Vercelletto.com

Informix Innovator-C Edition pushed to limits ?

Case study for a TPC-C based Stress test realized on a Linux Box

with IBM Informix Innovator-C Edition

Combrit, March 16, 2012

Informix Innovator-C Edition pushed to limits ?

When such a massive wave comes on you, the only possibility to survive is to trust your own resources and go for it! I know pretty well this feeling because I lived kind of similar situations (at my scale, divide the size of the wave by three at least!) and this is true: you feel very alone in this moment and the only friends you can rely on are yourself, your mental, your energy and your rib cage capacity.

I have also lived this type of situation in my professional life, when customers came to me asking to resolve the global application response times, with hundreds or more users virtually at the door, and the CTO in front of me putting a threatening angry face.

This can happen anytime when an application is deployed, and it is not nice to see the threatening wave come onto you if you do not trust your infrastructure. This is when you think "why the heck didn't I run the relevant stress test before deploying?"

I already posted a reflection about the choice of the right Gnu Public License RBDMS in this article, based on the assumption that if you don't know how your RDBMS, or worse if you know that it cannot escalate properly, you have probably done the wrong choice if the users load increases! Recent news are illustrating this fact. A number of software editors (including Cisco Systems and the SugarCRM and more), have been recently considering adding Informix in the supported platforms due to weak or insufficient performance and stability provided by other RDBMS editors. In technical forums, questions about migrating this or that RDBMS implementation to Informix now appear more frequently.

Many people running IBM Informix are convinced that the product is very good at escalating, but beyond the fact that those same people have never seen CPUs burning on a stressed Informix server, no realistic figures have been published for a number of years. What was the reason for such a silence? Was this due to competitors in charge of "burying" Informix yet alive? Was this due to a real lack of performance in front of the competitors? Was this due to a general indifference of the marketplace? In fact, I didn't care much. I was just willing to make a clear idea on how many terminal sessions of an average OLTP application the free of charge bottom range version of IBM Informix could stand.

TPC-C or not TPC-C ?

That is the question! Everyone knows the TPC Council in charge of publishing the specifications, scenario as well as the results of the longtime appraised DBMS benchmark standard. This non-profit organism has been created in 1988, and performs different types of DBMS benchmarks in order to set a ranking according to "as close to real life as possible" criteria.

Current TPCs are today TPC-C, TPC-E and TPC-H. TPC-C is as of today the most representative benchmark for the OLTP activity. A number of open-source TPC-C runners exist on the net, but most of them written in Java. I found a runner in a Spanish university, whose program director kindly accepted to let me adapt his team's development to run against IBM Informix.

Objectives of the test.

Considering I do not have the official copy of TPC-C delivered by the TPC Council, and also considering that my company is not a member of the TPC Council, this test cannot be validated as an official TPC-C benchmark. Nevertheless, it is guaranteed that all the rules of the TPC-C benchmark are respected. In consequence, we are going to run a STRESS TEST against Informix Innovator-C to understand how many terminal sessions (i-e tpc-c users sessions running in a transaction monitor) can run concurrently on a single Informix server.

Preliminary steps and selected configuration

Having found a source code that I could be adapted for a minimum cost, the field was open to start the operation. Here are the steps necessary to prepare to benchmark.

1) Install a server based on Linux (Fedora 14, kernel 2.6.35.14 x86_64), with the following specs:
   * 1 socket Intel Quad Q9400 (4x 2.66Ghz)
   * 16 Gb of DDR2 RAM
   * 4 X 500 Gb SATA II, 7200 rpm disk drives
   This configuration cost doesn't reach 900 Euros

2) Install and configure IBM Informix Innovator-C Edition on this Linux server.
   Since Innovator-C Edition is the free edition of Informix, the cost is zero Euro.
   Chosen version is 11.70 FC4.

3) Adapt the TPC-C application from the Valladolid University, Spain, to run against IBM Informix, initially developed in Esql/C for PostgreSql. This represented a relatively light task, main modifications being PosgreSql against Informix server mecanisms alterations. I also optimized the initial database creation statements, taking benefit of RAW tables and prepared statements. After all, this benchmark does not measure database load statements , so the less time to load, the better.

4) Compile and debug the application

5) Tune Informix

6) Run the test, stress more and more until the system performance decreases, and respecting TPC-C rules, ensure that the test passes.

The benchmark rules

Obviously, and even if this test is not an official one, it is necessary to respect a number of rules, which are the following:

1) Do not modify the database schema. The TPC-C database contains 9 tables, each table has a defined structure, identified indexes and integrity constraints. It is not allowed to modify, add or drop any of those.

