5 SYSTEM ACCEPTANCE

Purpose

System Acceptance is the point in the lifecycle at which every aspect of the application being developed, along with any supporting data conversion routines and system utilities, are thoroughly validated by the Customer Representatives prior to proceeding with System Implementation.

This entire phase is centered around gaining sufficient evidence of the system's accuracy and functionality to be able to proceed to System Implementation with the highest level of confidence possible in the success of the system. This phase differs from System Construction in that acceptance testing is the final opportunity to establish that the system performs as expected in an environment that mimics production as closely as possible. In addition, while the Customer Representatives were certainly engaged throughout prior testing efforts, they now assume an even more critical role in the testing efforts in that they now need to exercise the system in the same way that they will once the full system is implemented. With the testing roadmap established in earlier lifecycle phases, the Customer Representatives now take responsibility for maneuvering the system through its operations.

In addition to confirming the operation of the system and its fit to the business needs that it is intended to satisfy, System Acceptance is also the point in the lifecycle during which all supporting documentation and reference materials are updated to guarantee their consistency with the final delivered system.

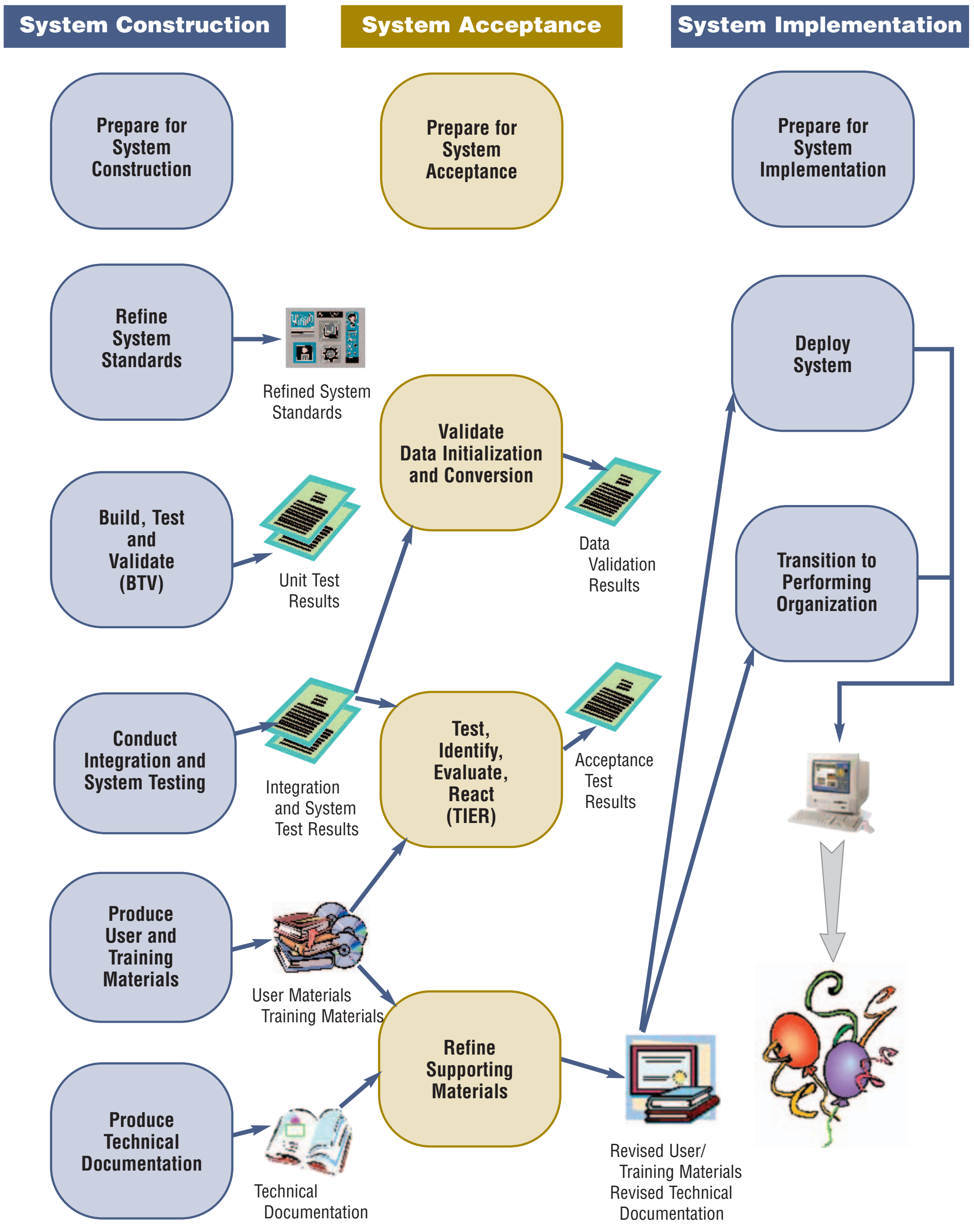
This phase consists of the following processes:

