Planning hardware capacity for IBM® Rational® Performance Tester

Agenda

- Overestimating IBM® Rational® Performance Tester memory usage
- Rational Performance Tester architecture: How it impacts scalability
- Scaling up for high-volume load tests – Guidance for best results
- Driver memory and processor* measurement results
- Summary driver sizing guidance for Rational Performance Tester 8.5.1.3
- Legal notices

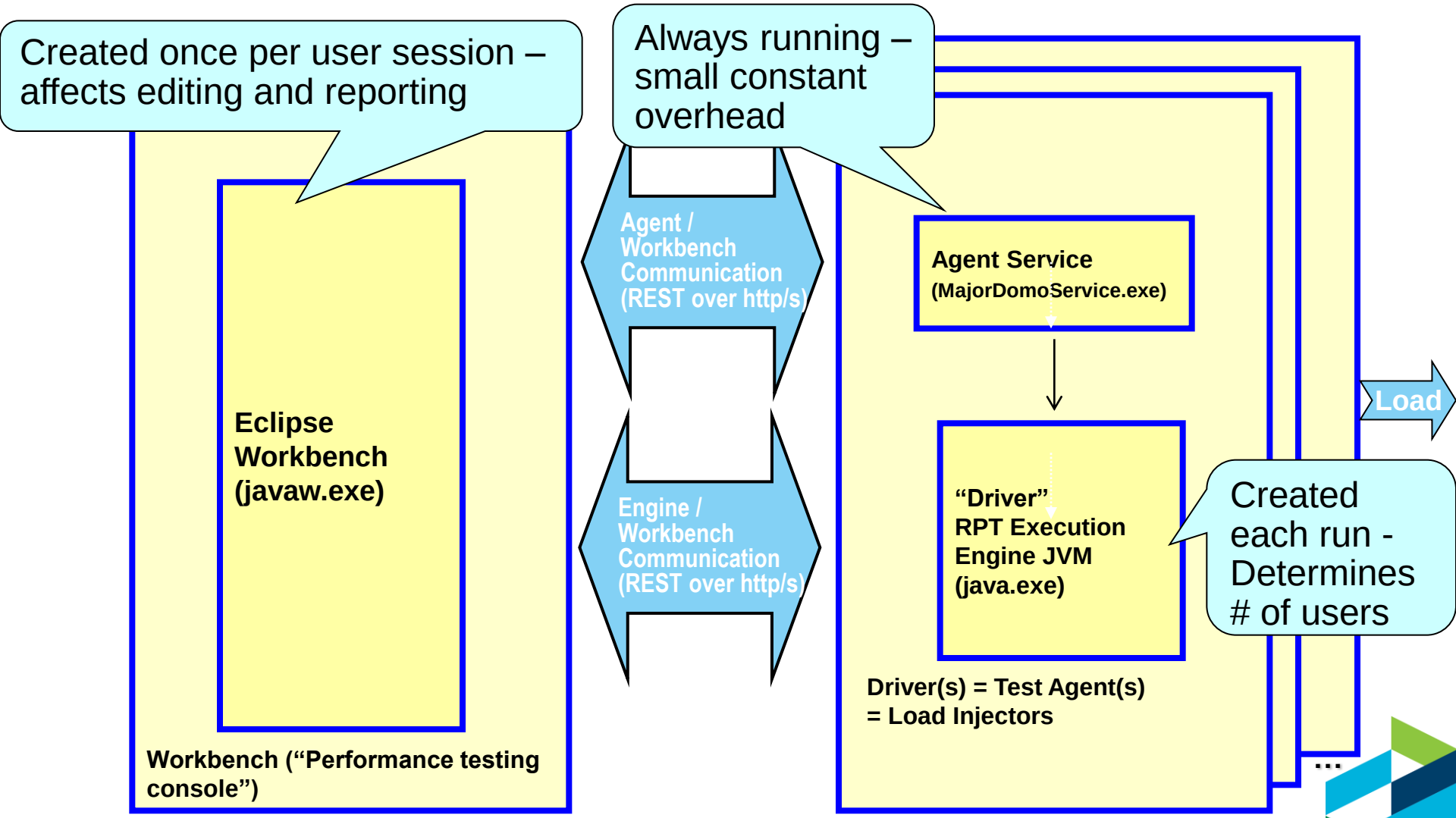
**In this document, "processor" and "CPU" refer to the central processing unit or units.*

Overestimating Rational Performance Tester memory usage

- Frequently first-time users of Rational Performance Tester severely overestimate the memory use of Rational Performance Tester.

- Common reasons for overestimation
  - Including workbench memory usage
  - Making one or two small runs and disregarding fixed driver engine overhead
  - Running users in “lock-step”
  - The presence of excessive page response times or timeouts

Rational Performance Tester process architecture (computer and process/JVM boundaries)



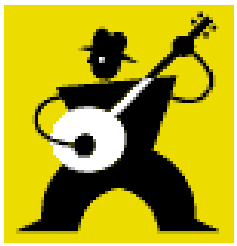
Memory, Java, JVM heap size, and Rational Performance Tester

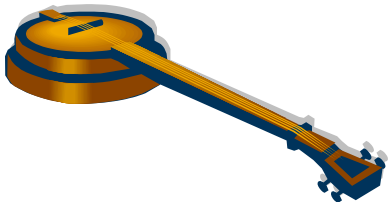
- High-volume load testing typically requires significant amounts of driver memory.
- Java® implementation means that available memory is limited to the JVM maximum heap size *regardless of the amount of memory on the computer.*
- Hitting the max heap size is not pleasant… ☹
- Rational Performance Tester is implemented in Java – so be aware of JVM max heap sizes to achieve high volume load testing with Rational Performance Tester.

Dueling JVMs

- Two JVMs are of particular interest in Rational Performance Tester:
  - The Eclipse workbench (javaw.exe) for Rational Performance Tester
  - The “driver” (Rational Performance Tester execution engine; java.exe)
- These two JVMs have different impacts on scalability based on their respective roles in the overall Rational Performance Tester architecture.

  The next five slides present the impact of these JVMs on Rational Performance Tester and how you can plan for that impact.





The Rational Performance Tester Eclipse workbench JVM: Process size

- The workbench JVM (javaw.exe) is “overhead” and independent of the number of users that are active in a running test.
- The process size starts at 50 - 100 MB and can grow significantly when viewing or editing large assets such as tests, reports, and datapools.
- However, the javaw.exe process memory consumption typically stabilizes at the maximum heap size (plus overhead).
- “Garbage collection” has the potential to reduce unused heap memory over time, but you can start the cleanup manually too.
  - TIP: Set this Eclipse general preference: **Show Heap Status**.

- The display shows the current heap size **used** of the **allocated** heap size – **not** of the max heap size.

- Click the **Garbage Can** icon to start manual garbage collection. **Note:** Starting garbage collection manually might help reduce memory use, but manual collection does not guarantee a reduction.

The Rational Performance Test Eclipse workbench JVM: Max heap size

- Install the workbench on 64-bit operating systems with a minimum of 4GB memory for best results.
- The default maximum heap setting is based on the amount of physical memory, 32 vs. 64 bit operating system, and 32 vs. 64 bit JVM, as follows:

Machine Memory		RPT 8.5		
		Windows and Linux		
Phy Mem	HalfMem	32wb on 32os	32wb on 64os	64wb on 64os
1GB	512	512m	512m	512m
2GB	1024	1024m	1024m	1024m
3GB	1536	1200m	1536m	1536m
4GB	2048	1200m	1900m	2048m
6GB	3072	1200m	1900m	3072m
8GB	4096	1200m	1900m	4096m
16GB	8192	1200m	1900m	4096m

- The default settings can be overridden by adding an –Xmx*NNNN*m argument to the -vm argument list in the eclipse.ini file (one each for streamlined & full Eclipse), where *NNNN* is the max heap size.
  - In general, do not exceed a1200 MB max heap size on 32-bit Windows.

The execution engine “driver” JVM for Rational Performance Tester: Memory factors

- Memory use relates primarily to the number of users and workload
- Memory use is primarily proportional to the number of users, allowing for fixed overhead
- Memory use is higher for workloads that have these characteristics:
  - Large GIFs and large posts: High I/O requires more memory.
  - Many page elements per page that are stored concurrently
  - Long response times. These times require memory to be held longer.
  - Short think times. High activity increases “density”.
  - Users that run in “lock-step.” Low “dispersion” leads to higher memory “peaks.”
  - Secure connections. SSL encryption and decryption processing and storage place demands on the processor and memory.
  - Higher browser cache hit rates. Certain data associated with cached responses such as data correlation references must be stored for the duration of the test.
- Memory use of the driver JVM determines the number of driver engines you require *if memory bound.* You can run multiple engines per computer or virtual machine.

