprise to include all development activities for new or redesigned systems. The Agency continues to evolve, elaborate and enhance its approach and methodology to the capture and documentation of requirements that meet the criteria of being complete, clear, consistent, verifiable, feasible, modular, singular and design independent so that the enterprise technical vision and architecture return the highest value to it beneficiaries.

Whether building a new system, making changes to existing software or using commercial off-the-shelf software, exploring, capturing and communicating requirements is necessary. Requirements are commonly used to communicate ideas. They are usually derived from a variety of informational sources and can represent many things to many people. Because requirements can be interpreted in many ways, defining them is a form of art that is commonly used to express the needs and wants of a system in its simplest form that removes all ambiguity.

In the world of systems development, requirements represent the things needed to know and the things needed to do that are critical for project success. The right product cannot be built unless it is known precisely what the product should do and how the product's success is to be measured. Projects succeed when there are clearly defined goals and attainable results, which focus on satisfying the need and delivering results users want. This is only possible when those needs are clearly defined and the problem is fully documented in writing.

The person responsible for writing requirements must understand the business goals and have knowledge of the intended product, what it has to do, what qualities it must have, what constraints it must conform to and what interfaces it must have to the outside world. As development continues, knowledge of the product increases and the analytical activity should continue to grow until all the requirements are known and analyses at all phases are complete. In this way, the entire analysis process deals with identifying the 'problem'.

1.2 Purpose

The purpose of the *CMS Requirements Writer's Guide (RWG)* is to provide guidance to managers, analysts, users, developers and others who are responsible for scoping, writing or reviewing requirements. It seeks to describe the level of detail that should be included in each written requirement and what stakeholders must look for when conducting formal reviews of internally or externally developed requirements. It also intends to standardize the nomenclature associated with writing requirements. The document employs an object oriented methodology for organizing requirements.

The RWG will assist in effectively managing requirements and providing guidelines for measuring and accepting requirements. Lastly, the document provides a clearly defined structure for written requirements and a standardized set of criteria to assess each requirement.

This guide provides guidelines to assist in the development of requirements. Although it is encouraged that these guidelines be rigorously followed, the unique characteristics of each IT project may warrant slight modifications to these guidelines.

1.3 Scope

This guide addresses how to write and organize business, user, functional and nonfunctional requirements for IT projects. It is the Agency's goal that all IT investments identified as new projects, system redesigns, platform ports, or incorporating major new business functions use the guidance provided in the RWG. It is not the intent that systems currently in production and performing routine maintenance must convert their requirements baseline to this format.

1.4 Intended Audience

The RWG is appropriate for all stakeholders of the Integrated IT Life Cycle Framework but especially for: business owners, project managers, business analysts, and subject matter experts charged with developing business, user, functional and nonfunctional requirements.

1.5 Reference Documents

CMS follows the guidance specified in Section 6.26.3 of IEEE/EIA 12207.1 1997 Guide for Information Technology with respect to requirements. The IEEE standard recommends tailoring its methodology to fit the needs of the organization using it. Captured within the RWG are the results of this tailoring. CMS also adopts best business practices as specified in Chapter 5 of the Business Analysis Body of Knowledge v1.6. The following documents were used in developing this guide:

- Business Analysis Body of Knowledge, Version 1.6
- IEEE/EIA Guide for Information Technology, Std 12207.1
- IEEE Developing System Requirements Specification, Std 1233
- IEEE Recommended Practice for Software Requirements Specifications, Std 830
- The Art and Discipline of Writing Requirements, Revision 2

1.6 Document Organization

It is strongly recommended that all readers familiarize themselves with the terms listed in the glossary. Novice guide users may find it helpful to refer to the glossary as the terms are used throughout the document. Experienced requirements documenters should review the glossary to determine if their understanding of a term coincides with how the guide intends to use it.

Readers unfamiliar with their role in the requirements writing process should first examine Section 2 where the tasks and responsibilities for the various people involved in the requirements writing process are identified. Following this, the reader may refer to the appropriate sections in Section 3 that guides them through the process of fulfilling each of the tasks they are responsible for completing. By contrast, if readers are familiar with what is expected of them, they should be able to refer to the Table of Contents and the Index to quickly locate specific, relevant sections.

It is recommended that all readers familiarize themselves with Section 4 where stylistic and syntactical writing considerations are discussed. Stylistic considerations are of importance to all

writers. Syntactical rules are especially important to business analysts because they standardize communication between the business owner and the designer, thus providing more clarity.

This document is organized as follows:

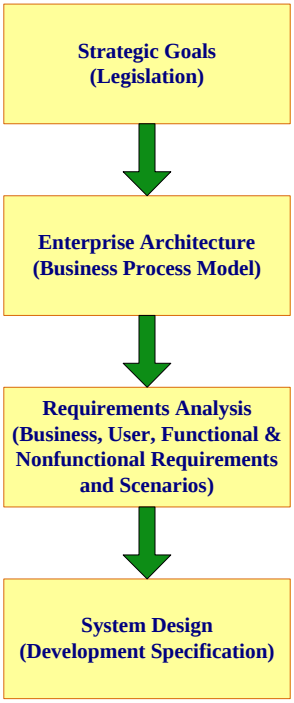
Table 1 - Document Organization

Section	Overview
Section 2: Guide Users	Identifies each type of user and what is expected of them throughout the requirements documentation process.
Section 3: Project Stakeholder Tasks	Describes the specific tasks and deliverables to complete as determined by the reader's role in the project.
Section 4: Style Guidelines	A discussion of a 'middle level' investigation of requirements including the proper layout, form and phrasing of requirements.
Section 5: Acronym List	A list of acronyms used in the document.
Appendix A: Requirement Attributes	How to ensure completeness of requirements and standardize how requirements are documented and tracked.
Appendix B: Advanced Stylistic Approaches	A discussion on methods that may be used to reduce ambiguity in written requirements documents.
Index	Index of common terms used in the RWG.

1.7 Process Hierarchy

This guide is based on a process of building a system that starts at a very broad level (Strategic Goals) and generates more detail about the system in successive levels. The need for a new system usually begins with either legislation from Congress or a strategic initiative from CMS Management. The business owner then translates these goals into business requirements. The business owner uses the business requirements to create more detailed user requirements, scenarios and functional and nonfunctional requirements with the help of business analysts and subject matter experts. The Requirements Document is then turned over to the systems developer where the process of solving the problem begins.

