COUNTRYWIDE HOME LOANS

Y2K CERTIFICATION PROJECT

Creating A Test Plan

TRAINING
WORKBOOK
FOR
TEST PLAN AUTHORS

Countrywide
HOME LOANS

Table of Contents

I. Introduction – Importance & Purpose of the Y2K Certification Test Plan 4
A. Purpose of the Course 4
B. Conventions used in this book 4
C. Materials 4
D. Importance of the Y2K Test Plan 5
E. Clarity + Full Disclosure + Efficiency 6
F. Roles in the Y2K Certification Process 8
1. Application Owners Roles: 8
2. Y2K Client Server Team Roles 8
3. The Test Plan Walk-through 10
4. Test Plan Approval Milestone 10
G. The 6 Sections of the Test Plan 10
H. Test Plan Continuity 11
I. Cover Page 12
J. Example Cover from SurveyPower 14
II. Description 15
A. Description Template 15
B. Objectives Of This Section 15
C. Summarize The Application's Business Purpose 16
D. Describe the Application's Y2K Readiness 17
E. Breakdown the Application Into It's Major Business Functions 18
F. List Application's Technical Component Inventory 19
1. Hardware Components and Software Components 19
2. Date Processing Components 20
G. Interface Inventory 22
1. The Input Interfaces Grid 23
2. The Output Interfaces Grid 24
H. Application Date Processing Flow Diagram 25
I. Application Architecture Diagram 26
J. Summary Questions 27
K. Example from SurveyPower 27
III. Test Scope 34
A. Test Scope Section Template 34
B. Objectives Of This Section 34
C. Action List – Functions and Processes To Be Tested 34
D. The Assumptions List 37
1. Technical Components NOT To Be Tested and Why 37
2. Other Assumptions 38
3. Functional Assumptions 39
E. Summary Questions 40
F. Example From SurveyPower 40
IV. Test Details 43
A. Managing Test Data Template 43
B. Objectives Of This Section 43
C. Overview 43
D. Test Conditions 43

E. TEST CYCLES 45
1. "Hot Dates" Dates to consider for Y2K Business Unit Preparedness *46*
F. TEST SCRIPTS 48
G. SUMMARY QUESTIONS 55
H. . EXAMPLE FROM SURVEYPOWER 55
V. Managing Test Data 67
A. MANAGING TEST DATA TEMPLATE 67
B. OBJECTIVES OF THIS SECTION 67
C. INPUT – DATA PREPARED FOR TEST 67
1. Test Data Preparation Strategy *67*
2. File Naming Convention *69*
D. OUTPUT – VALIDATION SNAPSHOT FILES 70
1. Validation Archive Snapshot Directory *70*
2. Recognized Output File Types *70*
E. SUMMARY QUESTIONS 72
F. EXAMPLE FROM SURVEYPOWER 72
VI. Test Environment 74
A. TEST ENVIRONMENT SECTION TEMPLATE 74
B. OBJECTIVES OF THIS SECTION 74
C. TEST ENVIRONMENT CLIENT-SIDE 75
1. Client Hardware Specification *75*
2. Client Hardware Configuration *75*
3. Client Software Specification *75*
4. Client Software Configuration *76*
D. TEST ENVIRONMENT SOFTWARE 76
1. Server(s) Software Specification *76*
2. Server(s) Software Configuration *77*
3. Server(s) Hardware Specification *77*
4. Server(s) Hardware Configuration *77*
E. OTHER HARDWARE/SOFTWARE/INTERFACES NEEDED 77
F. APPLICATION INSTALLATION 78
G. SUMMARY QUESTIONS 79
H. EXAMPLE FROM SURVEYPOWER 79
VII. Executive Summary 84
A. OBJECTIVES OF THIS SECTION 84
B. SUMMARIZING THE TEST PLAN 84
C. SUMMARY QUESTIONS 87
D. EXAMPLE FROM SURVEYPOWER 87
VIII. In Closing… 89
IX. Resources 90
A. YEAR 2000 CERTIFICATION TEST PLAN OUTLINE 91

I. Introduction – Importance & Purpose of the Y2K Certification Test Plan

A. Purpose of the Course

This training course in creating a Y2K Test Plan has several purposes.

1. At the end of the course, to have created the skeleton of your Y2K Test Plan in a workshop environment with the application you are familiar with.

2. To impress upon all concerned parties how important a credible test plan is to the entire Y2K certification process.

2. To eliminate the many authoring hours that go into developing a test plan by creating a Template for this specific test plan.

3. To create a standardized test plan style to improve the speed and accuracy of test plan review and auditing.

4. To convey to the test plan author the environment in which this test plan will be scrutinized, implemented, and archived.

5. To provide a forum for authoring questions and a workshop to actually begin the process of creating a Y2K test plan.

B. Conventions used in this book

Not listed

C. Materials

Sign-in sheet – Please fill out the Sign-in sheet so we can keep track of who has attended the Create A Test Plan Workshop. Thank you.

- **Name Card** – If a Name Card is present in your materials, please fill it out and place it near your workspace so that all workshop members can address each other familiarly.

- **Workbook** – This Workbook contains all of the information and work sheets necessary to create a Y2K Test Plan for Countrywide Home Loans applications. Starting in the

Description module, we will begin building the skeleton of your plan. Don't worry about it being perfect, the point is to build the Test Plan using the concepts presented, with information that you are familiar with.

- **PowerPoint Presentation** – A slide presentation highlighting principles presented in the Workbook may accompany and facilitate the workshop.

D. IMPORTANCE OF THE Y2K TEST PLAN

We all realize that there is significant concern about the ability of computer hardware and software to operate after the calendar changes to the year 2000. Countrywide is making a priority of assuring that its proprietary computer software applications will operate successfully in this new millennium. Much of the conversion of software code, hardware, and internal interfaces between CHL systems has already taken place and all are on schedule to finish this process well before January 1, 2000.

However, to complicate matters, there is possible liability for failing to meet this requirement so not only must Countrywide's proprietary applications pass tests to assure all concerned that these applications will operate correctly in the new century but the documentation of the process must be meticulous and clear. There is a possibility, however remote, that several years from now that there may be a need to completely re-create the testing process that validated an application's Y2K certification status.

At the heart of this matter is the Y2K Certification Test Plan. It is not only a plan for testing the application's operation but at least as important, it is a legal record, documenting the concept, plan and implementation of the Y2K testing process. This document will be archived with several other files including the actual certification document and all of the validating test results. The test plan is the key to the entire process and that is why so much of the Certification process concerns itself with proper test plan creation. Note Figure 1, the Y2K Certification Road Map.

It has been made clear to the Y2K Certification Group that there will be a concerted effort to continue certifying CHL applications as new releases of applications are released. A good test plan at this point will insure a very easy transition into certifying application production releases prior to the year 2000.

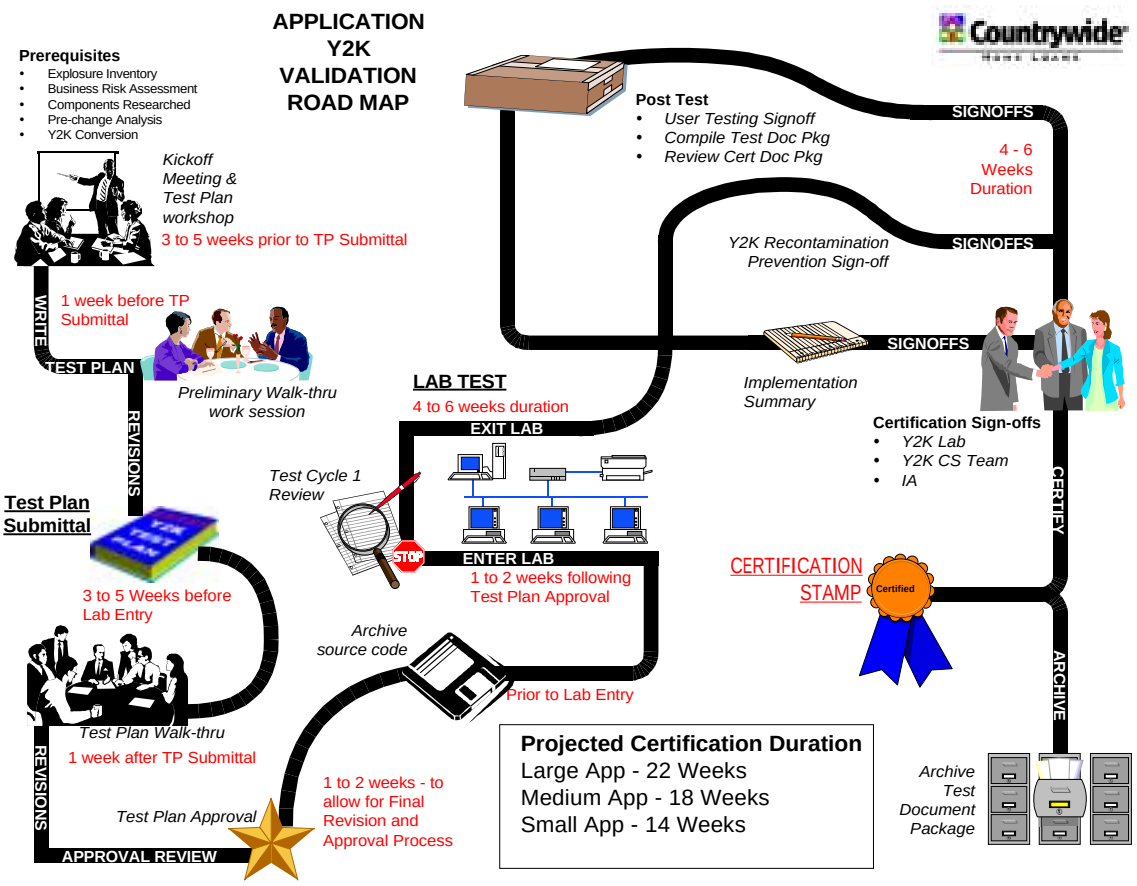


Figure 1

E. CLARITY + FULL DISCLOSURE + EFFICIENCY

There are three key principles to creating a successful Y2K Test Plan.

CLARITY – A test plan that is not logical and difficult to understand creates problems on every level. It will certainly lead to countless revisions until it reaches the point of comprehension by the reviewers and 3rd party auditors. Since the test plan may be used in the future as legal documentation of the testing process, there is a possible liability issue if the plan does not clearly describe and fully disclose the application, testing strategy, and testing process

Since this document may be exhumed from its archive at some future date and it may be necessary for a reader unfamiliar with the application's operation, and quite possible to CHL's systems, to become conversant quickly. It is not impossible that at some future date, it might be necessary to recreate the entire testing. Avoid acronyms and names that are Countrywide specific, unless you define what they mean. You can assume that your reader is intelligent but do not assume that they

necessarily know anything about your application or CHL procedures. Define and explain terms that are industry specific, CHL specific, or too technically specific. This will help clarity greatly. ***The most important attribute of the Y2K Test Plan is that people not familiar with your application can easily comprehend it.***

REQUIRED DISCLOSURE – It is important that the test plan detail all components of the application whether they are to be tested or not. One reason this is important is that by inventorying all of the components it is much easier to structure your test plan by indicating what you will be testing and what you won't be testing. This helps create Clarity in your test plan.

Another reason is your test plan will not leave itself open to question in regard to items unmentioned or left hanging. Since it is the job of the reviewers and 3rd party auditors to question everything, holes in the logic of the plan created by leaving components out or glossing over areas will lead to questions about how thoroughly you have thought through your plan. And this will lead to more revisions – and more questions since the reviewers then start to believe there is more for them uncover.

And there is the question of any future reference to this document for legal issues. The simple rule is: ***if you have to explain it in words during your walk-throughs, then it should have been written down somewhere in the test plan.***

A final point regarding disclosure. Some times test plan authors or business or MIS owners feel that there are sections of their application that just do not need to be looked at in the test plan. There may be a number of reasons for this, many of them having nothing to do with Y2K operations. However, since you and your other representatives of this application, as noted in the AMS Y2K database, are responsible for the authoring and implementation of the test, no one will be forcing you to test anything. The responsibility for the application's capability to operate in the year 2000 belongs to the application's business and MIS owners. So, if there are issues about parts of the application for which you have a reason to not test, then simply give that reason. The reviewers and auditors are looking mostly for the things you don't mention to make sure that you have considered them.

EFFICIENCY - Your Y2K Test Plan should be a model of efficiency. Once you have identified all of the application's pieces, the focus should shift only to those components that are effected by dates. Any words or sections that do not directly relate to this issue are unnecessary and only blur the test plan picture. ***A test plan that has gotten too long has lost its focus and will lead to numerous questions in the Test Plan approval process.***

F. ROLES IN THE Y2K CERTIFICATION PROCESS

As the Y2K Test Plan author, you will come into contact with or be affected by the considerations, opinions, and decisions of a number of people. These are their roles

1. Application Owners Roles:

Author: - If you are the one who is responsible for authoring the Y2K Test Plan then that's you. Following the processes and templates we are presenting in this workbook will cut hundreds of hours, at least, out of the job.