2) Do not modify tables cardinality. There are strict rules about tables cardinality, for instance one warehouse will host 100.000 items, one district will contain 3.000 customers etc…

3) Do not modify the transactions application code. TPC-C uses 5 different transactions which are designed to reflect a typical average OLTP application. Those transactions are New Order, Payment,Delivery, Order Status and Stock Level. The last one features a select count(distinct) and where clauses on non-indexed columns, so that the database engine is slightly put under pressure.

4) Each of the transactions category has a maximum admissible response time. For each transaction class, a minimum of 90% of these transactions must execute below. In the negative, the test is considered as failed.

5) Each test has a ramp-up or warm-up time, and a measure time. The ramp-up time serves for the server to accommodate the increasing load, so that the measure time can be executed while the performance is stabilized. The measure time is, you guessed it, the interval during which the performance is measured.

6) The checkpoints interval has no precise rules, except that at least one must be executed during the measure time. In Informix Innovator-C case, we don't care that much because the fuzzy checkpoints does not block anymore the transactions (or at least extremely few). During the test, I chose an interval of 15 minutes, which is still realistic for a real life system.

7) Disk implementation: no rule. I had 4 SATA II 7200 rpm disks, all on the same SATA controller, and I used them all, trying to balance as accurately as possible the location of each table and index.

8) Shared memory amount: Innovator-C is limited to 2Gb for all the instances on the same machine. I set full use of these 2Gb, leaving space for SHMVIRTSIZE which is going to be required by sort and similar operations.

What will this stress test measure?

We shall use the TPC-C Benchmark developed by a students team from the Universidad de Valladolid, managed by Professor Diego Llanos, who kindly accepted to let me adapt this development for the IBM Informix platform. This is not the TPC Council official source code, but it is compliant with TPC Council requirements. So this operation is a **STRESS TEST** and not an official benchmark published by the Council. As explained above, the test will measure the response times on 5 different OLTP typical transactions. If less than 90% of each transaction type have a response time less than the acceptable limit, the test fails. For each of those transactions, we shall provide the minimum, maximum, average and 90thtile response times. We will not detail the failed tests. At the end, the test will be granted a global result expressed in tpmC-uva, meaning number of valid transactions executed per minute.

Let's go for it!

We are now skipping directly to step # 6 of the action plan. We had to determine where was the breakpoint (not the point break) where the test would fail. I was thinking about Ronan Chatain's attitude (the "small" surfer in the big wave), and, confident in Innovator-C, I decided to start with 50 warehouses / 10 terminals per warehouse, making a total of 500 user terminals!

Run # 1: First shot to set performance expectations

Number of warehouses: 50
Number of terminals per warehouse: 10
Warmup time: 30 mn
Measure time: 60 mn
Result: Passed
tpmC-uva: 590.208

Looking at the results details, we have a margin to add more warehouses, although I noticed that the "bench" binary is sticking up at 100% CPU and nothing can be done at the moment to minimize this issue. For the general vmstat output, the average was 38% user time, min 31% and max 49%. Regarding io wait, the average was 11%, min 25% and min 6%. We were close to the good guess, but still have some power margin and push the test to 55 warehouses.

Run # 2: probably the right configuration

Number of warehouses: 55
Number of terminals per warehouse: 10
Warmup time: 45 mn
Measure time: 240 mn
Result: Passed
tpmC-uva: 610.728

This shot looks to be the right one. We still passed the test, but the server is showing a decreasing trend. We gained a mere 20 tpmC for 50 more terminals and we think the next shot will not pass. We have also observed that the checkpoints are consequently affecting the io wait system counter (up to 50% wait io), and though Informix transactions are not blocked by the checkpoints, the wait_io is hardly impacting the whole system.

Run # 3: Will probably reach the inflection point

Number of warehouses: 60
Number of terminals per warehouse: 10
Warmup time: 45 mn
Measure time: 60 mn
Result: Failed
tpmC-uva: 497.567

This result was expected. We have reached the limit of what this box can handle.

Conclusion

We have successfully achieved the 55 warehouses for 10 terminals run, making a total of 550 terminal sessions, which looks quite impressive for a test hosting the application AND the database server on the same tiny box.

Not happy with this result, we have attempted a more expensive client-server configuration to check how far we could go. Against all odds, we obtained similar results. While the DB Server system monitoring was showing a mere 15-20% average user CPU with 550 terminals, we detected that the “bench” binary who drives the whole benchmark did not stand the system load over 550 terminal sessions, rendering huge wait times for a too important part of the transactions.

