Using Performance Test Results for Capacity planning

Whitepaper prepared for the inaugural WOPR

By
Roland Stens
Stens Consulting Inc.
September 2003

Introduction

This paper describes the activities we deployed when we were confronted with very large requirements for a new Internet application. It highlights how a basically flawed process still can be useful in obtaining necessary guidance for a project team. I would not advocate this as an ideal approach, but it still provides us with some interesting lessons learned.

The following sections describe that process in more detail and highlight the issues and discussions we encountered along the way. I will also discuss some of the actual test results we acquired when the new Internet Data Center (IDC) was built and the application was ready to be load tested. And I will round off this paper by highlighting some of the current production measurements, a listing of what I felt went really well and what did not and last but not least my conclusions on the viability of using Performance Testing results for capacity planning purposes.

How it all started

In 2001 my client, who is a large public entity which implements the Worker's Compensation act for a Canadian province, decided to embark on developing a new Internet application that could potentially attract traffic from all of its 180,000 registered and active employers.

Up until that time, the largest Internet system deployed by my client was a system that typically did not attract more than 20 concurrent users. This system happily ran on a very light setup. However, it quickly became clear that for this new application, a more substantial new Internet Data Center (IDC) needed to be created. Concerned questions arose almost immediately about the size and capacity of this new IDC.

In order to jumpstart the thinking and decision-making process, I decided to take earlier performance test results from an application that I deemed similar (architecturally) and applied these numbers to a capacity model that also included the new requirements.

The requirements

The requirements for this new system were massive and the business analysts and the client were scrambling to come up with a correct estimation of the amount of users that would use the system in the first year of operation.

The main process that we were most worried about was a yearly mandatory reporting and payment process. All employers need to do their reporting and submit their assessment payments before end of February. Effectively automating this process would result in a really great ROI and much better data quality hence this project.

It was estimated by the business analysts and the client that the pickup would be between 15% and 30% (27,000-54,000) of the 180,000 potential users. It was also determined that 50% of the employers would report on the last day (13,500-27,000) and that 75% (10,125-20,250) of them would be reporting doing the business hours.

Other variables used were:

- Typical duration in which 75% of the users actually use the system: 6 hours
- Average Session length: 20 min

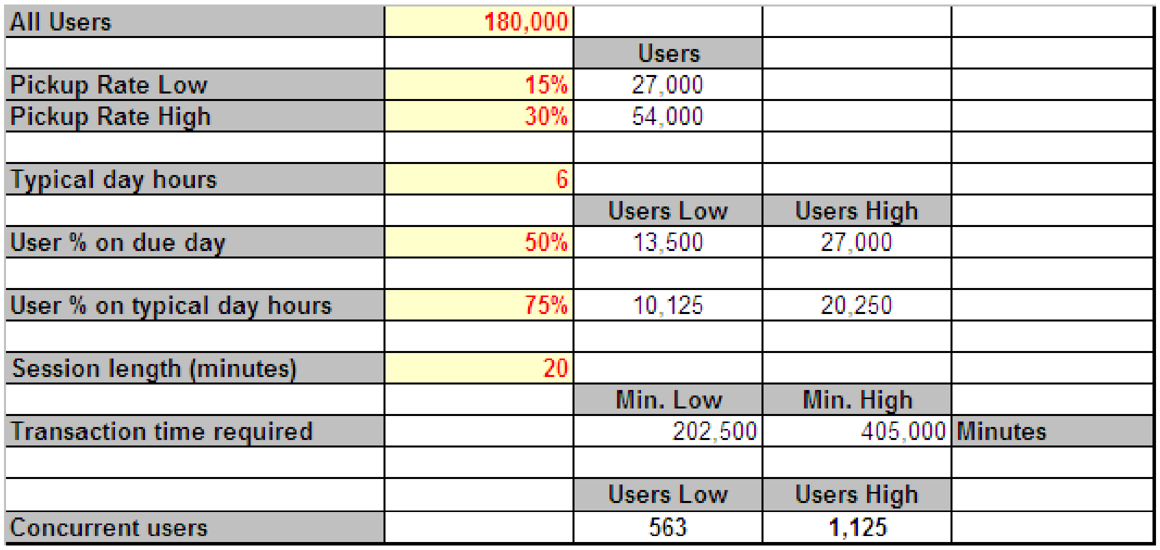
Immediately the quality of these estimates was questioned by a wide variety of stakeholders (most notably the operations folks).

