Performance testing guidelines

Version

0.1

**Document Detail**

|  |  |
| --- | --- |
| Title: | Performance testing guidelines |
| Electronic File Name: | Performance testing guidelines.docx |
| Electronic File Location: |  |
| Author: | Process & Control, AIA HK |
| Contributors: | Process & Control, AIA HK |

**Documentation Sign-off**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Position | Comments | Signature | Date |
|  |  |  |  |  |
|  |  |  |  |  |

**Change Control**

|  |  |  |  |
| --- | --- | --- | --- |
| Issue Date | Version | Details | Approval |
| May 22, 2019 | 0.1 | Draft version |  |



Table of Contents

[1 Objectives 4](#_Toc9525495)

[2 Type of performance tests 4](#_Toc9525496)

[3 Purpose of performance testing 5](#_Toc9525497)

[4 Triggers of Performance Testing 6](#_Toc9525498)

[4.1 The need of Regular Performance Testing 6](#_Toc9525499)

[5 Performance testing process 6](#_Toc9525500)

[5.1 Gather requirements 7](#_Toc9525501)

[5.1.1 Create baseline 7](#_Toc9525502)

[5.1.2 Business Capacity Estimate (BCE) 7](#_Toc9525503)

[5.1.3 Capacity Planning Strategy (CPS) 7](#_Toc9525504)

[5.1.4 Define test scope 7](#_Toc9525505)

[5.2 Plan performance test 8](#_Toc9525506)

[5.2.1 Design test scenarios 8](#_Toc9525507)

[5.2.2 Decide test environment, data, tool requirements 9](#_Toc9525508)

[5.3 Analyze result and report test 9](#_Toc9525509)

[5.3.1 Analyze result 9](#_Toc9525510)

[5.3.2 Report test 9](#_Toc9525511)

[6 Appendix 11](#_Toc9525512)

[6.1 Performance test types 11](#_Toc9525513)

[6.2 Notes on extrapolation 12](#_Toc9525514)

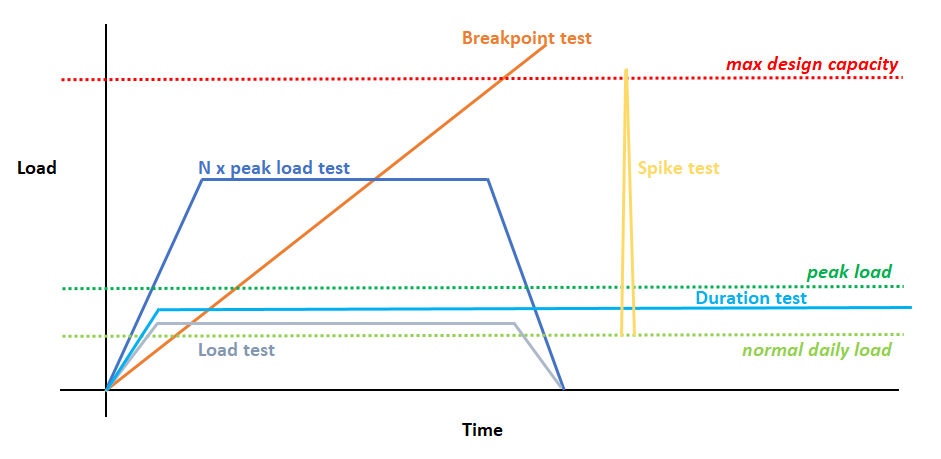
# Objectives

This document will provide the readers a high-level overview of why, when and how Performance Testing should be performed and the related procedures. The topics covered in this document are:

* The importance of performance testing
* Different types of performance testing and their objectives
* When and how to conduct these performance tests
* Criteria to determine whether performance testing is needed
* Performance testing procedure
  + Business capacity estimate (BCE) and Capacity plan strategy (CPS) to identify the performance requirements
  + Design test scenarios
  + Workload modelling for the test scenarios
  + Key performance metrics and Monitoring plan
  + Test environment and test data
  + Analyze result
  + Report the test

# Types of performance testing

The graph below depicts the different types of performance tests. The plan will identify the types of tests that are needed (breakpoint, stress, peak, spike etc.).



The table below describes the test objectives.

|  |  |
| --- | --- |
| **Performance test types** | **Objective** |
| Normal load (Load test) | To assess system performance under normal load condition |
| Peak load (Lead test) | To assess system performance under peak load condition |
| N x peak load (Stress test) | To assess system performance under N x peak load condition |
| Breakpoint test | To determine maximum capacity of the system  To understand system behavior when loading exceed max capacity |
| Spike test | To determine whether system can withstand occasional spikes in load |
| Duration test (Soak test) | To assess system performance under normal load condition for a long period of time |

# Purpose of performance testing

A performance problem in Production can cause catastrophic impact to our business. It may lead to service interruption, data corruption, and result in financial loss and damage to our reputation. What makes the situation worse is that performance problems usually cannot be fixed timely. The resolution may involve investigation, code change, or acquisition of new hardware, which takes time to implement.