We doubled checked with SQLTRACE that almost no query was executing over 20 seconds response time inside the db server. Fixing this issue may be the next phase of this project, the alternative being to run the same benchmark thru the official TPC Council…

So if we consider that we have run successfully 550 TPC-C terminal sessions against a small Quad Core@2.66Ghz Linux box, with 4 cheap 7200 rpm SATA II disks and 16 Gb of DDR2 memory, with both the DB Server and the application sessions on the same box, with a total cost of less than 900 Euros including the RDBMS license, IBM Informix Innovator-C Edition is an excellent choice to start a deployment project for departmental applications. Although not calculated here, the ratio of infrastructure cost / tpmC number should be extremely competitive, and more competitive even if one counts the very low cost of administration of IBM Informix. Much more power and stability for much less money!

Again, we did not want to play in the huge infrastructure area with tons of CPU,kilotons of RAM and gigatons of disk arrays, costing US$ kilotons too. We just wanted to demonstrate the real power and scalability of the entry point of the IBM Informix database servers. Now this was on a single server. What if we used the Informix Flexible Grid features, which are also available with the Innovator-C Edition? Well! this sounds like another challenge that we won’t take today!

I really wish to thank:

Professor Diego R. Llanos from the Universidad de Valladolid, we kindly allowed me to use his work
Ronan Chatain, the surfer on the picture made in Bro Vigoudenn, who demonstrates where self confidence can lead
Erwan Crouan, the photographer, who kindly allowed me to publish his masterpiece.

