United States General Accounting Office

GAO

Accounting and Information Management Division

November 1998

Year 2000 Computing Crisis: A Testing Guide

GAO/AIMD-10.1.21

Preface

This guide[1] presents a step-by-step framework for managing all testing activities related to the Year 2000 problem. It describes five phases of Year 2000 testing activities, beginning with establishing an organizational testing infrastructure, followed by designing, conducting, and reporting on four incremental levels of system-related testing (software unit testing, software integration testing, system acceptance testing, and end-to-end testing). To support these five phases, the guide also describes test oversight and control activities.

The guide recommends the structure and discipline in testing that characterize mature software and system development/acquisition and maintenance processes. While some organizations may forego some of the rigor and discipline recommended in this guide, this choice should be made after careful consideration of the level of business risk the organization is willing to assume in addressing the Year 2000 problem.

The Year 2000 problem is rooted in how dates are recorded and computed. For the past several decades, systems have typically used two digits to represent the year, such as "98" for 1998, to save electronic storage space and reduce operating costs. In this two digit format, however, 2000 is indistinguishable from 1900. Because of this ambiguity, date-dependent software, firmware, and hardware could generate incorrect results or fail to operate altogether when processing years beyond 1999.

Unless corrected, such failures may have a costly, widespread impact on federal, state, and local governments; foreign governments; and private sector organizations. All sectors of the economy, many of which provide goods and services that are vital to the nation's health and well being, are at risk, including telecommunications; public utilities; transportation; banking and finance; commerce and small business; national defense; government revenue collection and benefit payment; and health, safety, and emergency services. Moreover, Year 2000 problems in one sector will cascade to others due to the many interdependencies and linkages among them.

Since early 1997 GAO has sought to promote effective national Year 2000 program leadership and management. As part of this effort, GAO published and has since updated a guide that offers a structured, step-by-step approach for reviewing and assessing an organization's state of Year 2000 readiness.[2] The guide describes five generally sequential

---

[1]This guide was issued as an exposure draft in June 1998. It incorporates, as appropriate, comments received through October 1998.

[2]Year 2000 Computing Crisis: An Assessment Guide (GAO/AIMD 10.1.14, issued as an exposure draft in February 1997; issued final in September 1997).

Year 2000 program phases and program/project management activities that transcend the phases. The five phases are:

- Awareness
- Assessment
- Renovation
- Validation
- Implementation

To supplement this enterprise readiness guide, GAO is publishing more detailed guidance on key Year 2000 phases and transcending activities embedded in its five phase model.[3] One such transcending activity is Year 2000 testing. In fact, although the most concentrated level of testing activity occurs during the renovation and validation phases, important aspects of Year 2000 testing span all five phases of the conversion model.

Complete and thorough Year 2000 testing is essential to provide reasonable assurance that new or modified systems process dates correctly and will not jeopardize an organization's ability to perform core business operations after the millennium. Moreover, since the Year 2000 computing problem is so pervasive, potentially affecting an organization's systems software, applications software, databases, hardware, firmware and embedded processors, telecommunications, and external interfaces, the requisite testing is extensive and expensive. Experience is showing that Year 2000 testing is consuming between 50 and 70 percent of a project's time and resources.

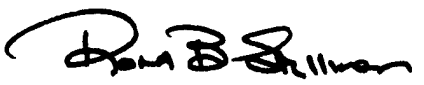
To be done effectively, this testing should be planned and conducted in a structured and disciplined fashion. This guide describes a step-by-step framework for managing, and a checklist for assessing, all Year 2000 testing activities, including those activities associated with computer systems or system components (e.g., embedded processors) that are vendor supported. This disciplined approach and the prescribed levels of testing activities are hallmarks of mature software and system development/acquisition and maintenance processes. Organizations that already have mature programs can easily extend them to incorporate effective Year 2000 testing; organizations with immature and undisciplined software development/acquisition and maintenance processes will find effective Year 2000 testing more challenging and demanding and should therefore ensure that sufficient management attention and resources have been allocated to Year 2000 testing to compensate for the added risk caused by this immaturity. Many organizations are attempting to reduce risk by prioritizing their systems maintenance efforts, limiting or freezing all changes not related to attaining Year 2000 compliance. As is true for all Year 2000 program decisions, the extent to which the testing rigor and discipline defined in this guide is embraced and instituted by each organization should be a business-based,

---

[3]Year 2000 Computing Crisis: Business Continuity and Contingency Planning (GAO/AIMD 10.1.19, issued as an exposure draft in February 1998; issued final in August 1998).

risk-driven decision (i.e., what level of business risk is an organization willing to assume by foregoing the proven tenets of effective testing defined in the guide).

An electronic version of this guide is available from GAO's World Wide Web server at the following Internet address: *<http://www.gao.gov>*. If you have any questions about this guide, please contact me at (202) 512-6412, Keith Rhodes at (202) 512-6412, Randy Hite at (202) 512-6256, Naba Barkakati at (202) 512-4499, or Madhav Panwar at (202) 512-6228. We can also be reached by email at *stillmanr.aimd@gao.gov*, *rhodesk.aimd@gao.gov*, *hiter.aimd@gao.gov*, *barkakatin.aimd@gao.gov*, and *panwarm.aimd@gao.gov*.



Dr. Rona B. Stillman
Chief Scientist for Computers
and Telecommunications

Contents

Overview of GAO's Managed, Five-Step Approach to Year 2000 Testing

This guide is intended to aid organizations in managing and assessing their Year 2000 testing programs. An effective testing program is an essential component of any Year 2000 program or project. More time is typically spent on testing than on any other program or project activity. Because Year 2000 conversions often involve numerous large interconnecting systems with many external interfaces and extensive supporting technology infrastructures, Year 2000 testing should be approached in a structured and disciplined fashion.

This guide presents a Year 2000 test model that provides such an approach. The test model sets forth five levels of test activity supported by continuous management oversight and control activities. The first level establishes the organizational infrastructure key processes needed to guide, support, and manage the next four levels of testing activities, and includes creating institutional structures, identifying and allocating resources, establishing schedules, and formulating policies, plans, standards, etc. for an organization's Year 2000 testing program. The next four levels provide key processes for effectively designing, conducting, and reporting on tests of incrementally larger system components: software unit/module tests, software integration tests, system acceptance tests, and end-to-end tests. The key processes focus on testing of software and system components that the organization is directly responsible for developing, acquiring, or maintaining. Key processes, however, are also defined to address organizational responsibilities relative to testing of vendor-supported and commercial, off-the-shelf (COTS) products and components (e.g., hardware, systems software, embedded processors, telecommunications, COTS applications). (Figure 1 summarizes the model.)

The test model builds upon and complements the five phase conversion model described in GAO's Year 2000 readiness guide.[4] The test model's five levels of test activity span all five phases of GAO's Year 2000 conversion model, with the preponderance of test activity occurring in the conversion model's renovation and validation phases. (Figures 2 and 3 relate the conversion and test models.)

The guide incorporates guidance and recommendations of standards bodies, such as the National Institute of Standards and Technology (NIST) and the Institute of Electrical and Electronic Engineers (IEEE), on Year 2000 testing practices and draws on the work of leading information technology organizations including the Software Engineering Institute, Gartner Group, Software Quality Engineering, Software Productivity Consortium, and the UK Central Computer and Telecommunications Agency.

---

[4]Year 2000 Computing Crisis: An Assessment Guide (GAO/AIMD-10.1.14, issued as an exposure draft in February 1997; issued final in September 1997).

