**Benchmarking Randomizer Performance Success Criteria LLD**

**1.0 Metadata**

* **Author**: [Apoorva Vasamsetti](https://quip-amazon.com/QUQ9EAwnTKe)
* **Owning Team**: PubTech Simulation and Experimentation
* **Executive Summary**: Introduction of benchmarking component geared towards identifying additional latency introduced by API plugins
* **Audience**: PubTech Leads, SimEx Science and Engineering Teams;
* **Reference Document Link**: [[LLD] PubTech Experimentation Randomizer](https://quip-amazon.com/Hda7AOYSxpSO)
* **Timeline**: [5.4 Major Milestones](https://quip-amazon.com/u27tASx8NO0a#temp:C:UQT2b3258e9a70c45339213df6c7)

**1.1 Review Goals**

Get initial feedbacks on Implementation and get required approvals on data-contacts with the partner teams involved for this project.

* Decide and finalize on P0, P1, P2 ...Pn items by working backwards from customer requirements and need by dates.
* Identify Benchmarking component is to avoid the Failure Scenarios/ Risks mentioned in LLD Doc.  
  What are the important metrics ?  
  Currently API plugin identified till now (**GetTreatmentAssignment)**  
  iterations = 100-200**,** time **=** 1**,** timeUnit **=** **TimeUnit.MICRO**SECOND?
* LLD - Implementation of Benchmarking Component
* Implementation of Benchmarking component in Randomizer Pipeline.

**1.2 Review History**

|  |  |  |
| --- | --- | --- |
| **Review Audience** | **Review Disposition/Key Callouts/Open Questions** | **Action Items** |
| Mari Koba (Principal Engineer) |  |  |
| Manish Singh (SDM) |  |  |
| SimEx Engineering Team |  |  |

**1.3 Glossary**

* **JMH(Java Microbenchmark Harness**)**:**Java harness for building, running, and analysing nano/micro/milli/macro benchmarks written in Java and other languages targeting the JVM: [openjdk.net/jmh](http://openjdk.java.net/projects/code-tools/jmh/).
* **Benchmark test**: Executes a series of steps under a known load and measures key metrics for comparison between builds. The steps, load, and all other variables remain consistent across each benchmark (which enables accurate comparison).
* **Latency test:** Measurement of duration required to execute a fundamental or critically important task; the task often being depended on by outside teams.
* **Throughput test:** Measurement of performance capacity of a system in terms of units of data transmitted or processed per unit time

**2.0 Overview**

This document serves as a formal recommendation for the randomizer deployment pipeline to be benchmarked for technical feasibility. This benchmarking should be a step in the pipeline to ensure that all changes we release meet our SLAs.

**3.0 Requirements**

**3.1 Functional Requirements**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Title** | **Priority** | **User Story** |
| **FR#1** | Benchmark Randomizer Plugin | P0 | Benchmarking component will be integrated with Randomizer deployment pipeline such that it runs pre-defined benchmarks geared towards identifying additional latency introduced by Randomizer plugin and fails the build if Randomizer plugin exceed pre-defined latency criteria |
| **FR#2** | Benchmark Deal Partitioning API Plugin | P1 | Benchmarking component will be integrated with randomizer deployment pipeline such that it runs pre-defined benchmarks geared towards identifying additional latency introduced by API GetPartitionAssignment to support partition randomization and fails the build if plugin exceed pre-defined latency criteria |

**3.2Non- Functional Requirements**

|  |  |  |
| --- | --- | --- |
| **ID** | **Priority** | **User Story** |
| **NFR#1** | P0 | As an experimenter defining the SLA's for all benchmarked API's, to address failure scenarios |
| **NFR#2** | P1 | P0, Initial Design is focussed on Randomizer pipeline requirements for experimentation. |

**4.0 Scope**

**Findings :**

* Currently according to the Randomizer LLD doc The experimentation client will offer a **GetTreatmentAssignment**APIto support randomization.

**4.1 In Scope**

* Benchmark Randomizer API Plugin
* Benchmark Deal Partitioning API Plugin

**4.2 Out of Scope**

* Other packages that need to be benchmarked without clear requirements in other experimentation pipelines will be taken up later for benchmarking.
* We need to benchmark the performance of all recognized component operations and provide a flexible way to extend the benchmarking mechanism at modular level inside benchmark package.

**5.0 Design**

A new package called **ExperimentationPerformanceBenchmark**will be created and used for Randomizer performance test. The benchmark should leverage the custom indices parameter to test the custom component of randomizer's internal index performance. The benchmarking test doesn't need Randomizer to expose any internal logic.  
In the pipeline this is a step that will run as an approval step and evaluate the performance of each type of evaluation causing latencies.  
The overall goal is to have a low latency and increased throughput in all experimentation system pipelines, therefore the evaluation is done on the basis of :  
**Timing**: These timings may be the full runtime of the application or can be timings for part of the runtime, for example, time per iteration or time per assignment trigger.  
**Throughput**: Measures the amount of assignment triggers that can be processed per unit of time.