Attached documents

Detailed results of the 55 warehouses – 10 terminals run

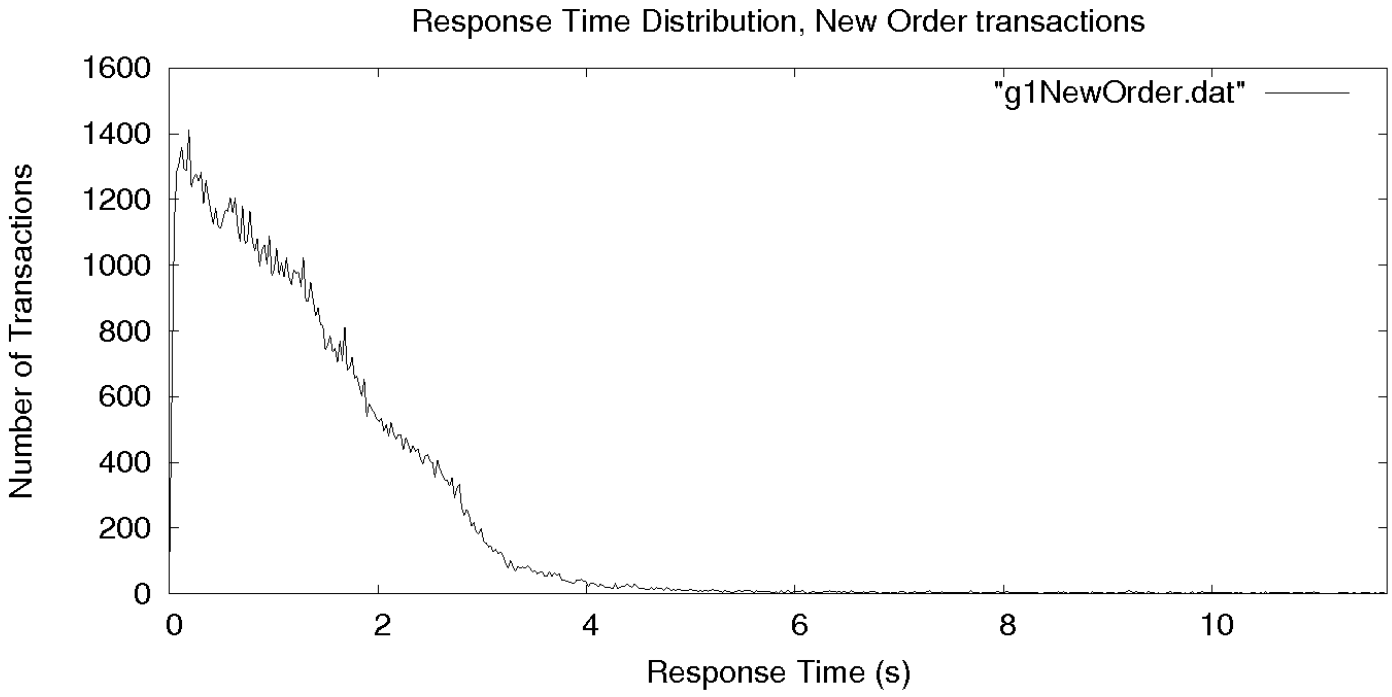
```
Test results accounting performed on 2012-02-23 at 19:17:31 using 55 warehouses.
Start of measurement interval: 45.016333 m
End of measurement interval: 225.016333 m
COMPUTED THROUGHPUT: 610.728 tpmC-uva using 55 warehouses.
252680 Transactions commited.

NEW-ORDER TRANSACTIONS:
  109931 Transactions within measurement time (130117 Total).
  Percentage: 43.506%
  Percentage of "well done" transactions: 94.221%
  Response time (min/med/max/90th): 0.008 / 3.327 / 107.466 / 2.920
  Percentage of rolled-back transactions: 0.967% .
  Average number of items per order: 14859.225 .
  Percentage of remote items: 0.001% .
  Think time (min/avg/max): 0.000 / 12.060 / 120.000

PAYMENT TRANSACTIONS:
  109824 Transactions within measurement time (130300 Total).
  Percentage: 43.464%
  Percentage of "well done" transactions: 95.213%
  Response time (min/med/max/90th): 0.001 / 2.664 / 107.702 / 2.800
  Percentage of remote transactions: 14.105% .
  Percentage of customers selected by C_ID: 39.337% .
  Think time (min/avg/max): 0.000 / 12.038 / 120.000

ORDER-STATUS TRANSACTIONS:
  10963 Transactions within measurement time (13012 Total).
  Percentage: 4.339%
  Percentage of "well done" transactions: 95.457%
  Response time (min/med/max/90th): 0.007 / 2.545 / 105.946 / 2.840
  Percentage of clients chosen by C_ID: 39.770% .
  Think time (min/avg/max): 0.000 / 10.096 / 93.000

DELIVERY TRANSACTIONS:
  10982 Transactions within measurement time (13042 Total).
  Percentage: 4.346%
  Percentage of "well done" transactions: 96.767%
  Response time (min/med/max/90th): 0.000 / 1.114 / 99.241 / 0.080
  Percentage of execution time < 80s : 99.727%
  Execution time min/avg/max: 0.023/2.518/101.781
  No. of skipped districts: 0 .
  Percentage of skipped districts: 0.000%.
  Think time (min/avg/max): 0.000 / 5.038 / 47.000

STOCK-LEVEL TRANSACTIONS:
  10980 Transactions within measurement time (13025 Total).
  Percentage: 4.345%
  Percentage of "well done" transactions: 97.304%
  Response time (min/med/max/90th): 0.003 / 2.630 / 98.372 / 2.720
  Think time (min/avg/max): 0.000 / 5.023 / 47.000

Longest checkpoints:
Start time  Elapsed time since test start (s)  Execution time (s)
```

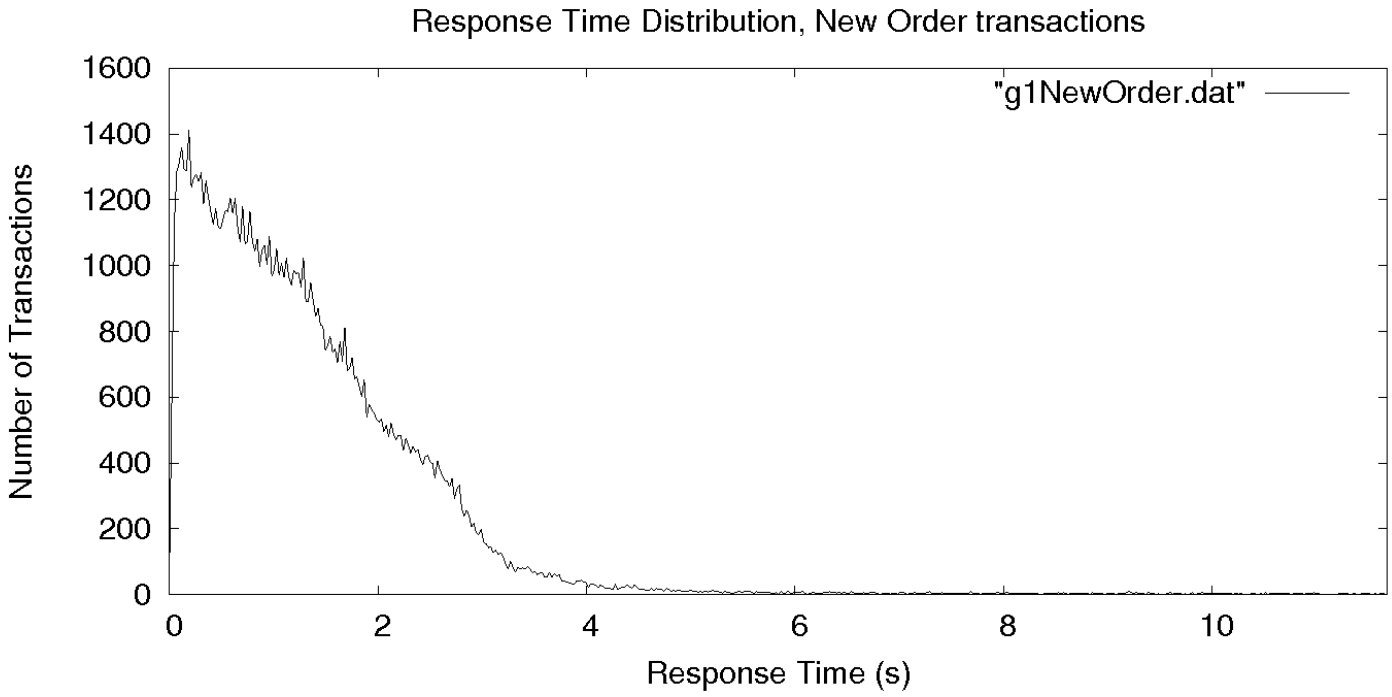
```
No vacuums executed.
>> TEST PASSED
```

