Rational. software

IBM

Using Automation in ERP Validations

Using Automation in ERP Validations

O V E R V I E W

Implementing packaged enterprise resource planning (ERP) applications, such as those offered by SAP, Oracle, and BAAN requires significant resources to install, model business processes, connect to legacy systems, and upgrade. While the suppliers of these applications extensively test them before customers use them, implementing organizations must still validate that these applications conform to business rules and operate correctly and efficiently. Implementation teams traditionally perform this validation process manually, which has inherent disadvantages in coverage, scaleability, and repeatability. This paper discusses the use of automated software tools for validating ERP implementations and the benefits that accrue to organizations that elect to use automation in their validation process.

I N T R O D U C T I O N

Organizations spend months deciding how to upgrade their information systems. They sort through the many potential implementation partners and select one with the people and methodology to match their implementation. Now for the tough part implementing and testing the results. Implementation teams face two fundamental questions:

- Will it work in my environment?
- Does it support my business processes?

Packaged ERP vendors extensively test and validate the operation of their software. However, with tailoring options available in these applications, there are millions of different ways an organization can operate them. These drive the need for validation of any packaged ERP implementation from a new installation to a routine software upgrade. Add to these the different ways that an organization can populate these applications with information. Now add the differences they can introduce when integrating with legacy applications, and you have the formula for a complex and time consuming implementation process one that demands results validation.

The software development industry has proven methodologies and software products for validation of complex systems called *automated software quality* (ASQ) products. These products, with extensions specifically designed for ERP applications, help raise the quality of ERP implementations. They help assure implementation success by reducing surprises and discovering defects earlier in the implementation process, leading to on-budget and on-time completion.

WILL IT WORK IN MY ENVIRONMENT?

Packaged ERP implementations can be complex. Besides the business modeling requirements, potential organizational process changes, conversion of information from legacy systems and user training, there are risks that the system will not perform as expected when tested in "real-life" situations. Every organization is different, and will stress the system in different ways. The real question is whether the system is set up correctly to ensure efficient and effective processing. There are four dimensions to the "will it work" question:

Figure 1: *The four dimensions of the "will it work" question.*

- *Validation* : Can I run my business from this system? Will it operate as expected?
- *Performance* : Will the system response be adequate for typical users? Can users get their jobs done interacting with the system?
- *Load* : Will the system continue to give adequate and accurate performance when supporting a full complement of users?
- *Volume* : Will the system continue to operate correctly when it contains all of my information, rather than just test data?

VALIDATION: MANUAL VERSUS AUTOMATED

Manual Validation

Organizations can use two different methods to confirm that packaged ERP implementations operate as expected in their environment manual and automated. The traditional method is manual bring users in for an extended period of "hands-on" use, where they are trained on the new system and validate that it performs as expected. Automated validation offers a new approach to confirming proper operation. Manual validation has four disadvantages:

- *Feature and function confusion* : Manual validation can confuse users. Since the user test is the first time the system encounters a real-life workload, it sometimes operates unexpectedly. Implementation teams expect this that is why they perform user testing. However, organizations often want to simultaneously validate and train users on the system. Novice users often can't discern errors from valid operation, particularly where the system exhibits subtle defects. This can result in users leaving with a misunderstanding of how the system operates and missed defect detection.
- *Cost* : Manual validation is expensive. It usually happens during the most schedule-critical stage of the project, it takes valuable people off-line, and its time requirements are unpredictable.

- *Thoroughness* : Most significantly, manual validation often fails to thoroughly test all phases of the implementation not because of poor design in validation coverage, but because of the interrelated nature of ERP applications and the potentially large number of separate steps that need validation. Testing these applications takes time: initial setup, generating test cases, and validating results for hundreds or thousands of cases. Manual methods are hard to track because it is difficult to determine whether users remembered to try every option and whether they repeated the required tests accurately. In the best case, they re-test all the previous milestones as they reach each new one. However, most organizations rarely do because of schedule and cost constraints.
- *Accuracy* : Finally, manual validation cannot predict how the system will operate when all the users access it at once. Without extensive testing user populations that accurately reflect the "live" user population (which are rarely possible prior to going live), there is little way for implementation teams to assess how the system will operate with the full load of users accessing it.