- **Prepare for System Acceptance,** where the system acceptance environment is established, and where the testing team is instructed in the use of processes and tools necessary throughout this phase.
- **Validate Data Initialization and Conversion,** where the processes and utilities used to populate the system database are tested to confirm that they provide a starting point from which the new system can begin processing.

- **Test, Identify, Evaluate, React (TIER),** where the system functions and processes undergo a series of exhaustive acceptance tests to validate their performance to specifications, and where examination of test results determines whether the system is ready to begin production.
- **Refine Supporting Materials,** where the various materials that support the use, operation, and maintenance of the system are updated to reflect any necessary adjustments resulting from acceptance test results.

The following chart illustrates all of the processes and deliverables of this phase in the context of the system development lifecycle.

Figure 5-1


System Construction
System Acceptance
System Implementation
Prepare for System Construction
Prepare for System Acceptance
Prepare for System Implementation
Refine System Standards
Refined System Standards
Deploy System
Validate Data Initialization and Conversion
Build, Test and Validate (BTV)
Unit Test Results
Data Validation Results
Transition to Performing Organization
Conduct Integration and System Testing
Integration and System Test Results
Test, Identify, Evaluate, React (TIER)
Acceptance Test Results
Produce User and Training Materials
User Materials Training Materials
Refine Supporting Materials
Produce Technical Documentation
Technical Documentation
Revised User/ Training Materials Revised Technical Documentation

List of Roles

The following roles are involved in carrying out the processes of this phase. Detailed descriptions of these roles can be found in the Introductions to Sections I and III.

- Project Manager
- Project Sponsor
- Business Analyst
- Data/Process Modeler
- Technical Lead/Architect
- Application Developers
- Technical Writer
- Software Quality Assurance (SQA) Lead
- Technical Services (HW/SW, LAN/WAN, TelCom)
- Information Security Officer (ISO)
- Technical Support (Help Desk, Documentation, Trainers)
- Customer Decision-Maker
- Customer Representative
- Stakeholders

List of Deliverables

The following table lists all System Acceptance processes, some techniques available for use in executing these processes, and process outcomes and deliverables.

Figure 5-2

Processes	**Techniques**	**Process Deliverables *(Outcomes)***
Prepare for System Acceptance	Interviews Site Walk-throughs Environmental Assessments Acceptance Test Plan Review	*Established Team and Environment for System Acceptance*
Validate Data Initialization and Conversion	Manual Testing Automated Testing Defect Tracking Regression Testing	Data Validation Test Results *Validated Data Initialization and Conversion Software*
Test, Identify, Evaluate, React (TIER)	Manual Testing Automated Testing Defect Tracking Regression Testing	Acceptance Test Results *Validated System* *Validated System Utilities*
Refine Supporting Materials	Technical Writing Illustration On-line Content Development Content Review	Revised User/Training Materials Revised Technical Documentation

5.1 PREPARE FOR SYSTEM ACCEPTANCE

Purpose

The purpose of **Prepare for System Acceptance** is to ensure that the testing environment to be used during System Acceptance is ready and operational, and to take any steps needed to prepare the acceptance testing team to successfully achieve their testing goals.