- The **pickup rate** was the only factor that had not a base in actual measurements or research. But this number was largely driven by the desire of top management to make this into a success. The marketing campaign and the incentives needed to draw this number of users in.
- The **number of employers that would report on the last day** was estimated to be half of the users.

- The **percentage of users reporting during business** hours was a long known observation from the usage of our own Internet systems.
- The **average session length** was estimated based on a simulated walkthrough of the application's storyboard and was generally deemed correct.

The first kick at the can

As with most requirements, a quantification of what is really asked for is often very enlightening. So together with the business analysts we put the numbers in a small calculation model and found out that the expected amount of average concurrent users would be somewhere between 563 and 1,125.



All Users	180,000			
		Users		
Pickup Rate Low	15%	27,000		
Pickup Rate High	30%	54,000		
Typical day hours	6			
		Users Low	Users High	
User % on due day	50%	13,500	27,000	
User % on typical day hours	75%	10,125	20,250	
Session length (minutes)	20			
		Min. Low	Min. High	
Transaction time required		202,500	405,000	Minutes
		Users Low	Users High	
Concurrent users		563	1,125	

This was our simple calculation for the high estimate:
- 180,000 * 30% to get the # of users that would pick up
- Previous result * 50% to get # of users on due day
- Previous result * 75% to get # of users that would submit their info on the due date
- Previous result * 20 to get the total amount of minutes of connection would be required
- Previous result / (6 hours * 60 minutes) to get the average concurrent users = 1,125

At that moment in time, the business analysts were quite content with the outcome. The systems and operation people however blew their collective lids and indicated that building a system for the support of this yearly event would likely prove to be a very costly affair and that they had no idea how costly it would be because they lacked any basis for comparison.

A need emerged to get a better idea on what kind of a machine park would be needed to support this application's peak demand. Since no real production data was available, I choose to take some of my earlier performance test results and extrapolate them.

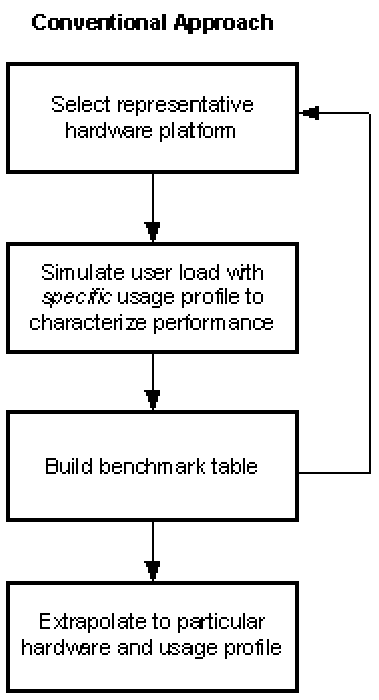
Now plain extrapolation is commonly seen as a flawed approach to make any statement about the capability and needed capacity of any setup. And I would typically agree that indeed the level of accuracy of the statements resulting form such a process are generally suspect. But as first step in understanding where you should go in defining a new architecture, it definitely has its value. In this particular case, the lack of alternatives was what drove us to this.

The base data

The new application needed to be built according to very specific architectural guidelines. These guidelines make that most of our Internet applications have very similar system use and behavior.

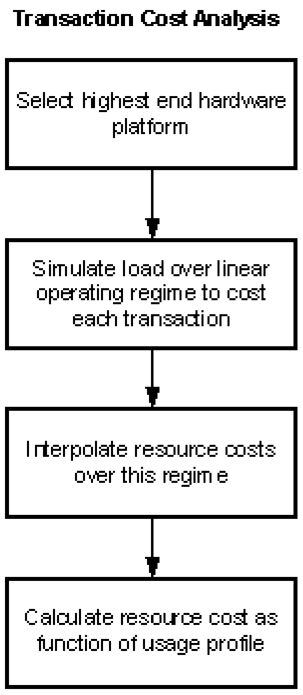
Using this fact, I reviewed our existing applications and choose one that was quite close in size and complexity to the proposed application. The only difference was that this application was built to support a maximum of 20 concurrent users and the new application would have to support significantly more.