Figure 1 - Process Hierarchy



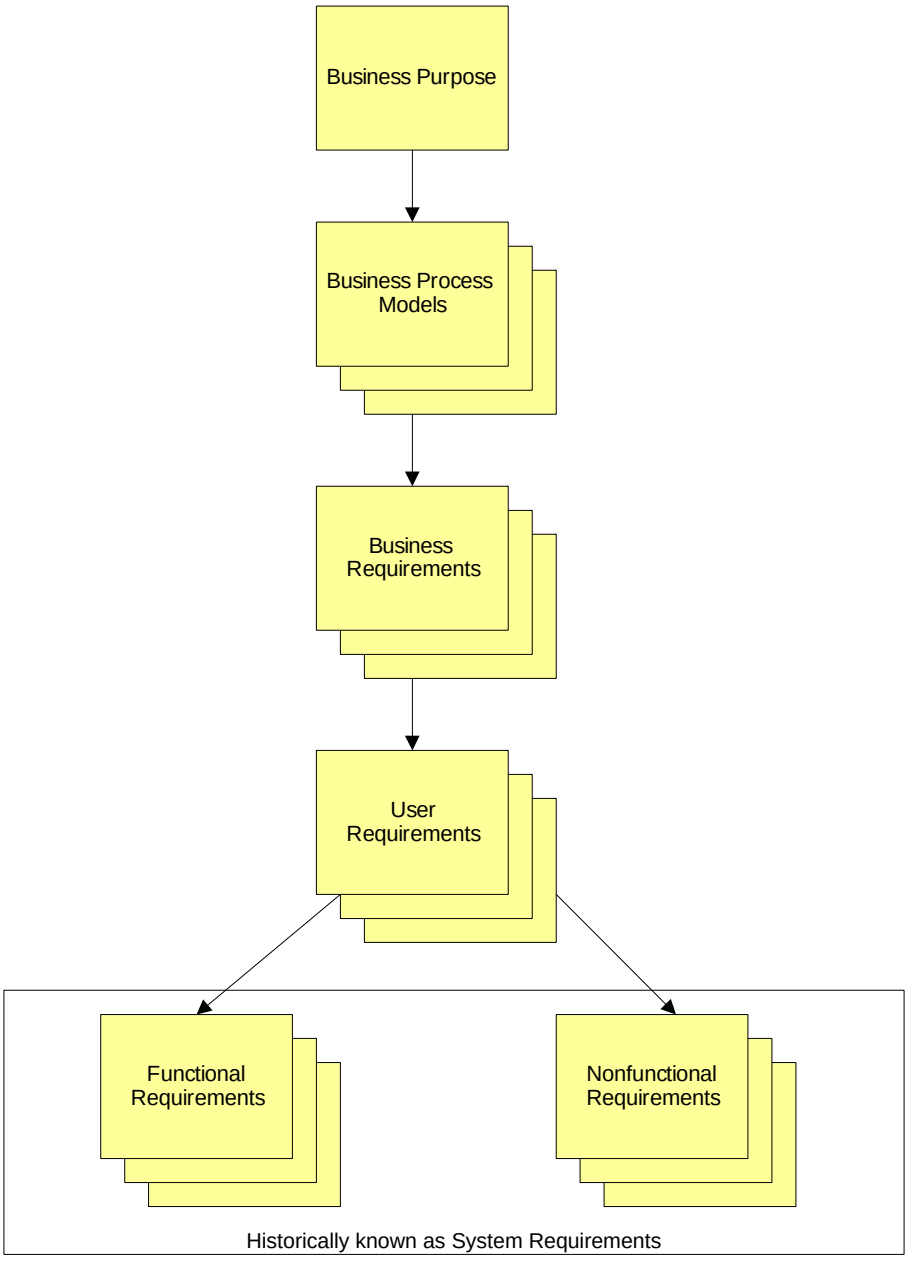
1.8 Requirements Hierarchy

The requirements hierarchy, as diagrammed below, represents the customary progressive elaboration that takes place when eliciting requirements in the early phases of a project. Ideally, business owners first identify all the business requirements necessary for supporting their business purpose. These requirements should be very broad and inclusive of all the work to be performed by the business unit so that all potential opportunities for leveraging IT investments are recognized. Along with documenting the business requirements, business rules are captured.

At this point, the business unit has the information needed to begin defining how an IT investment can be used to fulfill the business requirements identified. The scope of the IT investment is characterized at this level by the user requirements. Following this, functional and nonfunctional requirements are used to define what the proposed system will actually do.

The CMS Integrated IT Investment & System Life Cycle Framework shows the creation of the requirements document spanning the Concept, Planning, and Requirements Analysis Phases .At first the requirements document may elaborate down to just the business requirements level. As analysis continues, user and some functional/functional requirements are captured. The requirements document is considered complete when all requirements down to the functional/nonfunctional requirements have been documented therein. The requirements document requires sign off by the business owner.

Figure 2 - Requirements Hierarchy



Historically, the term "system requirements" was used to categorize all low level requirements intended to describe an IT deliverable. In order to ensure that requirements are more completely defined, the CMS Division of Requirements and Validation (DRAV) has discontinued use of that term in favor of describing requirements as either functional or nonfunctional (see Glossary). In addition, DRAV inserted a layer of requirements (User Requirements) which further assists in contextualizing functional and nonfunctional requirements as well as establishing a top level system elaboration of the business requirements.

1.9 Glossary

Table 2 - Glossary

Term	Process Hierarchy Level (see Section 1.7)	Definition
Alternate Scenario	Requirements Analysis	A scenario identifying the steps that occur when the outcome of a scenario differs from the expected outcome.
Business Requirements (BR)	Enterprise Architecture & Requirements Analysis	Business requirements are statements of the functions or program needs that must be met in order to accomplish the business objectives of the project. These business objectives address legislative mandates or strategic business goals (such as improved customer service, business efficiencies, business process reengineering, etc.).
Business Process Model (BPM)	Enterprise Architecture	A graphical depiction of either the existing (As Is) business work flow or the desired (To Be) business work flow.
Business Rule (RU)	Requirements Analysis & Requirements Analysis	The conditions under which a business requirement must be fulfilled.
Constraint	Requirements Analysis	A specific type of requirement that limits design, development, or deployment options. Constraints mandate the way the system must be produced and may exist at both the business requirements level and the functional/nonfunctional requirements level.
Domain	Requirements Analysis	A user or system that interacts with the system being designed (actor).
Event	Requirements Analysis	Historically, a synonym for user requirement.
Functional Requirement (FR)	Requirements Analysis	A statement of action that describes the behavior and information that the solution will manage. They describe capabilities the system will be able to perform in terms of behaviors or operations – a specific system action or response.

Term	Process Hierarchy Level (see Section 1.7)	Definition
Nonfunctional Requirement (NR)	Requirements Analysis	A statement that describes conditions that do not directly relate to the behavior or functionality of the solution, but rather describe environmental conditions under which the solution must remain effective or qualities that the systems must have.. See Section 3.3.4.1 for a list of various types.
Pass/Fail Statement	Requirements Analysis	A testable statement that describes how to know if a requirement is met.
Precondition	Requirements Analysis	The state the system must be in before a scenario can start.
Requirement	All levels	A measurable statement of intent about something that the system must do, a property the system must have or a constraint on the system.
Scenario	Requirements Analysis	A sequence of steps taken to complete a user requirement, similar to a use case.
Subject Matter Expert (SME)	All levels	A subject matter expert is an employee of CMS or a contractor with specific knowledge regarding a domain that interacts with the system.
Stakeholder	Enterprise Architecture	A person, system or entity that interacts with, is responsible for or has some interest/influence in the system being developed.
System Requirement	Requirements Analysis	A historical umbrella term, which included functional and nonfunctional requirements, and defined what a system must do in order to satisfy the user requirements. As such, system requirements are lower level requirements that address the specific characteristics that the system must address in order to be acceptable as an end product.
Traceability	Requirements Analysis	The ability to trace relationships between two or more requirements.
Trigger	Requirements Analysis	An action or activity that occurs that causes the scenario to begin.
Use Case	Requirements Analysis	A description of a system's behavior as it responds to a request that originates from outside of that system. The use case is made up of a set of possible sequences of interactions between systems and users in a particular environment and related to a particular goal. The use case should contain all system activities that have significance to the users. Use cases typically avoid technical jargon, preferring instead the language of the subject matter expert.
User Requirement (UR)	Requirements Analysis	A requirement that describes how a user interacts with the system in order to fulfill a Business Requirement. The UR could also be internally initiated by the system.

2.0 USERS OF THIS GUIDE

There are primarily four types of users this guide is intended to help:

- Business Owners
- Project Managers
- Business Analysts
- Subject Matter Experts

The guide will help each type of user identify what is expected of them throughout the requirements documentation process.

2.1 Business Owners

The business owner(s) are the entity or entities responsible for defining, promoting, endorsing and upholding the business needs and user requirements for the system and for performing user acceptance testing of the final product(s) based on those business needs and requirements. The business owner(s) may represent the interests of external partners such as Medicare Contractors, Fiscal Intermediaries or other Federal agencies and/or the interests of internal CMS Centers, Offices or Consortia. The business owner(s) define and verify system functionality, access rights, Business Rules and the privacy classification, timeliness, completeness and accuracy of data. The business owner may also be the project manager.

Below are the tasks for which the business owner is responsible in the requirements writing process. If a Business Process Model (BPM) was created for the project, the business owner should be thoroughly familiar with it. This will be of great assistance and in some cases will provide the information necessary to perform the following tasks.

- Writing Business Requirements
- Writing Business Rules
- Writing a Business Purpose
- Writing a Functional Purpose
- Determining the Measures of Success
- Assessing the Project Priorities
- Identifying the Stakeholders
- Determining the Risks
- Determining the Assumptions and Constraints

Sections 3.1 – 3.3 of this document detail these tasks.

2.2 Project Managers

The project manager is the individual at CMS who has the day-to-day responsibility for the success and management of the IT project. This individual may be at any level in the CMS organizational structure. The project manager is the primary point of contact for the project with relation to the CMS Integrated IT Investment & System Life Cycle Framework. The project manager communicates regularly with management on the needs and progress of the project and is responsible for obtaining and managing project resources. In those cases where contract support is utilized, the project manager may also serve as the Government Task Leader (GTL) and/or the Project Officer, or otherwise provide support to the assigned GTL and Project Officer for the contract. The project manager may also operate as a business analyst.

2.2.1 Scoping Requirements

It is essential that the project manager have the business strategy documentation produced by the business owner in order to begin the requirements documentation process. Typically, the first step in the requirements documentation process is to review the BPM and supporting narrative to capture any Business Requirements documented or implied there. It is the responsibility of the project manager to ensure the Requirements Document is complete.

If the project manager is also operating as part of the business analysis team, the project manager should refer to Sections 3.1 - 3.3 in order to start the requirements writing process.

2.2.2 Managing Requirements

The project manager is responsible for managing the requirements during requirements development as well as after the team has finished its work. This includes ensuring requirement traceability, reviewing requirements, baselining requirements (both formal and informal), tracking change requests (when appropriate) and enforcing writing standards. It is not the intent of this document to provide guidance on how to manage or maintain requirements.

