

SmartTest400™ – Overview

Why is Automated Testing Important?

Because, software bugs are costing the world economy billions according to the National Institute of Standards and Technology (Nist).

Testing has never had the grand appeal that adorns some areas of the computer industry. However, industry experts all agree that IT companies need to implement an effective and comprehensive testing strategy to ensure key applications continue to function after general maintenance, new developments or upgrades.

It is easy to cut corners in the testing phase; most projects do. But this will inevitably lead to increased maintenance, loss of production time and even company profits.

It is estimated that 25 to 35 percent of the entire software development life cycle should be allocated to software testing. The benefits gained from automating the development and change management phases of the life cycle also apply to software testing.

Important as unit testing is, it fails to test the overall functionality of an application as a "system". Just because each functional component works as designed through the construction of its unit tests, that doesn't mean that those components, when brought together into a coherent application, will work together to meet the system's overall functional requirements.

System-wide regression and functional testing are vital phases in the deployment of any application. However, the tools to enable full system testing are either cost prohibitive or overly complex, often to the point that system-wide testing is done only cursorily or a human being is forced to test each functional area manually.

Human testing can never (and arguably *should* never) be completely removed from the testing process. However, improving the testing, by using automation, of key business processes (order entry – billing – distribution) can free up time to allow programmers and testers to focus on specific changes or areas that can't be automated.

Figures from the Software Engineering Institute back this up. It estimates that it costs roughly £15,000.00 to manually clean every 1,000 lines of code, making manual testing impractical for most companies.

SmartTest400 has been designed and developed to meet the growing need to find a more effective and automated testing strategy for iSeries applications, without changing the current development methodology or establishing a separate dedicated testing department.

SmartTest400 will enable companies to automate over 80 percent of their functional and regression testing. As a result this will save considerable time and money and provide a better level of service to the end users.



www.thenon.com

Many of the world leading companies use Thenon's products to change manage and test their software.

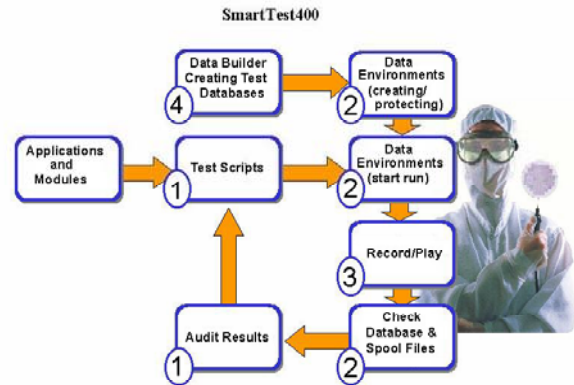
Thenon – designers of SEE/Change, the leading iSeries change management product.

SmartTest400™ – Overview

What is SmartTest400?

SmartTest400 is the route to automatic, repeatable, accurate and efficient testing. It is a superior testing strategy that, within days, will start to improve software quality and save development time.

SmartTest400 is easy to learn and easy to use, but still ensures that testing is done thoroughly every time you make a change. The sooner errors are detected, the more time you save. Even without auditors looking over your shoulder, thorough testing just makes good business sense.



SmartTest400 Organisation/Configuration: Allows the testing life cycle to be managed by creating applications, modules and projects and assigning them to testing teams. Within an application, test scripts can be created that drive the testing life cycle by using the testing modules:

1. **SmartTest400 Test Scripts:** Enables the tests to be created, written, documented, executed and audited. The script manager has access to all the other modules.
2. **SmartTest400 Data Environments:** Allows testing environments to be configured and files protected. During testing, checkpoints can be created that allow environments (databases) to be reset or advanced to a particular point in time, allowing test scripts to be rerun.
3. **SmartTest400 Record and Play:** Allows key business processes to be recorded and played back, to automate both functional and regression testing of iSeries applications. The record and play module has a full range of audit reporting and a maintenance function that enables new screens and fields to be added to test cases
4. **SmartTest400 Data Builder:** Provides an easy to use method of creating a testing database based on live data. Data Builder provides automatic access to the links between the files and fields, to enable build cases to be generated internally.