How much memory is really being used per user?

- Calculate from size of the driver execution engine (java.exe)
  - Don't factor in workbench memory, which is independent of the number of users.
  - Don't factor in MajorDomoService.exe, which is constant overhead.
- Don't assume that memory usage is strictly proportional (be careful when extrapolating)
  - Don't take the engine's java.exe process size at 10 users and linearly extrapolate to 1000 users!
- Determine Incremental Virtual User Memory Footprint (IVUMF)
  - Take steady-state measurements for 4 - 5 runs while varying the virtual user load to get a range of at least 3 - 5X from min to max values
  - Plot results and run all outliers again. Discard outlier data if necessary.
  - Determine the Intercept and Slope (IVUMF = Slope)

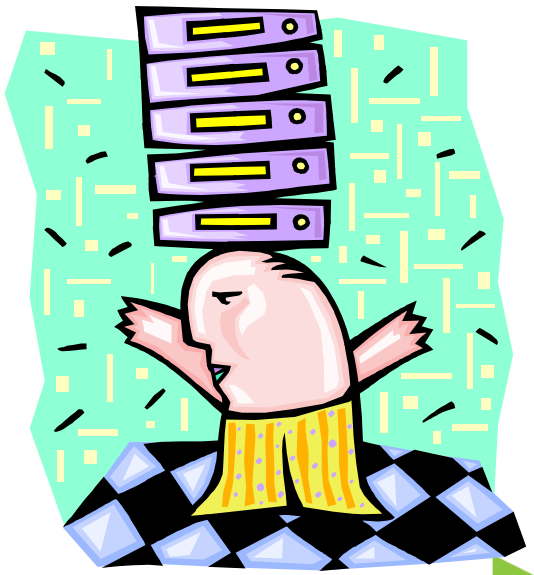
    $M = I + nS$, where I is Intercept, S is slope, and n is the number of virtual users
- Be cognizant of JVM heap-size

Rational Performance Tester assists in setting the JVM max heap size of the driver

- The JVM maximum heap-size of the driver is set automatically based on the amount of physical memory available on the computer. Here's how this function works:
  - The default JVM heap-size is set to 70% of physical memory, capped at 1200 MB for 32-bit JVMs and 12,000 MB for 64-bit JVMs, *unless set by user on that driver location.* See the following details.
- You can override the auto-set maximum heap size by adding the **RPT_VMARGS** property as an attribute of the location where the test is run. See the Rational Performance Tester Information Center.
- Rational Performance Tester attempts to detect the driver out-of-memory condition, but detection is not always possible.
  - Detection is post-occurrence! The event cannot be anticipated or prevented.
  - Auto-detection is based on the last few memory-use samples prior to the loss of communication with the driver.

Scaling up for high-volume load tests

- Moving from debugging small runs to high-volume runs requires a different scale of hardware and different Rational Performance Tester settings
- The different hardware must be configured.
  - Driver configuration guidance
  - Workbench configuration guidance



Driver (agent) configuration guidance [1 of 4]

- Use dedicated drivers for performing the load test.
- Use dual or quad processor systems with at least 2GB memory each.
- Use fewer faster and bigger driver systems rather than more smaller ones for best results *from a workbench scalability perspective:*
  - Fewer drivers mean less communication and startup and cleanup time.
  - Fewer driver systems also reduce statistical model aggregation overhead on the workbench, especially *if* you choose to keep individual statistical data for each driver computer. This setting **is not** the default, but if you clear the **Only store All Hosts statistics** check box in the Schedule dialog box, you get this result:
    - The statistical model size grows proportional to 1 + number of drivers.
  - Caveat: If NIC I/O bandwidth or the number of TCP ports is a bottleneck on the driver systems, then go with more smaller driver systems.

Driver (agent) configuration guidance [2 of 4]

- Avoid loops within a schedule over a short test; move the loop inside the test instead.
  - Loops inside tests are more efficient and enable re-use of HTTP connections across loops. Otherwise the default is to close connections at the end of the test.
  - Loops inside tests produce a huge gain in “blasting” tests!
- Beware of running out of processor capacity before memory
  - keep the average utilization of the process to < 70%.
- Use 1Gb NIC cards, but match the benchmark configuration if specified.
- Linux® drivers require a per-process open file limit higher than 1024.
  - As root, enter the **ulimit -n 30000** command (with an appropriate value) **before** starting MajorDomo (agent).
- Treat Windows and Linux as roughly equivalent for sizing of Rational Performance Tester Version 8.5 and later.
- Top-end desktops utilizing Intel Quad Core i-7 (Sandy Bridge) or later processors might offer the “best bang for the buck.”

Driver (agent) configuration guidance [3 of 4]

- Windows drivers may require an increase in the number of TCP ports. The following guidance applies to Windows versions 7 and 8, and Windows Server versions 2008 and 2012.
- The default number of TCP ports is 16,384, covering a range of port numbers starting at 49152 and ending at 65535.
- You can view the dynamic port range as follows:
  - As administrator, enter the **netsh int ipv4 show dynamicport tcp -n** command.
- You can change the dynamic port range as follows:
  - As administrator, enter the **netsh int ipv4 set dynamicport tcp start=1025 num=64510** command.
  - The minimum starting port number is 1025 and the maximum ending port can not exceed 65535, resulting in a maximum number of 64510 TCP ports.
  - No reboot is required for the changes to take effect.
  - See http://support.microsoft.com/kb/929851 for additional information.

Driver (agent) configuration guidance [4 of 4]

- How to leverage big iron – 16-core, 32-core or larger servers ?
  - Rational Performance Tester is typically processor-bound for most workloads. However, large multi-processor, multi-core servers can often provide enough processing capacity such that the 32-bit JVM the max heap limit (1.2 GB on Windows, ~3 GB on Linux) becomes the bottleneck.
  - To fully utilize large driver systems *(required only if memory-bound)*
    - Run multiple engines by defining separate locations that map to the same system. This setup enables utilization of up to 1200 MB memory **per 32-bit engine**

      You must use network aliases to supply unique hostnames to accomplish this:

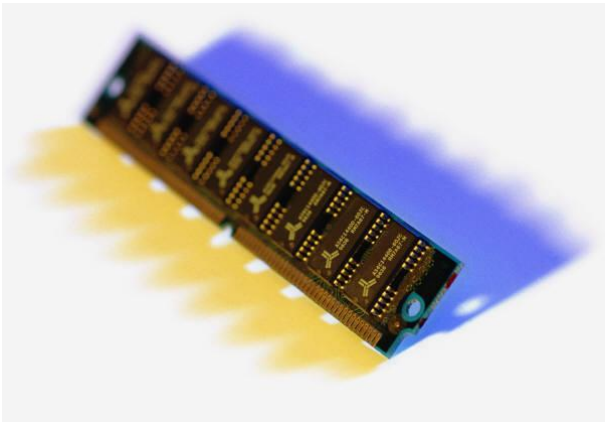
      /etc/hosts (on Windows: C:\WINDOWS\system32\drivers\etc\hosts)

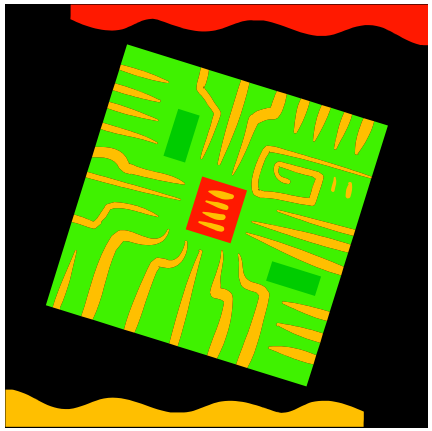
      x.x.x.x driverhostname, driver_2, driver_3, driver_4

      TIP: If you only need two engines per agent machine, you can avoid the use of /etc/hosts aliases by simply defining two locations: use the hostname as one location's address, and use the IP address for the other location's address.
    - Be sure to specify unique deployment directories for each location.
    - Because the physical memory of the agent computer is being shared by all engines on that agent computer, the sum of the default JVM max heap sizes for all engines – especially with 64-bit JVMs – can easily exceed the amount of available memory. Therefore you'll want to override the default max heap by explicitly specifying it on each location, for example: RPT_VMARGS= -Xmx2048m.