This selected application had been performance tested and our repository still contained all the test results and all the system monitoring results.
By taking these results and extrapolating them, we were taking the conventional capacity planning route.


Conventional Approach
Select representative hardware platform
Simulate user load with *specific* usage profile to characterize performance
Build benchmark table
Extrapolate to particular hardware and usage profile

Alternatively we could have chosen the "Transaction Cost Analysis" route as advocated by Microsoft[1].

[1] A document describing this method can be found at:
http://www.microsoft.com/applicationcenter/techinfo/planning/2000/wp_tca.doc

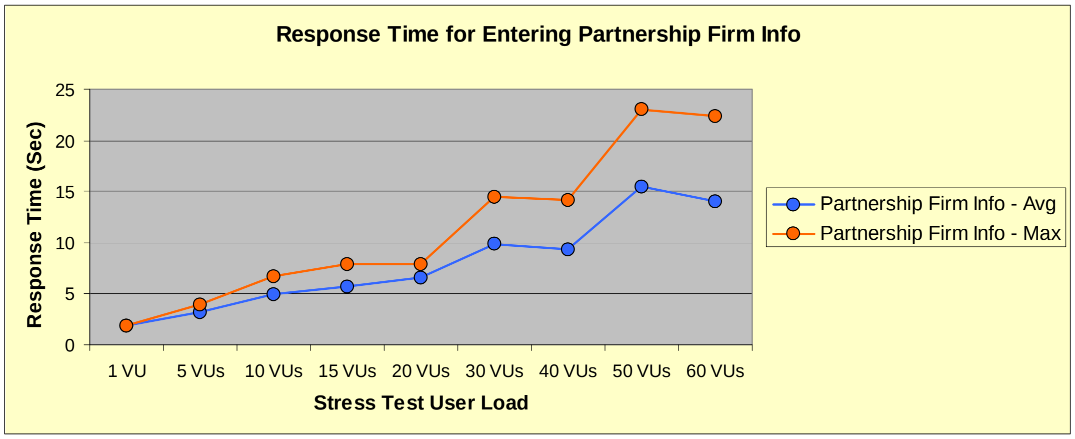

Transaction Cost Analysis
Select highest end hardware platform
Simulate load over linear operating regime to cost each transaction
Interpolate resource costs over this regime
Calculate resource cost as function of usage profile

This however would have required us to build new test scripts, find new servers and apply a new (for us) methodology. The time and opportunity were simple not there. (There is a future whitepaper in here though).

From the previous test results we took the CPU Usage% as the main indicator of the of the system usage. The servers from which we collected the information from were rather under-sized machines.
We also looked at the used network bandwidth. The new network bandwidth usage was a question that needed to be answered as well.

The test results indicated that at a 20 users stress test load, the performance was starting to degrade and that was also shown in the monitored data. At 20 users the systems were starting to show real signs of stress and any higher usage levels would typically cause all sort of problems for the machine.

Here's a sample of the results depicting one function under a series of stress tests.



Using the numbers from this very limited tests, I came up with a total of 47 servers (consisting of IIS, COM+ and SQLServer machines) that would be necessary to support this effort.
When these initial results were presented they caused, not surprisingly, quite some commotion. The sheer number of machines that came out of the calculation simply did not sound feasible. Interestingly enough, this number (as erroneous as it was) really acted as the spark that really ignited the effort and the interest from higher management. This was the moment that people fully realized that this would be a non-trivial exercise and that the organization was in for a very interesting 3[2] months, trying to come up with an architecture that would elevate the company from a mom and pop internet shop to a full-blown Internet data center.

Refining the ideas and working from a better number base

All the participants agreed that we had at least a beginning of an understanding. It was also understood that the model was extremely simplistic and needed more sophistication.