Performance testing is therefore an important step to ensure the system meets performance requirements from both business and technical perspectives before the system goes live. For systems already running in Production, regular performance testing is also required to prove that the systems fulfill SLA and capacity plan strategy.

Some of the main objectives of performance testing include the following:

1. To identify:
2. Application's response time
3. Application's throughput
4. Maximum number of concurrent users supported by the application
5. Application behavior under different load conditions
6. Application’s breakpoint
7. Resources (e.g. CPU, memory, disk I/O) utilization of the system
8. The system’s bottlenecks and the triggering conditions
9. To verify application/system stability under stress or long period of operation
10. To ensure application performance meets SLA
11. To demonstrate the system has sufficient capacity to support projected business growth

These are some of the situations where performance testing for an application should be considered:

1. Development of a new application
2. Major release with considerable changes to existing design or architecture
3. Hardware/Software upgrade/migration for an application
4. Updates in performance requirements, capacity plan, or business targets

# Triggers of Performance Testing

The table below shows the conditions that will trigger a performance test:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Type of implementation** | **Scale of change** | **Load Test** | **Stress Test** | **Breakpoint Test** | **Spike Test** | **Duration Test** |
| New application | All | Mandatory | Mandatory | Mandatory | Mandatory | Mandatory |
| Change to existing application | Large project | Mandatory | Mandatory | Conditional | Conditional | Mandatory |
| Small enhancement | Mandatory | Conditional | Not needed | Conditional | Conditional |
| Bug fix | Conditional | Not needed | Not needed | Not needed | Not needed |
| New infrastructure (OS, DB, server migration) | All | Mandatory | Mandatory | Mandatory | Mandatory | Mandatory |
| Change to existing infrastructure (OS, DB, server upgrade) | Large project | Mandatory | Mandatory | Conditional | Conditional | Mandatory |
| Small enhancement | Conditional | Conditional | Not needed | Conditional | Conditional |
| Patches | Conditional | Not needed | Not needed | Not needed | Not needed |
| Changes to SLA, capacity plan, business growth projection | Significant change (+25% from original) | Mandatory | Mandatory | Conditional | Conditional | Conditional |
| Non-significant ( < +25% from original) | Conditional | Conditional | Not needed | Not needed | Not needed |

Conditional: the need of performance test should be determined from impact analysis.

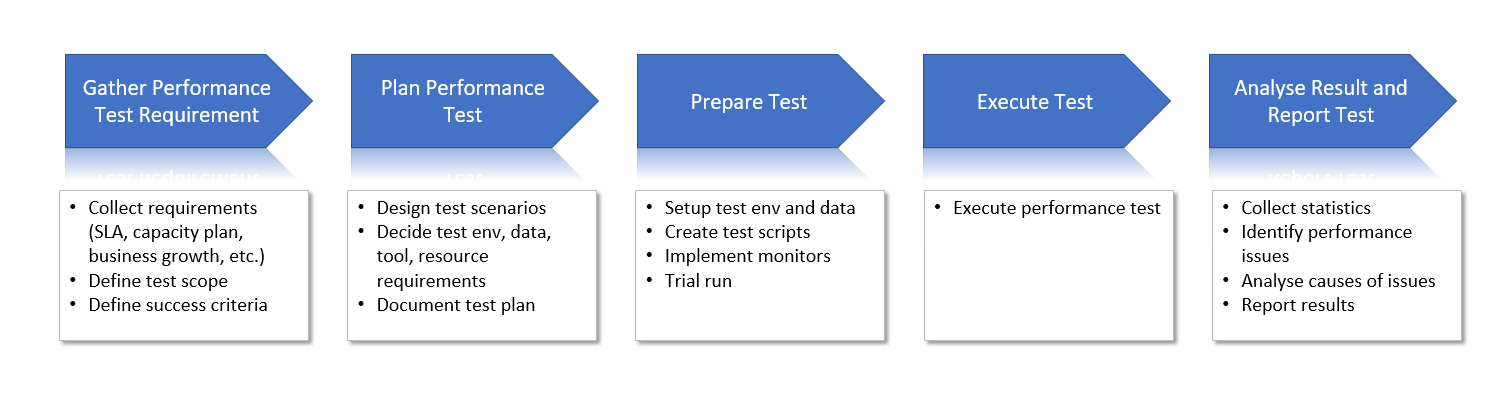
## The need of Regular Performance Testing

It is possible for an application to have no performance test performed throughout the whole year if the application does not have major rollout. However, the application owner still needs to plan to conduct load test and stress test at least once a year to ensure the system still exhibits the expected performance after having all the minor changes.