Business Owner: -This is the Application's Business Unit owner, as specified in the AMS database. They are one half of the partnership responsible for creating the test plan and its successful implementation.

MIS Owner: - This is the other half of the partnership mentioned above. Very often, the MIS owner also becomes the test plan author, or supervises the test plan author.

Managing Director or MD: - As specified in the AMS database, every Application will have a Managing Director who is responsible for it to Countrywide's senior management.

2. Y2K Client Server Team Roles

Coaches: – Your trainer for the ***Create A Test Plan Workshop*** is a coach. Their job is assist you in the test plan creation process to the point where your test plan has a reasonable chance to succeed in the examination process of the Test Plan Walk-through and Test Plan Approval processes. Your coach will invite you to a Preliminary walk-through, a live technical review of your document, about a week before your official Test Plan Walk-through. Coaching can make a significant difference in a successful or embarrassing Test Plan Walk-through. However coaching is very dependent upon the deadlines of your Project Plan. If you fall behind the Plan dates, coaching may no longer be an option.

Facilitator: – At the conclusion of coaching, you will be handed off to a Test Facilitator. The facilitator's primary job is to expedite the application through the entire Certification process in as efficient and timely manner as possible. Your facilitator will be the guide through entire process and will coordinate all issues up to and including actual certification. All questions and issues should be routed through them.

Technical Reviewer: – Once your test plan has been submitted the technical reviewer assigned to your facilitator's team will have a role in

reviewing your test plan for approval. Once the application has reached the testing phase, the technical reviewer and at least one other specialized technical reviewer will review the application's first test cycle. Since some reviewers may only see the Test Details portion of your test plan, this section must be able to logically stand-alone.

Senior Y2K Member: – Each facilitator is responsible to a Team Leader who is a Senior Y2K specialist. You will probably not meet this person until the Test Plan Walk-through. These are experienced Y2K personnel who have reviewed numerous test plans and you can expect them to meticulously examine your document with a skeptical eye.

Internal Auditors: - Internal Audit plays a major part in the overall Y2K process. They are responsible for making policy decisions, deciding what is at risk and overseeing Y2K readiness from Countrywide's legal perspective. They will be following the approval, testing, and archiving journey of your application.

3rd Party Auditors: - Countrywide has contracted with specialized outside auditors to insure that the entire testing, documenting, and archiving process is being competently handled. They act as "expert" witnesses for the entire process. They will review your document prior to testing and follow the testing phase until all of archival components are put away.

Lab Coordinator: - As a part of the test plan approval process, a lab coordinator will be assigned to assist your progress during the actual testing phase. They will set up your equipment, install requested software and assist you throughout your stay in the lab.

AS400 Analyst: - If there is an interface to an AS400, you can expect your application to be assigned to a specialist in this relationship. Their presence is very helpful in working through the proper testing relationships and logistics required to test this connection.

Systems Integrity (PVCS): - Before the application can enter the test lab a copy of the source code to be used in the lab must be registered with Systems Integrity. This group will be watching for the current version of your application and will stay involved for any further releases until the year 2000 is reached.

Interfaces Representative: - If your application interfaces to another CHL application, there will be a representative of the Interfaces group involved. They will probably contact you before the Test Plan Walk-through and they are a good reference for interface information.

3. The Test Plan Walk-through

One of the two major milestones that the Y2K Test Plan author must be aware of is the Test Plan Walk-through. Someone representing all of the above roles will be in attendance at the Walk-through. This is the acid test of your test plan. You can be certain that every possible question that can be asked regarding your plan will be asked. Anywhere from 15 to 20 people may be invited to this formal meeting. A poorly written test plan can be very embarrassing to the author but the questioning process must exist to insure a fully disclosed Y2K Test Plan. It is best to have answered your questions in advance as much as you possibly can.

4. Test Plan Approval Milestone

The next milestone in finishing a Y2K Test Plan is the approval process. Without doubt, there will be revisions requested from the Test Plan Walk-through. If you have written a successful test plan these will be minor revisions. When these final revisions are made, and all of the interests represented at the Test Plan Walk-through are satisfied, your Facilitator will sign off on the plan., This Test Plan Approval for is for *Lab Testing*. Applications can NOT enter the test lab until this final approval for the Test Plan has been given.

You may have further work to do as the application moves through the testing and post-test process. You're not really off the hook until the final sign-offs on everything just prior to certification. Usually, the revisions are relatively minor but there have been occasions where new sections have been added, or scripts that were needed are discovered missing. There has been one occasion when an entirely revised plan was written and then attached to the original.

G. THE 6 SECTIONS OF THE TEST PLAN

The Y2K Test Plan is outlined in six major sections. Below is a brief description of each.

1. Executive Summary

A brief summary of the entire test plan, written as the last step in the authoring process.

2. Description

The "required disclosure" section. An easy to understand description of the Application's purpose, an inventory of all the application's pieces, diagrams of how the pieces work.

3. Test Scope
A focusing section that defines what parts of the component inventory will be tested and what won't and why.

4. Test Details – Conditions, Cycles, & Scripts
The heart of the test plan, these components detail the testing process.

5. Test Data Strategy
A description of how and why you are preparing your test data to re-create Test Conditions and a note about how you are saving your test output files for validating each step necessary to prove your test script.

6. Test Environment
The hardware and software necessary to simulate the Production environment in which the application normally resides.

H. TEST PLAN CONTINUITY

Following the step by step process presented in this workbook will make the test plan authoring process as painless as possible. The key concept is continuity. By following each step accurately, the next step in the process is defined before you start it. The information in the section you will begin with, Description will provide information that can be filtered down the outline. Figure 2, below, illustrates this process.

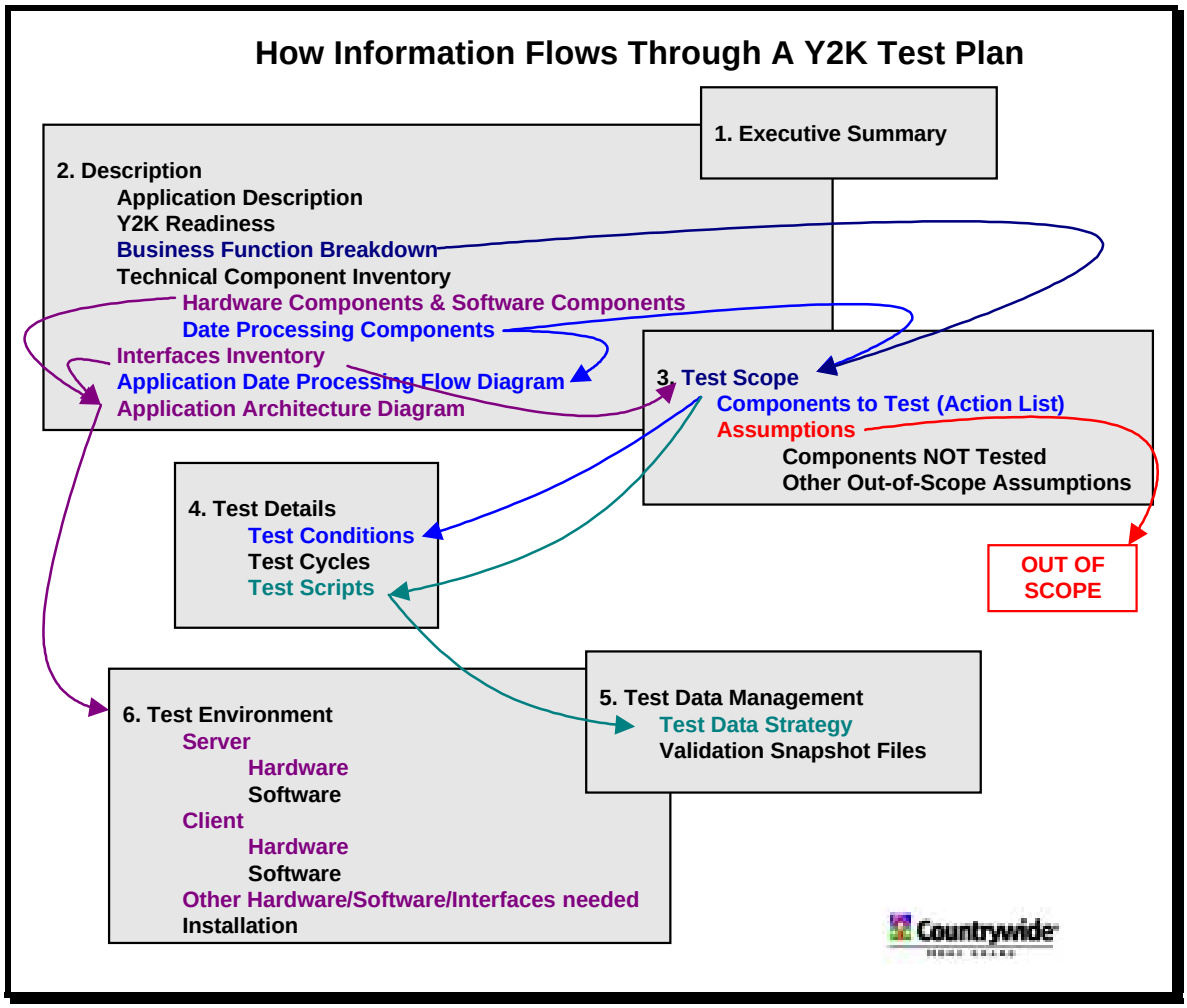


Figure 2

I. COVER PAGE

Through out this workbook, we will stop for each heading and write down an element or two for each heading. This will ensure that you are clear on the information presented and help the trainer know that the procedure is being communicated. This method should also provide you, as the author, a head start on your application's Y2K Test Plan first draft.

Test Plan Cover Page Worksheet

Let's start with the Cover page. In the blank page below, fill in the Cover sheet as indicated.

Countrywide

Write in APPLICATION NAME

Write in any Version number

YEAR 2000 CERTIFICATION
TEST PLAN

Author: write in Author's Name
Last Revision: write in today's date

J. EXAMPLE COVER FROM SURVEYPOWER

Countrywide

SURVEYPOWER

YEAR 2000 CERTIFICATION
TEST PLAN

Author: Y2K Client Server Test Team
Last Revision: February 3, 1999

II. Description

The methodology of this Y2K Test Plan preparation depends to a large degree on the success of accurately describing your application in the several perspectives outlined below. The principle here is that from this section every other section will flow. This means that there will be no holes in your plan because everything is inventoried in the front and then accounted for through every step of the plan. Because this section is so important you may find that you will spend a large portion of your time getting this section exactly right. However, you will much more than make up for it in the other sections because most of the guidelines will have been logically set by the information in this section.

A. Description Template

2.0 DESCRIPTION

2.1 Application Description
2.2 Year 2000 Readiness
2.3 Application Technical/Hardware Environment
2.4 Primary Business Function Breakdown
2.5 Technical modules or components
2.4.1 Hardware Components and Software Platforms
2.4.2 Code Organization (Date Handling Process Level)
2.5 Interface Inventory
2.5.1 The Input Interfaces Grid
2.5.2 The Output Interfaces Grid
2.6 Application Process Flow Diagram
2.7 Application Architecture Diagram

B. Objectives Of This Section

After completing this section you will be able to:

- Create a Description section that will assist you through preparing the rest of your plan.
- Create a Test Plan that guarantees you have met the required disclosure requirement.
- Define your Application from four different perspectives in lists and in two diagrams.
- Match elements within the Description module for clarity

C. Summarize The Application's Business Purpose

This section is simply the way you would describe your application if a reporter from Business Week asked you to describe it for a magazine article. Its purpose is to quickly introduce a reader who is unfamiliar with your application to its purpose at Countrywide. Initially, this is the myriad of people who are not familiar with your application yet will be taking some hand in the process. Of course, this includes everybody on the Y2K team, whose roles were defined above.

One thing you can count on as an author, you will definitely have readers who will hang on your every word.

Application Description Worksheet

This summary is high-level and should be stated as succinctly as possible. A paragraph or two should suffice. In the space on the next page, take no more than 5 minutes to write a quick description of your application's business purpose.

APPLICATION DESCRIPTION

D. DESCRIBE THE APPLICATION'S Y2K READINESS

All that is required here is a quick description of why the owners of the application believe it is ready for Y2K Testing. If it has been converted just briefly, in a statement or two, what was done. That's it. If any of the concerned parties believe there is reason to question this Y2K Readiness, you can stop writing now – there will be no test until the application is Y2K Ready.

To call the Y2K Test a test is almost a misnomer. Actually it is more of a Certification Proof. This is not a test to just check and see if your application should be converted to Y2K; it is a certified demonstration that it does, in fact, operate in a forward-dated environment. If your application owners do not believe that this application is Y2K Ready,

then the application MUST be converted to Y2K Readiness before proceeding with any Y2K Certification Testing.