It is also the project manager's responsibility to manage updates to the Requirements Document. A template for this document is available on the CMS website. The various parts of the Requirements Document are discussed throughout this guide.

2.3 Business Analysts

The business analyst is an individual tasked with gathering and writing requirements for the IT project. The person may also be the project manager, an employee of the CMS organization or a contractor. The business analyst communicates regularly with the project team and the subject matter experts while gathering and writing the requirements.

After performing the requirements gathering process, the business analyst is responsible for putting the requirements into a formal document that may be handed over to system designers to continue the project. The steps an analyst must take to accomplish this are as follows:

- Documenting Business Requirements & Business Rules
- Documenting User Requirements
- Creating Scenarios

- Writing Functional Requirements
- Writing Nonfunctional Requirements
- Capturing the Requirement Attributes
- Importing Requirements into the CMS Requirements Repository

These tasks are described in Section 3.3.

2.4 Subject Matter Experts

A Subject Matter Expert (SME) should generally be a stakeholder in the project, though not all stakeholders are SMEs. Their expertise may come from either a business or a technical perspective.

While a SME may not be responsible for writing requirements or the supporting documentation, they should be familiar with what the project team will expect from them during this process so that the team's deliverables will accurately reflect the SME's needs and knowledge.

SMEs should be involved throughout the entire process of requirements documentation and should specifically be familiar with:

- Section 3.1: Project Strategy tasks
- Section 3.2: Writing Business Requirements tasks
- Section 3.3.: Writing User, Functional and Nonfunctional Requirements tasks

3.0 REQUIREMENTS DEVELOPMENT TASKS

This section describes specific tasks and deliverables to complete as determined by the reader's role in the project. See Section 2 if you are unsure of which tasks require your participation.

3.1 Project Strategy

All the stakeholders that are identified in the BPM or during the project kickoff should participate in performing the business strategy tasks. Project managers cannot begin their work until these tasks are complete. The information collected in these tasks will be part of the Requirements Document.

3.1.1 Writing a Business Purpose

The business purpose describes *why* CMS would fund the project. The business purpose is always related to the mission of the organization and/or financial concerns. CMS projects are often the result of legislative initiatives, which are therefore related to the mission of the organization. Upgrades and enhancements to systems often involve financial concerns, allowing the agency to take advantage of current (more efficient) technology. The business purpose may be used to capture the business drivers if this information is not already covered in the introduction/purpose of the requirements document.

An example of a good business purpose would be:

> The business purpose is to comply with the provisions set by Congress in The Medicare Prescription Drug, Improvement and Modernization Act of 2003 (MMA) by reimbursing care providers for services rendered to beneficiaries. In accomplishing this, the process of making payments will be more efficient in terms of both time and money.

It is possible that there are multiple business purposes for funding the project, and each should be written in a complete sentence. When stating more than one purpose, ensure they don't compete. For example, complying with a law and updating technology to reduce maintenance costs are valid business purposes that do not conflict. If there are more than two business purposes, the project team may want to consider breaking the project up into smaller components.

3.1.2 Writing a Functional Purpose

The functional purpose describes *what* the project shall do for the business in a single, complete sentence. It is not unusual for stakeholders to have competing notions regarding what the functional purpose should be. While a compound sentence is acceptable, the purpose should still be limited to one sentence.

The purpose should be a statement containing strong transitive verbs like: build, compare, explore, investigate, perform, recommend or replace. This is the statement that gives the entire project its direction. When formulating the statement, consider the seriousness of the problem and why it needs to be solved.

An example of a functional purpose might be:

> The functional purpose is to replace the existing functionality of the APPS system by leveraging the technical architecture already in place at CMS and to add functionality that will reduce existing manual processes.

If a functional purpose cannot be reduced to a single sentence, the project team may want to consider if the project is too big and should be broken up into separate projects.

3.1.3 Determining the Measures of Success

The measures of success list metrics on how the project shall be measured against the functional and business purposes. The measures of success must always be concrete and measurable. This can be especially challenging if the processes used in an existing system have never been measured. The measures must also tie back to the business and functional purposes. Some types of measures include:

- Customer satisfaction: does it meet an expectation or improve the customer's perception of the business?
- Work cycle times: how efficiently should the system work or affect other work?
- Product and service quality: how does the system improve the business process?
- Cost performance: does the system reduce business expenses?
- Employee satisfaction: does it meet an expectation or improve the employee's ability to perform the business functions?

Some examples of these include:

- The system processes payments with 100% accuracy. (service quality)
- The system processes a transaction within .5 seconds. (work cycle time)
- The system reduces maintenance costs by 25%. (cost performance)

3.1.4 Identifying the Stakeholders

Stakeholders of the project are initially identified by the BPMs and are expanded here. They fall into one or more of the following four categories:

- Those who will pay for the project: these are the people who have the final say over the completeness of the requirements.
- Those who will use or influence the end product (including other systems): these are the people and systems where much of the desired functionality will be elicited. Stakeholders who also exert influence on the system design include those who set technical architecture policy or who deliver data exchange files (defined by Interface Control Documents).
- Those who will be positively affected by the project's implementation: these represent the most cooperative stakeholders and it would be natural to expect some scope creep from them.
- Those who will perceive to have limited benefit by the project's implementation: these represent the most resistant stakeholders and satisfying their requirements can be challenging.

It is important to list and discuss all categories to accurately determine each group's concerns and needs. Identify those who will use the system by describing how they will use it and from

where they will use it. It may also help to determine approximately how many users are in each grouping.

An example of the stakeholders for a project might be:

Payers

- CBC

Users (people and systems that interact with or influence the end product)

- CMS
  - CBC Management: responsible for approving payments
  - CBC Staff: responsible for processing payments
  - OIS: responsible for setting technical architecture policy
  - MARx system: provides beneficiary level payment information
  - FACS: notifies the Treasury of payments to plans
- Department of the Treasury: responsible for transferring funds to plans
- Plan Sponsors: receives payments and payment information

Positively Affected

- Plan Sponsors will have payments processed more accurately and timely
- CBC Staff/Management will have more robust tools to perform the monthly payment process

Limited Benefit

- Plans owing CMS money will be more carefully tracked

It is important to consider how the entire user and support staff will be affected by the new system. Many times, while a new system will ultimately make a user more productive, it often places additional burdens or demands on support groups. Understanding how they are adversely affected can uncover new assumptions that may need to be addressed.

3.1.5 Determining the Project Assumptions & Constraints

It is valuable to list the constraints and assumptions that the project team and developers identify about the environment in which they will base their project planning decisions. False or unstated assumptions and constraints can result in risks being realized and potential failure of the project.

An assumption is a statement believed to be true for the purposes of planning but is presently unverifiable. Over time, assumptions can evolve as they become validated (at which time they may become constraints or are incorporated into the business or technical requirements). It may be helpful to assess each assumption and state the likelihood that it is correct. Some assumptions might include:

- The new system will utilize the same ICDs as the existing system.
- The new system will not be treated as a system of record.

A constraint is an internal or external limitation on the project that affects its performance. It can be related to the business (e.g., budget or human resources) or it may be technical in nature (such as architecture standards). Typical types of constraints are both business and technical including:

- Compliance: what legislative initiates must be met?
- Solution: distinct from technological restraints, it defines the boundaries within which the problem may be solved and are absolutely non-negotiable.

- Schedule: how much time is allotted to build the system? Is there a window of opportunity that must be taken advantage of or deadlines that have to be met?
- Resources: how do the dollars, skills or people available to the project affect requirements capture and prioritization?
- Current Implementation: if applicable, a description of the components (automated, mechanical, organizational and otherwise) of the current system.
- Architectural: what technical decisions regarding the implementation are already in place?

For example, constraints might be:

- Compliant with the MMA
- The system is ready for processing payments by October 1, 2012.
- The system runs under the CMS 3 zone architecture.

Assumptions and constraints can also be interrelated such as when the project is constrained by certain facts and it assumed that those facts will be acceptable. In such cases, it may be helpful to state it as both an assumption and a constraint. For example:

- Assumption: the HITSP specification will be adopted by providers.
- Constraint: the HITPS specification must be employed.

Constraints and assumptions should be revisited periodically during requirements development to determine if any updates are necessary.

3.1.6 Determining the Project Risks

Constraints and assumptions will usually have associated risks. It is helpful to ask such questions as what would be the result should an assumption be proven false or a constraint prevents the project from meeting a measure of success. Identify the project risks that the organization faces in such instances. Assess the level of impact and likelihood of risk (high, medium, low), the consequences should the risk become a reality, as well as some methods for reducing the risk. Categorize the risk if appropriate (technical, operational, user acceptance, etc.)

