White Paper

Scapa Technologies

## Scapa® Test and Performance Platform

- Stress Testing
- Load Testing
- Capacity Test
- Soak Testing
- Scalability Testing
- Performance Testing
- Benchmarking
- Reliability Testing
- Performance Tuning
- Performance Optimization
- Bottleneck Identification
- Maximizing User Densities
- Migration Testing
- Performance Comparison
- Server Consolidation Testing
- Diagnostic Testing
- Right Sizing Systems
- Service Availability

# BMC Remedy® Optimization

## Guide to Best Practices

bmcsoftware

bmcsoftware TAP Validated

bmcsoftware TECHNOLOGY PARTNER

# CONTENTS





For more information: quotes/case studies/references please contact Scapa : www.scapatech.com | Contact@scapatech.com
Tel. USA : +1 415 287 4126 | Tel. UK : +44 (0)131 208 0652

# EXECUTIVE SUMMARY

**BMC Remedy Action Request System**
**IT Service Management Suite**
http://www.bmc.com

*Market coverage in more than 124 countries; Major offices located in Houston and Austin, TX; San Jose, CA; Boston, MA; Amsterdam; Singapore; Tel Aviv, Israel; and Pune, India; Approximately 6,000 employees worldwide; Member of S&P 500*

Scapa Technologies is a specialist software development and consultancy firm. It has created Scapa Test and Performance Platform (Scapa TPP), a load testing, performance analysis, capacity planning, bottleneck identification and monitoring tool for testing BMC ITSM's Remedy and ITSM infrastructures.

Scapa Technologies has played key roles in the testing and implementation of some of the largest and most complex Remedy installations, working with a number of well-known organizations, such as Comcast, BT, CDW and Cerner, to name just a few, in the past 15 years.

Scapa TPP has been validated by BMC Software a key technology partner of Scapa Technologies. Scapa Technologies also has significant technology alliances with Citrix, VMWare, Microsoft and others, and works closely with a number of consulting services partner organizations to ensure Remedy and ITSM system performance across the globe.

NOTE: Scapa TPP supports all versions of BMC Remedy ITSM



For more information: quotes/case studies/references please contact Scapa : www.scapatech.com | Contact@scapatech.com
Tel. USA : +1 415 287 4126 | Tel. UK : +44 (0)131 208 0652

# INTRODUCTION

## End User Metrics are Important

The overall responsiveness of IT systems is a key indicator of their state or "health". And, since businesses employ IT systems to help them achieve goals that are critical to their revenue streams and, therefore, to their very existence, i.e. customer interactions, transactions, payment processing etc., having the ability to measure system performance from the end users' perspective becomes increasingly important.

Organizations need their mission-critical, customer-facing, systems to be stable and reliable, through migrations, upgrades and any kind of change. In particular, mid-tier Remedy Server management needs to be comprehensive or it can lead to sub-standard IT systems performance, which, in turn could have adverse effects on business continuity and service levels.

Our aim, through this "BMC Remedy Optimization Guide to Best Practices" is to equip the reader with the necessary knowledge required to help them:

- reduce risks to system performance and business continuity,
- maximize the chances of success for IT projects and their business,
- deliver Service Level Agreements (SLAs) and
- provide successful Remedy system deployments and upgrades that are responsive to end users and within budgetary constraints.

This document will highlight the importance of accurate and expeditious collection, collation, monitoring and validation of end user experience metrics via the implementation of a series of effective and efficient performance testing strategies which the reader can implement, in order to achieve these objectives.

The practices and ideas discussed are honed from our years of expertise and extensive experience in working with high performance, enterprise level Remedy installations. Keeping the theme of simplicity in mind, this guide is aimed to point out certain simple, yet consistently overlooked, strategies which are well within an organization's operational considerations, and which can be used to optimize systems of any size or complexity.