Y2K Readiness Worksheet
In this section, in a sentence or two, describe why you believe your application is Y2K Ready.

Y2K Ready Status

E. Breakdown the Application Into It's Major Business Functions

This heading begins the most important parts of the Description section, if not in the entire Y2K Test Plan. From this inventory we will fulfill the Required Disclosure requirement of the plan. We will develop four lists of components from four different views, and from these views we will develop graphic diagrams that further illustrate the relationship between the components in these lists. The information from this section and the following three will provide the guidelines for the rest of the plan, as illustrated in Figure 2.

Major Business Functions
This first list should define how the application's user looks at the operation of the application. The Business Owner's perspective and application operator's of the application are good sources for this information. What you are doing is "chunking" out the actions the program takes to meet the application description written in section 2.1. Since this system defines the major pieces of the operation of your application, it will be a good source for our Test Conditions in the Test Details section of the plan.

Business Functions Worksheet

Write down what you perceive today as the major business functions of your application from your user's point of view. There is only room for

five. For many applications that may be enough but even if there are more, we won't need more than a couple to use in the workbook.

Major Business Functions (for eventual use as Test Conditions)

F. LIST APPLICATION'S TECHNICAL COMPONENT INVENTORY

1. Hardware Components and Software Components

This section describes the physicality of your application and it will become the basis for the Application Architecture Diagram. First, list your major hardware pieces but don't list duplicate hardware pieces. These are usually client workstations and application or data servers but also may include some terminals such as a fax machine or some other device that resides on a node outside of any of these pieces yet is still a part of your system.

Under each "box", list the software that is installed on each piece. List Operating Systems, Internal Application pieces, other installed software; the entire inventory of what is on each of the machines that drive your system. Be certain to include which pieces of hardware are home to your specific application.

Don't include interfaces to other CHL applications, AS400, external data feeds, or any utilities that are outside the scope of your application here. We'll cover interfaces soon. This is intended to be an internal list of your system's pieces. These components, along with the Interfaces section next, provide the information for the Application Architecture Diagram.

Hardware and Software Components Worksheet

Take a moment to write down these components in the space below. Once again, it is not important if the list is complete at this point, we are still working on concepts. List a Client box first and then indent the software components that reside on the Client directly beneath them. Do the same for a Server. We know you have these, after all we are Client Server Y2K!

Hardware Components and Software Components (for Application Architecture Diagram)

2. Date Processing Components

This may be the hardest section to define. Doing this list will require that you really think about your application in a certain way. By defining your application in major chunks of functional code (focused on date significance), you will provide one of the most important perspectives of your application. In the Process Flow Diagram, coming up, we will illustrate how these code modules(processes) interact with one another. Later, they will be the basis for what is "in scope" and what is "out of scope" for this test. And, they will provide a way for us to determine our Test Scripts and a way to help us avoid redundancy in our test. If we can get this right, everything's downhill from here.

Date Significance as focus

Don't get discouraged if you aren't quite sure how to start this "chunking" process. We'll try to help. Granularity is probably your first concern. How far do we drill down in this effort? Find your level of date

handling. To put things into focus, a Process Flow Diagram that shows application processing is essential to the plan, but ***only in reference to dates***. If you have database fields that have dates in them, ask yourself, what level of process do I need to show to illustrate how they are stored, accessed, manipulated, and displayed. This is the level that you will want to diagram. Detail is only needed in those chunks of functional code that are date significant. Much less detail is required from those chunks that are not. Non date significant processing chunks can be bunched under larger, higher-level chunks.

For instance, if you know your application, in some process or function, is converting 2 digit years to 4 digit years, you will want to emphasize that component here. Where you know date math or some other date manipulation is being done, make sure that component is emphasized as a process. Knowing these processes should help you find the granularity of your code component chunks and focus your list on date data handling.

Another example, you have 50 batch files. First chunk them into which ones are date significant and which ones aren't. The ones that aren't significant could be represented in one box. Now, to continue chunking, consider the logical functions of these batch files. Perhaps, they logically only really perform 3 or 4 permutations of ***date handling functions*** that could be grouped together. This level of detail will be exactly what we need.

Code Organization Worksheet

In the section below, take 5 minutes and rough out some major code chunks. It is not necessary for this to be complete at the workshop stage, so just make the assumptions necessary to get 3 or 4 date significant processes down. If you need help, refer to the SurveyPower Sample Test Plan at the end of this section.

Code Organization (for Test Scope, Process Flow Diagram, and Test Scripts)

G. INTERFACE INVENTORY

This is the final perspective, in list form, of your application and it is about your application's connectivity. The Interface Inventory is defined by filling in the blanks of the Input and Output grids below. This perspective is your view to how you are attached to the world outside of your application's scope both internally (CHL) and externally (3rd party data sources). The interfaces coupled with the Hardware and Software elements will provide the components of the Application Architecture Diagram. Maybe the Y2K Infrastructure group has tested these interfaces; note them here anyway. This will help them in confirming all of the interfaces from both sides. We'll look closer at them later but write them down here when writing your plan, so that you have demonstrated that they have been considered.

AS400 Internal Interfaces
Be sure to include any internal interface to the AS400 platform. This is a particular red flag. It's not a problem but it will be necessary to look carefully into this interface for a number of the interested parties.

Workbook
If your application interfaces to one or more systems outside the scope of your application, either internally or externally do your best to fill out the columns below for one or two of them. We'll start with inputs to your system.

This is what is required in each of the columns:

1. The Input Interfaces Grid

The column descriptions below should help you fill in the blanks in this grid to identify data ***input*** into your Application. If your application accepts data from outside itself, use the column descriptions below to fill to enter one or two.

Columns 1 – *Commonly known Interface Job Name, (if applicable)* **–** If the interface job has some commonly known name, enter it here.

Column 2 – *Application Name of Interface Source* - Enter the name of the application sending data to your application.

Column 3 – *Frequency of interface* - How often does the interface run its routine to your application;. daily, weekly, monthly, on demand, as needed etc.

Columns 4 – *Transfer software* – if your application uses a software application to transport the data, then list it here.

Column 5 –*Live or Simulated data from source* - **Input Only** - will the data be created by the source application for all cycles be received from interface as live data or will the data be simulated for the test. Enter "Live" or "Simulated". Live data is likely only in the case of an Out-of-Lab test.

Column 6 – *Will Interface Platform be Forward Dated? (Y/N)* – **Input Only** - will the source server from which the data is coming from be forward dated? Enter Yes or No or N/A if you are not testing the whole interface process.

Column 7 – *Dates Represented in 6 or 8 digits* – Enter 6 or 8 digits.

Column 8 - *Windowing* – If the dates are represented in 6 digits, and a windowing rule is being used to determine the century, state the windowing rule e.g. 50/50. Less than 50, the century defaults to 20xx, greater than 50, the century defaults to 19xx. Enter windowing rule or N/A.

Column 9 - *Comments* – Use this column to include any further clarifications.

Workbook

Enter an Input Interface into your application if you think your application uses one.

Interfaces that Input into your Application:								
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Commonly known Interface Job Name, (if applicable)	**Application Name of Data Source**	**Frequency of Interface Process**	**Transfer Software, if used**	**Live or Simulated data from source ?**	**Will Interface Platform Forward Dated ? (Y/N)**	**Dates Represented in 6 or 8 digits?**	**Windowing e.g, 50/50 Rule**	**Comments**

2. The Output Interfaces Grid

This grid is similar to the Input Interfaces grid but is minus two columns from that grid that are not applicable. The column descriptions below should help you fill in the blanks in this grid to identify data output from your Application. If your application sends data to some entity outside of itself, use the column descriptions below to fill to enter one or two.

Column 1 – *Commonly known Interface Job Name, (if applicable)* – If the interface job has some commonly known name, enter it here.

Column 2 – *Application Name of Data Source* - Enter the name of the application receiving data from your application.

Column 3 – *Frequency of Interface Process* - How often does the interface run its routine with your application; daily, weekly, monthly, on demand, as needed etc.

Columns 4 – *Transfer software* – if your application uses a software application to transport the data, then list it here.

Column 5 – *Dates Represented in 6 or 8 digits* – Enter 6 or 8 digits.

Column 6 - *Windowing* – If the dates are represented in 6 digits, and a windowing rule is being used to determine the century, state the windowing rule e.g. 50/50. Less than 50, the century defaults to 20xx, greater than 50, the century defaults to 19xx. Enter windowing rule or N/A.

Column 7 - Comments to describe any further clarifications.

Workbook

Enter an Output Interface to your application if you think it has one.

Interfaces that your Application Outputs To:						
1	**2**	**3**	**4**	**5**	**6**	**7**
Commonly known Interface Job Name, (if applicable)	**Application Name of Data Target**	**Frequency of Interface Process**	**Transfer Software, if used**	**Dates Represented in 6 or 8 digits**	**Windowing List Rule, e.g. 50/50 Rule**	**Comments**

H. APPLICATION DATE PROCESSING FLOW DIAGRAM

Now we get to test your artistic ability to draw boxes and lines. We are about to reap some of the benefit of the work that you have done. Referring to your first list, the Code Organization technical components, use some boxes and lines, **With Arrows showing process flow**, to show how ***DATE*** data is input, processed, transformed and output, or any other process that occurs. Our goal here is to be certain that every code chunk that we described in section **2.4.2 Code Organization** on page above is represented in this diagram. If you are having any trouble with this section, you may need to look back at how you organized the previous section. Chances are, that's where the problem is. You can also refer to the SurveyPower sample at the end of the section.

Process Flow Worksheet

Lick your pencil and in the blank space below, draw a rough diagram of how you believe your application works today. More research may show you that this needs revising but for today, your best guess will work just fine. A program like Visio will later transform this sketch into a work of art.

Application Process Flow Diagram

We're not quite done. Draw a square or circle around the components that have date handling in them. This will be your basis for the Test Scope, coming up.

I. APPLICATION ARCHITECTURE DIAGRAM

Another sketch. This time we will sketch the physical layout of your system including all of the components from your Hardware Components and Software Platforms list and from the Interfaces. Simple shapes will work just fine as Hardware pieces – a largish box for a server and a monitor-shaped box with a rectangle under it for the client workstation, a rough sketch of any other device.

Architecture Diagram Worksheet

In the space below, draw a rough diagram of how the physical components listed in the Hardware Components and Interfaces relate to each other. When you draw your lines, use arrows to indicate direction of flow, double-arrows for flow in both directions. Later you can use the neat shapes in Visio to make this diagram something to be proud of. The key issue is consistency – make certain that every component listed in the previous sections are accounted for in the diagram. Once again, if you are having trouble with this step, the problem probably lies in the previous list from which it is drawn.

Application Architecture Diagram

Two more boxes will help clarify this diagram. Draw a box around all of the components that are in your system. This is the **SYSTEM** box. Now draw a box around all of the components that are inside the CHL WAN. We'll call this the **CHL** box.

Any component outside the SYSTEM box should have a line with arrows to show flow, connecting it to a component in the SYSTEM box. If it is from inside the CHL box it is an internal interface, from outside of both boxes, an external interface. Between them; these interfaces should be accounted for in the Interfaces grid. These interfaces are of major concern in the Y2K effort and will scrutinized carefully during Test Plan review.

With the completion of this diagram we have completed the Description section. This should provide us with enough information to continue working in the Test Scope section, which comes next.

J. SUMMARY QUESTIONS

- How many list views of your application are there in the Test Plan Description? How many diagrams?
- Which technical component lists will make up the Process Flow Diagram? Which will make up the Application Architecture Diagram?
- What is the purpose of the Application Description section?
- What other purpose does this document serve aside from just being a Test Plan?
- How many people may be invited to a Test Plan Walkthrough?

K. EXAMPLE FROM SURVEYPOWER

Example from SurveyPower

2.0 Application Description

2.1 APPLICATION DESCRIPTION

The "SurveyPower" application is a system used to capture and report survey information about a customer's experience with obtaining a mortgage loan from Countrywide. The application has three major components:

- **SurveyPower Client** - An online survey-entry application (written in Visual Basic 5.0) that resides on an NT client
- **SurveyPower Database** - An Access 97 Database stored on the Skywalker server
- **SurveyPower Utility** - A report-generation application (written in Visual C++) that resides on an NT server and sends reports electronically via Lotus Notes email.

The inputs into the application are from NT Client workstations in the national branches attached to the WAN from surveys that are input by customers at the end of the closing process. These completed surveys are uploaded to a Countrywide's Simi Valley Hill Street location, where they are stored on the SurveyPower NT server (Skywalker) in an Access database (known as the SurveyPower Database).

