Software Quality Assurance Plan

For Database Applications

Document ID: SQAP
Version: 2.1a

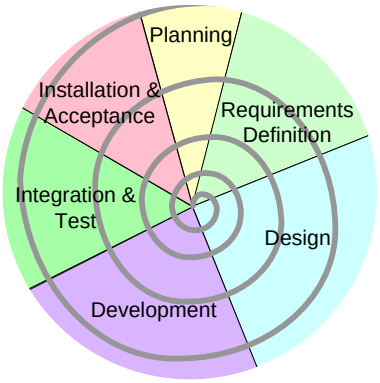


TABLE OF CONTENTS

INTRODUCTION

This document represents the Software Quality Assurance Plan (SQAP) for software development.

Software Quality Assurance (SQA) is a formal process for evaluating and documenting the quality of the work products produced during each stage of the Software Development Lifecycle (SDLC). The primary objective of the SQA process is to ensure the production of high-quality work products according to stated requirements and established standards.

SCOPE

This SQA process is tailored to fit the current software development effort and is related to the project planning and lifecycle description documents for this project.

METHODOLOGY

The methodology presented here is based on the Software Engineering Institute's (SEI) Capability Maturity Model (CMM) and the Institute for Electrical and Electronics Engineers (IEEE) standards for Information Management. This SQA process:

- Makes use of the principal project participants as defined in the SDLC and SPMP.
- Describes the processes for deliverable reviews and software testing.
- Defines deliverable class standards to be applied during reviews of stage deliverables.
- Identifies the work products produced as a result of the review and testing efforts.

The SDLC defines a series of stages; each stage is defined as a separate operation with specific inputs and outputs. This SQAP implements assessments and reviews at specific points within each of these stages.

Please refer to the SDLC for a description of the structure, inputs to and outputs from each of the stages. The terms and stage descriptions defined there are used extensively in this SQA plan.

Other terms common to the software development process are defined in a Glossary of Software Engineering Terms, available at:

http://www.shellmethod.com/refs/seglossary.pdf

Please refer to this glossary for definitions of the terms used in this document.

FORMAL REVIEWS

For each project deliverable, as many as three types of formal reviews are conducted after the end users and development team have informally agreed that the deliverable content is accurate. The three review types are:

1. End-user review, conducted by at least one Subject Matter Expert (SME) who is familiar with the software product under development.
2. Technical review, conducted by at least one experienced software developer who is familiar with the product under development.
3. Quality Assurance review, conducted by an independent Quality Assurance Reviewer (QAR).

Each review is conducted in alignment with the reviewer's area of expertise and in accordance with the review criteria described in the associated Deliverable Class Standard and review form. Refer to Chapter 2 for a discussion of Deliverable Class Standards. By tailoring the review focus to the expertise of the reviewer, this SQA plan prevents redundancy and inappropriate reviews.

PERSONNEL ROLES & RESPONSIBILITIES

In a database development effort, three principal roles are defined:

1. Primary End-user Representative (PER)
2. Primary Developer Representative (PDR)
3. Quality Assurance Reviewer (QAR)

The PER acts as the primary point of contact and principal approver for the end-user community. The PER is responsible for ensuring that end-user reviews are conducted on time and by appropriate subject matter experts.

The PDR acts as the primary point of contact and principal approver for the developer community. The PDR is responsible for the conduct of technical reviews in a timely manner and by appropriate development team members.

The QAR acts as the independent quality assurance reviewer for the project. The QAR will work independently from the development team to ensure objective audits and reviews of the work products and processes of this software development project.

STANDARDS

The following standards were used as guides to develop this SQA process. The standards were reviewed and tailored to fit the specific requirements of small database projects using the referenced SDLC:

- ANSI/IEEE 730.1: Standard for Software Quality Assurance Plans
- ANSI/IEEE 1028: Standard for Software Reviews and Audits
- ANSI/IEEE 1012: Standard for Software Verification and Validation
- SEI/CMMI: PPQA key process area

VERIFICATION AND VALIDATION

The following IEEE definitions apply to this SQA plan:

Verification: The process of determining whether or not the products of a given stage of the software development life cycle fulfill the requirements established during the previous stage.

Validation: The process of evaluating software at the end of the software development process to ensure compliance with software requirements.

VERIFICATION

Best practices in SQA identify the following activities as part of requirements verification:

- Evaluate requirements and relationships for correctness, consistency, completeness, accuracy, readability and testability.
- Assess how well the requirements document satisfies the high level requirements described in the project plan.
- Assess the criticality of requirements to identify key performance or critical areas of software.
- Produce a traceability matrix tracing all requirements back to high-level requirements in the project plan and forward to software design elements and downstream test case elements.

The first two activities are handled by the review cycles and DCS documents described in the following chapters. The last two activities are handled by the development of a Requirements Traceability Matrix, as described in the following section.

REQUIREMENTS TRACEABILITY

The linkages between test cases, their parent design elements, and their parent requirements are maintained through a technique termed Requirements

Traceability. In essence, it is necessary to be able to trace the linkage from a test case all the way through to a grandparent goal.

Requirements traceability is managed through the establishment and maintenance of a Requirements Traceability Matrix (RTM). The RTM is first established during the Requirements stage and is one of the formal deliverables for that stage. The RTM is then updated during each of the subsequent stages and reviewed to ensure that all elements have appropriate linkages (there are no "orphans").

A RTM can best be envisioned as an outline. As with any outline, parent elements are on the left and child elements move successively to the right.

- G1: High-level requirement 1
  - R1-G1: Requirement 1
    - D1-R1: Design element 1
      - T1a-D1: Test Case item 1a
      - T1b-D1: Test Case item 1b
  - R3-G1: Requirement 3
    - D4-R3: Design element 4
      - T6c-D4: Test Case item 6c
    - D7-R3: Design element 7
      - T5c-R3: Test Case item 5c
      - T6a-R3: Test Case item 6a
- G2: High-level requirement 2
  - R2-G2: Requirement 2
    - D2-R2: Design element 2…

Verification of requirements traceability is required for the requirements, design, and test stages of the SDLC.

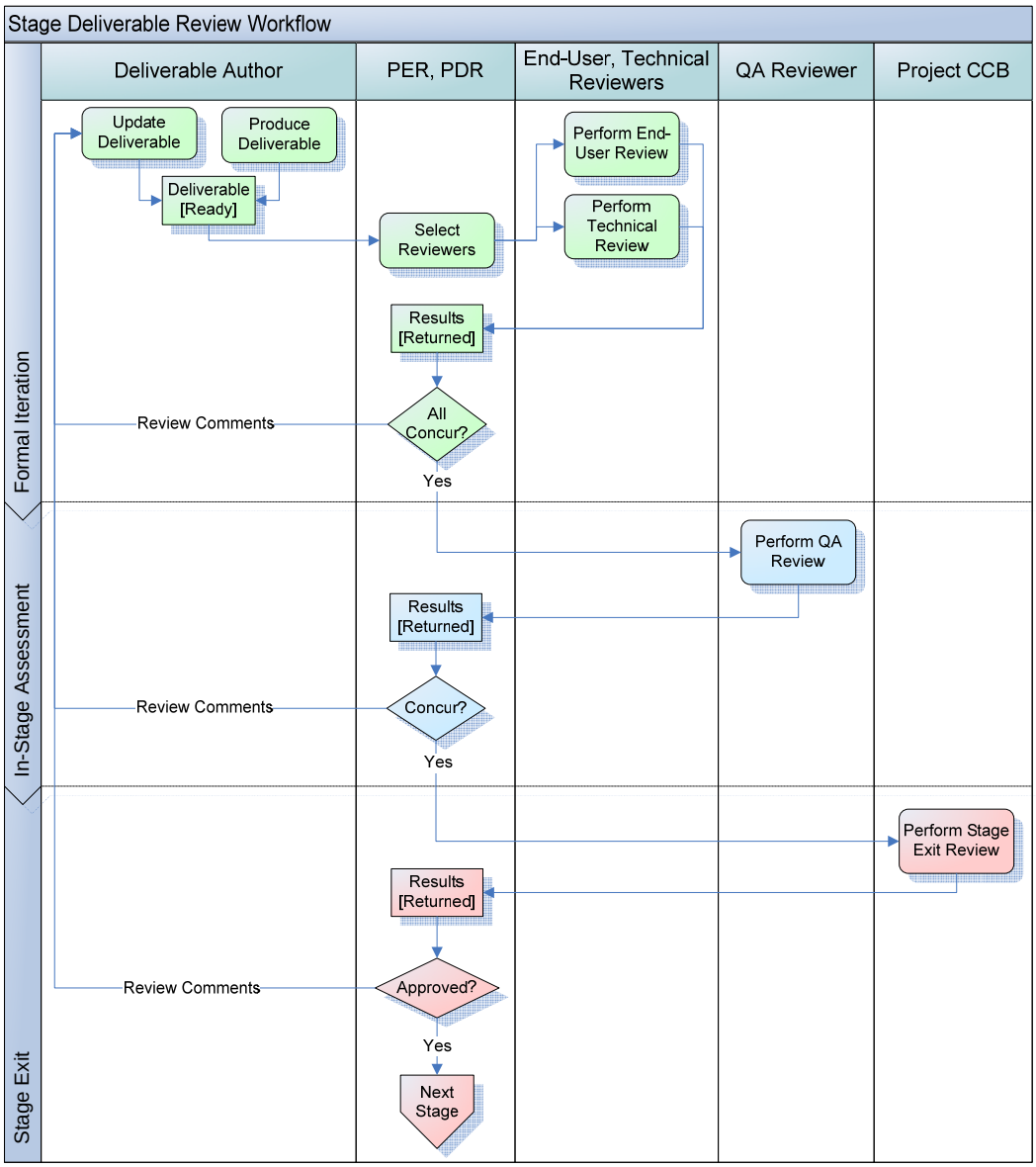
VALIDATION

In validation, the software is tested in a formal, structured manner intended to provide complete coverage of all requirements. This is accomplished through the execution of a series of test cases, each of which is traceable to parent requirements as described above.

