Enterprise Data Management Solutions
April 2008

IBM Information Management software

Enterprise strategies to improve application testing

Contents

Executive summary

Ensuring the accuracy, reliability and quality of your critical business applications has never been more important. Why? Because companies across industries depend on mission-critical enterprise applications to drive their business initiatives.
In turn, these applications rely on relational databases to store and manage the underlying enterprise data. The ability to enhance, maintain, customize and upgrade these sophisticated applications is critical for achieving long-term business goals. Companies are striving to speed the deployment of reliable, high-quality applications, while staying within tight IT budgets.

Now, more than ever, companies face new challenges when designing effective and efficient testing strategies for enterprise applications. Incomplete or flawed test data means inaccurate testing, which can lead to application failure and business disruption. More common approaches to building test environments include cloning application production environments and writing custom extract programs. However, these methods can be labor intensive, error prone and costly. No company wants to risk losing customers, market share, brand equity or revenue by delivering applications that have not been thoroughly tested. For this reason, end-to-end application testing is a strategic priority throughout the application development lifecycle.

So, how can IT organizations improve testing efficiencies and reduce the total cost of owning and maintaining enterprise applications? This white paper examines how proven test data management capabilities can help you deliver reliable applications and achieve maximum business value from your application investment.

Delivering reliable applications

Today's applications drive revenue and satisfy sophisticated marketplace initiatives. These customer-facing and mission-critical applications have a huge strategic impact. As a result, there is increasing focus on application accuracy, reliability and performance to minimize downtime and engender customer loyalty.

According to *Computerworld*, "buggy software is costing the economy billions of dollars."[1] Software defects directly affect a company's bottom line. Failing to accurately test applications through the various deployment phases could lead to potential downtime. Can your business sustain the revenue impact and customer loss that could result from an application failure?

Improving the way you manage enterprise application data in the testing environment can deliver dramatic results including improved reliability, faster time-to-deployment and lower development costs.

- **Speed time to deployment.** Deploying reliable applications, enhancements and upgrades in a timely manner provides companies with a competitive advantage. Shorter upgrade and development cycles are desirable, but reliability testing takes time. This classic struggle between these two business objectives often means that deployment plans and "go live" dates slip or that the level of quality testing is insufficient. How important would it be to your business if you could get applications to market faster without sacrificing accuracy and reliability?
- **Control development and quality assurance costs.** Reducing the overall cost of developing, testing and delivering reliable applications can result in significant cost savings throughout the entire application development lifecycle.

Companies can realize direct cost savings and productivity gains by reducing the work effort required by DBAs, developers and quality assurance testers, allowing them to address other business issues. Could your business benefit from reducing development costs and using talented IT resources more effectively?

- **Identify and resolve application errors early.** The costs associated with resolving application errors in the production environment are as much as 10 to 100 times greater than if these errors were resolved early during the application development and testing processes. Even worse, when your business users and customers are the first to discover application errors, you risk losing revenue and compromising customer loyalty. Could your company afford the increased direct expenses and indirect opportunity costs of releasing faulty software?

- **Lower infrastructure costs.** In application development and testing environments, there are, conservatively, four or more clones of the original production database. As data is duplicated, costs increase proportionally. Using "right-sized" databases rather than full production clones can help you save capacity resources. In addition, by reducing the size of test databases, you will also reduce the processing power needed to perform testing activities. Could your business benefit from using smaller, right-sized application test databases?

IT organizations realize that more planning is needed to satisfy the rigorous testing objectives necessary to deliver reliable application upgrades and enhancements. Designing a comprehensive testing strategy is often taken for granted. Organizations also recognize the need for proven test data management capabilities to improve the quality assurance process at a lower cost.

Many organizations are reluctant to make a significant investment in testing, and instead allocate scarce budget resources to other areas. However, without a commitment to quality assurance, the business consequences could be significant. It is necessary to understand the impact of unacceptable application accuracy and reliability on critical business objectives, such as generating revenue and ensuring customer satisfaction.

Protecting your investment in your custom and packaged applications by using effective test data management capabilities and repeatable testing processes will ultimately increase your return on investment. In today's market, reliability and quality are major competitive differentiators. To remain competitive, thoroughly testing your applications before deployment must be a high priority.

