Oracle Primavera Compression Server 5.0 Service Pack 1
Concept and Performance Results

1 Business Problem

The current Oracle Primavera P6 Project Management application is a fat client, with most of its required data loaded up-front. The price of loading data is paid mostly during login and when opening projects. This approach has a number of advantages:

- A responsive GUI interface for most purposes,
- An ability to perform complex calculations on the client machine. For example, scheduling, leveling, and applying actuals.

However, there is a downside to this "greedy" approach. Drawbacks include:

- High memory and CPU requirements on the client machine.
- Serious degradation in data load times on a WAN especially one with *low bandwidth* and *high latency*.

The compression server solves the WAN data load problem. There has been and continues to be efforts in reducing the up-front data needs on the client. In spite of these efforts, we cannot entirely avoid the need for large amounts of data on the client. In essence, the compression server is a layer between the client and the database that compresses data before sending to the client over the WAN.

A compression server has a number of advantages:

- It minimizes the number data packets sent to the client.
- It minimizes the amount of handshaking-related traffic that normally happens between database drivers like DBXPress or BDE and the database server.
- It is scalable since multiple compression servers can be run on different machines for the same database.

2 Compression Server vs. Citrix

One solution for slow WANs currently proposed for Project Management client users is Citrix. The following is a comparison between Citrix and the compression server.

Compression vs. Citrix			
Feature	**Citrix**	**Compression**	**Comments**
Advantage Citrix			
Low End Clients	Yes	No	
Multi-Platform clients	Yes	No	
Latency impact for Login/Open projects	No	Yes	In a high latency network the data will still have to be moved from the compression server to the client.
Proven Technology	Yes	No	
Central Administration	Yes	Server side	
Advantage Compression			
Cost	$250 per seat	Free to user	Cost may need to be updated
Snappiness: switching screens etc. in application	Degrades with latency	No degradation after login	The cutoff latency when Citrix is unusable is unclear.
Memory requirements on server	150MB/client	10MB/ active client. 64Kb/ passive client	Compression server has predictable load; in Citrix it depends on the data loaded into the client.
Server CPU	Unpredictable: Depends on actions of multiple users	Predictable	The peak CPU load for the compression server is a direct function of concurrent clients.
Setup and admin	Training required	Simple	For customers unaccustomed to Citrix there is a learning curve

3 Overview

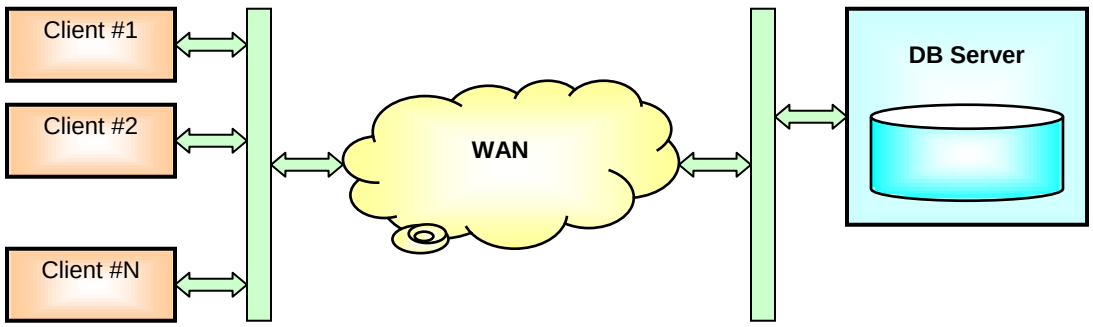


Figure 1. Classic Architecture

Figure 2 illustrates how the compression server fits into the Primavera Project Management architecture.

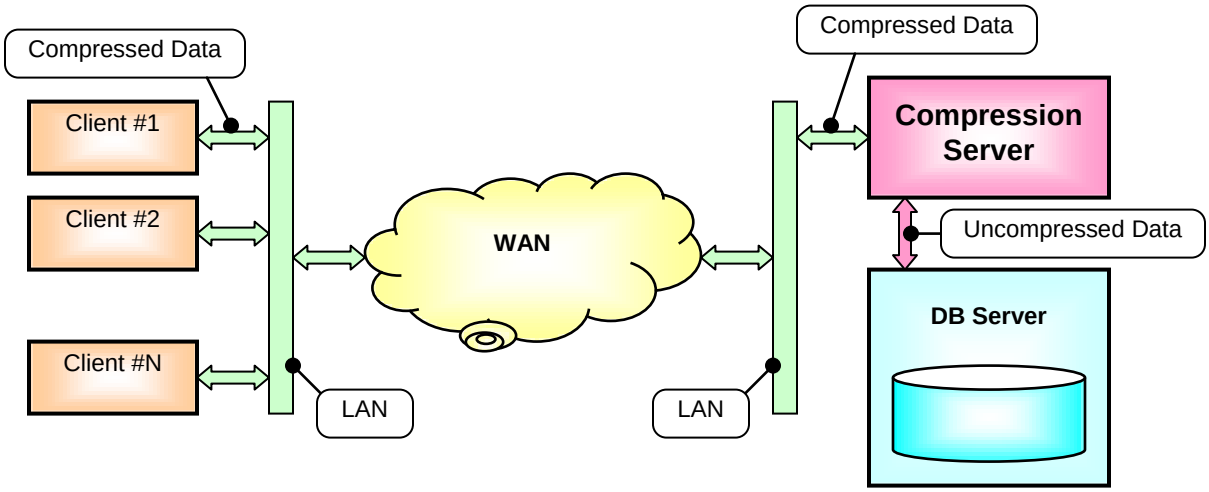


Figure 2. With Compression Server

In the current architecture (Figure 1), clients 1 to N interact with the database server over a WAN. With a compression server, the clients still get and send data over the WAN. But the data goes through the compression server either way (Figure 2). Data from the database server is first compressed on the compression server and then sent across the WAN to clients.