The test cases are created during the development stage and incorporated into the project test plan, which is one of the deliverables of the development stage. The test cases include criteria for compliance with all requirements, performance at boundaries, and under stress conditions. Test cases are run against the software during the integration & test stage, where they may be modified before being incorporated into the final acceptance plan. The Software Testing chapter of this document describes this process in greater detail.

STAGE DELIVERABLE REVIEWS

The stage deliverable review workflow manages the review and correction process for SDLC deliverables.



The intent of these reviews are to assure that the established system development and project management processes and procedures are being followed effectively and that deliverable content is both technically correct and satisfactory to the end user community.

ACTORS

The actors associated with stage deliverable reviews include:

1. The author(s) of each deliverable, generally an analyst or developer,
2. The PDR,
3. Subject Matter Experts (SMEs) from the end-user and development communities, acting as reviewers,
4. The quality assurance reviewer, who is typically brought in from outside of the development team, and
5. The members of the project Configuration Control Board.

FORMAL ITERATION PROCESS

As described in the SDLC, the formal iteration process develops the "final" versions of the current stage deliverables, based on the information and documents developed during the previous informal iteration process. End-user and technical reviews are performed to close out the formal iteration process.

END-USER REVIEW

Certain deliverable classes must be reviewed by at least one SME who is familiar with the software product under development. The SME will examine the deliverable for the presence of attributes specific to each class of deliverable, as described in the DCS. The intent of the end-user review is to insure that each deliverable is examined from the point of view of the ultimate users of the system, by someone who is knowledgeable about the process being automated.

TECHNICAL REVIEW

Each class of deliverable must be reviewed by at least one development team member who is familiar with the product under development. This review will be conducted from a technical point of view, with the reviewer examining the deliverable for the presence of attributes specific to each class of deliverable, as described in the DCS. The intent of the technical review is to insure that each deliverable is examined for technical accuracy by someone who is familiar with the processes and development tools for the project. In other development

methodologies, technical reviews may be known as "peer reviews," or "code walk-throughs," depending on the lifecycle stage of the project.

CONCURRENCE

Each reviewer conducts a review of the current stage deliverables for structure and content in compliance with the DCS, using the appropriate review form for each class of stage deliverable. When all reviewers have indicated substantial or unconditional concurrence on the review forms for each deliverable, the PDR notifies all team members that the formal iteration process is closed, pending successful completion of the in-stage assessment.

In the event a technical or end-user reviewer indicates non-concurrence during a review, comments are returned to the deliverable authors, PER, and PDR. The PER and PDR work with the reviewers and document authors to resolve the issue or issues raised, at which point the appropriate reviews are repeated. This cycle continues until concurrence is achieved for a majority of end-user, technical, and QA reviewers.

If the PER and PDR have one or more unconditionally or substantially concurring review forms covering end user, technical, and quality assurance review, they may choose to reject non-concurring reviews and proceed to stage exit. If this is the case, the stage exit review should note the non-concurring reviews and provide responses from the PER and PDR regarding their decision to proceed.

IN-STAGE ASSESSMENT

The Quality Assurance Reviewer (QAR), at the request of the Primary Developer Representative (PDR), conducts in-stage assessments. The review occurs after technical and end-user reviewers concur on the content of the deliverables in the Formal Iteration process of a stage. The QAR reviews stage deliverables for structure and content in accordance with the review criteria for each class of stage deliverable. The review criteria for each class of stage deliverable are incorporated into the relevant DCS, as described in the following chapter.

If the QAR indicates substantial or unconditional concurrence on the appropriate review forms, the in-stage assessment process is considered to be successful. Final document versions are placed under configuration control and posted at the document distribution site. At this point, the PDR notifies the project team that the in-stage assessment was successful via the In-Stage Assessment Report.

If the QAR is unable to concur with one or more deliverable review criteria statements, the PDR works with the QAR and appropriate project personnel to revise the deliverable in accordance with the issues raised. The formal iteration process is then reopened for the affected deliverables and a new set of end-user

and technical reviews are performed. The project iterates between the formal iteration process and the in-stage assessment process until all reviewers indicate substantial or unconditional concurrence on their review forms.

STAGE EXIT REVIEW

The Configuration Control Board (CCB) conducts stage exit reviews. Refer to the Software Configuration Management Plan (SCMP) for further descriptions of the CCB. The purpose of a stage exit is to review the current project plan and stage deliverables, provide a forum to raise issues and concerns, and ensure an acceptable action plan exists for all open issues. Refer to the SDLC for a more complete description of the stage exit process.

If any substantial issues are raised during the stage exit review, the PDR and PER, together with appropriate project team members, work to resolve the issues and modify any effected deliverables. At this point, the project has "backed up" to the formal iteration process. The project then continues forward as previously described.

If no substantial issues are raised, the minutes of the stage exit review are entered as a formal document of record. At this point, the current stage is considered to be successfully concluded.

DELIVERABLE CLASS STANDARDS

One of the most important parts of maintaining high quality output is the establishment of realistic expectations in the minds of the project participants. Those personnel tasked with the production of project deliverables need to have a clear picture of what they are expected to produce. Conversely, when a deliverable is produced, those personnel tasked with reviewing the deliverable need to be working from the same picture. This picture is known as the Deliverable Class Standard (DCS).

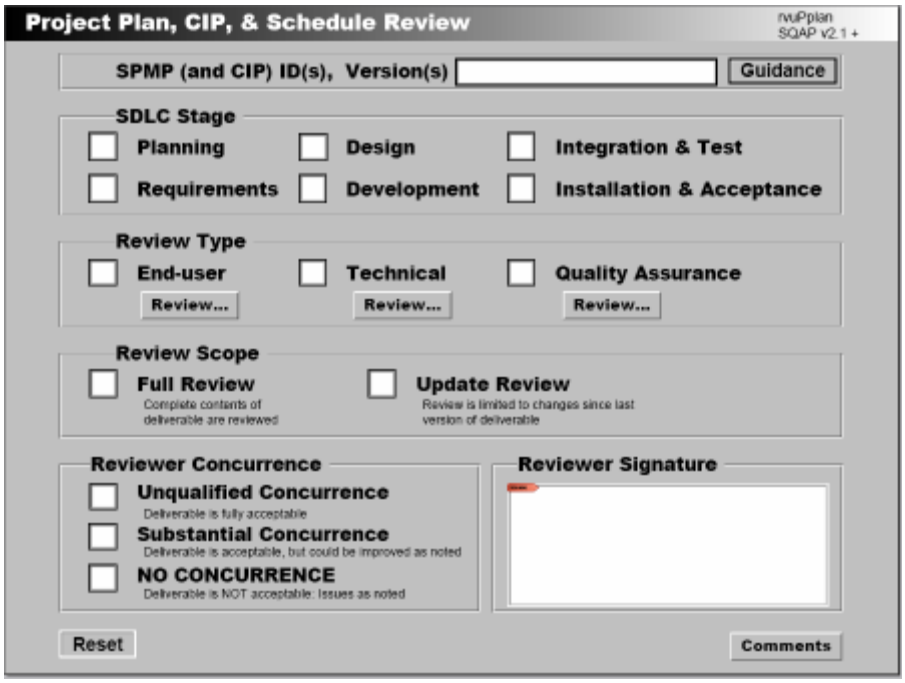
A DCS specifies the structure and content expected for a specific *class* of *deliverable*. A project may have many components, each of which are run through mutually exclusive SDLC iterations. Each SDLC iteration results in a set of deliverables for each stage. For example, a Software Requirements Document (SRD) is a primary deliverable of the requirements stage. Multiple iterations and multiple components result in a series of SRDs with different content. Although the content will differ from one SRD to another, the structure of each SRD is nearly identical. As such, each individual SRD is considered to be a unique instantiation of the SRD *class*.

Each DCS specifies the structure and content requirements for a specific class of deliverables. It describes the purpose of the class, the stages that produce deliverables of that class and the expected structure and content of the deliverable for each stage. The following deliverable class standards are defined in this SQAP:

- Software Project Management Plan DCS
- Requirements Document DCS
- Design Document DCS
- Online Help DCS
- Implementation Map DCS
- Test Plan DCS
- Deployment Plan DCS
- Acceptance Plan DCS
- Installation & Acceptance DCS

REVIEW REQUIREMENTS

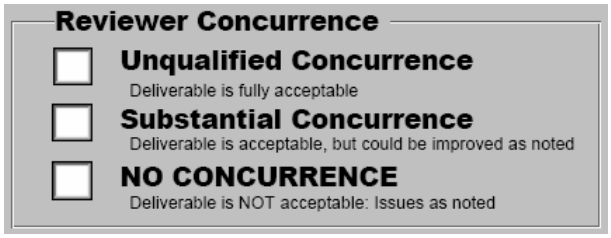
Each class of deliverable is subject to End User review, Technical review and Quality Assurance review. Each reviewer addresses specific issues based on their perspective and area of expertise and indicates their concurrence with the statements in the review form.



Each reviewer will use the review form for the appropriate deliverable class, as identified below. This review form is an electronic form, based on Adobe Acrobat. In general, each reviewer will fill the form out online and append their digital or physical signature on the first page. A digital signature immediately locks the content of the review form against further modification and provides an auditable record of the reviewer identity and review date. A physical signature is intended for use by those projects that are required to print out forms and archive them manually, or use other means of authentication.