This is software your staff will use from day one!



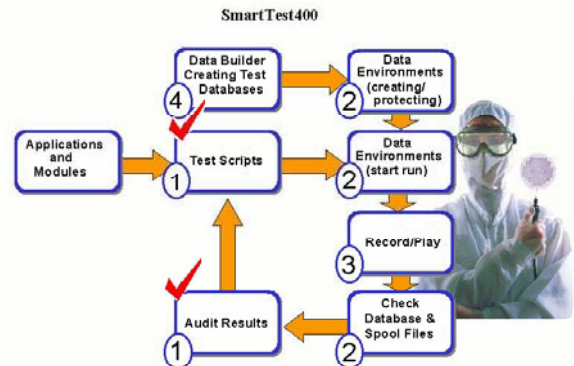
www.thenon.com

Many of the world leading companies use Thenon's products to change manage and test their software.

Thenon – designers of SEE/Change, the leading iSeries change management product.

SmartTest400™ 1 – Test Scripts

Test Scripts harness the power of the other SmartTest400 modules by enabling tests to be documented, written, executed and audited. For a test to be meaningful it should be repeatable and have an expected result. SmartTest400 provides repeatability through test scripts. You can use a test script to monitor database files, replay a record and play event, check a file rule, compare a database member, check spool file output and produce a signoff report for the test manager.



Test Scripts hold the knowledge of a test enabling it to be rerun time and time again, even if the creator has moved on.

Creating and Documenting Test Scripts: Test scripts can be created against an application (regression test) or modules (functional test). The test script is linked to a test type and a data environment. Test script documentation can be added or a link to a word document introduced.

Writing Test Scripts: The test script has access to all the SmartTest400 modules to enable the test to be defined, with prompt facilities to suggest appropriate test commands to use. Any record and play cases created will be stored with the test script. The test script is a collection of command macros that enable a test to be defined at a number of levels. These include 1. Set-up Commands 2. Test Commands 3. Test Successful Commands 4. Test Unsuccessful Commands 5. `Housekeeping and Cleanup Commands.

Executing Test Scripts: The test script can be executed in batch or run interactively. If run interactively, a script progress screen is available to monitor the script.

Creating Test Script User Variables and Input Files: Test scripts can be driven by data stored in user variables and database files. This enables a test script to run in Iterations mode, allowing many different types of tests to be performed from a single test script. For example, an input file might contain many combinations of customer numbers and order types; the test script would test each combination.

Check Results Printing Audit Reports: Once the test script has been run, a comprehensive set of results are available that enable the user to drill down and inspect the actual screens replayed, database transactions and spool files generated. Management and audit reports are preserved with the test script run until purged by SmartTest400.



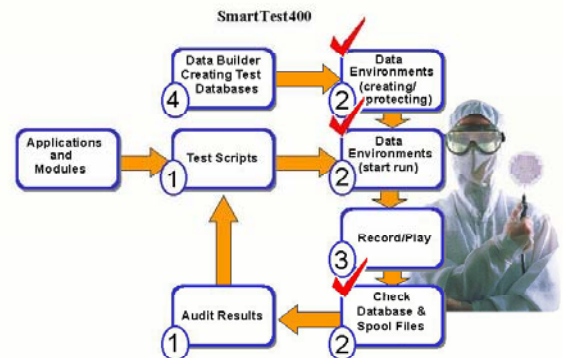
www.thenon.com

Many of the world leading companies use Thenon's products to change manage and test their software.

Thenon – designers of SEE/Change, the leading iSeries change management product.

SmartTest400™ 2 – Data Environments

Data Environments enable testing environments to be managed and protected ready for testing to commence. During testing, logical checkpoints are created that allow testing environments to be reset or advanced, allowing test scripts to be rerun. Once a test script has executed, the database effects can be viewed, showing the before and after image of each record that has changed; including program reference information. Additional functionality that allocates testing environments and refreshes libraries when file formats change, simplify the management of the testing environments.