On a nightly basis, a server routine (C++) summarizes the information gathered during the day and creates a daily overview report. At the end of the last business day of each month, a monthly routine creates a series of reports. At the end of the last business day of the year an annual routine creates a summary report for the year. These reports are delivered via a Lotus Notes email interface. The reports are named below:

- Daily Key Statistics Report
- End-of-month Satisfied Customer Profile
- End-of-month Dissatisfied Customer Profile
- Annual Customer Satisfaction Report Summary

These reports are saved in Access and reformatted in MS Word 6.0 (.doc) format on the NT server – this NT server is attached to the WAN, which distributes the formatted reports to key management personnel via a Lotus Notes email server. The server uses a Mail List sent to it daily in tab-delimited ASCII (.txt) format, which the Lotus Notes mail server converts to its own format.

A separate table in the SurveyPower Database contains email addresses of the various report recipients The SurveyPower utility triggers queries to create a tab-delimited, ASCII MailFile.txt, which is exported to the Lotus Notes email server to update its mail groups.

The significance of dates in "SurveyPower" are as follows:

- Day-end reporting is based on current system date of Client Workstation
- Month-end close dates are calculated, based on the system date of NT server (Skywalker)
- Date validation is applied to birth-dates (min/max = 1910/2010)

- Date validation is applied to loan initiation/close dates (min/max = 1990/2035)

At the end of each business day, the SurveyPower utility program triggers queries in the SurveyPower Database, which are then converted into a pre-formatted MS Word 6.0 document with tables for the report information. The report is then exported to a Lotus Notes database in native file format for attachment to the Mail List

On a monthly basis, the SurveyPower utility program uses the same method to create a two pre-formatted Monthly Reports.

On an annual basis, the SurveyPower utility program uses the same method to create a pre-formatted Annual Report.

2.2 YEAR 2000 READINESS STATUS

Prior to analysis and conversion, the client module used a two-digit number field to capture the year on all input fields. A "19" was appended to this date field to represent the century, and then the date was stored as a "DATE/TIME" field in SQL Server. (Note that the Access table used by the VB front-end for input is an "unbounded" form and uses DAO to update the SQL Server tables via ODBC). In order to ensure that this application was portable to another database, the developers did not rely on SQL Server's "windowing" to fix the century-problem, and instead coded the fix in the Access form code as follows:

- All birth-dates were changed to use a four-digit year in the input field
- All other dates continue to use a two-digit year; instead of hard-coding the century to "19," all years less than 50 will use "20" for the century, and all years 50 or greater will use "19" for the century

The server module did not require any conversion. The module extracts SQL Server DATE/TIME fields and produces reports that display 4-digit years.

2.3 TECHNICAL COMPONENT INVENTORY

2.3.1 Code Organization

Visual Basic – SurveyPower Client
Receive Input
Store
Forward (timer)
Print routines

Visual Basic – SurveyPower Server
Receive Data
Format Data
Trigger Survey Power Access Database "Input & Append Query"
Edit Mail List table
Print routines

Access 97
Tables with Dates
Tables without Dates
Mail List table
Queries

Append & Populate query
Daily Report query
Monthly Satisfied Customer query
Monthly Dissatisfied Customer query
MailList query

C++

Timed Triggers to run SurveyPower database queries
Trigger to run MailList query
Format queries for Lotus Notes
Upload to Lotus Notes server
Print routines

2.3.2 Hardware Components and Software Platforms Inventory

Multiple SurveyPower equipped workstations in CHL branches nationwide
OS – Windows NT 4.5 Client
SurveyPower Client-side application
WAN Connectivity

Windows NT Server – "Skywalker"
OS – Windows NT 4.5 Server
SurveyPower Server-side application
Access 97
C++
WAN Connectivity

Lotus Notes Server

2.4 INTERFACE INVENTORY

2.4.1 The Input Interfaces

There are no Input Interfaces from any Internal or External Data Feed.

Interfaces that Input into your Application:								
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
External/CHL Internal	**Application Name of Data Source**	**Frequency of Interface Process**	**Transfer Software, if used**	**Live or Simulated data from source ?**	**Will Interface Platform Forward Dated ? (Y/N)**	**Dates Represented in 6 or 8 digits?**	**Windowing e.g, 50/50 Rule**	**Comments**

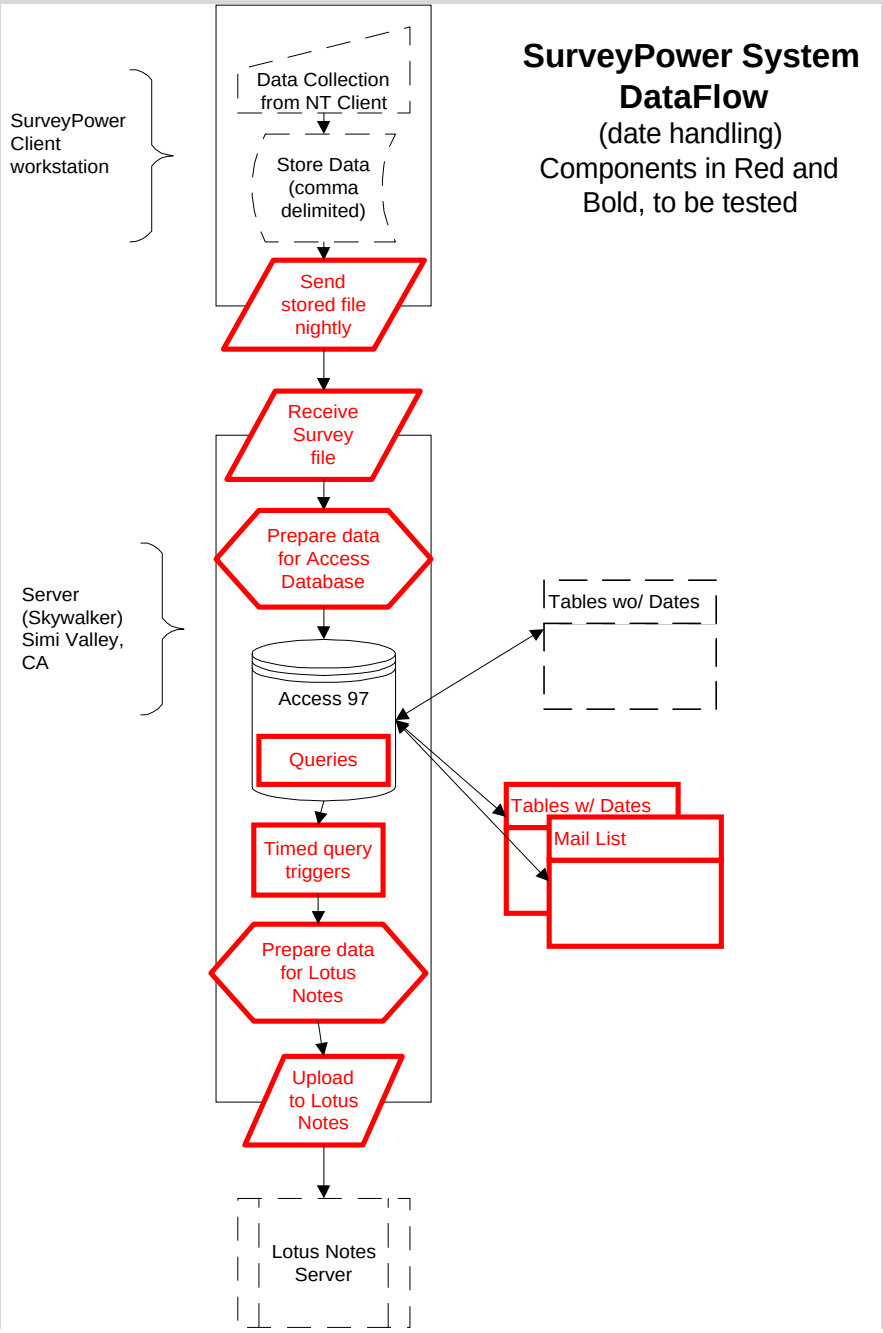
2.4.2 The Output Interfaces

Interfaces that your Application Outputs To:						
1	**2**	**3**	**4**	**5**	**6**	**7**
External/CHL Internal	**Application Name of Data Target**	**Frequency of Interface Process**	**Transfer Software, if used**	**Dates Represented in 6 or 8 digits**	**Windowing List Rule, e.g. 50/50 Rule**	**Comments**
Internal	Lotus Notes email server (server name)	Daily, Monthly, Annually	Internal C++ module	8 digits		No Y2k compliance issues expected

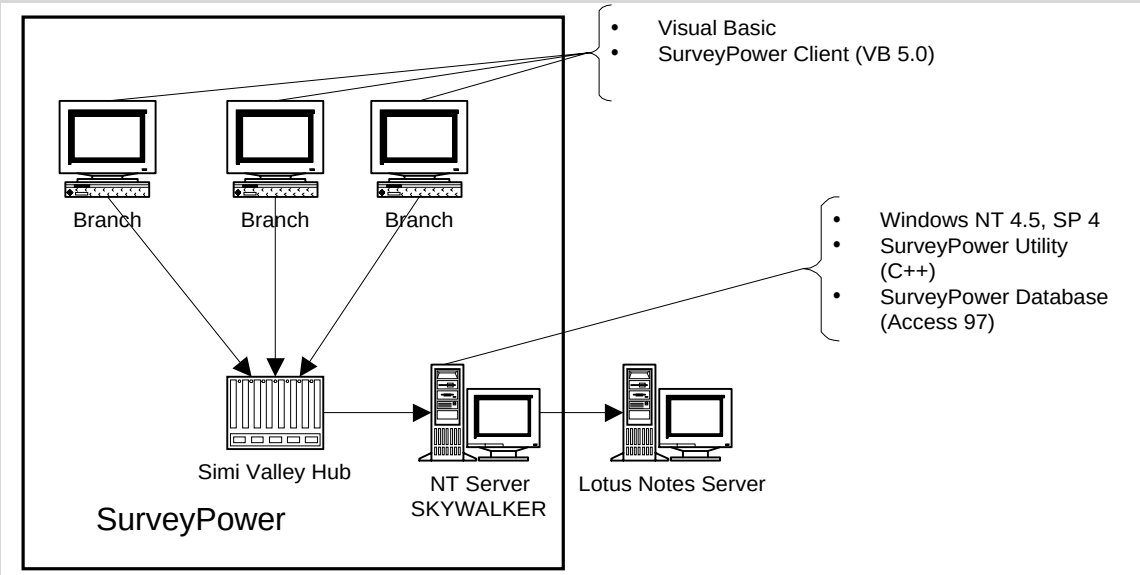
2.5 BUSINESS FUNCTIONS BREAKDOWN

Accept Customer Input
Send nightly data to Simi Valley
Store Data in SurveyPower Access database
Edit Electronic Mail List in SurveyPower database
Print Reports
Distribute Reports through Lotus Notes

2.6 APPLICATION PROCESS FLOW DIAGRAM



2.4 Application Architecture Diagram



III. Test Scope

The Test Scope section is about accountability and clearly defining what you are going to do and what you are not going to do. In some way or another, everything that has been mentioned in the Description section should be accounted for here.

A. Test Scope Section Template

3.0 TEST SCOPE

3.1 Business Functions and Application Modules Tested – Action List

3.2 Assumptions

3.2.1 Technical Components that will Not Be Tested

3.2.2 3rd Party Interfaces, Components, & Utilities

3.2.3 Functional Assumptions

B. Objectives Of This Section

At the end of this section, you should be able to:

- Use the Information from the Description section to determine all technical components that will be tested.
- Use the information from the Description section to determine all technical components that will NOT be tested.
- Use the information from the Description section to determine any other assumptions that are being made that might bear upon the Y2K testing.

C. Action List – Functions and Processes To Be Tested

FUNCTIONS

One of your easier lists was the breakout of business functions that you completed in the Description section. List here ONLY those business functions that have date handling as a part of their function. This list is fairly easily derived from the section ***II.C Summarize The Application's Business Purpose*** on page 16. Believe it or not, not only have we taken the first step in drawing an outline around what we intend to test, we have also taken a first step towards defining our Test Conditions in the upcoming section.

Business Function Worksheet

Looking at your Business Functions list, determine which ones have anything at all to do with dates and list them below. For workshop purposes we only really need a couple.

Business Functions to be tested

PROCESSES

The final step in describing the scope of your plan is to define the code processes that are used to perform the Business functions we just listed. Look back at section on *Error! Reference source not found.* on page **Error! Bookmark not defined.** and also use the Application Date Processing Flow Diagram on page 25 for reference. What we are looking for are those code components that perform the work for the Business Functions we are going to test.

Action List Worksheet

Take the first Business Function listed in the table above and put it in the worksheet below. Looking at the Code Organization section, you have listed the logical modules of your application. All modules that have any date handling whatsoever, even if you aren't sure whether they do, should be listed here. List, indented in outline form, all of those components that would be exercised in the Business Function you have just placed here. When you have finished, outdent and try another one.

Action List – Technical Components to be Tested

Also check your ***Interface Grids*** on pages 23 and 24. If you have interfaces that you believe should be tested, put them under the business function to which they apply. Can one Business Function have a code component under it that is used by another Business Function? Yes, it could happen. Don't worry we'll squeeze this redundancy out in the Test Scripts.

