

PERformance test plan

08 December 2015

# Document Control

## Revision History

|  |  |  |
| --- | --- | --- |
| Version | Date | Description of Changes |
| V 0.1 | Nov 17, 2015 | Initial Draft |
|  |  |  |
|  |  |  |
|  |  |  |

## Distribution

|  |  |  |
| --- | --- | --- |
|  | Names | Organization and Role |
| Authors | Bhakti Gandhi | Performance Test Engineer |
| Reviewers | Mukund Gadgil |  |
|  |  |  |

## List of Abbreviations

|  |  |
| --- | --- |
| Abbreviation | Description |
|  |  |

Table of Contents

## Contents

[0 Document Control 2](#_Toc435533905)

[0.1 Revision History 2](#_Toc435533906)

[0.2 Distribution 2](#_Toc435533907)

[0.3 List of Abbreviations 2](#_Toc435533908)

[0.4 Contents 3](#_Toc435533909)

[1 Introduction 5](#_Toc435533910)

[1.1 Objectives 5](#_Toc435533911)

[1.2 Approach 6](#_Toc435533912)

[2 test scope 7](#_Toc435533913)

[In Scope Components 7](#_Toc435533914)

[2.1 7](#_Toc435533915)

[2.1.1 In Scope Performance Tests 7](#_Toc435533916)

[2.1.2 Out of scope components 7](#_Toc435533917)

[2.2 Risks, Assumptions, Issues and Dependencies 8](#_Toc435533918)

[2.2.1 Issues 8](#_Toc435533919)

[2.2.2 Risks 8](#_Toc435533920)

[2.2.3 Dependencies 9](#_Toc435533921)

[2.3 Entry Criterion - Performance Testing 9](#_Toc435533922)

[2.4 Exit Criterion - Performance Testing 10](#_Toc435533923)

[3 performance requirements 11](#_Toc435533924)

[4 testing environments 12](#_Toc435533925)

[4.1 Hardware details 12](#_Toc435533926)

[4.2 Software details 12](#_Toc435533927)

[4.3 Tools 13](#_Toc435533928)

[4.3.1 Resource monitoring tools 13](#_Toc435533929)

[4.3.2 Result analysis tools 13](#_Toc435533930)

[5 test design 14](#_Toc435533931)

[5.1 Test Design 14](#_Toc435533932)

[5.1.1 Pre Test Setup 14](#_Toc435533933)

[5.2 Test Methodology for Hybris 14](#_Toc435533934)

[5.2.1 Hybris: Mixed transaction call flow 14](#_Toc435533935)

[5.2.2 Hybris: Mixed transaction call flow for a long duration 16](#_Toc435533936)

[5.2.3 Autoship cronjob 17](#_Toc435533937)

[5.2.4 Hybris mixed call flow + Autoship cronjob 18](#_Toc435533938)

[5.2.5 Boomi REST API's 18](#_Toc435533939)

[5.2.6 Hybris mixed call flow + Autoship cron job + Boomi 19](#_Toc435533940)

[5.3 Test Methodology for EIS 20](#_Toc435534063)

[5.3.1 EIS: Hybris 21](#_Toc435534064)

[5.3.2 EIS: Hybris Long duration 21](#_Toc435534065)

[5.4 Test Methodology for RFO 21](#_Toc435534066)

[5.4.1 Enrollment Workflow SP 22](#_Toc435534067)

[5.5 Post Test 23](#_Toc435534260)

[5.6 Test Reporting 23](#_Toc435534261)

[5.7 Test Metric 23](#_Toc435534262)

[5.8 Defect Management 24](#_Toc435534263)

[6 Deliverables 25](#_Toc435534264)

[6.1 Test Plan Document 25](#_Toc435534265)

[6.2 Detailed Test Results 25](#_Toc435534266)

[6.3 Final Test Report 25](#_Toc435534267)

[7 SLA 26](#_Toc435534268)

# Introduction

This document presents the details of the approach and plan for Performance Evaluation/Benchmarking of Rodan and Fields. Testing team will use this document as a guide for performance testing during various stages of the test cycle. The test report documents the experimental strategy, experiment list, plan of the Performance Evaluation exercise in details.