Driver memory and processor measurements

- A concentrated effort helped reduce Rational Performance Tester driver memory footprint in version 6.1.2
- A reliable methodology for measuring the Incremental Virtual User (VU) Memory Footprint was developed, and continues to be used with each Rational Performance Tester release.
- This methodology is applied to a variety of workloads and some variations
- Processor and memory results are now available for versions 8.5 and 8.5.1.3



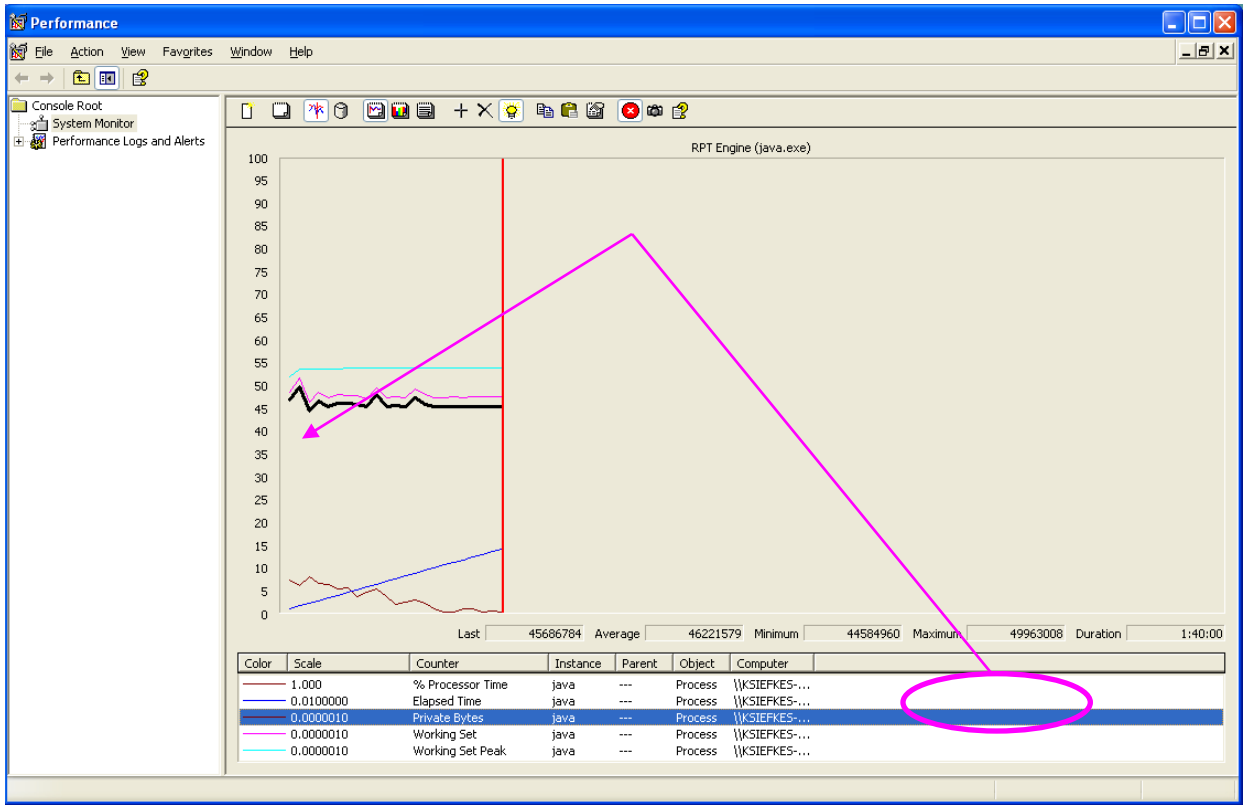


Measuring memory: Getting on the same page!

Windows Process Memory Metrics [units]		Definition
Task Manager [KB]	Performance Monitor [bytes]	
VM Size	Private Bytes	Private Bytes is the current size, in bytes, of memory that this process has allocated that cannot be shared with other processes.
Mem Usage	Working Set	Working Set is the current size, in bytes, of the Working Set of this process. The Working Set is the set of memory pages touched recently by the threads in the process. If free memory in the computer is above a threshold, pages are left in the Working Set of a process even if they are not in use. When free memory falls below a threshold, pages are trimmed from Working Sets. If they are needed they will then be soft-faulted back into the Working Set before leaving main memory.
Peak Mem Usage	Working Set Peak	Working Set Peak is the maximum size, in bytes, of the Working Set of this process at any point in time.

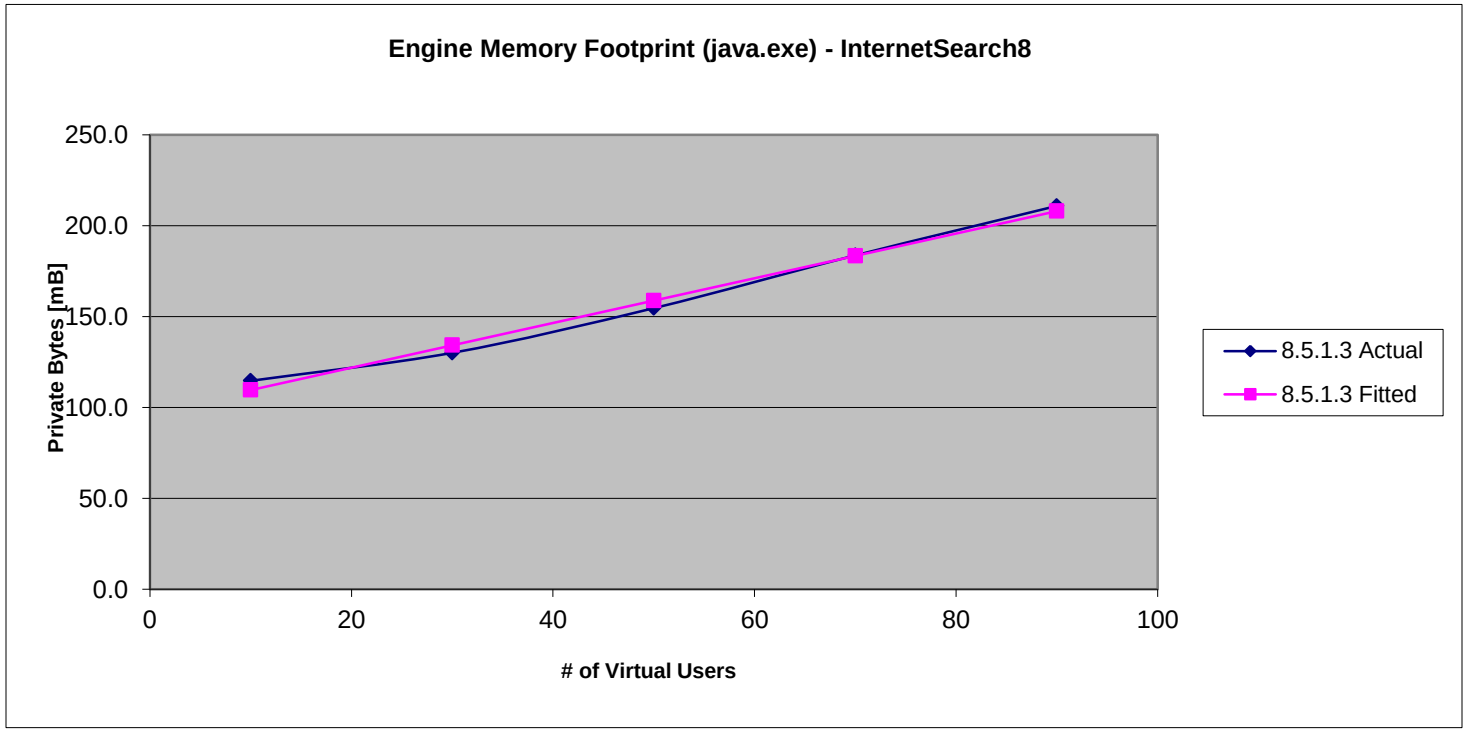
The Task Manager terms are the ones used in the column headings in the process page of Task Manager and are measured in kilobytes. The corresponding Performance Monitor counter values are equal to the Task Manager values multiplied by 1024.

Example of the Perfmon memory measurement methodology



- The selected value of Private Bytes is 49963008, rounded to 50.0 mB.
- Memory measurement has been automated using analysis of Perfmon performance logs ☺
- 60 second intervals are used. Outliers are removed using 3 sigma rule.