Automated Validation

The software applications development industry has two well-defined methodologies for determining both whether the system operates and continues to operate correctly, and how well it will operate with all the users accessing it. The first, which detects change-introduced defects is called regression testing. This is the process of recording all the previously performed business logic validation steps, and automatically performing them again as developers change the system.

The second methodology for validating ERP operation is called performance testing. This is the process of creating a realistic load on the system by simulating many users. Since these users are simulated in software, implementation teams that use performance testing software don't require hundreds or thousands of "live" users to know, in advance, the performance of the system when it goes live.

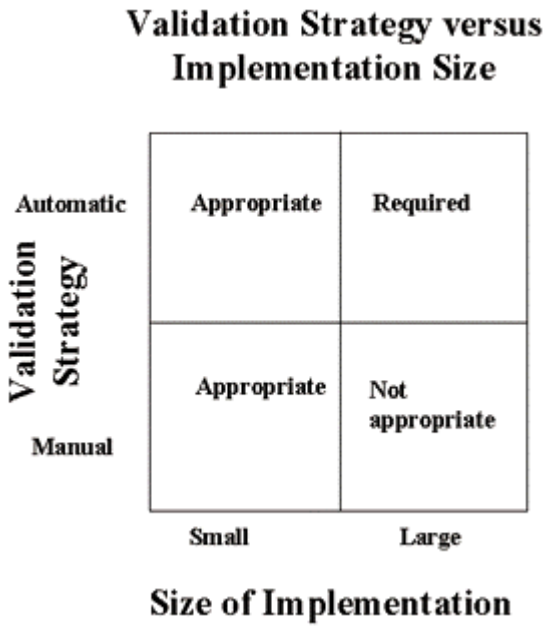

Figure 2: *Best practices dictate the use of automated methods in validation wherever possible. For larger implementations, automated validation becomes mandatory.*

Regression Testing

Regression testing addresses the "Does it support my business processes" question. Implementation teams often devote much of the validation effort to re-running previous validation tests i.e. performing regression tests to assure that none of the changes cause defects in another part of the system. These sorts of operations, repeated under different conditions, lend themselves well to automation.

Automated validation of ERP implementations employs the same methodology. The implementation team designs validation steps and records them for use in regression tests. They can design these tests to cover the complete range of application functionality. Recording enables them to play them back in regression tests and assure that system changes do not adversely affect other system components.

The record/playback process is usually performed with third-party ASQ tools. These tools perform non-intrusive recording of normal user actions the same steps a manual tester would perform. Users navigate

through the system and insert "test cases" selected places where the recording software should compare the ERP operation with baseline operation. This might be examination of a field for a specific value, examination of a table for a specific collection of values, or review of a report for specific information.

During playback, the ASQ tool automatically supplies the appropriate keystrokes and mouse clicks to the ERP application, which performs the exact same steps as the manual test performed. The ASQ tool automatically verifies test cases for correct value(s) and systematically steps through the ERP application like the manual tester. The difference is that implementation teams can run these tests repeatedly without manual intervention, while making changes to other parts of the system, and insure that changes don't have ripple-through affects.

Part of the playback process is verification of test cases, and ASQ tools usually include a comprehensive defect tracking and reporting system for acting on system events that cause differences in operation.