CONCURRENCE

Each review form allows the reviewer to indicate their overall level of concurrence with the statements of the review criteria:



A deliverable is considered to have passed review if each reviewer indicates Concurrence on the review form.

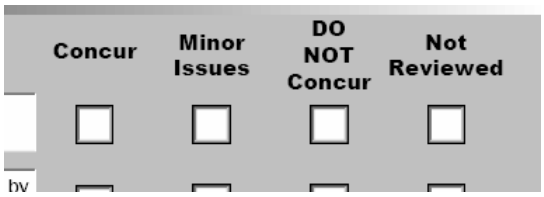
- Unqualified Concurrence means the reviewer has found no problems with any of the stated review criteria and has no additional comments.
- Substantial Concurrence means the reviewer has minor issues with one or more of the review criteria, but does not feel that the issues are sufficient to stop the project and fix them. The reviewer enters additional comments regarding minor issues on the last page of the review form.
- NO CONCURRENCE means the reviewer has found one or more review criteria with which they do not concur, and have entered additional comments on the last page of the review form.

REJECTION AND ACCEPTANCE

A deliverable may be rejected when one or more reviewers indicate NO CONCURRENCE. If the PER and PDR have one or more unconditionally or substantially concurring review forms covering end user, technical, and quality assurance review, they may choose to reject non-concurring reviews and proceed to stage exit. If this is the case, the stage exit review should note the non-concurring reviews and provide responses from the PER and PDR regarding their decision to proceed.

REVIEW CRITERIA CONCURRENCE

Each review form provides four possible answers to the statements of the review criteria:



Concur means the reviewer has found the deliverable to be fully acceptable in all respects, with no formal issues requiring revision of the deliverable. The reviewer feels the deliverable is complete and accurate within their domain of expertise.

Minor Issues means the reviewer has identified one or more minor issues that could, if implemented, result in an improvement of the deliverable, but these issues are not considered important enough to delay the process for another revision of the deliverable. These issues are placed into the review record solely as informal feedback to the author(s). The author of the deliverable may consider these suggestions when revising the deliverable in the future. Minor issues are not formal Test Incident Reports or Change Requests and are not tracked in configuration management systems.

DO NOT Concur means the reviewer has found one or more issues that must be resolved before the reviewer can accept the deliverable. Upon receipt of a non-concurrence result, the issues identified in the review form are submitted to the PDR, as described previously. The development team and involved subject matter experts will work to resolve the issue and release another version of the deliverable for review.

Not Reviewed means the reviewer either chose not to review a specific part of the deliverable or that the identified part is not applicable for the current development effort. The reviewer should clarify the reasoning behind the selection of Not Reviewed in the review comments area. The reviewer has full discretion in acceptance or rejection of the deliverable with one or more items Not Reviewed.

END USER REVIEW

This class of deliverable must be reviewed by at least one SME who is familiar with the software product under development. The SME will examine the deliverable for the presence of attributes identified on the End-user Review page of the review form associated with the deliverable class, as described below.

TECHNICAL REVIEW

This class of deliverable must be reviewed by at least one development team member who is familiar with the product under development. This review will be conducted from a technical point of view, with the reviewer examining the deliverable for the presence of attributes identified on the Technical Review page of the review form associated with the deliverable class, as described below.

QUALITY ASSURANCE REVIEW

The Quality Assurance Reviewer (QAR) will examine the deliverable for the presence of attributes identified on the Quality Assurance Review page of the review form associated with the deliverable class, as described below.

REVIEW FORM FORMAT

The review form for each class consists of a first page, containing topical information, an End-user Review page, a Technical Review page, a Quality Assurance Review page, and a Comments page, as described in each of the class standards below. The End-user Review, Technical Review, and Quality Assurance Review pages are mutually exclusive and are applicable only when the corresponding Review Type is checked on the first page of the review form.

SOFTWARE PROJECT MANAGEMENT PLAN DCS

The Software Project Management Plan (SPMP) class of deliverables are produced during the Planning stage and updated during the Requirements, Design, Development, and Integration & Test stages. The purpose of this class is to accurately reflect the scope, structure, current status and planned activities of the project.

STRUCTURE

The SPMP is composed of four main sections: Vision & Scope, Feasibility & Risk Analysis, Management Approach, and Technical Approach. Although there is some overlap, end user reviews will focus primarily on the first two chapters, while technical reviews will focus on the last two chapters.

CONTENT

The Vision & Scope chapter describes the conditions driving the development of the software product. It provides an overview of the application as initially envisioned and describes the scope and limitations of the development effort. Reviewers should focus on the accuracy of the overview, especially the high-level requirements described in the SPMP and any associated Component Iteration Plans (CIPs).

The Feasibility & Risk Analysis chapter addresses the issues of application complexity as well as the anticipated solution domain, database load and project size. Reviewers should focus on the accuracy of the conditions described as contributing factors to the conclusions described in this chapter.

The Management Approach chapter describes the structure of the project as well as processes for software quality assurance, configuration management, and measurement and analysis. Reviewers should focus on the feasibility of working within the processes described here.

The Technical Approach chapter describes the anticipated development tools, implementation software, documentation delivery formats and formal communications methods for the project. Reviewers should focus on the feasibility of the software implementation as envisioned and whether the anticipated final product will integrate well with the current client infrastructure.

Support items for this document include Use Cases, from which high-level requirements were derived, a high-level project schedule, and a glossary of project-specific terms. Reviewers should focus on the accuracy of the use case

descriptions, the accuracy of the schedule history, and the feasibility of future scheduled activities being completed on time.

REVIEW REQUIREMENTS

The SPMP is subject to End User review, Technical review and Quality Assurance review. Each reviewer addresses specific issues based on their perspective and area of expertise and indicates their level of concurrence with the statements in the review criteria.

SOFTWARE PROJECT MANAGEMENT PLAN REVIEW FORM

The review form allows the standard review form format as described in the previous chapter. Two review forms are available:

- Digital signature review form
- Physical signature review form

Instructions for using Shell Method review forms are available here.

END USER REVIEW

The Subject Matter Expert (SME) will examine the deliverable for the presence of nine attributes:

1. The problem statement and description of business risks are accurate.
   This statement is found in the Vision & Scope: Background section. The problem statement should describe (at a high level) the business problem the software is intended to resolve. Business risks are risks that the sponsoring organization will incur if the software is not developed.

2. The roles and responsibilities of the project team members are identified by reference and acceptable to the end user community.
   This information is summarized in the Vision & Scope chapter, under Principal Project Participants, and is fully described in a referenced PTTQ. You'll need to look over the referenced PTTQ in order to check "Concur."

3. The high-level requirements (goals) in the SPMP and any associated CIPs sufficiently describe all of the desired primary system functions.
   High-level requirements are found in the Vision & Scope: Solution Overview section. These major goals typically describe the general information to be maintained by the software, whether there will be access control, whether there will be ad-hoc reporting, etc. Subsets of these goals are reflected and extended into Component Iteration Plans if the architecture of the documentation stream includes components.

4. The assumptions and dependencies identified as critical success factors are accurate and acceptable to the end user community.

The critical success factors (CSFs) can be found in the Feasibility and Risk Analysis chapter. These represent project risks that can only be mitigated by the executive sponsor, as opposed to applying internal project controls.

5. The project size factors are accurate and acceptable to the end user community.

   Project size and associated sizing factors are described in the Feasibility and Risks chapter, under Project Size. Evaluate the project sizing factors and associated size mitigation items for accuracy. Check "Concur" if the sizing factors and mitigation items are accurate and acceptable to the end user community.

6. The project risk and impact factors are accurate and acceptable to the end user community.

   Project implementation risk, impact, and associated factors are described in the Feasibility and Risks chapter, under Implementation Risk and Operational Impact. Evaluate the associated risk factors and mitigation items for accuracy. Check "Concur" if the risk factors and mitigation items are accurate and acceptable to the end user community.

7. The software development lifecycle for the project is referenced and is acceptable to the end user community.

   The lifecycle is described (at a high level) and referenced in the Management Approach chapter, under Project Structure, and is fully described in a referenced SDLC document. You'll need to look over the referenced SDLC in order to Check "Concur."

8. The software quality assurance process for the project is referenced and is acceptable to the end user community.

   The quality assurance process is described (at a high level) and referenced in the Management Approach chapter, under Software Quality Assurance, and is fully described in a referenced SQAP document. You'll need to look over the referenced SQAP in order to Check "Concur."

9. The software configuration management process for the project is referenced and is acceptable to the end user community.

   The configuration management process is described (at a high level) and referenced in the Management Approach chapter, under Software Configuration Management, and is fully described in a referenced SCMP document. You'll need to look over the referenced SQAP in order to Check "Concur."

TECHNICAL REVIEW

The technical reviewer will examine the deliverable for the presence of eight attributes:

1. The high-level requirements (goals) in the SPMP and any associated CIPs describe all necessary major functions for a database effort.

   The major database functions are found in the Vision & Scope: Solution Overview section. Subsets of these goals are reflected and extended into Component Iteration Plans if the architecture of the documentation stream

includes components. Note if any high-level functionality is missing or inaccurate.

2. The elements selected in the project sizing matrix are correct.
   The project sizing matrix is found in the Feasibility & Risk: Project Size section. Check "Concur" if the highlighted size factors and the resulting size definition and mitigation items are accurate.