Using slope to calculate Incremental VU Memory Footprint



# virtual users	Private Bytes [mB]		% Delta
	8.5.1.3 Actual	8.5.1.3 Fitted	
10	114.7	109.6	4.6%
30	130.1	134.2	-3.1%
50	154.6	158.8	-2.6%
70	183.7	183.4	0.2%
90	210.9	208.0	1.4%
Intercept	97.3	97.3	
Slope	1.230	1.230	

Static Workload Characteristics

Workload	Static Workload Data - Per Test Iteration												
	# of Pages	# of Page Elements			# of VP's		Response Sizes [KB]			Data Correlation [8.5]			
		Total	Average Page	Largest Page	Page Title	Response Code	Total	Per Page	Per Request	# Ref-erences	Per Response	# Sub-stitutions	Per Request
InternetSearch8	3	185	61.7	113	3	185	1187	396	6.4	160	0.9	756	4.1
Plants41k6	41	696	17.0	30	41	696	15544	379	22.3	520	0.7	2435	3.5
DayTrader2	7	27	3.9	18	7	27	283	40	10.5	17	0.6	62	2.3
InternetShopping8	11	590	53.6	207	11	590	3986	362	6.8	254	0.4	4021	6.8

Unless noted otherwise in variant measurements, the following schedule and test settings were used for these workloads:

Workload	Schedule Settings					Test Settings
	Data Collection Levels			Think Time		
	Stats	Test Log	Problem Determination	Mode	Max	Page Cache
InternetSearch8	All	Primary Test Actions	Warning	Recorded think times	10	Disabled*
Plants41k6	All	Primary Test Actions	Warning	Recorded think times	10	Disabled
DayTrader2	All	Primary Test Actions	Warning	Recorded think times	10	Disabled
InternetShopping8	All	Primary Test Actions	Warning	Recorded think times	10	Disabled

* Page cache emulation was disabled for historical reporting consistency purposes. The impact of enabling cache is addressed separately.

Driver and dynamic workload characteristics

All workloads were run on driver “F” for reporting of base measurement results:

ID	Type	Vendor	Model	Memory	Operating System	Bits
F	Desktop	Lenovo	ThinkCentre M91p -[7034BD5]	8 GB	Windows 7 Ultimate SP1	64 **

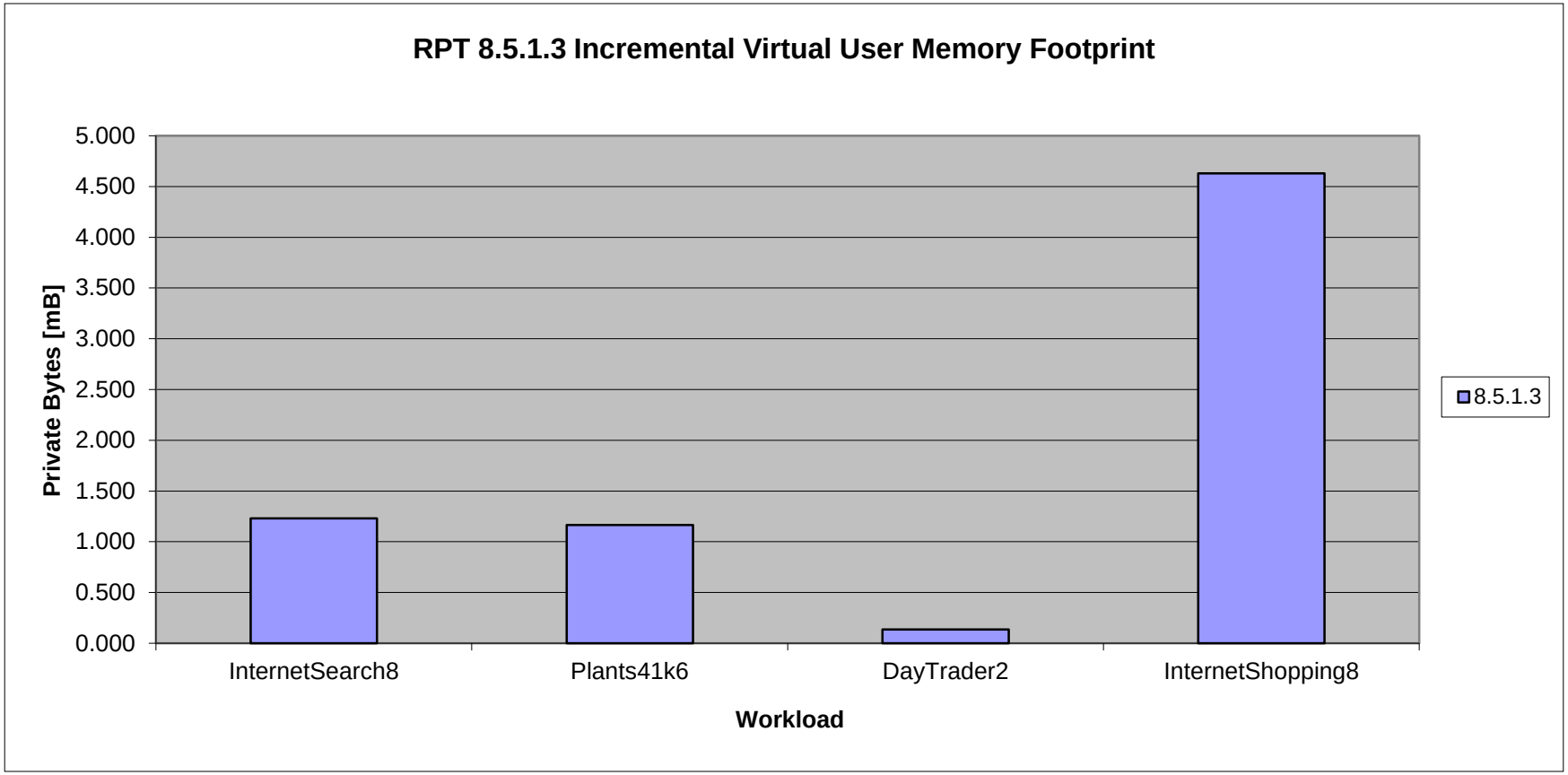
ID	Model	Processor Info	Chipset	Year	# Cores	# Logical Processors
F	ThinkCentre M91p -[7034BD5]	Intel Core i7-2600 @ 3.40 GHz	Sandy Bridge	2011	4	8

** Unless otherwise noted, driver “F” measurements were made using the 32-bit JVM for historical reporting consistency purposes.