Description

This phase of the SDLC is significant because it is the last time that rigorous testing will be performed on the system before it goes into production. It is also very likely the first time that Customer Representatives will be able to exercise the application in a near-production environment, which adds a unique perspective to the testing efforts.

Preparation of both the testers and the environment in which they will operate is crucial to the success of this phase. User and training materials must be distributed in advance of this effort, and any training sessions needed to familiarize the testers with the application must be conducted.

Roles

- Project Manager
- Project Sponsor
- Business Analyst
- Data/Process Modeler
- Technical Lead/Architect
- Application Developer
- SQA Lead
- Technical Services
- Information Security Officer
- Technical Support
- Customer Decision-Maker
- Customer Representative
- Stakeholders

In an ideal situation, those participating in the testing should receive the exact training and materials intended for Consumers, so that the usefulness and acceptability of the materials can be validated.

In addition to familiarizing the testing team with the system, preparatory efforts must clarify for the team all testing roles and responsibilities, the timeline allocated to these efforts, and all processes to be followed regarding recording of testing results and reporting of defects. Although prior testing activities should have included Customer Representatives as part of the test team, it is common for this team to include an increased number of representatives so that real production operations can be better emulated. As a result, the testing team may now consist of participants who may not be as accustomed to rigorous testing activities as were members of the integration and system testing team, who typically have a more systems-oriented background. Therefore, expectations of these individuals need to be clearly defined, as do such elements as the testing strategy to be followed, the extent of testing to be performed, the definition of acceptance, etc.

Preparation of the environment in which acceptance testing will be performed is primarily focused on confirming that it is as close to the production environment as possible and on migrating the application from the QA to the Acceptance environment.

5.2 VALIDATE DATA INITIALIZATION AND CONVERSION

The purpose of the **Validate Data Initialization and Conversion** process is to confirm before the system begins production that all utilities and processes needed to load data into the system work correctly, and that any data carried forward from another system is migrated completely and accurately.

As important as it is to ensure that the new application functions properly, it is equally important to ensure the accuracy of the data being processed by the system. This effort starts with the initial loading of data, also known as “Day 1” data. The data is most often populated using two main methods – the manual loading of information required by the new system that cannot be extracted or obtained from an existing system, and the automated loading of information currently available in one or more existing data sources.

Roles

- Project Manager
- Project Sponsor
- Business Analyst
- Data/Process Modeler
- Technical Lead/Architect
- Application Developer
- SQA Lead
- Technical Services
- Information Security Officer
- Technical Support
- Customer Decision-Maker
- Customer Representative
- Stakeholders

The acceptance testing team must exercise all aspects of the data initialization and loading of information into the system database. Testing of the data load should be conducted very much like the testing of the application itself, with all participants capturing the test results and identifying any defects. The goal is to determine whether the quality of the data load process and the resulting data are sufficient to proceed with implementing the system. Any data problems that jeopardize the eventual success of the system clearly need to be addressed. It may be perfectly acceptable, however, to advance into further application testing activities with a known set of low-impact data problems, as long as the impact of these defects on subsequent testing efforts is understood in advance, along with a defined timeframe by which the errors need to be corrected.

The key difference between acceptance testing activities and all prior testing efforts is that while it was reasonable to expect iterative testing cycles in earlier phases, the goal of acceptance is to demonstrate that the deployment and use of the system will be successful in a production-like setting. Therefore, whether validating data initialization efforts or specific system functions (as described in the following process), all activities performed in this phase should already have been successfully demonstrated in System Construction, albeit in a slightly different environment.

This does not mean that that there won't be decision points throughout this phase at which test results will need to be evaluated, usually as part of an overall set of test results, to determine the proper course of action. Once these test results are in hand, then an informed decision can be made to either move ahead with continued testing, or to address known issues as they are discovered, only moving forward when the error condition has been corrected.

◆ **Data Validation Test Results** – A comprehensive set of completed test plans identifying all data initialization and conversion tests that were performed, along with the detailed outcomes of these tests, the list of defects identified as a result of these tests, and the results of any subsequent retests. These test results are contained in the Acceptance Test Plan section of the Technical Specifications.

