Liberator 7.1

Benchmarks

Revision 1.1.0, July 2023

CAPLIN

Liberator 7.1: Benchmarks

Revision 1.1.0, July 2023

Branch main, commit 2ba3ac6 (2023-07-15 22:37:59 UTC)

Built on 2023-07-16 16:30:31 +0100

Version	Date	Notes
1.0.0	February 2019	First release
1.1.0	July 2023	Minor reprint including new graphs and results tables

Copyright © 2023 Caplin Systems Ltd. All rights reserved.

Caplin Systems Ltd, 107 Leadenhall Street, London EC3A 4AF, UK

The information contained in this publication is subject to UK, US and international copyright laws and treaties and all rights are reserved. No part of this publication may be reproduced or transmitted in any form or by any means without the written authorization of an Officer of Caplin Systems Limited.

Various Caplin technologies described in this document are the subject of patent applications. All trademarks, company names, logos and service marks/names ('Marks') displayed in this publication are the property of Caplin or other third parties and may be registered trademarks. You are not permitted to use any Mark without the prior written consent of Caplin or the owner of that Mark.

This publication is provided 'as is' without warranty of any kind, either express or implied, including, but not limited to, warranties of merchantability, fitness for a particular purpose, or non-infringement.

This publication could include technical inaccuracies or typographical errors and is subject to change without notice. Changes are periodically added to the information herein; these changes will be incorporated in new editions of this publication. Caplin Systems Limited may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time.

This publication may contain links to third-party web sites; Caplin Systems Limited is not responsible for the content of such sites.

Table of Contents

© Caplin Systems Limited 2023

1. Preface

This document details the results of a set of performance benchmark tests carried out on Caplin Liberator 7.1. The information provided in this report is intended to assist customers in production capacity planning when deploying Liberator 7.1.

1.1. What this document contains

Chapter overview:

- Chapter 2, *Overview* provides an overview of the benchmark tests and presents a selection of headline results.
- Chapter 3, *Test scenarios* describes the test scenarios in detail.
- Chapter 4, *Test results* contains the results, with performance graphs, of the standard tests.
- Chapter 5, *How the benchmark tests were conducted* gives detailed information on how the benchmark tests were conducted.
- Chapter 6, *Frequently asked questions* addresses frequently asked questions about how to configure and tune Liberator and its environment to achieve the required performance.

1.2. Who should read this document

This document is intended for anyone who is evaluating Caplin Liberator's performance characteristics, or who is planning to deploy Caplin Liberator.

Typical readers would be:

- Technical Managers
- System Architects
- System Administrators

© Caplin Systems Limited 2023

1.3. Feedback

Customer feedback can only improve the quality of our product documentation, and we would welcome any comments, criticisms or suggestions you may have regarding this document.

Please email your feedback to documentation@caplin.com.

1.4. Acknowledgements

Adobe Reader is a registered trademark of Adobe Systems Incorporated in the US and/or other countries.

Windows is a registered trademark of Microsoft Corporation in the US and other countries.

Sun, *Solaris* and *Java*, are trademarks or registered trademarks of Oracle Corporation, in the US or other countries.

Linux® is the registered trademark of Linus Torvalds in the US and other countries.

© Caplin Systems Limited 2023

2. Overview

The benchmark tests detailed in this document are designed to show how Caplin Liberator will perform on the Linux® platform, when deployed as a real-time financial internet hub, streaming data updates to web-based financial trading applications.

2.1. About the benchmark tests

The tests cover four scenarios:

Low Update Rate scenario (1 update/user/second)

Representative of a low-end information portal.

Medium Update Rate scenario (10 updates/user/second)

Representative of the workload when each user streams updates for two different on-screen instruments, with each instrument updating at 4 updates/second.

High Update Rate scenario (50 updates/user/second)

Representative of the workload when each user streams updates for 12 different on-screen instruments, with each instrument updating at 4 updates/second.

Very High Update Rate scenario (100 updates/user/second)

Representative of the workload when each user streams updates for 25 different on-screen instruments, with each instrument streaming at 4 updates/second.

The main factor affecting the overall performance of Liberator is the power of the machine on which it runs. The tests were conducted on servers representing typical commercially available machines that can be used to host web servers and server applications. A single Liberator instance was run on one machine, while test harnesses were run on other machines to provide data and client processes.

2.2. Message latency versus CPU usage

The key item measured in the tests was the end-to-end message latency against the number of logged in clients, and by implication the number of update messages being sent out to the totality of the connected client base.

Although some of the test results show CPU usage, in practice end-to-end message latency

© Caplin Systems Limited 2023

is more significant as a measure of Liberator performance than CPU usage. Message latency has a direct impact on users and may increase long before CPU usage reaches its maximum. The aim of sizing a system incorporating Caplin Liberator should be to achieve a maximum desired message latency for a given maximum update rate.

2.3. Test setup

For detailed information on the test set up used to conduct the benchmarks, see Chapter 5, *How the benchmark tests were conducted*.

While the benchmarks described in this document were designed to emulate real-world traffic and user scenarios, they were conducted using specific hardware running in an isolated environment, and therefore no guarantees can be made that identical results will be achieved with different hardware in other environments.

© Caplin Systems Limited 2023

2.4. Headline results

The tables below provide an easy comparison of mean latency for different numbers of users in four different workload scenarios.

Mean latency (milliseconds), 250–1,000 users

Users	High Update Rate	Very High Update Rate
250	0.524	1.089
500	0.648	1.581
750	0.704	1.557
1,000	0.754	1.657

Mean latency (milliseconds), 1,000–5,000 users

Users	Low Update Rate	Medium Update Rate	High Update Rate	Very High Update Rate
1,000	0.33	0.448	0.754	1.657
2,000	0.355	0.526	0.954	2.28
3,000	0.394	0.612	1.034	2.324
4,000	0.411	0.618	1.478	2.266
5,000	0.423	0.653	1.924	2.675

For full results, see the sections below:

- Section 4.2, “Low Update Rate results”
- Section 4.3, “Medium Update Rate results”
- Section 4.4, “High Update Rate results”
- Section 4.6, “Very High Update Rate results”

© Caplin Systems Limited 2023

2.5. Caplin's benchmark tools

Benchmarking a streaming server such as Liberator in a realistic manner is a challenge, because of the need to simulate the large numbers of users and high update rates that would be encountered in real-world business environments.

Caplin Systems has produced an internal suite of tools, the Benchtools, that make benchmark scenarios easier to set up and run. The Benchtools suite was used to run the benchmark tests described in this document.

The Benchtools suite comprises two components:

Benchsrc

An update publisher, which generates updates of a configurable size and frequency.

Benchrttp

A scalable client simulator, which creates real RTTP sessions, but simulates clients being spread across many computers and browsers.

For an illustration of how Benchsrc, Liberator, and Benchrttp were deployed to run these benchmarks, see Section 5.2, "Test setup".

© Caplin Systems Limited 2023

3. Test scenarios

Liberator was tested against a range of scenarios designed to simulate the different types of activity and data rates commonly seen in real-time financial applications. They demonstrate the performance that Liberator can achieve.

3.1. Low Update Rate scenario

The Low Update Rate scenario has a throughput of 1 message/second for each client. This message throughput is representative of a low-end information portal.

This scenario was tested with message batching disabled and one message size (54 bytes).

Low Update Rate scenario specification

Source subjects	1,000
Update rate per subject	0.5 updates/second
Total source update rate	500 updates/second
Subscriptions per client	2
Message size (bytes)	54
Update rate per client	1 update/second
Number of DataSources	1
`burst-min` **(seconds)**	0
`burst-increment` **(seconds)**	0.01
`burst-max` **(seconds)**	0 (message batching disabled)
Liberator session threads	8

© Caplin Systems Limited 2023

3.2. Medium Update Rate scenario

The Medium Update Rate scenario has a throughput of 10 messages/second for each client. This message throughput is representative of two different instruments updating four times a second. For example, a mobile app streaming an executable RFS price (4 messages/second) or a watchlist of two instruments (8 messages/second) would fall within the bounds of this scenario.

Compared to the Low Updates scenario, this scenario has an increased update rate, increased number of subjects, and more subscriptions from each client.

This scenario was tested with message batching disabled and one message size (54 bytes).

Medium Update Rate scenario specification

Source subjects	4,000
Update rate per subject	0.5 updates/second
Total source update rate	2,000 updates/second
Subscriptions per client	20
Message size (bytes)	54
Update rate per client	10 updates/second
Number of DataSources	2
`burst-min` **(seconds)**	0
`burst-increment` **(seconds)**	0.01
`burst-max` **(seconds)**	0 (message batching disabled)
Liberator session threads	8

© Caplin Systems Limited 2023

3.3. High Update Rate scenario

The High Update Rate scenario has a throughput of 50 messages/second for each client. This message throughput is representative of 12 different instruments updating four times a second. For example, a web application displaying 12 FX tiles (48 messages/second) would fall within the bounds of this scenario.

This scenario was tested with four message batching intervals (0ms, 25ms, 50ms, and 100ms) and four message sizes (54 bytes, 108 bytes, 162 bytes, and 216 bytes).

High Update Rate scenario specification

Source subjects	10,000
Update rate per subject	0.5 updates/second
Total source update rate	5,000 updates/second
Subscriptions per client	100
Message sizes (bytes)	54, 108, 162, and 216
Update rate per client	50 updates/second
Number of DataSources	2
`burst-min` **(seconds)**	0
`burst-increment` **(seconds)**	0.01
`burst-max` **(seconds)**	0, 0.025, 0.05, and 0.1
Liberator session threads	8

© Caplin Systems Limited 2023

3.4. Very High Update Rate scenario

The Very High Update Rate scenario has a throughput of 100 messages/second for each client. This message throughput is representative of 25 different instruments updating four times a second. For example, a web application displaying 12 FX tiles (48 messages/second) and a watchlist of 12 currency pairs (48 messages/second) would fall within the bounds of this scenario.

This scenario is representative of a very high-end single dealer platform, where each client has a large number of fast moving instruments on their screen. Compared to the other scenarios, it has a much larger data set, a higher update rate per object, and a higher number of subscriptions per client. It represents the most extreme online trading requirements.

This scenario was tested with four message batching intervals (0ms, 25ms, 50ms, and 100ms) and one message size (54 bytes).

Very High Update Rate scenario specification

Source subjects	20,000
Update rate per subject	1 updates/second
Total source update rate	20,000 updates/second
Subscriptions per client	100
Message sizes (bytes)	54
Update rate per client	100 updates/second
Number of DataSources	2
`burst-min` **(seconds)**	0
`burst-increment` **(seconds)**	0.01
`burst-max` **(seconds)**	0, 0.025, 0.05, and 0.1
Liberator session threads	8

© Caplin Systems Limited 2023

3.5. Summary

The table below compares the benchmark scenarios side-by-side.

Test scenarios compared

	Low	Medium	High	V. High
Source subjects	1,000	4,000	10,000	20,000
Updates per subject per second	0.5	0.5	0.5	1
Total updates per second	500	2,000	5,000	20,000
Subscriptions per client	2	20	100	100
Message sizes (bytes)	54	54	54 108 162 216	54
Updates per client per second	1	10	50	100
Number of DataSources	1	1	1	2
`burst-min` **(seconds)**	0	0	0	0
`burst-increment` **(seconds)**	0.01	0.01	0.01	0.01
`burst-max` **(seconds)**	0	0	0 0.025 0.05 0.1	0 0.025 0.05 0.1
Liberator session threads	8	8	8	8

© Caplin Systems Limited 2023

4. Test results

This section reviews the test results for Liberator 7.1 running on CentOS 7.4.1708.

For more information on the scenarios tested in this benchmark, see Chapter 3, *Test scenarios*.