Why is application testing a challenge?

Typically, each time a new application is developed, or an existing application is modified, new test databases are created. Since it is ideal to use "realistic" test data, the test database is usually a clone of the production database. Cloning and entire production database increases the time needed to run test cases because there is a larger volume of data. It is much faster to test with smaller, realistic subsets that accurately reflect the production data, without adding overhead to the testing process. In addition, cloned data many not support the specific error and boundary conditions required for effective testing. Special test cases may be added before testing begins.

Each phase, from unit testing through system integration and acceptance testing, has unique requirements and varying levels of complexity. Iterative testing involves executing the application using the test database and verifying the results to validate that the application is functioning as designed. Any problems discovered must be resolved, and the test data must be refreshed, before testing continues.

This process is repeated throughout the various testing phases (unit, integration, system, load, regression and acceptance testing) until the application is migrated into production. Finally, after a test is executed, there is no easy or automated method for verifying the results. In addition to finding and resolving application errors before deployment, another important goal is to create a repeatable testing process that improves application quality, reduces time to market and minimizes costs.

Complexities of testing with relational data. The fact that most applications rely on relational database technology introduces a major challenge for organizations in the testing process. The application data model may contain dozens, hundreds or even thousands of tables, with just as many interrelationships. Data model complexity is not limited to large-scale systems. Even a database of less than a dozen tables can contain relationships that make navigating the data model difficult.

Without the appropriate solutions, developers often need to write sophisticated extract programs just to create the test data. It is a challenge to navigate the numerous tables, rows and columns to create, manipulate and refresh the desired database subsets. It is also difficult to develop an extract program that considers relationships defined and enforced by the application. Finally, whenever the application or database is updated, the extract program must be updated as well.

Supporting heterogeneous database management systems and differing data models adds to the complexity. For example, CRM data may be managed in an Oracle® database and related billing data may be managed in an IBM® DB2® database. Clearly, comprehensive testing capabilities must support these relational complexities and “remember” to account for them in every extract, compare or update operation.

Test data from multiple databases. Most organizations have data stored in a variety of relational databases, such as DB2, Oracle, Microsoft® SQL Server®, Sybase® and IBM Informix®. In addition, data may be stored in hierarchical or non-relational formats, such as IBM VSAM® files and IBM IMS™ databases. When testing complex applications, it is not uncommon to require test data from multiple related databases including both relational and non-relational data sources. All database management systems have different methods for handling data. The integration of diverse systems is a typical requirement. Quality testing must also include seamless capabilities to handle data from different database management systems operating on different platforms.

Test data from homogeneous databases. In homogeneous database environments, the technology is the same or compatible. One of the challenges of creating test data in a homogenous database environment is the ability to extract complete subsets of related data and to keep that data referentially intact. It is necessary to navigate all relationships in the data model, whether they are defined in the database or the application (Figure 1).

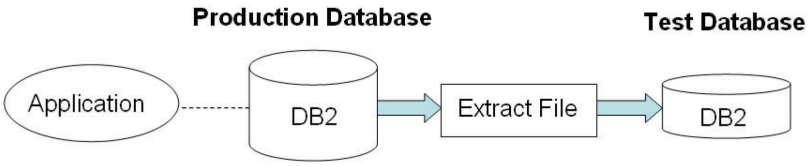

Figure 1. Application testing with homogeneous test data.

Test data from heterogeneous databases. Heterogeneous database environments can include different hardware, operating systems and data models, as well as syntactic and semantic differences. Creating test data from heterogeneous databases adds data compatibility and translation issues. Typically, this happens when moving an application to a different database or test environment. For example, it may be necessary to extract subsets of data from a DB2 production database and insert the data into an Oracle test database. This transition may cause problems with special registers or date and time stamps, which are handled differently by each DBMS (Figure 2).

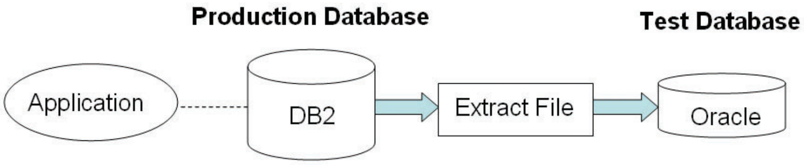

Figure 2. Application testing with heterogeneous test data.