For more information: quotes/case studies/references please contact Scapa : www.scapatech.com | Contact@scapatech.com
Tel. USA : +1 415 287 4126| Tel. UK : +44 (0)131 208 0652

# Current Challenges

It is quite interesting to note that no single system configuration can be classified as a true 'fits all' solution. It is also natural to assume that system complexity, frequent updates and changing customer demands are part of the technology lifecycle. These factors, however, give rise to multiple configurations and even multiple iterations of the same system and can result in a great deal of confusion among IT decision makers. This, coupled with the failure to consider data dependency as a key factor of system performance, it is not uncommon to witness large scale chaos during key projects in enterprise IT departments. The amount of speculation that takes place in such situations only exacerbates the issues.

To avoid such situations occurring, however, a change in the mind-set of IT decision makers is necessary. IT managers, CTOs and other key decision makers need to accept the fact that system deployments rarely happen "out of the box", and that a minimal amount of modification will be required. While solutions are advertised as 'off the shelf' and 'one size fits all', they are highly unlikely to be able to preserve the integrity of the system in the long run. Time is always a key factor in decision-making and solutions are always required in the shortest of time frames. But it is imperative that solutions are researched extensively, with the emphasis on which one offers the best 'fit', on long term business requirements and effectiveness, as well as the ever-present financial considerations. Testing is the **only** way to understand whether you are making the right IT decisions for your business.

## Optimizing your testing approach

How to get the most out of testing, with minimal effort:

- Start with simple tests. Simple tests produce a lot of rudimentary information and basic metrics useful for fine tuning.
- Introduce complexities to your tests in stages, rather than all at once.
- Following simple tests with the gradual addition of complexity allows you to run tests repeatedly and achieve fast test cycles.
- Try to use real data over synthetic data whenever possible. Real data produces more accurate results.
- Keep script creation time to a minimum - a few hours if possible!
- Rather than test one tier of your Remedy system at a time, aim to include all tiers of the Remedy system in the test.
- Strive to get results as early as the system can generate them. Early results mean more time spent with the system and more time means increased confidence in using the system.
- Understand what the results mean and how these may affect the system.
- Limit the time spent on an each test cycle. With careful planning a single testing cycle can be completed in a day.



For more information: quotes/case studies/references please contact Scapa : www.scapatech.com | Contact@scapatech.com
Tel. USA : +1 415 287 4126 | Tel. UK : +44 (0)131 208 0652

# Best Practices

Here are few of the most effective examples of best practice from our experience of testing Remedy and ITSM systems:

## Traditional vs Iterative Testing Approaches

The two main approaches to testing can be broadly classified as the '**traditional**' and the '**iterative**'.

In the 'traditional' approach to testing, we define a set of generic steps that make up the entire testing activity. These are as follows:

- Deploy
- Modify
- Optimize
- Test
- Launch

The above approach is quite common. Test departments tend to lean towards this approach as it is straightforward and allows the test engineers to start building their tests early in the project lifecycle. However, the traditional approach places much less emphasis on the use of real data which results in testing system behavior in real time being highly dependent on speculation. This approach is severely compromised by the fact that systems are more liable to go into production without being thoroughly tested for readiness.

An alternative, much better and more suitable approach for complex systems, would be to introduce a 'check' activity following each project cycle. The steps involved become the following:

- Deploy / Check
- Modify / Check
- Optimize / Check
- Test / Check
- Launch

A 'check' cycle allows for the system under test to be scrutinised in greater detail. We recommend that priority is given to testing the changes rather than testing the entire system repeatedly. The 'iterative' approach adds more flexibility with respect to changes which often occur between iterations. The changes between the various iterations can be anything from varying complexity, different scripts, hardware changes to data driven changes. However, the core concept of the iterative approach remains the same; validate, optimize and execute continuously.

Some advantages of following the iterative approach are listed below.