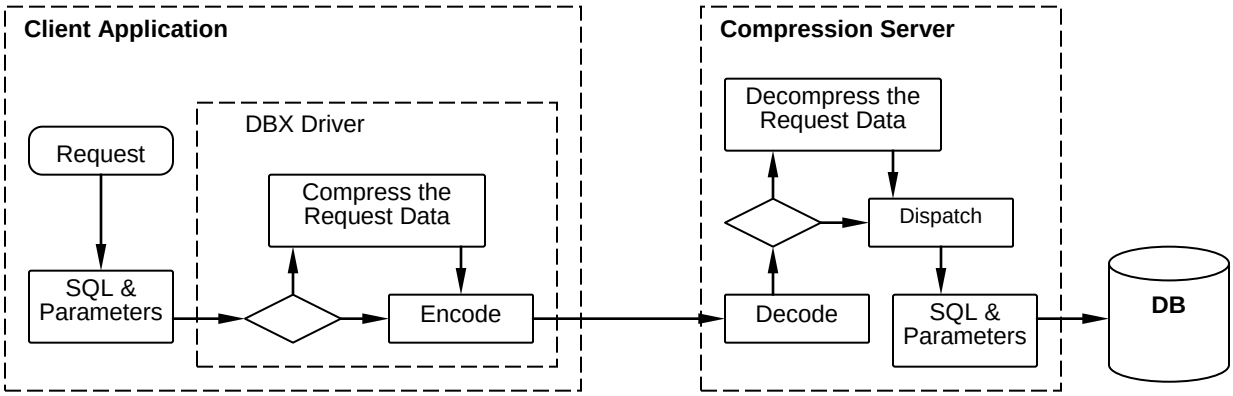


Figure 3. Request Data Flow

A typical scenario for the flow of data can be described as follows (see Figure 3): The user logs into the app or opens a project. This generates a sequence of SQL queries. (Although the client can compress data before sending it to the compression server, we do not compress most SQL statements unless the size of SQL text and parameters exceeds 880 bytes). The data is sent using HTTP. The compression server receives these SQL queries. The compression server behaves as a proxy for the client, and runs the SQL statement on behalf of the client. It receives the result set from the database. If the result set is over a configurable threshold, it compresses the response data (see Figure 4).

The compressed data is wrapped in HTTP and sent across the WAN back to the client. The client decodes and decompresses the data as required. The server does not wait until the entire result set is obtained from the database. Rather, the data is compressed into blocks of a preset size and sent to the client, even as the compression server is fetching additional rows of the same result set. This keeps the memory footprint on the compression server to a minimum, since it does not have to compile the entire result set into a huge block of compressed data, and also prevents the client from starving while the compression server compiles a large result set.

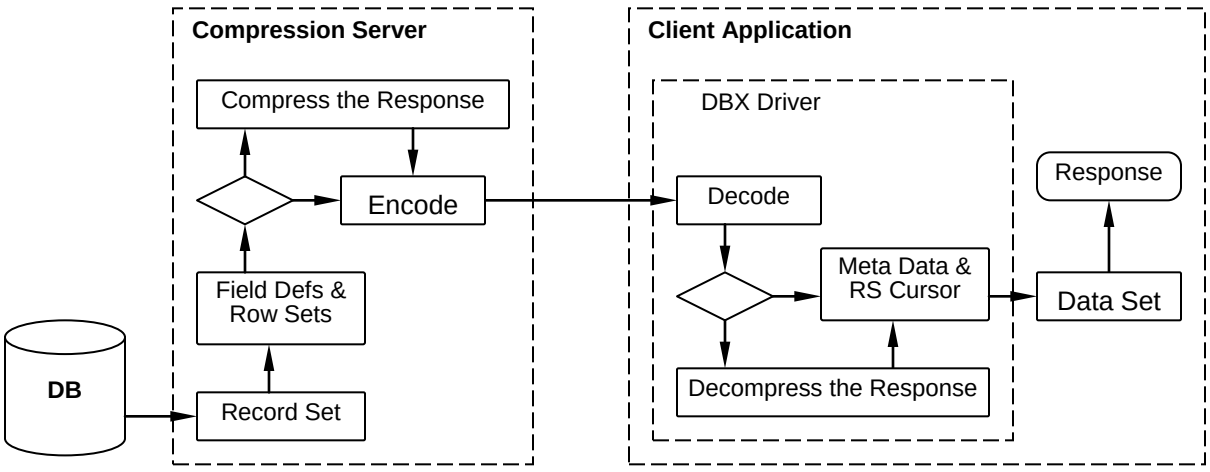


Figure 4. Response Data Flow

4 Batch SQL

In order to reduce the network traffic the communication protocol between the client and the compression server supports packing of multiple SQL requests and dataset responses. Using this Batch SQL feature the client can minimize the effects of the latency and achieve better compression ratio. When in batch mode, Primavera's DBExpress driver (PrCSDrv) for the compression server compiles multiple SQL statements and parameters together into a single compressed request and decompiles the compressed responses into messages, cursors and output parameters.

5 Basic Architecture

Figure 5 illustrates more details behind the client/compression server architecture. The main Primavera Project Management application will read and write data through Borland's DBExpress technology. A DBExpress driver provided by Primavera, which communicates with the compression server, will do the actual work of fetching and sending requests and response data. This indicates why there is no significant change in the Project Management client (the batch SQL option was one change in the client code to accommodate the compression server.) Instead of a DBExpress or BDE driver connecting to Oracle or SQL Server, the driver connects to the compression server.

For each client request, a worker thread will perform the necessary work of creating a database connection running the query, fetching the dataset, and compressing it before returning the data back to the client.