Beyond extracting referentially intact subsets of data, when working with heterogeneous databases, you need additional capabilities for managing compatibility and translation differences automatically.

Federated test data from multiple relational databases. Federated environments often require extracting related test data stored in multiple relational databases, which adds another level of complexity. For example, payroll data may be managed in a DB2 database and employee data in an Oracle database (Figure 3).

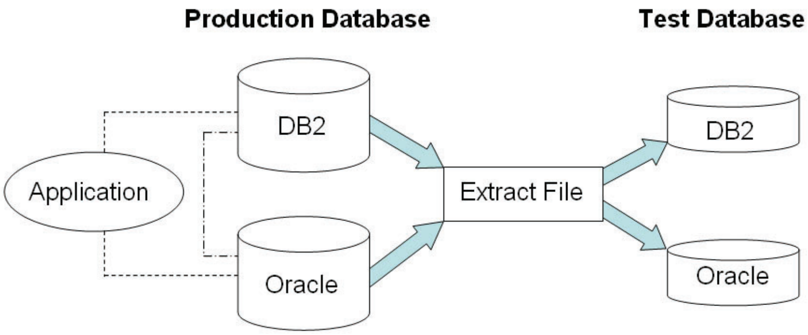

Figure 3. Application testing with federated relational test data.

In addition to maintaining referential integrity and handling data compatibility and translation issues, creating realistic test data now requires a federated data access capability, so you can extract related data from multiple databases using a single extract process.

Federated test data from non-relational and relational databases. While relational databases have inherent complexities, the task of testing an application that references both non-relational and relational data poses an even greater challenge. For example, an order entry application may require product data from a VSAM inventory file and customer data from a DB2 order entry table. Data stored in VSAM files or IMS databases have different structures and cannot easily be integrated with relational data (Figure 4).

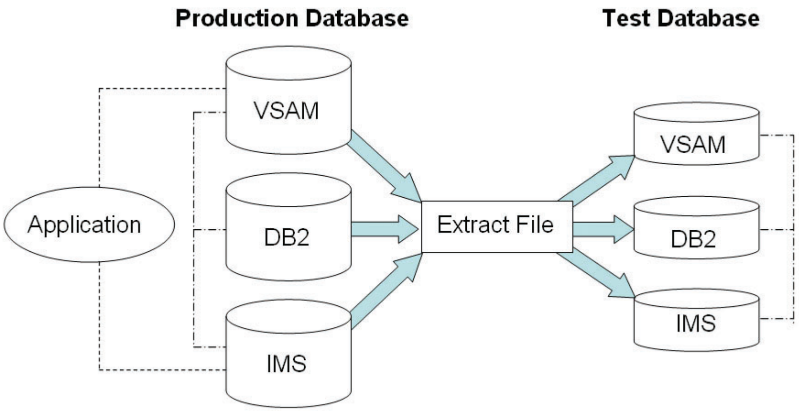

Figure 4. Application testing with federated non-relational and relational test data.

An effective testing strategy in these scenarios must include the ability to create precise subsets of a realistic federated test data from non-relational and relational databases. Because the database structures are significantly different, it is a major challenge to correlate data across complex non-relational and relational sources. What is needed is a tool that accesses VSAM and IMS data and correlates that data in a way that can be easily "understood" and managed.

Because VSAM and IMS databases are less flexible, it is difficult to extract subsets of data from these databases without the ability to specify simple or complex selection criteria. The ability to apply selection criteria dynamically to extract VSAM or IMS data would significantly reduce processing overhead and improve the quality of the test data.

Supporting your application testing requirements

So, what is needed to improve application testing, especially when the quality of the test data is critical?

- **Creating right-sized test databases.** Using realistic, referentially intact subsets of production data from multiple data sources promotes accurate and efficient testing and reduces capacity requirements. Subsetting allows you to create realistic test databases small enough to support rapid test runs and large enough to accurately reflect the variety in your production data. Working with predefined, realistic subsets of data and refreshing the test data iteratively throughout each test cycle improves testing and overall application quality. The ability to save and reuse processing specifications streamlines the testing process and provides a consistent and manageable test environment.