The performance test strategy revolves around comparative evaluation of the benchmark tests.

## Objectives

Performance testing will determine whether or not the software meets speed, scalability and stability requirements under expected network traffic.

The objectives of performance testing are:

* Define performance test scenarios for Hybris Store Front
* Define performance test scenarios for Hybris Autoship and cron job
* Define performance test scenarios for EIS
* Define performance test scenarios for RFO
* Define performance test scenarios for Boomi web service calls.
* Bring any related performance bottlenecks to light
* Establish overall system performance under load situations.
* Analyze and report all the finding to project stakeholders.

## Approach

Execution cycle will have various activities involved as mentioned below:

Information about workload model and workflows to be considered is shared below in separate section

1. This is an iterative model where as a first step, performance is measured.
2. Then the bottlenecks are identified and solution or recommendations are provided.
3. Once these recommendations are implemented by the Development team, next round of execution will be carried out to evaluate the performance improvement and another bottleneck.
4. There could be iterative approach of 1-2 execution cycle to measure the performance improvements post fixes by development team.
5. In case when functionality changes or new feature development occurs, the product is tested for performance. Test results are compared against the baseline results. Depending upon comparative study either the story is closed or defect is raised which needs to be verified further.

# test scope

## In Scope Components

| Component |
| --- |
| Hybris store front performance under varying load |
| Hybris cron job including autoship |
| EIS enrollment calls |
| RFO enrollment SP's |
| Boomi web service calls |

### In Scope Performance Tests

Below are the types of performance tests to be executed:

* Benchmark Tests - Benchmark tests are performance tests under anticipated load. The objectives for such tests are to determine the performance of server capacity and compare it with server capacity after
* Load Test - These tests are conducted on various servers by varying the concurrency of requests to the server. The objective of this is to capture the throughput, transaction per sec and response time under varying conditions.
* Stress Test - Some tests will be conducted longer durations. To identify servers processing capabilities in stressed conditions
* Fail Open/Close test - These tests are conducted to identify the server's maximum processing capabilities by measuring the maximum number of concurrency per second. Server will be loaded with 100 or more user load until installed server goes to fail open state, then it is also observed how much time it takes to return to normal state.

## Test Strategy

Below table shows the site wise Hybris store front load distribution that we would be testing

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Traffic Type** | **Corporate** | | | | **.COM** | | | | | **.BIZ** | | | |
| **Current Load** | 20% | 50% | 100% | **Current Load** | 20% | 50% | 100% | **Current Load** | | 20% | 50% | 100% |
| **Browse** | 945 | 1134 | 1418 | 1890 | 1104 | 1325 | 1656 | 2208 | 244 | | 293 | 366 | 488 |
| **PC + RC** | 42 | 51 | 63 | 84 | 90 | 108 | 135 | 180 | 6 | | 7 | 9 | 12 |
| **Adhoc** | 11 | 13 | 17 | 22 | 6 | 7 | 9 | 12 | 3 | | 4 | 5 | 6 |
| **Enrollment** | 11 | 13 | 17 | 22 | NA | NA | NA | NA | 50 | | 60 | 75 | 100 |
| **Conversion PC to RC & RC to PC** | NA | NA | NA | NA | NA | NA | NA | NA | 18 | | 22 | 27 | 36 |
| **Sponsor Search** | 42 | 51 | 63 | 84 | NA | NA | NA | NA | NA | | NA | NA | NA |
| ***Total*** | ***1051*** | ***1262*** | ***1578*** | ***2102*** | ***1200*** | ***1440*** | ***1800*** | ***2400*** | ***321*** | | ***386*** | ***482*** | ***642*** |