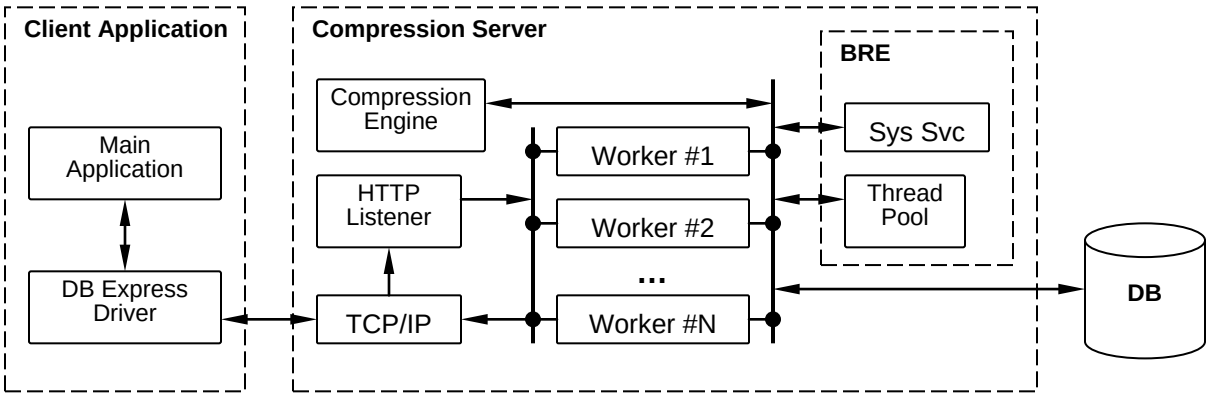


Figure 5. Design Detail

6 Brief Configuration Details

For this release, the Primavera compression server only supports Oracle and works only for the Project Management (PM) module and its connections with the PM database. Connections to the Methodology Management (MM) databases for example are direct and cannot use the compression server. Also, some of the features or tools that come with the PM module such as Schedule Analyzer or Update Baselines use the Primavera API. Since the Primavera API is not configured to use the compression server, you might see performance issues with these tools if you are connecting over high-latency lines.

The server-side dbconfig utility has provisions for connecting the client to a compression server on a specific port through Primavera's DBExpress driver (PrCSDrv). The same client can connect directly with the database if desired. The compression server has a number of configuration options as well, including the listener port number, number of worker threads, logging, thread pool, etc. These options are explained further in the *Compression Server Admin Guide*.

The client-side dbconfig utility which usually connects to Oracle, SQL Server, or MSDE will have an additional option of connecting to Primavera compression server. At this point specifics such as server name and port number will have to be entered into the client-side dbconfig utility.

7 Testing

A number of tests were conducted with the PM client against the compression server. At present the compression server can run on Windows 2000 Server or Windows 2003 Server operating systems.

7.1 Test Setup

PM clients are connected through a high-speed LAN (100 Mbit/s) to the compression server. We used Shunra Cloud™ on each client in order to simulate different network conditions. The compression server is connected *directly* with the database server without any network layer in between.[1] In order to do so, we use a Windows PC with two network interface cards (NICs): one connects the client machines to the compression server and the other connects the compression server directly to the database server. This 2-NIC configuration is detailed in the *Oracle Primavera Compression Server Admin Guide*.

[1] This was done so as to preclude any network related issues between the Compression Server and the Database server. This setup would be impractical in production, but we recommend that the connection between the Compression server and Database server to be the fastest possible to improve performance.

We test the performance of logging in and opening projects via the PM client using 1 client and 5 concurrent clients. For larger concurrent access, we use a tool called **SQLPlayer** that emulates the SQL load generated by the Project Management application during login. Up to 3 iterations are performed for each configuration (Number of clients, Network conditions) except for the Slow WAN case where one iteration was performed.

The machines we used had the following configurations:

Client machine: Intel P4 2.8 GHz, 2GB RAM

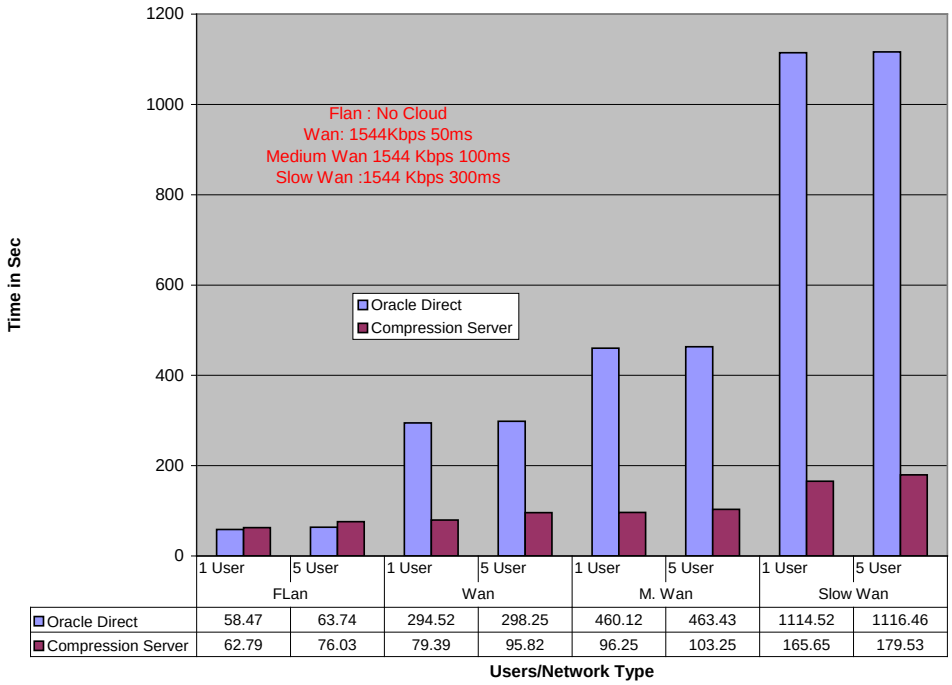
Compression Server: 2 CPUs Intel Xeon 3.2 GHz, 4 GB RAM

Database server: 2 CPUs 3.2 GHz, 8GB RAM

We tested performance with two sets of data:

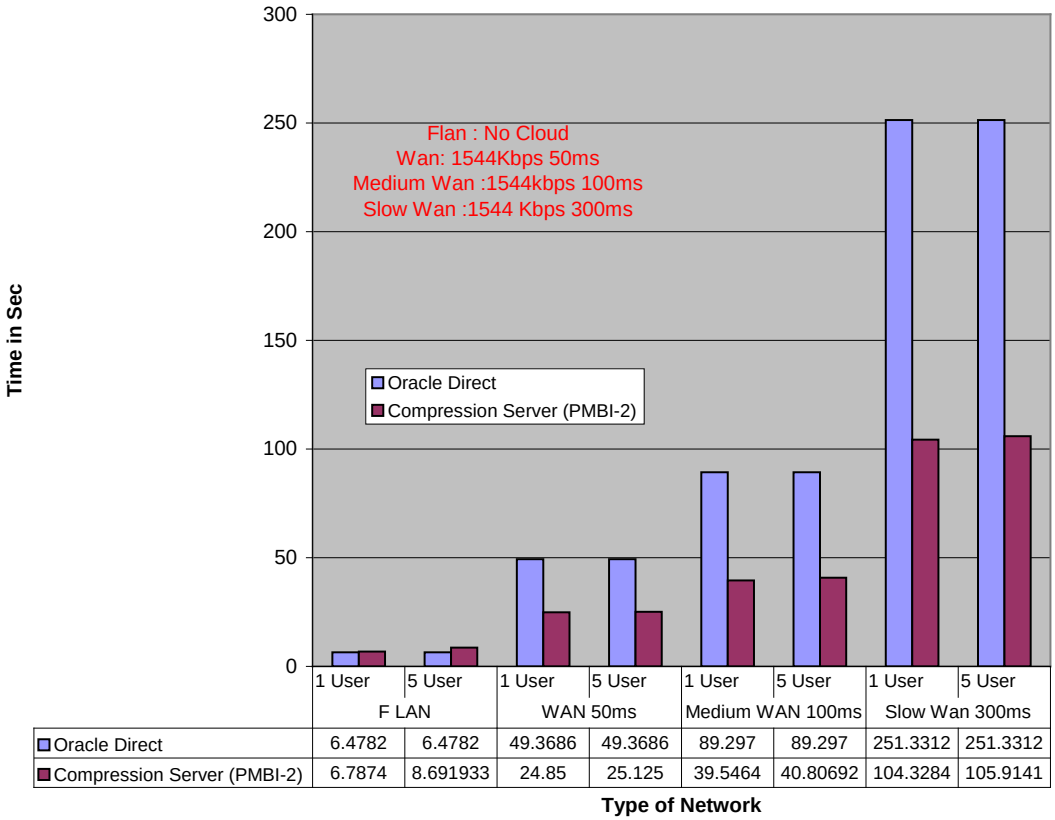
- A Large Database consisting of 16000 projects and 37000 resources. The project that we open for our tests on this database contains 6000 activities.
- A Small Database consisting of 286 projects and 23000 resources. The project that we open contains 1000 activities.

We tested a number of network conditions: high speed LAN (100 Mbps and 0 ms latency), WAN (1544 Kbps and 50 ms latency), a medium WAN (1544 Kbps and 100 ms latency) and a slow WAN (1544 Kbps and 300 ms latency). Charts 1 and 2 show login times with a large and small database for different network configurations:



	FLan 1 User	FLan 5 User	Wan 1 User	Wan 5 User	M. Wan 1 User	M. Wan 5 User	Slow Wan 1 User	Slow Wan 5 User
Oracle Direct	58.47	63.74	294.52	298.25	460.12	463.43	1114.52	1116.46
Compression Server	62.79	76.03	79.39	95.82	96.25	103.25	165.65	179.53

Chart 1: Login Performance Chart for a Large Database

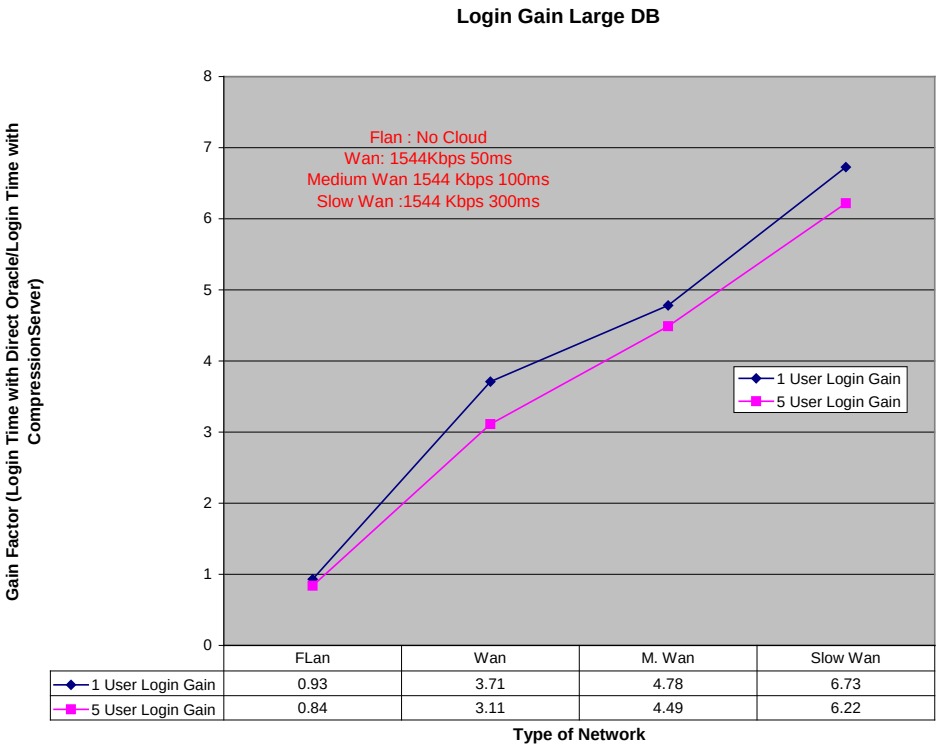


	1 User	5 User	1 User	5 User	1 User	5 User	1 User	5 User
	F LAN		WAN 50ms		Medium WAN 100ms		Slow Wan 300ms	
Oracle Direct	6.4782	6.4782	49.3686	49.3686	89.297	89.297	251.3312	251.3312
Compression Server (PMBI-2)	6.7874	8.691933	24.85	25.125	39.5464	40.80692	104.3284	105.9141