5.3 TEST, IDENTIFY, EVALUATE, REACT (TIER)

Purpose

The purpose of the **Test, Identify, Evaluate, and React** process is to execute a complete suite of tests against the application in a production-like environment, assess the results of the tests, and determine whether it is appropriate to proceed with System Implementation, or whether corrective actions are required to address any defects encountered.

Description

Roles

- Project Manager
- Project Sponsor
- Business Analyst
- Data/Process Modeler
- Technical Lead/Architect
- Application Developer
- SQA Lead
- Technical Services
- Information Security Officer
- Technical Support
- Customer Decision-Maker
- Customer Representative
- Stakeholders

This process is analogous in many ways to the Conduct Integration and System Testing process in System Construction. While the primary responsibility for conducting the testing has moved from the Application Developers to the Customer Representatives, many of the principles that applied to earlier testing efforts apply here as well. The need for capturing testing metrics remains essential for conducting quality assurance practices, and the adherence to rigorous configuration management and release migration procedures remains crucial to understanding exactly which versions of the software are being tested at any given time.

Because the Customer is anxious to implement the new system and restore testing personnel to their primary business functions, acceptance testing tasks are often underemphasized. Thorough testing procedures cannot be stressed strongly enough. Failure to perform these tasks with high quality and attention to detail could cause serious problems in the future, perhaps after the system has been placed into production. Time invested at this stage will save time overall.

Throughout this process, any problems identified by the testers must be recorded and tracked to closure on defect logs. Continual interaction is essential between those doing the testing and those who developed the application. The Project Manager must closely manage the activities in order to ensure adherence to the Project Scope and Schedule.

Another factor to consider during this process is that organizations may choose to perform parallel operations during acceptance testing. This requires a continuation of existing business processes at the same time that the new system is being tested. This may mean that while transactions are entered into the new system, they will also need to be entered separately into any existing legacy systems, or may need to be captured in whatever manual systems are currently being utilized. This often requires additional staff or extra hours in order to keep up with the additional workload, but allows the results of the two processes to be compared for accuracy. If this parallel approach is taken, the extra burden on the Performing Organization will need to be estimated and communicated to the Stakeholders so that they can make an informed decision regarding any additional costs, the duration for which the organization can sustain these costs, and the benefits resulting from this approach.

Regardless of whether or not parallel operations are planned, the recommended approach for testing applications is to drive a time-boxed set of coordinated tests that adhere to the TIER approach, as follows:

Test: Following the initialization of the Acceptance environment, acceptance testing will occur, during which all perceived defects in the application are recorded. The exact frequency with which these defects are reported will vary with each project – the important point to note here is that communication of these defects needs to be constant throughout the testing to avoid the 'big bang' effect that can occur when all issues are reported only upon completion of the testing.

Identify: The Project Manager will engage the appropriate team members to analyze each reported defect to determine the cause of the defect being reported, and to identify whether or not a true system error has been encountered. While defects are often related to adjustments needed in the application software, it is equally possible that the root cause of a reported defect is the tester's misunderstanding of exactly how the system was designed to operate. Changes to normal business operations due to new functionality, combined with the revised look and feel of the application, often result in system errors being reported that in fact are examples of the system working exactly as designed.

The Project Manager should keep in mind that system errors or defects may be reported that result more from a misinterpretation of expected functionality as opposed to a technical defect. Even though the system may be operating exactly as defined, this scenario may point to other non-technical errors associated with the system. It may be possible that the on-line help system may not sufficiently describe the system's operations, or that a component of the overall training package may require an increased level of detail in one or more areas. Take advantage of System Acceptance to evaluate the entire system and its supporting materials, and make adjustments now while you can still get out in front of the final implementation.