For more information on the hardware used in this benchmark, see Chapter 5, *How the benchmark tests were conducted*.

4.1. Interpreting the graphs

Each scenario includes two main graphs:

Mean latency

A plot of mean latency with batching disabled and a message size of 54 bytes. Each point on the graph represents the mean of all the messages received in a 30-second period.

Latency range and CPU usage

The blue plot is the latency range (the maximum and minimum latency received in that period). The white line embedded in the latency range shows the mean latency, and the darker line shows the CPU usage of the Liberator throughout the test.

Mean latency at different batching intervals

The High Update Rate and Very High Update Rate scenarios include an extra graph that measures the effect on mean latency of different batch intervals.

Mean latency at different message sizes

The High Update Rate scenario includes an extra graph comparing the effect on mean latency of different message sizes.

© Caplin Systems Limited 2023

4.2. Low Update Rate results

Throughput: 1 message *per* second *per* client.

Full dataset: results-low.pdf

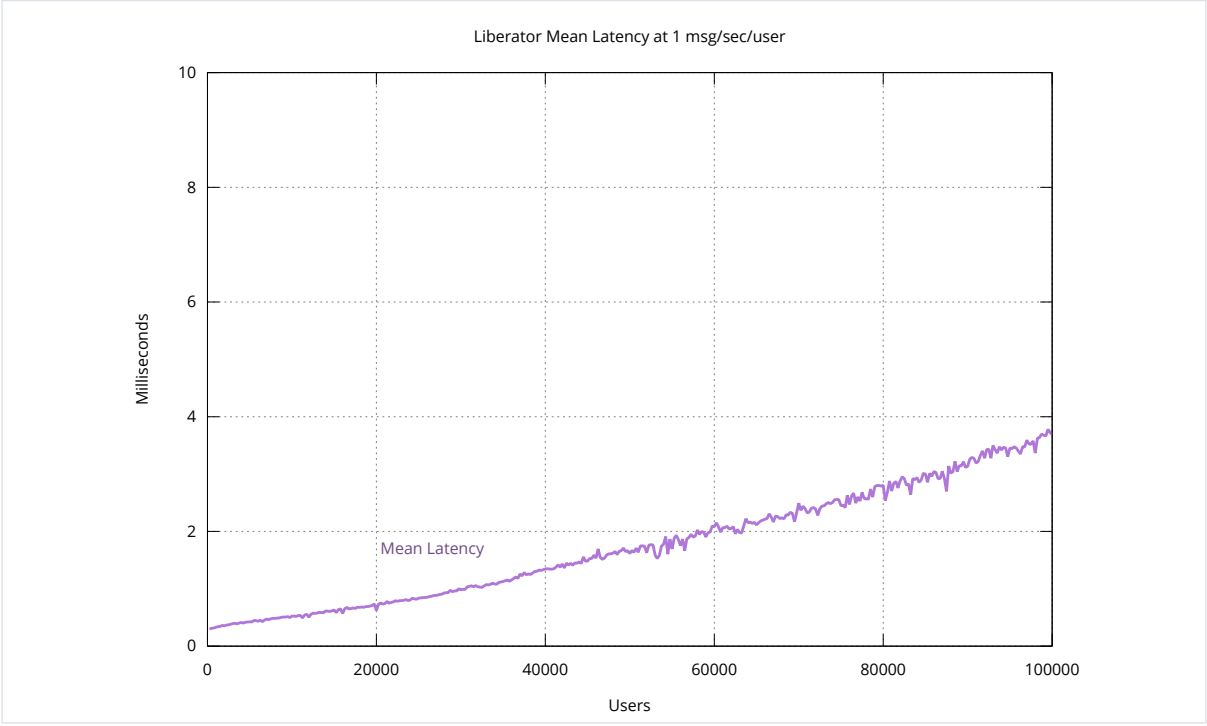

Figure 1. Low Update Rate: mean latency (250–100,000 users)

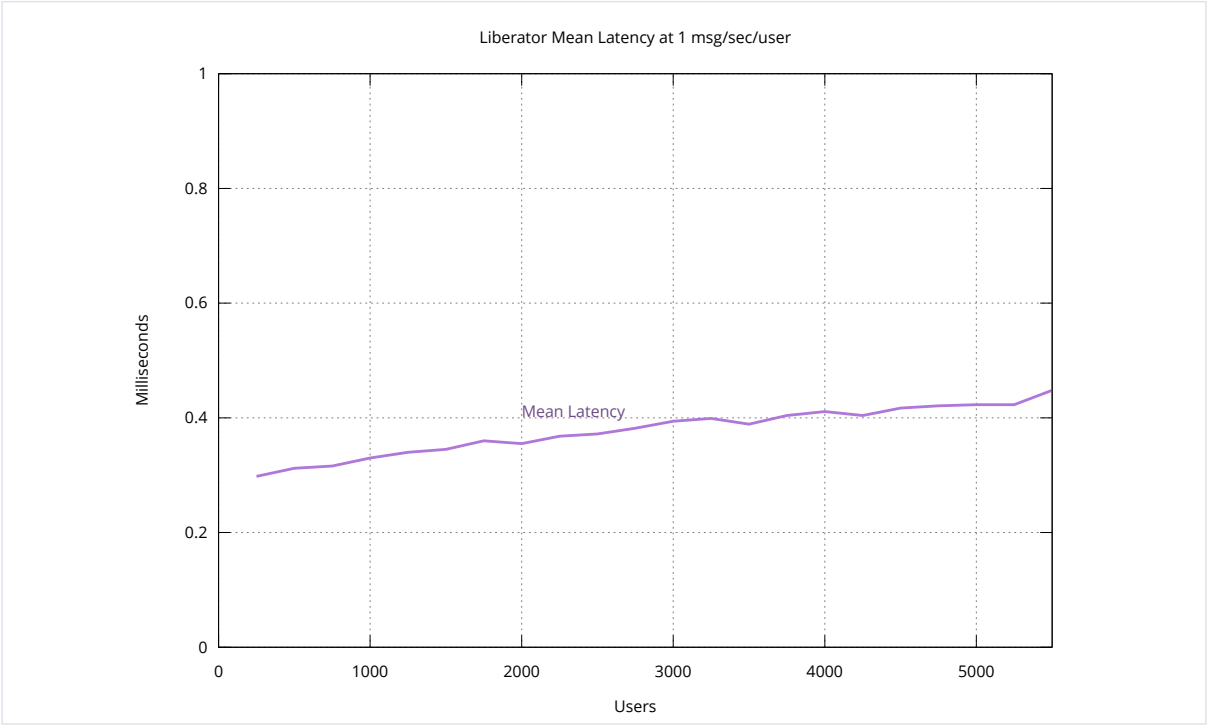

Figure 2. Low Update Rate: mean latency (250–5,500 users)

© Caplin Systems Limited 2023

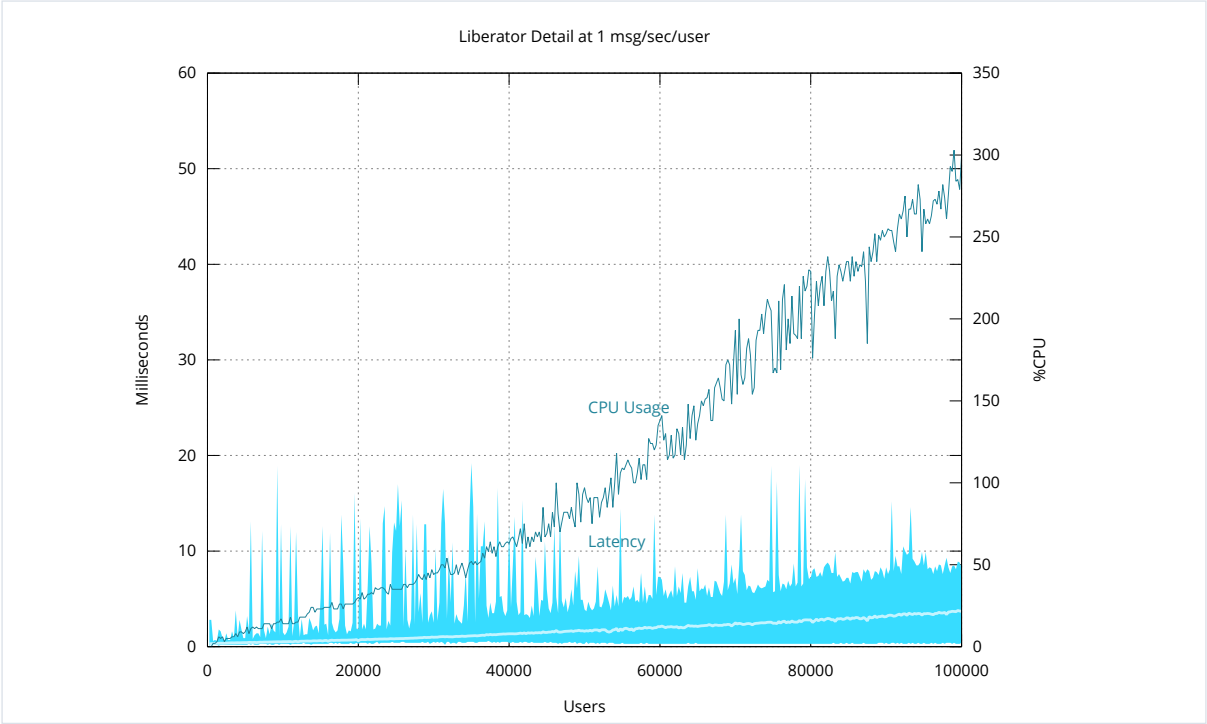

Figure 3. Low Update Rate: latency range and CPU usage (250–100,000 users)

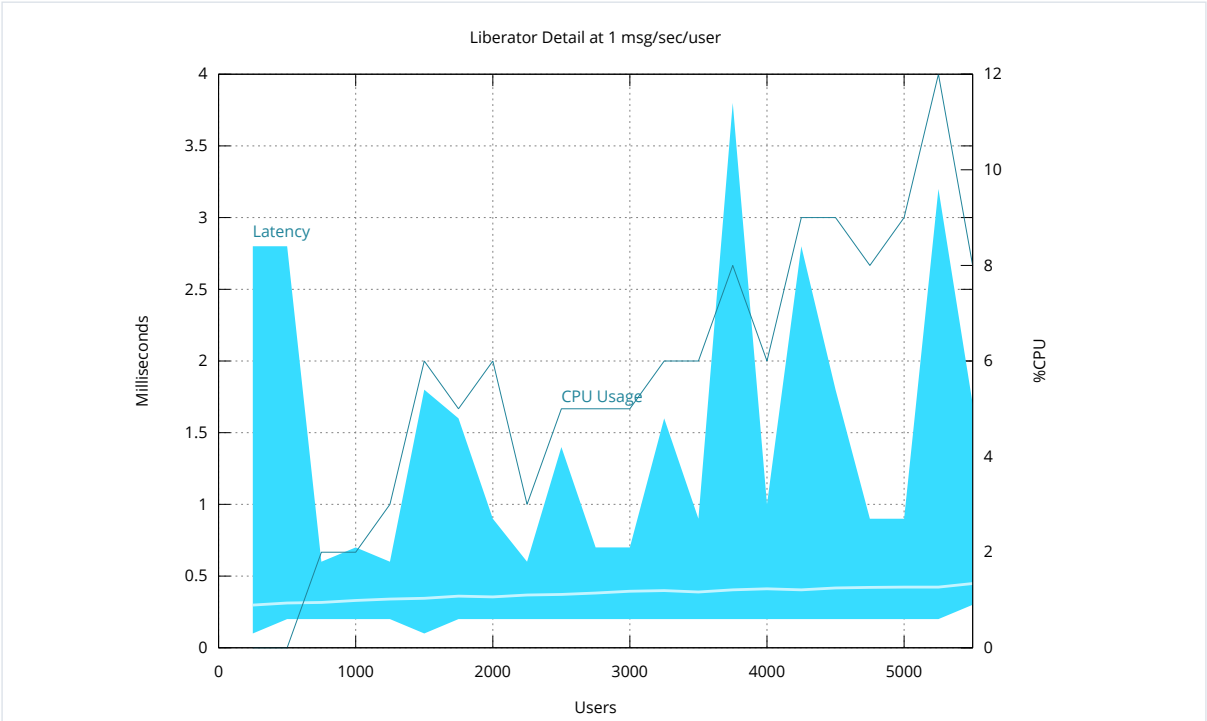

Figure 4. Low Update Rate: latency range and CPU usage (250–5,500 users)

© Caplin Systems Limited 2023

The Low Update Rate scenario has a throughput of 1 message/second for each client (user). This message throughput is representative of a low-end information portal.