3. The elements selected in the risk analysis matrix are correct.
   The risk analysis matrix is found in the Feasibility & Risk: Project Risk and Operational Impact sections. Check "Concur" if the highlighted risk factors, the resulting risk definition, and the associated mitigation items are accurate.

4. The anticipated development tools and production infrastructure are described adequately for the current project stage.
   This description is found in the Technical Approach: Anticipated Implementation Architecture section. Check "Concur" if the anticipated tools and the description for the production infrastructure accurately reflect what is expected for the project.

5. The project effort and schedule estimates for the next stage and out stages are present and adequately detailed.
   The project schedule is referenced from the Management Approach: Project Structure section. Check "Concur" if the project effort and schedule are appropriate for this project.

6. The project effort and schedule metrics for the current stage are present and adequately detailed.
   Project metrics are referenced in the Management Approach: Project Structure section. Check "Concur" if the project effort and scheduling metrics are appropriate for this project.

7. (Design Stage review only) The selected development tools and production infrastructure are suitable for this project.
   Development and production infrastructure tools are found in the Technical Approach: Anticipated Implementation Architecture section. Check "Concur" if the development and production infrastructure tools are appropriate for this project.

8. (Development Stage review only) The product testing process provides adequate coverage of software artifacts.
   Testing processes for the project are found in the SQAP, and are referenced in the Management Approach: Project Structure section. Check "Concur" if the descriptions of these processes are appropriate for the project.

QUALITY ASSURANCE REVIEW

The independent Quality Assurance Reviewer (QAR) will examine the deliverable for the presence of three attributes:

1. One or more End-user reviews for this deliverable have been completed.
   Check "Concur" if one or more end-user reviews have been completed.

2. One or more Technical reviews for this deliverable have been completed.
   Check "Concur" if one or more technical reviews have been completed.

3. All End-user and Technical reviews indicate substantial or unconditional concurrence.
   Check "Concur" if there is substantial or unconditional concurrence for all end-user and technical reviews.

REQUIREMENTS DCS

The Requirements class of deliverables are produced during the Requirements stage and updated if necessary during the Design, Development, and Integration & Test stages. The purpose of the Requirements class is to accurately define the scope, structure, and high-level functionality of the database application under design.

STRUCTURE

The Requirements class of deliverables is composed of three related documents:

- The Logical Database Description
- The Requirements Document
- The Requirements Traceability Matrix

CONTENT

LOGICAL DATABASE DESCRIPTION (LDD)

The LDD defines the basic structure of the application at a conceptual level. The LDD focuses on high-level data storage areas, known as entities, the actors that interact with these entities, and quantitative metrics about each entity.

The LDD consists of an introduction, a Logical Entity Relationship Diagram (Logical ERD), and a series of entity descriptions that define the relationships between the entities, actor interactions with the entities, and metrics.

The LDD is included by reference in the Requirements Document.

SOFTWARE REQUIREMENTS DOCUMENT (SRD)

The SRD refers to and expands upon the LDD by defining a set of functional requirements that are specific to each entity described in the LDD. These functions may include component selection, summary listing, data entry & detail display, simple searches, predefined complex searches, predefined reports, and operations.

The final section of the SRD is an Analysis Listing, which is used for verification of requirements traceability and project sizing.

REQUIREMENTS TRACEABILITY MATRIX (RTM)

The RTM makes use of the analysis listings in the SRD and its parent SPMP or Component Iteration Plan (CIP) document. The purpose of the RTM is to show

that each requirement is related to a specific goal in the SPMP or CIP, that all goals in the project plan have at least one associated requirement, and that no requirements in the SRD are related to non-existent goals.

REVIEW REQUIREMENTS

The Requirements class is subject to End User review, Technical review and Quality Assurance review. Each reviewer addresses specific issues based on their perspective and area of expertise and indicates their level of concurrence with the statements in the review form.

REQUIREMENTS REVIEW FORM

The review form allows the standard review form format as described in the previous chapter. Two review forms are available:

- Digital signature review form
- Physical signature review form

Instructions for using Shell Method review forms are available here.

END USER REVIEW

The Subject Matter Expert (SME) will examine the deliverable for the presence of seven attributes:

1. The application overview statement is accurate.
   This statement is found in the Introduction chapter and should be similar to the Vision Statement from the Project Plan document.

2. The design constraints identified are acceptable to the end user community.
   The design constraints can be found at the end of the Introduction chapter, just before References. Check "Concur" if the system won't need to impose any constraints beyond those described.

3. The Logical Database Description accurately represents the data entities for this application.
   The LDD basically describes the major types of data your system will be managing. If something has been missed, this should be noted.

4. The functionality of each data area is adequately identified.
   Typically, each data area of your system has a summary listing, a data entry form, and possibly some predefined searches, reports, or operations. Note if statements are inaccurate or anything has been missed.

5. The access control requirements are acceptable to the end user community.

   Access control requirements are found in the LDD under "Actor Interactions" for each data entity chapter. Note whether these statements accurately define requirements appropriate to the application.

6. The interface requirements are acceptable to the end user community.

   The user interface is defined in the Requirements document, under the first requirement for each "Manage xxx Area" section. External interfaces are data exchange mechanisms that connect to other computer systems, such as exporting data to Excel. External interfaces, if any, are generally described under the Operations header of each data area description.

7. There are no known issues or additional requirements not covered by this document.

   From this point on, the design and development efforts will focus on expanding what has been described in the Requirements documents. Adding new searches, reports, and operations to the current iteration will be difficult when the system has already been scoped and scheduled.

TECHNICAL REVIEW

The technical reviewer will examine the deliverable for the presence of seven attributes:

1. The performance design constraints fall within the normal capabilities of the target database.

   The requirements document references the standard design constraints document and may include additional, project-specific constraints. Evaluate the known ability of the target database engine and support hardware to perform within these constraints. Check "Concur" if you conclude that the described database configuration will meet these constraints.

2. The access control requirements fall within the normal capabilities of the target database.

   Note any special requirements beyond the standard "users and groups" model, especially if there is any need for row-level access restrictions. Check "Concur" if you determine the anticipated database engine configuration will be able to meet these requirements.

3. The interface requirements fall within the normal capabilities of the target database.

   Look for a user interface that is appropriate to the anticipated development environment and end-user client. Check for interfaces to other systems and evaluate the ability of the respective systems to communicate adequately. Check "Concur" if you conclude that the defined interfaces can be implemented within the constraints of the system.

4. All requirements are complete and useful for development of design criteria.

Look for "blurry" requirements that could result in a single design element pointing to two different requirements. Check "Concur" if the requirements are clear enough that a set of design elements can be developed from a single requirement.

5. All requirements are technically feasible.
   Check "Concur" if the requirements can be implemented using current technology.

6. There are no inconsistent or conflicting requirements.
   Check "Concur" if all requirements are clear, distinct, and don't step on each other's toes.

7. (Post-Requirements Stage review only) All requirements can be verified by inspection or formal testing of related design elements.
   This is a sanity check for those times when the SRD has been updated during a later lifecycle stage. Check "Concur" if all requirements are, or can be tied to easily testable design elements.

QUALITY ASSURANCE REVIEW

The independent Quality Assurance Reviewer (QAR) will examine the deliverable for the presence of four attributes:

1. One or more End-user reviews for this deliverable have been completed.
   Check “Concur” if one or more end-user reviews have been completed.

2. One or more Technical reviews for this deliverable have been completed.
   Check “Concur” if one or more technical reviews have been completed.

3. All End-user and Technical reviews indicate substantial or unconditional concurrence.
   Check “Concur” if there is substantial or unconditional concurrence for all end-user and technical reviews.

4. A Requirements Traceability Matrix Report indicates the traceability matrix is valid for the SPMP or CIP and Requirements Documents.
   The traceability matrix is posted on the project Web site. The requirements document can point to either the main SPMP if the project is not divided into components, or to the appropriate Component Iteration Plan. Check “Concur” if the matrix is present and correct.

DESIGN DCS

The Design class of deliverables are produced during the Design stage and updated if necessary during the Development and Integration & Test stages. The purpose of the Design class is to accurately define the scope, structure, and high-level functionality of the database application under design.

STRUCTURE

The Design class of deliverables is composed of three related documents:

- The Physical Database Description
- The Software Design Document
- The Requirements Traceability Report

CONTENT

PHYSICAL DATABASE DESCRIPTION (PDD)

The physical database description defines the basic structure of the application at a conceptual level. The PDD focuses on providing a detailed description of the database structure to be implemented for the application.

The PDD consists of an introduction, an Entity Relationship Diagram (ERD) and a series of table and field descriptions that define the relationships between the entities, field characteristics, and business rules.

The PDD is included by reference in the Design Document.

SOFTWARE DESIGN DOCUMENT (SDD)

The design document refers to and expands upon the PDD by defining a set of design elements that are specific to each data area described in the associated requirements document

The SDD defines a series of forms, methods, and access control mechanisms to be implemented for each data area described in the current requirements document. These functions include module selection, summary listing forms, data entry & detail forms, simple searches, predefined complex searches, predefined reports, and operations.

REQUIREMENTS TRACEABILITY MATRIX (RTM)

The RTM makes use of the analysis listings in the SDD and its parent SRD. The purpose of the RTM is to show that each design element is related to a specific requirement in the SRD, that all goals in the project plan have at least one associated requirement, and that no requirements in the SRD are related to non-existent goals.

REVIEW REQUIREMENTS

The Design class is subject to End User review, Technical review and Quality Assurance review. Each reviewer addresses specific issues based on their perspective and area of expertise and indicates their level of concurrence with the statements in the review form.

DESIGN REVIEW FORM

The review form allows the standard review form format as described in the previous chapter. Two review forms are available:

- Digital signature review form
- Physical signature review form

Instructions for using Shell Method review forms are available here.