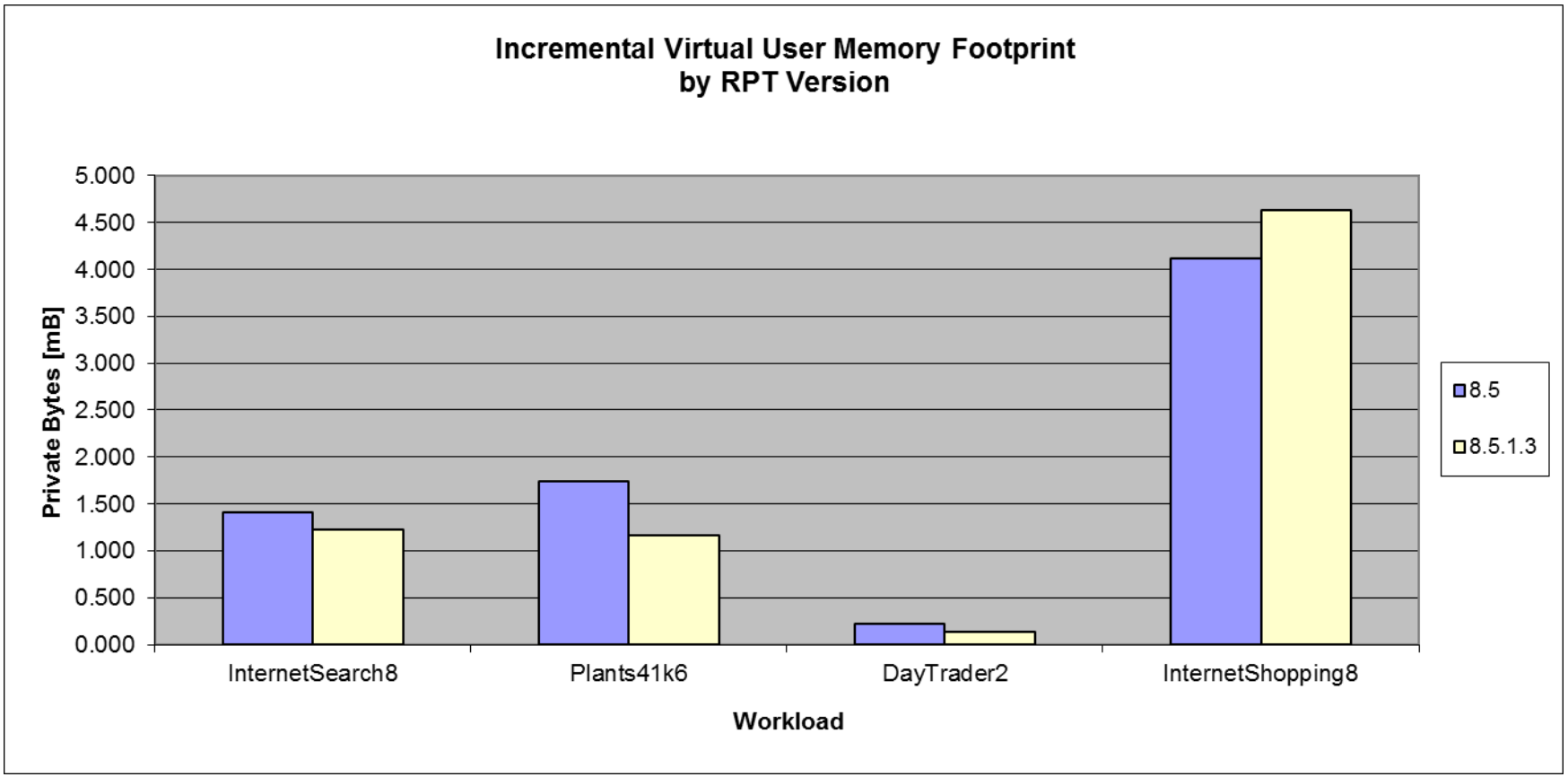
Workload	Driver Used	Dynamic Workload Data						
		Server/Network Configuration	Duty Cycle* [secs]	Per Virtual User		Total @ 70% CPU		
				Request rate [per sec]	Receive rate [KB/Sec]	Projected Users	Request Rate [per sec]	Receive Rate [MB/sec]
InternetSearch8	F	Internet	60	3.1	19.8	1155	3562.4	22.3
Plants41k6	F	Local, gigabit	240	2.9	64.8	9937	28818.2	628.5
DayTrader2	F	Local, gigabit	60	0.45	4.7	14095	6342.7	64.9
InternetShopping8	F	Internet	180	3.3	22.1	804	2635.1	17.4

* The average loop rate was controlled by randomly varying the delay between iterations (negative exponential distribution)

Virtual user memory footprint for sample workloads (v8.5.1.3)



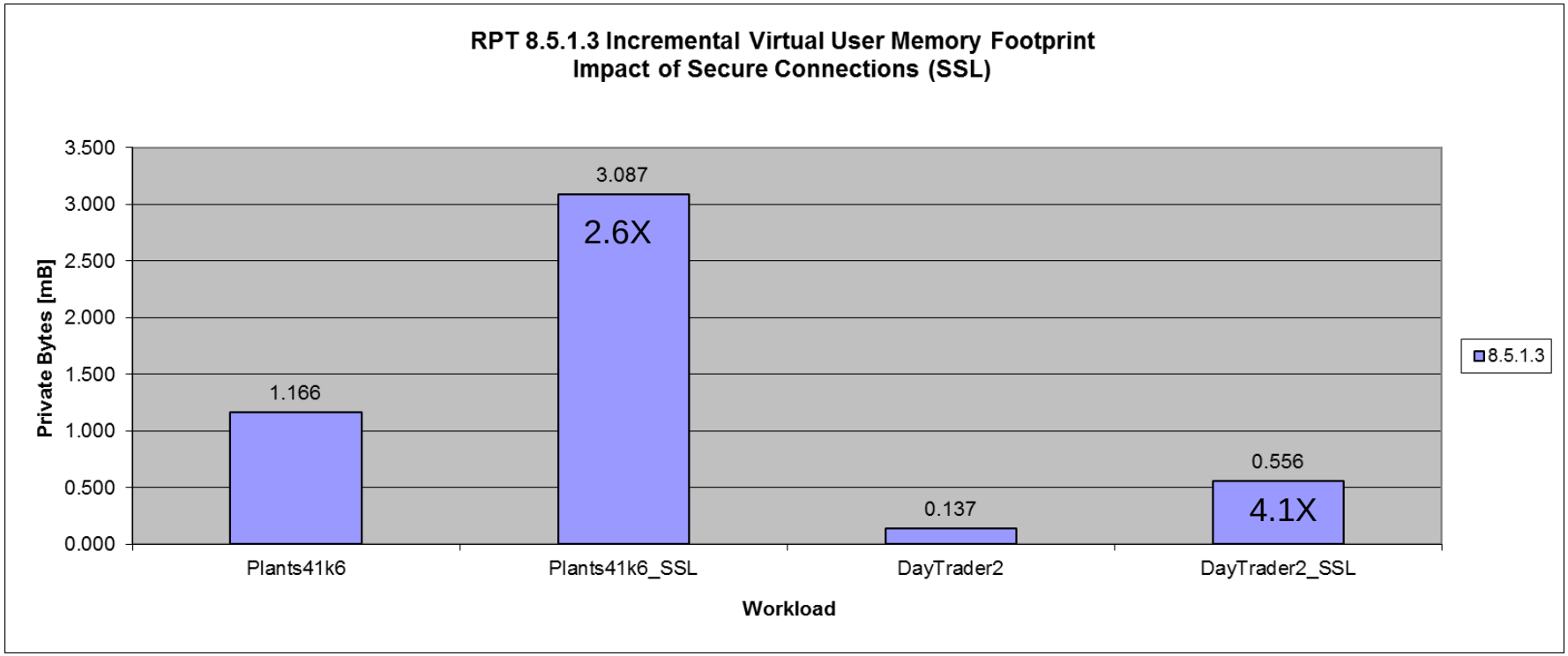
Memory footprint comparison with previous releases



On average* for these four workloads, the Rational Performance Tester version 8.5.1.3 memory footprint is 20% less than the memory footprint in version 8.5.