A project might include the following risks:

- Schedule (High Impact/Medium Likelihood): the subject matter experts are already overloaded with work and have little extra time for requirements elicitation. Consequence: the project will not meet the scheduled deadline. Mitigation: prepare requirements for SME review and use less interview time.
- Training (Low Impact/Low Likelihood): the new system may not have a similar look and feel to the existing system. Consequence: the new system will be adopted at a slower rate than desired. Mitigation: Do screen mock ups for the SME's to review.

Risks identified in the project's Business Case should be evaluated for inclusion in the requirements document.

3.1.7 Assessing the Project Priorities

Successful completion of a project depends on several factors. Among these are the resources available to do the work, the proper scope of the work, the work being completed on schedule to meet business needs and a quality solution that is free from defects. However, there is always an inherent conflict between scope, resources, schedule and allowable defects.

The project priorities help the team determine what is most important among the competing priorities, should a choice need to be made. The team is only allowed to select one of these as being of highest importance to the project. Ideally, only one of the four should be designated as second highest. Use a chart similar to the following to discuss the four priorities with the team. After making the selections on the chart, document why the team made its selections. For example:

Table 3 - Sample Project Priorities

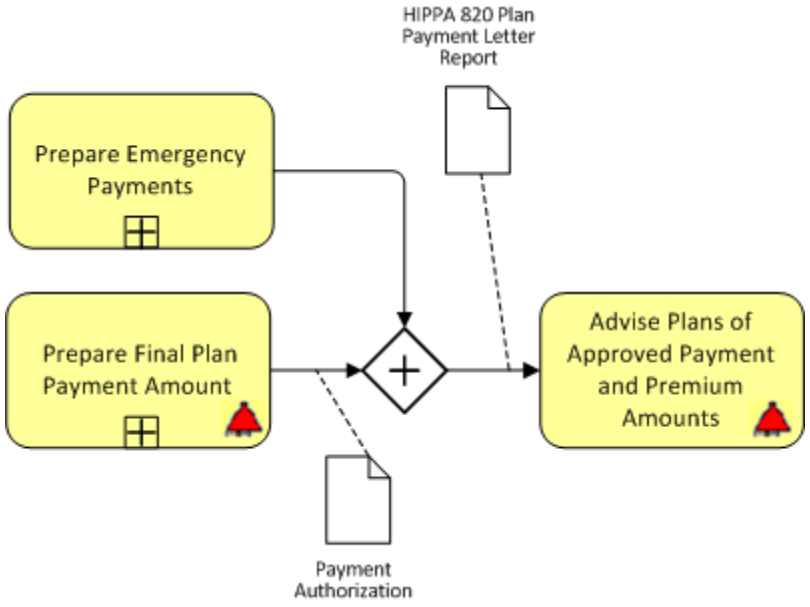
Product Quality Dimension	Priority Level (High, Medium, Low)
Scope (features)	Low
Schedule	High
Defects	Medium
Resources (manpower, budget)	Low

Schedule is most important. It will drive whether the team uses some existing, proven systems vs. investing in new technology. Minimal scope must be reached (enrolling, submitting payment request, calculating payment, payment), but complexity and depth of each task is negotiable, based on time/budget.

3.2 Writing Business Requirements

Business requirements are general statements describing what the stakeholders must do in order to fulfill the business objectives. These business goals address legislative mandates or strategic objectives that the core functionality of the system must support (such as improved customer service, business efficiencies, business process reengineering, etc.). The first place to begin deriving business requirements is the Business Process Model. Consider the following portion of a BPM:

Figure 3 - Sample Business Process Model Fragment



From this portion of the model the following BRs may be identified:

- The business owner shall process monthly payments for plans.

- The business owner shall process emergency payments for plans.
- The business owner shall notify the plans of prospective monthly payments.
- The business owner shall notify the plans of prospective premium amounts.
- The business owner shall secure plan payment information.

A business requirement never identifies what is expected of the system but only addresses what the stakeholder requires. For example, the first requirement below is a business requirement but the second one is a nonfunctional (performance) requirement:

1. The user shall perform their duties at any time during the business week including Saturdays.
2. The system shall be available for use 99.999% of the time.

Each business requirement is further elaborated by business rules and user requirements. Use the following guidelines for writing business requirements:

1. Start every requirement with "The [stakeholder] shall…" to support the business activities that must be performed.
2. The requirement should be written in clear, concise English language only and not reference any data elements, value codes, or other such constructs found in functional requirements or system design documents.
3. Business requirements are derived from legislation, Business Process Models or re-engineering efforts. Indicate what specific model if this is the case. Business Requirements can also specify expected standards or performance criteria.
4. The business requirement may represent the synthesis of the intent of multiple User Requirements.

While requirements are still under development, the typical way of identifying them is by using a temporary unique identifier. The final requirement numbers should be assigned by the Requirements Repository and *not* in the Requirements Document. Examples of business requirements and the identified business process activities include:

Table 4 - Sample Business Requirements

Business Requirement	*Business Process Activity*
CMS shall process Monthly Payments for each applicable Plan.	Monthly Plan Payment Process
CMS shall maintain an accounting of each payment made to a Plan.	Record Payments & Adjustments, Record Approvals
CMS shall trace all Payments that are disbursed or received from a Plan.	Record Payments & Adjustments
CMS shall generate Financial reports based on a Plans' Payment data.	Produce Management Reports.

Business Requirement	*Business Process Activity*
CMS shall trace all Payments that are disbursed or received from a Plan.	Record Approvals, Record Denials
CMS shall track the available funds in the Medicare Trust Fund Accounts for Plans.	Review Trust Fund Balances and Payment Amounts.

Refer to the Style Guide in Section 4 to better understand how requirements should be phrased.

3.2.1 Writing Business Rules

A business rule describes a policy, guideline, standard, or regulation upon which the business operates. A business rule is a statement that defines or constrains some aspect of the business. It is intended to assert business structure, or to control or influence the behavior of the business. The business rules that concern the project are atomic in that they cannot be further decomposed and they are not process-dependent, so that they apply at all times.

Business rules may apply to more than one business requirement and are uniquely identified with the prefix of RU. They typically fall under one of the following categories:

- Term: the meaning of a word or phrase.
- Fact: the relationship between two or more terms.
- Derivation: the result of a calculation or the result of a logical inference based on known information.
- Assertion: the values of a term or fact are considered valid by the business. They may be further broken down as authorization (who may perform an action), condition (the circumstance under which another business rule may be applied) or an Integrity Constraint (which values of a term or fact are permissible).
- Action Enabler: trigger an activity or a message if a certain if condition becomes true.

Most business rules employ the use of a “must” statement. More complex business rules may require a truth table or diagram such as a flowchart or activity diagram. To understand how the rule impacts the business requirement, it is useful to document the rule with the requirement. For example, consider the following business rules for a plan payment process:

- The business owner shall process monthly payments for each plan.
    - Plans must be paid by the 15th of the month. [*action enabler*]
    - Payments must be approved by management. [*assertion*]

3.3 Writing User, Functional and Nonfunctional Requirements

Writing requirements is a difficult task, but the benefits summarized below describe the purpose of capturing requirements:

- To communicate the needs of a user or organization
- To provide a solid foundation for the system or product

- To provide the first view of what the intended product must do
- To provide clear descriptions of how the system should perform
- To serve as a foundation for testing and user acceptance
- To provide a basis for design
- To provide clear goals and measurable objectives
- To show traceability back to the requirement sources
- To define compliance requirements (e.g., FISMA, HIPAA, FIPS, etc.)

Typically the business analyst and the SME will work together closely to produce the requirements.