If the application owner chooses to accept the risk and skip this annual performance test, the decision needs to be justified and accepted by the CTO.

# Performance testing process

Performance testing is the key activities to verify system performance in application development. The test should begin from the requirement gathering phase and should be incorporated in every stage until the rollout to production. The following diagram depicts the performance testing process.



## Gather requirements

This section describes the key components in performance requirement gathering.

### Create baseline

If the system is already live in Production, we should create a performance baseline by collecting below information from Production.

* Daily average load
* Daily peak load
* Load pattern (i.e. load distribution throughout the day)
* Throughput (number of transaction processed per unit of time) or response time
* Corresponding CPU, memory and I/O

This baseline assists us to assess the validity of the information written in Business Capacity Estimate and Capacity Plan Strategy. It also helps us to evaluate the accuracy of test result later after the performance testing. For example, if the CPU consumptions in Production and test environment have significant difference under the same load, we will need to find out why and adjust.

### Business Capacity Estimate (BCE)

The business capacity estimate (BCE) provides a guide of current and projected application load from business perspective. The BCE will, in addition to the application KPIs and SLAs, show expected short-term, medium-term and long-term usage of the application in terms of number of concurrent clients, trade frequent, trade volume, etc. So as part of this activity, we should have details of application's past/present/future usage statistics.

The BCE must comprise of average load, peak load of the application and span a five year look ahead. In order to ensure that the system implemented can always meet business requirements, it is the business users’ responsibility to keep BCE up to date.

### Capacity Planning Strategy (CPS)

The CPS is a technical capacity document that outlines the required Production hardware components capacity for the application to operate healthily. The capacity under different load conditions and demand from future business growth should be included in the document.

Major hardware components and relevant capacity are:

* Web server disk size, CPU & memory
* Application server disk size, CPU & memory
* DB server disk size, CPU & memory
* Network bandwidth

### Define test scope

This step aims to define what business functions, system components and measurements should be included in performance testing. The inputs for this step are project non-functional requirements, BCE and CPS.

Examples of business functions are:

* Client login
* Submit claim
* Quote price

Examples of system components are:

* Front end (Application user interface)
* Application Layer
* Middleware
* Database layer

Examples of measurements are:

* Concurrent users
* Throughput (number of transaction processed per unit of time)
* Average response time
* Error rate
* CPU consumption
* Growth in disk storage size
* Network bandwidth

## Plan performance test

This section describes the key components in performance test planning.

### Design test scenarios

This step identifies the use cases to be executed in performance testing. The objective is to create a proper set of scenarios that can adequately simulate the required load conditions and cover the defined test scope.

An example of test scenarios is shown in this table.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Scenarios** | **Steps** | **Duration** | **Load profile** | **Measurement** | **Success criteria** |
| Normal load | 1. Login iShop 2. Goto WisdomTerm 3. Fill in details   -DOB, gender…   1. Quote price 2. Buy 1 year product 3. Logout | 1 hour | * 100 concurrent clients * 10 login every 30 seconds * Each client quotes & buy every 5 mins * 10 logout per minute | * Login time * Quote response time * Logout time * CPU & memory | Login time<3sec  Quote time<5sec  CPU<40% |
| Spike | 1. Login iShop 2. Goto WisdomTerm 3. Fill in details and quote price 4. Buy 1 year product 5. Logout | 30 mins | On top of “Normal load”   * Login 200 clients * Quote & buy every 3 seconds for 5 mins * 200 logout all at once | * Login time * Quote response time * Logout time * CPU & memory | Login time<3sec  Quote time<5sec  CPU<40% |

### Decide test environment, data, tool requirements

Test environment and data

The performance test environment should be as similar as possible to Production in terms of system architecture, hardware configuration, O/S version, middleware version, and amount of data in database. If Production-like environment and data cannot be obtained, we may consult infrastructure SME on the feasibility of using extrapolation to estimate test result.

Test tools

Major test tools involved in performance testing are:

* Load generation tool (e.g. LoadRunner, JMeter)
* Stubs (programs that simulate response from interfacing systems that do not take part in the test)
* Response monitoring tool (for measuring response time, throughput)
* Server monitoring tool (for measuring resources utilization)

Usually the load generation tool is also capable of measuring response time. More sophisticated testing software like Microfocus Performance Center can even monitor server statuses.