Below table shows the Autoship load that we would be testing

| Cron Job | Current Load | + 50% |
| --- | --- | --- |
| AutoPlaceCRPOrderJob | 27000 | 40500 |
| AutoPlacePCPerksOrderJob | 53000 | 79500 |
| AutoPulseRenewalJob | 32000 | 48000 |

Below table shows the Boomi web service calls load that we would be testing:

| Threads | Batch Size |
| --- | --- |
| 1 | 1000 |
| 1 | 5000 |
| 1 | 10,000 |

## Out of scope components

Currently any direct evaluation of these components is be considered out of scope.

| Component |
| --- |
| Pulse application and SP |
| Iframes application and SP |
| Commissions Engine |
| Cscockpit application and SP |
| Boomi application |

## Risks, Assumptions, Issues and Dependencies

### Issues

| No. | Issue | Impact | Mitigation | Owner |
| --- | --- | --- | --- | --- |
| 1 | Dedicated test environment will be made available | High | Test executions should be planned before and in case of unplanned executions at least prior communication should be made to isolate the test environment. | None |

### Risks

| No. | Risk | Likelihood | Impact | Mitigation |
| --- | --- | --- | --- | --- |
| 1 | Insufficient resources available for support during testing | High | High – Lack of support will impact our ability to hit the milestones in the project plan. | The project must arrange appropriate resources. Support required includes system administrators, domain experts, Environment support etc. |
| 2 | Third party bugs | Medium | High - Delay in response from third party for bugs will impact the testing schedule and milestone in the project plan | The project must arrange third party support. |
| 3 | Delay on delivery of software | Low | High - If there are known issues affecting performance testing, delay in delivery of software will impact testing schedule | The project development organization should keep performance team in loop with known issues. |

### Dependencies

Performance testing has the following dependencies

| No. | Dependency | Validation |
| --- | --- | --- |
| 1 | A suitable real world-like environment is in a stable condition to allow performance testing. | Without a stable system performance testing will lose relevance. |
| 1 | Technical resource is available to support the performance tester if required. | Delays in support will adversely impact the project plan. |

## Entry Criterion - Performance Testing

To establish a managed performance test, below are the entry criteria

* Signed off Performance Test Plan (This document)
* A suitable test environment is available for the duration of the project.
* A suitable test machine is identified in same network as the test environment and is made available for the duration of the project.
* All support resources are available.
* Monitoring tools and/or processes are available.
* Logical and physical access is available as required.
* Application under test should be sufficiently stable for performance testing to commence (e.g. no runtime errors during the settlement)

## Exit Criterion - Performance Testing

The performance test exercise will be considered concluded when all of the below has happened:

* Execution of the all types of tests as per the plan.
* Performance requirements are met or plan for resolution is accepted by development and support team.
* Test completion final report is issued and signed off.

# performance requirements

Each test result should have only 5 % deviation allowed as compared to previously conducted test on same set of configurations.

It is expected to conduct performance test after every release and results to be published to project group. The performance reports could vary only by 5% from test to test

# testing environments

## Hardware details

| Purpose | Count | Hostname/IP Address | Hardware Type | Version |
| --- | --- | --- | --- | --- |
| Hybris Store Front App server | 4 | 10.223.192.68  10.223.192.165  10.223.176.76  10.223.176.143 | 16 CPU  64 GB Memory | 5.5 |
| EIS | 7 | 10.211.64.44  10.211.64.172  10.211.64.188  10.211.64.189  10.211.64.190  10.211.64.191  10.211.64.187 | 16 CPU  30 GB Memory |  |
| RFO | 1 | 172.16.15.113 | 64 CPU  512 GB Memory | Windows Server 2008 R2 Enterprise |
| Hybris db | 1 | 172.16.15.13 |  | Windows Server 2008 R2 Enterprise |
| Autoship | 4 | 10.223.80.80  10.223.80.62  10.223.80.128  10.223.80.131 | 8 CPU  64 GB Memory | 5.5 |
| Load Generator (Apache JMeter) | 4 | 10.223.224.235  10.223.249.201  10.223.242.205  10.223.242.213 | 4 CPU  16 GB Memory | 2.11 |
| Dynatrace | 1 | 10.223.224.102 | 4 CPU  16 GB Memory | 6.2 |