- **Using realistic test data.** Building and populating realistic test databases requires the capability to extract and move precise subsets of related database based on your specifications. Maintaining referential integrity with 100 percent accuracy is essential, particularly for the most complex application data models. Including metadata in extract processing enables replicating test environments and accommodating data model changes quickly and accurately.

- **Masking sensitive data to protect privacy.** Protecting data privacy in the application development and testing environments is necessary to prevent misuse of personally identifiable information and to comply with global data privacy legislation and industry standards. Capabilities for de-identifying confidential data allow you to protect privacy, while providing the necessary realistic test data to developers and quality assurance staff.

- **Improving testing efficiencies.** As the content of the test database is modified during the iterative testing process, it diverges further from the baseline test data, resulting in a less than optimal test environment. Without capabilities for saving and refreshing the test data, it is more difficult to reuse the test database to promote accurate results on subsequent test runs. The capability to save and reuse specifications for extract, insert and load processing helps streamline the testing process and maintain a consistent and manageable test environment.

- **Creating targeted test scenarios.** Creating test data to force error and boundary conditions requires intelligent browsing and relational editing capabilities. Relational browse features improve the ability to perceive data relationships clearly and to identify and resolve problems quickly. The capability to insert rows and edit database tables directly improves productivity and accuracy. Multi-level undo capabilities are essential.

- **Automating test result comparisons.** The ability to identify data anomalies and inconsistencies during testing is essential to the overall quality of the application. The only way to truly achieve this goal is to deploy an automated capability for comparing the baseline test data against results from successive test runs. Speed and accuracy are essential. Automated compare processing saves time and helps identify problems that would otherwise go undetected.

IBM Optim meets the challenge

The IBM® Optim™ Test Data Management Solution enables IT organizations to meet even the most complex application testing challenge by providing all the fundamental components of an effective test data management strategy:

- Extract referentially intact subsets of data with 100 percent accuracy to create realistic test databases no matter how many tables or relationships are involved.

- Insert or load subsets of related data to quickly build realistic test databases. Update or refresh the test data consistently to preserve the integrity of the test environment.

- De-identify sensitive data in the test environment to support compliance with regulatory requirements for data privacy. Transform test data to meet specific test case requirements.

- Browse and edit test data to force error conditions and resolve problems. Reviewing data in its relational business context provides a clear vision of the data model.

- Compare the test data before and after exercising the application to validate expected test results and identify anomalies automatically and with pinpoint accuracy.

- Integrate test data from multiple related databases and database management systems (DB2, Oracle, Sybase, SQL Server and Informix).

- Integrate relational and non-relational data to create a federated test environment.

- Integrate test data management with automated testing to provide comprehensive testing capabilities.

When these capabilities are in place:

- Developers can validate that new application functions perform as expected during unit testing and modifications do not introduce problems during integration testing.

- Quality assurance staff can validate that the entire system operates as expected and validate that interfaces with other systems work properly.

- Business unit users can ensure that the system meets their expectations for functionality and performance during acceptance testing.

Developers and quality assurance staff spend less time extracting data to create and maintain development and testing environments and more time on performance tuning, backup and recovery processing and managing production databases.

What makes Optim unique? The Optim Relationship Engine™ is a unique technology that understands and processes related data from multiple tables and ensures that each test subset is always referentially intact and logically complete. For example, in a subset of customer data, one customer may have items that are backordered while another may not. From one customer to the next, the number of rows retrieved from any number of tables will vary. But the Relationship Engine always gets the right items for the right customers, complete and intact, every time.

Optim uses an active repository to store the user-defined business rules. The repository automatically captures relationship information defined to the database. Additional data model information can be populated from third party dictionaries or data modeling software. Finally, users can define relationships that exist but are not known to the DBMS, such as application-managed relationships.

Using the active repository, IT staff members can define, share and reuse different Access Definitions that specify, in essence, varying subsets of the database — related rows from many tables. Optim's Relationship Engine technology assembles all related rows from every table into a single referentially intact subset.

