Test Data Management
Best Practice

Meridian Technologies, Inc.
5210 Belfort Parkway, Suite 400
Jacksonville, FL 32256

Author:
Stephanie Chace
Quality Practice Lead
srchace@meridiantechnologies.net

www.meridiantechnologies.net

Table of Contents

1. Overview

A major component of many test efforts is the development and management of test data – to the extent that it is not unusual for test data creation and maintenance work to consume as much as 30 to 50% of the total test effort. This effort typically represents a significant investment both in time and money. Regardless of the amount, you want to get the most out of your test data investment.

This paper will review why test data is important and help quantify what constitutes good test data and test data management practices. We will elaborate on the best practices you can apply to ensure you get the best possible test data for your investment. These best practices will be aligned to three basic test data management activities:

- Understanding your test data requirements
- Obtaining data to meet your test data requirements
- Making the test data process repeatable and scalable

These activities ensure we understand what we are trying to accomplish, enable us to actually accomplish it, and then repeat our success. The general goals of maintaining high quality while simultaneously minimizing costs are critical. In addition, effective test data management supports overall improvements in test efficiency thereby helping to optimize your total test effort and provide additional value to the business.

2. The Importance of Good Test Data

Our starting point is to understand why test data has become such a significant component of the test effort. Generally speaking, test activities are designed to mimic real world usage of the system with the fundamental goal of detecting problems before they impact the intended users of the system. The more comprehensive and realistic the test effort, the more reliably testing can predict, and provide the opportunity to correct, erroneous behavior.

In business applications, the impact of production failure can have devastating consequences including loss of revenue and customer trust. In cases where software failures impact regulatory compliance, companies may be subject to severe financial penalties or even litigation. In a comprehensive study completed by the US Department of Commerce, software defects were estimated to cost US businesses nearly $60 billion dollars annually[1], and the number is likely rising with the increase in regulatory requirements.

Since data plays a central role in today's core business applications, we need to ensure that data plays a similarly important role in our test efforts. Test teams need data with characteristics as close as possible to real production data to properly test and evaluate system behavior. No data set is perfect, and many production failures are due to anomalies in the data which can only be detected when these anomalies are present in the test data.

Good or high quality test data will reflect the characteristics of your production data set. However, just using a copy of production data is not a realistic or cost effective option. Risk and data security compliance measures are increasingly restricting the use of production data. Further, the

use of production size data sets drives up data storage and management costs. Working with large data sets also tends to bog down test efforts and can reduce overall test efficiency.

As an example of how large data sets impact efficiency, consider the time needed to test the processing of large volumes of transaction data. By reducing the size of the data set, you can reduce the overall execution times. In one Meridian client engagement, reduction of the data set from full volume to an appropriate subset reduced test cycle times from 1 week to ½ a day.

Fundamentally, good test data can then be described in terms of two basic qualities:

- It correctly represents the full range of production or "real world" data
- It is sized appropriately to support testing needs

Of course, fully quantifying these attributes can be difficult and this difficulty increases with system complexity (but then so does the impact of poor data choices). In the next section we will discuss the test data requirements process which is the key to defining these qualities. As you think about establishing test data requirements, remember the impacts of poor test data:

- Increased costs and longer cycle times due to test inefficiency
  - Unnecessary data storage and maintenance costs
  - Long test execution times
  - Increased analysis and debug effort
- Increased business risk due to incomplete or unreliable test results
  - Test results are impacted by lack of relevant or appropriate data
  - Potential for data security breach when using production data

Test data may remain a significant component of the overall cost of testing, but…

The higher the quality of your test data… The higher the quality of your test efforts

In a competitive marketplace, can you afford the cost of failure?

3. Start with Requirements

At this point, we should all recognize why good test data is an important part of our test effort. Now the question becomes how to quantify good or high quality data for your specific test effort. Ultimately, the definition of good or high quality data comes down to the requirements of our test effort. As we stated in the previous section, the first quality of good test data is that it is representative of production data. Data profiling and discussions with business users are critical in understanding production data and help us to also understand what makes the data interesting. For example:

- Are there data quality concerns? Is the data complete, reliable, and timely?
- What is the relative importance or significance of specific data? What data could they not live without?
- What data or combinations of data are most commonly used? Which reports are run most often?