## Analyze result and report test

### This section describes the key components in result analysis and reporting.

### Analyze result

After the test is done, we have to evaluate the following.

* Are all test scenarios executed as planned? If not, can the test accomplish the test objectives and do stakeholders agree with it?
* Are test results reasonable when compare to current Production baseline collected in the beginning? If not, how should test results be adjusted, or if a re-run is needed?
* Are all success criteria fulfilled? If not, what are the follow-up actions?
* Is there any unexpected behavior observed during the test? If yes, what are the follow-up actions?

If the performance test fails, application team will have to investigate and fix the issues. This is a topic under change management. The details will not be explained here.

### Report test

Issue a test report that include the following topics.

* Test objectives
* Test requirements & success criteria
* Test scope
* Test scenarios
* Test results and evaluation on success criteria
* Findings and recommendation

The Business capacity estimate and Capacity planning strategy may need adjustment based on the findings from performance test. The application owner is responsible for taking the actions to keep the documents updated.

# Appendix

## Performance test types

|  |  |  |  |
| --- | --- | --- | --- |
| **Performance test types** | **Objective** | **Test condition** | **Success criteria** |
| Normal load | To assess system performance under normal load condition | Test against normal production load for <1 hour | 1. All defined test scenarios performed 2. System performance SLA fulfilled |
| Peak load | To assess system performance under peak load condition | Test against peak production load for <1 hour |
| N x peak load (Stress test) | To assess system performance under N x peak load condition | Test against N times of peak production load for < 1 hour  N is derived based on project non-functional requirements, BCE and CPS |
| Breakpoint test | To determine maximum capacity of the system  To understand system behavior when loading exceed max capacity | Test against ever increasing peak load until system stops responding | 1. Maximum capacity identified 2. Performance bottleneck identified |
| Spike test | To determine whether system can withstand occasional spikes in load | Test against normal/peak production load with sudden jumps to N x peak load for < 1 hour | 1. All defined test scenarios performed 2. System performance SLA fulfilled |
| Duration test (Soak test) | To access system performance under normal load condition for a long period of time | Test against normal production load for 8 hours | 1. All defined test scenarios performed 2. System performance SLA fulfilled 3. Growth in memory, storage size and system response time meet requirements |

## Notes on extrapolation

It should be noted that the capacity of performance test environment should not be vastly different from the production environment. The level of disparity will of course affect the reliability of the test results. However, this can be impractical and advances in extrapolation and auto-scaling applications and infrastructure can mitigate this if planned for and calculated carefully. The stress test must therefore be planned on test environments that have at least or are a minimum of 20-25% of capacity, when compared with the production environments. In some cases, there is no linear mathematical formula which can be used to extrapolate performance test results observed on a lower configuration to a higher configuration. E.g. if a transaction takes 4 seconds to process on a 1 CPU server, it may not necessarily process in one second on a 4 CPU server. This reinforces the need for regular performance and stress tests to create the required exponential knowledge to provide a lower variance in the extrapolation. In addition, vendors can provide detailed information on application and system performance that can support extrapolation.

An application's underlying infrastructure is constructed of many different components such as caches, web servers, application servers, Middleware and disks (I/O). Bandwidth and CDNs (content delivery networks) also play a role in its function and therefore have to be taken into consideration during scaling.

Each component behaves differently in the application according to how it was configured and scaled. However, the tiered structure makes it difficult to calculate how each should be tested and scaled. The most efficient way to run a performance test of application is by simulating an environment that is the most identical to that of production probably through a cloud setup. It is best to run this simulated performance test as early in the development phase as possible. The cloud setup allows a temporary and unlimited amount of capacity to be used for performance tests.

In case of scaled down performance environments, the expected workload should be scaled down with the same factor for the scaled down test environments in capacity though it may give near accurate results. E.g. let's say there is a requirement to test the expected peak work load of 10000 users in performance test environment and the test environments are only 25% in capacity when compared to production environment. In such a case, stress testing can be planned with 2500 users (in the same factor of test environment capacity) in the performance test environment to understand the system behavior on scaled down environment capacity. However, as mentioned in previous paragraph, scaled down performance environment is not recommended and this approach can only be adopted only if there is no option to set up an testing environment identical or close to identical to production environment, detailed analysis of the test result is mandatory at component level to avoid overlooking performance bottleneck.

Based on the test results, the required extrapolation can be done to identify and judge the application behavior in the production environment. Once the extent of workload a system can handle is analyzed in a scaled down environment, the necessary extrapolation can be done to identify the required capacity to support the full blown transaction volumes or load.