But first some more work needed to be done on the capacity-planning model.
We realized that there were some, Largely fixable, problems with it and those needed to be addressed in order to acquire better (more precise) figures.
The identified problems:

- The requirements:
  - The networks manager, who is also responsible for all the phone lines, brought statistics from the year before on the phone line usage of the call center which would typically handle the requests and questions from the employers during reporting time. This historic numbers did not indicate that an extreme large portion of the employers (50%) would wait until the last day. His figures indicated only an increase in traffic of only 25% on the last day. Also the total amount of calls was well below the numbers shown in the first requirement calculations. Quickly was pointed out that the telephone calls and the submission of a report and subsequent payment were not identical processes and you really could not compare them in this fashion. Still doubt remained about the high numbers quoted in the requirements.
  - The uptake of 15% to 30% was considered a pipe dream and after much deliberation is was decided that 20% uptake would be the high uptake number that should be taken into account. When the business and the business analysts were challenged on this number, it became clear that this was a goal set by higher management and that that goal should be attained.
- The peak factor: Production monitoring indicated to us that our peak usage on the current systems was 2.55 times higher than our average usage. We decided to adopt this factor to calculate our peaks.
- The percentage of users that would report on the last day seemed excessively high. Further analysis indicated that the percentage should be 27%. This number came straight from the CCRA (Canadian IRS) as they experienced this kind usage with their E-Filing system on the tax due date. And since this process was deemed very similar in nature, the business analysts found that this was a valid assumption.
- The servers that were used in the original run were old and tired 400 MHz, single processor NT servers. The new environment would be running Windows 2000 and most likely on multi-processor machines. So it was decided that we would rerun our tests on better machines and with higher user counts in order to build a better base to extrapolate from.
- The user percentage that would use the system during business hours also seemed a bit high and after some discussion it was decided to bring it back to 70%. This seemed in line with our observation from our production Internet system.
- The total number of participating employers was challenged as well. Stimulated by the idea that a small change in the business requirements could mean a significant reduction in cost for the new production system, the business analysts and the client sat down to try and bring the total number of employers down. A solution was found when they realized that with the new system they really did not need to have the fixed cut off date. This date was originally established to provide the processing department enough time to deal with all the incoming reports and payments. But now, this was all

[2] Actually 2.5 months. This was one veritable deathmarch.

changing and becoming more efficient. By breaking up the large, 180,000 employers, group into smaller groups with different due dates, they effectively diminished the requirement significantly.

When we applied all the requirements changes the model looked like this.

All Users	**91,132**			
		Users		
Pickup Rate Low	**15%**	13,670		
Pickup Rate High	**20%**	18,226		
Typical day hours	**6**			
		Users Low	**Users High**	
User % on due day	**27%**	3,691	4,921	
User % on typical day hours	**70%**	2,584	3,445	
Session length (minutes)	**20**			
		Min. Low	**Min. High**	
Transaction time required		51,672	68,896	**Minutes**
		Users Low	**Users High**	
Average Concurrent users		**144**	**191**	
Peak factor	**2.55**			
Peak Users		**366**	**488**	

The business analysts and the client had determined that the largest group of employers that the system could expect to have to handle was 91,132. This number was retrieved directly from the production database, where we counted all the employers with a specific business characteristic.

It is clear that these changes were quite dramatic, but everybody felt that they were significantly more realistic. In the end the nice round number of 500 is chosen as the number of concurrent users that the system need to be able to support.

The new numbers

New servers were found to run the test on and we decided to run realistic tests (not stress tests) for 20, 40, 60, 80, 100 and 200 users. This was done to find the level of users that would use the maximum amount of allowable CPU capacity. For our Microsoft Windows 2000 servers we had determined that that was a usage of 70%. We knew from experience and from MS documentation that at this level the processor operates at its peak efficiency and beyond this number the system performance starts to suffer.

We did not just look at the CPU usage we also kept an eye out for the response times. These response times were not allowed to go over 10 seconds for most transactions.
After evaluation of the results, it was clear that the results of the 60-user test met all the above stated requirements and we decided to use that result as our basis for the extrapolation.

Calculating the extrapolation is a rather basic mathematical exercise.
What we basically want to calculate is how many equivalent machines (equivalent to the used test machine) would we need to support the required load and get a similar usage on our production machine.

Below is the calculation page for the different machines and the network capacity.