- Which data or combinations of data tend to be problematic? Which reports are slow or incomplete?

The more questions you ask, the more you will learn. All of this information helps to inform the data selection process, and also helps you prioritize your test efforts. Some key pieces of information you will collect during this process include:

- **Domain values:** The full range of valid and meaningful values for a data field.
  - For example, the set of domain values for a gender field might include Male, Female, Unknown, or Blank.
- **Data ranges and limits:** Especially those that define our equivalence classes.
  - For example, deposits over a set dollar amount might trigger special anti-money laundering processes. In this case, we need test data which includes deposit amounts both above and below these limits.
- **Significance of date time or sequence fields:** These may be simple indexing or audit fields, but many of these fields are used to drive reports and processing cycles. To be effective in your test effort, you need to know the details.
- **Data relationships:** This includes a wide variety of data characteristics including cross-system data mappings and sources for derived or calculated data.
- **Upstream and downstream data dependencies:** It is critical to know where data is coming from and where it might be going. This enables you to put the data into the proper business context. You may still limit your testing to direct input and output interfaces, but without at least a general understanding of the data flow, you may miss critical test cases.

And of course, you need to understand how all this information is consumed by the business users. If your organization uses a formalized design process, you may be lucky enough to find a lot of this information in existing design documentation. If not, you may just need to roll up your sleeves and dive in. Getting to know the business users and learning how to use data profiling tools are ultimately the best ways to develop a detailed understanding of the data.

The more you know about your data… The more you can optimize your test data set.

And ensuring an optimized data set is the second quality of good test data. As a preliminary means of optimizing data, you should begin with an established testing practice. Specifically, an initial set of test cases (and the associated data requirements) can be obtained through a variety of standard test techniques including:

- Boundary value analysis
- Equivalence class partitioning
- Pairwise testing and other parameter combinatorial techniques
- Model based testing

Clearly, the more detail you have about the underlying data, the more precisely you can apply these techniques and the more refined your test data requirements will become. And don't forget about the data for negative test cases such as special character data, invalid formats, and bogus field values. Be creative and try to find the most unusual and unlikely data you can.

On average only about ½ of computer instructions get exercised because the data that is needed to exercise the code is not input or simply not there!

Finally, we need to consider the overall context in which the test data will be used. The following table identifies some of the questions you'll want to ask to ensure a complete set of test data requirements.

Additional Questions to Ask	
How much data is needed?	▪ Too little data and testing will be ineffective ▪ Too much data drives up costs (both storage and maintenance) and often leads to test inefficiencies
What data is needed?	▪ Understand the potential values of data elements and their business relevance ▪ Business relevance determines risk which then establishes test priorities
When is the data needed?	▪ Data becomes stale overtime ▪ Test schedules and refresh cycles need to be properly aligned
Where will the data be needed?	▪ Coordinate data refresh and environment availability with all impacted teams
How will the data be protected?	▪ When leveraging production data sources, sensitive information may need to be obfuscated or masked
What are the dependencies?	▪ Referential integrity, cross-system integrations, or application specific requirements
Who will need the data?	▪ Can the information be shared, or does the data need to be dedicated?
What type of testing will the data be used for?	▪ Automation requires highly stable, predictable data sets where as manual testing can adapt to a higher degree of variability ▪ Performance tests require data to be either production scale or representative of production distributions
How will the data be managed?	▪ Develop processes to create and maintain you data ▪ Understand market trends and plan for future data requirements and system demands (e.g. volumes) ▪ Control access, leverage test data management tools to address data privacy concerns, and resolve usage contention issues.

As you begin to ask these questions, you will likely come up with more questions specific to your applications and environments. Use whatever time you have in your test planning and development schedules to ask as many questions as you can. Be sure to document both the questions and the answers so that you can build on them overtime. The goal is to learn as much as you can in the time you have and then learn more the next time around.