### 

## Tools

### Resource monitoring tools

There are many tools which can be used for resource monitoring. Some of them are as below -

* Iostat: The iostat command is used for monitoring system input/output device loading by observing the time the devices are active in relation to their average transfer rates. This would be used during autoship testing.
* Vmstat: Vmstat (virtual memory statistics) is a computer [system monitoring](http://en.wikipedia.org/wiki/System_monitor) tool that collects and displays summary information about [operating system](http://en.wikipedia.org/wiki/Operating_system) memory, processes, interrupts, paging and block [I/O](http://en.wikipedia.org/wiki/Input/output). Vmstat can specify a sampling interval which permits observing system activity in near-real time. Figure below shows the output of vmstat. This would be used during Hybris call flow testing.

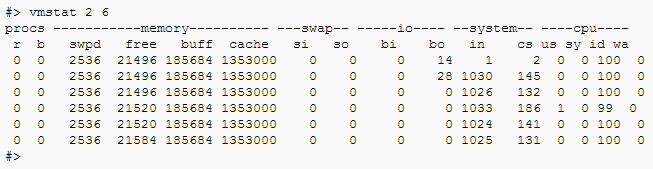


Fig 5.5 Sample Output from Vmstat

* Dynatrace: Dynatrace is profiling and application/resource monitoring tool.

### Result analysis tools

* Dynatrace - Dynatrace is profiling and application performance management tool.

# test design

## Test Design

### Pre Test Setup

Software setup –

Before starting any test, all the necessary logging should be enabled. E.g. vmstat, iostat, logging of load generating tool etc.

The JMeter test scripts should be created for all call flows.

User Profile –

~100 Adhoc subscribers will be enrolled into the system.

## Test Methodology for Hybris

Below are the details of all the phases of testing.

All the tests would be executed once with a think time enabled between each request to simulate real world traffic and once without any think time. For all workflows, think time of 30 seconds would be used between each request.























All tests would be carried out with and without Boomi web services being called.

### Hybris: Mixed transaction call flow

Purpose: Demonstrate the concurrency and scalability of Hybris while different consultants are performing a mixed transactions of all the call flows mentioned below.

Procedure: Iterative Apache JMeter test scripts will be executed simulating the mixed transaction flow. Initial attempts will be to extract the performance of a single Hybris instance. The concurrency attempted will be chosen in order to supply data points at regular intervals so that higher concurrency rates can be extrapolated from the data.



#### Current Load:

Below distribution of transactions will be used during this testing:

| Transaction Mix | Threads |
| --- | --- |
| Browse | 2293 |
| Standard Enrollment | 31 |
| Express Enrollment | 30 |
| Adhoc orders | 20 |
| PC Orders | 69 |
| RC Orders | 69 |
| RC to PC Upgrade | 9 |
| PC to RC Downgrade | 9 |
| Sponsor Search | 42 |

Based on the above distribution, the below load would be run through 4 jmeter nodes without think time:

| Load Type | Threads | Ramp-Up (sec) | Duration (min) |
| --- | --- | --- | --- |
| Transactional | 279 | 300 | 30 |
| Browse | 2293 | 450 | 30 |

Below load would be run with think time:

(The thread count for this test needs to be calculated)

| Load Type | Concurrency | Threads | Ramp-Up (sec) | Duration (min) |
| --- | --- | --- | --- | --- |
| Transactional | 279 | 5000 | 300 | 60 |
| Browse | 2293 | 10000 | 450 | 60 |

