Application Performance Engineering
DHL's Experience

Ashok Shah

May 2006

Deutsche Post World Net
MAIL EXPRESS LOGISTICS FINANCE

Introduction:

- Who am I?
- What is my role?

A bit of history:

In 2000 DHL was:

- A federated organization in >220 countries
- Autonomous IT in many of these countries
- Slowly centralizing core applications
- Growing organically at about 10 %
- Capacity never an issue
- Approximately 90,000 employees
  1000 regional applications
- 1 Global Outsource partner

Now DHL is:

- A Global Organization servicing 228 countries
- IT serviced out of 4 regional centers
  - > 1700 legacy applications
  - > 30 Global applications
  - > 500,000 People, 1 Global outsource partner
- Grown by aggressive acquisition
- Applications so big even the largest machines small (some systems expected to top 70B transaction / year)

These changes meant a dramatic:

- Increase in business - business transactions
- Increase in number of stake holders & user base
- Increase in competition - need for increased responsiveness
- Need for Automation, higher fault tolerance
- Change in the deployment footprint
- Focus on rapid integration and assimilation.

The Results:

- (almost) 100% UAT Passes
- (almost) 100% Performance Failure

All the correct remedial Action was taken:

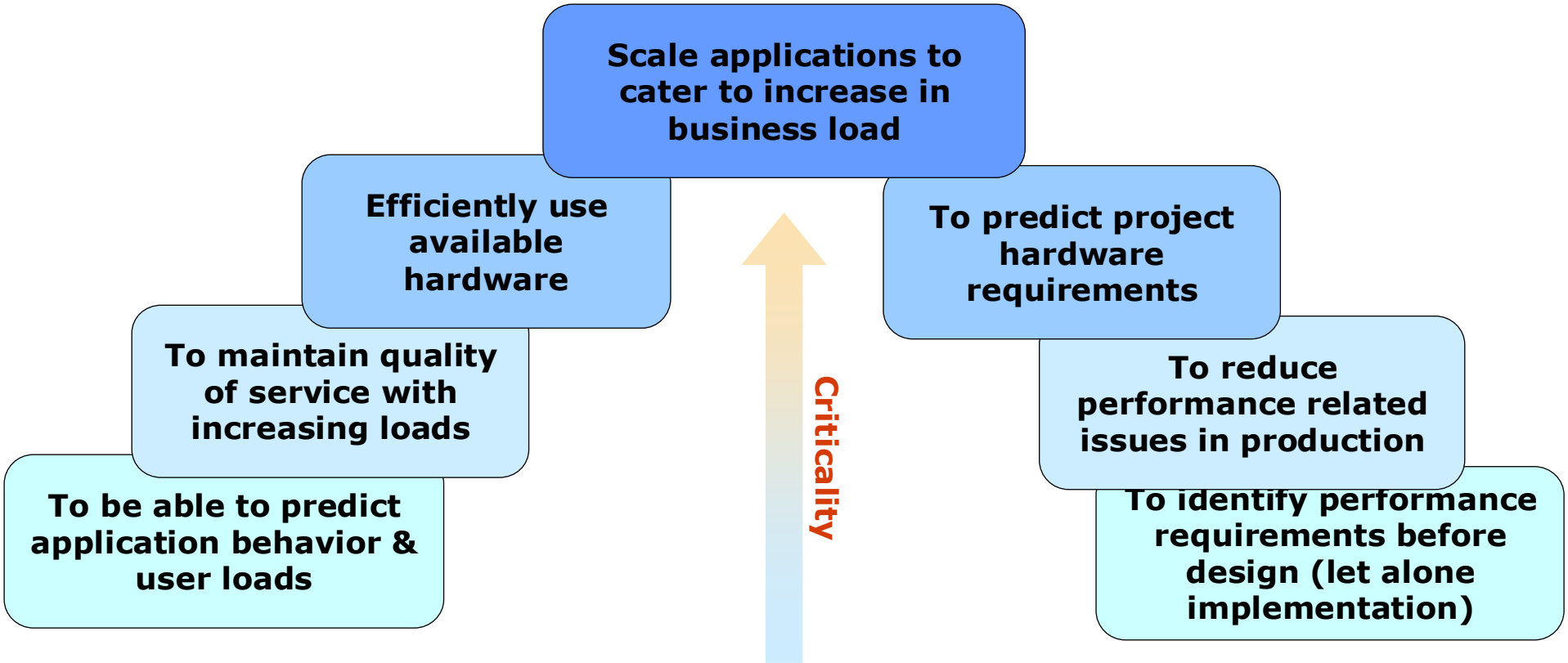
We put the failing applications into intensive care:
- Met daily to assess progress
- Blamed our suppliers