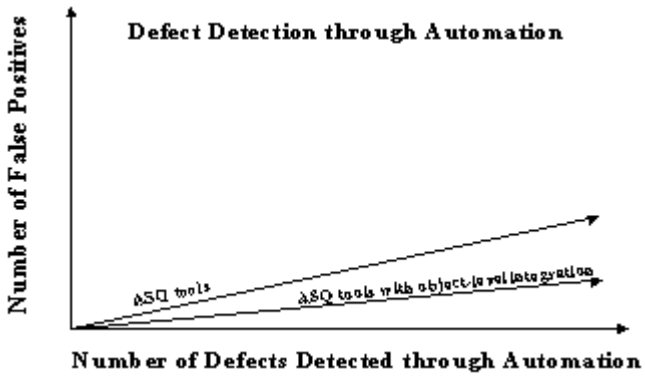

Figure 3: *Some "false-positive" defects occur with any automation. ASQ tools that achieve object-level integration with the ERP system can minimize these false-positive reports.*

A key to successful use of automation in business logic validation is correctly detecting and reporting bona-fide defects while ignoring, or at least suppressing, application changes that don't affect operation. The operation of ERP applications can change for a host of different reasons new data, changed operation, different user privileges, etc, and the challenge in using automation is detecting and acting upon the changes that represent true defects. ASQ tools use various strategies to mitigate reporting "false positive" defects. The most successful are ones that are the result of a technical integration of the ASQ tool with the ERP application. Those that use object-level integration have a better understanding of the ERP application, and therefor can mitigate the "false positive" defect reports, while maximizing the true defect detection.

Automated validation has four advantages over manual processes:

- *Repeatability* : When used in a comprehensive regression testing methodology, automated validation can guarantee validation of the whole system, even when making changes at the last moment. Recording validation tests for automated playback and organizing them into a structured hierarchy lets an implementation team validate the whole system at any time.
- *Cost* : Automating validation is less expensive and time consuming than manual methods. The implementation team simply re-runs the validation test suite at any appropriate time, even overnight while off-duty. User testing is still appropriate, but these tests are considerably less likely to uncover errors since the implementation team has developed more comprehensive coverage tests. When user testing does uncover errors, the implementation team can quickly validate whether changes they make to correct the error adversely affect other parts of the system.
- *Phased implementation* : Automated testing facilitates phased-implementations which are inherently less risky. You can divide the project into a series of sub-components, each with a low risk of completion. The team designs validation scenarios for each sub-component and re-uses them as new components go live to ensure that new changes don't affect previous successful operation.
- *Benchmarks* : Finally, automated testing provides a documented benchmark suite. These systems operate in dynamic environments software gets upgraded, hardware changes, networks require reconfiguration, and business models change. Any of these changes can affect ERP applications operation. Without a demonstrated benchmark in place, organizations cannot assure that any of these changes haven't adversely affected the system operation. Organizations that use automated testing have a documented suite of tests they can run after system changes to determine readiness.

Performance Testing

The second methodology for validating system operation addresses the "Will it work in my environment" question. Performance testing is vital because virtually every organization's environment is different. While the ERP vendors guarantee functional operation of their systems, they cannot guarantee the performance characteristics of these systems when they operate in different hardware, software and network infrastructure environments.

Performance testing generates a realistic system load through a variety of approaches, then measuring the system response to this load and comparing it to required performance parameters. Implementation teams may select several different methods for validating system performance:

- *Published benchmarks* : These go by the names of Whetstones, Dhrystones, Webstones, MFLOPS, and etc. They all have the disadvantage of not accurately representing your system. Manufacturers derive these benchmarks from optimal lab conditions where the system is specifically tuned to operate the benchmark well. They can be excellent for measuring one system against another, but rarely yield useful information when it comes to measuring a specific system's response to live user loads.
- *Hardware load generation* : Some organizations use computer hardware to generate system load. This approach can be very accurate, since if done properly, it exactly represents the actual dynamics of the user population. However, this approach is generally not practical for medium to large organizations. They simply cannot organize the necessary hardware and operators to perform a realistic test of the system response.
- *Software load generation* : This is the preferred method for generating system load and measuring performance for medium and large organizations. In this approach, a software system simulates users from hundreds to tens of thousands to gauge the system response to a live user population. Since this is a simulation of a live user population, it is most important that the simulation be accurate and that it scale to the required numbers of users.