Optim capabilities allow users to migrate, browse, insert, load, transform, edit and compare complete subsets of related data. Organizations can design comprehensive testing strategies that include realistic test data while improving productivity and overall application quality and reducing development costs (Figure 5).

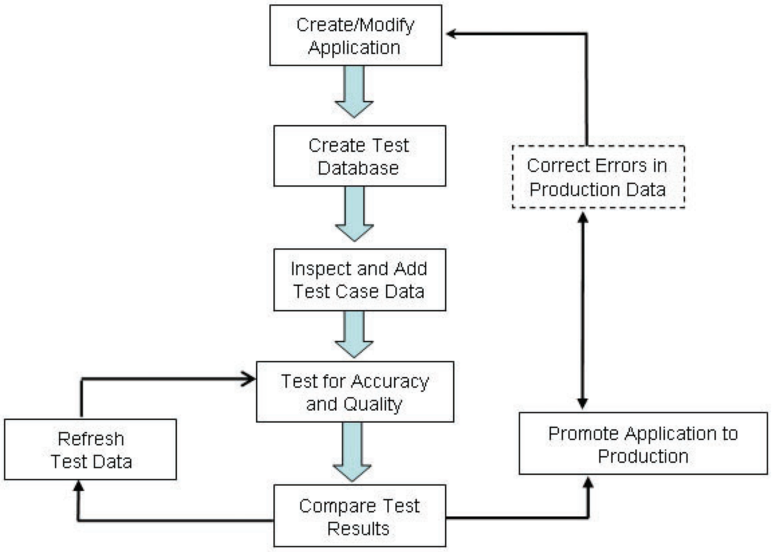
Create/Modify Application
Create Test Database
Inspect and Add Test Case Data
Test for Accuracy and Quality
Compare Test Results
Refresh Test Data
Promote Application to Production
Correct Errors in Production Data

Figure 5. IBM Optim improves every stage of the application testing process.

Creating targeted testing environments

Typically, it is impractical to clone an entire production database comprising hundreds of highly interrelated tables just for testing purposes. First, there is a capacity issue. Second, there is a quality issue — when working with large test databases, developers may find it difficult to track and validate specific test cases.

Creating realistic test data. Instead of cloning, Optim's subsetting capabilities allow for creating realistic, right-sized test databases that are easy to manage and maintain. With Optim, you can extract referentially intact subsets of realistic application test data without writing custom programs. Simply specify the selection criteria to identify the precise set of related data you need to test a particular application function. It is much faster to test with smaller, realistic subsets that accurately reflect application production data, without adding overhead to the testing process.

Preserving the referential integrity of the test data. Optim respects the Referential Integrity (RI) rules enforced within the database and your applications. Typically, the application-enforced RI is more complex. For example, the application may include relationships that use compatible but non-identical data types, composite and partial columns, and data-driven relationships. Optim has proven capabilities for handling every type of relationship. Optim provides the flexibility to build focused test databases to validate the functionality and performance of the application modules you need to test. These capabilities support your requirements through each phase from unit testing through system integration and acceptance testing.

Forcing error and boundary conditions. Creating realistic subsets of related test data from a production database is a reasonable start, but it is sometimes necessary to edit the data to force specific error conditions or to validate specific processing functions. Optim includes comprehensive relational editing capabilities that not only simplify the tasks necessary to create this special data, but also make it easy to browse data and resolve application issues. The ability to browse data in its relational or business context also provides a clear way to envision the data relationships and structure of the application data model.

Masking and transforming test data. With the increased focus on data privacy, the ability to transform and de-identify sensitive data in the development and testing environments is critical to prevent data breaches and severe penalties. Optim provides flexible technology for mapping source and destination columns and capabilities for using a variety of transformation functions and algorithms.

For example, it would be easy to de-identify customer identification numbers by simply applying a random number function. More sophisticated masking capabilities allow you to use substrings, sequential numbers, date aging, currency conversion or table lookup functions. The capability to include user-defined data transformation programs provides even greater flexibility to satisfy complex or site-specific requirements.

All of the methods described are effective techniques for de-identifying test data. However, in testing relational database applications, there is an added complication. Specifically, the tester requires the ability to propagate a masked data element in a parent table to all of the related child tables in the database. Key propagation is necessary in order to retain the referential integrity of the transformed data. Otherwise, the relationship between parent and child tables is severed, test data is inaccurate and application testing yields invalid results.