D. THE ASSUMPTIONS LIST

Sometimes the word "Assumptions" throws people off here. Assumptions, for our purposes, are components that you "assume" will not have to be tested. There are two main reasons why you would make such an assumption:

- Because you are certain that a particular component of your application has no possible connection to any kind of date handling. This component never even "sees" a date, or…

- Because they are outside the scope of your application and you have no control over them. External interfaces and 3rd party components that are connected to, or embedded in, your application but which you obviously cannot test are examples of these.

Let's look a little closer below.

1. Technical Components NOT To Be Tested and Why

In this list, every component on our Software Code Organization list that was NOT in the Action List above should appear on this list. In terms of required disclosure, we have accounted for all the components that were listed in the Description section. It will also help us in terms of script writing later because we obviously won't need to worry about scripts that exercise these components - we're not going to test them.

Worksheet
First, write down all of the items on your ***Business Function Breakdown*** list on page 18, and any code processes from the ***Code Organization*** list on page **Error! Bookmark not defined.**, that you did NOT list to be tested above, in the first column below. If you don't have any items to write down because you're testing them all, no problem - Don't, and skip the step below, too.

Items Not To Be Tested Worksheet

You obviously had a good reason for leaving components off your previous list of components that you will test. So, next, whatever your good reason is, just write it down in the second column below.

Technical Components NOT To Be Tested And Why	

2. Other Assumptions

- **Internal Interfaces**
- **External Interfaces**
- **3rd party Languages, Operating Systems, Applications and Platforms**
- **3rd Party Components (embedded)**
- **3rd Party Utilities (invisible but not embedded)**

Now we'll look at things that are beyond the scope of your application and, often by definition, beyond your ability to test. It would be hard to imagine that Microsoft will send you any of their source code to test. These kinds of assumptions, where you are "assuming" the code will work in the upcoming millennium, fall generally into three categories:

- **Internal Interfaces** to other Countrywide systems – There is usually a lot of interaction between Countrywide's proprietary systems. You may remember that we have inventoried all of our interfaces, both coming into and going out of our application. Those that we are testing are mentioned above as part of the Test Scope. Note here your assumptions about those that are NOT on that list here.

- **External Interfaces** - Many of Countrywide's applications receive data from outside the company for use in preparing quotes, setting property values, and other information important to the operation of the company. If there is piece of the external's interface living in one of our system's hardware components, consider it a code component and place it under the Business Function that uses it in the Action List.

- **3rd party Languages, Operating Systems, Applications and Platforms** – These are generally "shrink wrap" applications that must be accounted for in our system. Our architecture diagram is the source for these components of our system. The assumptions we make about these components are based on their own Compliance statements. Your facilitator will supply with the documentation later, they just need to be listed at this point.

- **3rd party utilities**. Some of the trickiest 3rd party components to winkle out of our system are hidden in either the client or the server somewhere, maybe as a transport utility, maybe as some kind of data packager. Nevertheless, our application will come to a screeching halt if they do not operate correctly in the upcoming century. On the other hand, it is again well beyond the scope of our mission to test these. What assumptions can we make regarding these programs?

- **3rd party Embedded controls or modules** – Deep inside our application there may be 3rd party VB Controls or C++ modules that were licensed for use. These may be difficult to determine but they

must be found so we can account for them. Check for "add-ins" in the source code IDE. Interviewing every developer who had anything to do with writing or revising the application may be the only way to find out. What are we assuming about them?

Clearly we will not be able to test these types of components, so we must rely upon their Y2K Compliance statements. For Countrywide 3rd party standard components, your Facilitator will be able to supply you with the Compliance statements that you will need to attach to your Test Plan. If any of these kind of components are NOT Countrywide Standard, then you will need to find the manufacturer and get a legal statement from them about their Y2K Compliance Status. A good place to look is at their Website.

You should be able to think of examples that fall into each of these categories.

Other Assumptions Worksheet
In the worksheet below, write down any of these kinds of assumptions and the reason why you are making the assumption that they will work in our forward-dated environment. The reason will almost always be based on the Y2K Compliance status of these kinds of components.

Other Assumptions	

3. Functional Assumptions

This last worksheet deals with anything we didn't catch in the previous worksheets. Normally, anything in this section will be situations rather than strictly components. You may have to make some assumptions about your environment to make your test work. Perhaps you are going to simulate some part of your production environment that goes beyond the Test Environment section (coming up later) and so, must make some assumptions about that simulation.

In the actual preparation of your plan, you may find that this section has to be filled in later to account for some assumption that will need to explain that you didn't think of. It should be obvious by now that there will be many times when

you will be working forward through your plan but that you will often work backwards through it to achieve consistency until it is completely finished.

Functional Assumptions Worksheet
If you can think of any assumption of this sort that might apply to your application, write it to below.

Functional Assumptions

E. SUMMARY QUESTIONS

What are some commonly used interfaces at Countrywide?

Can you think of an example of a 3rd party application, interface, or utility that could be in our applications?

Can you name any possible 3rd party components that could be embedded in the application's code?

Can you think of a situation that would require mentioning in the "Functional Assumptions" heading of this section?

F. EXAMPLE FROM SURVEYPOWER

Example from SurveyPower

3.0 Test Scope

3.1 BUSINESS FUNCTIONS & TECHNICAL COMPONENTS TO BE TESTED

Both the client and server modules have date-related functionality, and both modules are critical to the core business function provided by the application. Therefore, both modules will be tested during this effort. The SurveyPower Utility on the Server sends date related information to the Lotus Notes server, so the interface to a Lotus Notes server will be simulated for certification. The interface is critical for upper management reporting and will also be included in the test.

Send nightly data to Simi Valley

Append and store date data in comma delimited file text file

Upload to Server (timer)

Store Data in SurveyPower Access database

Receive Data

Reformat certain text fields to number fields (including dates)

Queries

- Daily Report Query
- Monthly Report Queries
- Annual Report Query
- MailList Query

Triggers

- Daily (Report and MailList)
- Monthly
- Annual

Distribute Reports through Lotus Notes

Format Reports for Lotus Notes

Upload to Lotus Notes

3.2 ASSUMPTIONS

3.2.1 Business Functions & Technical Components NOT to be Tested

Accept Customer Input

This function does not need to be tested for two reasons

1. Entry filters define how date fields can be entered
2. All data is stored in text form and expanded in the Server so no actual dates are being used in the process.

Edit Electronic Mail List in SurveyPower database

This function does not need to be tested because there is no date input or handling. Email addresses and recipients are added and deleted to this dynamically. Just before and of the "timed" uploads, this table is used to refresh the Lotus Notes mail group and the trigger and upload events will be tested.

Print Reports

While printing reports would normally be an issue, the local printing of reports is a mirror of the reports that are formatted and sent to the Mail List recipients. Testing them in the more complex process that is involved also tests this function.

3.2.2 Other Assumptions

- SurveyPower operates using Microsoft Y2K-compliant operating systems, database, and application software, which will not be tested other than where they support the SurveyPower application.

3.2.3 Functional Assumptions

- Since the test will occur in a closed lab environment with no access to the WAN, a Client workstation will be connected directly to the server through a normal WAN connection. Data type will remain the same. This simulation assumes that the WAN will be operative on any of the Date Cycles tested.

IV. Test Details

When we reach this part of your Test Plan we have reached the heart of the plan. So far we've identified pieces and parts of the application and we've identified which ones we're going to test and which ones we're not. We are finished with the ones we're not going to test. From this point forward we will focus entirely on the components in our Action List exclusively.

A. Managing Test Data Template

4.0 TEST DATA MANAGEMENT

4.1 Test Conditions
4.2 Test Cycles
4.3 Test Scripts, Validation Points, and Test Data

B. Objectives Of This Section

At the end of this section, you should be able to:

- Understand how to write Test Conditions based on your Business Function Breakdown from the Description section.
- Understand how to choose your Test Cycle dates.
- Understand how to define a Test Script and how to synchronize them with the Test Conditions, Test Cycles, and Test Data.

C. Overview

Once you have finished with your Test Details it would be very useful to return to the top of this section and give a brief overview of how your chose your Test Conditions, Test Cycles. While you're at it, why not mention why you're certain that your Test Scripts will recreate every necessary iteration of your Test Conditions. This is not required, but there is no doubt that it will make your Test Details much easier to comprehend.

D. Test Conditions

The term Test Conditions can be a little misleading. We also often use the word scenario as well. It's actually part of the process of "chunking" our application up into testable pieces. Fortunately, if we've done our Description section correctly, we can cut right to the chase.

Almost always, Test Conditions are no more than the Business Functions described in the Description section and identified in the Test Scope – Action List. If we've defined these correctly we should be able to imagine the application user performing one of these functions. If the level of granularity leads you to believe that there may be more than one function involved here and they are similar or logically grouped, that problem is solved in the Test Script. If you now see that two or more of the functions you have earlier defined should really be combined, then do that before finishing this section.

If, on the other hand, you feel you have been a little too high-level and disparate functions are combined in this Business Function, go back to the Description section and "chunk" some more and move those changes forward to the Test Scope section.

Test Condition Worksheet

For our workshop, we will take one of our Business Functions and turn it into a test condition. Of course, we will want to take it from the Test Scope Action List, it wouldn't make any sense to test one that has been assumed to have no relevance. In the space below, rephrase the Business Function slightly to include how it relates to date significance. Also, as a part of the description of the Test Condition, explain a little bit about the function if it is not self-evident. Remember some members of the reviewing team may never see more than this Test Details section. Test Conditions need their own identifier so we have used a simple "TC-01" to signify that this is Test Condition 1. If you are having any trouble with this exercise, refer to the sample SurveyPower test conditions.

Test Condition Description TC-01

Test Condition Synchronization Worksheet

If you have looked at the Annotated Test Plan shell or the Sample, you have probably noticed that there are some other columns that need to be synchronized with the Test Condition. These will become very clear by the time we've finished the section so we'll let that go for now and move on to Test Cycles.

Test Cycle(s)	**Script(s)**

E. TEST CYCLES

Test Cycle Dates
Test Cycles are easy to define; they are a series of dates or date ranges in which a Test Condition will be tested. That means that the system dates throughout the Test Environment are set to the cycle date when the test is run. That's simple enough but the tricky part is figuring out which of these dates need to be a Test Cycle for your particular application. The auditors are looking for ***four key Test Cycles:***

- **Baseline** – A date prior to Y2000, that can be used for comparison
- **January 4, 2000** – First business day at CHL (or first processing date in Y2000)
- **February 29, 2000** – Leap Year Day
- **Upper Bound Date** – The furthest day into the future that your application can be certified to work. This is usually based on the earliest compliant date of any component your application is dependent upon.

You should probably consider another issue. If your application does any date manipulation over time, a roll-over test of the System Clock moving from 1999 to 2000. Also, if your application manipulate dates from the past make sure that the cycle has scripts to prove a condition where this is a requirement. There maybe very specific dates associated with your application but we have put a list of "Hot Dates" just below the Test Cycle Worksheet. Refer to them for cycle possibilities beyond the ones mentioned here. Even though they are pretty generic, let them jiggle your imagination as you think about your own application.

Description
Each Test Cycle should have a description. Here you give the exact date (or date range) and time (if applicable) of the ***SYSTEM DATE*** that all components of the Test Environment will be set to, to run the script, or group of scripts assigned to it. Note that this has nothing to do with input dates. That's actually a part of the Test Data and Test Script steps and should have nothing to do with the Test Cycle.

Include your reason for choosing this particular Test Cycle date. It may be the generic reasons given above or a reason specific to your application. Just so that it is logical, put your Test Cycles in chronological order. It's interesting that every time a Test Cycle is listed out of chronological order, there is question from somebody wondering if there is any significance in that.

Test Cycle Worksheet
Choose a couple of the key Test Cycle dates above. Write them in the worksheet just below. Each Test Cycle should have an identifying code number for synchronization and for use, later, in the naming convention for archiving the files. We've chosen the format in column one below. Complete the Test Cycle Description blank in the worksheet.

Test Cycle #	Test Cycle Description
CY - 01	
CY -02	

We've only written in a couple of Test Cycles but that's all we'll need to demonstrate how these elements work together. While there are some key dates and some obvious dates in the "Hot Dates" table, the MIS or Business Owner must be aware of important dates to your application that are not mentioned here. Don't assume that all the dates that are significant to your application are necessarily in this list.