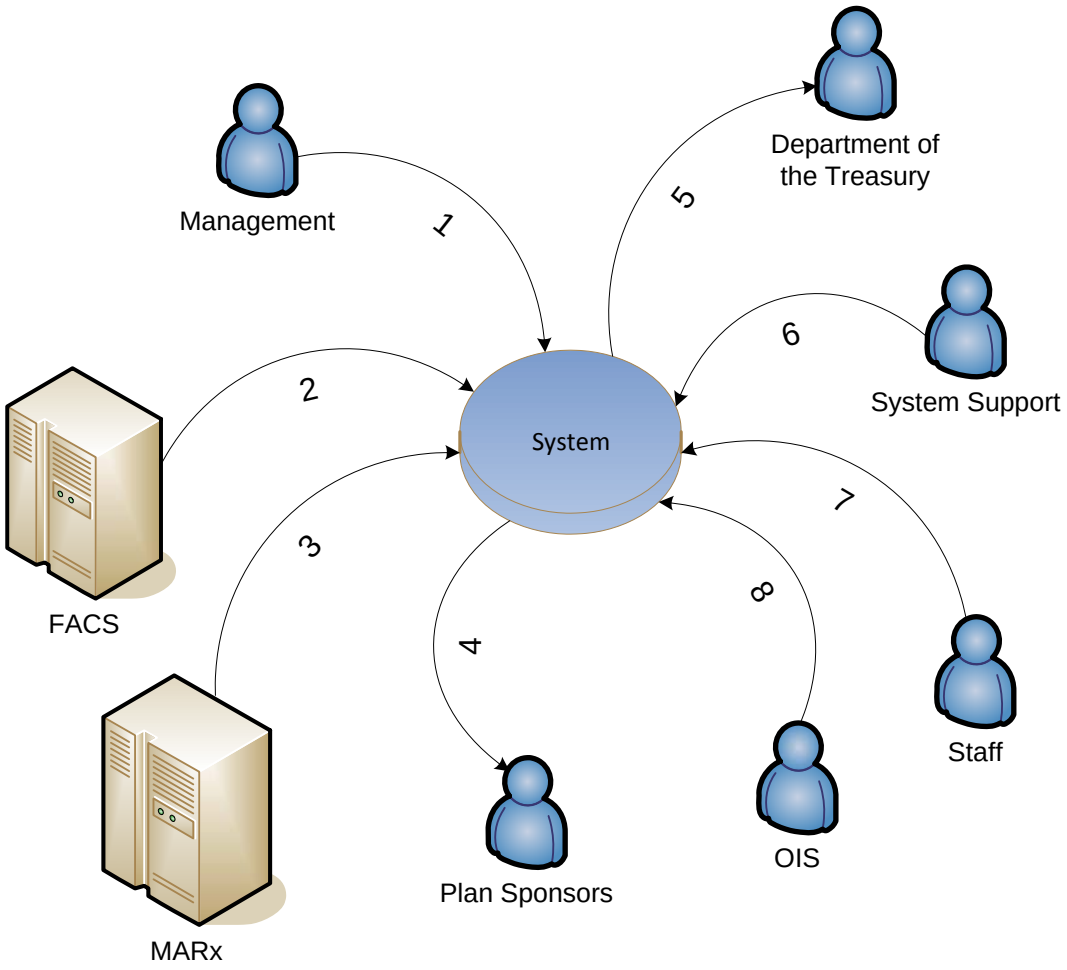
3.3.1 Creating the Project Diagrams

At the beginning of a system development project, it is best to define the boundaries of the system to be analyzed and built. Within the requirements writing process, it is essential to know the scope of this system. One example of a good tool for this purpose is the Work Context Diagram.

The context diagram defines the boundaries of the system being analyzed by showing how it connects to the outside world. The diagram is largely created before you begin to break the system into its functional pieces. It shows how the system under analysis connects to the other stakeholders in the diagram. These connections help convey the precise scope of analysis. The diagram demonstrates the scope of the systems analysis to the users. Without an agreed context, there is no way of telling if the correct system is being analyzed. There is no ideal place to start the analysis and no real place to stop.

The sample work context diagram below shows all entities that will have knowledge of the system and that will interact with it. The direction of the arrows indicates which entity will *initiate* the user requirement. After a user requirement is initiated, there is usually a two-way communication. The arrows of the work context diagram simply show who begins the user requirement. Note that it is possible for the system to initiate a user requirement that interacts with itself or the stakeholders. Timed processes such as an automated backup would fall under this category.

Figure 4 - Sample Work Context Diagram



3.3.2 Writing User Requirements

Writing user requirements is the first step in decomposing business requirements. A user can represent either a stakeholder who interacts with the system or another system that exchanges data with the new system. URs describe how a user intends to interact with the system in order to fulfill a business requirement. A BR may be decomposed into several URs. For example:

- Business Requirement: The staff shall process emergency payments for each applicable Plan.
    - User Requirement: The user shall create emergency payments.
    - User Requirement: The user shall modify emergency payments.
    - User Requirement: The user shall delete emergency payments.
    - User Requirement: The user shall view emergency payments.
    - User Requirement: The user shall authorize emergency payments.

Using the work context diagram the project team identifies the list of user requirements initiated by that domain for each line and identifies the business requirement that that it derives from. For the sample work context diagram, these included:

Staff to System (line 7)

- Staff shall issue monthly payments.
- Staff shall issue interim payments.
- Staff shall issue emergency payments.
- Staff shall manage accounts receivable.
- Staff shall authenticate themselves to the system.

Writing a purpose for each UR will help reduce any ambiguity as to why the UR exists. All URs should use an action verb and indicate who is doing what to whom. For example, use 'The [stakeholder] shall view reports' or 'The [stakeholder] shall create a report'. In the examples above, the Staff is doing something (creating) to payments. The project manager is responsible for identifying these URs with the assistance of subject matter experts. Be wary of losing focus by adding too much detail here.

From the URs, scenarios will be generated that describe in more detail how the system will fulfill the UR. Functional and nonfunctional requirements can later be generated for both URs and scenarios.

In some cases, when the number of scenarios is expected to be very limited, it may make the most sense to create a higher level UR to encompass all the scenarios envisioned. For example:

- Business Requirement: CMS shall process Emergency Payments for each applicable Plan.
    - User Requirement: The user shall maintain emergency payments.
        - Scenario 1: Create Payments
        - Scenario 2: Modify Payments
        - Scenario 3: Delete Payments
        - Scenario 4: View Payments
        - Scenario 5: Authorize Payments.

Not all stakeholders will necessarily have user requirements identified. For example, OIS sets policy for the technical architecture of a system and hence only nonfunctional requirements will be generated from that stakeholder.

The business analyst uses their best judgment on how to clearly organize the URs and scenarios given the time and budget allowed. Note that in order to avoid identifying duplicative requirements or scenarios, it is permissible to categorize scenarios and requirements that are applicable for all users within a global requirements section in the Requirements Document. Examples of global requirements include performance and authentication requirements. You may also want to add a brief flow diagram to indicate the context of the user requirement as this can make it easier for the reader to understand the context of when the UR occurs.

3.3.3 Documenting Scenarios

Scenarios, which are similar to use cases, describe the steps a user takes to satisfy a user requirement. They are a convenient structure around which to organize requirements. There are usually several scenarios associated with a user requirement. The steps of a scenario typically encompass all the functional requirements for that scenario. Scenarios are particularly helpful

because they encourage the complete analysis of the system's failure or error conditions. By starting with the user's goals and understanding not only how the system guarantees the successful achievement of these goals but also how the system responds when it cannot achieve these goals, resulting specifications and testing scenarios are more complete.

All the user requirements indentified in the work context diagram should have at least two or more scenarios associated with it. The first scenario is the primary scenario and is usually the most common way the subject matter expert believes the scenario will play out.

Often it helps to diagram a scenario before writing the formal description using a flow chart. Any diagramming methodology the team is familiar with is acceptable. When documenting a scenario the following aspects must be considered:

- What are the preconditions that must be met before the scenario can occur? For example, before an emergency payment can be made, the plan information must exist in the system. Thinking through the preconditions can sometimes lead to new URs.
- What are the triggers that cause the scenario to occur? Is there a schedule that triggers the scenario? Multiple triggers will often indicate a need for multiple scenarios.
- What is the expected result of the scenario? Knowing this will help you identify alternate scenarios, aid in testing the scenario and clarify the purpose of the scenario.
- Steps are the process that must be followed in order to go from the trigger to the expected result. This typically is a back and forth process between one or more users interacting with the system or performing some business process.

A portion of a sample scenario is outlined below. Note one means of how to handle conditional situations along the way. In some cases, the condition results in the scenario terminating before the scenario is complete – this is normal. Note too that functional requirements are documented as part of the scenario in order to best understand the context of the requirement.

1. Staff User Requirements

1.1 Create Emergency Payments

The user shall create emergency payments.

1.1.1 Purpose

The purpose of this UR is to provide a means for the user start the process of making an emergency payment.

1.1.2 Primary Scenario: Create Payment for Existing Plan

This scenario covers creating a payment when the plan to be paid was paid last month and the plan contract was not scheduled for termination in a prior month.

1.1.2.1 Precondition

The amount the plan was paid in the previous month is available.

1.1.2.2 Trigger

The scheduled plan payment is due and the calculated plan payment for the current month is not available.

1.1.2.3 Expected Result

An emergency payment is created and ready for approval.

1.1.2.4 Steps

1.1.2.4.1 User requests to view a list of plans paid in the previous month and are scheduled to receive a payment in the current month.

1.1.2.4.2 The system displays the requested list of plans.

1.1.2.4.3 The user selects one or more plans to create an emergency payment.

1.1.2.4.4 The user selects an explanation text to assign to the payment from the pre-populated list of explanations or enters a new explanation text.

1.1.2.4.5 If the user enters new explanation text:

1.1.2.4.5.1 The system prompts the user to confirm creation of a new explanation text.

1.1.2.4.5.2 If the user confirms the prompt, the system adds the new text to the pre-populated list.

1.1.2.4.5.3 If the user rejects the prompt, the system clears the explanation text entered and returns to step 4.

.
.
.