Key propagation also provides the ability to "manufacture" new sets of test data. This technique is helpful when additional data is required for testing, as is the case when testing new applications or testing applications when new tables are added to the data model.

Validating the test results. Without the capability to automatically compare the test data before and after a test run, validating test results and identifying changes are next to impossible. First, there are a variety of changes: inserts, deletes and updates spread across hundreds of tables. Second, there may be unexpected problems (for example, orphaned rows) and other anomalies that may go undetected if the results are analyzed manually.

Optim provides the capability to compare subsets of related data and to identify the differences automatically. With relational data, this means more than comparing row to row. It means using data model intelligence to compare related sets of rows. For example, after developing a new product rate table, you can compare the "before" and "after" test results to validate that rate changes are being applied appropriately.

Maintaining the quality of the test data. The quality of the test environment is just as critical as the quality of the test data. The content of the test database is modified during the testing process. Over time, it diverges further from the baseline test data, resulting in a less than optimal test environment. Optim provides capabilities for saving and reusing the test data, it is easier and faster to refresh the test database to ensure accurate results on subsequent test runs.

Optim's extract process also includes metadata, making it easier to accommodate changes in the data model during the testing phases. Metadata is definitional data that provides information about the structure of the data managed within an application or environment. For example, metadata would document the structure of the database including the tables, columns, relationships, views, triggers and so on.

Working with predefined realistic subsets of data and refreshing the test data iteratively throughout each test cycle improves testing and overall application quality. Including metadata in extract processing enables recreating the test data and accommodating data model changes quickly and accurately. The ability to save and reuse processing specifications streamlines the testing process and helps promote a consistent and manageable test environment.

Improving test data management delivers business value

You invest millions in your enterprise applications and the supporting infrastructure to support optimal operating performance, improve decision-making and gain a competitive advantage. Optim provides the power of enterprise data management, so you can derive the most business value across your enterprise.

Optim provides a single solution for managing enterprise application data throughout every stage of the information lifecycle. Now you can assess and classify application data by age and status. You can apply business rules to archive, subset, access, store, retain and protect your enterprise data. Optim capabilities are based on a consistent and proven data management methodology that aligns with your business objectives and scales across applications, databases, operating systems and hardware platforms.

Speed application deployment by streamlining the way you create and manage test environments. Subset and migrate data to build realistic and “right-sized” test databases. Eliminate the expense and effort of maintaining multiple database clones. De-identify, mask and transform confidential data to protect privacy and minimize the potential for unauthorized disclosure.

Optim supports all major enterprise databases and operating systems, including DB2, Oracle, Sybase, SQL Server, Informix, IMS, VSAM, Microsoft Windows®, UNIX®, Linux® and IBM z/OS®. And it supports the key ERP and CRM applications in use today — Oracle® E-Business Suite, PeopleSoft® Enterprise, JD Edwards® EnterpriseOne, Siebel® and Amdocs® CRM, as well as your custom and packaged applications.

About IBM Optim

IBM® Optim™ enterprise data management solutions focus on critical business issues, such as data growth management, data privacy compliance, test data management, e-discovery, application upgrades, migrations and retirements. Optim aligns application data management with business objectives to help optimize performance, mitigate risk and control costs, while delivering capabilities that scale across enterprise applications, databases and platforms. Today, Optim helps companies across industries worldwide capitalize on the business value of their enterprise applications and databases, with the power to manage enterprise application data through every stage of its lifecycle.

For more information

To learn more about IBM Optim enterprise data management solutions, contact your IBM sales representative or visit: **www.optimsolution.com**.

IBM Software Group
111 Campus Drive
Princeton, NJ
USA, 08540-6400
www.optimsolution.com

Produced in the United States of America
04-08

[1] *Sue Hildreth, “Buggy Sofware: Up From a Low-Quality Quagmire,” Computerworld, July 25, 2005.*

References in this publication to IBM products, programs or services do not imply that IBM intends to make them available in all countries in which IBM operates or does business.

TAKE BACK CONTROL WITH Information Management