4. Understanding Your Data

As you go about developing your test data requirements, you will have a lot of questions to answer and you will need a methodical approach for organizing and analyzing the information. A test data management framework is helpful in not only organizing information about your data, but in helping you to maintain this information over time. As you build this framework, it helps to consider the following data characteristics or attributes:

- Data classifications – as they relate to your test data requirements
- Data sources
- Data selection rules – including overall quantity requirements

4.1 Data Classifications

There are a lot of different ways to classify data, but for the purpose of test data management, it makes sense to consider three fundamental classifications:

- Environmental Data
- Baseline Data
- Input Data

These classes combine to form the basis for a complete set of test data requirements and provide the context for establishing all the fundamental data management processes. Specifically, data setup, data creation, and on-going change management. We will discuss test data management later, for now we will look at each of these data classifications in more detail.

Environmental Data defines the application operational environment and is a foundational component of the test effort since it establishes our execution context. Environmental data includes:

- System configuration: Operating system, databases, application servers, hardware configuration, etc.
- User authorization, authentication, and credentials: User ID's, passwords, and system access levels for either generic, role-based or tester specific account[2].
- Configuration options: Firewall port settings, application server settings, machine resource allocations, etc.

Ideally, this data is established at the start of the project and maintained as part of your system management process. If not, you need to be sure you have a general understanding of this data since your environment is the operational context for your testing. It is unfortunately not unheard of for a test team to be pointing to the wrong environment for their test efforts. And whether you are using the wrong environment or just an improperly configured environment, you run the risk of invalidating your test results.

Baseline Data has two fundamental purposes – to establish a meaningful starting point for testing and to establish a set of expected results. The initial baseline, or starting point, is established by your test case pre-requisites and typically includes some meaningful set of business data (deployed in the appropriate environment). The exact data will depend on both the data characteristics and

the type of testing you are performing. For any type of testing you perform on a frequent or repeated basis (e.g. build smoke tests or regression tests), it is critical to establish a reliable and repeatable process for instantiating the pre-requisite test baseline. Without this process, test data issues can result in a high number of false failures leading to unnecessary analysis and an increase in test maintenance effort.

Expected results – especially at a database level – will be even more dependent on a reliable and repeatable data starting point. When testing can be run repeatedly from a known starting point, the evaluation of expected to actual results can be easily automated by simply maintaining an appropriate expected results baseline. In the absence of a reliable starting point, actual results typically need to be evaluated using manual and highly time consuming direct inspection techniques.

Input Data is the data you enter into the system under test to evaluate how it responds to the provided input. Observed behavior establishes your actual results which must then be compared to expected results to determine the correctness of the behavior. Input data is typically a component of the test case itself.

4.2 Data Sources

Data comes from a variety of sources and can be found in almost any format imaginable. To simplify the discussion of data sources, it is helpful to think in terms of the following three categories:

- Simulated or hand crafted data
- Copies or derivatives of production data sets
- Live production data

Simulated or hand crafted data is useful in cases where the production data sets may not contain values of interest for the test. Examples of this include fault injection or error checking (below the user interface level) as well as unit testing. You may also need to hand craft data when working with a new system or set of data fields without historical or existing production equivalents. Generally speaking, the use of simulated or hand crafted data is considered a best practice for unit or other white box testing activities. For integration, end to end testing, or other complex types of testing, simulating or hand crafting of data is usually too time consuming to be cost effective. There will always be exceptions to this rule, but you will usually be better off if you limit the use of hand crafted data to specific scenarios where production data samples are not available or impractical to obtain.

Copies or derivatives of production data sets form the majority of test data we should be using in our test effort. Production data is clearly the best source for obtaining data with production like characteristics. However, we want to avoid the use of full production copies and we need to ensure that the data set is properly sanitized to minimize the risk of data security breaches. A comprehensive set of data requirements establish the criteria needed to properly optimize and secure this data while also ensuring we have high quality data available to meet our testing needs. A variety of test data management products are available which have been designed specifically to

simplify and streamline the process of creating optimized, secure data from you existing production data sets.