Chart 2: Login Performance Chart for a Small Database

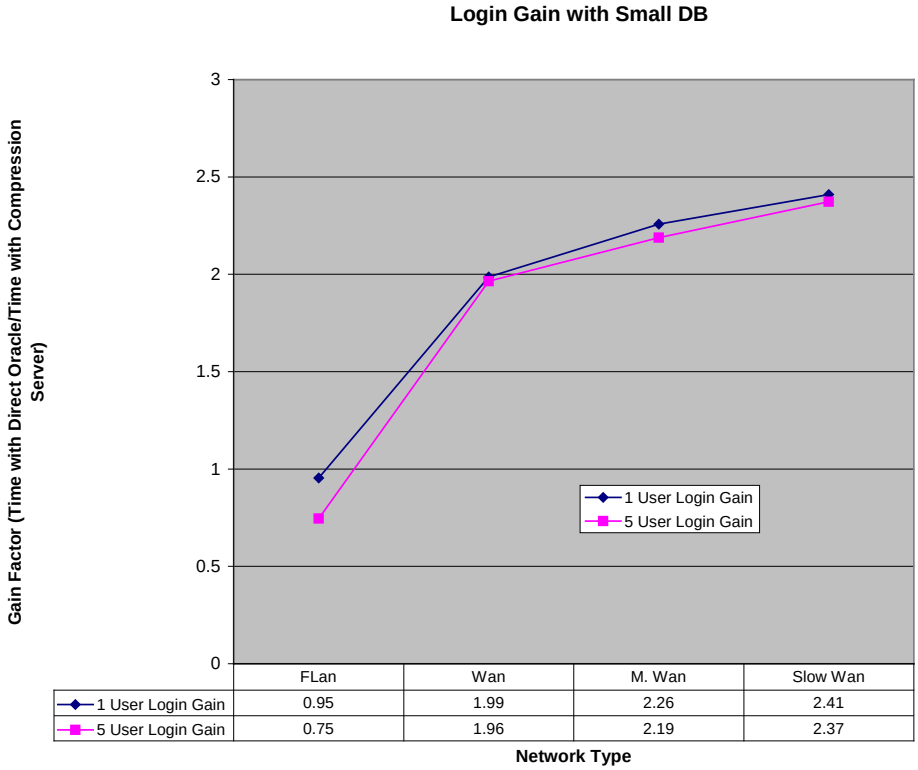
7.2 Test Results

Charts 3 and 4 show the same results above but use the concept of *Gain*. Gain is defined as the ratio between the time to perform an action using a direct Oracle connection and performing the same action using the Compression Server (every other parameter viz. number of users etc. are held constant). As is obvious with charts 1 and 2, charts 3 and 4 show that the benefits of the compression server are magnified with larger databases. For example, on a Medium WAN the compression server gets data 4.8 times faster than a 2-tier setup, but 2.3 times faster on a small Database. This is a result of the compression server being able to compress more data with larger loads. The greater the latency, the greater the benefit of the compression server on account of compression resulting in fewer packets flowing over the network.



	FLan	Wan	M. Wan	Slow Wan
1 User Login Gain	0.93	3.71	4.78	6.73
5 User Login Gain	0.84	3.11	4.49	6.22

Chart 3: Gain using the Compression server on a large DB



	FLan	Wan	M. Wan	Slow Wan
1 User Login Gain	0.95	1.99	2.26	2.41
5 User Login Gain	0.75	1.96	2.19	2.37

Chart 4: Login Gain on a small DB

The next set of charts explains how the compression server scales with larger numbers of concurrent users. Chart 5 shows the results with up to 15 concurrent users on the large database. Chart 6 shows login performance on the small database. As expected, the performance of the compression server degrades beyond 5 concurrent users except for the slow WAN. The bottlenecks in this case are two-fold:

1. The CPU load on the server box
2. The data load on the network card on the box itself.

Both sets of data were generated using the SQLPlayer application which emulates the PM client.

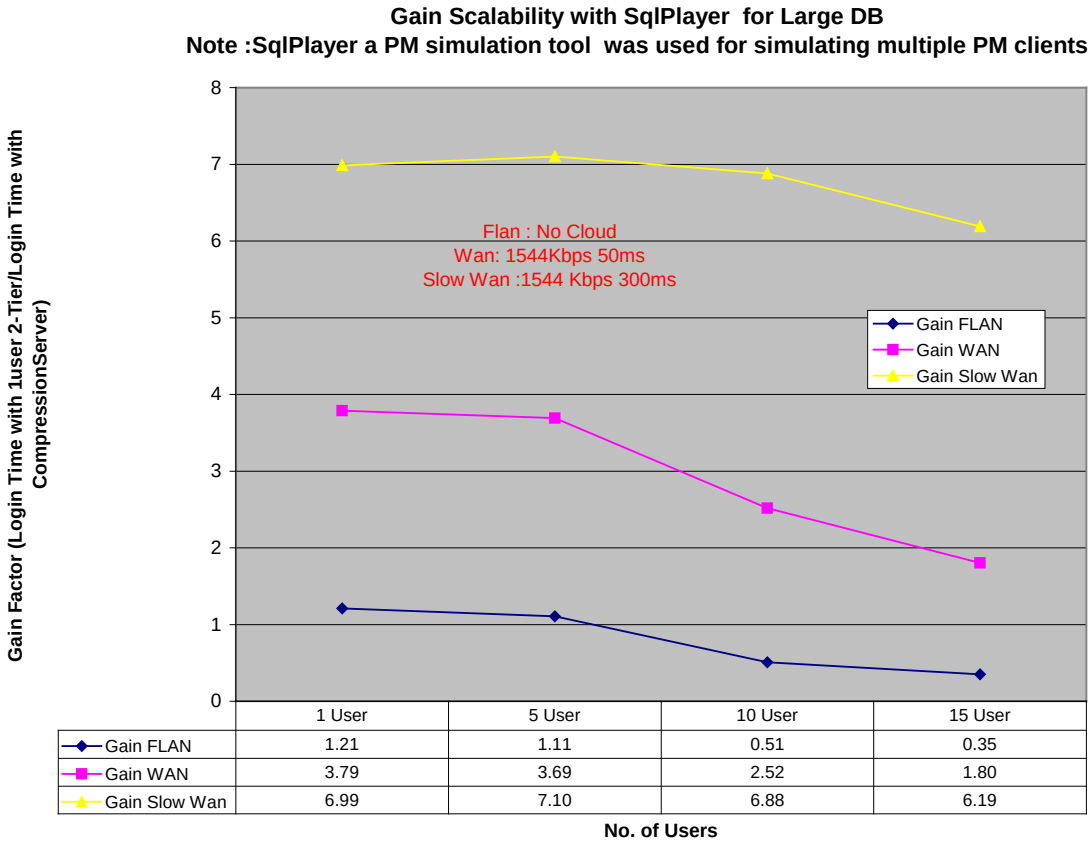
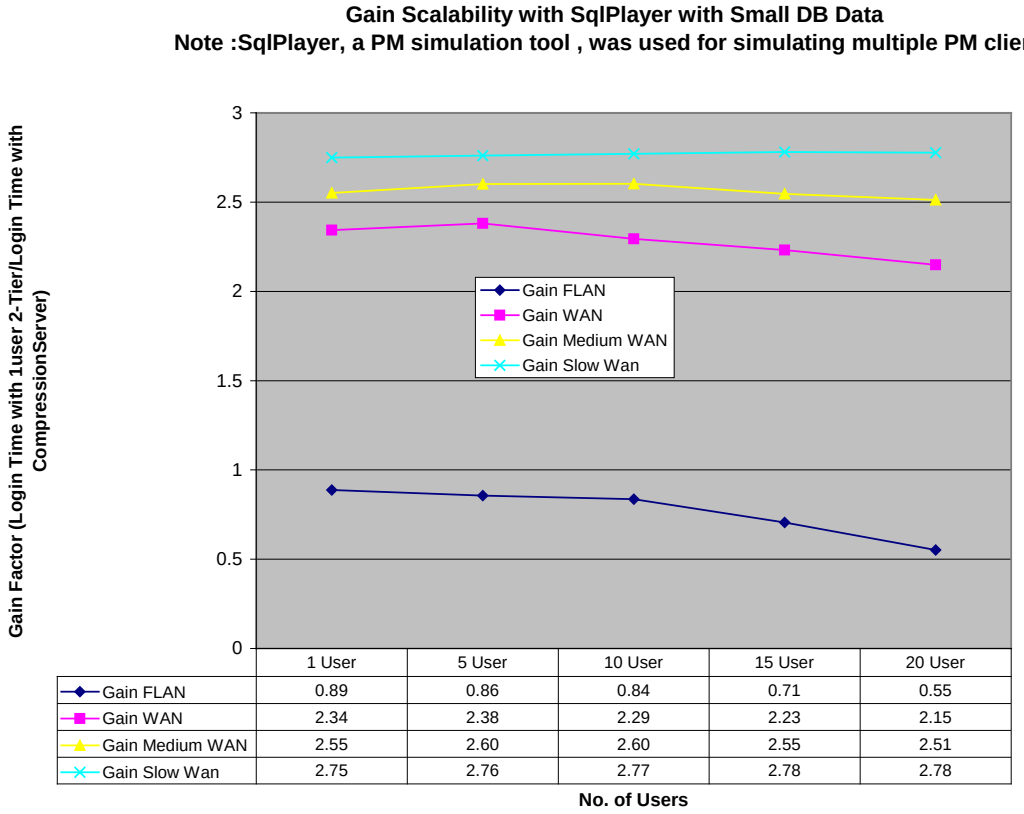