#### Current load + 20%

Below distribution of transactions will be used during this testing:

| Transaction Mix | Threads |
| --- | --- |
| Browse | 2752 |
| Standard Enrollment | 35 |
| Express Enrollment | 36 |
| Adhoc orders | 24 |
| PC Orders | 83 |
| RC Orders | 83 |
| RC to PC Upgrade | 11 |
| PC to RC Downgrade | 11 |
| Sponsor Search | 51 |

Based on the above distribution, the below load would be run through 4 jmeter nodes without think time:

| Load Type | Threads | Ramp-Up (sec) | Duration (min) |
| --- | --- | --- | --- |
| Transactional | 334 | 300 | 30 |
| Browse | 2752 | 450 | 30 |

Below load would be run with think time:

(The thread count for this test needs to be calculated)

| Load Type | Concurrency | Threads | Ramp-Up (sec) | Duration (min) |
| --- | --- | --- | --- | --- |
| Transactional | 334 | 5000 | 300 | 60 |
| Browse | 2752 | 10000 | 450 | 60 |

#### Current load + 50%

Below distribution of transactions will be used during this testing:

| Transaction Mix | Threads |
| --- | --- |
| Browse | 3440 |
| Standard Enrollment | 46 |
| Express Enrollment | 46 |
| Adhoc orders | 31 |
| PC Orders | 102 |
| RC Orders | 103 |
| RC to PC Upgrade | 12 |
| PC to RC Downgrade | 13 |
| Sponsor Search | 63 |

Based on the above distribution, the below load would be run through 4 jmeter nodes without think time:

| Load Type | Threads | Ramp-Up (sec) | Duration (min) |
| --- | --- | --- | --- |
| Transactional | 416 | 400 | 30 |
| Browse | 3440 | 550 | 30 |

Below load would be run with think time:

(The thread count for this test needs to be calculated)

| Load Type | Concurrency | Threads | Ramp-Up (sec) | Duration (min) |
| --- | --- | --- | --- | --- |
| Transactional | 416 | 5000 | 400 | 60 |
| Browse | 3440 | 10000 | 550 | 60 |

#### Current load + 100%

Below distribution of transactions will be used during this testing:

| Transaction Mix | Threads |
| --- | --- |
| Browse | 4586 |
| Standard Enrollment | 61 |
| Express Enrollment | 61 |
| Adhoc orders | 40 |
| PC Orders | 138 |
| RC Orders | 138 |
| RC to PC Upgrade | 18 |
| PC to RC Downgrade | 18 |
| Sponsor Search | 84 |

Based on the above distribution, the below load would be run through 4 jmeter nodes without think time:

| Load Type | Threads | Ramp-Up (sec) | Duration (min) |
| --- | --- | --- | --- |
| Transactional | 558 | 500 | 30 |
| Browse | 4586 | 550 | 30 |

Below load would be run with think time:

(The thread count for this test needs to be calculated)

| Load Type | Concurrency | Threads | Ramp-Up (sec) | Duration (min) |
| --- | --- | --- | --- | --- |
| Transactional | 558 | 5000 | 500 | 60 |
| Browse | 4586 | 10000 | 550 | 60 |

### Hybris: Mixed transaction call flow for a long duration

Purpose: Demonstrate the stability of Hybris while different consultants are performing a mixed transactions of all the call flows mentioned above.

Procedure: Iterative Apache JMeter test scripts will be executed simulating the mixed transaction flow.

##### Current Load:

Below distribution of transactions will be used during this testing:

| Transaction Mix | Threads |
| --- | --- |
| Browse | 2293 |
| Standard Enrollment | 31 |
| Express Enrollment | 30 |
| Adhoc orders | 20 |
| PC Orders | 69 |
| RC Orders | 69 |
| RC to PC Upgrade | 9 |
| PC to RC Downgrade | 9 |
| Sponsor Search | 42 |

Below load would be run with think time:

(The thread count for this test needs to be calculated)

| Load Type | Concurrency | Threads | Ramp-Up (sec) | Duration (hours) |
| --- | --- | --- | --- | --- |
| Transactional | 279 | 5000 | 300 | 6 |
| Browse | 2293 | 10000 | 450 | 6 |