- Early result availability
- Capacity to include modifications
- Overall confidence to scale up or down the tests based on available and future resource allocation
- Availability of before and after change values; useful to identify any issues that might have occurred after modifications and relate issues to last made changes



For more information: quotes/case studies/references please contact Scapa : www.scapatech.com | Contact@scapatech.com
Tel. USA : +1 415 287 4126| Tel. UK : +44 (0)131 208 0652

The flipside of introducing an iterative approach to testing is that a greater level of coordination is required between the various teams involved. To leverage the benefits of the iterative approach, key teams such as development, quality assurance and test teams are required to work together and in sync. This helps to quickly identify relevant and avoid irrelevant results.

The benefits of iterative testing outweigh the minor inconveniences. Iterative testing proves to be a highly effective approach when successfully integrated into the overall project plans.

## User Load Model

Making progress with the iterative approach involves testing and validating changes continuously. The advantages brought by the iterative approach would be negated, however, when testing a highly complex load model. Complicated load models make it difficult to test all but the most recent changes and are usually counterproductive.

Some fundamental strategies should be considered when deciding to apply loads for testing. As discussed earlier, it is always more reliable to base tests on real user data rather than synthetic data.

- Always start with simple tests. Even the most simple of tests can provide vital information to build upon for complex testing. For example, a simple login, application loading and logout activity can generate key metrics such as login time, load time, memory leaks etc.
- Avoid introducing complex load mechanisms in a single go. Introduce complexity in a linear and gradual manner. Gradual build-up of complexity helps the testers to get used to the system under test itself and gain the confidence to move forward.
- Continuously improve, calibrate and validate the load model
- Try cross referencing with production systems whenever possible. This will give you a good idea of where the system under test stands against currently live systems and helps set expectations more accurately.

## Test Execution

The first step in test execution is to establish baseline metrics. Quantifying each step of the user activity helps to directly compare performance before and after any changes. A more aggressive approach would be to simulate a ‘worst case’ scenario, i.e. the maximum load the system is expected to handle. Although this might sound counter-intuitive, such a method is quite useful to determine the current state of the test system even before beginning to optimize it.

Focusing on testing the incremental changes from previous steps rather than the entire system every time has proven to be highly efficient in tracking even minor changes in performance. A simple load model, as mentioned in the preceding sections, makes it easy to monitor and track even the smallest effects, by even minor changes. Execution must be further supported by continuous validation and optimization at every stage.

## Result Analysis and “Edge of Capacity” Evaluation

The edge of capacity is sometimes referred to as the ‘sweet spot’ or pinch point and is the point where a system is at its maximum throughput, whilst maintaining an acceptable response time. Beyond the edge, end user experience will suffer.



For more information: quotes/case studies/references please contact Scapa : www.scapatech.com | Contact@scapatech.com
Tel. USA : +1 415 287 4126| Tel. UK : +44 (0)131 208 0652

Following these best practices for load building and test execution, result analysis becomes quite rapid and simple:

- Before and after results comparison becomes much easier. Testers have the information they need to be able to pinpoint the exact issue rather than blindly speculate.
- Highly simplified result correlation.
- With the ability to dynamically control load by multiple drivers, testers get the most out of a single test run.

For example, figure 1 is a screenshot of a typical 30 minute test to determine the edge of capacity on a system.

