AccommoDATE

A Date Simulation Testing Tool
for
Unisys 2200 Series and ClearPath HMP/IX Environments

(White Paper)

Introduction

Date Simulation Testing for applications in development on mainframe systems can be a highly cost-intensive process, requiring legions of knowledgeable programmers. A project that started with a requirement as seemingly simple as verifying a smooth date transition at the fiscal year-end can rapidly become bigger and more costly than anticipated. Costs associated with such testing include:

- ✓ System downtime
- ✓ Modifications or corrections to problem code
- ✓ Verification and Validation Testing
- ✓ Acceptance and Implementation.

Applications must be tested not only to make sure the modifications will work as intended after the target date, but also to make sure no new bugs were introduced during code modification or correction. Implementing a comprehensive and workable test plan is essential to ensure that the applications will work correctly when installed in production.

This article presents an overview of Date Simulation Testing, and presents an introduction to FCI's AccommoDATE product now used by over 60 businesses to streamline their Date Simulation Testing process in the Unisys 2200 Series and ClearPath HMP/IX environments.

Overview of Date Simulation Testing

The conventional definition for testing is "the execution of a software program or system for the purpose of detecting errors and gathering information about it." Within the context of Date Simulation Testing, that definition is broadened to include initial evaluation testing. Thus, date simulation testing is performed in two phases: evaluation testing and validation testing. These two phases bracket the actual changes that are made to the code. Further, the validation testing phase includes regression testing, which determines if anything was broken by the modifications.

During evaluation testing (phase one), the software or system is subjected to a series of test cases designed to analyze and evaluate its current performance with regard to the correct use and processing of date information. Date transition sensitivities are located, thus indicating where changes in the software are required. Evaluation testing also determines the risk associated with whether or not the software will adequately support the desired user capability or mission. This risk must then be further evaluated for acceptability; and, if the risk is not acceptable, a mitigation strategy must be designed.

Based on the results of evaluation testing, the necessary code changes or modifications are made, followed by validation testing (phase two). Testing code modifications is the most critical aspect of a successful date simulation testing program. First, regression testing is performed to ensure that the renovated software operates as well as it did prior to the changes. Then, validation is performed to verify that the software or system will operate at the transition to the target date(s) and afterwards. To do this, the software or system is tested in a "time-advanced" environment. At least one test is required at each of the identified critical dates, as well as one for each relevant date horizon at each critical date.

Important Considerations for Date Simulation Testing

The emphasis of Date Simulation Testing should be on functional tests. Such tests exercise the programs from a real-world perspective while addressing the most critical, highest-risk business functions. Code with the greatest date sensitivity and business impact should get the most attention. Functional testing focuses the effort on the most mission-critical areas of the code using "critical date transitions" in a manner that matches actual software usage before and after the defined target dates.

For a software program or system to be validated as functional for a specific transition date, it must correctly process dates before, during, and after the target date. It must correctly process the transition itself (that is, midnight of the target date) and must continue to correctly process dates on both sides of the transition point.

The software or system must operate properly and provide correct and unambiguous date information as it encounters other critical date transitions. Critical date transitions are those for which evidence, either hard or circumstantial, exists that date-related processes might encounter error. Some examples of critical date transitions are:

- ✓ December 31 to January 1 of the target year
- ✓ February 28 to February 29 of a leap-year
- ✓ February 29 to March 1 of a leap year
- ✓ December 31 to January 1 of subsequent years

The above are the minimum set of critical date transitions that should be tested on all software applications and systems under development. Other critical date transitions may exist for specific software, such as financial systems that work with fiscal years. These may be date sensitive to the fiscal year rollover. For systems that follow the Government fiscal year, the date transition from September 30 to October 1 should be considered. Systems that follow different fiscal years should adjust the dates accordingly.

When testing the integrity of target date transitions, the software being evaluated must be shown to receive, calculate, display, print, and store dates unambiguously. This means that the software program or system must accept date-oriented input and interpret it correctly. It must provide date-related data to the outside world in a format that is at once acceptable, understandable, and correct. And it must display and print date information in a fashion that is understandable and unambiguous to its users. Questions to ask include: Where are dates accepted as input? How are

dates stored by the application? That is, are they stored as dates or strings or in some other format? Are there conversion functions to go back and forth to other date storage locations? What date calculations are performed? Is there a leap year calculation function? How are the dates formatted for display? What are the rules for converting both to and from two digit years? Do users of the system ever enter a special date that has an understood meaning different from an actual date, such as 1/1/11?

The driving motivation behind a date testing strategy recognizes that not all foreseeable problem dates can be given equal time and emphasis. High risk systems require more extensive test data development and regression (functional equivalency) testing, with an emphasis on special case testing. Special case testing ensures that key business dates are accommodated in a combination of transactions and situations.

To maximize testing effectiveness, testers should set up test environments with the capabilities required for date-sensitive testing, such as the ability to set the system clock to future dates. Using date-simulation tools to perform simulated testing of a target transition date is much simpler than setting the actual system clock—a function that not only involves careful coordination among testers, but may also require having to solve licensing, password and security issues.

The good news is that there is already a proven date simulation testing tool available for use in the Unisys 2200 Series and ClearPath HMP/IX environments. It was created by Formula Consultants Incorporated, and is called AccommoDATE.

What is AccommoDATE?