Figure 1: Year 2000 Test Model

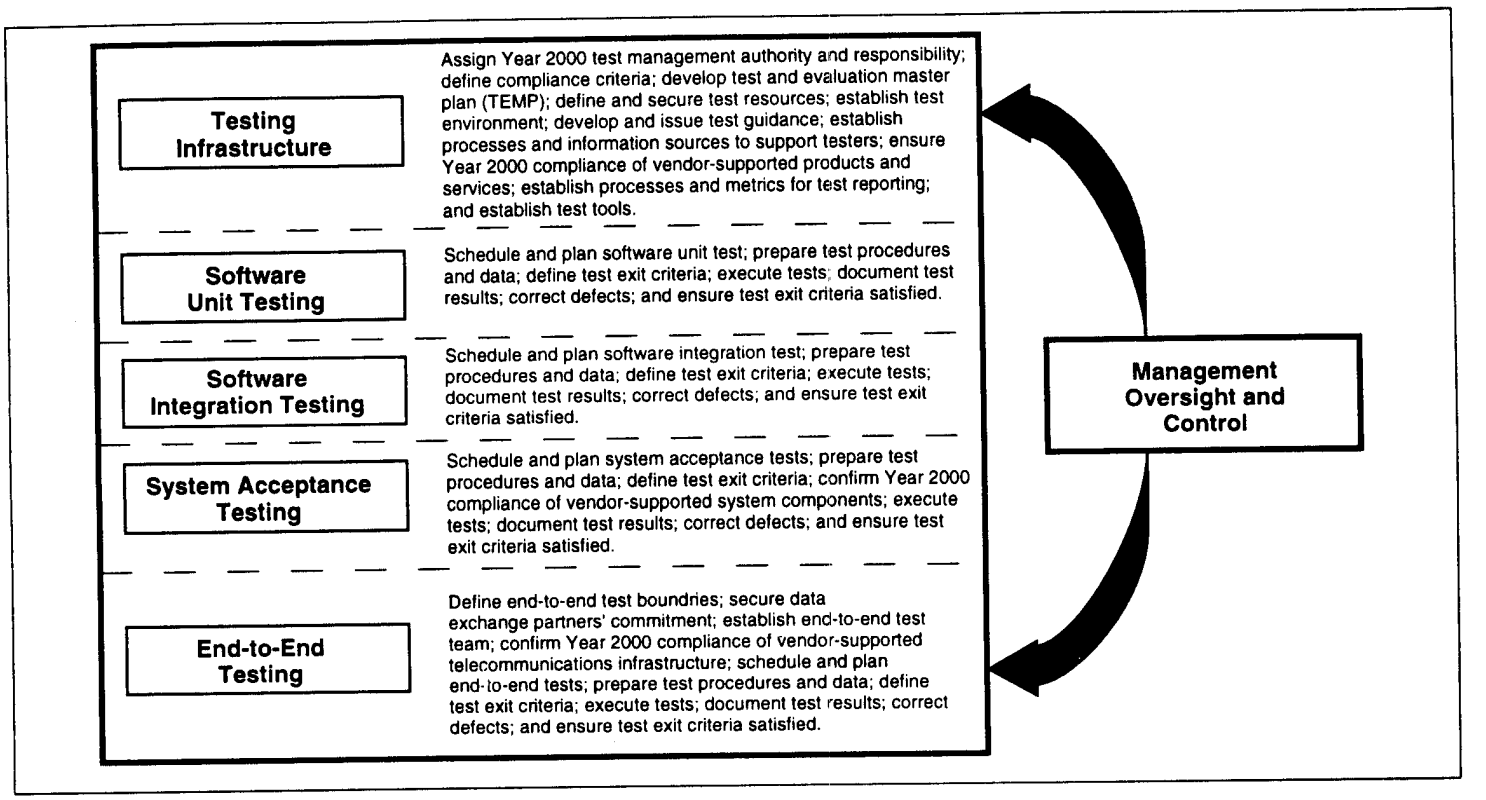


Figure 2: Crosswalk Between Conversion Model Phases and Testing Model Levels

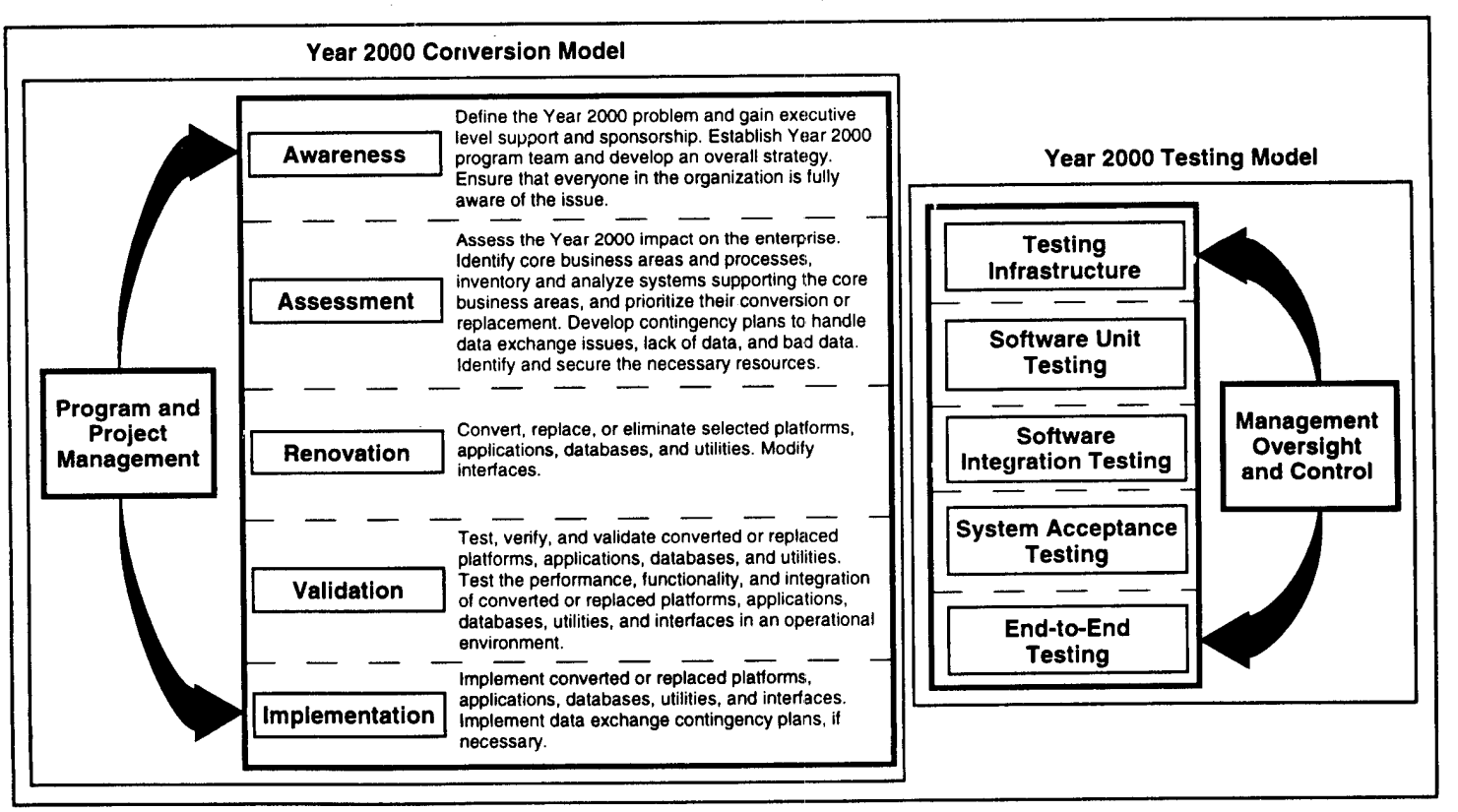
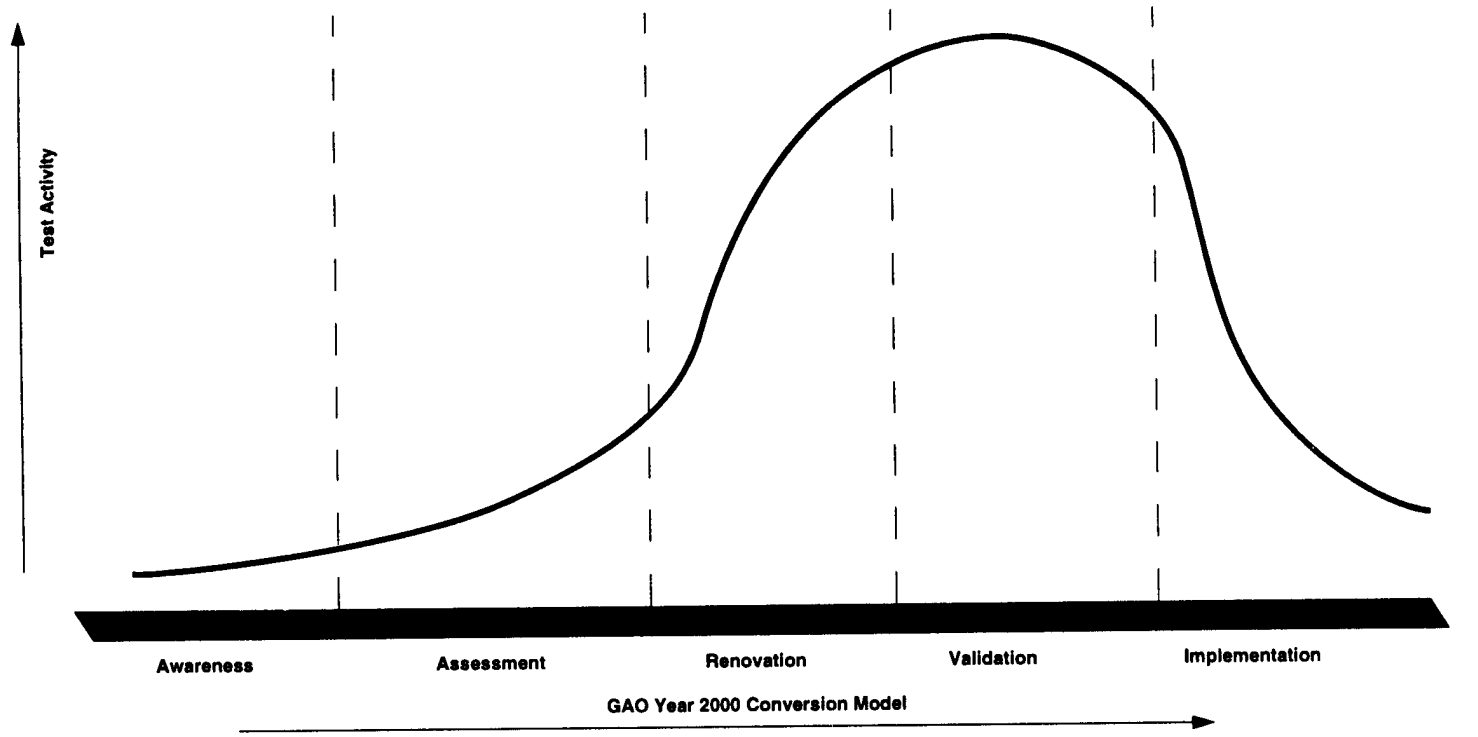