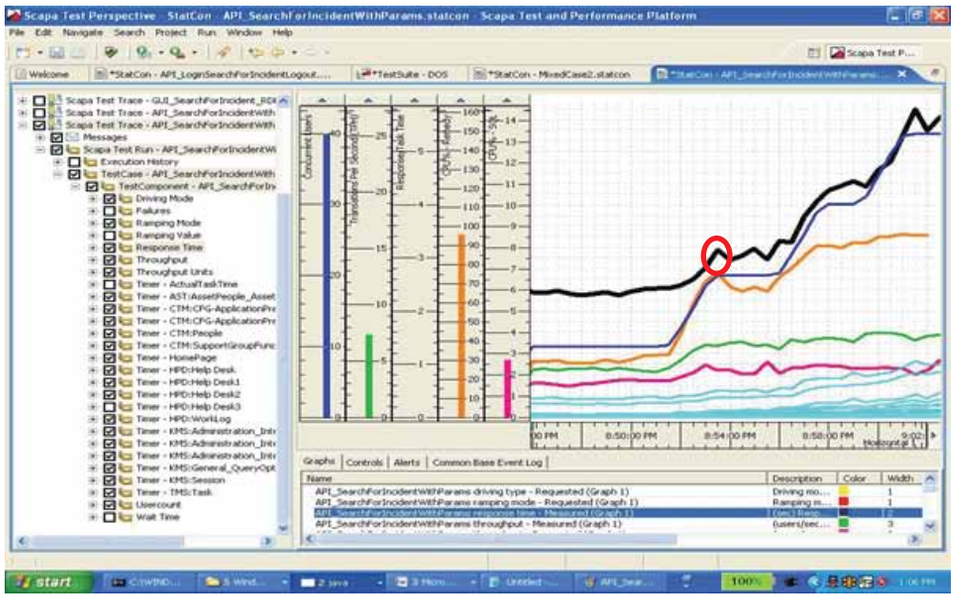


Figure 1 - the “edge of capacity” is circled in red.

The test can be controlled by multiple load drivers such as the number of concurrent users, transactions per second etc. For the purposes of demonstration let us take the number of concurrent users (indicated by the dark blue graph line) as our primary load driver. With a steady increase in user count we find a correlative increase in response times (black lines). However, at the point indicated by the circle in red, we see that response times hit a sudden high and continue to increase exponentially even with a steady rate of load. The circled point thus indicates the edge of capacity of the system.

It is also beneficial if transactions can be broken down into their various sub-transactions. Information such as the fastest and slowest performing transactions is vital in showing where most time is spent. The information can also be viewed across the multiple tiers as shown in the screen shot below (Figure 2). All this information is almost instantaneous.



For more information: quotes/case studies/references please contact Scapa : www.scapatech.com | Contact@scapatech.com
Tel. USA : +1 415 287 4126| Tel. UK : +44 (0)131 208 0652


Figure 2

## Problem Identification

A key benefit of running repeated tests with minor changes is the relative ease with which problems can be identified and analyzed. It can be said that most newly occurring issues can be traced back directly to the latest change. This allows us to recreate the test, expand the scope of the test itself and fine tune the system. If needed additional interfaces such as the Action Request API or even the database server can be tested directly to understand the effectiveness of the system as a whole.

During problem analysis multiple 'correct' configurations may be identified. Repeated modifications and validation can give rise to several configurations that might seem right for the system. However, it is the responsibility of the test engineers to test each configuration under load, document their effectiveness and recognize the most appropriate one, taking into account all service level and operational considerations. It is also possible to obtain vital information from a single test run when combining real time analysis with dynamic load management rather than repeating the same test after every single change.

We will now look at some examples to demonstrate the effectiveness of the above described problem analysis setup. The systems in question are highly sensitive to load changes and are generally more intricate to decipher under traditional problem analysis approaches.



For more information: quotes/case studies/references please contact Scapa : www.scapatech.com | Contact@scapatech.com
Tel. USA : +1 415 287 4126 | Tel. UK : +44 (0)131 208 0652