1.1.4 Scenario: Create Payment for New Plan

This scenario covers creating a payment when the plan to be paid was not paid last month and the plan contract was not scheduled for termination in a prior month.

.
.
.

You should expect to document complex URs using several scenarios. Name the scenario and include a description to help distinguish what is unique about each scenario.

3.3.3.1 Alternate Scenarios

Alternate scenarios are intended to document how unexpected exceptions are handled by the system. These are often discovered by going through each step in the scenario outline(s) and speculating on what could go wrong at that point. Exceptions could include system resource problems (e.g., connection is dropped or disk runs out of space).For example:

1.1.4 Alternate Scenario: Create Payment for Existing Plan

This scenario covers creating a payment when the plan to be paid was paid last month and the plan contract was not scheduled for termination in a prior month.

1.1.4.1 Precondition
The amount the plan was paid in the previous month is available.
1.1.4.2 Trigger
The scheduled plan payment is due and the calculated plan payment for the current month is not available.
1.1.4.3 Expected Result
An emergency payment is created and ready for approval.
1.1.4.4 Steps

.
.
.

3.3.4 Writing Functional & Nonfunctional Requirements

As the scenarios are fleshed out, requirements should be identified along the way. Requirements will always clearly state who is doing what to whom. They identify *what* is expected. For example:

> The system shall allow access to only credentialed users.

Functional and nonfunctional requirements should always be stated in the form of what is required of the system, not what is required of the user. For example, after writing a log in scenario the following requirements may apply:

1. The system shall require a user to change their password monthly.
2. The system shall require that a user not use the same password twice in a row.

This makes it clear what is expected from the system as well as the conditions/constraints that are imposed on the capability. Contrast requirements 1 and 2 above with the following:

a) The user will change his password monthly.
b) A user cannot use the same password twice in a row.

Requirements a) and b) sound like policies not functional requirements.

When documenting a requirement, it is also helpful to consider the following attributes of the requirement.

- What is the purpose of the requirement?
- Does the requirement conflict with another requirement? If it does, there may be some additional scenarios or a business decision that needs to be made.
- Does the requirement depend on another requirement? If it does, does the scenario account for this?
- Is the requirement modular enough so that it can be changed without excessive impact on the system?

- What is the precedence and criticality of the requirement? Low precedence/critical requirements could be postponed for later releases should the schedule become a problem. Requirements may be rated as Essential, Conditional (enhances final system) or Optional (may not be worthwhile).
- Track the history of the requirement. When was it approved and by whom? When was it changed? When was it deleted? Who acted on it? What was the rationale?

It is generally simpler to document the attributes of a requirement using a requirements repository.

Avoid creating requirements that specify a particular design or implementation method (unless there is a policy that constrains these options). For example, this requirement defines how the system will be designed, it is a 'how' statement:

> The System shall use a Web interface.

While this might be a valid requirement if the organization's policy is to use Web applications only, the more acceptable way to write this would be:

> The system shall allow access for remote, non-network users.

Finally, ensure that the requirement is feasible. Can it be done within the scope of budget and time allotted to the project? If not, several options exist:

- Throw out or postpone the requirement (for a later release)
- Change the requirement or its parent dependency
- Cancel the project (assuming the requirement is critical)

Refer to the style guidelines in Section 4 for more information on phrasing requirements.

3.3.4.1 Types of Nonfunctional Requirements

Nonfunctional requirements will detail the properties of the system. In the process of documenting these requirements, it's not unusual to uncover additional functional requirements which may need to be captured. Nonfunctional requirements include:

- Required States and Modes: identify the various states or modes in which the system can exist
- Performance Requirements: this is the expected response time of critical system functions or intervals at which functions are expected to trigger. Other types of performance requirements include safety, up time and load.
- Security Requirements: identify the security measures (management, operational, and technical) that provide adequate protection against threats and vulnerabilities to the system based on its system security level. Consider any audit checks or logging that must be performed (e.g., content and handling of log files). Information Security requirements should be based on the CMS information security policies and standards found at http://www.cms.hhs.gov/InformationSecurity/ which provides guidance on how to implement federal security standards such as FISMA, FIPS and NIST.

- Privacy Requirements: this would include privacy (HIPAA) and data protection requirements (e.g., media control, files, data transmission, archives and encryption). CMS policies on privacy requirements can be found at http://www.cms.hhs.gov/InformationSecurity/.
- Section 508 Requirements: this would include Section 508 requirements that apply to all government systems based on the specifications of the contract/task order or the technologies in use. Please note that depending upon the scope of your project, it is possible that your project may need to comply with additional CMS Section 508 standards. These standards can be found at http://www.cms.hhs.gov/InfoTechGenInfo/03_Section508.asp.
- Legal Requirements: these include other rules and regulations imposed by the government or other regulatory agencies outside the Agency.
- Human-factors Engineering (Ergonomics) Requirements: these are also referred to as Look and Feel requirements. They may take the form of interface sketches or more detailed scenario models. They will help the designer understand how the user works with the functionality of the system. Identify areas that need concentrated human attention and are sensitive to human errors. Also consider manual operations to be performed that affect the system.
- Design Constraints and Qualification Requirements: identify what laws the system must uphold as well as any applicable standards the system must comply with. Consider any organizational policies that will affect the operation or performance of the system. List any existing components that could be reused.
- System Quality Characteristics: quantify the expected accuracy of the results produced by the system such as precision or units of measure.
- Internal Data Requirements: identify the essential objects/entities/classes that are germane to the system. This could be in the form of a first cut data model or a domain model. Consider integrity constraints, as well as retention and data replication requirements. Determine if any data conversion is required or any Interface Control Documents that must be negotiated or adhered to.
- Usability: identify the ease of learning for the expected users of the system. This would include statements of how easy the system is to use from the perspective of all users. A description of the help services that the system will provide would also be appropriate.
- Operational: specify the physical environment in which the system will operate. This would include statements on the expected technological environment and partner applications that the system will interface with.
- Maintainability: identify the amount of time necessary to make specified changes to the system or if there are special conditions that apply to the maintenance of the system.

3.3.5 Determining Pass/Fail Statements

The pass/fail statement should be a statement that describes how we will know if the requirement was met. Include the pass/fail statement with the description of the requirement for easy reference. For example:

- The system shall require a user to change their password monthly. (requirement)
  Pass/Fail statement: Upon login, the system will check to see if more than 1 month has passed since the password was last changed and prompt the user to choose a different password when that occurs.

Writing the pass/fail statement can also help you see if there are extra steps (or a whole scenario) that were overlooked and if your requirement is ambiguous. In this example, the pass/fail statement reveals an additional requirement about how to handle an error condition:

- The system shall require that a user select a different password when required to change their password. (requirement)
  Pass/Fail statement: If the user enters their existing password as their new password, the system will notify them that their entry is invalid and request a different password.

Writing the pass/fail statement now not only helps the designer understand the requirement better, it makes writing test procedures simpler. It may also help determine whether this is a requirement or an assumption. For example, consider the following requirement:

- The system shall utilize the user's network username as their logon id.

If your system lacks the ability to test whether or not the user's network username actually exists, what you have uncovered is an assumption rather than a requirement.

Pass/Fail statements may also be written from a prospective of whether a requirement from the standard scenario or the alternate scenario is being evaluated. For example, these two requirements are applicable in both scenarios but the pass/fail statements reflect what is being evaluated in light of the expected result of the scenario:

Standard Scenario:

- The system shall authenticate a user after credentials are validated.
  Pass/Fail statement: The user is authenticated to the system when valid credentials are submitted.

Alternate Scenario:

- The system shall authenticate a user after credentials are validated.
  Pass/Fail statement: Invalid user credentials are rejected when submitted.

4.0 STYLE GUIDELINES

The process of documenting requirements is a never ending cycle. When using thorough writing techniques, it is quite easy to spend a considerable amount of time eliminating ambiguity in order to ensure all assumptions, as well all the requirements, have been identified. This process inevitably leads to new requirements that should be examined as well. It falls on the project manager to decide what level of detail and investigation is required for the Requirements Document before a diminishing amount of return is gained from effort put forth.

This section of the guide discusses what might be considered a 'middle level' investigation of requirements. Proper layout, form and phrasing of requirements can help ensure the document is complete and clear. For an even deeper level of requirements specification, please refer to Appendix B.

4.1 Writing Style

CMS uses the Government Printing Office (GPO) Style Manual as the basis for writing. The GPO Style Manual is available through the following URL:

http://www.gpoaccess.gov/stylemanual/browse.html. In some cases, the procedures in this guide depart from GPO rules. This guide takes precedence over conflicting GPO rules.