Figure 3: Illustrated Approximation of Test Resources Expenditure By Conversion Model Phase



1.0 Testing Organizational Infrastructure

As explained in GAO's Year 2000 readiness guide,[5] one of the first and most critical steps in the Year 2000 conversion process is appointing a Year 2000 program manager and establishing the enterprise's Year 2000 program office. When establishing this program management structure, a Year 2000 test manager (program level) should be designated and given the authority and responsibility for ensuring that Year 2000 testing is planned, conducted, and reported on in a structured and disciplined fashion, and is independently reviewed by a quality assurance/verification and validation group.

The test manager should, at a minimum, (1) define and assign roles, responsibilities, and expectations for Year 2000 testing (program level and project level), (2) define criteria for verifying a system as Year 2000 compliant, (3) develop and maintain a test and evaluation master plan, (4) establish independent quality assurance/verification and validation of test activities, (5) obtain necessary test budgets, (6) establish test environments/facilities, augment them as needed, and schedule their use according to established priorities, (7) develop and issue test guidance defining policies, principles, strategies, standards, and processes relevant to planning, executing, and reporting on each level of testing (software unit/module, software integration, system acceptance, and end-to-end), and ensure that this guidance is understood and followed, (8) establish support processes and resource/information sources, (9) establish policies and expectations on securing assurance that vendor-maintained and supported products are or will be Year 2000 compliant, (10) establish formal test activity and progress reporting requirements, and (11) establish a library of test tools.

[5]Year 2000 Computing Crisis: An Assessment Guide (GAO/AIMD 10.1.14, issued as an exposure draft in February 1997; issued final in September 1997).

Key Processes

1.1 Assign Year 2000 test management authority and responsibility
1.2 Define Year 2000 compliance criteria
1.3 Develop organizational Year 2000 test and evaluation master plan (TEMP)
1.4 Engage the quality assurance/verification and validation group
1.5 Define and secure test budgets
1.6 Establish new or augment existing test environments and schedule their use
1.7 Develop and issue organizational Year 2000 test guidance
1.8 Establish processes and information sources to support testers and test activities
1.9 Provide for ensuring Year 2000 compliance of vendor-supported products and services
1.10 Establish processes and metrics for reporting test activity and progress
1.11 Establish a library of test tools

1.1 Assign Year 2000 test management authority and responsibility

Organizational authority and responsibility, and thus accountability for testing, need to be clearly and formally defined, first within the Year 2000 program office, and then within each Year 2000 project office. If such a test management organizational structure already exists, it should be used and augmented as needed; if it does not, one should be established. Regardless, this structure needs to specify the respective organizational components' authorities and responsibilities for each applicable level of testing (software unit/module, software integration, system acceptance, and end-to-end) as well as for test management and oversight (planning, execution, reporting, and quality assurance).

1.2 Define Year 2000 compliance criteria

In order for an organization to verify that an application or system is Year 2000 compliant, the organization must clearly understand and specify what Year 2000 compliance means. One way to do this is to define Year 2000 compliance criteria. These criteria can also serve as the organization's test objectives. Examples of Year 2000 compliance criteria/test objectives include:

- *No valid date value for current date will cause any interruption in system operation.*
- *Date-based system functionality must behave consistently for dates prior to, during, and after year 2000.*

- *In all system interfaces and data storage areas, the century in any date must be specified either explicitly or by unambiguous algorithms.*

- *All years divisible by four must be recognized as leap years, unless they are also divisible by 100. There is an exception in that years divisible by 400 are leap years even though they are divisible by 100. (The year 2000 is an example of the exception and thus must be recognized as a leap year.)*

1.3 Develop an organizational Year 2000 test and evaluation master plan (TEMP)

As part of the organization's establishment of an enterprise Year 2000 program strategy, one or more test strategies or TEMPs should be developed, issued, and continuously updated to reflect changes in the test organization's structure, actual test progress and experience, actual resource availability, and changing priorities. The TEMP should describe the test organization and its components' roles and responsibilities, system/project priorities, a master schedule of high-level test activities for each system/project, and the test resources to be used in carrying out these activities (people, tools, facilities, contractors). The TEMP should provide sufficient detail to allow system/project-specific test planning to occur, as well as to permit program office tracking of high-level test activity progress. For example, the TEMP should have milestones, including completion dates, for application/system acceptance tests, should specify project progress metrics, and should allocate common test facilities and other resources among system renovation projects competing for these facilities and resources.

1.4 Engage the quality assurance/verification and validation group

The role and responsibilities of the independent quality assurance group or independent verification and validation (IV&V) group should be defined.[6] *This group should work with the Year 2000 program manager and/or individual project managers, as appropriate, and should develop a plan of action that provides the group visibility into the various levels of test activities and permits it to ensure that product and process test standards and guidance are being met.*

1.5 Define and secure test budgets

Estimate test budgets and request funding sufficient to satisfy test resource requirements as defined, and continuously updated, in the TEMP. These estimates and funding requests should be refined as organizational component

---

[6]The quality assurance and the IV&V function will review the Year 2000 test efforts to ensure that they are complete and accurate and in conformance with the Year 2000 test and evaluation master plan.

test planning occurs and better information on test resource needs becomes available. Shortfalls in funding should be assessed for impact and reported to program management.

