INTLAND SOFTWARE

codeBeamer ALM's Official Scalability Performance Test Report 2019

- 1,000 concurrent users
- 10 million work items
- 35,000 requirements in a single document

Please contact sales@intland.com for more information about high availability clustering!

codeBeamer ALM's Official Scalability Performance Test Report 2019

1,000 concurrent users, 10 million work items, 35,000 requirements in a single document

Table of Contents

The purpose of this document

In February 2019, we carried out a comprehensive performance testing session of codeBeamer ALMwith 1,000 users, 10 million artifacts (work items), and 35,000 requirements in a single document.

The purpose of this document is to give you an overview of both the process and the environment that we have used for testing the performance of our ALM platform in order to help you interpret and understand the results. These tests may be repeated with the same configuration any time, and can be reasonably expected to yield similar results.

Testing showed that codeBeamer ALM worked swiftly and smoothly with 10 million work items (artifacts). If your environment requires the management of an even larger volume of work items, please contact us for more performance data.

For this performance test we used a Fujitsu PRIMERGY TX2550 M4 Server with 12 CPUs x Intel(R) Xeon(R) Gold 6128 CPU and 128 GB RAM, running a Linux (CentOS 7) with an Oracle 12c database. These performance tests do not yet cover Windows-based installations or MySQL databases.

Hardware Architecture

Test server

The server used for testing was a Fujitsu PRIMERGY TX2550 M4 Server with 2 CPUs x Intel(R) Xeon(R) Gold 6128 CPU and 128 GB RAM of total RAM. The same piece of hardware was used for all the performance tests. The approximate market price of the test server is around $10,000.

VMware ESXi 6.7 was running on the server.

Hardware	
Manufacturer	FUJITSU
Model	PRIMERGY TX2550 M4
CPU	
Logical processors	24
Processor type	Intel(R) Xeon(R) Gold 6128 CPU @ 3.40GHz
Sockets	2
Cores per socket	6
Hyperthreading	Yes, enabled
Memory	127.43 GB
Persistent Memory	0 B
Virtual flash	0 B used, 0 B capacity
Networking	fujitsu1.fritz.box
Storage	
Physical adapters	4
Datastores	

Name	Type	Capa...	Free
fujitsu2	NFS	931.3...	197 GB
NVMe_M.2	VMFS6	953.7...	38.11...
SSD1	VMFS6	1.86 TB	1.44 TB
SSD2	VMFS6	1.86 TB	1.86 TB
ESXi6.7	VMFS6	22.25...	20.84...

The tests ran on the same ESXi server. It is recommended to use **two separate servers** for codeBeamer and Oracle instances and at least 10 Gbit Network is required between these servers archiving the optimal performance on the Production environment.

codeBeamer ALM instance

The virtual server of this codeBeamer ALM instance ran with the following configuration:

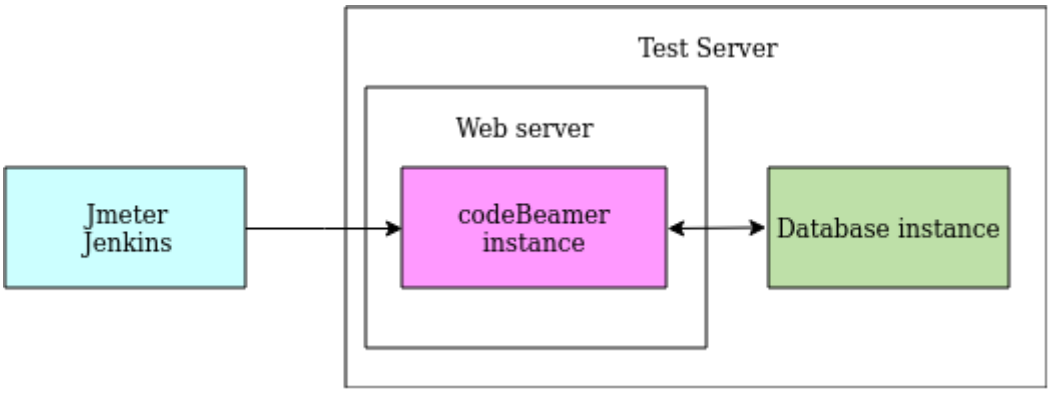
Hardware Configuration	
CPU	
CPU Cores	10
Sockets	10
Core per Socket	1
IOMMU	Disabled
Memory	32 GB
Hard disk 1	250 GB
USB controller	USB 2.0
Network adapter 1	VM Network (Connected)
Video card	0 B
CD/DVD drive 1	ISO [fujitsu2] ISO_images/CentOS-7-x86_64-DVD-1810.iso Select disc image
Others	Additional Hardware

Database instance

The virtual server of this Oracle instance ran with the following configuration:

Hardware Configuration	
CPU	
CPU Cores	10
Sockets	10
Core per Socket	1
IOMMU	Disabled
Memory	32 GB
Hard disk 1	600 GB
USB controller	USB 2.0
Network adapter 1	VM Network (Connected)
Video card	0 B
CD/DVD drive 1	ISO [fujitsu2] ISO_images/CentOS-7-x86_64-DVD-1810.iso Select disc image
Others	Additional Hardware

Test topology



Software components and versions

Both codeBeamer ALM and Oracle virtual machines ran on CentOS 7.

codeBeamer ALM was configured to run with OpenJDK version 8 and a maximum of 16 GB heap size (from max. available 32 GB memory).

codeBeamer ALM	version 9.3.0-final version
Server	Apache Tomcat 8.5.37
Java	OpenJDK 1.8.0_191
Database	Oracle Database 12c Enterprise Edition Release 12.1.0.2.0
JDBC driver	oracle.jdbc.driver.OracleDriver, Implementation-Version: 12.1.0.1.0-ProductionBuild-12
OS	CentOS Linux 7

codeBeamer ALM configuration

Tomcat server configuration

ThreadPool configuration

To serve a higher load we increased Tomcat's thread pool count in server.xml:

```
 <Executor name="tomcatThreadPool" namePrefix="catalina-exec-"
maxThreads="500" ></Executor>
```

Database connection configuration

To match the volume of database connections we increased the **maxActive to 150** and **maxIdle to 50** in the database connection settings in **my-applicationContext.xml**

```
<bean id="sharedPoolConfig"
class="org.springframework.beans.factory.config.MapFactoryBean"
abstract="true">
    <property name="sourceMap">
        <map>
            <entry key="defaultAutoCommit" value="true"></entry>
            <entry key="defaultReadOnly" value="false"></entry>
            <entry key="defaultTransactionIsolation"
value="READ_COMMITTED"></entry>
            <entry key="maxActive" value="150"></entry>
            <entry key="maxIdle" value="50"></entry>
            <entry key="minIdle" value="5"></entry>
            <entry key="initialSize" value="0"></entry>
            <entry key="testOnBorrow" value="true"></entry>
            <entry key="testOnReturn" value="false"></entry>
            <entry key="testWhileIdle" value="false"></entry>
            <entry key="minEvictableIdleTimeMillis"
value="300000"></entry>
            <entry key="timeBetweenEvictionRunsMillis"
value="60000"></entry>
            <entry key="removeAbandoned" value="false"></entry>
            <entry key="removeAbandonedTimeout"
value="300"></entry>
            <entry key="logAbandoned" value="false"></entry>
        </map>
    </property>
</bean>
```

Network

For network connections between the codeBeamer ALM server and the database, we recommend using a low latency time connection (less than 0.5 ms) where the two servers are in the same network. During testing, all connections used HTTP ports.

Testing tools

For test planning and execution we used JMeter version 3.3 with Jenkins integration version 2.107.3 (for automatic execution only).

Test data shape and volume

The work items, users, projects, trackers, comments, reference (links), history and other elements were evenly distributed among the projects. The following table represents the number of work items by type. The table below shows that for 10 million work items (i.e. 10 million requirements), there were 30 million comments, and 203 million history entries in the database repository.

Artifact type	10M Work items
Work Items	10,170,913
Named Users	3,196
Projects	45
Trackers	585
Comments	30,512,425
Work Item References	53,012,983
History entries of Work Items	203,402,425

Test cases, scenarios

We used Apache JMeter to simulate the workload and to measure the performance of codeBeamer ALM. One test script was created for all three user actions. In the script, each HTTP request represented a user action. We started with 100/250/500/700/1000 concurrent users respectively, with all users logged in over a period of 15 minutes. Tests were then run for another 45 minutes. User actions during testing were simulated with a 1 minute thinking time for each user.

The following table shows the use cases and the percentage of users running each use case:

Use Case	Description	Number of Users
Browse trackers and items	Login. Visit projects page. Open a project. Go to trackers page. Open a requirements tracker in document view. Open a tasks tracker. Open a bugs tracker. Open a bug, open related Task, open related task requirement, open related system requirement and open related customer requirement. Open members page.	70%
Create work items and comments	Login. Visit projects page. Open a project. Go to trackers page. Create a requirement, add an association, and add 3 comments to it. Create a task, add an association, and add 3 comments to it. Create a bug, add an association, and add 3 comments to it.	20%
Search for items	Login. Enter text in the search field. Scroll to the second page. Open a search result.	10%

Test performance results with up to 10 million work items and 1,000 concurrent users

Stress test with 1,000 concurrent logins

We used a database of **10 million work items** (i.e. requirements, bugs, test cases) to measure the response time for login requests. The login stress test simulates 1,000 concurrent user logins in 300 seconds. See the response times below.