WILL IT CONTINUE TO WORK?

In the rush to implement, organizations sometimes fail to plan for the continued upgrade and maintenance of these systems. Packaged ERP applications operate in multi-vendor environments, where the combined total of software and hardware upgrades can mean frequent changes as often as every calendar quarter. ERP applications need validation with any change to the components not just packaged ERP software upgrades, but upgrades to the underlying database, network infrastructure and operating hardware.

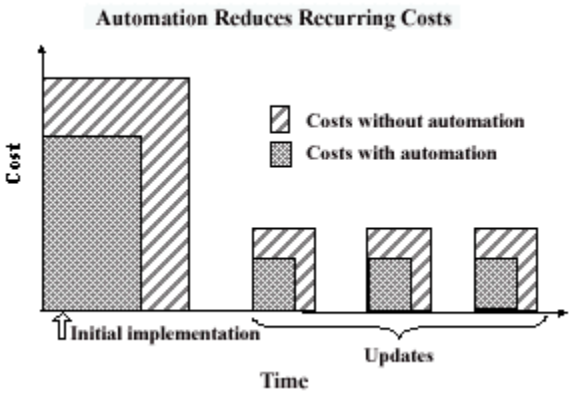

Figure 4: *Automation reduces initial implementation time and cost, as well as reducing costs on an on-going basis. Software and system upgrades usually occur at regular intervals, each necessitating a re-validation of the system.*

Implementations that employ automation for validation have built-in methods for validation at any point in the future. You can use the same suites of tests that certify the system before live use to validate the correct operation of the system after any system maintenance upgrades, new configurations, and changes. In a matter of hours, you can verify that the change that was "invisible to the end user" really has no effect.

AUTOMATION IN ERP VALIDATIONS -- SUMMARY

Experienced implementers use repeatable processes for ERP implementations. Automated validation can be a vital part of a successful implementation when employed early in the process. Organizations elect to use automated validation tools during initial ERP implementations and as part of upgrade processes. Through the use of these tools, organizations can finish implementations sooner and reduce the risk inherent in these large-scale projects.

Automation tools help implementation teams finish sooner by uncovering defects earlier in the process, and assuring that changes don't introduce unintended results. These same tools reduce the risk of any implementation by thoroughly testing the applications in real-life scenarios that range from tests that insure the applications support the business logic to tests that assure the applications perform with adequate performance.

IBM software integrated solutions
IBM Rational supports a wealth of other offerings from IBM software. IBM software solutions can give you the power to achieve your priority business and IT goals.

- *DB2® software helps you leverage information with solutions for data enablement, data management, and data distribution.*
- *Lotus® software helps your staff be productive with solutions for authoring, managing, communicating, and sharing knowledge.*
- *Tivoli® software helps you manage the technology that runs your e-business infrastructure.*
- *WebSphere® software helps you extend your existing business-critical processes to the Web.*
- *Rational® software helps you improve your software development capability with tools, services, and best practices.*

Rational software from IBM
Rational software from IBM helps organizations create business value by improving their software development capability. The Rational software development platform integrates software engineering best practices, tools, and services. With it, organizations thrive in an on demand world by being more responsive, resilient, and focused. Rational's standards-based, cross-platform solution helps software development teams create and extend business applications, embedded systems and software products. Ninety-eight of the Fortune 100 rely on Rational tools to build better software, faster. Additional information is available at www.rational.com and www.therationaledge.com, the monthly e-zine for the Rational community.

IBM Corporation
Software Group
Route 100
Somers, NY 10589
U.S.A.

Printed in the United States of America
01-03 All Rights Reserved.
Made in the U.S.A.

The IBM home page on the Internet can be found at **ibm.com**