1.6 Establish new or augment existing test environments and schedule their use

Testing on the scale demanded by the Year 2000 program will likely require a larger, more robust testing environment than may currently exist. A test facility may be required that replicates the organization's system operating environments (or at least its mission-critical system operating environments) and simulates operating in a twenty first century setting (i.e., a Year 2000 time capsule). This includes simulating the century date transition (e.g., the time progression from midnight, December 31, 1999, to morning, January 1, 2000, the first business day in 2000, the end of the first month in 2000, etc.). In doing so, organizations should ensure that effective test configuration control is exercised to recognize that replicating a Year 2000 environment will require setting date clocks ahead for testing and later setting them back before the system is returned to a live production / operational environment. Doing this, however, introduces the possibility of "backwards discontinuity," meaning that advancing the date clock can cause certain software licenses or passwords to expire, and can cause certain files, which are programmed to automatically be purged or deleted on the basis of pre-set aging parameters, to be lost. The result can be a system that will not properly function when the date is set back.

If an organizational test facility already exists, its infrastructure and logistical capability and capacity relative to Year 2000 test requirements needs to be assessed and enhanced as necessary. Infrastructure items include test data, tools, hardware / software platforms, and support resources; logistical matters include staffing, hours of operation, and non-Year 2000 system test needs. Enhancements to an existing test facility to fulfill Year 2000 test requirements can range from adding a separate and dedicated Year 2000 test platform(s) to logically partitioning the existing test platform(s) between normal maintenance testing and Year 2000 peculiar testing. If an organizational test facility does not exist, such a facility should either be established in-house, or access to such a facility obtained, for example, by contracting for its use.

1.7 Develop and issue organizational Year 2000 test guidance

If organizational test guidance exists, augment this guidance as necessary to include Year 2000 activities. If such guidance does not exist, develop and issue it. In either case, ensure that the test guidance is understood by the Year 2000 test team (program level and project level).

The guidance should specify the objectives of Year 2000 testing (i.e., ensuring that the Year 2000 compliance criteria are satisfied) and the organization's

expectations relative to Year 2000 test coverage[7] and completeness (e.g., date-related conditions tested and types of tests conducted and documented) and other defined test metrics. It should also provide the rationale for all test scope and coverage decisions and disclose any operational risks associated with these decisions. Further, the guidance should describe each level of testing (software unit/module, software integration, system acceptance, and end-to-end), specify requirements for each level (e.g., requirements for planning and managing the testing and for documenting the results), and should encourage organizational components to build upon existing test plans, procedures, data, etc. to the maximum extent possible in performing this testing. The guidance should also specify the type of test result products expected, and should address such test preparation activities as baseline testing (to provide a baseline assessment of component and system performance before changes are made so that a basis for comparison exists, if one does not already exist), as well as such oversight and control activities as problem reporting and tracking, configuration management, and quality assurance/IV&V.

Additionally, the guidance should define organizational expectations (e.g., nature, scope, and timing) relative to testing back-up systems (or components) or alternative system configurations that are the foundation of the organization's contingency plans for continuity of operations in the event of Year 2000-induced system failures.

1.8 Establish processes and information sources to support testers and test activities

Inherent in any testing program are management processes and resources that facilitate test planning, execution, and reporting. If these processes and resources already exist, augment them as necessary to address Year 2000 nuances. If they do not exist, establish them. Examples of these include (1) configuration management processes, which provide the means for identifying, controlling, and reporting on changes to systems and system components, (2) risk management processes, which provide the means for proactively identifying and reporting test-related risks, devising risk mitigation strategies, and managing the implementation of these strategies, and (3) an intranet Year 2000 testing web site containing, for example, organizational Year 2000 test requirements, experience-based lessons learned, such as Year 2000 test pitfalls to avoid and best practices to emulate, and Internet linkages to other organizations Year 2000 testing Internet Web sites.

---

[7]Test coverage is a measure of the date-dependent software (e.g., lines of code, executable statements, objects, logic branches, logic paths, units of data) that has been tested in relation to the system's total date-dependent software.

1.9 Provide for ensuring Year 2000 compliance of vendor-supported products and services

Many system components that support an organization's core business areas and functions were not developed and are not maintained by the organization. Rather, these components have been purchased or are leased, under licensing agreements, from various vendors, who are responsible for their maintenance. The components include hardware, operating systems software and utilities, COTS application software, telecommunications equipment and lines, and embedded processors used, for example, in biomedical devices, avionics, and command and control systems.

For these system components, it is the organization's responsibility to obtain assurance that the vendors' products or services are already Year 2000 compliant or will be Year 2000 compliant in time to meet the organization's conversion needs. To do this, organizations should (1) develop a complete inventory of these products and services, (2) obtain vendor certification of its products' and services' Year 2000 compliance, (3) obtain vendor schedules for making noncompliant products and services compliant, (4) independently validate, as appropriate, the vendors' compliance certifications through, for example, independent testing, joint testing with the vendor, review and witnessing of vendor testing, or regulatory entity normal oversight and approval of vendor products (e.g., Food and Drug Administration regulation of biomedical devices), and (5) develop and implement plans for mitigating the impact of vendor products and services not being compliant on time.

1.10 Establish processes and metrics for reporting test activity and progress

Effective Year 2000 test management requires formal mechanisms for periodically reporting meaningful (i.e., timely and reliable) information about test planning and execution progress and results. In defining these mechanisms, it is important for test management to establish clear and timely standards relative to report content, format, and frequency so that testers can (1) construct their respective test activities to produce this information, (2) report consistent information over time (to facilitate trend analysis) and across test projects (to facilitate information synthesis and aggregation), and (3) have an effective avenue for surfacing test issues.

Central to an effective reporting process are well-defined metrics or measures that provide management with timely visibility into test planning, execution, and results. While the specific metrics that an organization selects and the frequency with which they are reported will vary among organizations, examples of such metrics include (1) test readiness indicators, such as whether test procedures, cases, and data sets have been prepared, (2) test schedule indicators, such as whether key milestones have been successfully met and whether any

milestones have slipped, (3) test coverage indicators, such as percentage of date-dependent logic paths tested, the percentage of date-dependent logic branch alternatives tested, or the percentage of date-dependent decision statements tested, (4) test completeness indicators, such as the nature and number of date-related conditions tested and types of tests (e.g., software unit, software integration, system functional, system performance, system security), and (5) test results indicators, such as number of defects discovered, categorized by severity, and their closure rate.

1.11 Establish a library of test tools

Identify, select, and acquire Year 2000 test tools to support testers and test activities. If a library already exists, augment it as needed with Year 2000-specific tools. If a library does not exist, establish one. Various categories of Year 2000 tools exist, some of which support multiple test phases or activities. These categories include (1) automated test tools for designing and developing test scripts, executing tests, and analyzing test results, (2) file and data management tools for gathering, customizing, reformatting, comparing, and aging test data, (3) software analysis and debugging tools for code analysis, test coverage analysis, date simulation, and code repair, (4) problem management and tracking tools for controlling the identification and resolution of defects, and (5) integrated toolsets for supporting various phases and activities in the Year 2000 conversion process, including testing.

The mix of tools selected will depend on, for example, the organization's systems environment, Year 2000 renovation technique, risk exposure, and budgets/schedules. In establishing a library, an organization should consider such questions as:

- *What software maintenance tools does the organization already possess?*
- *What programming languages must be supported?*
- *What platforms (systems environments) do the tools run on? For example, some tools are designed to analyze programs written in COBOL on a PC-based Windows NT or Windows 95 platform. For a mainframe-based COBOL application, this would require porting the application software to the Windows system to use the tool.*
- *What features and capabilities does the tool provide? For example, if a tool both locates date-related code and keeps track of test coverage, it can help ensure that all identified date-dependent code has been corrected and tested.*
- *Is the tool appropriate in light of the renovation approach selected? For example, when the windowing approach is used, program logic is added, increasing the need for coverage tools to ensure that all instances of added logic are tested.*

2.0 Software Unit Testing

The purpose of software unit testing is to verify that the smallest defined module of software (i.e., individual subprograms, subroutines, or procedures) work as intended. Unit tests are constructed to identify and facilitate correcting defects in modified or newly developed software units before they are integrated with other units. Unit testing is usually performed by the software engineer or programmer who created or modified the unit.