Threw more hardware at the problems:
- Applications not designed to multithread or cluster
- Blamed other suppliers

Scaled back deployment:
- Upset our users
- Blamed our users

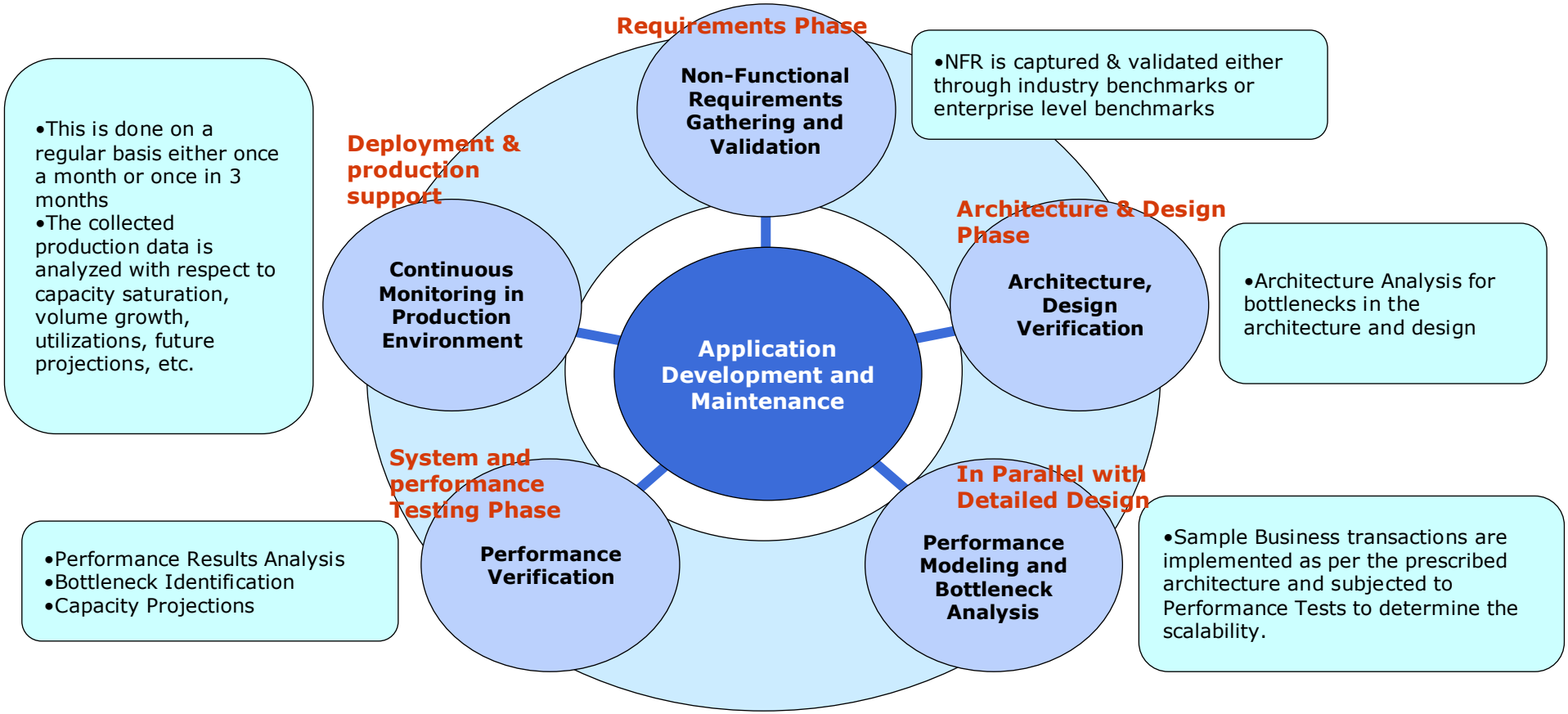
There had to be a better way!