Extrapolations based on:	60	Users Test

IIS Servers Needed	Users Low			Users High			
	Min Process.	Avg Process.	Max Process.	Min Process.	Avg Process.	Max Process.	
	2.50%	11.99%	18.96%	2.50%	11.99%	18.96%	Measured Processor 0
	2.50%	12.61%	22.29%	2.50%	12.61%	22.29%	Measured Processor 1
Equivalent to RC14ID09	1	1	2	1	2	2	

COM+ Servers Needed	Users Low			Users High			
	Min Process.	Avg Process.	Max Process.	Min Process.	Avg Process.	Max Process.	
	50.83%	81.74%	99.79%	50.83%	81.74%	99.79%	Measured Processor 0
	54.38%	80.26%	100.00%	54.38%	80.26%	100.00%	Measured Processor 1
Equivalent to RC14ID10	4	5	7	5	7	9	

COM+ SCON Servers Needed	Users Low			Users High			
	Min Process.	Avg Process.	Max Process.	Min Process.	Avg Process.	Max Process.	
	1.67%	11.84%	23.73%	1.67%	11.84%	23.73%	Measured Processor 0
	0.52%	8.70%	20.50%	0.52%	8.70%	20.50%	Measured Processor 1
Equivalent to RC14ID14	1	1	2	1	1	2	

SQL Servers Needed	Users Low			Users High			
	Min Process.	Avg Process.	Max Process.	Min Process.	Avg Process.	Max Process.	
	0.73%	7.27%	18.33%	0.73%	7.27%	18.33%	Measured Processor 0
	0.83%	5.46%	12.50%	0.83%	5.46%	12.50%	Measured Processor 1
Equivalent to RC14SD04	1	1	1	1	1	2	

Bandwidth	Users Low			Users High			
	Min Mbs	Avg Mbs	Max Mbs	Min Mbs	Avg Mbs	Max Mbs	
	0.09	0.47	0.87	0.09	0.47	0.87	Measured Bandwidth
Extrapolated from iReg	0.55	2.87	5.31	0.73	3.82	7.08	

(We have defined 2 types of COM+ machines: a general type and SCON type. SCON stands for SecureConnect our security framework).
By calculating the figures for the low and high uptake and for the minimum, average and maximum measurements, a clearer picture is starting to emerge.

Our advice based on this was now quickly summarized in the following sheet:

Our Advice	
IIS Servers Needed	2
COM+ Servers Needed	7
COM+ SCON Servers Needed	2
SQL Servers Needed	1
Bandwidth Mbs	4.56

Based on:

Peak Usage
60-User test
and
Average of:
High Users
Average measurement
Low Users
Maximum measurement

To give you an idea what the numbers would be if we had chosen another base for our numbers, check out the following list. This list contains all the calculation for all the different bases. This illustrates nicely that the high numbers of users bases give typically lower numbers of needed servers.

Summary

IIS Servers Needed	Users Low			Users High		
Based on:	**Min**	**Avg**	**Max**	**Min**	**Avg**	**Max**
20 User Test	1	2	3	1	2	4
40 User Test	1	1	2	1	2	2
60 User Test	1	1	2	1	2	2
80 User Test	1	1	2	1	1	2
100 User Test	1	1	1	1	1	2
200 User Test	1	1	1	1	1	1

COM+ Servers Needed	Users Low			Users High		
Based on:	**Min**	**Avg**	**Max**	**Min**	**Avg**	**Max**
20 User Test	10	12	15	13	16	20
40 User Test	5	7	9	7	9	12
60 User Test	4	5	7	5	7	9
80 User Test	3	4	5	4	6	7
100 User Test	2	3	4	3	4	5
200 User Test	1	2	2	2	2	3

COM+ SCON Servers Needed	Users Low			Users High		
Based on:	**Min**	**Avg**	**Max**	**Min**	**Avg**	**Max**
20 User Test	1	1	2	1	2	3
40 User Test	2	1	2	3	1	2
60 User Test	1	1	2	1	1	2
80 User Test	1	1	2	1	1	2
100 User Test	1	1	1	1	1	2
200 User Test	1	1	1	1	1	1

SQL Servers Needed	Users Low			Users High		
Based on:	**Min**	**Avg**	**Max**	**Min**	**Avg**	**Max**
20 User Test	1	1	3	1	1	4
40 User Test	1	1	7	1	1	9
60 User Test	1	1	1	1	1	2
80 User Test	1	1	2	1	1	2
100 User Test	1	1	3	1	1	4
200 User Test	1	1	2	1	1	2