##### Current load + 20%

Below distribution of transactions will be used during this testing:

| Transaction Mix | Threads |
| --- | --- |
| Browse | 2752 |
| Standard Enrollment | 35 |
| Express Enrollment | 36 |
| Adhoc orders | 24 |
| PC Orders | 83 |
| RC Orders | 83 |
| RC to PC Upgrade | 11 |
| PC to RC Downgrade | 11 |
| Sponsor Search | 51 |

Below load would be run with think time:

(The thread count for this test needs to be calculated)

| Load Type | Concurrency | Threads | Ramp-Up (sec) | Duration (hours) |
| --- | --- | --- | --- | --- |
| Transactional | 334 | 5000 | 300 | 6 |
| Browse | 2752 | 10000 | 450 | 6 |

##### Current Load + 50%

Below distribution of transactions will be used during this testing:

| Transaction Mix | Threads |
| --- | --- |
| Browse | 3440 |
| Standard Enrollment | 46 |
| Express Enrollment | 46 |
| Adhoc orders | 31 |
| PC Orders | 102 |
| RC Orders | 103 |
| RC to PC Upgrade | 12 |
| PC to RC Downgrade | 13 |
| Sponsor Search | 63 |

Below load would be run with think time:

(The thread count for this test needs to be calculated)

| Load Type | Concurrency | Threads | Ramp-Up (sec) | Duration (hours) |
| --- | --- | --- | --- | --- |
| Transactional | 416 | 5000 | 400 | 6 |
| Browse | 3440 | 10000 | 550 | 6 |

##### Current Load + 100%

Below distribution of transactions will be used during this testing:

| Transaction Mix | Threads |
| --- | --- |
| Browse | 4586 |
| Standard Enrollment | 61 |
| Express Enrollment | 61 |
| Adhoc orders | 40 |
| PC Orders | 138 |
| RC Orders | 138 |
| RC to PC Upgrade | 18 |
| PC to RC Downgrade | 18 |
| Sponsor Search | 84 |



The below load would be run for 6 hours with think time using 4 jmeter nodes.

(The thread count for this test needs to be calculated)



| Load Type | Concurrency | Threads | Ramp-Up (sec) | Duration (hours) |
| --- | --- | --- | --- | --- |
| Transactional | 558 | 5000 | 250 | 6 |
| Browse | 4586 | 10000 | 300 | 6 |

### Autoship cronjob

Purpose: Demonstrate the performance of Autoships and cron jobs.

Procedure: 3 Autoship jobs would executed

- AutoPlaceCRPOrderJob

- AutoPlacePCPerksOrderJob

- AutoPulseRenewalJob

4 cron jobs would be executed:

- pcPerksSoftTerminationCronJob

- crpPulseSoftTerminationCronJob

- autoRenewalEmailJob(Consultants)

- renewedConsultantshipEmailJob

The distribution of the cron jobs is as follows:

#### Current Load:

| Cron Job | Current Load |
| --- | --- |
| AutoPlaceCRPOrderJob | 27000 |
| AutoPlacePCPerksOrderJob | 53000 |
| AutoPulseRenewalJob | 32000 |

#### Current Load + 50%:

| Transaction Mix | Orders |
| --- | --- |
| AutoPlaceCRPOrderJob | 40500 |
| AutoPlacePCPerksOrderJob | 79500 |
| AutoPulseRenewalJob | 48000 |

The distribution of the cron jobs would be finalized as we gather some data from production.

The test data for autoships and cron jobs would be generated by manipulating the scheduling dates of the orders already present in the database.

### Hybris mixed call flow + Autoship cronjob

Purpose: Demonstrate the performance of Hybris while mixed call flows and cron jobs run in parallel.