Chart 5: Scalability on Large Database with number of concurrent users



	1 User	5 User	10 User	15 User	20 User
Gain FLAN	0.89	0.86	0.84	0.71	0.55
Gain WAN	2.34	2.38	2.29	2.23	2.15
Gain Medium WAN	2.55	2.60	2.60	2.55	2.51
Gain Slow Wan	2.75	2.76	2.77	2.78	2.78

Chart 6: Scalability on a Small Database with number of concurrent users

The next set of charts shows the impact of using a single NIC on the compression server rather than two NICs as recommended in the *Admin Guide*. Chart 7 shows the performance on a large database. It is immediately obvious that performance degrades over 50% when using a single NIC from 1 to 5 users. Chart 8 shows the same behavior for a small database. The degradation in performance is not as dramatic in this case but is still evident.

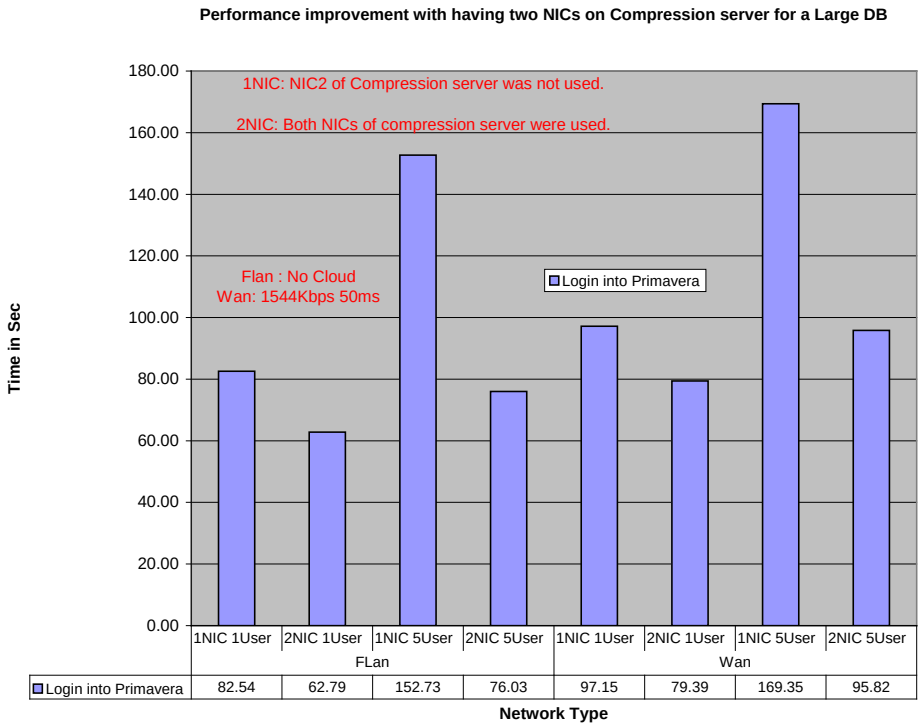


Chart 7: Performance difference between 1 and 2 NICs on the compression server for large data