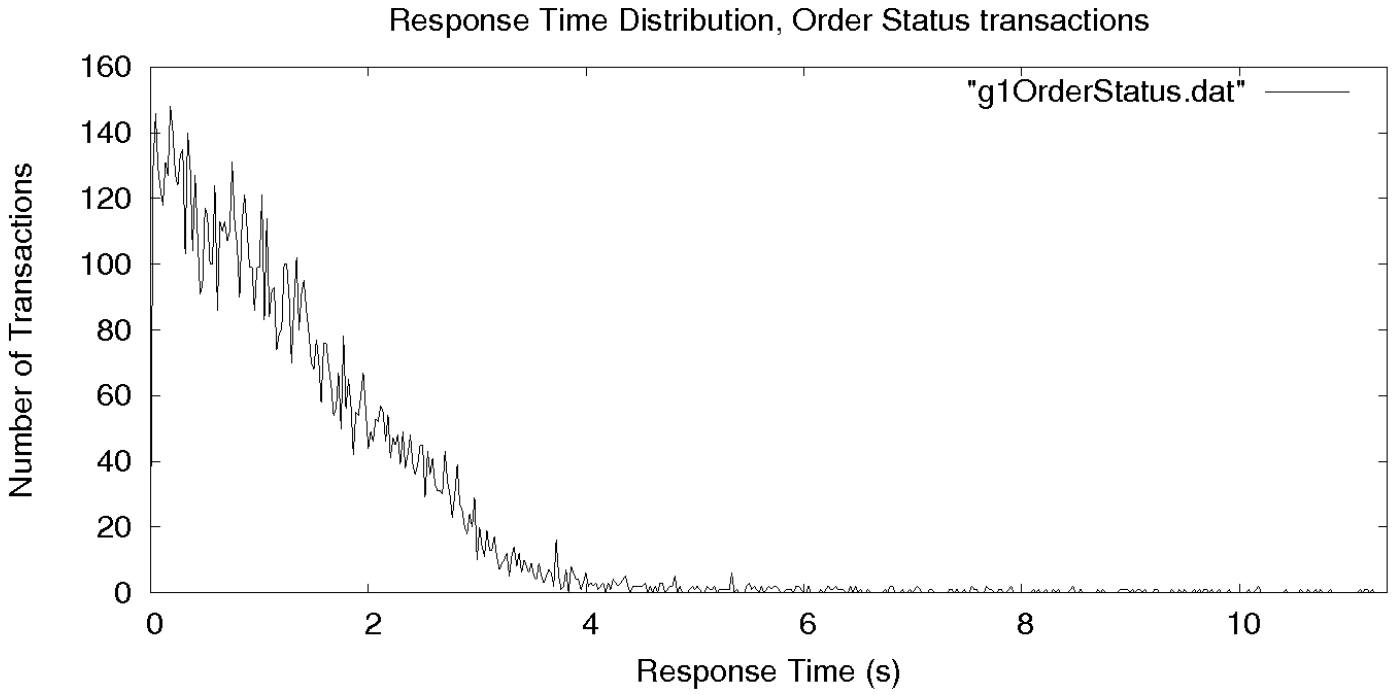
Analysis of the results as per TPC Council methodology