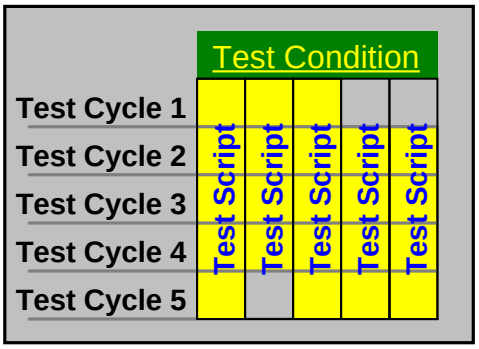
1. "Hot Dates" Dates to consider for Y2K Business Unit Preparedness

Date	*Group or Business Unit Impacted*	*Reason*
March 1, 1999 or	All	First day of Fiscal Year 2000 for Countrywide
August 22, 1999	Any Group Dependent on Global Positioning Satellite Systems	Overflow of "end of week" rollovers (e.g. GPS). GPS time based on 1024 weeks. Rollover from week 1023 to week 0000.
September 9, 1999	All	9/9/99 or possibly 9999 – to ensure that the digits "99" or "9999" do not trigger a red flag, other program subroutine(s), or cause a processing error, Date used sometimes as a program verification date.
January 1, 2000	Information Systems	First day of new year (Saturday). May be 1st business day of year. Key date in any compliance

January 3, 2000	All	First full work day in the New Year for some CHL applications and the First Monday in Year 2000
January 4, 2000	All	CHL's official first day of business in Year 2000
All significant processing dates	All	End of first business week, 1/7/2000, first month end 1/31/2000, first year-end 12/29/2000 or 12/30/2000 (Sat)
February 29, 2000	All	Leap Year
February 30, 2000	All	Ensure that this date is NOT processed
February 31, 2000	All	Ensure that this date is NOT processed
March 1, 2000	All	Ensure that this date is processed
October 10, 2000	All	First 8 character date in new millennium
January 01, 2001	All	First day in the 21st Century
February 29, 2001	All	Ensure that this date is NOT processed as a leap year
April 1, 2001	C++	This is actually more of a virus than a year 2000 problem. Microsoft claims that it is NOT a Y2K bug. It only occurs when daylight savings time falls on April 1 (year 2001), and it self-corrects after one week. Microsoft will be issuing a fix this year. However, as a matter of due diligence C++ developers should be aware of the issue.
February 29, 2004	All	Ensure that this date is processed as a leap year
January 01, 2010	TBD	Overflow ANSI C Library
Any month 28 years from baseline	All	First year that the same date in current year falls on the same day of week in the next century (for a 1999 baseline, 2027 would have the same date on the same day of week)
January 18, 2036	All	Windows NT Overflow date
January 19, 2038	TBD	Unix systems Overflow date
September 18, 2042	TBD	IBM System/360 Overflow date
Upper date limit of System	All	Upper bound of application development platform.

F. TEST SCRIPTS

A simple definition of a Test Script is a series of steps that recreate a Test Condition or some part of a Test Condition. As illustrated in the diagram to the right, the Test Condition is a Business Function with date handling in it. One or more Test Scripts vertically granulate the re-creation of the Test Condition. Test Cycles horizontally intersect the Test Scripts, depending upon whether a certain script needs to be run for that particular system date.



For instance, in the diagram, the Test Condition could be "Creates reports correctly". Each Test Script represents a different kind of report. Reports that use exactly the same code should run in the same script. You only need to prove that functionality once so we could group our reporting function into 5 categories that use different parts of the reporting software. So Script 1 might be "Management Reports", Script 2 might be "Accounting Reports", Script 3 might be "Sales Reports", and so on. Possibly, testing the functionality of Script 2 at Test Cycle 5 is not required because running the script again would simply duplicate a proof that was already made during an earlier cycle. Similarly, our last two scripts are not necessary to run in Test Cycle 1 (usually a Baseline date such as Today's date or December 31, 1999) because there is no significance to a baseline test for these scripts.

Steps

Since we have defined a Test Condition as a Business Function, a Test Script is generally what a user does to execute that particular Business Function. As we have shown, that can often involve more than one script. In our example above, we need 5 scripts to run and exercise 5 different code components and to fulfill the Test Condition

The steps to perform the script are simply the actions the user takes in the application to perform each iteration of the Business Function. We need to number each step because they will have significance in our Test Archive Snapshot directory, covered in the next section.

Step Description

The description cell should contain an explanation of what should be done at this specific step including data to be entered, key presses, mouse moves, function keys, whatever is necessary to show what the tester is doing in this step. Be specific.

Test Script Steps Worksheet

Let's take 5 minutes to try and create the steps necessary for a script in the worksheet below that will recreate all or some part of our Test Condition. The step numbers are already in, just complete the Step Description cell.

Step #	Step Description
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	

Detailed Expected Result.
There are really two pieces of related data that should appear in this cell. Let's start with the first one and get this information into our workbook.

- First, what should result from taking the step described? What should be seen on the screen or printout as a result of the action taken?

Detailed Expected Result Worksheet

Relating the worksheet below to the Step Descriptions above, write in the expected results for the steps in the worksheet below. Where there is no significant result, put in N/A.

Step #	Detailed Expected Result
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	

Technical Reviewer Directions

- The second item that should appear in the Detailed Expected Result cell is Technical Reviewer Directions. These are only necessary if there is a Validation Point.

You may ask, "what the heck is a Validation Point?" and rightfully so. We'll take a look at Validation Points and Validation Files before we cover what these Reviewer Directions should look like.

Validation Point (File)
Maybe a tricky thing to figure out at first, but once you get the hang of it, it's simple. A Test Script is going somewhere. You start at one place, make some steps until you reach the final step in the process. The final step in a Test Script is generally a point you want to document as proof that the application can operate the defined part of the Test Condition at the date of the Test Cycle. Clearly this is an important step in the script so we call this a "proof step", a Validation Point. A screen capture, or file creation that can be examined by a Technical Reviewer proves a Validation Point. So a Validation File is the file that supports a Validation Point. The final step is usually not the only Validation Point in the script. There are logical points in the process where supporting screens prove a subsequent step. These are also Validation Points and represent another place where you will want to create an output file.

We will cover the Validation File more thoroughly in the upcoming section, Managing Test Data. When a Validation File is created, the preceding cell should contain Technical Reviewer Directions on how this Validation File can easily be reviewed. If you expect to total a column of numbers on October 10, 2000, then tell the reviewer what that total should be in the output file. If there is a date field that is significant, let the reviewer know what he should see and where he can locate it on the screen. Even more importantly, if there are date fields that are NOT significant, and show as 6 digit dates, make sure the reviewer understands why.

A common situation should serve as a good example. Many programs still display 6-digit output to printed reports though all date handling in the application is actually done at 8 digits. If the reviewer were to see just the final output screen with a 6-digit date, everything would screech to a halt. However, the tester can create a file prior to that screen, which demonstrates that the supporting date is handled in SQL Server or some other method that operates successfully in the forward-dated environment. When the final step points out that this 6-digit date is just being displayed in a shortened format then the reviewer will understand and pass the test of that script.

Reviewer Directions Worksheet

Relating the worksheet below to the Step Descriptions worksheet above, determine which ones are Validation Points and write in reviewer instruction for those steps below. For now, don't worry about the Validation File name, we'll come back to that after we've discussed Test Data Management, next.

Step	Reviewer directions for Validation Points	Validation File
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		

Data Cross Reference Worksheet

The last synchronization point in the Test Script is the cell that references the test data file or files that are needed to run this script, if it is needed, to prove the Test Condition. We will cover this in depth in the next section. Each step that is referencing a particular Test Data file should be noted in the Test Script steps for synchronization. So after we've looked at the next section, we'll come back to fill this worksheet in later.

Step #	Test Data File
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	

More Script Description

Now, a little housekeeping:. To make our Test Script a little more descriptive to the reviewer, the Y2K Test Plan shell asks us to put some information at the top of each Test Script.

Test Script Heading Worksheet
Script Name – Choose some descriptive name like "Basic Reporting" or "Loan Processing" for your Script Name and write it in the space below.

Script Number - Test Scripts need an identifier as well so we will call our first Test Script – TS-01. Write this identifier below

Plat form: – All we need to determine here is whether this Test Script runs the application on the server side, client side, or both. As appropriate, put C for client, S for server, or CS, if this condition tests both.

Script Name:	
Script Number:	
Platform:	

Synchronization Worksheet
We have finished the three main elements at the heart of our Test Plan. Test Conditions, Test Cycles, and Test Scripts. We know have enough information to go back to our Test Condition and synchronize it to its supporting scripts and cycles.

On page 44, find the Worksheet that we left blank. Since we only have one Test Condition, two Test Cycles, and one Test Script, this should be fairly easy.

Test Cycles – In this column put either, neither, or both of the Test Cycles from our Test Cycles Worksheet on page 45. Are both of them significant to our test? Either of them, or maybe none?

Test Scripts – The second column asks for which Test Scripts are involved in re-creating this Test Condition. Since we designed our only Test Script from this Test Condition, We can safely put TS-01 in this column.

G. SUMMARY QUESTIONS

- What is our source for Test Conditions?
- Can a Test Condition ever have more than one Test Cycle referenced to it? More than one Test Script?
- Should a Test Condition ever contain specific date information?
- What is another name for a Test Cycle?
- What is the minimum number of Test Scripts it would take to prove a Test Condition?
- What is the relationship between a Test Script Step, Validation Point, and Validation File?
- Should a Test Script ever mention in its description that it is testing a specific date?

H. . EXAMPLE FROM SURVEYPOWER

Example From SurveyPower

4.0 Test Details

4.1 TEST CONDITIONS

After a review of standard Y2K test considerations and specific date functions in the SurveyPower application, the following list of test conditions was generated that need to be tested during the certification effort. These test conditions are referenced to the appropriate test cycles and scripts:

Cond. #	Test Condition Description	Cycle	Test Data	Script	Client/Server Test
TC-01	Test SurveyPower's Client's ability save input data into text, comma-delimited files and send the created file to Simi Valley Server over WAN.				
TC-02	Test SurveyPower's Server-side's ability to expand number and date fields and append and populate the SurveyPower database				
TC-03	Test SurveyPower's Server-side ability to "trigger" and produce a Daily Report and upload it for attachment to Lotus Notes email				
TC-04	Test SurveyPower's Server-side ability to "trigger" and produce a Monthly Report and upload it for attachment to Lotus Notes email				
TC-05	Test SurveyPower's Server-side ability to "trigger" and produce an Annual Report and upload it for attachment to Lotus Notes email				
TC-06	Test SurveyPower's Server-side ability to "trigger" and upload a Mail List				

4.2 TEST CYCLES

All of the test conditions will be executed with system dates set at following cycle dates. Cycles will only be tested where they are relevant to the Test Condition to be reproduced.

These cycles are referenced to the test scripts, to establish an execution order for each system date setting:

Test Cycle #	Test Cycle Description	Scripts Executed per Cycle
CYC-01	December 1, 1999 - This test will establish the baseline for testing reconciling against all Test Conditions	
CYC-02	January 4, 1999 - First CHL business day of the year 2000. This condition will include Test Scripts that check handling of birth dates within the lower bound limit of Jan 1, 1910.	
CYC-03	January 30, 1999 – End of first month of the year 2000	
CYC-04	March 2, 1999 – First business day following Leap Year Day	
CYC-05	December 31, 1999 – End of the year 2000	
CYC-06	December 31, 2035 – Upper Bound Test (Based on Windows NT Compliance dates)	

4.3 TEST SCRIPTS AND DATA

Based on the identified test conditions, the following series of test scripts were developed to fully reproduce those conditions at chosen Test Cycle date.. These test scripts are referenced to the necessary test data to execute the script, including specific expected results and the corresponding validation file.

Script Name: **Store Data in Client in Text, comma-delimited**

Script Number: SCR-01

Platform: Client

Step #	Test Condition	Step Description	Detailed Expected Result	Validation File	Data Cross-Ref
					DAT-01

Script Name:	**Nightly Upload from Client to Server**
Script Number:	SCR-02
Platform:	Client / Server

Step #	Test Condition	Step Description	Detailed Expected Result	Validation File	Data Cross-Ref

Script Name:	**Receive nightly upload, Append & Populate SurveyPower database**
Script Number:	SCR-03
Platform:	Server

Step #	Test Condition	Step Description	Detailed Expected Result	Validation File	Data Cross-Ref

Script Name:	**Trigger and upload MailList Query to Lotus Notes**
Script Number:	SCR-04
Platform:	Client

Step #	Test Condition	Step Description	Detailed Expected Result	Validation File	Data Cross-Ref
1					
2					
3					
4					
5					

Script Name:	**Trigger and Upload Daily Report**
Script Number:	SCR-05
Platform:	Server

Step #	Test Condition	Step Description	Detailed Expected Result	Validation File	Data Cross-Ref
1					
2					
3					
4					
5					

Script Name:	**Trigger and Upload Monthly Reports to Lotus Notes**
Script Number:	SCR-06
Platform:	Server

Step #	Test Condition	Step Description	Detailed Expected Result	Validation File	Data Cross-Ref
1					
2					
3					
4					
5					

Script Name:	**Trigger and Upload Annual Report to Lotus Notes server**
Script Number:	SCR-07
Platform:	Client

Step #	Test Condition	Step Description	Detailed Expected Result	Validation File	Data Cross-Ref
1					
2					
3					
4					

Script Name: **Print all Reports by manually calling queries.**

Script Number: SCR-08

Platform: Client

Step #	Test Condition	Step Description	Detailed Expected Result	Validation File	Data Cross-Ref
1					
2					
3					
4					
5					

V. Managing Test Data

A. Managing Test Data Template

5.0 TEST DATA MANAGEMENT

5.1 Input Test Data Strategy

5.2 Output Validation Data Strategy

B. Objectives Of This Section

At the end of this section, you should be able to:

- Determine the kind of test data you will need to prepare for your test.
- Determine what is a representative data set.
- Be able to define a Naming Convention for your output validation files.
- Know what kind of standard validation output file types you should use.
- Understand how your validation files from the Validation Points in your Test Scripts will be archived.