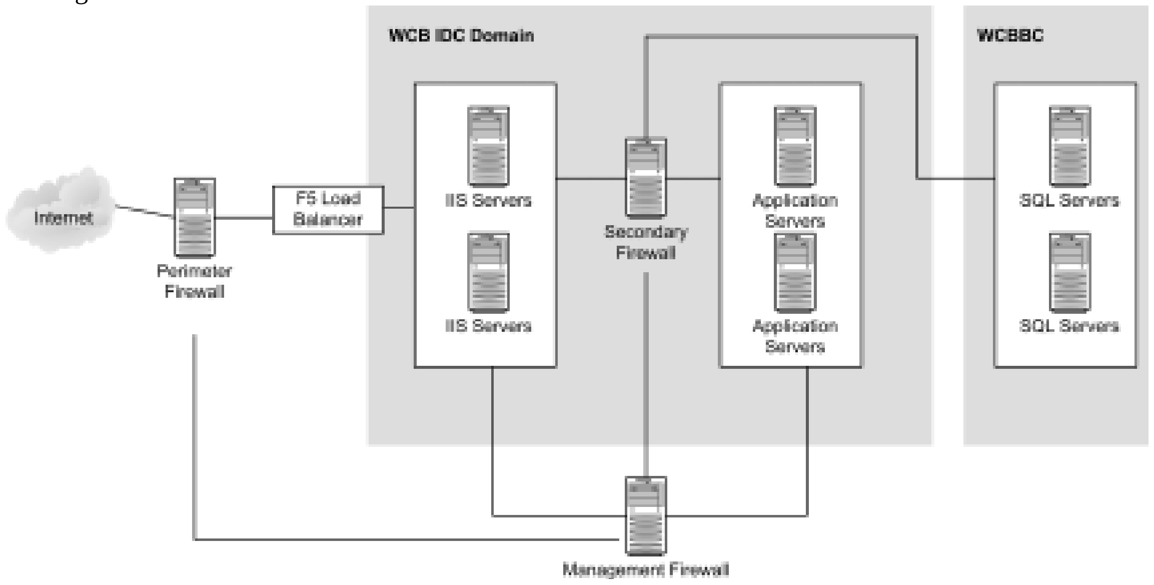
Bandwidth Mbs	Users Low			Users High		
Based on:	**Min**	**Avg**	**Max**	**Min**	**Avg**	**Max**
20 User Test	2.00	5.00	11.00	1.71	6.10	14.64
40 User Test	0.92	3.39	5.95	1.22	4.51	7.93
60 User Test	0.55	2.87	5.31	0.73	3.82	7.08
80 User Test	0.64	2.38	4.44	0.85	3.17	5.92
100 User Test	0.55	2.16	3.48	0.73	2.88	4.64
200 User Test	0.15	0.79	1.96	0.20	1.05	2.61

The chosen configuration

In all our testing of applications at my client, we have observed that, because of the chosen architecture, the COM+ layer always needs the largest capacity and is commonly regarded as the bottleneck.
But contrary to our advice, the system architects decided that they would need fewer machines for the COM+ layer than our advice. Their main argument was that the intended machines were even more powerful than our test machines and that would erase any capacity problems.

	Our Advice	Chosen
IIS Servers Needed	2	3
COM+ Servers Needed	7	6[3]
COM+ SCON Servers Needed	2	
SQL Servers Needed	1	2[4]
Bandwidth Mbs	4.56	10

The high level schematic of the new environment:



This setup was built and made available to us for testing. For the organization this environment meant a very serious step forward with technology deployed. The above graphics does not do the complexity of the setup justice. Not only did they built all the machines, the security setup had gotten major attention, redundancy was built in together with monitoring and management facilities.

According to Microsoft, this IDC was the most advanced setup in Western Canada and Microsoft's consultants showed a strong interest and commitment to make this environment work well.

[3] SCON COM+ servers and COM+ servers were consolidated.
[4] 1 redundant machine, so in fact only 1 machine that carries the load.

Putting the theory to the test

Finally we go the change to put our lofty theories to the test.
The environment was complete but instead of the 3 planned IIS servers, there were only 2 installed and of the 6 COM+ server there were only 4 installed. This was closer to a production environment we had (and will) ever tested on and we were quite excited to give it its first big workout. All the proud parents (the people who worked day and night to get this ready) were very keen on hearing the results from usas soon as they came available.