Because software units are not stand-alone entities but are written to interact with other units, testing these units individually requires construction of a simulated ambient environment. Stubs and drivers, which accept the units' output and provide it with expected inputs, are typically used to provide this environment. The stubs and drivers allow the unit to be executed (i.e., tested) independent of (and isolated from) the rest of the system.

If a library of appropriate stubs and drivers from past development or maintenance activity exists, then they should be made Year 2000 compliant and used to perform unit testing. If there are no appropriate stubs or drivers, it may be too great and time-consuming an effort to build them for Year 2000 software unit testing. In that case, alternative approaches, such as peer reviews of the software unit (a line-by-line inspection of the code by the programmer's peers) must be used. In any event, the project's quality assurance/IV&V group should ensure that every newly developed or modified unit has been appropriately tested.

Key Processes

2.1 Schedule and plan unit test
2.2 Prepare unit test procedures and data
2.3 Define unit test exit criteria
2.4 Execute unit tests
2.5 Document unit test results
2.6 Correct defects
2.7 Ensure that unit test exit criteria are met

2.1 Schedule and plan unit test

Time and resources (personnel, facilities, and tools) for conducting and documenting software unit testing should be defined and allocated, and project schedules and budgets should reflect these needs. Project-specific guidance specifying, for example, whether to conduct unit tests or peer reviews, what level of test/review coverage is necessary, how defects surfaced during test/review will be managed, how test/review results will be documented, etc. should be issued to software engineers and programmers responsible for unit testing/review. This guidance should also be reviewed and approved by the project's quality assurance/IV&V group.

2.2 Prepare unit test procedures and data

If effective unit test data and procedures and the associated baseline test results already exist from past development or maintenance activities, use these as a starting point and augment them with Year 2000-specific test cases, conditions, and expected outcomes. If they do not already exist, prepare them. The following are test scenarios that should be included to test correct date handling:

- *Current date: Tests to verify that the unit performs correctly through the remainder of the century. These should include, for example, tests to verify correct determination and use of the day of the week and day of the month.*

- *Year 2000 rollover: Tests to verify that the unit performs correctly when the year field changes from "99" to "00". These should include, for example, tests to verify the last day in 1999 (December 31, 1999), the first day in 2000 (January 1, 2000), the first business day in 2000 (Monday, January 3, 2000), the first 7 digit date field in 2000 (January 10, 2000), the last day of February 2000 (February 29, 2000), the first and second days following the last day in February 2000 (March 1 and 2, 2000), the end of the first month in 2000 (January 31, 2000), and the end of the first quarter in 2000 (March 31, 2000).*

- *Year 2001 rollover: Tests to verify that the unit functions correctly when the year field changes from "00" to "01" (see above). These should include, for example, tests to verify the correct determination and use of the day of the week and month.*

- *Leap year: Tests to verify that algorithms for performing leap year calculations perform correctly. These should include, for example: for the years 2000 and 2004, tests to verify correct determination and use of number of days in the year (366) and number of days in February (29), as well as the day of the week and month; for the years 2001 and 2002, tests to verify*

correct determination and use of number of days in the year (365) and number of days in February (28), as well as the day of the week and month.

- *Windowing[8] rollover: Tests to verify that the date years within the window relative to the current year, including the designated rollover year, are determined and used correctly. These should include, for example, tests to verify correct determination and use of the day of the week and month.*

- *Date calculations: Tests to verify that date-related mathematical functions are executed correctly, including multiyear projections forward and backward.*

- *Fiscal years: Tests to verify that the unit functions correctly when the fiscal year changes from "99" to "00" (e.g., fiscal year 2000 for the federal government begins October 1, 1999, and for many states it begins July 1, 1999). These should include, for example, tests to verify correct determination and use of the day of the week and month.*

- *Other: Tests to verify that other logical scenarios are executed correctly. These should include, for example, tests to ensure that division by "00" does not occur; that date sorting is performed correctly; and that special dates such as September 9, 1999 (when written in the Gregorian Calendar, 9999 is often assigned special meaning, such as invalid date or end of file), or April 9, 1999 (in the Julian Calendar the 99th day of 1999, or, once again, 9999), are processed correctly.*

If units are to be peer reviewed via code walk throughs or inspections, specify the peer review procedures to be followed, including the specific conditions to check for, and ensure that the procedures are understood by all reviewers. Provide peer reviewers with source code and documentation in advance of the review.

Regardless of whether unit tests or peer reviews are performed, ensure that the quality assurance/IV&V group reviews and approves the test/review procedures and witnesses the test/review.

---

[8]An approach to date conversion in which, rather than expanding date fields from two to four characters, software is written to associate a fixed or sliding period of years with either the twentieth or the twenty-first centuries. For example, software could be written to interpret the fixed period of years 00 through 25 as occurring in the twenty-first century (i.e., having a 20th century prefix), and the fixed period of years 26 through 99 as occurring in the twentieth century (i.e., having a 19th century prefix).

2.3 Define unit test exit criteria

Establish the conditions or requirements for successfully completing unit testing. These include error-free compilation of code, absence of defects with stated level of severity,[9] update of documentation (new or redlined version), sign-off by quality assurance/IV&V group, and acceptance by configuration management group into software library.

2.4 Execute unit tests

Run the unit test or conduct peer walkthroughs of the source code per the defined procedures and compare results to expected outcomes. Ensure quality assurance group/IV&V observation of the test/review.

2.5 Document test results

Document the results of the test/review in accordance with defined procedures. Documentation includes written trouble reports for all or selected defects, compiler listings, test/review summary reports, and quality assurance/IV&V sign-off.

2.6 Correct defects

Ensure that all software defects needed to exit unit testing are corrected (i.e., recoded, recompiled, retested, and documented in accordance with defined procedures) and that any new defects caused by these corrections are similarly addressed in accordance with test exit criteria.

2.7 Ensure that test exit criteria are met

Compare test results to test exit criteria and ensure that specified conditions are met. Lesser severity defects (e.g., documentation) should be documented and managed in accordance with software problem reporting and tracking

---

[9]A typical defect severity scale might be: *emergency, test critical, high, medium*, and *low*. Using the scale *emergency* could mean, for example, that testing must stop and an immediate fix or work around the problem must be implemented before restarting the test; *test critical* could mean that the problem impedes test progress and must be resolved prior to the next scheduled accumulation and reporting of defects; *high* could mean a problem that must be corrected before a test activity is completed; *medium* could mean a significant software or system problem that does not require resolution in order to complete the test activity; and *low* could mean a minor or insignificant problem that does not require resolution in order to complete the test activity.

procedures to ensure later correction. Obtain quality assurance/IV&V sign-off on unit test criteria satisfaction and configuration management sign-off of acceptance into software library.

3.0 Software Integration Testing

The purpose of software integration testing is to verify that units of software, when combined, work together as intended. Typically, a number of software units are integrated or linked together to form an application. Multiple applications often work together to accomplish a system function. Because the units being integrated have already been tested successfully, integration testing focuses on ensuring that the interfaces work correctly and that the integrated software meets specified requirements.

Integration testing is generally performed by a test team, but may also be performed by the senior programmer(s) who oversaw the development/maintenance of the individual units being integrated.

Key Processes

3.1 Schedule and plan software integration test
3.2 Prepare software integration test procedures and data
3.3 Define software integration test exit criteria
3.4 Execute software integration tests
3.5 Document software integration test results
3.6 Correct defects
3.7 Ensure that software integration test exit criteria are met

3.1 Schedule and plan software integration test

Time and resources (personnel, facilities, and tools) for conducting and documenting software integration testing should be defined, and project schedules and budgets should reflect these needs. Project-specific guidance specifying, for example, what level of test coverage is necessary, how defects surfaced during testing will be managed, how test results will be documented, etc. should be issued to the integration test team. This guidance should also be reviewed and approved by the project's quality assurance / IV&V group.