Specifically the challenges for the internal IS department and third party vendors in maintaining/developing these applications were:



Although caught off guard by the sudden growth of business and the exponential complexity this caused, we realised we did not treat Non Functional Requirements seriously.

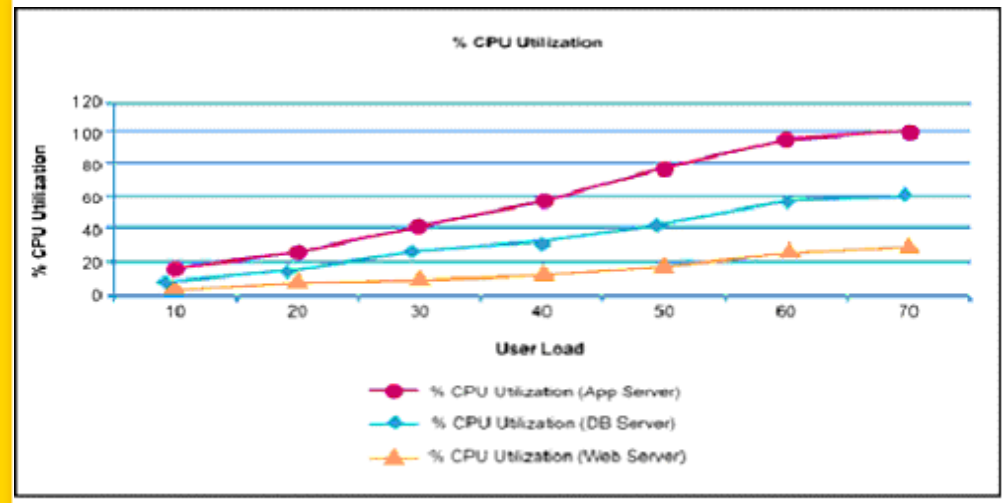
We had to learn (fast) that performance engineering was not a ‘nice to have’ and certainly was not an end of development ‘bolt on’.



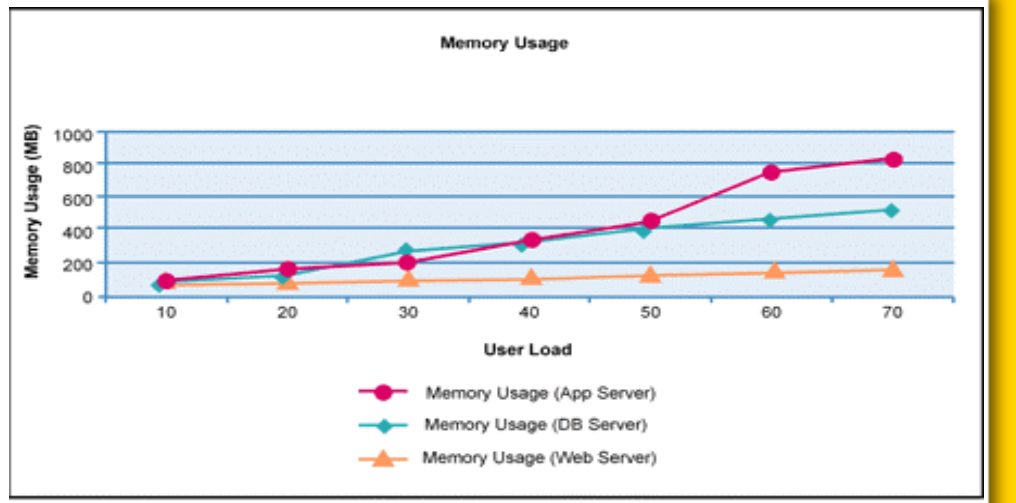
But wasn't it a bit late for the current applications?