Data Environment commands can also be embedded within the test scripts, so you can test the impact on the database concurrently with testing the impact on the user interface.

Creating Environments & Protection: Testing environments can be made up of one or more data libraries. Once configured, they can be protected (journalled) so that the testing environment can be reset or advanced after a test script has run.

Starting a Data Run: Each time a user requests a test, a data run is started. From that point, until the data run is closed, all the database effects will be recorded to that run instance. Any number of data runs can be active at any time in multi user testing environments.

Checkpoint Management: Each time a data run is started, a new checkpoint is created if no other data runs are open. Checkpoints mark the logical testing boundaries within the database and provide pointers to rollback or advance the database. Checkpoints are used to manage and control the integrity of a multi user testing environment.

Data Rules: Data rules can be constructed to check specific records and fields in a database to determine if a test has passed or failed.

Compare Database Files: Physical or logical files can be compared and the differences displayed or printed.

Compare Spool Files: Spool files on the system or in database format can be compared and the differences displayed or printed.

Change Management Interface: Commands are available to refresh a testing environment configured in SmartTest400 from within a change management system, when file formats change.

Data Environments Advanced Program Interfaces (APIs): SmartTest400's open design means Data Environments commands can be used within other SmartTest400 modules and PDM.



www.thenon.com

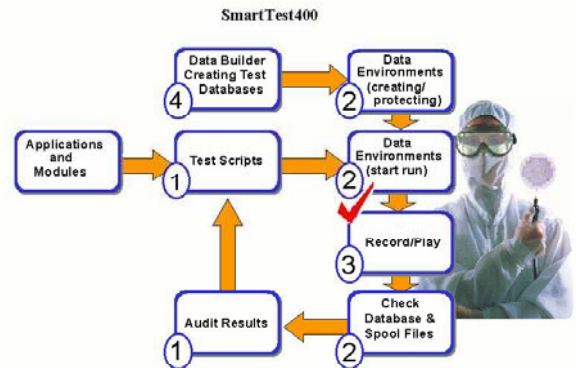
Many of the world leading companies use Thenon's products to change manage and test their software.

Thenon – designers of SEE/Change, the leading iSeries change management product.

SmartTest400™ 3 – Record & Play

Record and Play automates the manual task of testing interactive processes for business analysts, developers, testing departments and users.

You can build test cases incrementally by stacking scripts that include commands, and playing them back, as a set, any time you need to test your application. You can even call Data Environment and Data Builder commands from Record and Play, so you can reliably reconstruct the correct starting point in the database and test the impact on the database while testing the user interface.



Testing Interactive Processes

Screen Capture: Captures screen images, both those sent by the host, and the data entered by the user, and stores them in a database file for later replay. The device used for capturing the images may be any device that can be attached to an iSeries.

Screen Replay: Once a test script sequence has been recorded, it may be replayed in interactive or batch mode. As screens are replayed, they are compared to the original to ensure that the test is proceeding successfully. Record and Play highlights differences between the before and after screens, and the tester uses command keys to accept the difference and continue, or reject the difference and abort the test.

Screen Masking: Areas of a screen, like date and time, may be masked to exclude them from the compare, speeding up the testing process. Field display attributes, such as reverse image and highlights, may be either included or excluded in the replay screen comparison.

Volume Testing: Since no real display devices are required, several replay tests may be run simultaneously to automate volume replay and performance testing. Replay any combination of previously recorded tests to observe the impact and interaction between different modules or applications in a controlled environment.

Testing Interactive Processes in Batch

Midnight Shift: Another advantage of not requiring a physical device for test runs is that they may be scheduled to run after hours, leaving machine resources available to others during prime time. Also, there is no potential security risk because no devices need be left switched on or logged on.



www.thenon.com

Many of the world leading companies use Thenon's products to change manage and test their software.

Thenon – designers of SEE/Change, the leading iSeries change management product.

SmartTest400™ 3 – Record & Play

Maintaining Test Cases

Screen Maintenance: Test scripts can be replayed in maintenance mode. If a difference is found during the replay, command keys are available to replace the old screen format stored in the test case with the new screen format. Test script maintenance allows new screen formats or new fields with input data to be added to an existing test script.