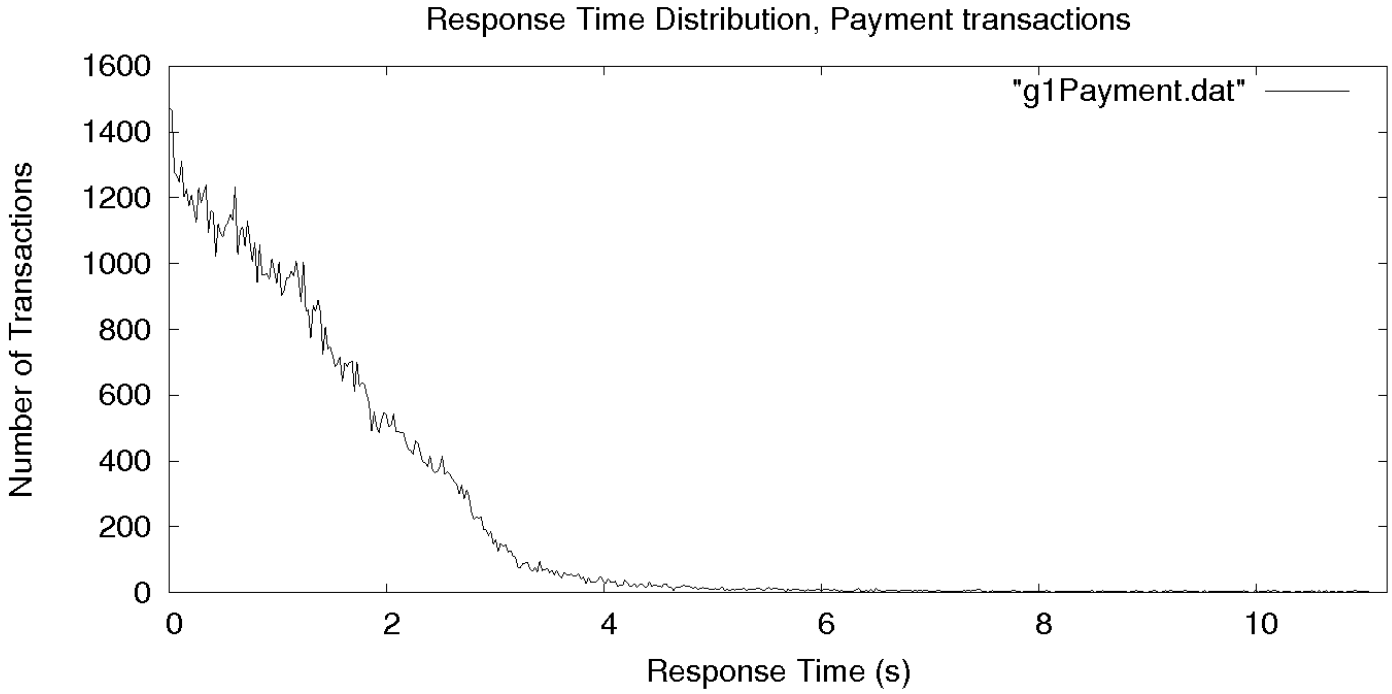
You will find the definitions for the following graphs in the tpcc current specs document, chapter 5.6 Required Reporting