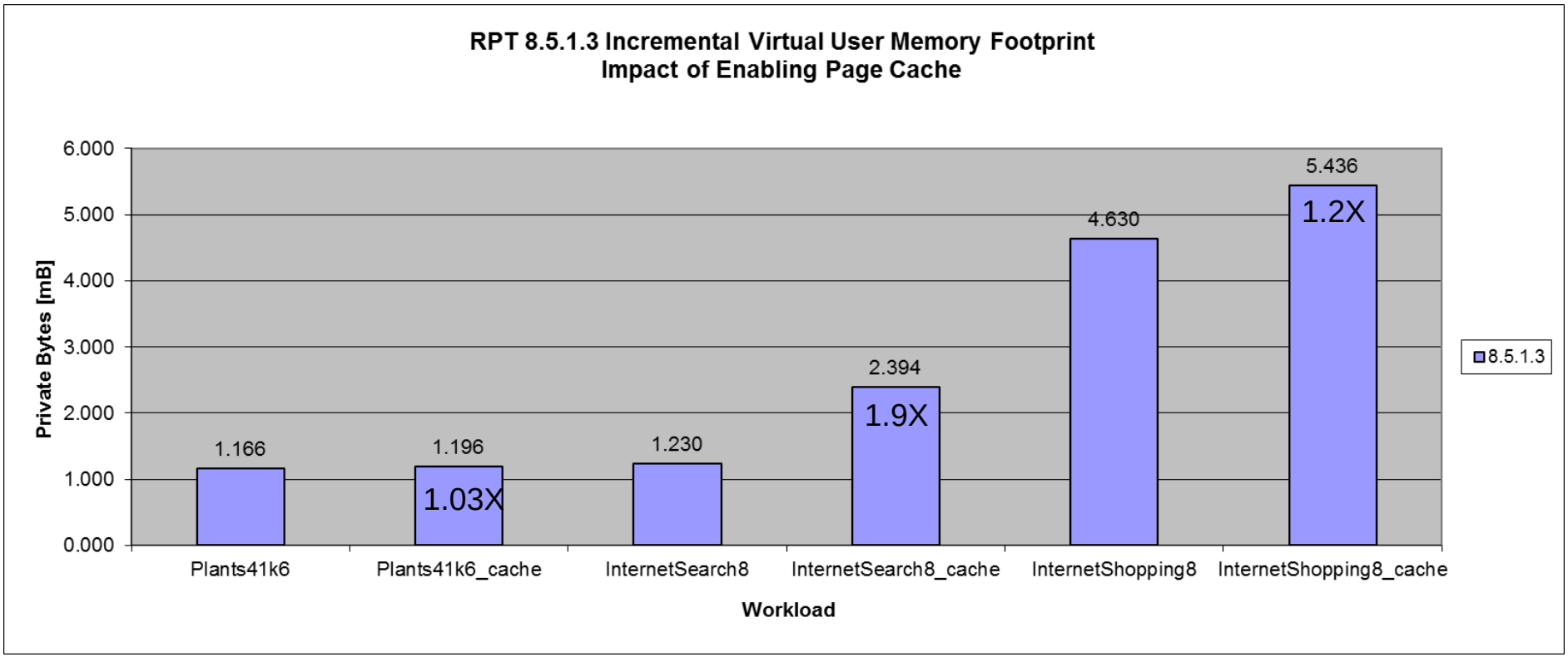
* geometric mean

Virtual user memory footprint: SSL



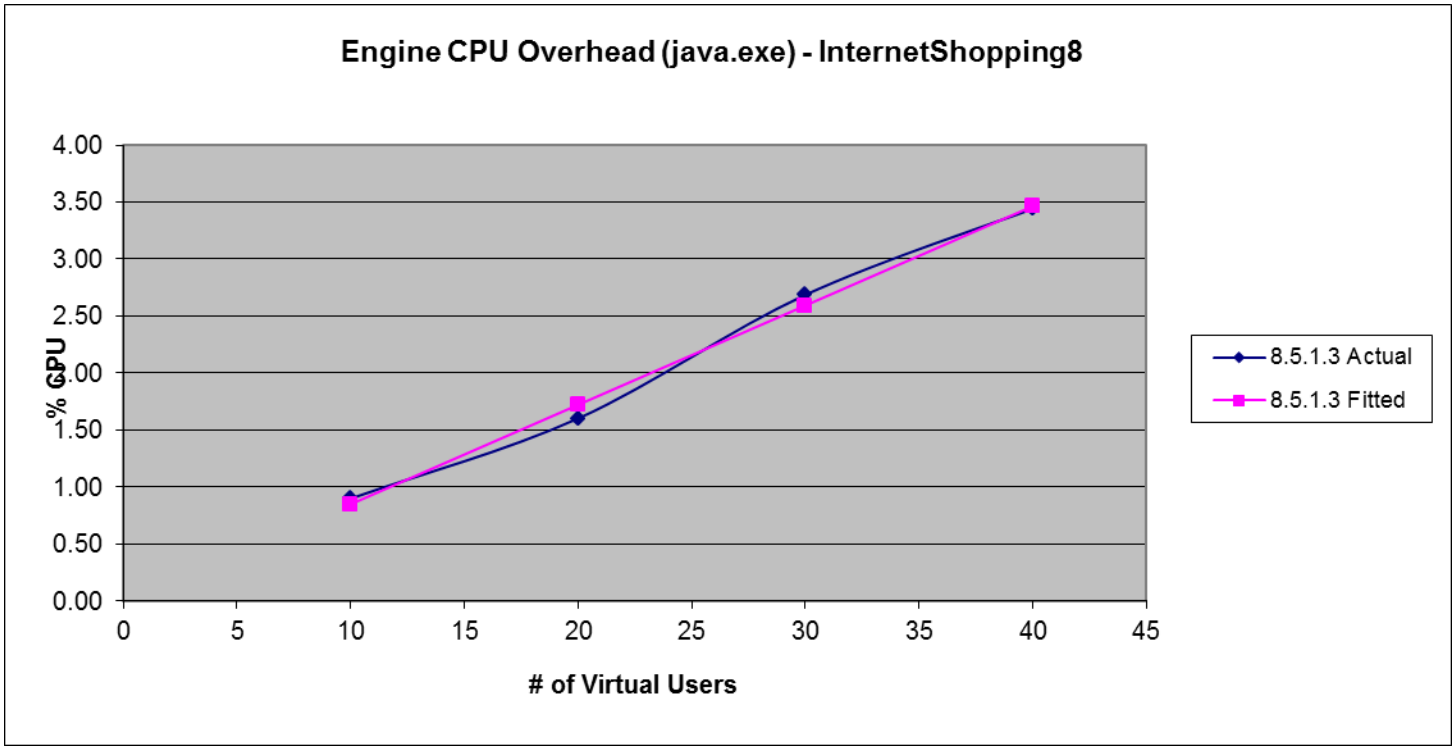
- Geometric Mean: A 3.3 X larger memory footprint with SSL connections; however the SSL overhead penalty varies by workload.
- Relative impact is proportionally larger for smaller non-SSL memory footprints.

Virtual user memory footprint: Enabling page cache



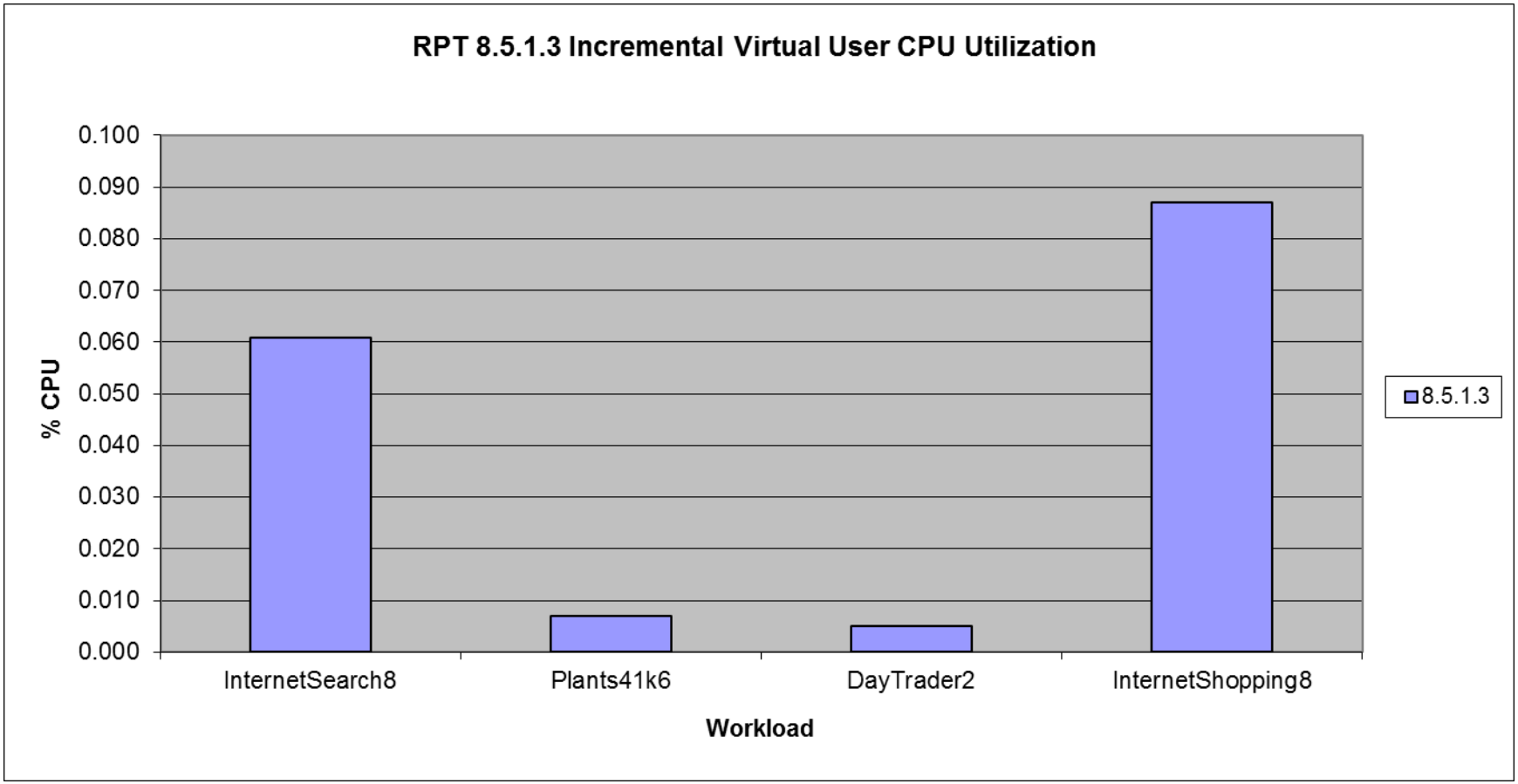
- Geometric Mean: A 1.33 X larger memory footprint with page cache enabled; however the impact varies considerably by workload.
- The extra memory is due to the number & size of references and content verification points associated with cached responses.