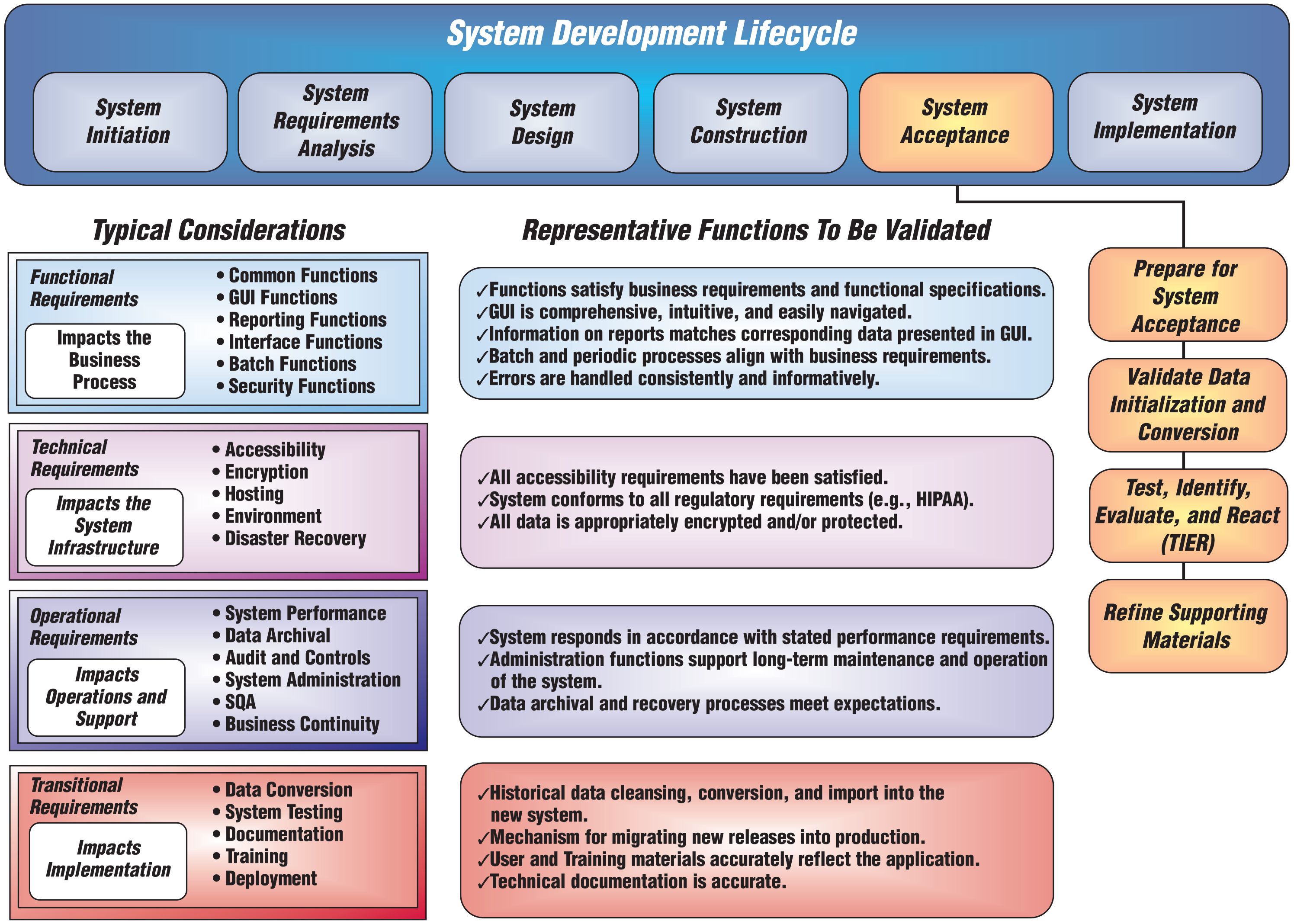
Evaluate: If a defect in the application is identified, the Project Team will work together to identify the appropriate corrective action. A decision will be made regarding whether or not system modifications are required, whether data loaded in the prior process needs to be corrected, whether operational procedures need to be adjusted, or whether some other form of correction is required. Once a corrective action is identified, it will then be prioritized, along with all other on-going activities, to determine if this issue is of sufficient impact to warrant adjustments being made during System Acceptance. Since all testing efforts should adhere to the Project Schedule, the underlying question becomes, "Can the system be placed into production with the existence of this condition, or is its impact such that implementation of the system is not possible due to inability to perform essential business operations"? If an acceptable work-around exists, or if the impact is minimal, then a determination can be made to handle the correction as part of a normal production support issue once the system is implemented.