## Contention and Locks Identification

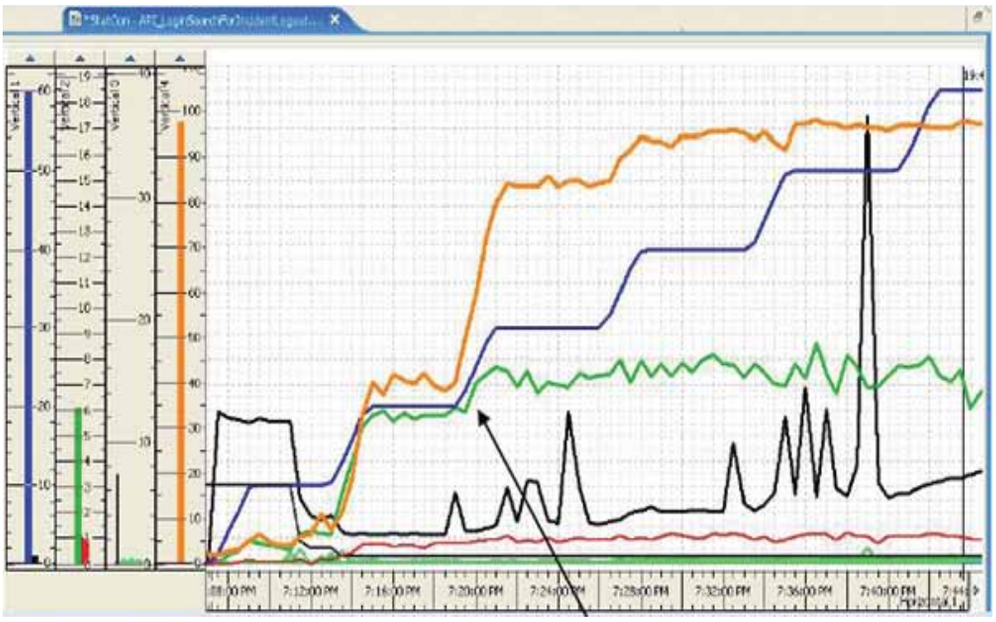

Figure 3

The blue line in Figure 3 indicates a steadily increasing load (number of users). The number of transactions handled by the system is indicated by the green line. At the point indicated by the arrow we are able to observe that the number of transactions handled by the system has reached its optimum value and has begun to stay constant, irrespective of the steadily increasing load. The black lines can be considered as an indication of the overall stability of the systems. The peaks in the graph are signs that some transactions take much longer to complete than normal, indicating that there are locks and contention for resources. Such responses occur in situations where end user experience is affected randomly, without prior signs, but system resources such as CPU and memory remain under acceptable levels most of the time. These seemingly isolated issues are difficult to identify using normal system metrics. By combining the knowledge from repeated test runs and relating it with real time analysis we are able to identify transaction bottlenecks and system performance locks.

## User Experience Consistency Evaluation

It is also possible to obtain a highly accurate portrayal of user experience across a system using real time analysis. User experience consistency can be determined by looking at variability in any given set of transactions. Variability could be a result of even slight differences in data sets, user actions or other factors. The test in Figure 4 shows how variability (yellow plots) can be affected by a simple and consistent load ramp (blue plot). The system shown in this screenshot can be classified as a highly unstable one, due to the huge differences between the minimum, maximum and average response times each indicated by the yellow graph plots.



For more information: quotes/case studies/references please contact Scapa : www.scapatech.com | Contact@scapatech.com
Tel. USA : +1 415 287 4126| Tel. UK : +44 (0)131 208 0652