AccommoDATE was originally created to fulfill the need for a Year 2000 (Y2K) testing tool. When AccommoDATE was introduced, it assisted businesses facing extensive Y2K testing by enabling them to set a simulated date for program testing without having to reboot and without affecting costly production runs. It enabled users who had a single system, or those who had to use time on a production system, to test their Y2K software even when that system was booted with the current date.

Because of AccommoDATE's robust ability to test batch runs, demand runs, and TIP transactions under a date other than the current system date, this unique date simulation tool has become a valuable asset for businesses beyond the Y2K transition.

How AccommoDATE Helps

Date simulation testing (the Y2K problem being but one example) forces many 2200 users to dedicate computer resources to validate code changes. However, some businesses cannot afford to re-allocate dedicated machine time for production runs in order to test their applications. AccommoDATE helps businesses who are faced with this quandary by enabling them to set a simulated date for program testing without having to reboot and without affecting mission critical production runs.

AccommoDATE can assist users who simply need to test programs with dates other than the current date, such as quarterly or year-end programs. AccommoDATE can also be used to allow programs to run with a simulated date from the past. This means if a critical bug is discovered on the target transition date in a program that was thought to be correctly coded, the program could be run with a simulated prior date until a modification is made to eliminate the error.

The primary benefits of AccommoDATE are:

- ✓ First, AccommoDATE enables users to test a variety of programs over a range of dates, without arranging special time on a computer booted with a simulated test date. This means that different programs may be tested simultaneously, each using different dates.

- ✓ Second, AccommoDATE requires no changes to existing runstreams or programs. Are you inserting special code to "date-test" your software, only to remove it after testing is completed? With AccommoDATE installed on your system, you can place your programs directly into production after testing, without code modifications.

How does AccommoDATE work?

Through use of the AccommoDATE ADMIN utility, a file is produced to control which run identifiers and TIP programs will receive the alternative date. Input to ADMIN includes the future or past date to be used for the run or transaction program identifier or simply an offset of today's date. When AccommoDATE is activated by a keyin from a system console, the EXEC reads the file and determines which IDs are candidates for simulation.

Programmers and testers use the AccommoDATE Administrator utility to create a table of runids and/or TIP transaction names, and simulated dates or offsets. The date can be entered either as an absolute date or a relative date, but is saved as a relative date. Any date, past or future, can be specified in the table. This table is saved in a separate system file. The file can store target date values, which can be changed when building the system. Each entry in the table defines a single run id (or transaction id) and its associated date.

Prior to beginning a test session, the system operator activates AccommoDATE. When a run (whose id is entered in the table) requests a date through one of the standard date-fetch ERs (TDATE$, DATE$, DWTIME$), the system returns a date as specified in the table. Based on the ER used, the date is returned in the same format as the request. The dates entered in the table will progress just as the actual date would progress. That is, if AccommoDATE is on at midnight, the dates entered in the table are increased by one day.

The AccommoDATE date simulation is enabled and disabled with system console keyins. When the keyin is issued to enable AccommoDATE, the file created with the ADMIN utility is read into the AccommoDATE EXEC data bank and AccommoDATE is then enabled. The file can be re-read at anytime, without disabling AccommoDATE. New console keyins are introduced in AccommoDATE to turn it ON and OFF, or display the status of the feature.

Which systems can run AccommoDATE?

AccommoDATE is supported on the following system base levels.

SB / HMP / CP Level	Exec Levels	PME
SB 4	43R8	SB4
SB 5	44R4, 44R4A, 44R4B, 44R4C, 44R4D	SB5BASE, SB5DELTAS
SB 6	45R1, 45R1A, 45R1B, 45R1C, 45R1D	SB6
SB 7, HMP 3.0	45R2	SB7
HMP 3.1	45R2A	HMP3
HMP 4	46R1, 46R1A,46R1B,46R1C,46R1D	HMP4
HMP 5	46R2, 46R2A	HMP5
HMP 6	46R3	HMP6
HMP 6.1S	46R4, 46R4S	HMP61S
HMP 6.1V	46R5, 46R5S	HMP61V
CP OS 2200 7.0	46R6	HMP7
CP OS 2200 7.1	46R7	HMP71
CP OS 2200 8.0	47R1	HMP8
CP OS 2200 8.1	47R2, 47R2A	HMP81
CP OS 2200 9.0	47R3, 47R4, 47R4B	HMP9
CP OS 2200 10.0	47R5, 47R5A	HMP10
CP OS 2200 10.1	47R5B, 47R5C	HMP10-1
CP OS 2200 11.0	47R5D	HMP11
CP OS 2200 11.1	48R1	HMP11-1

Installation is performed through the standard COMUS/SOLAR process and requires changes to OS 1100. UCS object module execution must be supported. This includes support for extended mode by OS 1100 and installation of the LINK and URTS products. Care must be exercised when testing databases to avoid entering this date information into a live audit trail. Typically, this means setting up a separate application within the system to do the testing.

Summary and Conclusions

Special care must be taken to exhaustively test mission critical applications under development to make sure they will behave correctly at key future transition dates. This is especially true when files are expanded after fielding the application, and the chance of introducing new errors is greater. Utilization of AccommoDATE, a proven date simulation tool, allows businesses to segment their operational systems and test future transition dates without system downtime. AccommoDATE was not only the answer to the Y2K problem, but continues to be a valuable solution for businesses developing applications in the Unisys 2200 and ClearPath HMP/IX environments.

You can reach Formula Consultants Inc. at (714) 778-0123 or send us a web-mail message.