React: Once the appropriate actions and priorities have been identified, the defect will be resolved. For those defects requiring changes to the application, the appropriate changes should be made, tested, and re-released to the Customer Representatives for validation. For all other defects, the agreed-to resolution should be communicated to all parties.

The key to successful completion of System Acceptance is the clear definition of go/no-go criteria that can be used to define the set of circumstances that would preclude placing the application into production. Should a “show stopper” be identified in these final days of testing, the Project Team must estimate and plan the appropriate corrective actions and retesting needed to resolve the problem, and then adjust the testing schedule accordingly using the standard Project Change procedures. However, if the list of issues at the end of acceptance testing contains only low priority, low impact modifications, (i.e., those that do not significantly inhibit the use of the application), testing can be considered complete. At this point, the project should progress to the next phase, with all remaining issues addressed through the application support mechanisms.

- **Acceptance Test Results** – A comprehensive set of completed test plans identifying all acceptance tests that were performed, along with the detailed outcomes of these tests, the list of defects identified as a result of these tests, and the results of any subsequent retests. These test results are contained in the Acceptance Test Plan section of the Technical Specifications.

Figure 5-3 System Acceptance Considerations


System Development Lifecycle
System Initiation
System Requirements Analysis
System Design
System Construction
System Acceptance
System Implementation
Prepare for System Acceptance
Validate Data Initialization and Conversion
Test, Identify, Evaluate, and React (TIER)
Refine Supporting Materials

Typical Considerations

Functional Requirements — Impacts the Business Process
- Common Functions
- GUI Functions
- Reporting Functions
- Interface Functions
- Batch Functions
- Security Functions

Technical Requirements — *Impacts the System Infrastructure*
- Accessibility
- Encryption
- Hosting
- Environment
- Disaster Recovery

Operational Requirements — *Impacts Operations and Support*
- System Performance
- Data Archival
- Audit and Controls
- System Administration
- SQA
- Business Continuity

Transitional Requirements — *Impacts Implementation*
- Data Conversion
- System Testing
- Documentation
- Training
- Deployment

Representative Functions To Be Validated

✓Functions satisfy business requirements and functional specifications.
✓GUI is comprehensive, intuitive, and easily navigated.
✓Information on reports matches corresponding data presented in GUI.
✓Batch and periodic processes align with business requirements.
✓Errors are handled consistently and informatively.

✓All accessibility requirements have been satisfied.
✓System conforms to all regulatory requirements (e.g., HIPAA).
✓All data is appropriately encrypted and/or protected.

✓System responds in accordance with stated performance requirements.
✓Administration functions support long-term maintenance and operation of the system.
✓Data archival and recovery processes meet expectations.

✓Historical data cleansing, conversion, and import into the new system.
✓Mechanism for migrating new releases into production.
✓User and Training materials accurately reflect the application.
✓Technical documentation is accurate.

5.4 REFINE SUPPORTING MATERIALS

Purpose

Refine Supporting Materials ensures that all materials relating to the new application are kept up-to-date with any changes that may be introduced during System Acceptance.

Description

Roles

- Project Manager
- Project Sponsor
- Business Analyst
- Data/Process Modeler
- Technical Lead/Architect
- Technical Writer
- SQA Lead
- Technical Services
- Information Security Officer
- Technical Support
- Customer Decision-Maker
- Customer Representative
- Stakeholders

Despite the best efforts of the Project Team throughout the earlier phases of the lifecycle, it is common for acceptance testing activities to uncover issues that require changes to the application. In the best cases, these may be nothing more than small cosmetic changes. In extreme cases, defects detected during testing could result in major subsystems of the application being redesigned and rewritten. Regard- less of the situation, all supporting materials, (both Consumer- and Technical Support-oriented), should be reviewed to make sure that they still accurately reflect the system that will be deployed in System Implementation.

- **Revised User/Training Materials** – An updated set of materials aimed at assisting Consumers with the use and operation of the application, reflecting any changes that were introduced as a result of acceptance testing efforts.