END USER REVIEW

The Subject Matter Expert (SME) will examine the deliverable for the presence of nine attributes:

1. The table structure shown in the Physical Database Description (PDD) accurately represents the desired storage structure.

   The technical review takes care of making sure that the tables are set up properly for a relational database. What the SME should focus on is the information that is supposed to be stored in the database tables and how they relate to each other. Note if there is anything missing or wrong. Check "Concur" if all the main tables needed by the user community are accurately described.

2. The field definitions shown in the PDD accurately represent the fields needed in this application.

   The SME should make sure that all the information needed is accurately represented. NOTE: some fields may be there for internal use they the software. Check with the systems analyst if anything looks unfamiliar. Check "Concur" if all the fields are correctly described.

3. The core application design elements accurately represent the common functions expected to be available for each data area.

Core functions are often defined as a separate component, but may be included in a single design document for a small application that has not been subdivided into components. Check "Not Reviewed" if the core functions are in a separate component that is not part of this review. Check "Concur" if core functions are a part of this review and appear to be accurate.

A data area is a major component of a system and is a collection of related tables designed to support a specific process, like container preparation or visual examination. The data area entry point is generally the first design element following the data area heading. Usually, a data area is entered via a menu item or hyperlink from a higher level page. Check "Concur" if the data area design elements accurately describe the functionality needed for each data area.

4. The access control permissions for each data entry area are accurately described.

   This information is located in the Access Control Matrix in the Physical Database Description. Make sure all the appropriate types of user are represented, along with all the tables and who can do what to each table. Check "Concur" if the access control matrix is accurate.

5. The summary listing screens for each data area are accurately described.

   These are the screens that show up in response to queries in each data area. The intent of a summary screen is to provide enough information to quickly find the record or records that are needed. As a result, only important fields are shown (for performance reasons). NOTE: Some applications do not support or require summary listing screens. Check "Not Reviewed" if summary lists are not part of this application. Check "Concur" if the summary lists are showing the correct subset of fields for each data area.

6. The data entry & detail display screen layouts and business rules for each data area are accurately described.

   The data entry screen is the heart of each data area. Here, all the fields needed should be available and the rules imposed by the screen are described. Check with the system analyst if anything is missing or does not make sense. Check "Concur" if the data entry and detail screens are accurately represented in the design document.

7. The simple, pre-defined, and ad hoc searches for each data area are accurately described.

   The simple and ad hoc searches are the same across the system, and may be described in the Core Functions, which may or may not be part of the current review. Note if the descriptions for the pre-defined searches are not accurate or not complete. Check "Concur" if the descriptions are accurate and complete. Check "Not Reviewed" if no searches are needed for any data areas.

8. The pre-defined reports for each data area are accurately described.

   Ensure that any pre-defined reports are correct, both in terms of mockup/screen shot appearance as well as business rules used to summarize or consolidate information. Check "Concur" if the descriptions are accurate and complete. Check "Not Reviewed" if there are no pre-defined reports needed for any data areas.

9. The pre-defined operations for each data area are accurately described.
   If pre-defined operations (also known as batch operations) are present. Check "Concur" if their descriptions are accurate and complete. Check "Not Reviewed" if there are no operations needed for any data areas.

TECHNICAL REVIEW

NOTE: The term “sufficient specification” in the following review criteria has a specific meaning:

> The objective of a design document is to describe the functionality of the application in normal language as opposed to "code-speak." The end-user community needs to be able to understand the design well enough to sign off on it. Developers are fully capable of taking a plain language design element and implementing it in code.
>
> Over-specifying design elements (describing functions and method calls, detailed flow of control descriptions, detailed descriptions of algorithms and variables in use) is counter-productive. Over-specified design elements are confusing to most end-users, leading to poorer understanding, poor review quality, and increased design defects. Sufficiently specified design elements have the following general characteristics:
>
> 1. User interface design elements describe the user interface using either mock-ups or screen shots of prototype work.
> 2. Business rules and functionality are described at a level that is comprehensible to end users and in a manner that allows the developer flexibility in implementation.
> 3. Design elements should not include direct code calls or detailed control flow definitions.
> 4. Tab order on forms is assumed to be left-to-right, top-to-bottom, either across the entire page or within defined blocks, and does not need to be formally specified. Only exceptions need to be described.
> 5. Normal form characteristics like scroll bars, combo boxes, and the behavior of fields (limiting text entry to field length, requiring numbers in numeric fields) are normal and expected features and do not need to be formally described as design elements.

The technical reviewer will examine the deliverable for the presence of nine attributes:

1. The table structure described in the Physical Database Description (PDD) is relationally compliant to at least the Third Normal Form.

Some databases require normalization beyond the third normal form, so this is a minimum criteria. Portions of the database may have been de-normalized to optimize performance. De-normalization is acceptable when it is limited to a small set of attributes and is intentional on the part of the developer. Check "Concur" if the database structure is properly normalized.

2. The PDD accurately describes all fields, their default values, formats, range limitations, "Tip Help", and business rules for the application.

   Some application development environments don't support tip help or database triggers. Some development environments require range limits, format masks, and business rules to be driven into the form designs. In most cases it is best to define all of these in the PDD and define specific business rules for form-based validation and use assistance in the design document. Discuss these issues with the database developer. Check "Concur" if you find consistent placements for these definitions either in the PDD or in the design document.

3. The access control mechanisms described in the design document are compatible with the default capabilities of the target development tools.

   Normal access control requirements include the users and groups/roles concept, restricting access at the table level. Access control restrictions that extend this model beyond the defaults will need a specific set of requirements and design elements. Check "Concur" if the default access controls of the database engine are utilized OR if there is a specific set of design elements for a more extensive model.

4. The design elements for the core application functions are specified at a level sufficient for development to begin.

   Core functions are often defined as a separate component, but may be included in a single design document for a small application that has not been subdivided into components. Check "Not Reviewed" if the core functions are in a separate component that is not part of this review. Check "Concur" if core functions are a part of this review and sufficiently specified as defined above.

5. The summary listing forms and associated business rules for each data area are specified at a level sufficient for development to begin.

   Summary listing forms are a common implementation in rich client database applications. Listings are less common in enterprise-level applications and in Web-based applications. Check "Not Reviewed" if summary listing forms are not part of the application under review. Check "Concur" if summary listings are a part of this review and sufficiently specified as defined above.

6. The data entry & detail display forms and associated business rules for each data area are specified at a level sufficient to begin development.

   Data entry and detail display forms are typically documented as follows:

   a. *Entry criteria, or what action the user takes to open the form.*
   b. *A mockup or screen shot of the form.*
   c. *Preconditions, which are rules that must be satisfied before the form is allowed to be displayed. Access control restrictions are typically described here.*

d. *Initialization Rules, which are rules that "set up" the form for use. These may include populating popups, setting defaults, and initializing primary key values.*
e. *Interaction Rules, which are rules that describe how the form elements respond to user interactions. "When the user pokes this widget, it responds this way."*
f. *Conclusion Rules, which are rules that happen when the form is exited, either via Submit/OK or Back/Cancel. These typically include saving various records, triggering notifications, and committing/cancelling transactions.*

Check "Concur" if the data entry and detail display forms are sufficiently specified.

7. The simple, pre-defined, and ad hoc searches for each data area are specified at a level sufficient for development to begin.

   Ad-hoc searches may or may not be supported by the application. If the application does not support ad-hoc searches, then the ad-hoc portion of this review criteria does not apply. Simple and ad-hoc searches may also be described in the Core Functions design elements, which may or may not be part of the current review. Pre-defined searches may optionally include the presentation of selectable and/or enterable query conditions prior to execution of the query. Check "Concur" if all applicable elements are sufficiently specified. Check "Not Reviewed" if no searches are needed for any data area.

8. The pre-defined reports for each data area are specified at a level sufficient for development to begin.

   Pre-defined reports may have optional parameters dialogs in addition to their mockups/screen shots and associated business rules for data summarization, transformation, and/or consolidation. Check "Concur" if all applicable elements are sufficiently specified. Check "Not Reviewed" if there are no pre-defined reports needed for any data area.

9. The pre-defined operations for each data area are specified at a level sufficient for development to begin.

   Pre-defined operations may have optional parameters dialogs in addition to the business rules describing what they need to do. Check "Concur" if all applicable elements are sufficiently specified. Check "Not Reviewed" if there are no operations needed for any data area.

QUALITY ASSURANCE REVIEW

The independent Quality Assurance Reviewer (QAR) will examine the deliverable for the presence of six attributes:

1. A design constraints compliance statement is included in the design document.

   The design constrainsts compliance statement can be found in the Introduction. Check “Concur” if this statement is present.

2. A physical database description is made available and included directly or by reference in the design document.

   The physical database description is in the Global Design Elements chapter. Check "Concur" if this description and a valid link to the Physical Database Description document is present.

3. Specific sets of design elements are in place for each data area identified in the design document.

   Data areas are generally described in a fairly consistent manner (entry rule, screen shot, preconditions, initialization rules, interaction rules, and conclusion rules). Check "Concur" if each data area is appropriately described.

4. One or more End-user reviews for this deliverable have been completed.

   Check "Concur" if one or more end-user reviews have been completed.

5. One or more Technical reviews for this deliverable have been completed.

   Check "Concur" if one or more technical reviews have been completed.

6. All End-user and Technical reviews indicate substantial or unconditional concurrence.

   Check "Concur" if there is substantial or unconditional concurrence for both the end-user and technical reviewers.