Low Update Rate (250–100,000 users), abridged

Users	Mean Lat. (ms)	Min Lat. (ms)	Max Lat. (ms)	CPU (%)
250	0.298	0.1	2.8	0
500	0.312	0.2	2.8	0
1,000	0.33	0.2	0.7	2
5,000	0.423	0.2	0.9	9
10,000	0.523	0.3	1.5	14
25,000	0.836	0.4	12.1	35
50,000	1.616	0.3	3.9	97
75,000	2.448	0.3	5.7	167
100,000	3.688	0.4	8.2	300

© Caplin Systems Limited 2023

4.3. Medium Update Rate results

Throughput: 10 messages *per* second *per* client.

Full dataset: [results-medium]

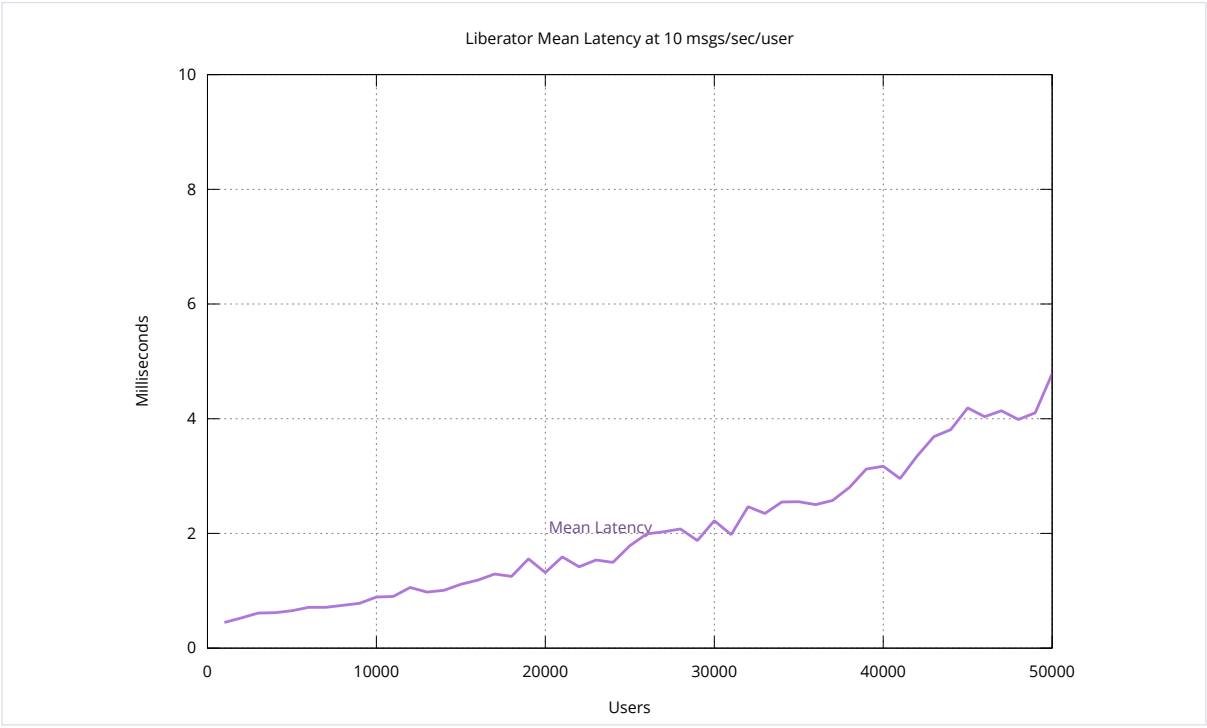

Figure 5. Medium Update Rate: mean latency (1,000–50,000 users)

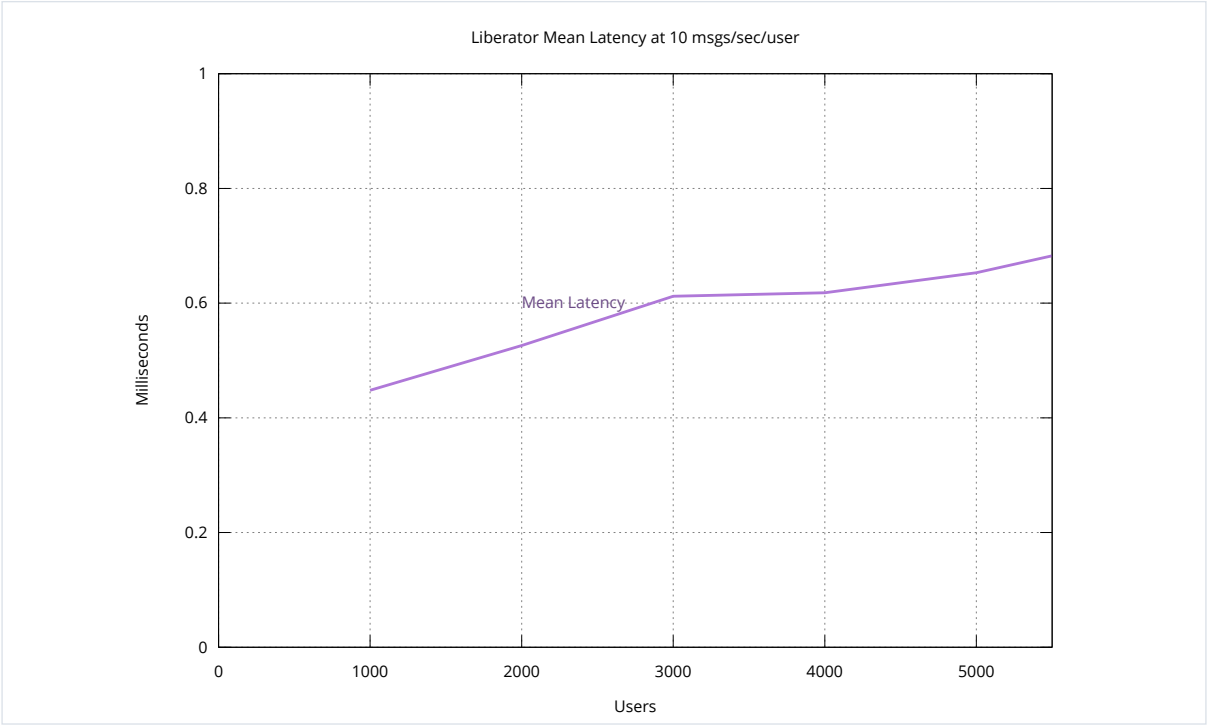

Figure 6. Medium Update Rate: mean latency (1,000–5,500 users)

© Caplin Systems Limited 2023

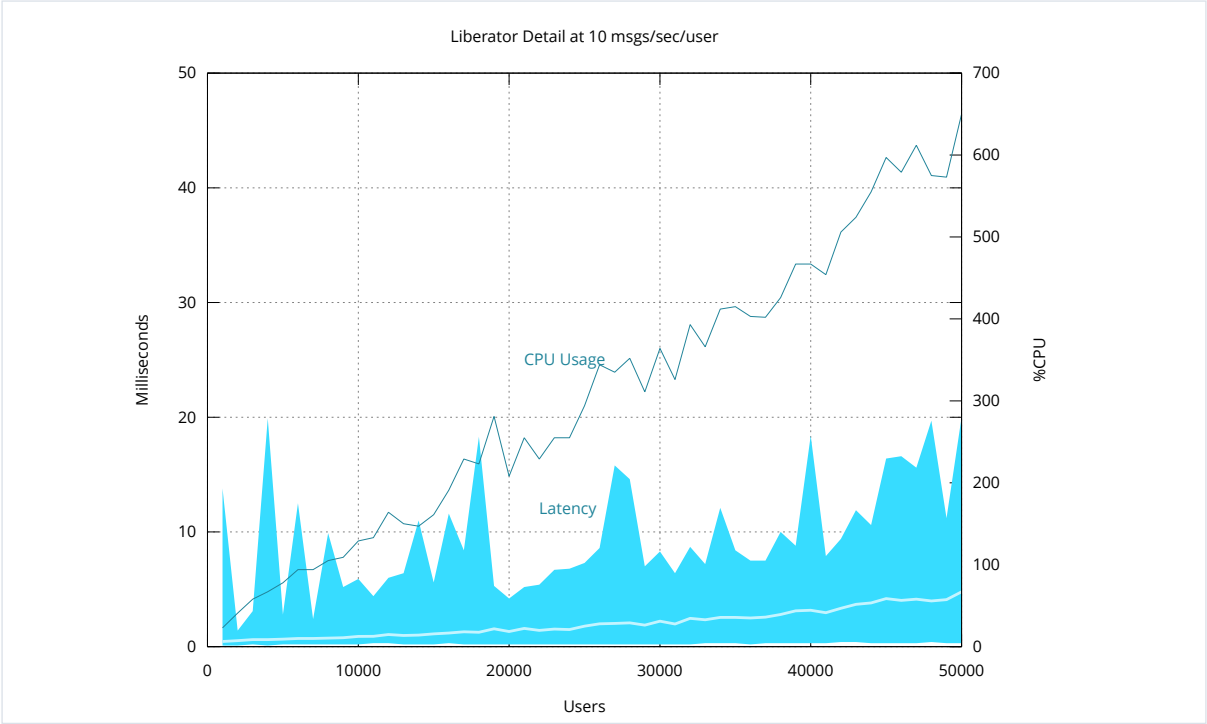

Figure 7. Medium Update Rate: latency range and CPU usage (1,000–50,000 users)