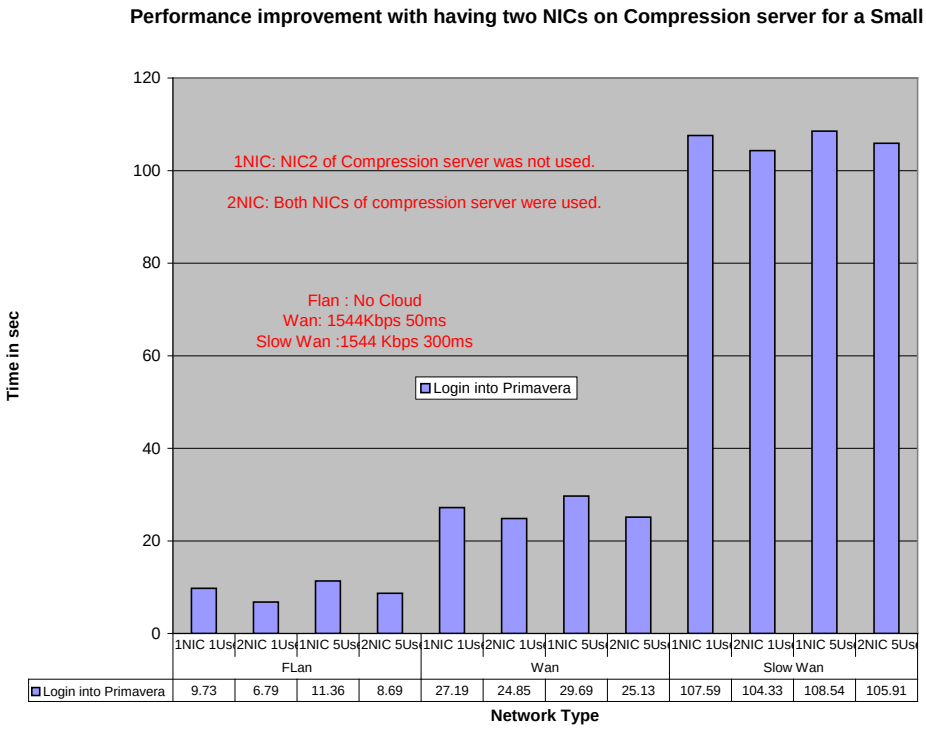
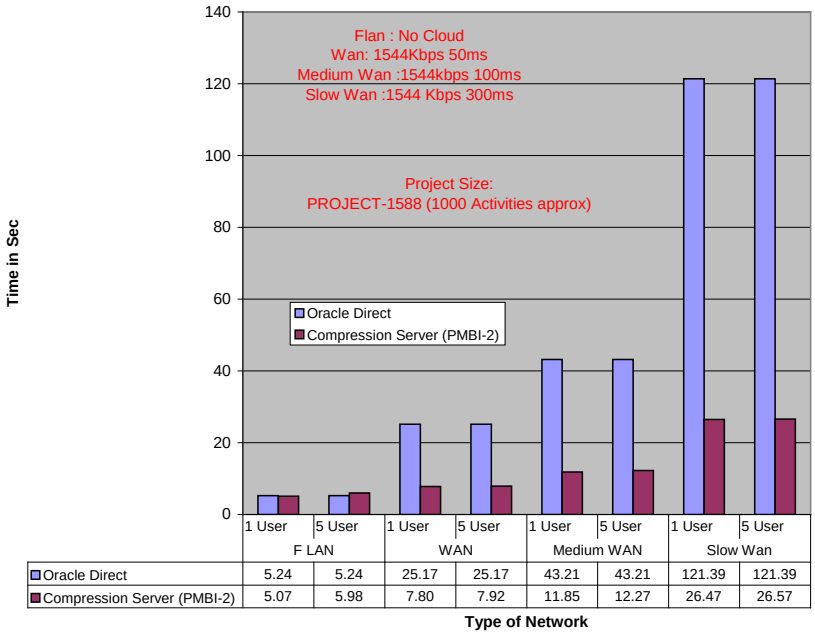


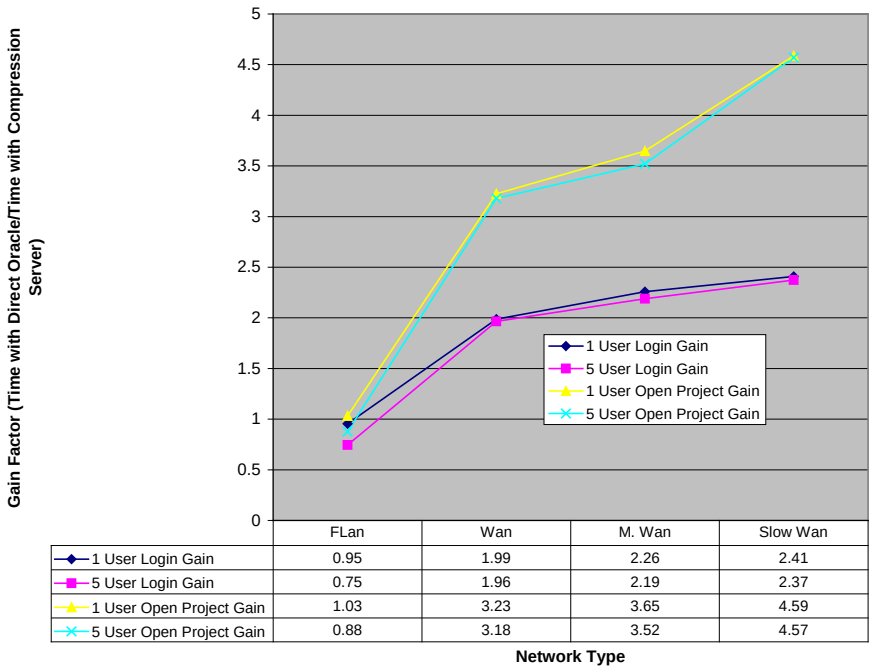
Chart 8: Performance difference between 1 and 2 NICs on the compression server for small data

The next set of charts shows Open project performance with the small database. Chart 9 shows the raw numbers comparing the compression server with a direct connection to the database. Chart 10 shows how this translates into gain. As seen here, the compression server has a greater gain for opening a project when compared to login. This indicates that on high-latency networks, using the compression server provides performance gains even on small databases.