3.2 Prepare software integration test procedures and data

If there is an existing suite of integration tests and the associated baseline test results, start with these. In light of Year 2000 compliance criteria (see key process 1.2 in this guide), augment the existing integration test suite with Year 2000-specific test cases and data that verify the correct handling of dates. In doing so, the following steps may be necessary to prepare specific data sets:

- *Augment the dates in the existing test data to include all dates of interest (see key process 2.2 of this guide for these dates). This is referred to as "data aging."*
- *If existing test data are not date-dependent, add new test data to exercise the above-mentioned dates of interest.*

If no integration test suite already exists, the following steps should be performed:

- *Analyze application/functional requirements and develop corresponding test cases, data, and expected outcomes covering dates of interest (see key process 2.2).*
- *Ensure that test cases exercise interfaces among software components (e.g., units) in accordance with test exit criteria (see key process 3.3).*
- *Develop procedures for executing the test cases, documenting the results, and correcting errors.*
- *Trace test procedures, cases, and expected outcomes to application/functional requirements.*

Ensure that the quality assurance/IV&V group reviews and approves the test procedures and data.

3.3 Define software integration test exit criteria

If already available, use and, if necessary, augment existing conditions and requirements for successful completion of software integration tests. If not available define these criteria (see key process 2.3 for examples).

3.4 Execute software integration tests

Run the integration tests according to defined procedures and compare results to expected outcomes. Document and assess any deviations from expected outcomes for severity and cause. Ensure that the quality assurance/IV&V group observes the tests.

3.5 Document software integration test results

Test results should be documented so that the information can be used to (1) validate that test exit criteria have been met and (2) assist in assessing and correcting software defects discovered during testing. If the exit criteria uses, for example, number and severity of software trouble reports (STR), then the test documentation should address these criteria. Assuming that the exit criteria are based on some type of STR count by severity, the test documentation should include:

- *Test logs or records showing test activities performed, when, and by whom.*
- *Record of the results of each test case/procedure executed, including a pass/fail determination and a test results summary report signed-off on by both test team and quality assurance/IV&V.*
- *Written STR for each failed test case/procedure.*
- *Severity designation for each STR.*
- *Sign-off by the project manager's designee (e.g., configuration or change control body representative) indicating STR placed under configuration control and tracking.*

3.6 Correct defects

On the basis of defect severity and test exit criteria, prioritize defects for corrective action (cause analysis, fix determination, recoding, retesting, and redocumenting results). Residual defects (unrepaired and new caused by coding changes) should be documented and addressed iteratively until test exit criteria are met.

3.7 Ensure that software integration test exit criteria are met

Compare test results to test exit criteria and ensure that specified conditions are met. Lesser severity defects should be documented and managed in accordance with software problem reporting and tracking procedures to ensure later correction. Obtain quality assurance/IV&V sign-off on software integration test criteria satisfaction and configuration management sign-off on acceptance of software application/function into software library.

4.0 System Acceptance Testing

The purpose of system acceptance testing is to verify that the complete system (i.e., the full complement of application software running on the target hardware and systems software infrastructure) satisfies specified requirements (functional, performance, and security) and is acceptable to end users. System acceptance tests are a set of specialized tests run either separately or in some combination in an operational environment (either actual[10] or simulated). Collectively, these tests verify that the entire system performs as intended. The set of tests can include:

- Functional testing: The purpose of functional or "black box" testing is to verify that the system correctly performs specified functions. As such, the test team's ability to design the functional tests is limited by the completeness and precision of the functional specifications. Starting with these specified functions, the test team develops test cases using a range of valid input conditions and options as well as invalid or unexpected inputs. The test team then compares the test outputs to expected outputs.

- Performance testing: The purpose of performance testing is to assess how well a system meets specified performance requirements. Like functional testing, performance testing can only be as complete and precise as the system's defined performance requirements. Examples of performance requirements include specified system response times under normal workloads (e.g., defined transaction volumes) and specified levels of system availability and mean-times-to-repair.

- Regression testing: The purpose of regression testing is to demonstrate that newly added or modified system components (hardware or software) have not compromised system functionality and performance (i.e., have not introduced new errors).

- Stress testing: The purpose of stress testing is to analyze system behavior under increasingly heavy workloads (e.g., higher transaction rates) and severe operating conditions (e.g., high error rates, lower component availability rates), and, in particular, to identify points of system failure.

- Security testing: The purpose of security testing is to assess the robustness of the system's security capabilities (e.g., physical facilities, procedures, hardware, software, and communications), and to identify security vulnerabilities.

---

[10]Risks of testing in the production environment must be thoroughly analyzed and precautions taken to preclude damage to systems and data.

Integral to all aspects of system acceptance testing is participation by the project's quality assurance/IV&V function and user groups.

Key Processes

4.1 Schedule and plan system acceptance test
4.2 Prepare system acceptance test procedures and data
4.3 Define system acceptance test exit criteria
4.4 Confirm Year 2000 compliance of vendor-supported system components
4.5 Execute system acceptance tests
4.6 Document system acceptance test results
4.7 Correct defects
4.8 Ensure that system acceptance test exit criteria are met

4.1 Schedule and plan system acceptance tests

Plans specifying the scope of the project's system acceptance testing should be established, including the type and combination of specialized testing to occur, and whether the users will actively participate or just witness the tests. Time and resources (personnel, facilities, and tools) for conducting and documenting system acceptance testing should be defined in the plans, and project schedules and budgets should reflect these needs. Project-specific guidance should be issued to the project team and the test team specifying, for example, (1) who should witness what tests, (2) who should develop and approve test plans, (3) who should design and approve test cases and procedures, (4) how defects surfaced during the tests will be managed, including their documentation in trouble reports, etc. The plans and guidance should also be reviewed and approved by the project's quality assurance and user groups.

4.2 Prepare system acceptance test procedures and data

If there is an existing suite of system acceptance tests (e.g., functional, performance, regression, stress, security) and the associated baseline test results, begin with these. Using established Year 2000 compliance criteria (see key process 1.2 in this guide for examples of these criteria), augment these plans, procedures, and data with steps, cases, and input conditions that verify the

correct handling of dates. In doing so, the following steps may be necessary to prepare specific data sets:

- *Augment the dates in the existing test data to include all dates of interest (see key process 2.2 of this guide for these dates). This is referred to as "data aging."*
- *If existing test data are not date-dependent, add new test data to exercise the above-mentioned dates of interest.*

If no system acceptance test suite already exists, the following steps should be performed:

- *Analyze system functional, performance, workload, and security requirements and develop corresponding test plans (procedures, cases, inputs, and expected outputs) covering dates of interest (see key process 2.2 for these dates).*
- *Ensure that test coverage is in accordance with test exit criteria (see key process 4.3). In doing so, trace test cases, inputs, and expected outputs to corresponding system requirements to determine test coverage.*
- *Develop procedures for executing the tests, documenting the results, and correcting errors.*

Ensure that the quality assurance/IV&V and user groups review and approve the test procedures and data.

4.3 Define system acceptance test exit criteria

Establish the conditions or requirements for successfully completing system acceptance testing. If test conditions and requirements are already available, use and, if necessary, augment them. If they do not already exist, define them (see key process 2.3 for Year 2000-specific examples).

4.4 Confirm Year 2000 compliance of vendor-supported system components

In order to execute system acceptance tests and ensure system Year 2000 compliance, all system components (application software, systems software, hardware, firmware, and communications networks) must be compliant and ready for testing as an integrated system. Ensuring that the vendor-supported components of the system are compliant is a process that should have begun as part of establishing the test infrastructure (see key process 1.9 in this guide). By now this process should have been completed for each system component.

4.5 Execute system acceptance tests

Execute the system acceptance tests in accordance with plans and procedures. Ensure that representatives for the quality assurance and user groups participate, as planned, in the tests. Adhere to configuration management and risk management process requirements.

4.6 Document system acceptance test results

System acceptance tests results should be fully documented so that the information can be used to (1) validate that test exit criteria have been met and (2) assist in assessing and correcting software defects discovered during testing. Examples of test results documentation are described under key process 3.5 of this guide. Additional test results documentation for system acceptance testing also includes management summary reports. The quality assurance/IV&V group should also ensure that test documentation requirements are met.

4.7 Correct defects

On the basis of defect severity and test exit criteria, prioritize defects for corrective action (cause analysis, fix determination, recoding, component replacement, retesting, and redocumenting). Residual defects (unrepaired or newly discovered problems) should be documented and addressed iteratively until test exit criteria are met.