7. A Requirements Traceability Report indicates the traceability matrix is valid for the SPMP/CIP, Requirements, and Design documents.

   The traceability matrix is posted on the project Web site. Check "Concur" if the matrix is present and correct.

ONLINE HELP DCS

The Online Help is produced during the Development stage and updated if necessary during the Integration & Test stage. The purpose of the Online Help class is to provide references and guidance to the end user concerning the operation of the application or component.

STRUCTURE

The Online Help class of deliverable is a composed of a set of web page files organized into an HTML-based Online Help system.

CONTENT

The HTML-based Online Help system is composed of three main topics:

- Data Areas
- Optional Scenarios
- Data Dictionary

The Data Areas topics describe the features and functionality specific to each data area. Optionally, the help system may include usage scenarios, which are typically represented as “How do I…” topics. The Data Dictionary describes the structure built into the database engine as well as applicable access restrictions.

REVIEW REQUIREMENTS

The Online Help class is subject to End User review, Technical review and Quality Assurance review. Each reviewer addresses specific issues based on their perspective and area of expertise and indicates their level of concurrence with the statements in the review form.

ONLINE HELP REVIEW FORM

The review form allows the standard review form format as described in the previous chapter. Two review forms are available:

- Digital signature review form
- Physical signature review form

Instructions for using Shell Method review forms are available here.

END USER REVIEW

The Subject Matter Expert (SME) will examine the deliverable for the presence of four attributes:

1. The online help contains an application overview that describes the main objectives of the software.
   This is a general set of text intended to acquaint the reader with the main goals of the application or component. The overview should be a re-statement of the goals and design constraints from the parent requirements document. Check "Concur" if the overview is present and appears to be accurate.

2. The online help adequately describes expected user interactions with the software.
   This is typically done in two ways:
   i. *Data Area full descriptions, which include screen shots and descriptions of the business rules associated with each form.*
   ii. *Optional scenarios, or descriptions of common tasks and activities. These are sometimes presented as "How do I..." links.*

   Check "Concur" if the data areas for the application or component are adequately represented.

3. The online help adequately covers all operational features of the software at a level sufficient for successful use of the software.
   All features of the application or component as described in the design document should be present. The business rules associated with each form should be available to the reader. Check "Concur" if the help system provides adequate coverage.

4. The online help contains a data dictionary with an entity-relationship diagram and complete descriptions of the tables and fields in the database.
   At a minimum, there should be a separate link to the Physical Data Description document for the application or component. Optionally, an integrated listing of tables and associated fields can be made available. Check "Concur" if a dictionary listing or link to the PDD is available.

TECHNICAL REVIEW

The development team member will examine the deliverable for the presence of three attributes:

1. The online help is made available to end-users as a standard Web URL.
   This allows direct access to the help system from either a separate Web browser or from links provided within the application. Check “Concur” if this link is available and functions appropriately.

2. The online help contains an accurate description of common functions and application-specific functions.
   If there is a separate Core Functions component, make sure that the operational component works properly within the core flow of control. If the core functions

are built into the same design set, the testing process should have taken care of this. Check "Concur" if the core functions are accurately represented.

3. The online help contains an accurate data dictionary.
   Verify the contents of the data dictionary. Check "Concur" if the help dictionary matches the data structures as described in the PDD.

QUALITY ASSURANCE REVIEW

The independent Quality Assurance Reviewer (QAR) will examine the deliverable for the presence of three attributes:

1. One or more end-user reviews for this deliverable have been completed.
   Check "Concur" if one or more end-user reviews have been completed.

2. One or more technical reviews for this deliverable have been completed.
   Check "Concur" if one or more technical reviews have been completed.

3. All end-user and technical reviews indicate substantial or unconditional concurrence.
   Check "Concur" if there is substantial or unconditional concurrence for both the end-user and technical reviewers.

IMPLEMENTATION MAP DCS

The Implementation Map is produced during the Development stage and updated if necessary during the Integration & Test stage. The purpose of the Implementation Map is to tie the software design elements to the application source code.

STRUCTURE

The Implementation Map class of deliverables is composed of two documents:

- The Implementation Map
- The Requirements Traceability Matrix

CONTENT

IMPLEMENTATION MAP

The intent of the Implementation map is to provide a quick navigation method from a specific application design element to the corresponding part of the source code. The Implementation Map is not intended to exhaustively document the structure of the source code itself; that is a function that is best left to automated tools.

For any specific design element, the Implementation Map identifies the associated "root node" in the source code. A root node is the top-level source that initiates the specified functionality. A root node may call on other source entities and objects and may itself be called by other parts of the source code.

The intent of identifying the root node is to allow a programmer with moderate experience in the programming language to quickly locate and begin work on a specific feature of the application, tracing the code from the root node as necessary to add enhancements or correct errors.

REQUIREMENTS TRACEABILITY MATRIX (RTM)

The RTM makes use of the design elements list in the IMP and its parent SDD. The purpose of the RTM is to show that each root node is related to a specific design element in the SDD.

REVIEW REQUIREMENTS

The Implementation Map class is subject to Technical review and Quality Assurance review, but *not* End User Review. Each reviewer addresses specific

issues based on their perspective and area of expertise and indicates their level of concurrence with the statements in the review form.

IMPLEMENTATION MAP REVIEW FORM

The review form allows the standard review form format as described in the previous chapter. Two review forms are available:

- Digital signature review form
- Physical signature review form

Instructions for using Shell Method review forms are available here.

TECHNICAL REVIEW

The technical reviewer will examine the deliverable for the presence of three attributes:

1. The root nodes identified in the Implementation Map are present in the source code.

   Compare the root node list elements with the application object tree in the development environment. Not all objects are root nodes, but all root nodes should be objects or setup scripts. Check "Concur" if all root nodes have matching elements in the object tree or application build scripts.

2. The root nodes in the source code are functional parents. They have no parent nodes calling them that also trace to the same function.

   The objective here is to make sure that the root nodes are actually useful for debugging (which is the whole purpose of an IMP). When a defect is reported, the analysis phase of the change control process identifies the SDD and the specific design element or elements that are associated with the defect. When a developer is assigned to fix the defect, the IMP is used to locate the root node(s) associated with the identified design element(s). If these root nodes don't point to the right places to start troubleshooting (they're not functional parents), then it's worse than not providing any guidance to the developer at all.

   Your job as technical reviewer is to make sure these root nodes are correct. For example (depending on the development tool), the root node for a for a form widget could be a portion of the overall page script, or could be an independently called method. A root node for a complex report would be the initial script, package, procedure or method that presents the parameters dialog and transfers those parameters to the associated report definition. Check "Concur" if all root nodes are functional parents.

3. A reference to the source code listing for the software has been included in the Implementation Map.

   This is typically a statement pointing the reader to the source code configuration management tool. Check "Concur" if the statement is accurate enough for a new developer to locate the code.

QUALITY ASSURANCE REVIEW

The independent Quality Assurance Reviewer (QAR) will examine the deliverable for the presence of three attributes:

1. One or more Technical reviews for this deliverable have been completed.
   Check "Concur" if one or more technical reviews have been completed.

2. The Technical reviews indicate substantial or unconditional concurrence.
   Check "Concur" if there is substantial or unconditional concurrence from the technical reviewer(s).

3. A Requirements Traceability Report indicates the traceability matrix is valid for the Implementation Map and Design Document.
   The traceability matrix is posted on the project Web site. Check "Concur" if the matrix is present and correct.

TEST PLAN DCS

The Software Test Plan (STP) is produced during the Development stage and updated if necessary during the Integration & Test stage. The purpose of the Test Plan class is to document the test procedures, test cases, and test steps required to validate the development effort.

STRUCTURE

The Test Plan class of deliverables is composed of two related documents:

- The Test Plan
- The Requirements Traceability Report

CONTENT

SOFTWARE TEST PLAN

The Test Plan defines two specific test procedures:

- Acceptance Test procedure
- Regression Test procedure

Each test procedure utilizes a specific series of test cases drawn from a global pool of test cases. Each test case focuses on a specific feature set of the application.

REQUIREMENTS TRACEABILITY MATRIX (RTM)

The RTM makes use of the analysis listings in the STP and its parent SDD. The purpose of the RTM is to show that each test case element is related to a specific design element.

REVIEW REQUIREMENTS

The Test Plan class is subject to Technical review and Quality Assurance review, but *not* End User Review. Each reviewer addresses specific issues based on their perspective and area of expertise and indicates their level of concurrence with the statements in the review form.

TEST PLAN REVIEW FORM

The review form allows the standard review form format as described in the previous chapter. Two review forms are available:

- Digital signature review form
- Physical signature review form

Instructions for using Shell Method review forms are available here.

TECHNICAL REVIEW

The technical reviewer will examine the deliverable for the presence of two attributes:

1. The test cases demonstrate that the code adequately performs all intended functions, produces valid results, and meets the design criteria.
   The objective here is to make sure that the test cases cover all the design elements in the parent design document. Check "Concur" if there is a test case for each of the defined functions for the system and each test case allows the tester to adequately check that a result returned from a test is a valid result for the given function. Look for wording that came from the design document that isn't clearly testable, and was not modified in the test plan to create a workable test element. Check "Concur" if the test plan and any referenced test scripts provide adequate coverage.

2. All test criteria are stated in measurable terms.
   Check "Concur" if each of the test criteria listed can be quantified.

QUALITY ASSURANCE REVIEW

The independent Quality Assurance Reviewer (QAR) will examine the deliverable for the presence of three attributes:

1. One or more technical reviews for this deliverable have been completed.
   Check "Concur" if one or more technical reviews have been completed.