Procedure: Similar Apache JMeter test scripts will be run with a concurrency along as explained below:

The mixed load considered is 50% less than the current US traffic load.

| Transaction Mix | Threads/Orders | Duration (hours) |
| --- | --- | --- |
| Autoship cron jobs | 168000 | 3 |
| Browse | 1147 | 3 |
| Transactional Flow | 140 | 3 |

### Boomi REST API's

Purpose: Demonstrate the performance of Boomi REST API's reading data from Hybris DB.

Procedure: Iterative Apache JMeter test scripts will be run to simulate Boomi API's which read data from Hybris.

| API's |
| --- |
| Categories:getCategories  updateOrders:updateOrders |
| ResassignSponsor:ResassignSponsor |
| autoshipSwapLog:findByDateRange |
| autoshipCancellation:findByDateRange |
| autoshipLog:findByDateRange |
| Products:getProducts |
| commissioninfo:commissioninfo |
| Orders:findUpdatedOrders |
| ReturnOrders:findReturnOrders |
| terminate:terminate |
| Autoships:findUpdatedAutoships |
| Consignment:Consignment |

Tests would be carried out at different batch sizes and the throughput would be measured.

| Threads | Batch Size |
| --- | --- |
| 1 | 1000 |
| 1 | 5000 |
| 1 | 10,000 |

### Hybris mixed call flow + Autoship cron job + Boomi

Purpose: Demonstrate the performance of Hybris with a mixed transaction of store front call flows, autoship cron jobs and Boomi running in parallel.

Procedure: Similar Apache JMeter test scripts will be run with lower concurrency along with some Autoship cron jobs and Boomi running in parallel.

The mixed load considered is 50% less than the current US traffic load.

| Transaction Mix | Threads/Orders | Duration (hours) |
| --- | --- | --- |
| Autoship cron jobs | 168000 | 3 |
| Browse | 1147 | 3 |
| Transactional Flow | 140 | 3 |
| Boomi | 1 (1000 batch size) | Run every 5 mins |











## Test Methodology for EIS

US and Canada production data analysis was done for peak hours. Based on this data, the below list of API's were shortlisted to be used for during performance testing:

|  |
| --- |
| HYBRIS |
| GetPendingPolicy |
| Login |
| SaveAccount |
| GetBasicAccount |
| GetSiteUrlAvailability |
| AuthorizeCreditCard |
| Capture |
| ValidateAccountEmailSSN |
| SaveShippingProfile |
| SaveAccountAddressRecordAR |
| ValidateCreditCard |
| SaveBillingProfile |
| AccountPolicyAcceptance |
| SaveSiteUrls |

### EIS: Hybris

Purpose: Demonstrate the concurrency and scalability of EIS with incoming requests from Hybris only.

Procedure: Iterative Apache JMeter test scripts will be executed simulating Hybris enrollment load. Initial attempts will be to extract the performance of a single EIS instance. The concurrency attempted will be chosen in order to supply data points at regular intervals so that higher concurrency rate can be extrapolated from the data.

| Threads | Ramp-Up (sec) | Duration (min) |
| --- | --- | --- |
| 20 | 10 | 5 |
| 50 | 25 | 5 |
| 100 | 50 | 5 |
| 150 | 75 | 10 |
| 300 | 150 | 10 |
| 350 | 180 | 10 |
| 400 | 200 | 10 |

### EIS: Hybris Long duration

Purpose: Demonstrate the stability of EIS for ~6 hours with incoming requests from Hybris and Pulse

Procedure: Iterative Apache JMeter test scripts will be executed simulating a mixed load of Hybris enrollment.

The max threads would be decided based on the results of test case 5.3.1. 40% less than the max sustainable threads would be used to run for 6 hours.

| Threads | Ramp-Up (sec) | Duration (hours) |
| --- | --- | --- |
| 300 | 250 | 6 hours |

## Test Methodology for RFO

Below are the details of all the phases of testing.

### Enrollment Workflow SP

Purpose: Demonstrate the concurrency and scalability of stored procedures to load account data into RFO.