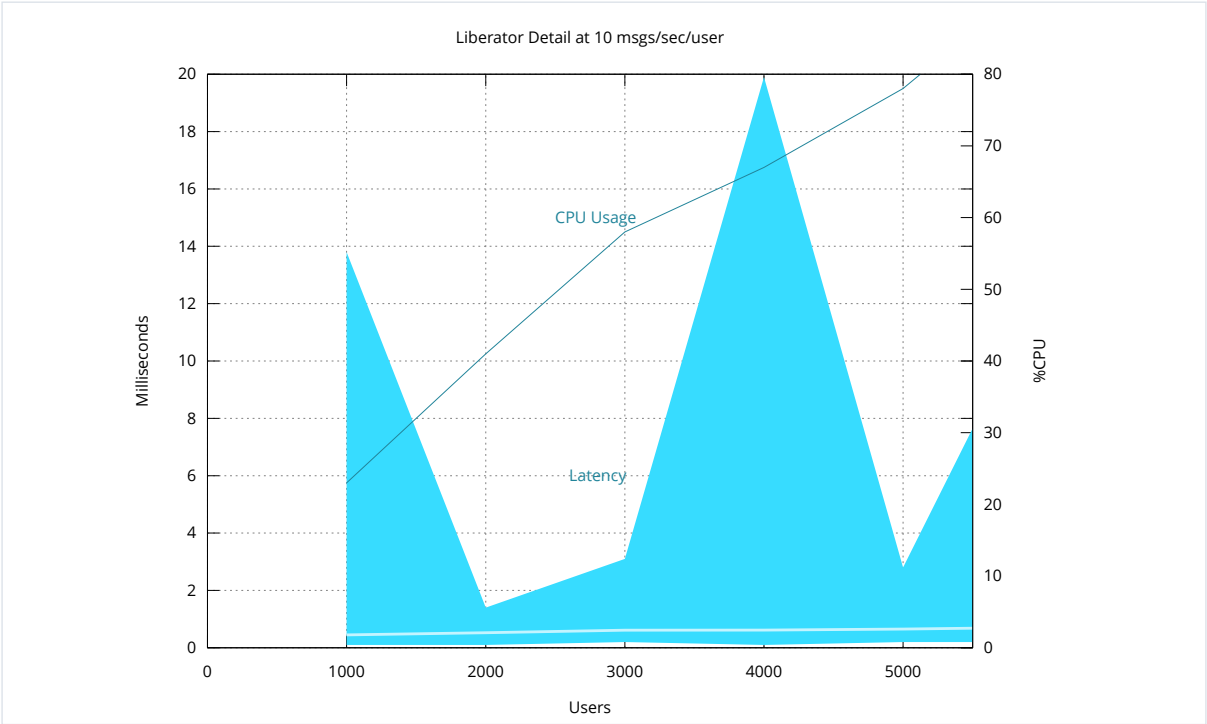

Figure 8. Medium Update Rate: latency range and CPU usage (1,000–5,500 users)

© Caplin Systems Limited 2023

The Medium Update Rate scenario has a throughput of 10 messages/second for each client (user). This throughput is representative of the workload when each user streams updates for two different on-screen instruments, with each instrument updating at 4 updates/second. For example, a mobile app streaming an executable RFS price (4 messages/second) would fall within the bounds of this scenario.

Medium Update Rate: 1,000–50,000 users, abridged

Users	Mean Lat. (ms)	Min Lat. (ms)	Max Lat. (ms)	CPU (%)
1,000	0.448	0.1	13.8	23
2,000	0.526	0.1	1.4	41
3,000	0.612	0.2	3.1	58
4,000	0.618	0.1	19.9	67
5,000	0.653	0.2	2.8	78
10,000	0.891	0.2	5.9	129
20,000	1.319	0.2	4.2	208
30,000	2.222	0.2	8.3	364
40,000	3.172	0.3	18.4	467
50,000	4.783	0.3	20	650

© Caplin Systems Limited 2023

4.4. High Update Rate results

Throughput: 50 messages *per* second *per* client.

Full dataset: [results-high]

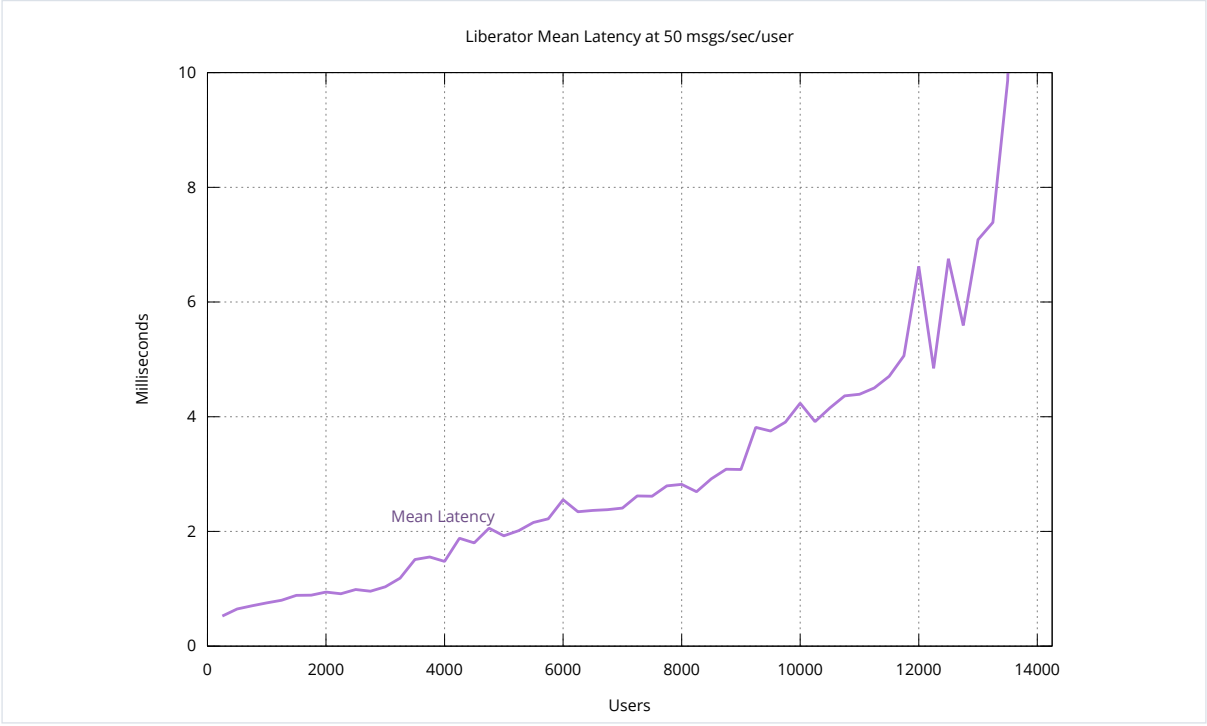

Figure 9. High Update Rate: mean latency (250–14,000 users)

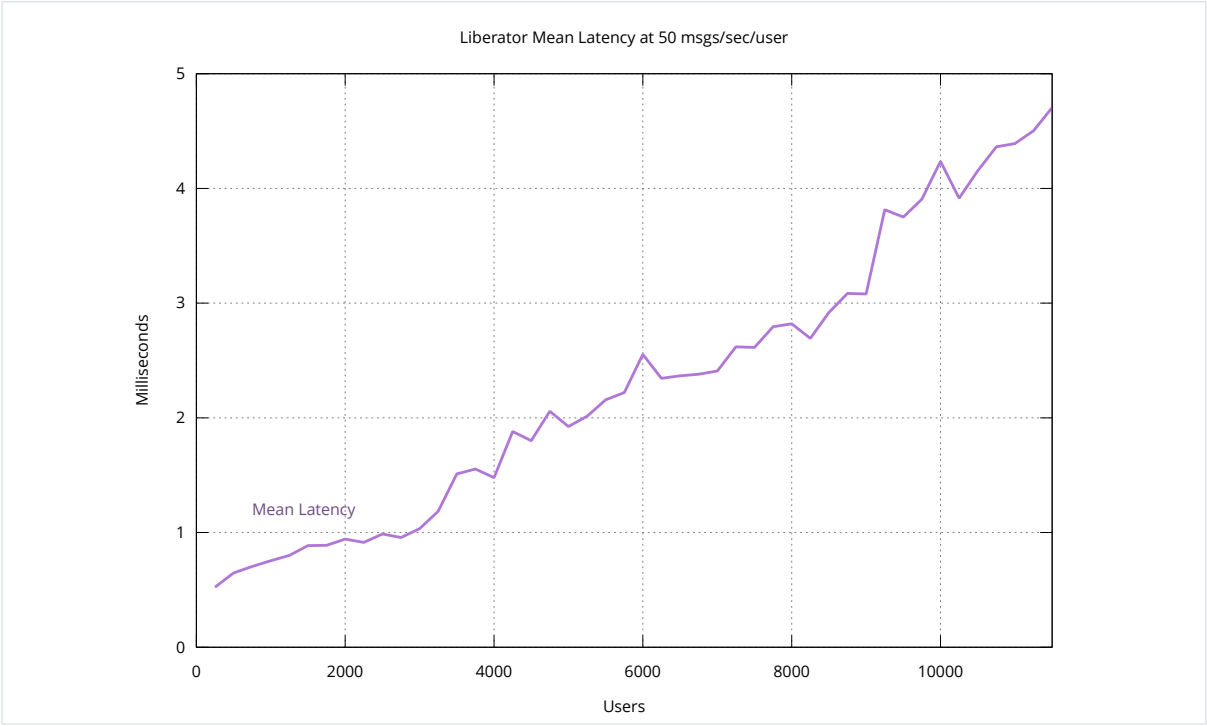

Figure 10. High Update Rate: mean latency (250–2,000 users)

© Caplin Systems Limited 2023

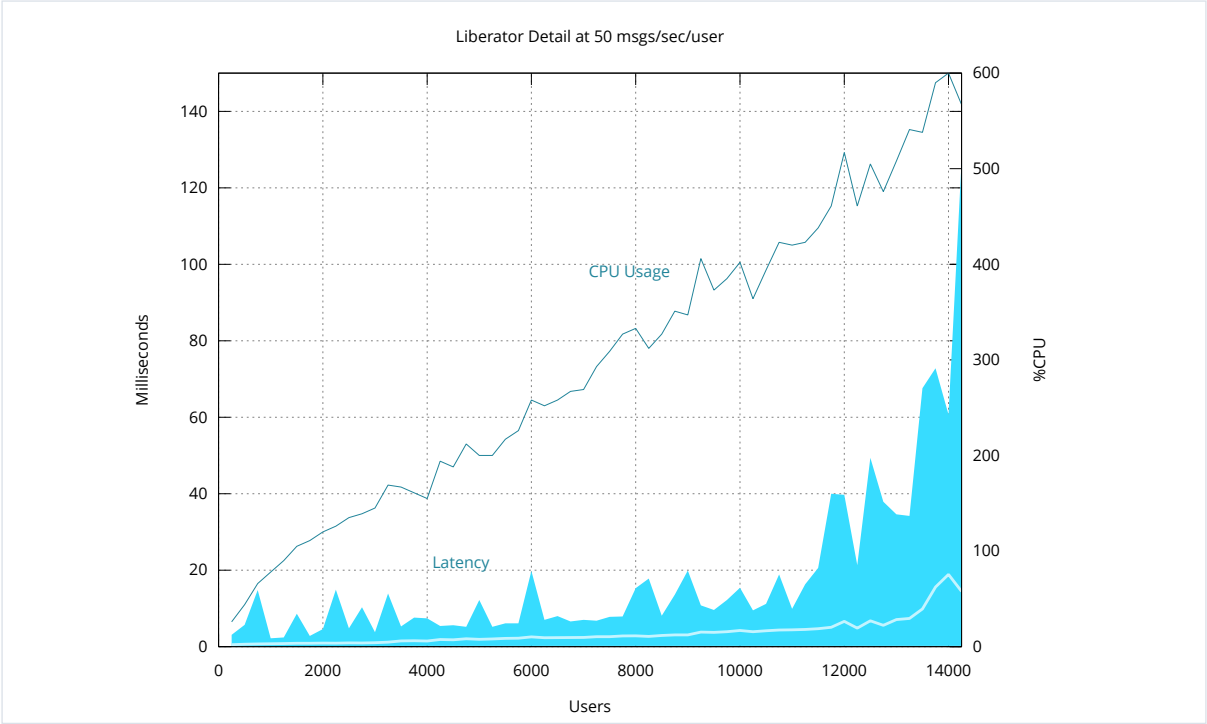

Figure 11. High Update Rate: latency range and CPU usage (250–14,000 users)