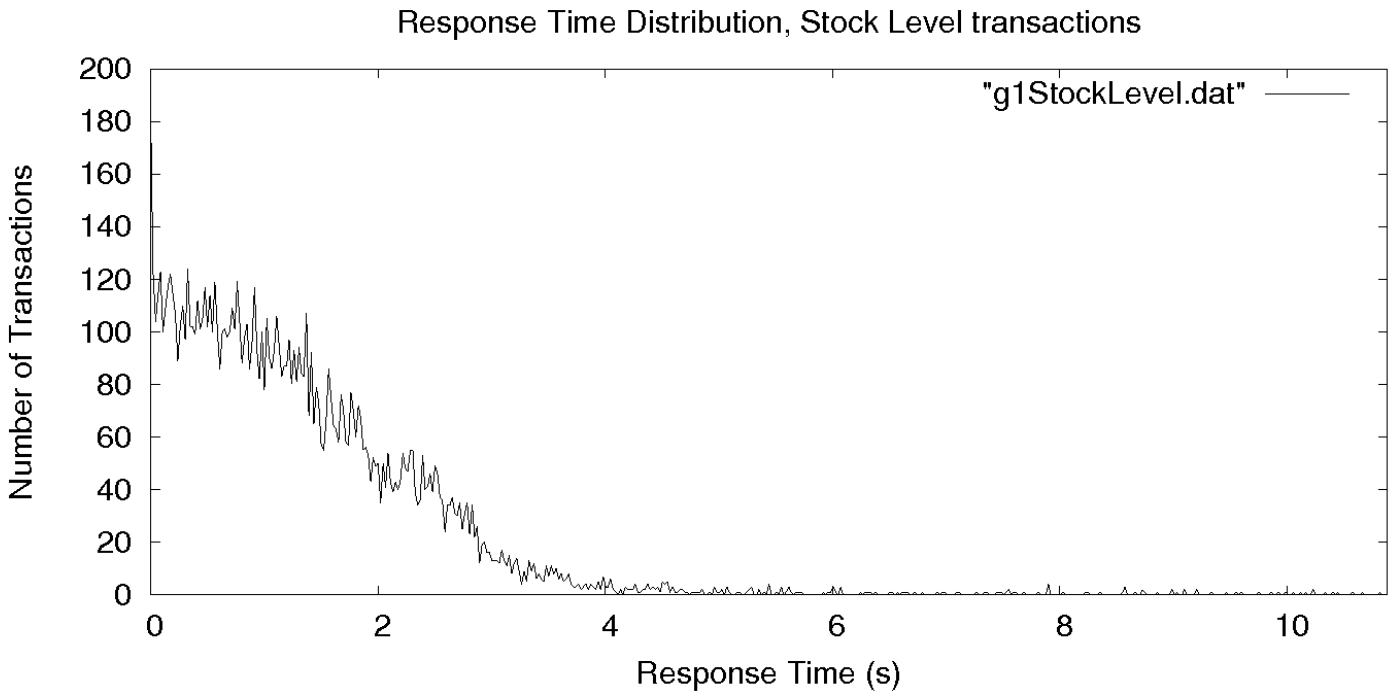
TPC Clause 5.6.1



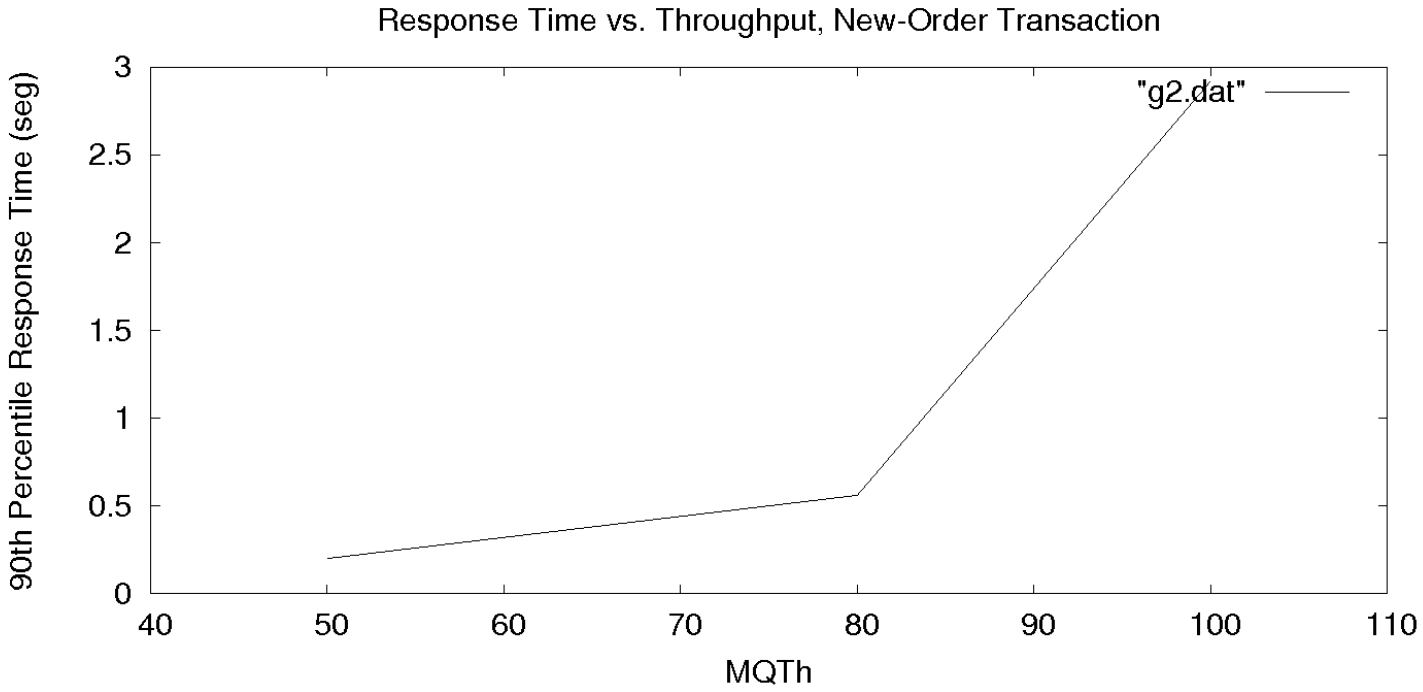



Response Time Distribution, Order Status transactions
"g1OrderStatus.dat"
Number of Transactions
Response Time (s)
0
20
40
60
80
100
120
140
160
0
2
4
6
8