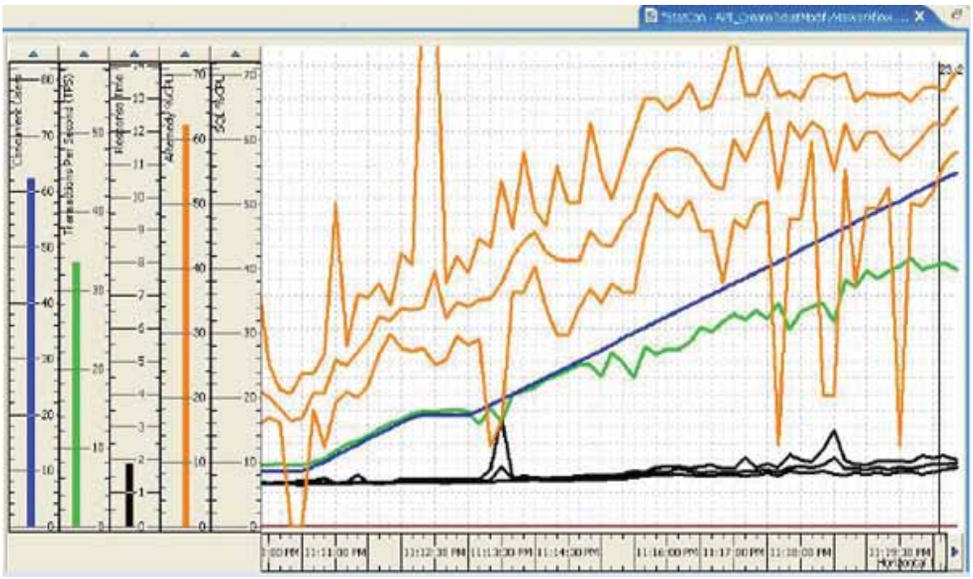

Figure 4

## Memory Leak Identification

Figure 5

Memory leaks can be useful in determining how long a system would last under simulated conditions. The above test was run for about an hour. The load (blue plot) was increased gradually and held constant. Response times (black plot) were consistent most of the time but recorded an exponential increase towards the end of the hour. A closer looks reveals a constant drop in memory metrics (yellow plot) preceded the spike in response times. Such a system would not last long in real time conditions and certainly not qualify for production.



For more information: quotes/case studies/references please contact Scapa : www.scapatech.com | Contact@scapatech.com
Tel. USA : +1 415 287 4126 | Tel. UK : +44 (0)131 208 0652

# Conclusion

As we stated initially, the aim of this Best Practices Guide is to outline just some of our experience in testing Remedy and ITSM installations across the globe. Most systems will encounter common scenarios, however, it is highly unlikely that these will be exactly alike. Every system is different and the types of best practices required to circumnavigate them are beyond the scope of this document. The key points to remember when optimizing your systems are:

- **Create simple test schedules**
  Start initial tests with simple transactions and aim to complete the testing activity in a day.

- **Define objectives clearly**
  Identify your aims and objectives clearly. This will allow you to compare your expectations to the actual test results.

- **Optimize and re-validate**
  Test and retest with multiple parameters to arrive at solid conclusions and identify the most relevant results and viable choices.

- **Gradually introduce complexity**
  Add more complex transactions at a steady rate after the initial test cycles to obtain a greater understanding of and confidence in the system

- **Understand your results**
  Relate your results to the system's performance and components. Simple test cases allow for a much more transparent comparison.

- **Understand system performance profile**
  Familiarizing yourself with the system's nuances allows you to optimize the system to its fullest and maximize your outputs. Break down your testing exercise into smaller, more manageable activities

- **Focus on relevant results**
  Undertake a sensitivity analysis to identify different load generators and how these affect the system. Experiment with different data structures, data loads, searches etc.

- **Continuously monitor changes**
  Monitor even the most minor system changes to uncover how these affect performance.

## About Scapa Test and Performance Platform for Remedy

To demonstrate the key requirements of a successful testing approach we have used the Scapa Test and Performance Platform (Scapa TPP) for Remedy. The test suite features real data utilization, quick test creation, and dynamic multi-load manipulation among its capabilities making it ideal to emphasize the influence of user data, for rapid test creation and real time analysis which are central to an effective testing approach. For a full list of features and learn more about the Scapa Test and Performance Platform visit **http://www.scapatech.com/products/remedy/**

Other Testing Solutions Available from Scapa Technologies

Citrix® XenApp™
Citrix XenDesktop®
Microsoft Terminal Services
Microsoft Remote Desktop Services
VMware® View™
Other Solutions (Enquire for details)



For more information: quotes/case studies/references please contact Scapa : www.scapatech.com | Contact@scapatech.com
Tel. USA : +1 415 287 4126| Tel. UK : +44 (0)131 208 0652



Scapa Technologies