Using slope to calculate Incremental Virtual User CPU Utilization

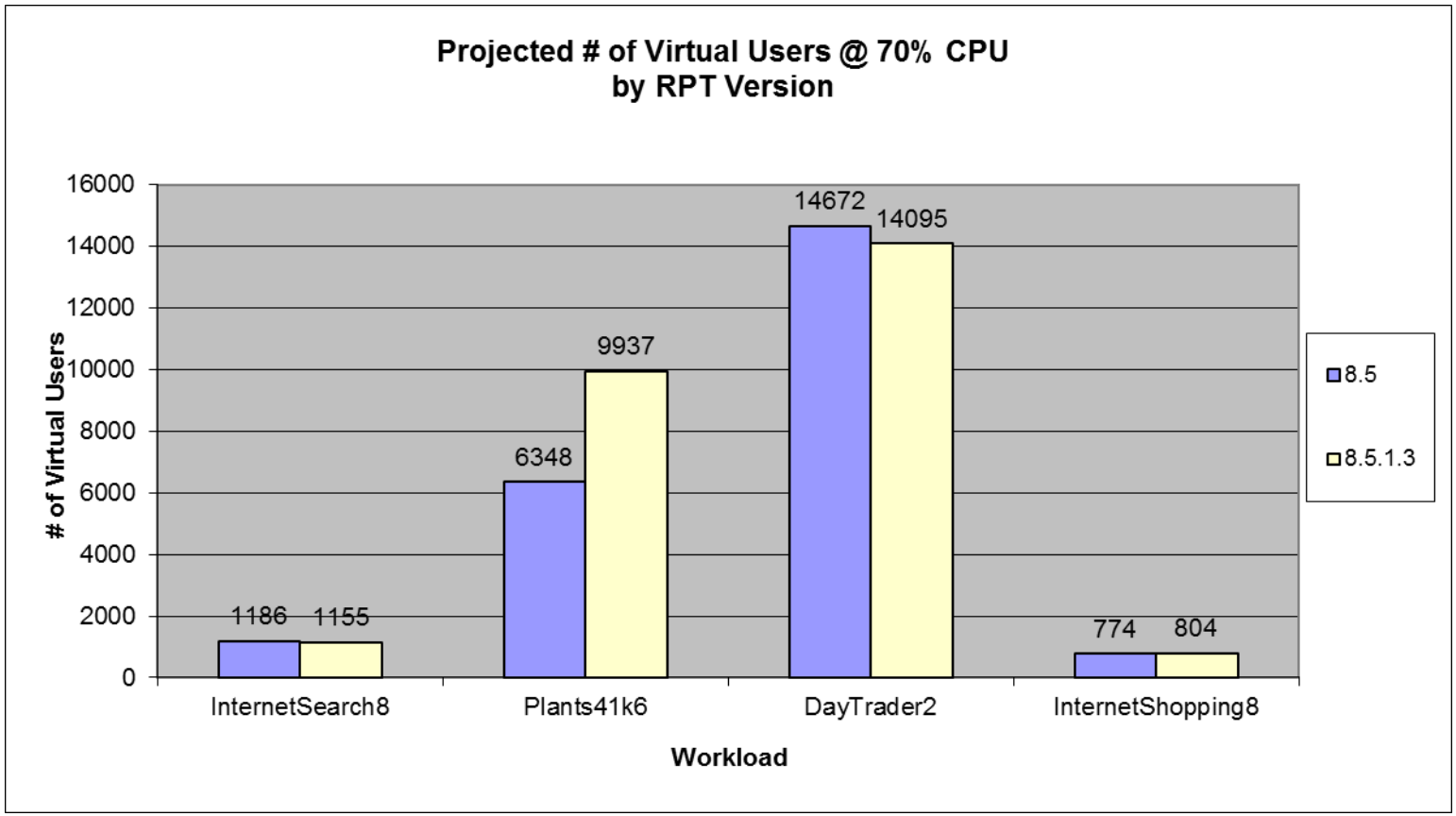


# virtual users	% CPU		% Delta
	8.5.1.3 Actual	8.5.1.3 Fitted	
10	0.90	0.9	5.7%
20	1.60	1.7	-6.9%
30	2.69	2.6	3.6%
40	3.44	3.5	-0.6%
Intercept	0.0	0.0	
Slope	0.087	0.087	

Sample CPU utilization by workload (v8.5.1.3)



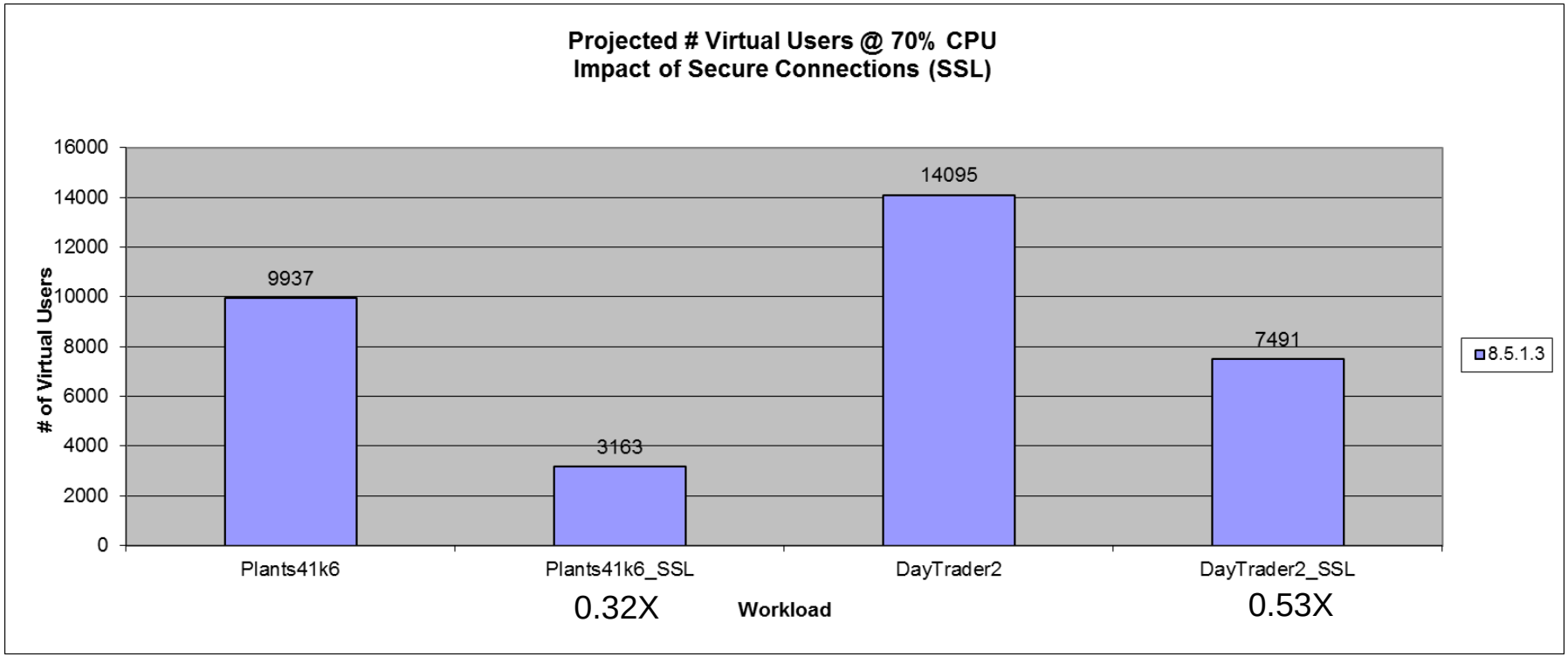
CPU sizing comparison with previous releases



On average* for these four workloads, Rational Performance Tester 8.5.1.3 capacity is 11% more than version 8.5.