Editing Images: Test screen images and associated responses may be modified. Responses may also include special values, such as the current date, or a specified number of days from the current date. The function key sent with the response may also be modified.

Customising Test Cases

User Variables: *User variables* may be defined and used for response data and screen masking. A user variable may be loaded from any location on a screen, and then used on subsequent screens, as well as in other test runs. When the specified screen is displayed during test replay, the contents of the specified location are copied into the user variable. The value of the user variable is kept from one test run to another, so a captured value may be used in subsequent test runs. User variable values may also be set manually via a CL command, thus allowing a value to be calculated by a user-written program and loaded into the user variable, prior to running a test case. User variables can also be retrieved via a CL Command, allowing a user-written program access to the contents of the variable.

Scripting Command Macros: The simple scripting command macros are designed to complement the replay process by allowing user defined jobs and iSeries commands to be executed during the replay life cycle.

Scripting commands can be established at the following levels:

- Setup
- Execute the test
- Post test “successful” processing
- Post test “unsuccessful” processing
- Housekeeping and cleanup
- User variables

The scripting command macros also allow the Data Builder and Data Environments commands to be interfaced to the replay process, in order to construct the correct database starting point and capture all the database transactions made during the replay of a test case.



www.thenon.com

Many of the world leading companies use Thenon's products to change manage and test their software.

Thenon – designers of SEE/Change, the leading iSeries change management product.

SmartTest400™ 4 – Data Builder

Accurate test data is required to thoroughly test changed code and application functionality. The best test data comes from the live production environment. However, it is not always possible to test applications on massive databases due to runtime, processing power and memory constraints.

Data Builder automates the process of deriving a minimum set of test data from a large amount of live data. It uses the latest Application Program Interfaces available to find the links between

files and fields. The extraction process allows data to be copied incrementally, so you can add new conditions to your test databases without regenerating the entire set.

Creating Test Applications and Build Cases: An Application is the top-level container within Data Builder and a Build Case is a set of extraction rules for files in one or more libraries. The Build Case supports files that have triggers, members, constraints and related fields that do not match across files.

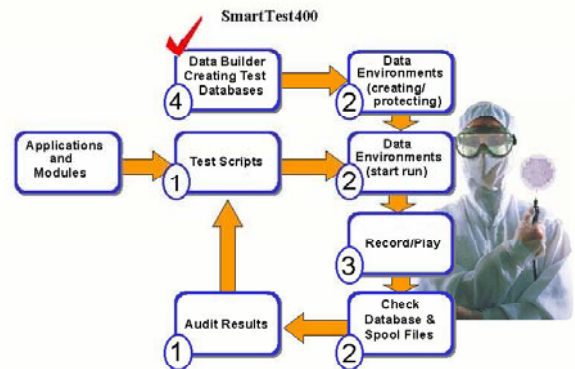
Editing the Build Case: Once the Build Case is known to Data Builder, it can be edited. All the physical files are displayed. You identify the first file in the extraction process and enter the selection criteria. To apply the selection criteria to all related files, simply use the Auto Link Dependent option. This option analyses the fields in the selected file and automatically links the fields with the selection criteria to all the related files. Done manually, this can be a time consuming job, but Data Builder can accomplish it within seconds; you can add additional levels of dependent files and selection criteria as required.

Execute Data Builder: When you run a Build Case, you specify the target libraries and if the files should be recreated, the data copy option (add or replace) and if the job should be run in batch.

Progress Reporting: During the extraction job, on-line metrics information is available to advise on the build stage being processed, the file being copied and the estimated extraction time remaining.

Warping a Build Case: After a build case has been executed sensitive data might need to be protected before the database can be used or date related fields might need to be advanced to keep the database current.

Data Builder Application Program Interfaces (APIs): SmartTest400's open design means Data Builder commands can be used within other SmartTest400 modules and PDM.



www.thenon.com

Many of the world leading companies use Thenon's products to change manage and test their software.

Thenon – designers of SEE/Change, the leading iSeries change management product.