Live production data. Today, live production data is rarely used for testing given the risk this poses to the production environment. And if your organization is still using live data for a significant portion of your test effort, it is likely that increasing data security concerns and regulatory compliance requirements will quickly eliminate this as an option. To put this in perspective, the average cost for a data breach is estimated at $214 per record and $7.2 million per incident[3]. With this kind of price tag, most organizations are just no longer willing to tolerate the potential risk. That said, there are still a few scenarios where testing against live data may be appropriate. The most common is for production install validation. For this type of testing, the production environment is seeded with the test data needed for the validation effort. Test teams need to be aware of exactly which data they are working with so as not to inadvertently impact real customer or other business data.

4.3 Data Selection Criteria

The output of the test data requirements process is both a detailed understanding of your data and a set of test specific needs. To help clarify the test specific needs, refer to the examples below:

- Savings account and checking account statement processing is performed through two independent systems.
  - The baseline data set needs to include examples of both checking and savings accounts.
  - The input data set also needs to include deposit transactions for both types of accounts.
- Costs for in-network and out-of-network providers are calculated using different rules.
  - Our baseline data set must include both in-network and out-of-network providers.
  - The input data set must include claims submitted for both the in-network and out-of-network providers.
- Valid domain values for a gender field were: Male, Female, Unknown, and Blank.
  - Our baseline data set should include at least one customer record with each of the valid field values.
  - The input data set options will then be driven by the available input mechanisms. For example, a graphical user interface might provide a drop down list containing only the above options – making an invalid entry impossible. But if the data is input through a text file, we can easily incorporate invalid values to ensure proper system error handling.
  - The input data set may also need to include other scenarios which trigger gender specific processing (such as in a healthcare system). For example, will an error be thrown if a claim is submitted by a male patient for a gynecology visit?

As you build your test base, you will develop a long list of test data criteria. During the process, be sure to account for:

- Positive and negative testing scenarios
- Possible overlaps and redundancies – either eliminating them or validating that they are appropriate.
- Demographic or statistical characteristics of the data
- Default conditions
- Cross-project dependencies

The resulting list of data criteria feeds into the next step in the process.

5. Building Test Data

If you are unable to develop a full set of requirements (either due to time constraints or lack of complete information), you may need to take a more general approach to building test data. For example, you might choose one of the following general types of options:

- Full, properly sanitized, copy of production data
- A 5% sampling of production data
- All data for offices/facilities in the state of Florida
- All data for plans A, B, and C

This general data can then be augmented or customized to meet test specific means by manually editing the data or automating the process with custom scripts.

Some teams use a combination of general and test specific data sets to meet the needs of diverse testing groups. Regardless of the approach, there are three general methods for building or creating the data for testing:

- Direct data entry via system interfaces (such as a GUI or batch file)
- Copy and edit
- Specialized test data management solution

Direct data entry is commonly used when the volume of test data needed is low and the test team has limited or no access to the underlying data storage systems. The direct data entry method has the benefit of putting full control of the data in the tester's hands, but doesn't scale well to test efforts requiring large volumes of data. Some teams will develop automation to address this issue, but it will always be more efficient (from a time and cost perspective) to instantiate the data directly at the source.

Copy and edit is a relatively easy to implement technique and allows us to leverage production data. If specific data is required for testing, the data can typically be customized via standard editing interfaces (such a SQL clients and text editors). This approach also scales well if only small amounts of data need to be customized, but usually requires a high degree of knowledge about the underlying data. Depending on the type of data you are working with, this approach can provide good results. However care must be taken when working with relational databases or data sets with complex relationships. Changes in one location need to be properly propagated to all impacted areas of the data set – which can be extremely difficult to do manually. And failure to

make complete changes to the data set typically results in false failures and wasted time chasing "red herrings".

In addition, the copy and edit approach has been adopted as a common strategy for addressing data security. In this scenario, full production data sets are copied and selected fields edited (or masked) to obscure sensitive information. There are several tools on the market designed specifically for this type of process, and many organizations have built their own using ETL tools. However, this doesn't provide the ability to optimize the size of the data set.