Concurrent Users	Login response time with 10 million work items in the database
1,000	240 ms

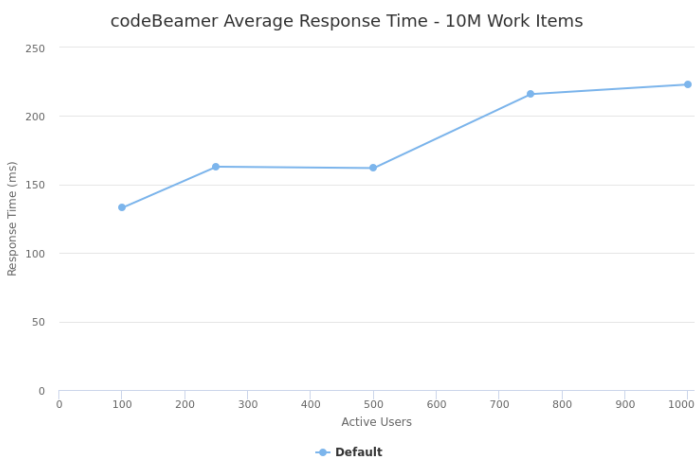
Average response time

The following table summarizes the overall average response time for each user request using different user levels for the tested software / hardware configurations. Tests were carried out with 10 million work items (i.e. requirements, bugs, or test cases) and 100 - 1000 concurrent users.

Concurrent Users	10 million Work Items
100	133 ms
250	163ms
500	162 ms
750	216 ms
1000	233 ms

Response Time Diagram

The diagram shows the average response time vs concurrent users and for repositories with 10 million (10M) Work Items (i.e. requirements).



Detailed results of performance test runs

10M Work Items - 1000 concurrent Users

URI	Samples	Average (ms)	Min(ms)	Median(ms)	Line 90.0(ms)	Max(ms)	Http Code	Errors (%)
Add Association	720 +0	35 0	12 0	21 0	49 0	2115 0	200	0.0 % 0.0 %
Add Attachment	1999 +0	67 0	33 0	49 0	69 0	2307 0	200	0.0 % 0.0 %
Create Item	746 +0	257 0	135 0	247 0	348 0	2568 0	200	0.0 % 0.0 %
Get Requirements Document View Center Panel	368 +0	4901 0	651 0	4800 0	7804 0	11701 0	200	0.0 % 0.0 %
Get Requirements Document View Tree	377 +0	5447 0	838 0	5756 0	8755 0	11440 0	200	0.0 % 0.0 %
Get Trackers Dashboard First Time	2628 +0	17 0	5 0	9 0	13 0	1853 0	200	0.0 % 0.0 %
Get Trackers Tree	2568 +0	126 0	90 0	109 0	164 0	2280 0	200	0.0 % 0.0 %
Get Wiki Tree	2318 +0	89 0	5 0	84 0	121 0	2284 0	200	0.0 % 0.0 %
Login	1000 +0	58 0	44 0	55 0	65 0	315 0	200	0.0 % 0.0 %
Search	4901 +0	76 0	21 0	38 0	176 0	2161 0	200	0.0 % 0.0 %
Set Load Tes Mode	1000 +0	10 0	5 0	8 0	13 0	231 0	200	0.0 % 0.0 %
Show Bugs Tracker Items	2005 +0	282 0	117 0	275 0	389 0	2645 0	200	0.0 % 0.0 %
Show Tasks Tracker Items	2061 +0	338 0	125 0	330 0	475 0	3202 0	200	0.0 % 0.0 %
Visit Add Association Page	733 +0	90 0	44 0	72 0	117 0	1687 0	200	0.0 % 0.0 %
Visit Add Attachment Page	2039 +0	38 0	17 0	26 0	37 0	2332 0	200	0.0 % 0.0 %
Visit Bugs Tracker Page	2020 +0	121 0	29 0	113 0	159 0	2685 0	200	0.0 % 0.0 %
Visit Item Create Page	773 +0	68 0	40 0	56 0	76 0	2826 0	200	0.0 % 0.0 %
Visit Login Page	1000 +0	13 0	7 0	11 0	26 0	195 0	200	0.0 % 0.0 %
Visit Members Page	1730 +0	309 0	207 0	286 0	413 0	2960 0	200	0.0 % 0.0 %
Visit Project Page	400 +0	45 0	26 0	37 0	47 0	1405 0	200	0.0 % 0.0 %
Visit Projects Page	2792 +0	38 0	19 0	26 0	52 0	2446 0	200	0.0 % 0.0 %
Visit Requirements Tracker Page	387 +0	149 0	47 0	133 0	176 0	2179 0	200	0.0 % 0.0 %
Visit Tasks Tracker Page	2106 +0	117 0	29 0	112 0	153 0	2660 0	200	0.0 % 0.0 %
Visit Tracker Page	799 +0	142 0	39 0	124 0	167 0	2142 0	200	0.0 % 0.0 %
Visit Trackers Page	2675 +0	42 0	22 0	31 0	42 0	2611 0	200	0.0 % 0.0 %
Visit Wiki Page	2355 +0	48 0	28 0	38 0	51 0	1686 0	200	0.0 % 0.0 %
Visit a Bug	1960 +0	226 0	114 0	204 0	325 0	2766 0	200	0.0 % 0.0 %
Visit a Customer Requirement	1775 +0	286 0	182 0	256 0	397 0	3136 0	200	0.0 % 0.0 %
Visit a System Requirement	1817 +0	300 0	166 0	266 0	409 0	2918 0	200	0.0 % 0.0 %
Visit a Task	1912 +0	466 0	180 0	458 0	631 0	2894 0	200	0.0 % 0.0 %
Visit a Task Requirement	1865 +0	511 0	185 0	489 0	751 0	3452 0	200	0.0 % 0.0 %
All URIs	**51829 +0**	**223 0**	**5 0**	**95 0**	**398 0**	**11701 0**		**0.0 % 0.0 %**