C. Input – Data Prepared For Test

1. Test Data Preparation Strategy

Preparing the data necessary for our test is a lot easier now that we have our Test Details defined. It should be clear what kind of data we will need to run each of our Test Scripts, or whether several of them can share the same data.

Where does the data come from?
Our first determination is where does each Test Script get its data in the Production environment? If you are familiar with the application, this should be a straightforward answer. We will want to prepare test data that will simulate that production data for our test.

The first question asked, once we know what we need, "How much test data do I need?" There are some clues.

- Representative Sample - Does the data set I'm using represent a normal application production cycle? Is it necessary that it does?
- Will the test data set exercise all parts of the code processes that handle dates?
- In production, does the data come from and internal interface with another CHL application or server? From an outside data feed?
- How many data files will be required to recreate the Production environment?
- How often will it need to be the test data need to be refreshed?

Representative Sample
Answering the questions posed in the first two bullet points is a key to your whole test. Just how much test data will be representative of the production environment to completely exercise all of your date handling. Unlikely that one record would suffice but also unlikely that you need to run a full production run. Refer to your ***Date Processing Components*** on page **Error! Bookmark not defined.** and ***the Application Date Processing Flow Diagram*** on page 25 for inspiration. The test data MUST exercise all of the processes identified as "date significant" in ***Test Scope*** on page 34.

Data from another CHL application or server
If the interface itself requires testing, test data can be set up on the internal interface for testing. Plan for your data to simulate the data normally prepared for upload or download (or other processing). If you do not need to test the interface between applications, simulate the data that is entering your application (on your side of the "pipe") for the test.

Data from outside CHL
Since the Lab environment is totally isolated from the outside world, data from the outside via an interface will need to be simulated. Copying historical data and making changes in appropriate date fields to represent live data should accomplish creating this test data set.

Test Data Refresh
You may be able to plan the execution of your Test Scripts logically to move your test data set through serial steps, from script to script and/or condition to condition, aging data as it moves from function to function as a part of your test. If not, each time you introduce a change to your data that is outside the test, including refreshing it, the lab must stop and do a complete back up of the test environment for archive.

Inter-cycle Test Data Preparation
Besides adding to the clarity of your plans, this description of data refreshing for your application is important for the Lab. They must make backups of the entire environment before you begin testing and at every point where data is refreshed, or changed outside of the test itself. (And of course, a post-test backup.) Good planning on your part may help eliminate many of these backups. You'll be out of the lab sooner.

In most of the applications that the Y2K Team has seen, test data can propagate smoothly through a complete cycle with no need of a data refresh. Or, could have anyway, if it had logically been planned that way. Then the same test data set is forward dated for the next cycle and run through all of the scripts for that condition, comparing the results against the baseline set by the first cycle. And so on, through all following cycles. This will limit backups to between cycles only. In the best of all possible worlds, of course, the same test data (with no revision or refresh) could be propagated through every condition and its scripts. Check how your Test Details are setup. Are your Test Conditions and Test Scripts set up chronologically and logically?

Test Data Strategy Worksheet
In our worksheet below, write a brief description of how you would prepare Test Data for the Test Script TS-01 answering the appropriate questions above.

Test Data Strategy

2. File Naming Convention

You will want to briefly describe how your Test Data files are named. The naming convention should show the position in the script that is will take. This is really for your tester's benefit and it will show nice logical thinking to the reviewers. For example, in the scenario of the paragraph above, where test data is forward-dated between cycles, This might be a convention you could use:

TestDataSet**1.**ext	Cycle 1
TestDataSet**2.**ext	Cycle 2
TestDataSet**3.**ext	Cycle 3
TestDataSet**4.**ext	Cycle 4

If you need to use more than one data set, additional files could be identified by:

TestDataSet**1a**.ext
TestDataSet**1b**.ext
TestDataSet**1c**.ext

The important thing to remember about file naming for this test, make it logical, make it easy for the tester. Once you know the data files and have their names, you cross-reference them to the Test Scripts to show what test data is being used to run that script.

Test File Name Convention Worksheet
In this worksheet write down the name of one or two data files that might be required to support your strategy above. How would you name them? The worksheet allows room for two. That should be enough for our workshop.

File Name	Brief Description of Test Data file purpose

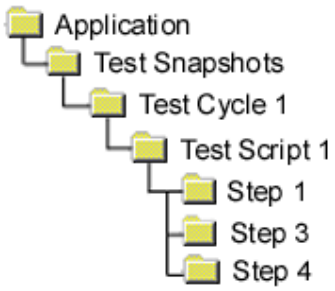
Cross-reference Test Data file to Test Script Worksheet
Now that we have a couple of data files to refer to in out Test Script TS-01, let's return to page 53 for the worksheet there on Test Data. We will want to reference our Test Data file or files to the appropriate step in the Test Script. This may only be one step or it may be several. You many need to refer to the worksheet on page 49 for the description of each step in TS-01.

D. OUTPUT – VALIDATION SNAPSHOT FILES

The other part of the Test Data Management section refers now to the test's OUTPUT files. After we have saved all of our Validation Files that prove our Validation Points, where will put them? This is practically boilerplate for all of the Y2K Test Plans since it almost the same for each. We'll take a quick look at how this process works just to understand why we need to describe our file naming convention here.

1. Validation Archive Snapshot Directory

Once your Test Plan is approved, your facilitator will prepare a directory structure to archive every item of the Certification process. **Note**: this directory, once created, cannot be changed by anyone other than your facilitator. If you believe there is some reason that a change should be made, be sure to contact your facilitator.



Your Test Plan and all of your Test Validation Files will be stored in a directory structure similar to the one illustrated here. You can see that the storage folders are organized by Test Cycles with each Script that ran under that Test Script. Finally each step that had a Validation File is stored in a Folder with the Test Script's step number.

2. Recognized Output File Types

In another directory named "Reader Tools" under your Application's name, there are a series of "Viewer" files that will open standard file types. This is in case

these files need to be viewed at a time in the future when the program files we are using today are obsolete, can still be accessed.

These standard Reader Tools are:

- Adobe Acrobat Reader 3.01
- Microsoft Excel 97
- Microsoft PowerPoint 97
- Microsoft Word 97

These are usually enough readers to open 99% of the Validation Files captured. If, for some reason, you need to use a different file type to capture a Validation Point, just make sure you can obtain a reader tool that will open it up and that your facilitator receives it to put in your specific Reader Tools directory.

All of the above is just to describe the process of archiving, not something you have to write about in the Test Plan. However, understanding the number of files that will be generated and the "Reader" file they must correspond to, should help you determine a file naming convention for your output Validation Files.

File Naming Convention
More than likely there will be a great number of Validation Files that will need to be kept track of. All well and good that they will be archived in separate folders for each step but we all know that files can get mis-filed. It's the nature of things. A good file naming convention can eliminate a lot of that. While you are free to invent your own names, we can suggest one for you.

CnnTnnStepnn.ext

Cnn- Test Cycle with cycle number, ***Tnn*** – Test Script with script number, ***Stepnn*** – The step number. The "***.ext***", of course, refers to one of the standard file types like .doc, .txt. .ppt, and so on. This makes certain that any file that gets misplaced can easily be noticed and re-located. You are free to develop your own naming convention, especially if you used different identifiers for your Test Detail elements but whatever you choose should help in file location.

Validation File Archive Worksheet
In the space below, describe what kind of file types you will be using and the file naming convention you will use. Go into more detail if you intend to use a file type that is non-standard, and what Reader file you will be supplying.

Validation Snapshot Files Strategy

E. SUMMARY QUESTIONS

What would you consider a Representative Test Data set to be?

Under what circumstances would a Test Script not require Test Data?

Under your application's archive directory, what is the root directory name that determines the directory structure?

What are the four standard file types that can be used to format Validation Files?

Why is a Naming Convention for Validation Files important?

F. EXAMPLE FROM SURVEYPOWER

Example from SurveyPower

5.1 Test Data Management Approach

To test the ability of SurveyPower Client to store data, we will prepare automated input files that will simulate input from users. (***Data01a.txt***)

The same data set stored above will be used to perform the nightly Upload task from Client to Server

The same data set will be used to expand number and date fields to be appended to the SurveyPower database.

The same data set will be used to create queries, triggers, and reports.

An electronic mail list will be prepared for trigger and Upload. (***MailData.dat)***

A data set similar to the data set above to pre-populate the SurveyPower database so that appending and populating can be shown. (***Data01b.txt***)

Between Cycle Dates, data sets will be aged appropriate to the cycle.

Cycle 2	(***Data02a.txt***)	(***Data01b.txt***)
Cycle 3	(***Data03a.txt***)	(***Data01b.txt***)
Cycle 4	(***Data04a.txt***)	(***Data01b.txt***)
Cycle 5	(***Data05a.txt***)	(***Data01b.txt***)
Cycle 6	(***Data06a.txt***)	(***Data01b.txt***)

5.2 VALIDATION DATA ARCHIVE APPROACH

For each test condition identified, a specific expected result is presented in the test script step that executes that particular condition. This step will also reference a data file (either a report, screenprint, or SQL output) that validates this expected result.

These data files will be organized logically in the "Test Snapshots" directory that can be archived for certification purposes. Files will be named for the Test Script and Test Cycle and Script Step to which they refer, following the generic example below, where .ext is the file type saved.:

a) CnnTSnnStepnn.ext

VI. Test Environment

It may be stating the obvious to point out that the purpose of the Lab hardware/software setup is to simulate the Production environment that the application normally works in. However, starting from an obvious point and once again looking back to our Description section, we can extrapolate a lot. Knowing the environment we want coupled with knowing the Test Conditions that we want to test should give us all of the answers or, at least the right questions. You may have to ask some questions but you'll know the right questions to ask. The Lab Coordinator will see to it that your Lab requirements are approved and met well before your lab date.

A. Test Environment Section Template

6.0 TEST ENVIRONMENT

6.1 Client-side Environment

6.1.1 Client Software Specification

6.1.2 Client Software Configuration

6.1.3 Client Hardware Specification

6.1.3 Client Hardware Configuration

6.2 Server-side Environment

6.2.1 Server Software Specification

6.2.2 Server Setup Configuration

6.2.3 Server Hardware Specification

6.2.4 Server Hardware Configuration

6.3 Other Hardware/Software/Interfaces Needed

6.4 Lab Environment Installation Procedures

B. Objectives Of This Section

At the end of this section, you should be able to:

- Define your Client-side Test Environment
- Define your Server-side Test Environment
- Define any other needs for your Test Environment

C. TEST ENVIRONMENT CLIENT-SIDE

1. Client Hardware Specification

What is the minimum required hardware needed for the client workstation. The answer is not just a PC. We must know the exact specifications for this PC. On the side of most shrink-wrap commercial packages they list the specifications necessary to operate the program. This is where we want to start. Be certain you list all of the obvious things like Megabytes of RAM, Hard Drive size, Processor, but also list any special cards on the bus, any special configurations of the PC's hardware. You cannot be too specific here.

If you need a printer attached to your workstation, mention it here and the type of printer needed. The key here is that hardware outside of your PC considered to be client-side must attach directly into the PC's bus.

Client Hardware Workbook

For the sake of our workshop, lets just get some basics down here. In the five rows below, write down Processor, amount of RAM, size of hard drive space, Monitor video card, and network card minimum requirements to run your application.

Client-side Hardware for Test Environment

2. Client Hardware Configuration

The Client workstation may require some specific hardware or firmware setup that wouldn't logically be covered in the specification. Anything that refers to how the Client's hardware should be configured should be mentioned here.

For the sake of brevity, we won't put anything in a worksheet here.

3. Client Software Specification

Here we need to write down our application's name if it resides on the client workstation and all of the software required to support the app. Be certain to include Version numbers, Service Packs, Patches, and any other identifying information necessary for each entry.

Client Software Specification Worksheet
Let's give it a try. We shouldn't have any trouble thinking of two or three. Don't forget the Operating System.

Client-side Software for Test Environment

4. Client Software Configuration

The software on the client may require something beyond just being installed. Excel or MS Word macros may need to be invoked, Access triggers or queries set, other configurations to the client's software may have to be set.