10

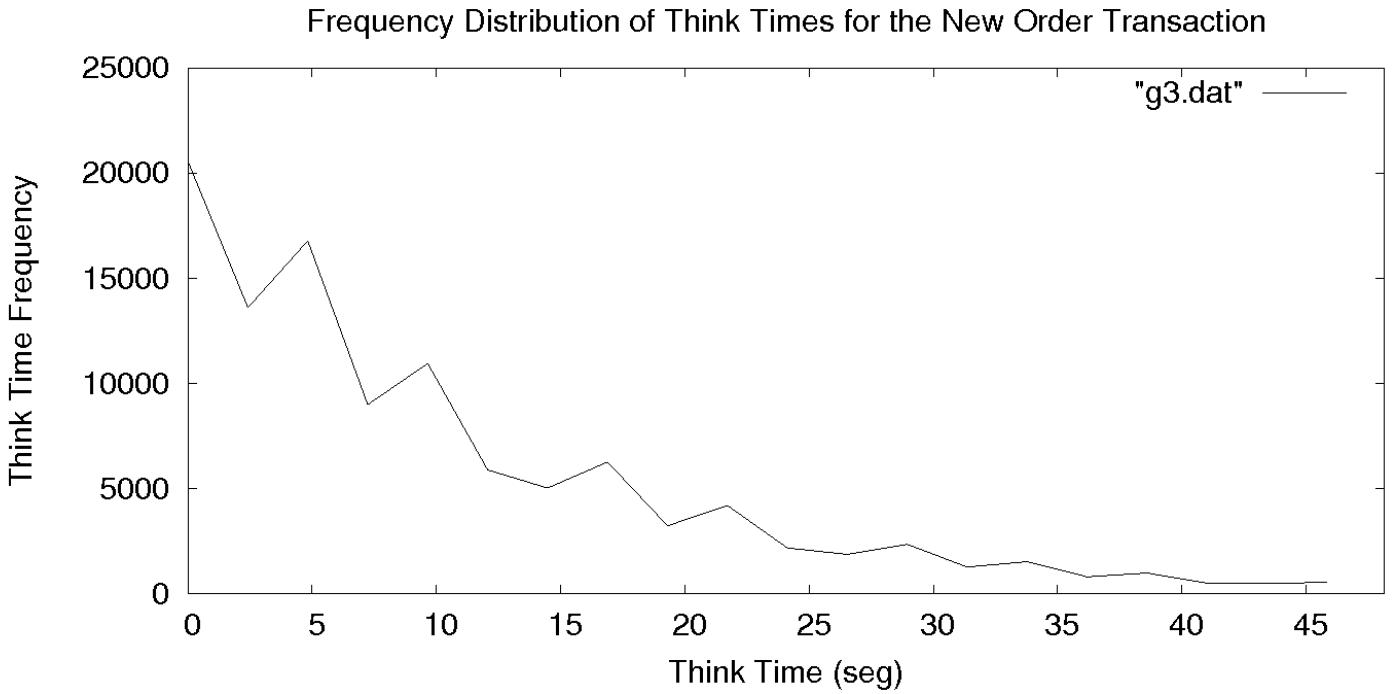

Response Time Distribution, Payment transactions
"g1Payment.dat"
Number of Transactions
Response Time (s)
0
200
400
600
800
1000
1200
1400
1600
0
2
4
6
8
10



TPC Clause 5.6.2



TPC Clause 5.6.3



TPC Clause 5.6.4