**5.1 Success Metrics**

**Randomizer API Plugin:**  
GetTreatmentAssignmentAndRecordTrigger will create a latency response and it is benchmarked for 1 Nanosecond.

|  |  |
| --- | --- |
| **Metric Name** | **GetTreatmentAssignment** |
| avg | 10.7 µs |
| p50 | 8.20 µs |
| p95 | 49.20 µs |
| p99.9 | 67.44 µs |
|  |  |

**Deal Partitioning API Plugin:**

**5.2 Implementation**

Simply annotating a method with @Benchmark will tell JMH to run its benchmark tests on that method. The default benchmark mode is Throughput but JMH supports several different metrics which can be specified using the @BenchmarkMode annotation.

**import** org.openjdk.jmh.annotations.Benchmark**;**  
import org.openjdk.jmh.annotations.BenchmarkMode;  
import org.openjdk.jmh.annotations.Mode;  
  
**@Measurement(iterations = 100-200(adjustable), time = 1, timeUnit = TimeUnit.NANOSECONDS)**  
// or  
**@Warmup(iterations = 100-200(adjustable), time = 1, timeUnit = TimeUnit.NANOSECONDS)**  
public class Benchmarks {  
  
 @Benchmark  
 @BenchmarkMode(Mode.SampleTime)  
 @Benchmark @BenchmarkMode(Mode.Throughput)   
 @OutputTimeUnit(TimeUnit)  
 @Threads(1)  
 public String GetTreatmentAssignment(@NonNull final BenchmarkState state) {  
 int experimentRondomizerIndex = state.providerCounter;  
 // API plugin input that need benchmarking  
 return getExperimentAssignment(\*, state.sessionInfo);  
 }   
}

@Benchmark will tell JMH to run its benchmark tests on that method

**5.3 Testing**

* **Unit tests:** We will aim for 80% coverage across functional code paths.
* **Integration and load tests:**This will be covered in the ‘Experimentation client end to end integration test plan’.
* **Local Testing:**

Run brazil-build benchmark to build the package and run JMH benchmarks contained in the benchmarks directory in the package root.  
It's a good idea to run brazil-build clean before another benchmark run.  
Sample Command:

brazil-build benchmark -DLimit=?+Sheet1!A2 -DthresholdsMicrosecs='{  
 "getTreatmentAssignmentAndRecordTrigger":{  
 "scorePercentiles":{"50.0":8.20,"95.0":4.90,"99.0":67.44},  
 "score":3},}

**5.4 Major Milestones**

|  |  |  |  |
| --- | --- | --- | --- |
| **Milestone 0: Recognizing All Parameters for LLD** | **Status** | **Target Date** |  |
| Finalizing Latency causing components in Randomizer | Complete | 4/5/2024 |  |
| Finalizing Benchmarking Metrics | Complete | 4/5/2024 |  |
| Setting Up Benchmarking Package | In Progress | 4/5/2024 |  |
| **Milestone 1: Prepare LLD + Implementation** | | | |
| First Draft - P0 requirement | Complete | 4/8/2024 |  |
| Implementation of Package | In Progress | 4/19/2024 |  |
| Finalize P1 requirement | Not Started | Today |  |
| **Milestone 2: Implementations + Integration Tests** | | | |
| Final Design Document | In Progress | 4/24/2024 |  |
| Integration Tests Code complete | Not Started | 4/29/2024 |  |
| **Milestone 3: Deal Partitioning API Implimentation** | | | |
|  |  |  |  |
|  |  |  |  |

**6.0 FAQ**

1. **DataIngestionWorkflow (Fetching Data from remote source is done via DataIngestionWorkflow provided by PDF Library) need benchmarking metrics too?**  
   Currently we do not need benchmarking for DataIngestionWorkflow as it is run on**an async thread** to keep the local Experiment Metadata cache updated.  
   Reference: [Fetching Data from remote source is done via DataIngestionWorkflow provided by PDF Library. This Workflow object supports: [LLD] PubTech Experimentation Randomizer](https://quip-amazon.com/Hda7AOYSxpSO#temp:C:QUFb34ef798e2334904ab6e25909)
2. **What are the defined SLAs for this system?**

**7 Reference**

* [PubTech Experimentation Randomizer Recommendation](https://quip-amazon.com/vSMEATLvrDcw)
* [[LLD] PubTech Experimentation Randomizer](https://quip-amazon.com/Hda7AOYSxpSO)
* JMH: <https://w.amazon.com/bin/view/JMH>
* <https://w.amazon.com/bin/view/MonitoringTeam/MetricAgent/Troubleshooting-new#HOutOfMemoryErrororMaxmetricsnumberreached>
* <https://w.amazon.com/bin/view/Weblab/Service/WeblabAllocationProvider/1_8/WeblabUtils>
* <https://pipelines.amazon.com/pipelines/LabDaemon-release>
* <https://code.amazon.com/packages/WeblabAllocationProviderBenchmark/blobs/mainline/--/benchmarks/com/amazon/weblab/service/Benchmarks.java>