Once again, for brevity we will skip the worksheet.

D. TEST ENVIRONMENT SOFTWARE

1. Server(s) Software Specification

The same type of list that was used above for the client-side will be needed for the server-side software. Once again, be absolutely certain that you have listed every piece of software and it's version number, Service Pack, Patch, or other distinctive attribute. If your application or part of it lives on the server, be sure to list that component.

Server Software Worksheet
Let's write down up to four server-side software components that will need to reside on your server (or servers).

Server-side Software for Test Environment

2. Server(s) Software Configuration

As above, there may be configurations that need to be made to the software before testing can begin. Maybe SQL Server databases may need to be populated or some other configuration may be required.

We won't use a worksheet here.

3. Server(s) Hardware Specification

You have probably already guessed how to fill this list out. Once again, any hardware attached to the server directly through the bus or internally in the box is a part of the server-side software. You may run to more than one server. That's fine, just add another section in the server-side specification duplicating the first, number them and fill them out for each server.

Worksheet

Write down up to four server-side minimum required hardware specifications in the space below.

Client-side Hardware for Test Environment

4. Server(s) Hardware Configuration

Any particular setup required for the Server to simulate the production environment should be listed here.

No worksheet for this topic.

E. OTHER HARDWARE/SOFTWARE/INTERFACES NEEDED

This is another catchall category that will not apply to many applications. However, you may have a data feed to an IBM AS400. While this is an interface, it is possible for an AS400 to be available for your testing in the lab. Anything that you might need, write it here. If you have a question about a piece of hardware or software, contact your facilitator to help answer your question.

Other Hardware/Software/Interface Worksheet

If you can think of anything of this nature that your application requires, write it below, otherwise, skip it.

Client-side Hardware for Test Environment

F. APPLICATION INSTALLATION

While your Lab Coordinator will help you with any technical support needed, you will install your own application as well as test it. The reason that you write down the steps involved here, is that at some future time, this installation may need to be recreated and there might not be anyone around who knows how to do it. To add further due diligence to our responsibility to make a re-createable Year 2000 test, a person unfamiliar with your application should be able to follow your written directions here to install your application

Application Installation Worksheet

Give it a try. How do you install your application? It should be by following the steps you write below.

	Step Installation Instructions	Expected Result
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		

G. SUMMARY QUESTIONS

Is the software description SQL Server version 6.5 enough? If not, what else is needed?

If your application used Stored Procedures, where would you list them?

If your application used a batch file to make periodic calls to the server, where would you list this component?

If part of your application's Test Environment required a Fax machine physically located on a network node, where would you list this hardware component? How would your figure out where the software driver software was located?

H. EXAMPLE FROM SURVEYPOWER

Example From SurveyPower

6.0 Test Environment

6.1 LAB ENVIRONMENT REQUIREMENTS
Certification testing will be performed in the lab environment provided by the Y2K Client/Server team. The test environment will include a single Compact Professional Workstation 5000 and a single Compact Proliant 4000 series server.

Client
Minimum Hardware Configuration

- 1 Pentium Pro 200 MHz processor
- 32 MBytes memory
- 50 Megabytes of disk drive space, configured on drive "C:"
- 1 network card

Operating System Configuration
The following software will be installed:

- Microsoft Windows NT Workstation 4.5 with Service Patch 3 (build nnnn)
- A printer driver suitable for the printer attached to the test server will be installed

The system will be named "Client". The operating system will be configured with a 32 MByte page file on logical drive C:. The time zone will be set to Pacific Time.

System Software Configuration
The following software will be installed:

- Microsoft Office 97 Professional (including Microsoft Access).

Server
Hardware Configuration
The server will be equipped with at least:

- 1 Pentium Pro 200 MHz processor
- 128MB memory
- 1 physical 1 GByte disk drive space, configured on logical drive "C:"
- 1 printer suitable for printing text reports
- 1 network card

Operating System Configuration

The following software will be installed:

- Microsoft Windows NT Server 4.0 with Service Patch 3 (build 1381)
- A printer driver suitable for the attached printer will be installed

The system will be named "SurveyPowerDB". The operating system will be configured with a 1 GByte page file on logical drive C:. The time zone will be set to Pacific Time.

SQL Server Configuration

The following software will be installed:

- Microsoft SQL Server 6.5 with Service Patch 4 (product version 6.50.258)

It will be installed with the following options selected:

- Binary sort order (Sort Order 33, Character Sort Order, Case Insensitive, Upper Case Preference)
- 850 Multilingual character set (CP 437, US English Character Set)
- master device with 500 Mbyte (default)

It will be configured as follows:

SQL Server Configuration

Parameter	Setting
Allow updates	0
Backup buffer size	1
Backup threads	5
Cursor threshold	-1
Database size	2
Default language	0
Default sortorder id	40
fill factor	0
free buffers	12500
hash buckets	7993
language in cache	3
LE threshold maximum	200
LE threshold minimum	20
LE threshold percent	0
locks	100000
logwrite sleep (ms)	10
max async IO	16
max lazywrite IO	16
max worker threads	255
media retention	0
Memory	250000
nested triggers	1
network packet size	4096
open databases	20
open objects	2000
priority boost	1
procedure cache	1
RA cache hit limit	4
RA cache miss limit	3
RA delay	15

RA pre-fetches	3
RA slots per thread	6
RA worker threads	16
recovery flags	0
recovery interval	5
remote access	0
remote login timeout	5
remote query timeout	0
resource timeout	10
set working set size	0
show advanced option	1
SMP concurrency	0
sort pages	64
spin counter	10000
tempdb in ram (MB)	400
user connections	64

6.2 LAB ENVIRONMENT INSTALLATION PROCEDURES

Client

Application Software Configuration

- Insert SurveyPower installation diskette into drive A:
- From Start->Run option, run the following command: A:\ClientSetup.exe. This will run a routine that copies all the necessary files to clients machine and adds SurveyPower icon to it.
- By going to Control Panel and using ODBC32 setup program, add a system DSN for "SurveyPowerDB" to the client machine.

Server

Application Software Configuration

- The complete file structure and all files found in the "A:\SurveyPowerServer" directory of the installation diskette will be copied onto the test server under a "common" share created off the root of logical drive C: of the test server.
- A "\Jobs\Log" directory will be created off the root of logical drive C: of the test server. No files will be created.

VII. Executive Summary

Though our readers will read our application's Y2K Test Plan from front to back, the information we present right here will set the background for everything that is to come. That is why it is best to write the beginning from the perspective of the end. It's Monday morning quarterbacking, and in this case its allowed, in fact recommended! Setting our target after we've reached our goal should result in 100% accuracy.

A. Objectives Of This Section

At the end of this section, you should be able to:

- Summarize all of the sections of your test plan within a paragraph or two.
- Give a reader an overview of what's special about this test plan
- Highlight your approach to this document so that the reader can easily follow the following sections.

B. Summarizing The Test Plan

Here we are at the end and we're just starting on the beginning….

The purpose of this section is to quickly bring the casual reader up to speed on what you are planning to do. A really well written Executive Summary gives any reader a nice feeling of what he's going to be looking at. Also, a cursory glance will let a reader know whether he needs to proceed further into the plan. For the auditors, facilitators and others who are likely unfamiliar with your application, they will have an immediate grasp of how your are thinking about the test for your application.

We have covered five previous sections. What we want to do here is to draw from these sections to paint a quick picture of the app and your approach to testing it. Going forward from the first section we covered:

Description: The quickest possible description of your application. You should be able to do it in no more than a paragraph. A sentence is better. Any more than a paragraph is probably redundant considering on the next page you have expanded on this subject. After that, don't mention the obvious. Instead, mention what might be unusual about your application from the inventory perspective this section gives us. Maybe an unusual interface that presents a challenge to your testing environment. Perhaps a special piece of hardware or software or a Business Function that has special significance in a Y2K test.

Description Summary Worksheet
Give it a try! See if you summarize the description in a few sentences. Glance through your Description section and grab a highlight or two.

Summarize Description

Test Scope – Summarize in a statement or two what you chose to test and what you didn't and why. Again, don't restate the obvious. If there are any unusual Assumptions that will be made, say why and what you plan to do about it.

Test Scope Summary Worksheet
See if you can summarize your Test Scope in a statement or two, below.

Summarize Test Scope

Test Details – If you wrote the Overview of your Test Details, this is easy. Just summarize what you wrote there but only if there is something to emphasize. Otherwise you will want to point out the significant goal or goals of your test details and why these elements will cover them.

Test Details Emphasis Worksheet
For our workshop, come up with a statement you can make about what our Test Details will accomplish.

Summarize Test Details

Test Data Strategy – There may nothing to write about this topic if there is nothing out of the ordinary about your test data strategy. On the other hand, a quick description of why the test data you chose will make the test effective will help the reader grasp any challenges to be overcome. Your plan for archiving test data probably does not need to be mentioned since every application's plan will be so similar.

Test Data Highlight Worksheet
If you think you have something you can say about your test data strategy, write it in the space below.

Summarize Test Data Strategy

Test Environment – There may not be much to mention about the Test Environment. On the other hand if there was some special problem in how to set up the test environment or something unusual about the setup, summarize it her.

Test Environment Highlight Worksheet
We have reserved the space below to write a statement or two about your Test Environment setup.

Summarize Test Environment Strategy

C. SUMMARY QUESTIONS

- What is the purpose of the Executive Summary section?
- Why did we choose to write it last instead of first?
- What is the general theme of this summary?

D. EXAMPLE FROM SURVEYPOWER

Example from SurveyPower

1.0 Executive Summary

"SurveyPower," a client/server application used to capture and report survey information for Countrywide, will be tested in a Y2K-compliant lab environment to demonstrate the application's overall ability to function properly in the year 2000. This test will address date-related functionality in both the client and server modules of the application. The test is organized into a series of cycles testing various dates known to be at risk for potential year 2000 problems. Each test condition will be independently verified with validation data that can be archived along with this plan for future reference.

VIII. In Closing…

If you were very familiar with your application during this workshop, you have gone a long way toward writing your Y2K Test Plan. Take the information you have started and continue the process, using the Test Plan shell to transfer the information into. If you weren't too familiar with the application you are writing the test plan for, even if you didn't know all the answers, you now know all the questions. Just get them answered and you'll be well on your way to an excellent Y2K Test Plan.

A Preliminary Walkthrough will be scheduled shortly, to review your first draft of the Test Plan. It is a good idea for the test plan author to make it to that meeting along with any other concerned parties.

This workshop, by its nature, is somewhat generic even though the attendees are working on very different applications. While consistency is important it is sometimes difficult to achieve. In the Preliminary Walkthrough all of our attention is specifically on your application and how we can help you improve its clarity and efficiency and to check for any holes that will make reviewers think there is something missing.

Thank you for your attendance.

IX. Resources

Various numbers, AMS entries, any other information useful to a Test Plan Author

A. YEAR 2000 CERTIFICATION TEST PLAN OUTLINE

1.0 EXECUTIVE SUMMARY
2.0 APPLICATION DESCRIPTION
- **2.1 Application Description**
- **2.2 Year 2000 Readiness**
- **2.3 Primary Business Function Breakdown**
- **2.4 Application Technical/Hardware Environment**
- **2.5 Technical modules or components**
  - ***2.4.1 Date Processing Components***
  - ***2.4.2 Hardware Components and Software Platforms***
- **2.6 Interface Inventory**
  - ***2.5.1 The Input Interfaces Grid***
  - ***2.5.2 The Output Interfaces Grid***
- **2.7 Application Date Process Flow Diagram**
- **2.8 Application Architecture Diagram**

3.0 TEST SCOPE
- **3.1 Application Modules and Business Functions Tested – Action List**
- **3.2 Assumptions**
  - ***3.2.4 Technical Components that will Not Be Tested***
  - ***3.2.5 3rd Party Interfaces, Components, & Utilities***
  - ***3.2.6 Functional Assumptions***

4.0 TEST DETAILS
- **4.1 Test Conditions**
- **4.2 Test Cycles**
- **4.3 Test Scripts**

5.0 TEST DATA MANAGEMENT
- **5.1 Input Test Data Strategy**
- **5.2 Output Validation Data Strategy**

6.0 TEST ENVIRONMENT
- **6.1 Client-side Environment**
  - ***6.1.1 Client Software Specification***
  - ***6.1.4 Client Software Configuration***
  - ***6.1.5 Client Hardware Specification***
  - ***6.1.3 Client Hardware Configuration***
- **6.2 Server-side Environment**
  - ***6.2.1 Server Software Specification***
  - ***6.2.2 Server Setup Configuration***
  - ***6.2.5 Server Hardware Specification***
  - ***6.2.6 Server Hardware Configuration***
- **6.3 Other Hardware/Software/Interfaces Needed**
- **6.4 Lab Environment Installation Procedures**

7.0 APPENDIX 1 (IF NEEDED)