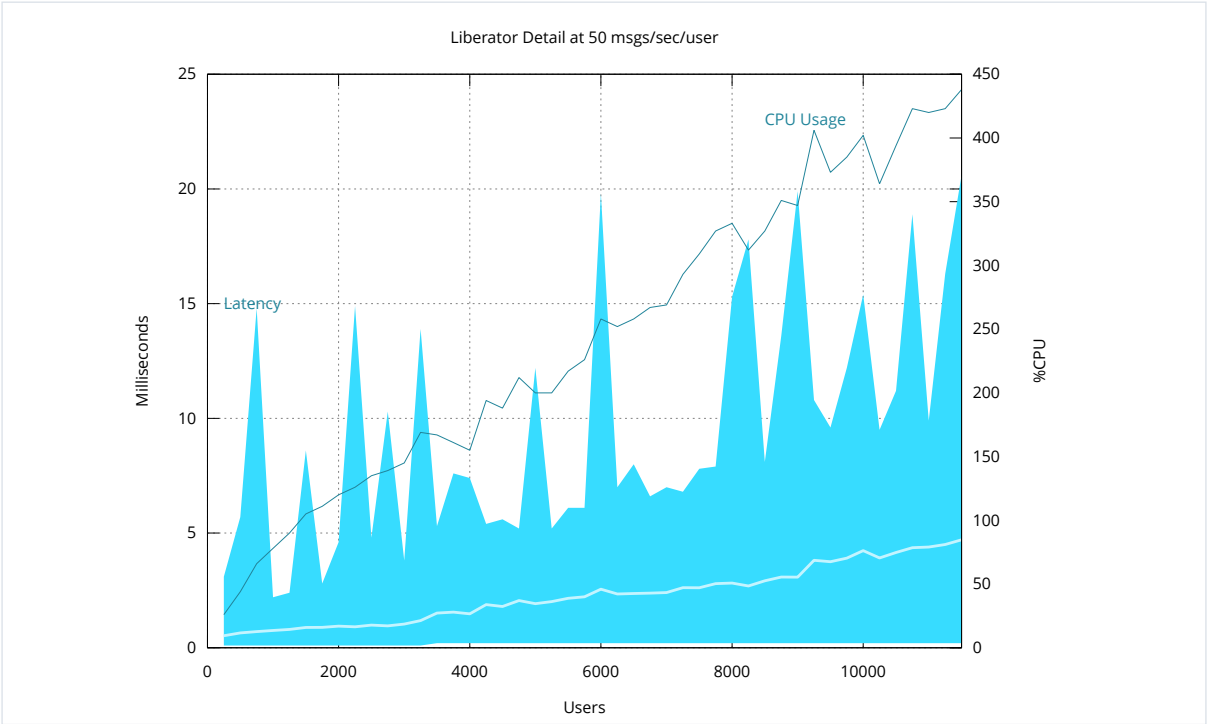

Figure 12. High Update Rate: latency range and CPU usage (250–2,000 users)

© Caplin Systems Limited 2023

The High Update Rate scenario has a throughput of 50 messages/second for each client (user). This throughput is representative of the workload when each user streams updates for 12 different on-screen instruments, with each instrument updating at 4 updates/second. For example, a web application displaying 12 FX tiles (48 messages/second) would fall within the bounds of this scenario.

High Update Rate: batching disabled, 250–10,000 users, abridged

Users	Mean Lat. (ms)	Min Lat. (ms)	Max Lat. (ms)	CPU (%)
250	0.524	0.1	3.1	26
500	0.648	0.1	5.7	44
750	0.704	0.1	14.8	66
1,000	0.754	0.1	2.2	78
1,250	0.801	0.1	2.4	90
1,500	0.886	0.1	8.6	105
1,750	0.889	0.1	2.8	111
2,000	0.943	0.1	4.6	120
2,250	0.914	0.1	14.9	126
2,500	0.987	0.1	4.8	135
5,000	1.924	0.2	12.2	200
7,750	2.794	0.2	7.9	327
10,000	4.236	0.2	15.4	402

© Caplin Systems Limited 2023

4.5. High Update Rate results (batching enabled)

Liberator can be configured to batch messages together at the point they are sent to the client. This does not affect the data being sent, just how it is sent. Instead of sending a lot of small packets containing one message each, Liberator sends larger packets, less frequently, containing multiple messages.

The batching feature will only batch messages together when the update rate is over a configured amount, therefore it can be used to handle peak data rates. However, the scenarios tested here have a more uniform update rate so the batching is always active, when configured.

The following graph shows the mean latency of multiple test runs with different configurations for batching.

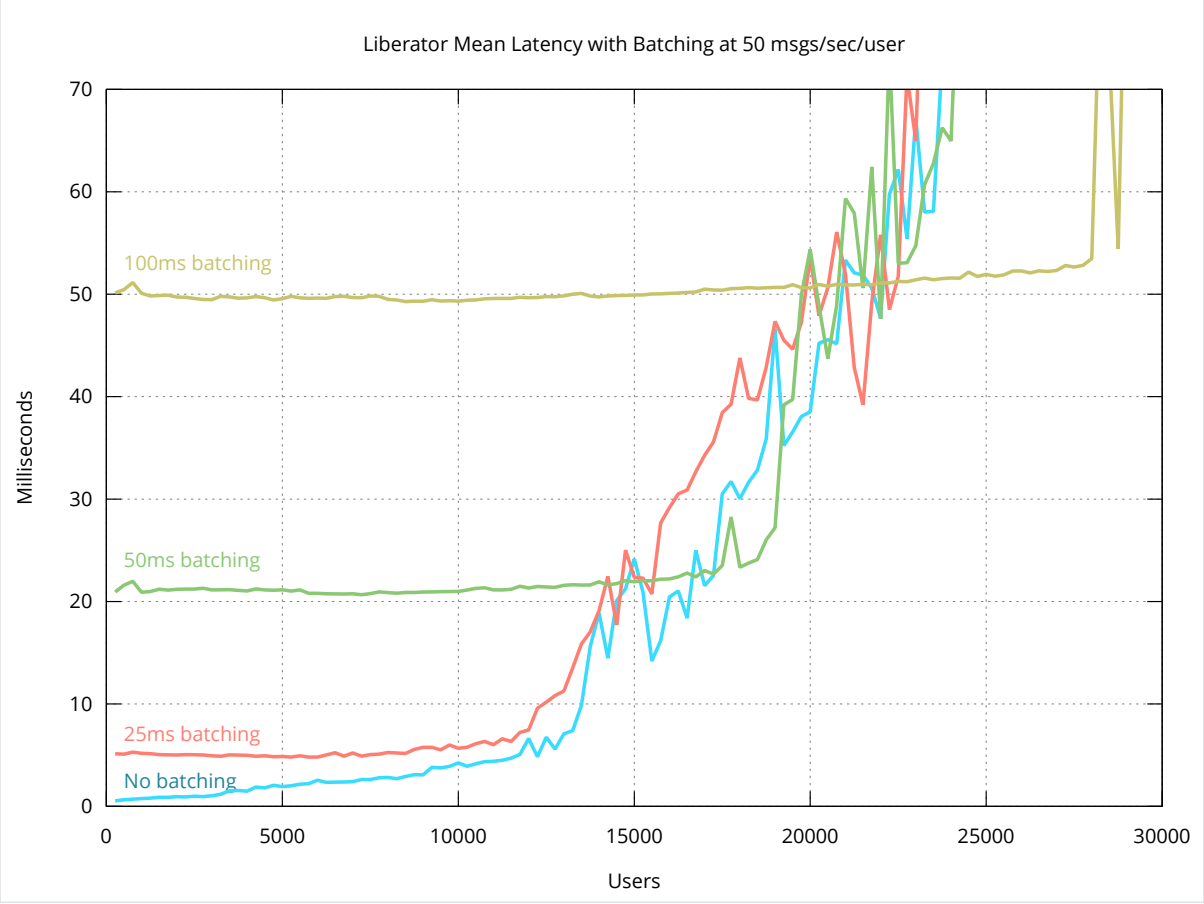

Figure 13. High Update Rate: batch intervals 0ms, 25ms, 50ms, and 100ms

The blue line shows the mean latency with no batching configured; this is the same run as the graphs above. The other three lines show how batching messages over intervals (25ms, 50ms, and 100ms respectively) allows Liberator to support many more clients at the expense of increasing latency.

© Caplin Systems Limited 2023

With a 100 milliseconds batching interval (`burstmax 0.1`), Liberator served 28,000 clients with a mean latency of 53 milliseconds.

The benefits of batching are more apparent at intervals of 100ms than they are at 25ms. In the High Updates scenario, each client received 50 updates per second. At 100ms batching (10 batches sent per second), each batch contains an average of 5 messages. At 25ms batching (40 batches sent per second), each batch contains an average of 1 message.

© Caplin Systems Limited 2023

4.6. Very High Update Rate results

Throughput: 100 messages *per* second *per* client.

Full dataset: [results-very-high]

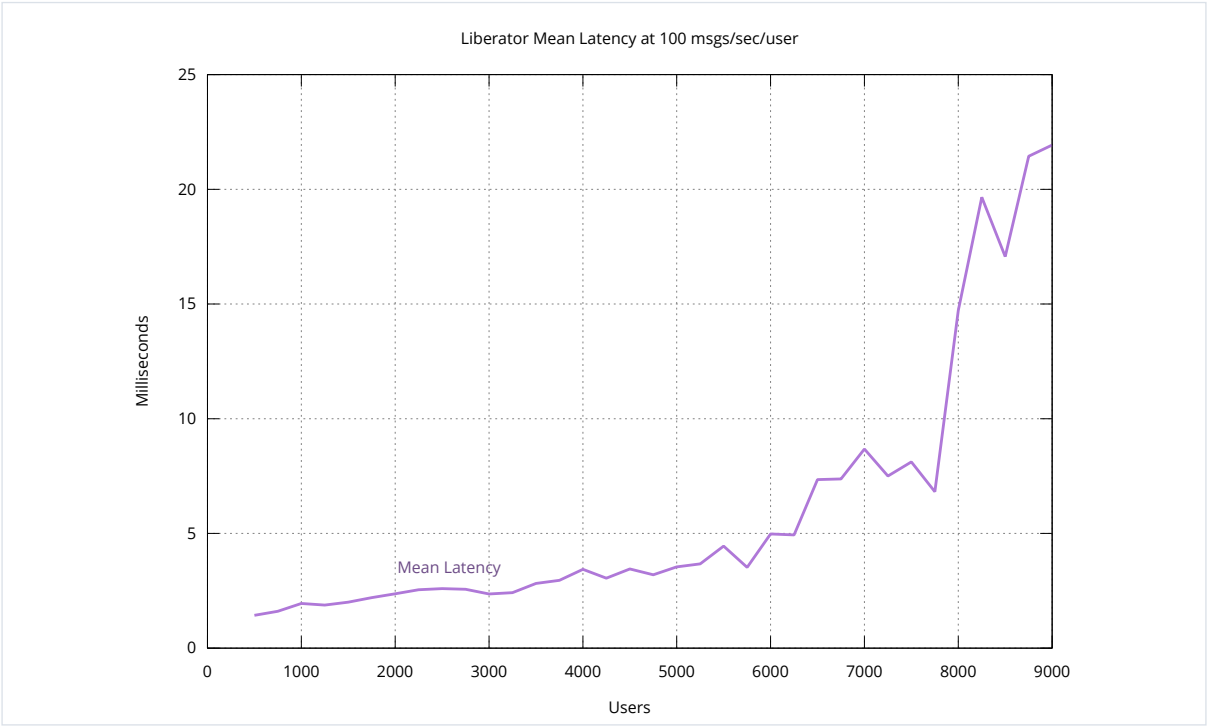

Figure 14. Very High Update Rate: mean latency (500–9,000 users)