We already had created our test scripts and had performance tested the application on smaller sized machines and the first couple of rounds of tuning were already over. We basically had a stable and, as far as we could tell, nicely tuned application.

We had scripts for this reporting application but also for the all the other applications that were already running in the Internet environment. We proceeded to test every application individually and then together with all the others.

Our individual tests with the reporting application showed that up to 300 users in a realistic test, everything was just fine, the machines were busy but managed to do the work. However over 300 users, the machine were quickly running out of steam and particularly the COM+ suffered from a serious problem cause by excessive context switching. This problem was later traced back (way after the tests) to a significant architectural problem. At the 500-user level, responses times of 25 seconds and higher were measured and we had to conclude that the application and the environment failed the requirement of 500 concurrent users operating within the specified response times. Given the fact that we were missing 2 COM+ servers, the client was willing to assume that those would cause the 500 users to operate within specifications. We formally could not support that conclusion, because the general sentiment was that the 2 servers would only marginally add to the capacity because of the rampant context switching problems.

Follow up stress tests however, showed that the system was remarkably stable but slowed down to a crawl under heavy stress. The fail over tests to verify the built-in redundancy and fail over mechanisms resulted in a pass for the environment. We even tested the SCON (The security framework) up to 1000 users and this proved to be a huge success due to a slightly different architecture and some smart choice made in the application design.

After the tests, we had to conclude that our capacity planning numbers had been pretty close. We only should have expanded our performance and usage indicators in our planning. The appearance of this context switching problem was unique to this environment and the large number of users, but we could have had an early warning if we had taken this into account when we extrapolated.

In production

Before the deadline neared, we had a meeting with the operational people to decide if we wanted to install an additional 2 COM+ servers, particularly because our tests had indicated that the machines under maximum load were under a lot of stress and that the response time deteriorated to over 25 seconds. Since we had the machines available, we decided to just add them to the set up. So now we had 13 powerful machines ready for the onslaught.

Then the final days of February and the first deadline for the 91,000+ employers came. In the few weeks before the deadline we had already observed increased traffic and increased registration of employers to our Internet systems. However the registrations (necessary to use the reporting and payment applications) were not up to numbers close to the expected volume. But everybody was optimistic that the last day of February would see a flurry of activity.

Last day of February comes and goes and system capacity has not been used over 15% during the peak processing time. A morning session with the system monitoring people, the performance testers and operational support is cut short after 15 minutes, because we see very little happening, Next day when the submission counts are done, we come to the sobering conclusion that only **2%** of the expected employers have actually submitted their reports and payments through the Internet reporting system. The common reaction from the operational side of the company was: “I told you so!”.

It was apparent that wanting something does not automatically include getting it. As systems people, we felt that we did our job based on the requirements (which we had challenged time after time) and that we did exactly what was asked of us.

More than one jokester mentioned that “All our applications will run like greased lightning now!”.

What did we learn for our capacity plan?