- **Revised Technical Documentation** – A corresponding set of updated technical materials, again reflecting any changes introduced as a result of acceptance testing efforts and defining aspects of the application that will be useful to those individuals responsible for on-going system maintenance.

Measurements of Success

The ultimate measurement of success for System Acceptance is the agreement by the Customer to move the system into production.

Meanwhile, the Project Manager can still assess how successfully the project is proceeding by utilizing the measurement criteria outlined below. More than one “No” answer indicates a serious risk to the eventual success of the project.

Figure 5-4

Process	Measurements of Success	Yes	No
Prepare for System Acceptance	Do you have the commitment from Customer Decision-Makers to make the right people available to the extent necessary for the duration of Acceptance activities?		
	Does your testing community agree that they are adequately prepared for the Acceptance activities?		
	Does everyone have access to and the correct security level for the system?		
Validate Data Initialization and Conversion	Can you say with confidence when each outstanding data initialization and conversion defect in the log will be corrected?		
	Do your Customers agree with your assessment?		
Test, Identify, Evaluate, and React (TIER)	Can the developers fixing the defects determine, based on the defect log and test results, what the problem scenario was and what outcome was expected vs. what was experienced?		
	Are retesting efforts demonstrating that reported defects are being resolved with new releases, and that the same issues are not being reported from iteration to iteration?		
Refine User and Training Materials	Have you made changes to the user/training materials as a result of your experiences in user training and acceptance testing in this phase?		
	Have you made changes to the Technical Documentation as a result of its review by a representative of the group that will assume responsibility for the system once it’s deployed?		

Phase Risks / Ways to Avoid Pitfalls

PITFALL #1 – YOU EXPECT ME TO DO WHAT?

The long Construction cycle is finally over. The system is pretty much done. Your team knocked itself out delivering what was promised, on time, within budget. You can hardly curb your enthusiasm as you call the Customer Decision-Maker to invite his cohort to spend a few weeks in the trenches slugging it out with the remaining system bugs. Curiously, all you get is dead silence, followed by a string of strangely unintelligible exclamations. Oops!

Customers (and Consumers), especially ones not experienced with formal system acceptance activities, assume that the new system will just materialize on their desktops, free of defects and perfect in every way. They view it the same way they view shrink-wrapped software packages, and have no idea how much effort goes into getting the system to the turnkey stage. It is a great shock for them to learn that, in addition to letting the system developers know what they wanted at the beginning, they need to verify at the end that what the developers actually developed meets their expectations.

Since the acceptance activities are fairly rigorous and protracted, it behooves an astute Project Manager to set those expectations way up front. Disarm them with the intention of making sure they got what they asked for, detail for them the acceptance activities and the expected level of participation and commitment, and keep reminding them, as System Acceptance nears, of their promises of people and time.

PITFALL #2 – WHAT THEY WANT VS. WHAT THEY ASKED FOR

OK, you avoided the pitfall #1 above, and an eager and agreeable group of Customers traipsed over to your neck of the woods to try out the new system. However, the honeymoon is over real quick when they actually try out the new functions. Between System Requirements Analysis and System Acceptance, time has passed, things changed and people moved around, and now nobody remembers who wanted what and why; they just know that what they see is not something they want to get.

One of the hardest things to manage during the system development lifecycle is expectations. Keeping a good audit trail should help. How good were your deliverable definitions? How tight were your acceptance criteria? Were those deliverable approval signatures written in blood – or water?

The answers to these questions spell a difference between orderly change control, and unmitigated disaster.

PITFALL #3 – FLOATING THE GARBAGE DOWNSTREAM

Finally, you avoided both of the above pitfalls, and the Customer Representatives are oohing and aahing about the system design … until they actually try to DO something. Then, all heck breaks loose: the navigation is off, the business logic is faulty, the system crashes, and the dreaded hourglass just keeps flipping over and over and over and over… endlessly…until the Customers see red behind the Blue Screen of Death. It is obvious that the system was not tested properly, and the Customers naturally resent it. Nasty rumors begin to spread, and instead of the welcome mat, the Consumers ready tar and feathers for the system deployment ceremonies.