A properly structured requirement must be a complete sentence that has only one interpretation. Any individual who writes requirements should avoid using phrases, single words or a collection of acronyms to express requirements. Each requirement should consist of a subject, verb and predicate as noted in the requirement written below:

Requirement example:

> The Medicare Managed Care System shall assign a disenrollment date for a beneficiary in a Managed Care Organization when a termination date for the beneficiary's Part B eligibility is received.

The subject of the requirement is "The Medicare Managed Care System" and the predicate is "assign a disenrollment date for a beneficiary in a Managed Care Organization when a termination date for the Beneficiary's Part B eligibility is received." The subject of any requirement should imply ownership and shows the entity to which the requirement should be related. The predicate should be an action phrase or expression of something that must be accomplished for, by or to the subject. The verb used in requirements is always 'shall'. This implies that the requirement must be adhered to.

Keep the following in mind:

- Write in such a way that the requirement does not pose a specific solution. It should always be design independent.
- Write in such a way that there is only one interpretation of the requirement.
- Increase readability by: using grammatically correct language and symbology; using simple words/phrases/concepts; define unique words or symbols.

4.2 Eliminating Ambiguity

All natural languages (e.g., English) are inherently ambiguous. Use the following techniques to assist in making the requirements document clearer:

- **Multiple requirements in the same requirements statement** – Do not use conjunctions (*and, or, with, also*). However, if the requirement has two or more conditions that must be met simultaneously, then conjunctions may be used.
- **Let-out or escape clauses** – Do not use the following expressions: *if, but, except, unless, possibly, eventually or although.*
- **Rambling** – Avoid long sentences.
- **Mixing different kinds of requirements** – Do not mix business requirements with functional requirements in the same section of the document.
- **Speculation** – Avoid speculative words such as usually, generally, often, normally, or typically.
- **Vague terms** – Do not use unverifiable, subjective terms (typically adjectives). Look for:
    - adjectives related to intrinsic characteristics: clear, well, easy, strong, weak, good, bad, efficient, low, user-friendly, flexible, effective etc.

- adjectives related to environmental characteristics: useful, significant, adequate, fast, slow, versatile, a minimum, as applicable, as appropriate, etc.
- adjectives related to time characteristics: old, new future, recent, past, today's, normal, timely, etc.
- adjectives related to location characteristics: *near, far, close, back, in front, etc.*
- adverbs: approximately, be able to, be capable, but not limited to, capability of/to

Replace these with actual measures.

- **Suggestions or possibilities** – Do not use terms such as can, optionally, may, might, should, ought, could, perhaps, or probably.
- **Wishful thinking** – Avoid phrases such as 100% reliable, totally safe, handles all unexpected failures, pleases all users, runs on all platforms, never fails, or fully compatible to all future situations.
- **Negative statements** – Identification of duplicate requirements is facilitated if requirements are consistently written in the affirmative. In addition, negative statements can be impossible to test.
- **Jargon** – Jargon often uses terms that all readers are not familiar with. This includes acronyms, abbreviations and abstract words or phrases.
- **Passive Voice** - Verbs may be either *active* (The executive committee approved the new policy) or *passive* (The new policy was approved by the executive committee) in voice. In the active voice, the subject and verb relationship is straightforward. In the **passive voice**, the subject of the sentence is acted upon by some other agent or by something unnamed. Active voice is stronger and engages the reader.
- **Implicit Subjects** – an implicit subject is inherently vague and open to interpretation. Implicit subjects are found in sentences where the subject:
  - contains a demonstrative adjective indicated by *this, these, that, those*
  - is expressed by means of pronouns indicated by *it, they*
  - is specified by a preposition such as: *above, below*
  - is specified by an adjective such as: *previous, next, following, last, first*

  For example: *This counter* shall be incremented by 1 each time an update occurs.
- **Under referencing** – documents or entities not explicitly defined when referred to can create ambiguity. Look for phrases such as: *according to, on the basis of, relative to, compliant with, conform to, etc.* For example: The software shall be designed according to the rules of object oriented design.
- **Multiple Terms for the Same Subject** – Do not use different terms for the same subject. Agree on a uniform, or standard, term to be used to describe the same thing. For example in the Medicare Program, the "standard" term that describes a patient, a subscriber, or a beneficiary is "beneficiary." The Glossary located within the Medicare.gov Website may be helpful in standardizing terms.

5.0 ACRONYM LIST

BPM	Business Process Model
BR	Business Requirement
CMS	Centers for Medicare and Medicaid Services
DOORS	Dynamic Object Oriented Requirements System
DRAV	Division of Requirements and Validation
EIA	Electronic Industries Association
FIPS	Federal Information Processing Standards
FISMA	Federal Information Security Management Act
FR	Functional Requirement
GPO	Government Printing Office
GTL	Government Task Leader
HIPAA	Health Insurance Portability and Accountability Act
IEEE	Institute of Electrical and Electronics Engineers
IS	Information Services
IT	Information Technology
NIST	National Institute of Standards and Technology
NR	Nonfunctional Requirement
OIS	Office of Information Services
RU	Business Rule
RWG	Requirements Writer's Guide
SME	Subject Matter Expert
UR	User Requirement

APPENDIX A ATTRIBUTES OF REQUIREMENTS AND RULES

These attributes help ensure completeness and to help standardize how requirements and rules are documented and tracked in the Requirements Repository. However, the unique characteristics of each project may necessitate the tailoring of attributes within this appendix.

A.1 Business Requirement Attributes

A.1.1 Directions

Requirement Number – Provide a unique identifying number. This number will be important when establishing traceability.

Requirement (the "shall" statement) – Provide a complete sentence using the word "shall" that describes the capability needed by the business to meet their objective.

Source (optional) – The legislation or strategic objective document, the BPM, information regarding meetings or emails from which this requirement originated.

Priority (optional) – Identifies the requirement as "High" if essential to the project/system, "Medium" if desirable, but not essential, and "Low" if optional.

Purpose (optional) **–** A brief rationale for the requirement.

Comments (optional) – Any additional information associated with the requirement is captured under the comments column for the requirement.

A.1.2 Example

Requirement Number – BR-001

Requirement - CMS shall implement its payment calculation using established Medicare processes to ensure the integrity of the payment calculation methodology.

Priority - High

Purpose - This is why CMS must interact with Grouper, OCE, OCE for Indian Health, MCE, Pricer, UPIN, Oscar, Physician Fee Schedule, and Ambulance Fee Schedule, and EDB

Source - Medicare Modernization Act of 2003

A.2 Business Rule Attributes

A.2.1 Directions

Rule Number – Provide a unique identifying number. This number will be important when establishing traceability.

Rule – Provide a complete sentence that defines or constrains some aspect of the business.

Associated Business Requirement(s) – Provide the business requirement number(s) of the parent business requirement(s).

Type – the category of the rule (Term, Fact, Derivation, Assertion, Action Enabler).

Source – The legislation, strategic objective or policy guidance document.

A.2.2 Example of a Business Rule

Rule Number – RU-0099

Rule – A Plan Sponsor is an entity that sponsors a health plan. This can be an employer, union, or some other entity.

Associated Business Requirement(s) – BR-001, BR-002.

Type – Term

Source – CMS Website

A.3 User Requirement Attributes

A.3.1 Directions

Requirement Number – Provide a unique identifying number. This number will be important when establishing traceability.

Requirement (the "shall" statement) – Provide a complete sentence using the word "shall" that describes the capability needed by the business to meet their objective.

Associated Business Requirement – Provide the user requirement number of the associated business requirement.

Source (optional) – The stakeholder who identified or requested the requirement.

Priority (optional) – Identifies the requirement as "High" if essential to the project/system, "Medium" if desirable, but not essential, and "Low" if optional.

Purpose (optional) **–** A brief rationale for the requirement.

A.3.2 Example of a User Requirement

Requirement Number – UR-001.

Requirement (the "shall" statement) – CBC shall issue monthly payments.

Associated Business Requirement – BR-100

Purpose (optional) **–** This allows for plans to be paid for services.

Source (optional) – CBC

A.4 Functional/Nonfunctional Requirement Attributes

A.4.1 Directions

Requirement Number – Provide a unique identifying number. This number will be important when establishing traceability.

Requirement (the "shall" statement) – Provide a complete sentence using the word "shall" that describes the action or expectation of what the system will do (functional requirement), or the behavioral property the system must have.

Purpose (optional) **–** A brief rationale for the requirement

Validation Measure (Demonstration, Test, Analysis, Inspection, Special Qualification Method, or Not Applicable) – Provide a description of how the requirement will be validated.

Pass/Fail Statement – A testable statement that describes how to know if a requirement was met.

Associated User Requirement – Provide the user requirement number of the associated user requirement.

Dependent Requirement – List any requirements that this requirement is dependent on.