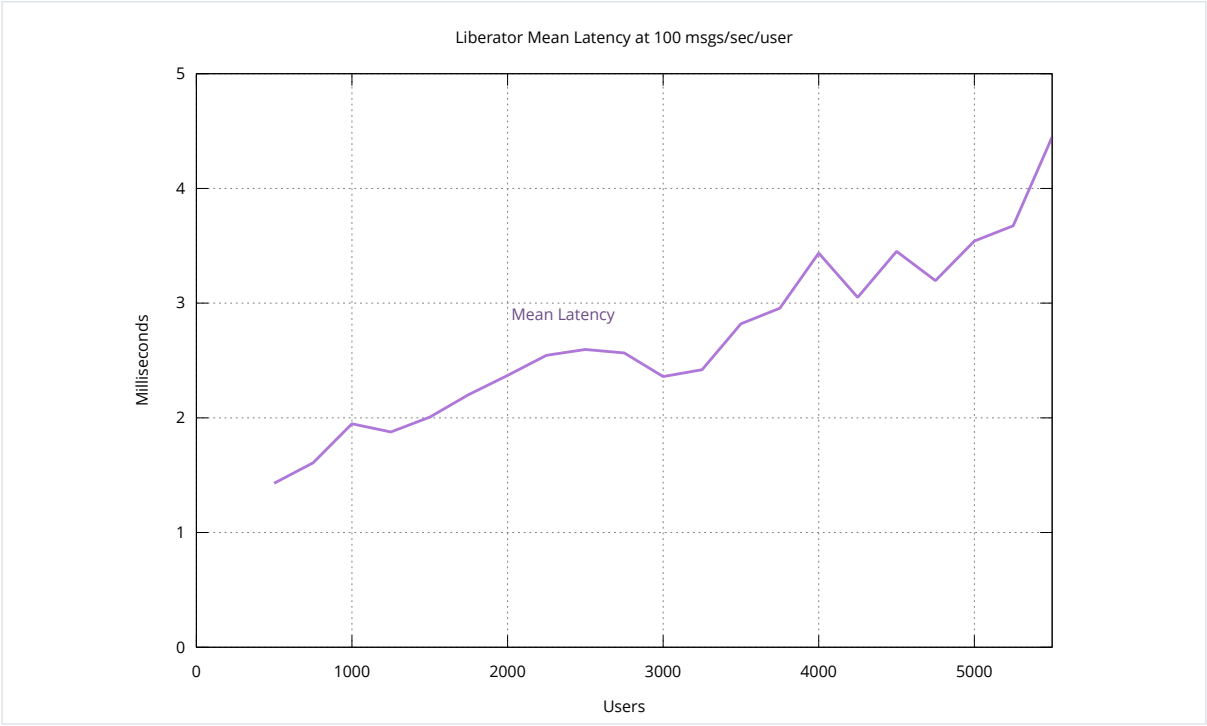

Figure 15. Very High Update Rate: mean latency (500–5,500 users)

© Caplin Systems Limited 2023

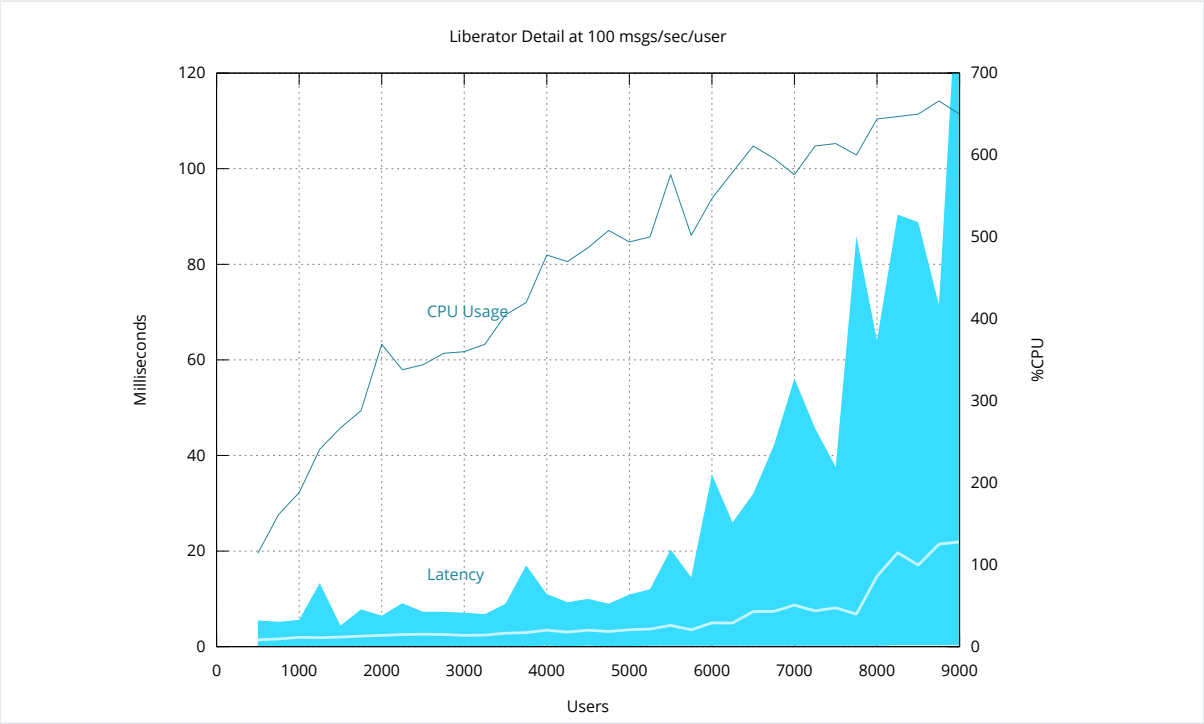

Figure 16. Very High Update Rate: latency range and CPU usage (500–9,000 users)

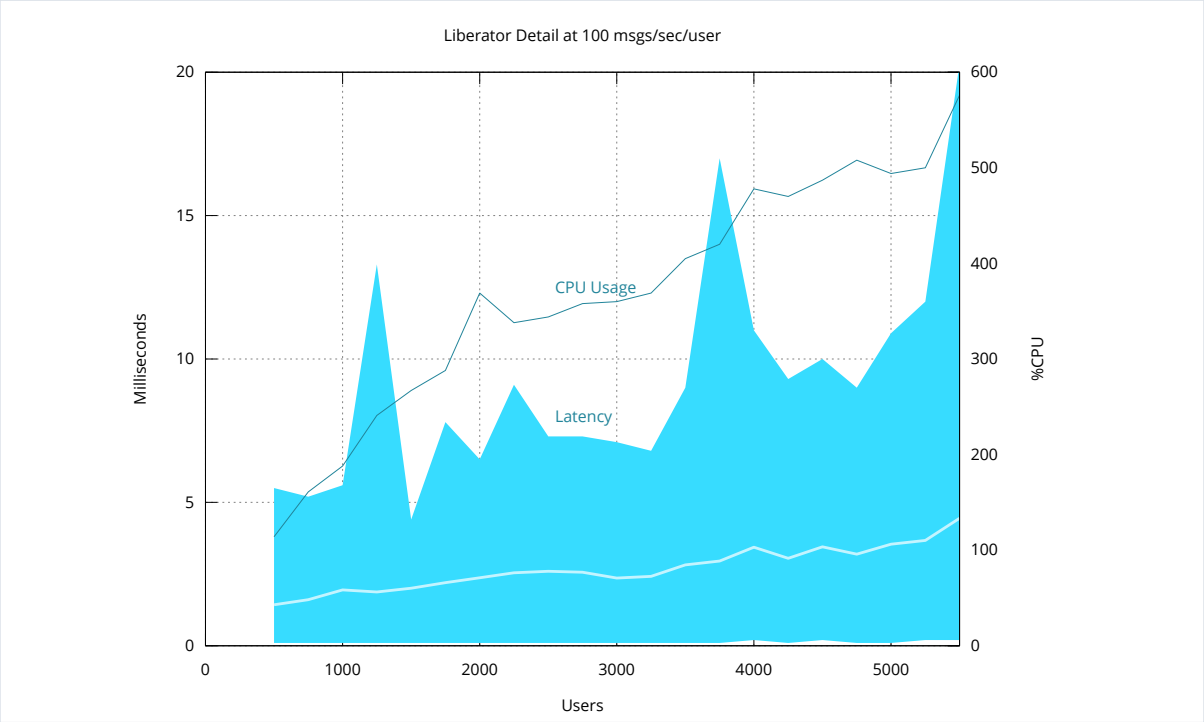

Figure 17. Very High Update Rate: latency range and CPU usage (500–5,500 users)

© Caplin Systems Limited 2023

The Very High Update Rate scenario has a throughput of 100 messages/second for each client (user). This throughput is representative of the workload when each user streams updates for 25 different on-screen instruments, with each instrument streaming at 4 updates/second. For example, a web application displaying 12 FX tiles (48 messages/second) and a watchlist of 12 currency pairs (48 messages/second) would fall within the bounds of this scenario.

Very High Update Rate: batching disabled, 250–5,000 users, abridged

Users	Mean Lat. (ms)	Min Lat. (ms)	Max Lat. (ms)	CPU (%)
250	1.089	0.1	3.9	61
500	1.581	0.1	5.1	123
750	1.557	0.1	5.4	158
1,000	1.657	0.1	12.4	206
2,000	2.28	0.1	9.7	300
3,000	2.324	0.1	7.9	457
4,000	2.266	0.1	8.9	458
5,000	2.675	0.1	14	520

© Caplin Systems Limited 2023

4.7. Very High Update Rate results (batching enabled)

Liberator can be configured to batch messages together at the point they are sent to the client. This does not affect the data being sent, just how it is sent. Instead of sending a lot of small packets containing one message each, Liberator sends larger packets, less frequently, containing multiple messages.

The batching feature will only batch messages together when the update rate is over a configured amount, therefore it can be used to handle peak data rates. However, the scenarios tested here have a more uniform update rate so the batching is always active, when configured.

The following graph shows the mean latency of multiple test runs with different configurations for batching.