	1 User	5 User	1 User	5 User	1 User	5 User	1 User	5 User
	F LAN		WAN		Medium WAN		Slow Wan	
Oracle Direct	5.24	5.24	25.17	25.17	43.21	43.21	121.39	121.39
Compression Server (PMBI-2)	5.07	5.98	7.80	7.92	11.85	12.27	26.47	26.57

Chart 9: Open Project performance with a Small Database



	FLan	Wan	M. Wan	Slow Wan
1 User Login Gain	0.95	1.99	2.26	2.41
5 User Login Gain	0.75	1.96	2.19	2.37
1 User Open Project Gain	1.03	3.23	3.65	4.59
5 User Open Project Gain	0.88	3.18	3.52	4.57

Chart 10: Login and Open Project gains for a small database

The next chart shows the impact of CPU on compression server performance. It can be seen that the use of a better CPU translates to better performance even for a single user. Furthermore, the difference between a single user and 4 concurrent users is more pronounced with the low-end box.

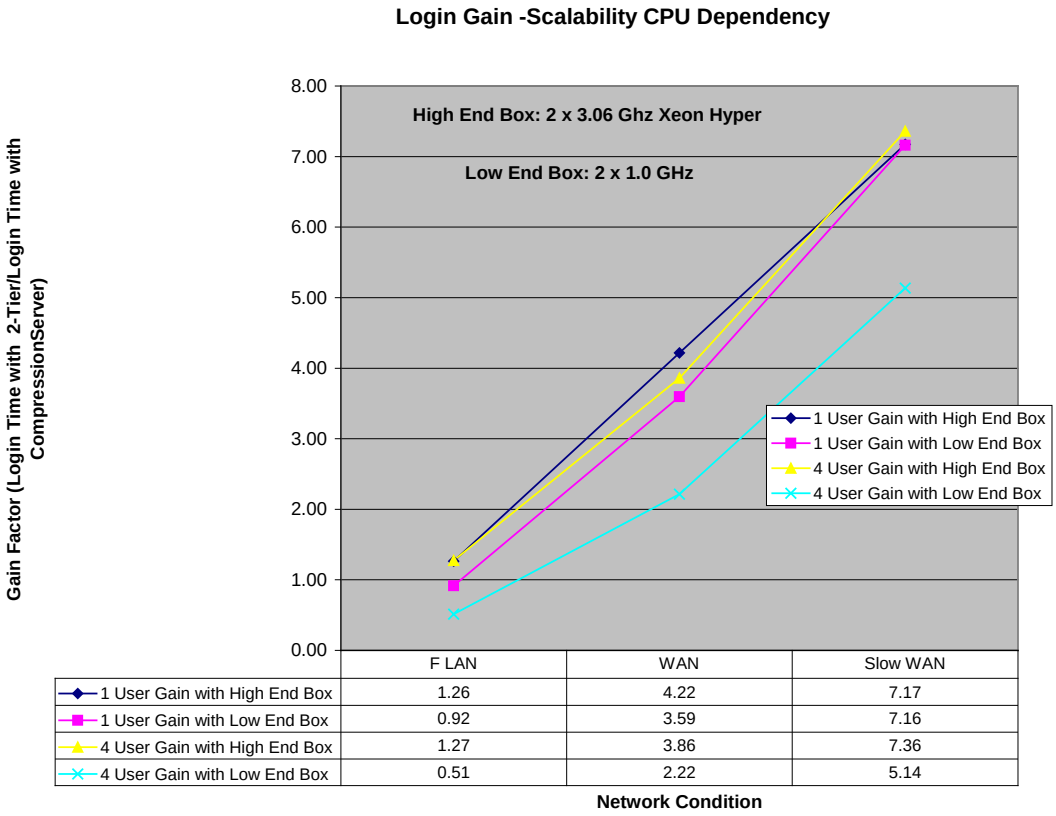


Chart 11: Compression server performance on high end and low end box

8 Conclusion

The compression server provides for faster data load for Primavera Project Management compared to the standard 2-tier connection. Its benefits in comparison to the two-tier setup are magnified on high-latency networks and with larger databases. That said, even with smaller databases there are significant gains under certain situations as our tests with Open Project showed.

Our tests show that using a faster CPU and 2 NICs on the compression server improves its performance considerably.

APPENDIX: Compression Server FAQ

Question #1: How many total concurrent sessions were simulated?

Answer #1: We simulated 200 concurrent sessions with 5 of them being simultaneously and continuously used in processing client requests. The limit of 5 clients was dictated by the CPU frequency and the amount of data to be compressed.

Question #2: What was performance like under larger loads?

Answer #2: There are two kinds of large loads for compression server.

First type addresses the case of many clients sending Web-like (small) requests, expecting small responses. In this case the compression benefits are minimal (next to none) since there is not enough data to compress. As a consequence of this, the number of requests served simultaneously is limited by the overhead involved in maintaining threads and finding sessions. For the recommended hardware, the limit in this case is between 75 and 100 requests processed simultaneously.

The second type of large load addresses the case of few clients updating or loading large amounts of data through compression server (large databases). In this case most of the CPU power is used for compressing the data. For the recommended hardware the limit is between 5 and 10 requests processed simultaneously.

Question #3: How many users can one compression server support?

Answer #3: The number of users depends on the hardware power, most used scenario and the database size. See answers #1 and #2 for details.

Question #4: Is the size of the project a consideration?

Answer #4: Yes. The size of the project directly affects the loading time during open project.

Question #5: How much RAM does each user session require?

Answer #5: The amount of RAM required by one user session is about 64KB. However, the amount of RAM used while processing requests is 10MB on average and can grow beyond 200MB when large blob data is uploaded or downloaded from the database. The 200 MB statement is true only if there are very large blobs (over 5 MB) in the data.

Question #6: Can multiple compression servers be used to support large user populations?

Answer #6: Yes. However, because of the high traffic, each compression server requires a dedicated NIC in the database server machine and therefore the number of compression servers that can be used could be limited by the Oracle database machine. One of the problems we had was not being able to support massive reads and writes on the same NIC. This could be a problem that could be solved differently by the customer; our approach was to use multiple NIC's.

Question #7: Does each compression server require a dedicated hardware platform or can it host other processes, i.e., Progress Reporter Server (formerly Group Server), P6 Web Access (formerly myPrimavera), etc?

Answer #7: Compression server, when configured and used correctly, requires more than 80% of the CPU power. It is not recommended to run any other CPU intensive applications on the same hardware with compression server.