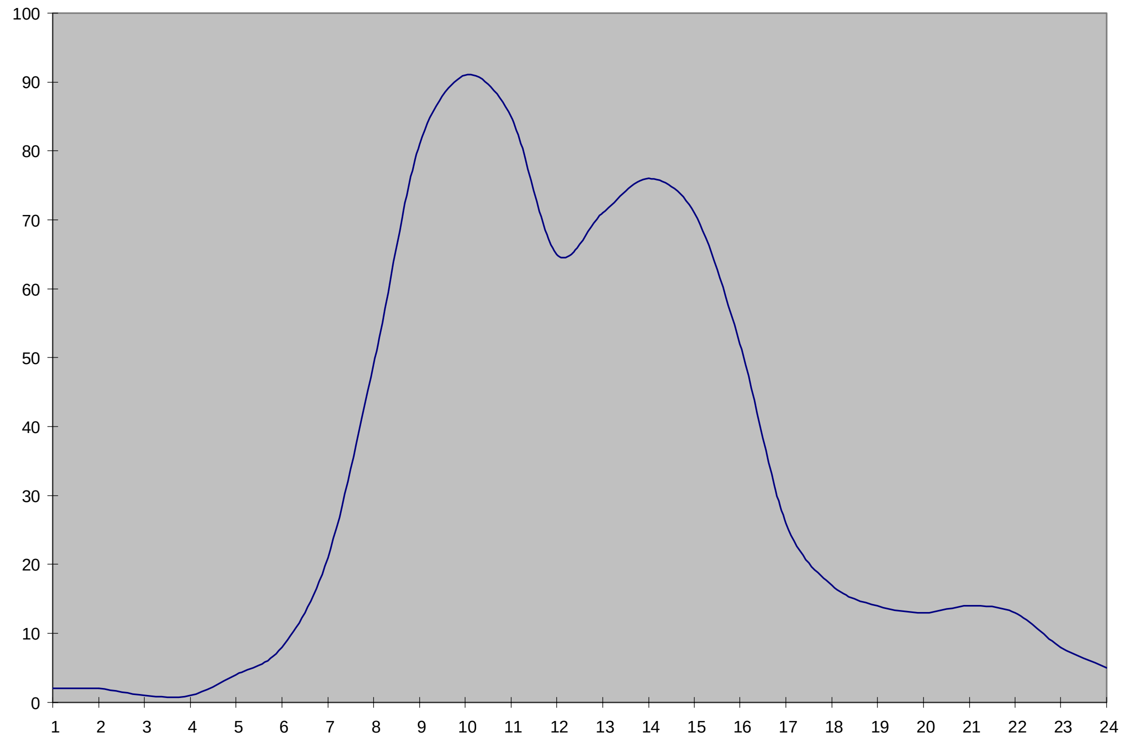
We found that instead of 70% of the users visiting us during 6 hours, we found that we actually attract the 70% over 7 hours. This 1-hour difference would have taken 70 users from our requirement.

We also found that our peak factor is very close to 2. Supported by over 6 months of data we have to adapt this new peak factor.

We now also know that Wednesday is our busiest day and that day is typically 25% busier than Friday, easiest weekday. The weekends are flat.

And it became clear that most of the employers do their work in the morning and between 9 and 10 is their preferred time. Lunches are very punctual at 12 and the afternoons are spent leisurely in anticipation of the evening.

Here’s the graph depicting the average weekday usage over the last 3 months:



All these new facts would have given this estimation:

All Users	**91,132**			
		Users		
Pickup Rate Low	**15%**	13,670		
Pickup Rate High	**20%**	18,226		
Typical day hours	**7**			
		Users Low	**Users High**	
User % on due day	**27%**	3,691	4,921	
User % on typical day hours	**70%**	2,584	3,445	
Session length (minutes)	**20**			
		Min. Low	**Min. High**	
Transaction time required		51,672	68,896	**Minutes**
		Users Low	**Users High**	
Average Concurrent users		**123**	**164**	
Peak factor	**2**			
Peak Users		**246**	**328**	

2 simple changes, that causes the model to loose 160 users. It is painfully clear that even pretty minor factors can have such an influence on the outcome of the model. This makes you indeed wonder about the viability of such a model particularly because the test results seem to corroborate the model we used.

Conclusions

The requirements gathering process was driven by desire and not reality. Later I read that pick up rates of 2%-5% for new corporate Internet applications are considered normal and 20% is considered a runaway success. In the future these fact will have to be brought forward. But the risk of moving yourself into a political minefield is great.

The model worked for us because we had nothing else and even the roughest approximation was better than nothing. We were lucky that our tests in the chosen environment seemed to corroborate the model and our advice.

The base numbers were acquired by using a different application as the target application and on hardware that was significantly different. Furthermore, we had too little data concerning usage in our production environment and we made several bold assumptions about user behavior.

Extrapolation using one or two performance/usage indicators is simply not complete enough. During our testing of the application in the new environment we found significant problems in the architecture, problems that we could have had anticipated if only we had expanded our range of indicators. An omission like that is easily made, because you typically do not know what you are going to encounter in your performance tests.

Quantifying requirements and doing this kind of rough capacity planning is a very powerful tool to get people involved. In our case, several stakeholders went from a “whatever” attitude to completely focused on achieving a positive result. As such, I can really advise you to use this as a tool.

We are painfully aware of our inadequacies with regards to capacity planning. I have got a follow-up job on my plate for next year to see if we can come up with a better model. I am tempted to give Microsoft’s transaction cost analysis methodology a shot.