2. The technical review(s) indicate substantial or unconditional concurrence.
   Check "Concur" if there is substantial or unconditional concurrence from the technical reviewer(s).

3. A Requirements Traceability Report indicates the traceability matrix is valid for the Test Plan and Design Document.
   The traceability matrix is posted on the project Web site. Check "Concur" if the matrix is present and correct.

DEPLOYMENT PLAN DCS

The Deployment Plan is produced during the Integration & Test stage. The purpose of the Deployment Plan is to define the hardware, software, configuration and test cases for the test and production environments for the combined set of application components.

STRUCTURE

The Deployment Plan class of deliverables is a composed of a single document:

- The Deployment Plan

CONTENT

The Deployment Plan contains:

- Specifications for the test and production environments.
- Regression test cases for all incorporated components.
- User roles for the test and production database systems.
- Production users and associated roles for the production system.

REVIEW REQUIREMENTS

The Deployment Plan class is subject to Technical review and Quality Assurance review, but *not* End User Review. Each reviewer addresses specific issues based on their perspective and area of expertise and indicates their level of concurrence with the statements in the review form.

DEPLOYMENT PLAN REVIEW FORM

The review form allows the standard review form format as described in the previous chapter. Two review forms are available:

- Digital signature review form
- Physical signature review form

Instructions for using Shell Method review forms are available here.

TECHNICAL REVIEW

The technical reviewer will examine the deliverable for the presence of three attributes:

1. The configurations of the test and production environments are accurately described.

   The configurations for the test and production environments should be described in sufficient detail that a technically capable person can build the environments from scratch if necessary. For organizations that control their property items, the property identifiers of the systems should be included. Host names and IP addresses should also be included. The intent is to allow both pre-production setup and post-production verification of the hardware and support software. Check "Concur" if the environments are described in sufficient detail that they can be executed without deep expertise in the design of the system.

2. The data loading / migration sources and/or processes are accurately described.

   There are two sets of data loading/migration activities described in the deployment plan. One for the test environment, and one for the production environment.

   i. *Evaluate the test environment data sets for adequate coverage across all tables for all components, especially for data volume as described in the associated component LDD metrics.*
   ii. *Evaluate the production environment data for accuracy. Newly installed components will need certain reference tables loaded, while updated components may need to have their production data modified to handle any structural differences.*
   iii. *Check "Concur" if the test and production environment data loading and/or migration processes are described in sufficient detail that they can be executed without deep expertise in the design of the system.*

3. The user roles and production users are accurately listed.

   Each component PDD describes a set of user roles. These roles need to be listed in the user roles setup listings for each component in the test and production environments. For the production environment, a complete listing of authorized users, combining existing users and new users for new components. For existing users, any new roles need to be properly associated. Although this listing will probably be out-of-date immediately after the system is accepted, it provides a starting point that supports the existing user base. Check "Concur" if the user roles and production users are described in sufficient detail that they can be implemented by a DBA without deep expertise in the design of the system.

QUALITY ASSURANCE REVIEW

The independent Quality Assurance Reviewer (QAR) will examine the deliverable for the presence of two attributes:

1. One or more technical reviews for this deliverable have been completed.

   Check “Concur” if one or more technical reviews have been completed.

2. The Technical review(s) indicate substantial or unconditional concurrence.

   Check “Concur” if there is substantial or unconditional concurrence from the technical reviewer(s).

ACCEPTANCE PLAN DCS

The Acceptance Plan is produced during the Development stage and updated if necessary during the Integration & Test stage. The purpose of the Acceptance Plan class is to define the set of acceptance tests to be performed in order for the customer to accept the system into production.

STRUCTURE

The Acceptance Plan class of deliverables is a composed of a single document:

- The Acceptance Plan

CONTENT

The Acceptance Plan contains a set of acceptance test cases for each component in the target production system.

REVIEW REQUIREMENTS

The Acceptance Plan class is subject to Technical review and Quality Assurance review, but *not* End User Review. Each reviewer addresses specific issues based on their perspective and area of expertise and indicates their level of concurrence with the statements in the review form.

ACCEPTANCE PLAN REVIEW FORM

The review form allows the standard review form format as described in the previous chapter. Two review forms are available:

- Digital signature review form
- Physical signature review form

Instructions for using Shell Method review forms are available here.

TECHNICAL REVIEW

The technical reviewer will examine the deliverable for the presence of two attributes:

1. All production components have associated acceptance test cases.
   The acceptance plan covers the entire application, including all components that may have been undergoing separate development iterations. Check

"Concur" if all components of the application are represented with associated test cases.

2. The test cases adequately cover the high-level functionality of the application.

   The acceptance testing process does not test every feature of the application. That level of testing occurred previously, during the regression test. The intent of the acceptance testing process is to show that each of the application components have been placed into production. If each component can be accessed at the data area top or summary listing level, then the application has been correctly configured for production use. We can rely on the previous regression test to insure that all the code is correct. Check "Concur" if the test cases adequately address the high-level functionality of each component.

QUALITY ASSURANCE REVIEW

The independent Quality Assurance Reviewer (QAR) will examine the deliverable for the presence of two attributes:

1. One or more technical reviews for this deliverable have been completed.

   Check "Concur" if one or more technical reviews have been completed.

2. All end-user and technical reviews indicate substantial or unconditional concurrence.

   Check "Concur" if there is substantial or unconditional concurrence from the technical reviewer(s).

INSTALLATION & ACCEPTANCE DCS

The Installation & Acceptance (I&A) class of deliverables are produced during the I&A stage. The purpose of the I&A class is to define the final hardware, software, configuration, and test results for the test and production environments for the combined set of application components.

STRUCTURE

The I&A class of deliverables is a composed of two primary documents:

- The Acceptance Configuration Plan
- An updated Acceptance Plan

CONTENT

ACCEPTANCE CONFIGURATION DOCUMENT

The Acceptance Configuration document contains detailed implementation specifications for the test and production environments. These configuration descriptions include detailed hardware, server, OS, and network configurations of the test and production server sets for the application. This information is used to establish a baseline configuration for the released system.

UPDATED ACCEPTANCE PLAN

The Acceptance Plan initially contains listings of the acceptance test cases to be run prior to the production release of the system. These listings are now updated with the TCRs and any TIRs associated with the performance of the tests. This information is used to document the successful execution of the acceptance test suite for the system.

REVIEW REQUIREMENTS

The I&A class is subject to Technical review and Quality Assurance review, but *not* End User Review. Each reviewer addresses specific issues based on their perspective and area of expertise and indicates their level of concurrence with the statements in the review form.

I&A REVIEW FORM

The review form allows the standard review form format as described in the previous chapter. Two review forms are available:

- Digital signature review form
- Physical signature review form

Instructions for using Shell Method review forms are available here.

TECHNICAL REVIEW

The technical reviewer will examine the deliverable for the presence of three attributes:

1. The Acceptance Configuration document accurately describes the final configuration of the test environment.

   The purpose of the Acceptance Configuration document is to document the detailed configuration of the test and production environments. These configuration descriptions include detailed hardware, server, OS, and network configurations of the test and production server sets for the application. This information is used to establish a baseline configuration for the released system.

   Many project team members have "production fever" at this point, resulting in poor accuracy for this information. Auditors know this. As a result, this baseline information is one of the first things they check during an audit. As technical reviewer, you will need to double-check each entry for accuracy, which means looking directly at hardware labels, directory listings, and so forth.

   Check "Concur" if the baseline information in the Acceptance Configuration document is exactly correct for the test environment.

2. The Acceptance Configuration document accurately describes the final configuration of the production environment.

   This is the same check as described above, only for the production environment. The reason it's a separate check is that some test and production environments are located in different facilities. Check "Concur" if the baseline information in the Acceptance Configuration document is exactly correct for the production environment.

3. The Acceptance Plan has been updated with accurate acceptance test results.

   Just as the Deployment Plan is updated with regression test results in the form of TCRs and TIRs associated with specific test cases, so is the Acceptance Plan updated for the acceptance test cases. Check "Concur" if the test case listings have been accurately updated with associated TCRs and TIRs.

QUALITY ASSURANCE REVIEW

The independent Quality Assurance Reviewer (QAR) will examine the deliverable for the presence of two attributes:

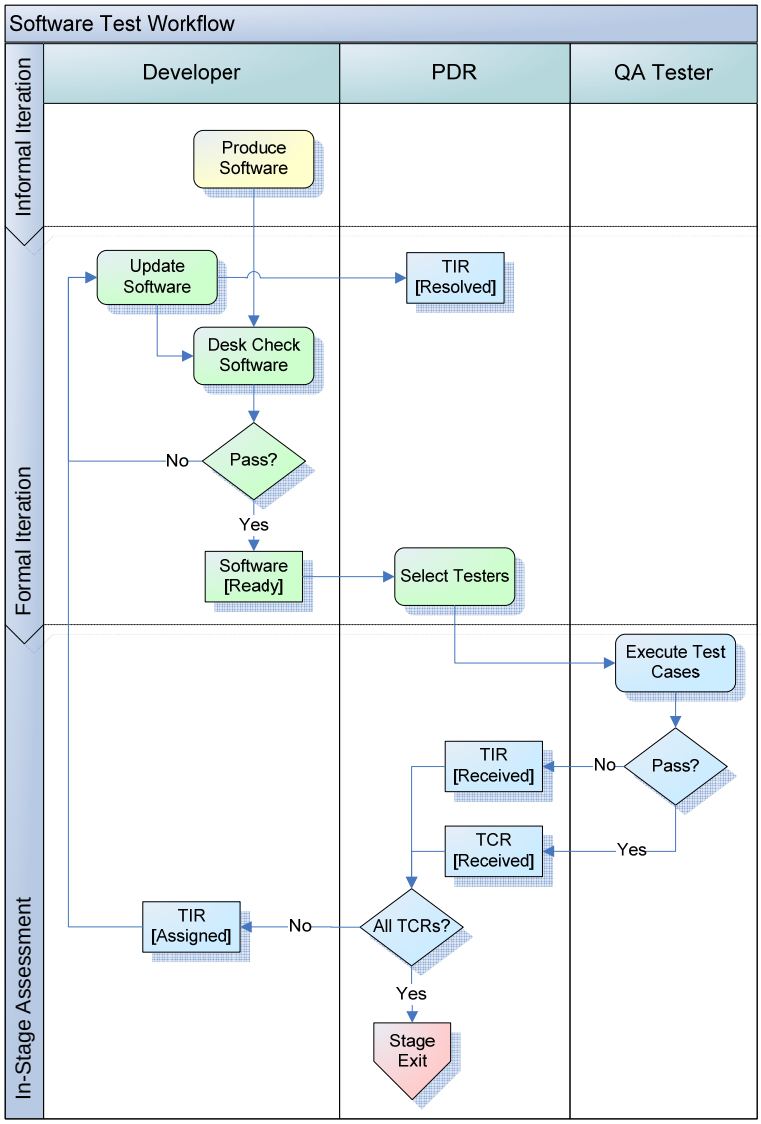
1. One or more technical reviews for this deliverable have been completed.