Priority (Essential, Conditional, Optional) – Provide the priority of implementing this requirement from the user perspective. Essential means "must have," Conditional means "like to have," and Optional means "could do without".

Source – Provide the name of the stakeholder who identified the requirement.

Requirement Type – For nonfunctional requirements, the category of requirement it falls within.

Comments – Please provide any comments if necessary.

Note[1]: Nonfunctional requirements will be a subset of the RD. Be sure to include which category (from Section 3.3.2.1) the nonfunctional requirement falls into.

Note[2]: These attributes convey the recommended information to be captured in each requirement. Project managers are encouraged to add any other information that is appropriate for their projects. If the requirements are initially written in software other than DOORS, the formatting of this information is left to the discretion of the project manager. Multiple requirements may be displayed on one page. A tabular format (e.g., as in MS Word or MS Excel tables) may also be used.

A.4.2 Example of a Functional Requirement

Requirement Number – FR-001

Requirement (the "shall" statement) – The system shall be able to confirm that a check has not been cashed.

Purpose – This allows the Customer Service Representative to know if the customer has already received payment.

Validation Measure (Demonstration, Test, Analysis, Inspection, Special Qualification Method, or Not Applicable) – Test

Pass/Fail Statement – Examining the current status of the check will indicate if the check has been cashed.

Associated User Requirement – UR-001

Dependent Requirement – None

Priority (Essential, Conditional, Optional) – Essential

Source – John Doe

Comments – Implement in Release 2

A.4.3 Example of a Nonfunctional Requirement

Requirement Number – NR-009

Requirement (the "shall" statement) – The system shall meet Section 508 compliance for all interface screens.

Purpose – Section 508 is a legal requirement which enhances the accessibility of electronic and information technology.

Validation Measure (Demonstration, Test, Analysis, Inspection, Special Qualification Method, or Not Applicable) – Test

Pass/Fail Statement – Each screen is evaluated according to the HHS checklist for web based applications.

Associated User Requirement – UR-002

Dependent Requirement – None

Priority (Essential, Conditional, Optional) – Essential

Source – Jane Smith

Requirement Type – Legal compliance

Comments – None

APPENDIX B ADVANCED STYLISTIC APPROACHES

Natural language is widely used in industry to state requirements for all types of systems, because it is flexible and universal. However, natural language has one major drawback, which is its inherent ambiguity. This appendix describes methods that may be used to reduce ambiguity in written requirements documents.

B.1 Deletion

The process of deletion reduces the perception of a person to a scope they can deal with. In other words, things that seem to be not important are not recognized by a person or in the context of requirements not mentioned in a requirements document.

B.1.1 Implicit Assumptions

Implicit assumptions are those statements that are obvious or banal to the writer and are therefore omitted. Employ these techniques to aid in making implicit assumptions explicit.

Determine the main verb of a sentence. Form a new sentence by negation of the verb. Question: Which statements must be true in order to guarantee that both sentences make sense? All statements found may indicate an implicit assumption.

Requirement example:

> The Medicare Managed Care System shall assign a disenrollment date for a beneficiary in a Managed Care Organization when a termination date for the beneficiary's Part B eligibility is received.

The negation of this requirement would be:

> The Medicare Managed Care System shall not assign a disenrollment date for a beneficiary in a Managed Care Organization when a termination date for the beneficiary's Part B eligibility is not received.

The statements (assumptions) that must be true in order to guarantee that both sentences make sense include:

- that the beneficiary's Part B eligibility will be transmitted
- that the beneficiary is part of a Managed Care Organization

Under specified process words are verbs, adjectives and adverbs that aren't sufficiently quantifiable.

To detect under-specified process words determine the verbs of the sentence. If it is possible to have sentences with multiple actors something was deleted in the original sentence (e.g. the word transmit implies the questions what is transmitted, what are the aim and the source of the transmission).

B.1.2 Process Words

Process words are verbs, adjectives and adverbs. These types of words are often ambiguous and should be replaced by quantifiable measures.

- The system shall supervise the resources assigned to the patient care to ensure the maximum limit is not reached.

Use questions about the process words to determine if they are sufficiently specified:

- Supervise: what exactly is the system supervising and how?
- Resources Assigned: who or what can be assigned?
- Ensure: when and how is the maximum limit ensured?
- Maximum: What is the maximum limit?

B.1.3 Incomplete Superlatives and Comparisons

Incomplete superlatives and comparisons need a reference point to be completely specified. Moreover, there must be a metric and a predefined scale to compare certain aspects.

Use the following questions to detect incomplete superlatives and comparisons: Is there a reference point? What is the metric to measure a certain characteristic? What is the scale of the metric? For example:

- The functions should be easily changeable.

'Easily changeable' infers a reference point to something that is not easily changeable. What is the feature being compared to? Of course, 'easily' is also a process word that should be examined (see above).

B.1.4 Modal Operators of Possibility

In addition to the "what" a system shall do, the means to realize the "what" must be specified. Examples of modal operators of possibility include: *may*, *can* etc. Statements that describe a possibility or impossibility shall be analyzed by using the question: Which means enable or prevent a certain system reaction. For example:

- Updating a patient's pre-existing conditions may not overwrite any existing history.

By what means is the update prevented?

B.1.5 Modal Operators of Necessity

Ensure that statements regarding the system's behavior contain the desired behavior and the exception behavior. Examples for modal operators of necessity include: must, shall, and should. Each sentence that contains a modal operator of necessity shall be inspected if it is necessary to define the behavior of the system in the case of an exception. For example:

- The system shall forward confirmation of the activity to the user.

What if forwarding services are not available or an invalid forwarding destination is provided?

B.2 Generalization

Generalization is defined as a process that leads to a detachment of an experience from its context and to assume that the experience is overall valid. In the requirements analysis process,

generalization may lead to the omission of several behavior aspects of the system (e.g. special cases or error exceptions).

B.2.1 Universal Quantifiers

These elements are statements about a certain amount. They summarize a set of objects. Using universal quantifiers may lead to the problem that some objects summarized in the set do not realize the specified behavior. Universal quantifiers include: *never, ever, not any, everyone, no one, someone, or nothing.*

Find all universal quantifiers. Check if the specified behavior of the system is really valid for all objects of the set summarized by the universal quantifier. Furthermore, check each sentence if there is an explicitly defined set of objects, the specified behavior is valid for. For example:

- Each communication shall be marked with a timestamp.

Every communication? Are there no exceptions?

B.2.2 Incomplete Specified Conditions

Conditions have the following sentence structure: *if condition then behavior1 else behavior2*. In requirements specifications the second case is often omitted. Key words to look for are: *when, then, if, in the case of, dependent on.*

Find the condition of the statement. Be sure that the requirement specifies the behavior for all possible cases (then and the else branch respectively). Additionally the analyst may check the following questions: Are all the possible conditions covered? Are all possible variants described? For example:

- In the case that the automatic data communication is disrupted, it should be possible to edit the individual fields of the transmission received.

What is possible if no disruption occurs?

B.2.3 Nouns without a Reference

This omission occurs when a noun of a statement is not specified sufficiently. Examples for insufficiently specified nouns are: *the user, the controller, the system, the message, the data, and the function*.

Check each noun of the requirements specification to determine if it specifies a defined person, a defined group of persons, or a defined real world object (group of objects). Insufficiently specified nouns can be identified by means of the following questions: *Who exactly? What in detail? Which part of the set?* Each noun specified insufficiently must be extended by a completion, which specifies the noun exactly. For example:

- The system shall display the data to the user electronically.

Which data exactly? Which users exactly?

B.3 Distortion

This phenomenon is related to the so-called nominalization, i.e., a noun stands for a complex process. Nominalization does not cause any problems as long as the process is unique, exactly

specified and unambiguous in the current context. Examples of nominalizations include: the recording, the playback, the take off etc.

Check each noun of all sentences, with regard to the question: Is it possible to find a verb which describes a process and which is similar to the noun (for example, a ‘recording’ is both a noun and a verb)? You can use the following phrases:

- Is it possible to add the substantive to the phrase: a continuous...?
- Does the noun describe something that is not touchable?

If a question is answered with ‘yes’, the analyst has to check to see if any information was lost when describing a process by means of a noun. For example:

- The system shall report on the status of the activity.

What is reported? Where is it reported?

B.4 Other Guidance

An overall quality aspect of requirements statements is non-ambiguity. In the English language the frequent use of the gerund leads to misunderstandings and ambiguities. When using the gerund, the relation of the verb and the subject, performing the verb, is lost. For example:

- Reporting violations should be immediate.

This has two interpretations:

- Violations should be reported immediately.
- Reports of violations should appear immediately.

B.5 Writing Stronger Sentences

The federal government encourages using simpler words or phrases and avoiding the use of words or phrases that may unnecessarily complicate documentation.

INDEX

T

U

V

W