Specialized test data management solutions optimize our test data set. In most cases, full copies of production data are not needed and a streamlined, appropriate subset of data (where the sensitive data is masked) provides the needed coverage while increasing overall test efficiency. However, obtaining this subset can be difficult and may require complex processing, especially when working with data in a distributed heterogeneous environment. Some test data management solutions solve this problem better than others, and it may be a significant project in its own right to get the solution properly configured. So you will want to carefully evaluate options to ensure a good return on investment when choosing a test data management tool.

Of course, you don't need to limit yourself to just one approach. Since your test requirements are likely diverse, a combination of all the above techniques may be required to obtain the results you need. Again, the more you know about your data, the easier it will be to be to find the best solution for your environment.

6. Managing the Process

We've covered two of the major components of the test data management process:

- Defining data requirements
- Building the data to meet those requirements

However, data needs aren't static, and so we also have to manage change. Changes to your test data need to be controlled and coordinated or you could undo all the hard work you completed in defining and creating your data. In most cases, the change management process for your test data will be based on a standard organization change management process. If your organization doesn't already have a standard process for managing change, now might be a good time to create one and there is no shortage of good ideas.

For the purpose of this paper, we'll assume you have a change management process. For the test data management effort, the fundamental goals in applying this process are to:

- Minimize disruption to projects
- Reduce the need for back-out activities
- Ensure proper utilization of resources (specifically, eliminating unnecessary changes)

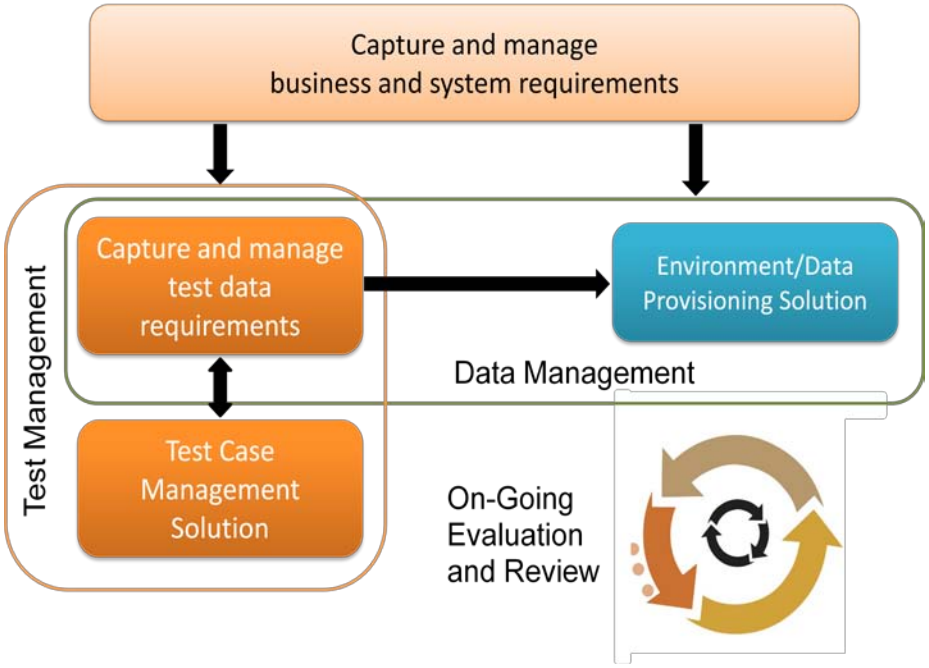
When necessary, customize or tailor your change management process to ensure these goals are met.

After each project, you should always revisit the process. This provides you with the opportunity to consistently refine the process and supports continuous improvement. As you get started, you

will likely find areas where you need additional information about your data, find opportunities to automate more of the data creation effort, and establish better ways to communicate changes. The bottom line is that you must always plan for change and you need an effective process to manage this change. It is the one thing you can count on happening in every project.