Procedure: Iterative Apache JMeter test scripts will be executed simulating the load from Hybris to RFO using SP's. The concurrency attempted will be chosen in order to supply data points at regular intervals so that higher concurrency rates can be extrapolated from the data.

This test will be carried out without Boomi and Autoship. Below SP's will be tested:

| Stored Procedure |
| --- |
| RFO\_Accounts. usp\_Accounts\_SelectByNameOrId |
| RFO\_Accounts.usp\_ValidateAccountEmailSsn |
| RFO\_Accounts.usp\_ValidateAccountEmailSsn |
| RFO\_Accounts.usp\_Site\_SelectByAccountPrefix |
| RFO\_Accounts.usp\_Account\_Upsert |
| RFO\_Accounts.usp\_EmailAddress\_Upsert |
| RFO\_Accounts.usp\_Account\_Get |
| RFO\_Accounts.AccountSecurity\_Upsert |
| RFO\_Accounts.usp\_AccountPhone\_Upsert |
| RFO\_Accounts.usp\_FullEnrollments\_Upsert |
| RFO\_Accounts.AccountNotes\_UpsertSF |
| RFO\_Accounts.usp\_AccountAddresses\_Upsert |
| RFO\_Accounts.usp\_AccountAddress\_SelectByAddressId |
| RFO\_Accounts.usp\_AccountPaymentProfile\_Upsert |
| RFO\_Accounts.usp\_PaymentProfile\_SelectByPaymentProfileID |

| Threads | Ramp-Up (sec) | Duration (min) |
| --- | --- | --- |
| 1 | 1 | 5 |
| 50 | 25 | 5 |
| 100 | 50 | 5 |
| 200 | 100 | 10 |
| 400 | 150 | 10 |
| 600 | 300 | 10 |
| 800 | 400 | 15 |
| 1000 | 600 | 15 |



























## Post Test

After successful completion of the test all the desired metrics are collected. They are collected from many sources such as load generator tool, monitoring tools or some combination of tools.

Reports are prepared from the collected data and are published across the development and QA teams.

If any performance bottleneck is identified or SLAs are not met, appropriate defects will be raised in the defect tracking system and are assigned to respective teams. Analysis will be performed on the issues and relevant data will be provided on the tickets to assist the developers with troubleshooting.

## Test Reporting

Performance team shall have the results collected and analyze them after each test execution. Performance team will be using scripts and tools to prepare the performance report for the executed test. The report along with observations for each planned test shall be distributed to the nominated project group for review once complete.

Once the final testing has been complete in line with the approach, the final test report shall be produced and released to the project group for sign off.

We shall also be sending status reports about the activities that were done or planned by the performance testing team to project group.

## Test Metric

While carrying out performance tests following are the key metrics that are collected –

* + Transactions per Second
  + Average Response Time.

These metrics are specific to the type of performance test.

Following are the system metrics that will be captured on the server nodes –

* + CPU related counters
  + Memory related counters
  + DISK IO related counters
  + Network usage related counters
  + Other system and product counters as relevant.

These metrics are collected for every test. From the comparative study of baseline and benchmarking test one can measure impact of the agent on the system resources.

## Defect Management

All issues raised by performance team will be registered in Jira. These will be assigned to project group or appropriate person who will be the owner of the defect.

# Deliverables

Below are the details of the project deliverables from the performance team.

## Test Plan Document

This relates to this document. It details the planned activities, and the roles and responsibilities of the activities involved in the exercise.

## Detailed Test Results

The interim test reports which will be shared after every test execution.

## Final Test Report

This report summarizes the test results and identifies / illustrates any notable findings and tunings. This report is issued to the project team for sign off and to assist when making the decision to deploy to production.

# SLA

As per discussion with DPM, no R+F standard SLA's have been defined. Thus we have considered the ecommerce industry standard SLA's as below:

* + Non transactional page response times should be less than 3 sec
  + Transactional page response times should be less than 5 sec