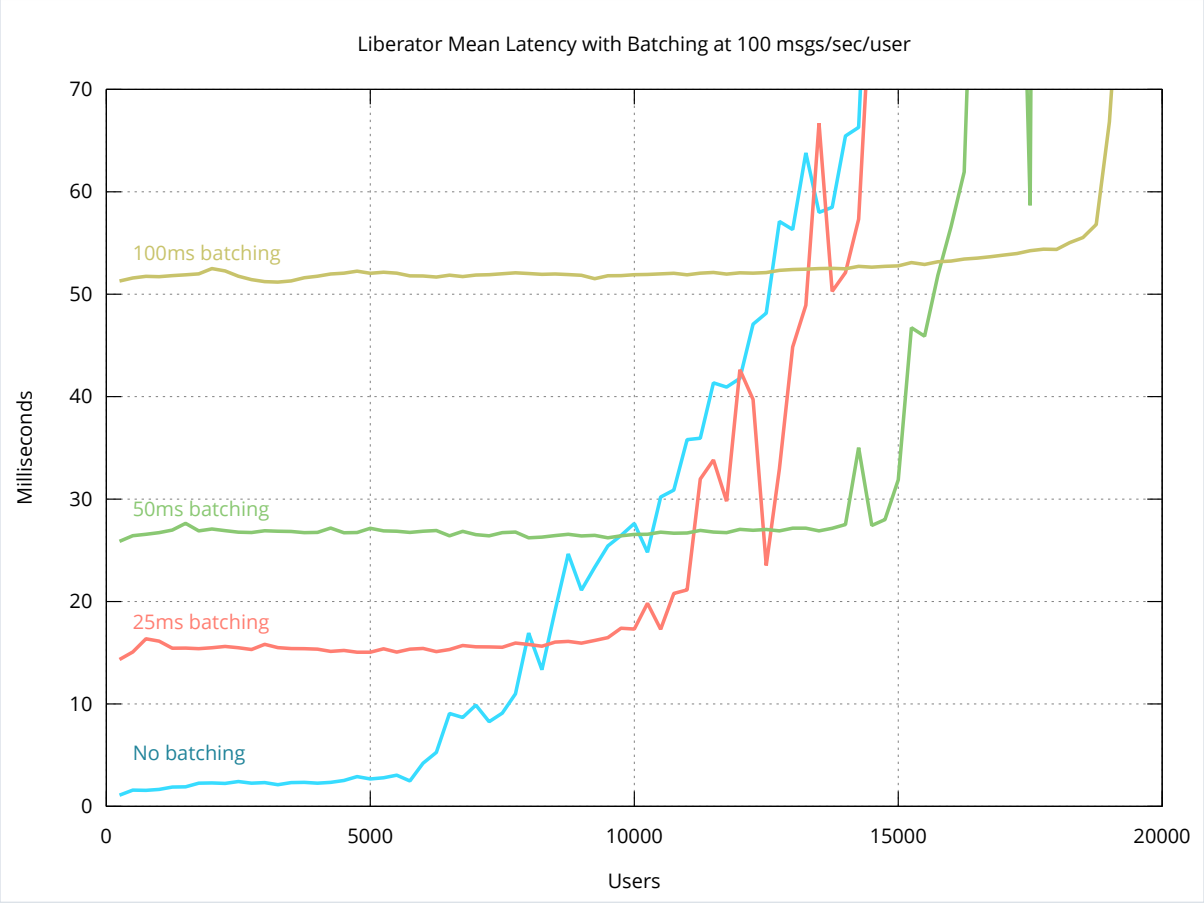

Figure 18. Very High Update Rate: batch intervals 0ms, 25ms, 50ms, and 100ms

The blue line shows the mean latency with no batching configured; this is the same run as the graphs above. The other three lines show how batching messages over intervals (25ms, 50ms, and 100ms respectively) allows Liberator to support many more clients at the expense of increasing latency.

© Caplin Systems Limited 2023

With 100 milliseconds batching, Liberator served 18,000 clients with a mean latency of 54 milliseconds.

The benefits of batching are more apparent at 100ms than they are at 25ms. In the Very High Updates scenario, each client received 50 updates per second. At 100ms batching (10 batches sent per second), each batch contains an average of 5 messages. At 25ms batching (40 batches sent per second), each batch contains an average of 1 message.

© Caplin Systems Limited 2023

4.8. Message sizes

So far all the tests have used a message size of 54 bytes. This may seem like a small message, but it contains 5 fields, typical of a financial application. RTTP, the protocol used between Liberator and clients, is optimized to keep message sizes as small as possible.

To compare the effect of increasing message size on latency, we re-ran the High Updates scenario with three additional message sizes: 108 bytes, 162 bytes, and 216 bytes.

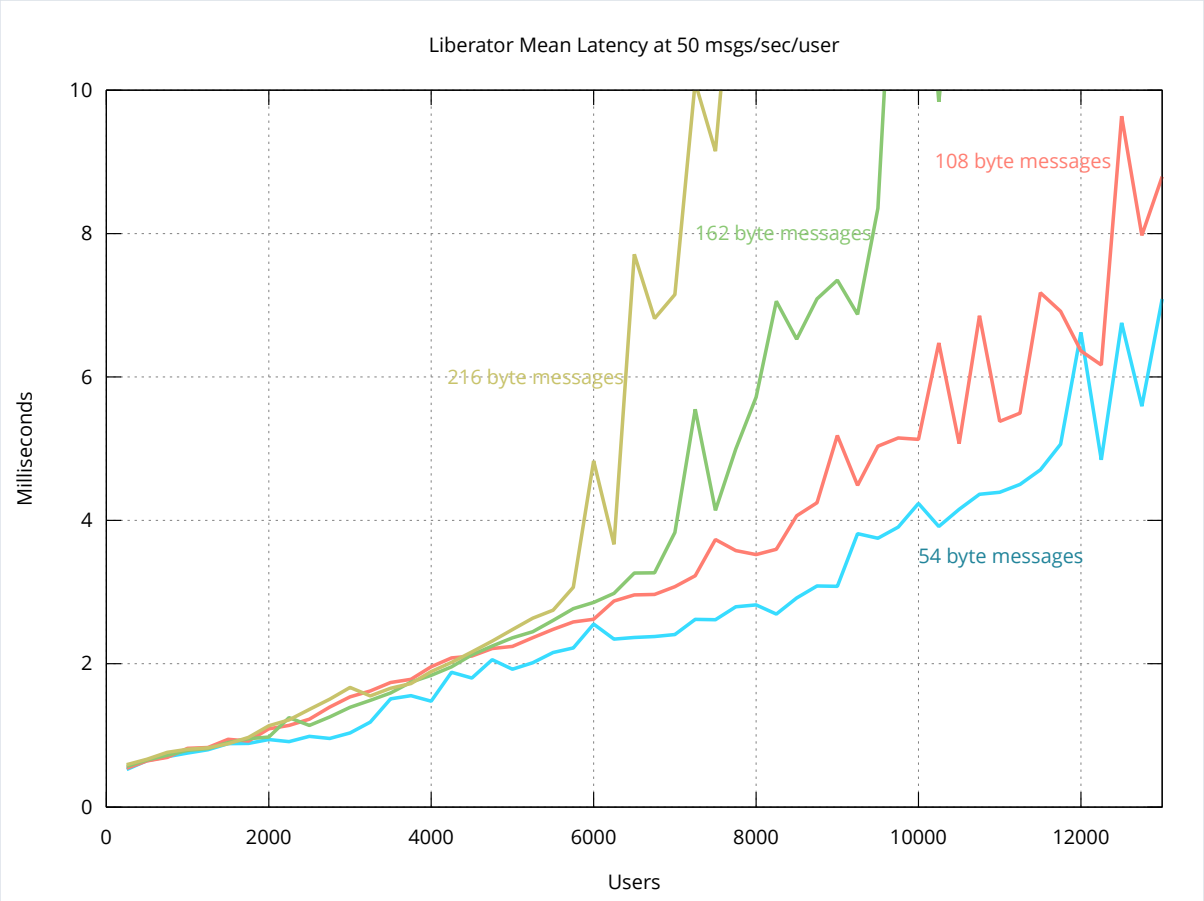

Figure 19. High Update Rate: message sizes 54 bytes, 108 bytes, 162 bytes, and 216 bytes

The graph shows that as message size increases, latency is not affected much, but the number of clients that Liberator can support is reduced.

© Caplin Systems Limited 2023

5. How the benchmark tests were conducted

The following sections describe the test method used, give information about the test configurations, and detail the test software, test hardware, and the network used.

5.1. Test approach

Although the benchmark consisted of several different tests, they all followed a similar method. Each test consisted of one or more DataSources publishing messages into a Liberator which pushed the messages out to a set of subscribing clients through RTTP connections. Each subscribed object was updated at a regular rate by the supplying DataSource. Additional clients were logged on to the Liberator throughout the test run to determine the effect of increasing the load on the Liberator.

5.2. Test setup

The multiple RTTP client connections were simulated using a specially written application called Benchrttp. The DataSource application supplying the Liberator (Benchsrc) was also specially written. Both Benchrttp and Benchsrc are controllable using a command protocol, thus allowing message rates and number of clients to be remotely managed using scripts.

© Caplin Systems Limited 2023

The following diagram shows the hardware configuration used for the tests, and how the test software was distributed across the hardware.

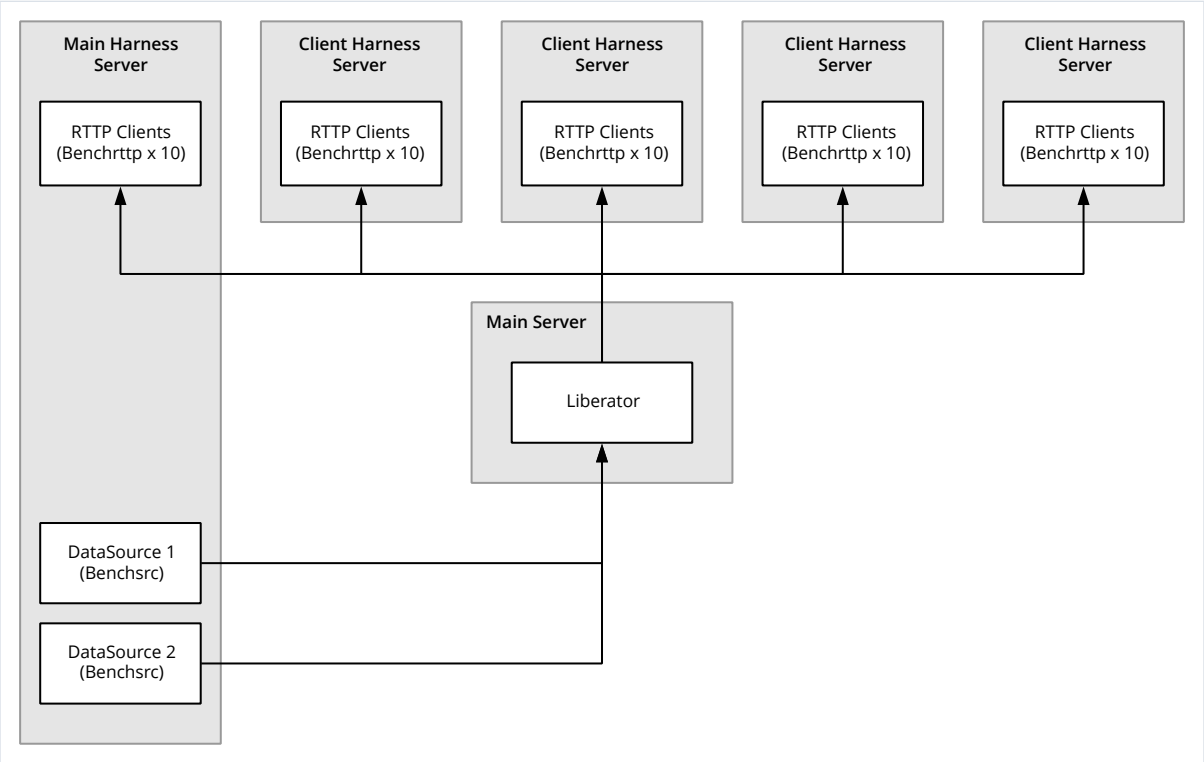

Figure 20. Benchmark test setup

The Liberator server was hosted on a machine of its own (Main Server). A separate machine (Main Harness) was used to host two DataSources (Benchsrc) feeding the Liberator. The Main Harness and four further machines (Client Harnesses) hosted ten instances each of Benchrttp.

The number of Benchrttp instances required for each test was determined by the maximum number of simulated clients needed to run the test; enough Benchrttp resource was required to ensure that Liberator limits could be reached before any limits imposed by Benchrttp.

5.3. Test configurations

Most of the tests determined the mean latency of updates against the number of subscribing clients. This gave a measure of how the Liberator performed as the overall client update rate increased.

Each client subscribed to a number of objects (the number varied according to the test). Because each subscription required Liberator to update those objects on the client in line with the updates from the DataSources, increasing the number of logged on clients had the effect of proportionally increasing the Liberator's client update load.

© Caplin Systems Limited 2023

For example, if each client subscribed to 100 objects and there were 2,000 clients logged on to the Liberator, then the Liberator would have to supply 100 x 2,000 object updates per second in total.