7. Automating the Process

Before beginning any automation effort, it is important that the tasks or processes you will be automating are well defined. Generally speaking, automation improves repeatability, stability and scalability of your process. But if your process currently generates poor quality data, then automation just enables you to make poor quality data faster, adding little if any value and potentially compounding the problems you are already facing. The main purpose for this paper is to define the best practices that lead to a well defined process, so we'll assume that these practices are what you have implemented. In that context, the general capabilities you need to automate as part of this process are shown in the following figure.



Using this view, we see the two fundamental components of test data management highlighted in the data management box – specifically the test data requirements and the means for building or instantiating the data which meets these requirements (shown as a provisioning capability).

Test data requirements are truly at the heart of this process since they are part of the bridge between your project requirements and the test cases themselves. They also link your test cases to the test environment in which they will run. Whatever solution you choose, you need to keep these integration points in mind – especially if you already have test case management and environment provisioning solutions in place.

You will also want a solution that helps you to manage complexity – not add to it. Simplicity is best in this case, so focus on the capabilities that add clear measureable value and avoid the frills. As we've discussed, the core capabilities you need to ensure high quality test data are:

- The ability to precisely select the data you need – ensuring an optimized test data set.
- The ability to properly handle the full range of complexity (or requirements) associated with your data set. This includes the ability to properly address data security concerns and subsetting requirements while maintaining referential integrity – both within a single data source and across your environment.

Some fundamental capability questions you need to ask are:

- Does the solution support all the types of data within your test environment?
- Does the solution provide the necessary scalability and performance to handle all your test environment data needs?

You should also consider the management and reporting features. But these should be secondary to the core functionality and fundamental capabilities. After all, fancy reports don't add value if you are unable to create the data you need.

Finally, keep in mind the skill sets you need to implement the solution. Specifically,

- Do you need a DBA? Or can someone with more fundamental data knowledge maintain information within the system?
- How long does it take to master the solution? Specifically, what level of training will be required?
- What sort of administration is required? Will you need a dedicated administrator?

The more you understand your data and your current organizational capabilities, the better positioned you will be to evaluate the available test data management solutions and ensure the best possible solution for your organization.

8. Summary

Test data has become a critical component of testing for many teams, and a thoughtful investment in test data can help you improve the overall quality of your test effort. Better testing translates into improved product quality which reduces business risk. High quality test data also lowers the cost of delivery by increasing the effectiveness and efficiency of the test effort. Making good data choices and then building the required data can be a complex task. But automated solutions are available which streamline the process while also providing the mechanism for ensuring repeatability and scalability. Appropriate automation choices can also help you to reduce and better manage this complexity, enabling your team to spend more time focused on the tasks which help find defects before they become production problems.

Ultimately, ensuring quality test data relies on good knowledge about your data. Involve business users and leverage the test data requirements process to incrementally increase your data knowledge. Formalize your test data process and automate where possible to ensure repeatability

and scalability. Understand the impacts of changes to your test data before they negatively impact your project and look for ways continuously improve.

Notes:

[1]Costs associated with software errors are from a 2002 press release from the National Institute of Standards and Technology, a branch of the US Commerce Department:

http://www.abeacha.com/NIST_press_release_bugs_cost.htm

[2] As a general best practice, generic ID's are discouraged and pose a definite security risk when used on systems which access business critical or client sensitive data. However, they may be appropriate or required to test systems where roles based security rules need to be tested. In these cases it is critical that sensitive data be obfuscated or otherwise obscured to minimize the potential business risk.

[3]Cost of data security breach are based on the following news article and are consistent with costs published by Bloomberg:

http://www.crn.com/news/security/229300791/data-breach-costs-skyrocket-response-lags.htm;jsessionid=SergPB1WYLbca2HlK+YaLg**.ecappj02

[4]Complete National Institute of Standards and Technology report used as the basis for data contained in the above press release:

http://www.nist.gov/director/planning/upload/report02-3.pdf

[5]Bloomberg report on the costs of data security breaches:

http://www.bloomberg.com/news/2011-03-08/security-breach-costs-climb-7-to-7-2-million-per-incident.html