   Check "Concur" if one or more technical reviews have been completed.

2. The Technical review(s) indicate substantial or unconditional concurrence.
   Check "Concur" if there is substantial or unconditional concurrence from the technical reviewer(s).

SOFTWARE TESTING

Software testing validates the developed software to demonstrate that it functions in accordance with its design. The software test workflow manages the process of testing and feedback during the Integration & Test stage.



ACTORS

The actors associated with software testing include:

1. The software developers,
2. The PDR, and
3. Testing personnel, who may be drawn from the end-user community or development team, as determined by the PDR.

PROCESSES

The processes associated with software testing include:

1. Informal Iteration,
2. Formal Iteration, and
3. In-Stage Assessment

INFORMAL ITERATION PROCESS

During the informal iteration process, software artifacts that are prototypes and other forms of exploratory work are segregated from the artifacts intended for production delivery. The final set of production artifacts comprises the "produced" software for the current iteration and is usually referred to as a "candidate build."

FORMAL ITERATION PROCESS

The candidate build is tested by the development staff. This informal testing is typically executed against the design document. Formal test cases and automated testing scripts may be in place to assist this effort, depending on the resources available to the project. This developer self-testing process is termed a "desk check." Desk Checks are also referred to as Unit Tests. Once the software passes the desk check, the development team sets the build status to "ready" and informs the PDR. The PDR identifies testers with appropriate domain knowledge and initiates the in-stage assessment process.

IN-STAGE ASSESSMENT PROCESS

Testers execute test cases against the candidate build. For each test case that passes, the tester generates a Test Completion Report (TCR). For each test case that fails, the tester generates a Test Incident Report (TIR). Forms for both reports are available on the Shell Method Web site.

TEST COMPLETION REPORT (TCR)

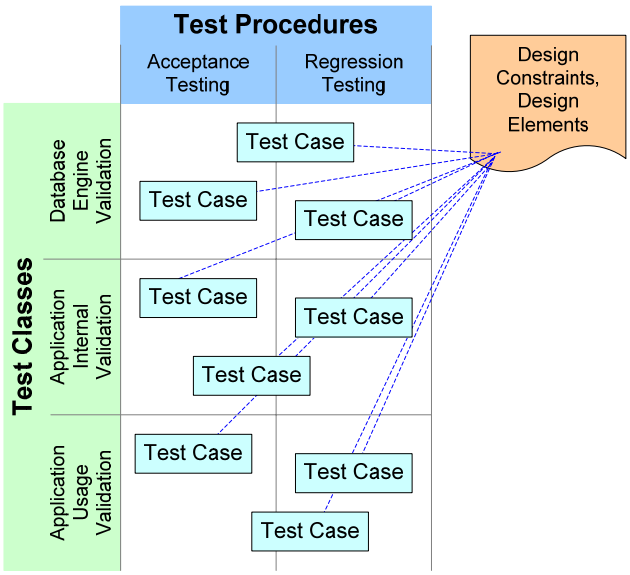
The tester uses the TCR to describe the details of the tests executed for the test steps associated with each test case. One TCR is filled out for each test case. Different projects will have different resources available for testing. For those projects constrained to manual testing, the description of how the testing was performed should include details of actions taken, values entered, and results observed. For projects with access to dedicated testing resources, the description may be as simple as a statement describing the use of a pre-defined automated test script.

TEST INCIDENT REPORT (TIR)

When one or more of the steps associated with a test case fail, the tester generates a TIR. One TIR may describe many test step failures, but all test steps must be associated with a single test case (one TIR per test case). The content of a TIR is developed first by the tester and then by the developer assigned to correct the failure. The tester describes the failure in sufficient detail to enable reproduction of the failure by the developer. The developer assigned to correct the failure uses this information to reproduce and identify the root cause of the failure. When the software has been corrected and desk checked by the developer, the TIR is updated to reflect the changes associated with the fix.

TEST METHODOLOGY

Testing is based on three key concepts: Test Cases, Test Classes, and Test Procedures. Basically, test cases belong to one of three test classes, and each test case may be utilized by one or more test procedures, as shown in the diagram below.



TEST CASES

Each test case describes a specific set of goals designed to validate the correct operation of a specific subset of the software. Test cases come in a variety of forms, each specific to the task at hand. In general, each test case is written specifically for end-users and developers that are familiar with the application.

TEST CLASSES

The three test classes include black-box testing, white-box testing, and database engine validation. Any single test case will belong to only one of these three test classes.

BLACK-BOX TESTING

Test cases of this class are generally performed by knowledgeable end-users and are used to validate the features that are presented to the end-user via the user interface. In simple terms, these test cases validate the usage of the features available to the end-user, treating the software as a "black box," because the internal functionality of the software is not visible. Test cases of this class are sometimes referred to as external testing.

The six categories of Black-box tests are:

1. Module entry & flow
2. Precondition business rules
3. Initialization business rules
4. Interaction business rules
5. Conclusion business rules
6. Performance testing

WHITE-BOX TESTING

Test cases of this class are generally performed by members of the development team and are used to validate the internal operations of the software that are visible to the developer. To accomplish this, members of the development team use special tools, such as a debugger, to validate the internal operation of the software. Test cases of this class are sometimes referred to as glass-box testing (because you can see inside the software) or internal testing.

The two categories of White-box tests are:

1. Database engine validation
2. Access control validation

DATABASE ENGINE VALIDATION

Test cases of this class are generally performed by members of the development team and are used to validate that the structure of the database as implemented matches the design as specified. This class of tests also validates that the performance of the database engine meets the constraints identified in the design document. To validate the structure, members of the development team use the administrative interfaces of the database engine to compare stored dictionary information with the physical database description developed during the design stage of this project.

To validate database engine performance, testers may use specialized software to perform load testing (simultaneous users) and volume testing (database capacity) on a database server configured to match the production server.

ACCESS CONTROL VALIDATION

Access control is validated by examining the user roles defined in the database engine and comparing them to the roles defined in the PDD, and then examining the table access restrictions to validate they are associated with the proper roles. This is typically a white-box test conducted by a database administrator using a standard engine administration tool.

TEST PROCEDURES

This test methodology utilizes two classes of test procedures:

1. Acceptance testing
2. Regression testing

ACCEPTANCE TESTING

The acceptance test procedure uses a "lightweight" set of test cases intended to validate that the software is functioning correctly at the top level. Acceptance tests are used for two reasons: Acceptance into testing, and acceptance into production.

When the application (or a portion thereof) is deemed ready for test by the development team, the acceptance test procedure is run to ensure that the test instances of the database server and application software are basically functional. If the software fails to pass this limited set of tests, the software is returned to the development team for correction. If the software passes the acceptance tests, it is then subjected to comprehensive test procedures as described below.

Once the application has passed the comprehensive test procedure, a production instance of the application is installed. The acceptance test procedure is run once again to ensure that the software was properly migrated, without forcing a repetition of the comprehensive test procedure. Successful completion of the

acceptance test procedure on the production instance of the application, coupled with the documentation showing that the software has previously passed the comprehensive test procedure, serves as the basis for acceptance of the software by the customer.

REGRESSION TESTING

The regression test procedure makes use of test cases that perform a detailed examination of the software and database engine. A regression test is a comprehensive test, incorporating all known features of the application. The objective of a regression test procedure is to examine everything in detail.

During production, the occasional bug may be discovered. Regression tests are also designed to ensure that all reported bugs have been fixed in all subsequent releases of the software. This becomes especially important when developers other than the original team are performing bug fixes. Certain test cases are specifically designed to make sure a bug has been fixed. The regression test procedure also uses test cases involving functions and features that may have been impacted by the bug fixing process.

CRITICAL SUCCESS FACTOR: APPLICATION FAMILIARITY

Familiarity with the application under test is the single critical success factor of this test plan, as it allows the specification of test cases that are focused on testing a specific area of the application, rather than broad spectrum test cases that walk a tester through, step-by-step, from the top of the application to the desired test areas.

For example, when a test case specifies the examination of a specific data entry screen, it does not provide a step-by-step checklist to arrive at that screen. Instead, the tester is expected to be familiar enough with the application that accessing the screen is not a mystery. This allows each test case to be tightly focused on a specific feature set.