The latency of the client update messages was measured at the client end. In all cases measurements were taken from the application log files and the results were plotted.

The most important Liberator parameters affecting performance are:

- The number of DataSource threads.
- The number of session threads (`threads-num` configuration parameter).
- The maximum interval in seconds during which messages are batched together (`burst-max` configuration parameter).

Many of the tests were performed with varying numbers of DataSource threads and/or session threads, in order to determine the impact of the thread settings on Liberator performance. The test results show the optimum configuration for each scenario.

5.4. Test software

5.4.1. Caplin Liberator

The tests were run against a Caplin Liberator 7.1 server.

The following changes were made to Liberator's configuration:

- Log Cycling. This was modified to prevent the very large amounts of data being processed from using up too much disk space.
- System Max Files. This was increased to 131,072 to allow high numbers of clients to connect. See the `system-max-files` configuration item.
- Object Throttling. The default object throttling of 1 second was turned off for benchmarking. This allowed all clients to receive all the messages they were subscribed to. See the object-throttle-off configuration item.
- Threads. The number of DataSource threads and session threads (`threads-num`) were adjusted according to the needs of the individual tests. Many of the tests were repeatedly run with different numbers of threads in order to determine the impact of the

© Caplin Systems Limited 2023

thread settings on Liberator performance.

- Burst Settings. The default setting for `burst-max` is 0.5 seconds. This was altered to 0 seconds for most of the tests. The High and Very High scenarios test several values for this parameter (0, 0.025, 0.05, 0.1).

5.4.2. Operating system

The following changes were made to CentOS 7.4.1708 running on the Main Server (hosting Liberator):

- The number of open file descriptors was increased to allow Liberator to support high numbers of client connections.

This manual change to the operating system's configuration was required because our benchmarking tools do not start Liberator as root. Without initial root-level privileges, Liberator cannot automatically increase the file-descriptor limit assigned to its process.

After Liberator is started as root, it changes the account it runs under to the account named in the `runtime-user` configuration item.

5.4.3. Test DataSource application (Benchsrc)

The Benchsrc test tool was configured to produce updates to sets of objects at a known message rate. Benchsrc is a single-threaded application, so for those tests where the Liberator was configured to use multiple DataSource threads there was an instance of Benchsrc for each DataSource thread.

5.4.4. Test RTTP client application (Benchrttp)

The Benchrttp test tool was configured to request sets of objects that were published by the DataSource. It measured the number of messages being received and also the latency of the messages. The simulated clients all used RTTP Type 2 connections (RTTP over a persistent HTTP connection).

© Caplin Systems Limited 2023

5.5. Test hardware

The benchmark tests used 6 machines, as shown in the diagram Figure 20, “Benchmark test setup”.

Caplin Liberator ran on a dedicated machine (Main Server).

The RTTP client processes (Benchrttp) ran on 5 machines to spread the load (Main Harness and Client Harnesses).

The DataSources (Benchsrc) ran on the Main Harness.

All our test machines run Broadcom network cards, which we found perform better than Intel network cards.

5.5.1. Main Server

The main server runs Liberator.

Components	Liberator
Vendor	DELL
Model	PowerEdge R7415
Processors	AMD EPYC 7551P 32-Core 2GHz/2.55Ghz/3GHz
NUMA nodes	4
Memory	64GB
Operating system	CentOS 7.4.1708
Network card	Broadcom NetXtreme BCM5720 Gigabit Ethernet PCIe

5.5.2. Main Harness

The main harness hosts two Benchsrc instances, which provide the data for the tests. It also runs 10 Benchrttp instances, so that latency can be measured using the same clock from source to destination.

Components	Benchsrc, Benchrttp(s)
Vendor	HP

© Caplin Systems Limited 2023

Model	ProLiant DL360p Gen8
Processors	2x Intel® Xeon® CPU E5-2650 v2 @ 2.60GHz
Memory	96GB
Operating system	CentOS 7.4.1708
Network card	Broadcom NetXtreme II BCM57810 10 Gigabit Ethernet

5.5.3. Client Harnesses

These machines run the rest of the client harnesses. Latency is still measured on these machines, to assert that thresholds are not broken, but graphs are all based on times recorded on the main harness machine.

Client harness 1

Components	Benchrttp(s)
Vendor	DELL
Model	PowerEdge R415
Processors	2x Six-Core AMD Opteron 4180 2.6GHz
Memory	16GB
Operating system	CentOS 5.5 (final)
Network card	Broadcom NetXtreme II BCM5716 Gigabit Ethernet

Client harness 2, 3, and 4

Components	Benchrttp(s)
Vendor	DELL
Model	PowerEdge R210
Processors	1x quad-core Intel Xeon X3460 2.8GHz
Memory	4GB
Operating system	CentOS 5.5 (final)
Network card	Broadcom NetXtreme II BCM5716 Gigabit Ethernet

© Caplin Systems Limited 2023

5.5.4. Network

The test machines were set up on a Gigabit network that was used solely for transmitting the test data. Control messages and terminal access used a separate network.

© Caplin Systems Limited 2023

6. Frequently asked questions

The following sections discuss various issues concerning how to configure and tune Liberator and its environment to achieve the required performance.

6.1. What batching configuration should we use?

Batching messages together is a more efficient method of sending messages and allows Liberator to support higher message rates. High message rates are associated with fast-changing data and/or a high number of users.

Batching is configured by three Liberator configuration items:

Configuration item	Default value (seconds)
`burst-min`	0.1
`burst-max`	0.5
`burst-increment`	0.05

By default, Liberator is configured to batch messages in intervals of 0.1 seconds (`burst-min`). If more than three messages arrive in a batching interval, the next interval is increased by 0.05 seconds (`burst-increment`) up to a maximum batching interval of 0.5 seconds (`burst-max`). If no messages arrive during a batching interval, the next interval is decreased by 0.05 seconds (`burst-increment`) to a minimum batching interval of 0.1 seconds (`burst-min`).

For a full description of the interplay between these three configuration items, see How batching works on the Caplin website.

The benefits of batching are bought at the expense of the extra latency introduced by the batching interval. If the default maximum batching interval of 0.5 seconds is too high for your application, reduce `burst-max` in increments to a level acceptable to your requirements. As demonstrated by the results for the High Updates (Batching) and Very High Updates (Batching) scenarios, even a lower `burst-max` setting of 0.1 seconds is significantly more efficient than no batching at all.

© Caplin Systems Limited 2023

6.2. How many threads should we configure?

There isn't a simple answer to this question.

Liberator's configuration option `threads-num` sets the number of session threads used to service client connections. Liberator also implements multiple DataSource threads — there is one thread per DataSource connection.

How many threads you require depends on the maximum anticipated update rate from Liberator's DataSource peers, the maximum expected number of subscribing clients, the subscription profile of the clients (see Section 6.2.1, "DataSource threads"), and the hardware characteristics of the machine running the Liberator (CPU speed and number of CPUs).

Caplin Systems can provide customers with expert guidance on configuring the optimum number of Liberator threads to meet their particular requirements.

6.2.1. DataSource threads

DataSource threads enable Liberator to better handle high update rates from its DataSources. Even if there is only one DataSource instance feeding Liberator, you can still configure more than one connection to the DataSource so that the Liberator's performance can benefit from using multiple DataSource threads.

There is also a relationship between the number of clients using the Liberator and the number of DataSource threads required, but this depends on the subscription profile of the clients, as follows.

At one extreme, if each client subscribes to a different set of objects, then the update demand on the DataSources will increase roughly in proportion to the number of subscribing clients. So in this case, as the maximum anticipated number of clients increases, more DataSource threads will be required in order to achieve the same message latency.

At the other extreme, if each client subscribes to the same set of objects, the update demand on the DataSources will remain constant as the number of subscribing clients increases. So in this case the number of DataSource threads required will not depend on the number of subscribing clients.

You should configure enough DataSource connections (and hence DataSource threads) to allow Liberator to handle the maximum anticipated update rate from the DataSource(s) with

© Caplin Systems Limited 2023

acceptable message latency for the maximum number of clients expected to use the Liberator.

Don't use more DataSource threads than you need, because the additional threads will not produce a significant additional improvement in performance. There will be a limit above which adding more DataSource connections has little extra benefit, because the limiting factor becomes the number of session threads.

6.2.2. Session threads

Provided there are enough DataSource threads to handle updates from the DataSources, increasing the number of session threads will allow more clients to subscribe with no unacceptable increase in message latency. However there will be an upper limit on the number of session threads, beyond which performance will decrease – see Section 6.2.3, "CPU resource considerations".

6.2.3. CPU resource considerations

Once CPU resource limits have been reached on the machine running the Liberator, adding more threads will not necessarily improve performance. In these tests, 8 session threads gave the best results.

On servers with fewer than 12 cores, reduce the number of session threads according to this formula: `num_session_threads + 4 = num_cores`

6.3. How does message size affect performance?

When high numbers of clients and messages are used, the size of the message plays a significant part in the overall performance. Larger update messages decrease Liberator's maximum effective message rate. Liberator also performs slightly better when handling update messages consisting of a small number of large fields rather than messages containing a larger number of smaller fields.

6.4. How does network latency affect performance?

The tests in this document were performed on an internal network. Over the Internet, there will be an increase in latency of anywhere between 20-250 milliseconds. The exact latency will very much depend upon the geographical locations of the sites – with 100-120

© Caplin Systems Limited 2023

milliseconds exhibited for transatlantic communications, for example.

6.5. How many subscriptions can Liberator handle?

The number of subscriptions a client has does not significantly affect performance directly, rather the number of messages is far more significant. This can be controlled using throttling.

6.6. How much disk space will our Liberator need?

Liberator uses approximately 60 MB of disk space when it is installed. Running Liberator requires extra disk space for log files.

The amount of disk space needed for the log files depends entirely on messages rates and client activity. Liberator supports configuration options to control the cycling of log files, so it is possible to limit how much disk space is used and how much information is saved in log files. Log files can grow to high single-digit gigabytes per day in some setups.

© Caplin Systems Limited 2023

Appendix A: Glossary

This section contains a glossary of terms and acronyms relating to the Liberator benchmark.

Benchsrc

A Caplin benchmark test tool that provides DataSource messages as input to a Caplin Liberator server. It is primarily intended to be used in conjunction with Benchrttp and the control scripts from the Caplin Benchmarking kit.

Benchrttp

A Caplin benchmark test tool that connects to a Liberator server and simulates a configurable number of clients and contributors of streamed RTTP data. It is primarily intended to be used in conjunction with Benchsrc and the control scripts from the Caplin Benchmarking kit.

Batching / Bursting

In a system with a large number of clients, the efficiency of Caplin Liberator can be increased by writing output in defined batches (bursts).

`burst-max` is a Liberator configuration parameter that controls bursting. It is the maximum time in seconds of buffering before Liberator sends updates to a client.

For more information see Bursting (batching) on the Caplin website.

Caplin Liberator

Caplin Liberator is a real-time financial internet hub that delivers trade messages and market data to and from subscribers over any network.

Caplin Platform

A framework for building single-dealer platforms that enables banks to deliver multi-product trading direct to client desktops.

DataSource

DataSource is the internal communications infrastructure used by Caplin Platform's server components such as Caplin Liberator, Caplin Transformer, and DataSource adapters.

DataSource adapter

A DataSource application that integrates with an external (non-Caplin) system,

© Caplin Systems Limited 2023

exchanging data and/or messages with that system.

DataSource application

A Caplin Platform application that uses the Caplin DataSource APIs to communicate with other Caplin Platform applications via the DataSource protocol.

DataSource peer

Alternative name for a DataSource application.

© Caplin Systems Limited 2023