4.8 Ensure that system acceptance test exit criteria are met

Compare test results to test exit criteria and ensure that specified conditions are met. Lesser severity defects should be documented and managed in accordance with problem reporting and tracking procedures to ensure later correction. Satisfaction of the test exit criteria should be acknowledged in writing by the test, quality assurance/IV&V, user groups, and should be reported to management.

5.0 End-to-End Testing

The purpose of end-to-end testing is to verify that a defined set of interrelated systems, which collectively support an organizational core business area or function, interoperate as intended in an operational environment (either actual[11] or simulated). These interrelated systems include not only those owned and managed by the organization, but also the external systems with which they interface. For example, since agencies that administer key federal benefits payment programs, such as the Department of Veterans Affairs, exchange data with the Department of the Treasury which, in turn, interfaces with various financial institutions to ensure that benefit checks are issued, end-to-end testing of the federal benefits payment function would include systems for all entities involved, as well as their supporting telecommunications infrastructures.

Generally, end-to-end testing is conducted when one major system in the end-to-end chain is modified or replaced, and attention is rightfully focused on the changed or new system. In the case of Year 2000 testing, however, most if not all of the systems in the end-to-end chain will have been modified or replaced. As a result, the scope and complexity of the testing is dramatically increased, as is the difficulty of isolating, identifying, and correcting problems.

The boundaries on end-to-end tests are not fixed or predetermined, but rather vary depending on a given business area's system dependencies (internal and external) and criticality to the mission of the organization. Therefore, in planning end-to-end tests, it is critical to analyze the organization's core business functions, the interrelationships among systems supporting these functions, and potential risk exposure due to date-induced system failure(s) of any system in the chain of support. It is also important to work early and continuously with the organization's data exchange partners so that end-to-end tests can be effectively planned and executed.

[11]Risks of testing in the production environment must be thoroughly analyzed and precautions taken to preclude damage to systems and data.

Key Processes

5.1 Define the system boundaries of the end-to-end test(s)
5.2 Secure the commitment of key data exchange partners
5.3 Establish an interorganizational end-to-end test team
5.4 Confirm Year 2000 compliance of vendor-supported telecommunications and other infrastructure(s)
5.5 Schedule and plan the end-to-end test(s)
5.6 Prepare end-to-end test procedures and data
5.7 Define end-to-end test exit criteria
5.8 Execute end-to-end test(s)
5.9 Document end-to-end test results
5.10 Correct defects
5.11 Ensure that end-to-end test exit criteria are met

5.1 Define the system boundaries of the end-to-end test(s)

The business impact of Year 2000-induced system failures should drive organizational decisions about the nature and scope of end-to-end tests. Business impact is a function of both business priorities, which should have been established early in the Year 2000 conversion process, and the level of business risk an organization is willing to assume by foregoing, or limiting, such testing. Organizations therefore need to assess their mission-critical business functions in light of inter- and intraorganization system dependencies, as well as the probabilities and impacts of any of these systems suffering a date-related failure. On the basis of this assessment, system boundaries for end-to-end tests can be defined.

5.2 Secure the commitment of key data exchange partners

End-to-end testing addresses business areas or functions that involve multiple organizations (internal and external). Participation by all key business area date exchange partners should therefore be solicited. Executive-level commitments to participate in the end-to-end test(s) should be secured.

5.3 Establish an interorganizational end-to-end test team

A team composed of representatives from each of the organizations participating in the end-to-end test should be formed to manage the planning, execution, and reporting of the test. Team leadership should be established, and a team

charter, specifying how the team will carry out its responsibilities, should be agreed to by all representatives and approved by the organizations' responsible business area executive(s).

5.4 Confirm Year 2000 compliance of vendor-supported telecommunications and other infrastructure(s)

In order to execute end-to-end testing and ensure that all systems in the chain of support to core business areas function as intended, the telecommunications infrastructure that interconnects the systems must be compliant and ready for testing. Ensuring that the vendor-supported components of the telecommunications infrastructure are compliant is a process that should have begun as part of establishing the test infrastructure (see key process 1.9 in this guide). By now this process should have been completed for all telecommunications systems. Confirmation that it has should be obtained. Confirmation on Year 2000 compliance should also be obtained from other infrastructure service providers, such as power and water companies, as these services are critical to carrying out business operations.

5.5 Schedule and plan the end-to-end test(s)

An end-to-end test plan should be developed and agreed to by team representatives. Key tasks and requirements in preparing for, executing, and documenting the results of testing should be defined, the milestones and resources (personnel, facilities, and tools) associated with performing these tasks should be established, and organizational funding commitments should be secured. Responsibility for performing key tasks needs to be assigned, and interorganizational processes supporting end-to-end testing, such as quality assurance/IV&V, need to be established.

5.6 Prepare end-to-end test procedures and data

Interorganizational test procedures and data need to be prepared and approved by team representatives. In developing these procedures and data, participating organizations' existing system acceptance test scenarios and data may be a helpful starting point. These scenarios and data should include steps, cases, and input conditions that verify the correct handling of dates of interest to each organization (see key process 2.2 of this guide for these dates).

5.7 Define end-to-end test exit criteria

The conditions or requirements for successfully completing end-to-end testing need to be established and agreed to by team representatives. In establishing these criteria, participating organizations' existing system acceptance test exit criteria provide a useful starting point. These criteria should include the kind

of Year 2000-specific conditions and requirements discussed under key process 2.3 of this guide.

5.8 Execute end-to-end test(s)

Execute end-to-end test(s) in accordance with established plans and procedures. Adhere to established interorganizational test management processes.

5.9 Document end-to-end test results

End-to-end test results should be documented in accordance with established plans and procedures so that the information can be used to (1) validate that test exit criteria have been met and (2) assist in assessing and correcting problems discovered during testing. Examples of test results documentation are described under key process 3.5 of this guide. Additional test results documentation for end-to-end testing can also include results and corrective action summary reports for the participating organization's responsible business area executive(s).

5.10 Correct defects

On the basis of interorganizational specified criteria, such as defect severity and test exit criteria, prioritize defects for corrective action (fault isolation, cause analysis, fix determination, recoding, component replacement, retesting, and redocumenting). Residual defects (unrepaired or newly discovered problems) should be documented and addressed iteratively in accordance with established procedures until test exit criteria are met.

5.11 Ensure that end-to-end test exit criteria are met

Compare test results to test exit criteria and ensure that specified conditions are met. Lesser severity defects should be documented and managed in accordance with interorganizational processes and procedures to ensure later correction. Satisfaction of the test exit criteria should be acknowledged by all test team representatives, and should be reported to each participating organization's responsible business area executive(s).

6.0 Management Oversight and Control

The purpose of test management oversight and control is to ensure that established test requirements are being met and to remedy any situations or circumstances where they are not. These test requirements, which were established in the policies, guidance, standards, plans, and budgets discussed in the first section of this guide, in combination with the project-level plans and budgets discussed in the subsequent four sections of the guide, provide the criteria against which to evaluate actual test activity. Test oversight entails collecting and assessing status and progress reports to determine, for example, whether specific test activities are on schedule and within budget and whether defect discovery and closure rates are consistent with projections and indicative of a maturing system renovation or new development project.

To augment these test activity results, Year 2000 program and project management can use periodic (weekly or monthly) reports from the independent quality assurance or IV&V group concerning projects' adherence to test requirements and reports from user groups relative to test progress and results. Using this information, management can effectively control testing activities by determining whether corrective action is needed, and if so, what action should be taken. Such actions can include infusion of additional resources (people, tools, or money), establishment of system renovation or replacement contingency plans, and reprioritization of competing test projects and activities.

Key Processes

6.1 Ensure that established test activity and progress reporting requirements are met
6.2 Solicit reports from the quality assurance/IV&V and user groups
6.3 Identify and assess deviations from plans
6.4 Take appropriate action to address deviations

6.1 Ensure that established test activity and progress reporting requirements are met

The reporting requirements defined and implemented as part of establishing the Year 2000 test infrastructure (see key process 1.10), as well as the associated project-level test objectives and plans, provide program and project management with the means through which to collect insightful information on test activity and progress. These requirements need to be enforced for management to effectively oversee Year 2000 testing.