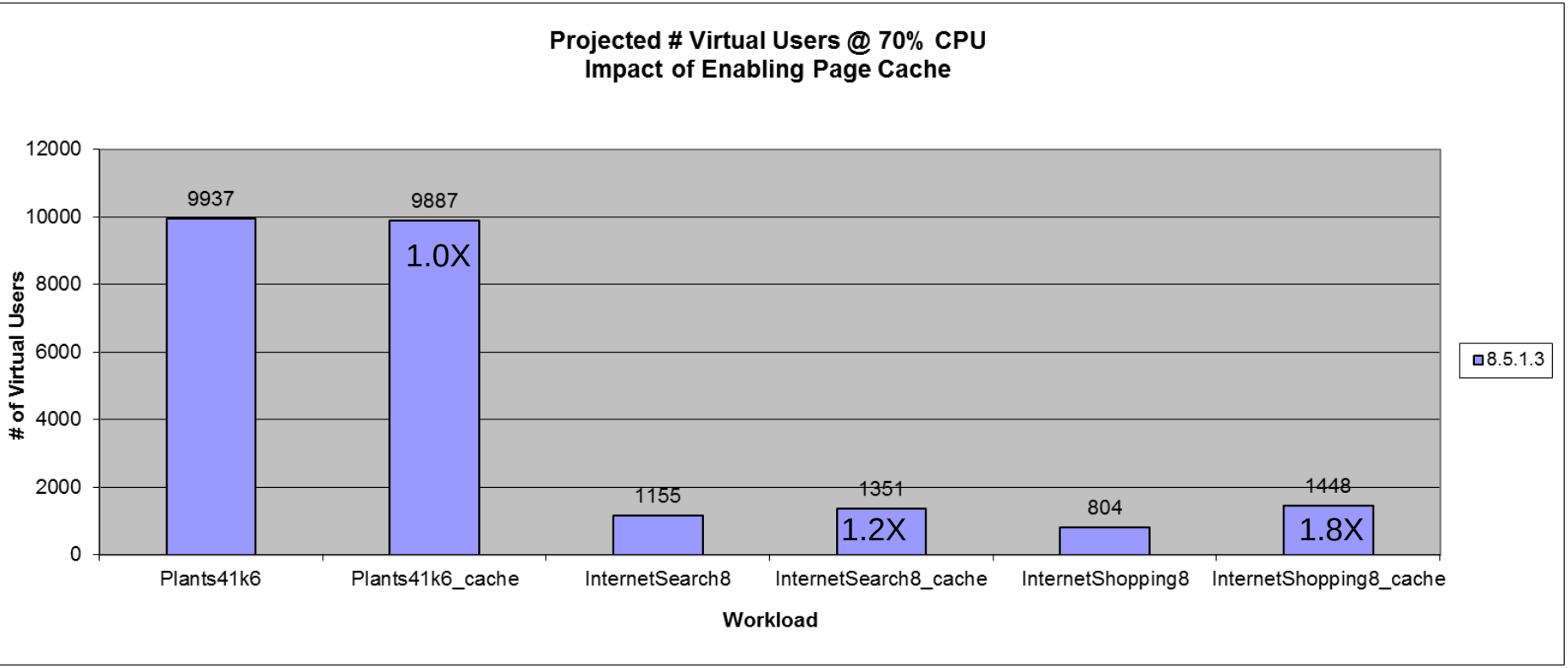
* geometric mean

CPU capacity: SSL



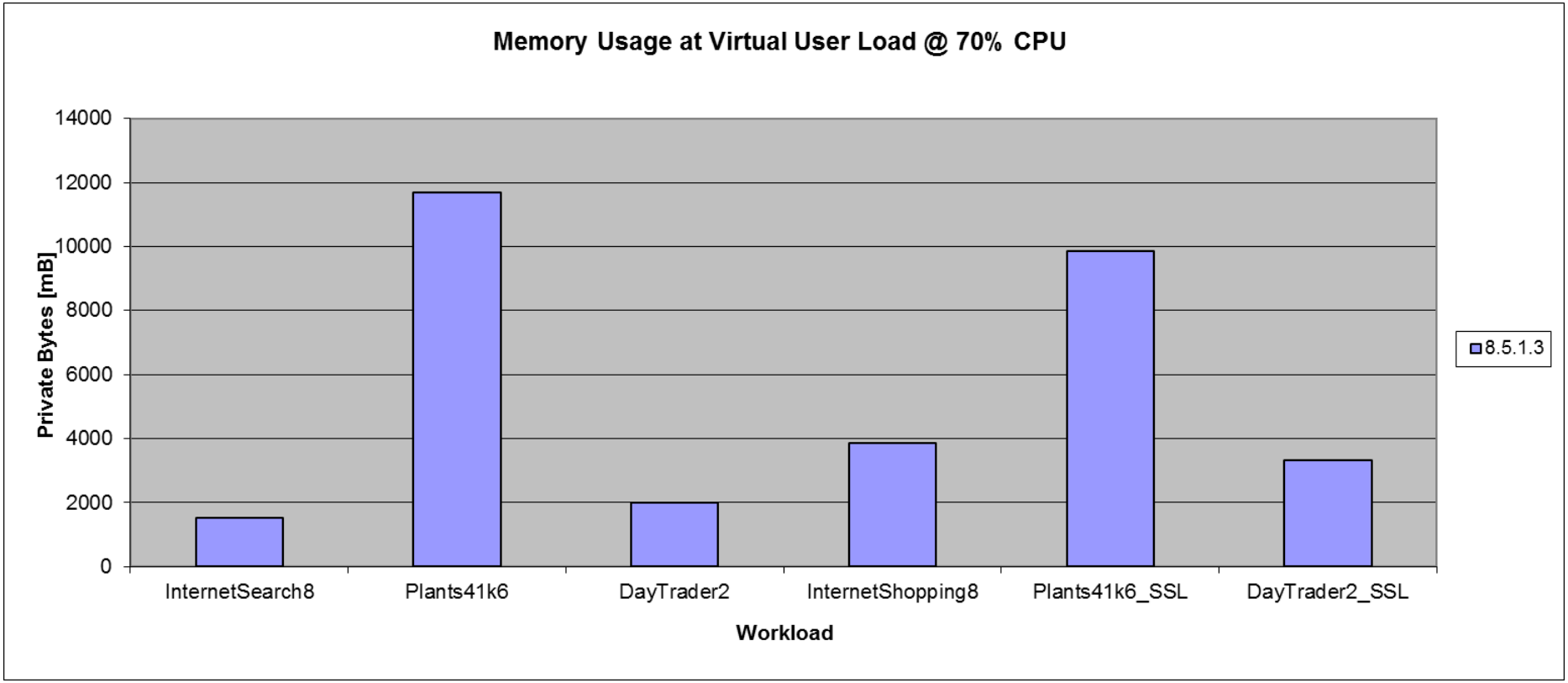
- Geometric Mean: The SSL virtual user capacity is 0.41 X of non-SSL (equivalently, SSL imposed a 2.4 X larger processor footprint)
- The SSL overhead penalty will vary by workload and is proportionally larger for smaller non-SSL processor footprints.

CPU capacity: Enabling page cache



- Geometric Mean: The virtual user capacity is 1.3 X larger when enabling the page cache; however the impact varies considerably by workload.
- The reduced processor overhead is due to fresh-in-cache requests that aren't submitted to the server, and additional savings due to not having to (re)process cached responses that are confirmed by the server, including the regular expression overhead for locating references used for data correlation.

Primary Rational Performance Tester computer sizing factor is normally CPU, not Memory



For most workloads and typical computer configurations, Rational Performance Tester has plenty of spare memory when it hits the recommended processor limit of 70%, although faster more recent vintage systems – such as the Driver "F" used for the data shown above – may need to be configured with multiple engines. This is true over a wide range of memory profiles, as workloads that have larger memory footprints also tend to have more processor overhead and vice versa.

Caveat: Powerful desktops, which are typically configured with less memory than servers, may be memory constrained for some workloads.

Summary driver sizing guidance for non-SSL workloads

- Typical driver memory footprint ranges:
  - 100 – 300KB/user – small pages, fast response times, light activity, medium to high dispersion
  - 400KB – 1.5 MB/user – medium to large pages, fast to medium response times, light to medium activity, medium to high dispersion
  - 2 – 6 MB/user – large to very large pages, slow to very slow response times, high to extreme activity (zero think times), low dispersion (lock-step)
- If you have *no access to application recordings or workload,* use these guidelines:
  - You are mostly “flying blind” here, so consider using initial estimates of 1.5 MB per virtual user, and 1000 users per DualProcessorServer, and having the ability to add additional computers if needed.
  - If all hardware has to be procured up front and there is very low tolerance for coming up short, consider 2.5 MB per virtual user and 400 users per DualProcessorServer as a very conservative approach.
- Equivalent SSL workloads might require 2 - 5 X more memory and 2 – 3 X more processor capacity.

Legal notices

U.S. Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: ***INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.*** *Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.*

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

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

Intellectual Property Dept. for Rational Software
IBM Corporation
5 Technology Park Drive
Westford, MA 01886
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Legal notices

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at www.ibm.com/legal/copytrade.shtml.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.