In the heat of the construction homestretch, the temptation is to take short-cuts assuming any problems can be fixed downstream: cutting corners on software quality assurance at the unit test level, hoping to make it up during integration testing; skipping functionality reviews, hoping that the Customers will catch the errors during acceptance testing; even short-shrifting the initial training, hoping to make up for it during Implementation.

The problem is, there is never enough time in subsequent phases either. Plus, the expectations have not been set up. So if you float the consequences of bad decisions downstream, you'll just have a bigger pile of trash to deal with, instead of unfurling your sails and parading across the finish line.

PITFALL #4 – "IT'S TOO LATE, BABY!"

Another consequence of trying to short-cut the process upstream and hoping to make it up downstream is that a point comes when it's too late to address some issues. In System Acceptance, it's too late to fix system performance problems. It's too late to correct data conversion routines. It's too late to redefine core functionality, and it may even be too late to introduce minimal business process changes.

Problems like that cannot be fixed in this phase. If you can't avoid them, what you need to do is go back, and loop the lifecycle over. For data conversion problems, you probably need to go back to Construction. For performance problems, to Design. And as for problems with core functionality (with apologies to Carole King) – "Something inside has died" and you'd better push the old STOP button and rewind to the beginning; then, maybe, "there will be good times again."

PITFALL #5 – PLAYING THE BLAME GAME

When the Customer is unhappy with the new system, and is threatening to "pull the plug" on acceptance, the temptation on the part of many members of the team is to play the blame game: it's someone else's fault! The database administrators blame the network people; the network folks blame the system developers; and the programmers blame the ultimate catch-all: user error. The problem is, among all the finger-pointing, the real problem goes unsolved.

As Project Manager, your job is to keep the team together and remind people that only by pulling together will they prevail. Insist on everyone acting as a team, concentrating on solutions rather than problems, and helping each other rather than blaming the other guy.

Frequently Asked Questions

What do I do when my Project Budget does not include a QA function?

The simple fact is that quality is an indispensable part of any effort (from both project management and system development perspectives). Building a product without quality controls is wrought with risk: it will not satisfy your Customer, and will reflect poorly on your reputation.

Assuming that you cannot do a change control to add a QA function to your Project Budget, the good news is that, in a pinch, you can do without a separate QA function by incorporating quality assurance into every procedure and task, and taking on quality control responsibilities yourself.

You need to incorporate rigorous review cycles into production and acceptance of every deliverable, by using peer review mechanisms as well as inviting external SME's. As is stated in the text above, "It is more important *that* the reviews be done than *how* they are done." Sometimes even having non-technical independent observers sitting in on reviews brings extra gravity and rigor to the proceedings. Another trick is getting the Customers even more closely involved in reviewing works in progress and providing feedback.

Finally, you will need to roll up your sleeves and personally check out test scripts and acceptance procedures, and join in the testing activities – not necessarily to do any significant testing yourself, but to ensure that it gets done thoroughly and correctly.

Who should be doing the testing? I can't trust the developers to check their own work! Can I?

There are organizations where the testing function is separated into its own business unit. There are other organizations that have a separate QA function, but its members join the Project Teams at certain parts of the lifecycle, and perform testing on site. Finally, there are organizations that incorporate quality in everything they do, and thus have no separate QA function.

Each approach has its own pluses and minuses, but the important concepts are:

1. Test plans and scripts need to be developed before any coding is done
2. Test scripts need to be executed faithfully, and the results communicated immediately to the developers
3. System development should not proceed until the defects have been corrected
4. The same defects in different testing cycles point to a serious problem that has to be resolved before any further work is done

As far as the developers are concerned, sure you can trust them! To check each other's work, that is.

Who owns development of acceptance test plans? Who should the Project Manager enlist in their development?

Since Customer Representatives execute acceptance test plans, they ought to participate in their development. They need to understand the process, and be comfortable with its documentation. Plus, they probably can think up testing scenarios that the developers would never imagine!

The ownership of the process, though, still rests with the Project Team, and ultimately with the Project Manager.