Performance test with 35,000 requirements in a single document

Most requirements documents only have a few hundred or a few thousand requirements. In working with our customers, however, we often encounter over 30,000 requirements in a single document. The test scenario below shows our performance test for such documents.

ReqIF file import performance with 35,463 requirements

We use a ReqIF file (55 MB) with 35,463 requirements for the import performance test. The ReqIF file import takes 25 minutes.

ReqIF file import with 35K requirements	25 min

Requirement Edit/View performance with 10,000 and 35,000 requirements in a single document

This performance measurement shows you how much time it takes to open a requirements document (one document or one module) in codeBeamer's "Document View" mode using a web browser. In the two test scenarios, the requirements documents have 10,000 and 35,000 requirements respectively. As an example, a modern car's entertainment system software might have 35,000 requirements. Google Chrome 59.0.3071.86 browser was used in our test.

Open 10,000 requirements from the web browser in Document View mode	~7 sec
Open 35,000 requirements from the web browser in Document View mode	~15 sec

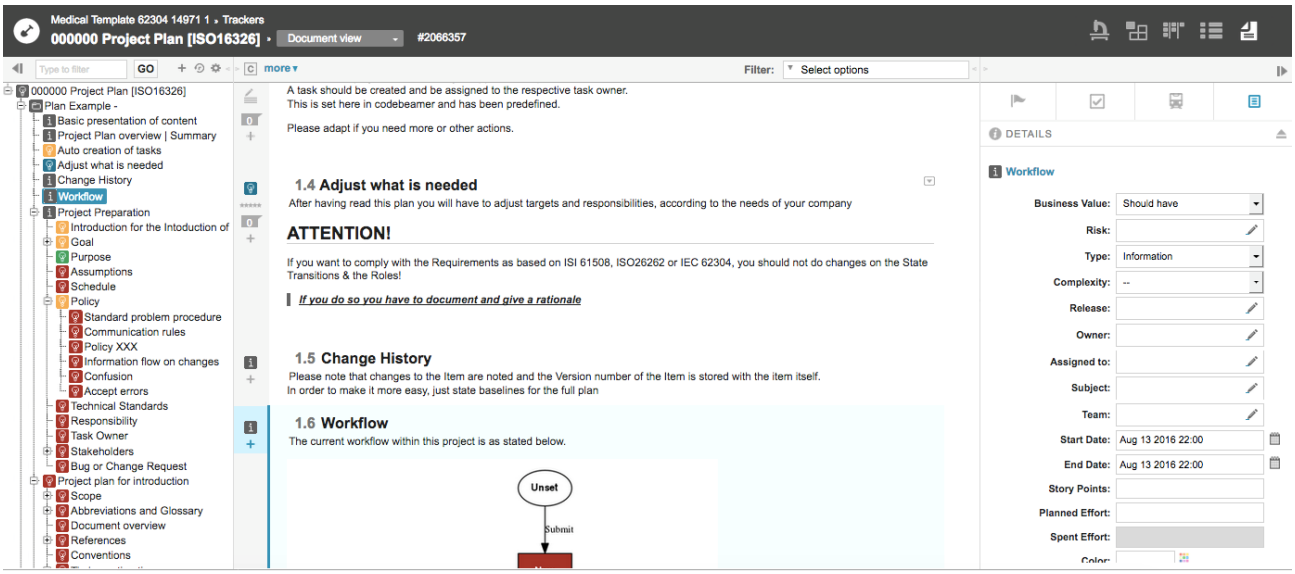


Figure: "Document View" mode in web browser