6.2 Solicit reports from the quality assurance/IV&V and user groups

As discussed in each of the test activity levels in this guide, quality assurance/IV&V is an integral component of an effective Year 2000 program. If properly involved, the quality assurance/IV&V function will collect definitive information about test activity satisfaction of established test objectives, milestones, and requirements. Additionally, both the quality assurance/IV&V function and the users groups that are working with the system testers will possess a great deal of knowledge about test progress and results. In overseeing test activities and progress, management should augment its test reports with frequent reports from these entities.

6.3 Identify and assess deviations from plans

Using established reporting mechanisms, management can effectively control Year 2000 testing. First, management should analyze the reported information to pinpoint the causes of (1) unsatisfied test objectives and requirements, and missed milestones, (2) unfavorable component and system test results, and (3) identified test problems and risks. Then, management should prioritize the results in light of business needs and risks and develop a corrective action agenda.

6.4 Take appropriate action to address deviations

Management should take quick and decisive actions to correct problems in light of established priorities. Examples of actions include infusion of additional resources (people, tools, or money), establishment of system renovation or replacement contingency plans, and reprioritization of competing test projects and activities.

Year 2000 Computing Crisis: A Testing Guide Checklist

❒ Testing Infrastructure

❒ Software Unit Testing

❒ Software Integration Testing

❒ System Acceptance Testing

❒ End-to-End Testing

❒ Management Oversight & Control

Testing Infrastructure

❒ Has Year 2000 test management authority, responsibility, and accountability been assigned?

Has it been assigned at both the program and project levels?
Has it been assigned for each level of testing (unit, integration, acceptance, and end-to-end)?

❒ Has Year 2000 compliance criteria been defined?

Is the compliance criteria documented?
Has the compliance criteria been distributed?
Is the compliance criteria the basis for test plans?

❒ Has an organizational Year 2000 TEMP been developed?

Has the TEMP been distributed?
Is there a process to update the TEMP?
Does the TEMP describe test roles and responsibilities, system/project priorities, test resource needs, individual project test schedules, and progress metrics?

❒ Has the organization defined the roles and responsibilities for the quality assurance or IV&V groups?

Does this quality assurance or IV&V group have a reporting chain to senior management?

- ❒ Has the organization estimated test budgets and allocated resources and funding for the test activities?

  Are shortfalls in funding assessed for impact and reported to management?

- ❒ Have the test environments been updated to allow Year 2000 tests?

  Have one or more test facilities been established that replicate the operating environment(s)?
  Have the facilities' infrastructure and logistical capabilities been assessed and augmented?

- ❒ Has the organization developed and issued organizational Year 2000 test guidance?

  Does the guidance define the objectives of Year 2000 testing?
  Does the guidance define the types of testing expected?
  Does the guidance define the progress metrics that are to be reported?

- ❒ Has the organization established test management processes and information sources?

  Have configuration management processes been defined?
  Have quality assurance processes been defined?
  Has a change control process been defined?
  Has a risk management process been defined?
  Has a central library of test information been established?

- ❒ Has the organization ensured that vendor-supported (COTS) products are compliant?

  Has an inventory of COTS products been established?
  Have vendor certifications of its COTS products' compliance been obtained?
  Have steps been taken to validate vendors' claims?

- ❒ Has the organization defined the test metrics that will be reported?

  Has the report format been defined?
  Has the frequency of reporting been determined?
  Have measures of test progress and results been established?

- ❒ Has the organization established a library of support tools?

  Have test tools needs been defined?
  Has the adequacy of existing tools been assessed?
  Have new tools been selected?

Have the tools' acquisition been coordinated across the organization?

Software Unit Testing

- ❒ Have unit test activities been planned and scheduled and is the quality assurance or IV&V group involved in each phase of unit testing?

  Will peer reviews be used in lieu of unit tests?

- ❒ Have unit test procedures and data been generated?

  Do the test procedures address relevant date conditions?

- ❒ Have the exit criteria for unit tests been defined?

- ❒ Have unit test or peer reviews been conducted?

- ❒ Have unit test or peer review results been documented?

- ❒ Have defects identified during unit test or peer reviews been corrected?

- ❒ Have the unit test exit criteria been satisfied?

Software Integration Testing

- ❒ Have integration test activities been planned and scheduled and is the quality assurance or IV&V group involved in each phase of integration testing?

- ❒ Have integration test procedures and data been generated?

  Do the test procedures address relevant date conditions?

- ❒ Have the exit criteria for integration tests been defined?

- ❒ Have integration tests been conducted?

- ❒ Have integration test results been documented?

- ❒ Have defects identified during integration tests been corrected?

- ❒ Have the integration test exit criteria been satisfied?

System Acceptance Testing

- ❒ Have acceptance test activities been planned and scheduled, and is the quality assurance or IV&V group involved in each phase of system acceptance testing?

  Do acceptance tests include functional, performance, regression, stress, and security testing?

- ❒ Have acceptance test procedures and data been generated?

  Do the test procedures address relevant date conditions?

- ❒ Have the exit criteria for acceptance tests been defined?

- ❒ Have compliant vendor-supported systems (COTS) been acquired and installed?

- ❒ Have acceptance tests been conducted?

- ❒ Have acceptance test results been documented?

- ❒ Have defects identified during acceptance tests been corrected?

- ❒ Have the acceptance test exit criteria been satisfied?

End-to-End Testing

- ❒ Have the system boundaries for end-to-end testing been determined?

  Have mission-critical business functions been identified?
  Have systems (internal and external) supporting these mission-critical business functions and the systems interrelationships been identified?
  Have the probabilities of the systems in the chain suffering a Year 2000-induced failure been assessed?

- ❒ Have relevant data exchange partners committed to participating in end-to-end testing?

- ❒ Has an interorganization end-to-end test team been established?

- ❒ Has the telecommunications infrastructure been confirmed as Year 2000 compliant?

- ❒ Has the end-to-end testing been planned and scheduled?
- ❒ Have end-to-end test procedures been generated?
- ❒ Have end-to-end test exit criteria been defined?
- ❒ Have end-to-end tests been conducted?
- ❒ Have end-to-end test results been documented?
- ❒ Have defects identified during end-to-end tests been corrected?
- ❒ Have the end-to-end test exit criteria been satisfied?

Management Oversight and Control

- ❒ Has the agency ensured that test activity and progress reporting requirements have been met?

  Are the projects reporting test progress and activity in accordance with defined requirements?
  Are reporting requirements being enforced?
  Are reports from quality assurance, IV&V, and users groups being used?

- ❒ Has the agency identified deviations from requirements?
- ❒ Has the agency taken appropriate action to address deviations, problems, and risks?

(511134)

Ordering Information

The first copy of each GAO report and testimony is free. Additional copies are $2 each. Orders should be sent to the following address, accompanied by a check or money order made out to the Superintendent of Documents, when necessary. VISA and MasterCard credit cards are accepted, also. Orders for 100 or more copies to be mailed to a single address are discounted 25 percent.

Orders by mail:

U.S. General Accounting Office
P.O. Box 37050
Washington, DC 20013

or visit:

Room 1100
700 4th St. NW (corner of 4th and G Sts. NW)
U.S. General Accounting Office
Washington, DC

Orders may also be placed by calling (202) 512-6000 or by using fax number (202) 512-6061, or TDD (202) 512-2537.

Each day, GAO issues a list of newly available reports and testimony. To receive facsimile copies of the daily list or any list from the past 30 days, please call (202) 512-6000 using a touchtone phone. A recorded menu will provide information on how to obtain these lists.

For information on how to access GAO reports on the INTERNET, send an e-mail message with "info" in the body to:

info@www.gao.gov

or visit GAO's World Wide Web Home Page at:

http://www.gao.gov

United States
General Accounting Office
Washington, D.C. 20548-0001

Bulk Rate
Postage & Fees Paid
GAO
Permit No. G100

Official Business
Penalty for Private Use $300

Address Correction Requested