Well Yes and No:

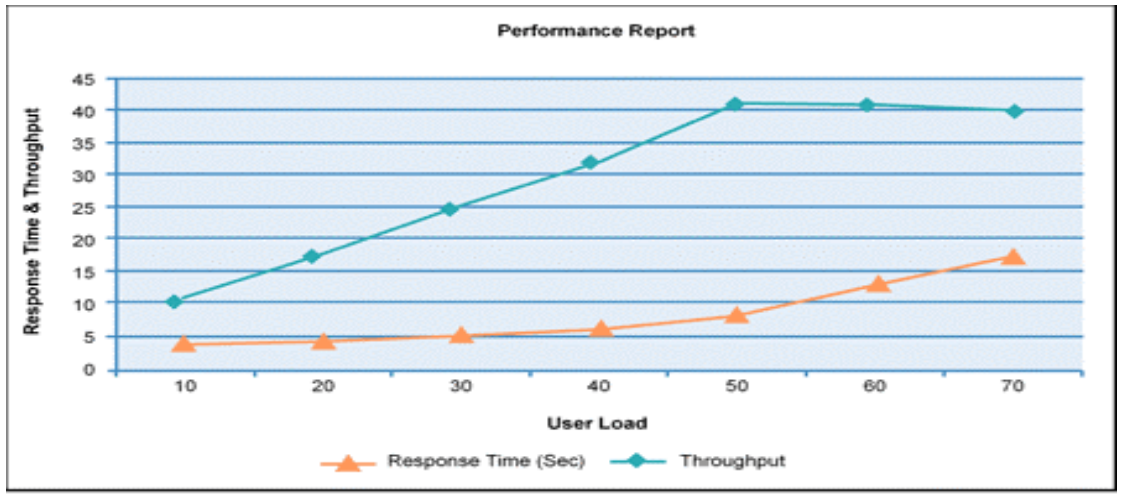
Application Performance Engineering



Percentage CPU Utilization



Memory Usage



Response Time and Throughput Graph

Application Performance Engineering

- WebSAM – scaled to support 8X users
- SCL - improved response time (~20%), reduced memory (~40%), CPU usage (~15%)
- NPTS – scaled to support 20X users
- CDU & CSV - performance improvement suggestions and capacity planning for regions
- CALMS – scaled to support 1.3X users
- PQT – 3X improvement in response times
- SDS - 10X increase in tps, 4X improvement in response times

Typical tools used for performance bottleneck analysis and improvements…

Tool Type	Tools
• Load Generator • Load Generator, Load Coordinator, Transaction Recorder	• HTTP - WebLoad, Web Application Stress Tool, Mercury LoadRunner, Rational TestStudio, Segue SilkPerformer • DCOM - Mercury LoadRunner, Rational TestStudio, Segue SilkPerformer • CORBA, RMI - Mercury LoadRunner, Segue SilkPerformer
• Performance Monitor	• Windows NT - PerfMon • Windows 2000 - Performance • Unix - vmstat, iostat, top, sar
• Instrumentation	• ARM
• Application Optimization	• Rational Purify - Memory error detector for C++, Java • Rational Quantify - Profiler for VB, C++, Java • Numega TrueTime - Profiler for VB, VC++, Java • Numega TrueCoverage, Rational PureCoverage - Coverage tool for VB, VC++ and Java • VMGear OptimizeIt, Sitraka JProbe - Java Profiler • Intel Vtune
• Infrastructure Design	• TeamQuest, Hyperformix, Metron
• Network Simulators	• Shunra Cloud, Ganymede Chariot
• Benchmarks	• InFlux™ Benchmarks Page

What we learnt:

Application Performance Engineering

- Performance management is an iterative process that must start at the beginning of the project
- The Business has to ‘own’ performance requirements in the same way as functionality
- For critical applications, almost by default you will never have all the HW/SW you need so model and test - then repeat.
- This will create a performance model that will be increasingly accurate over time. It must be maintained.
- This is a resource intensive activity; good planning helps avoid the resource contentions (people and hardware)
- End of project performance tuning is unpredictable,expensive and improvement is dependent on several parameters and not guaranteed – this is a last resort.
- Vendors hide their light. World class companies such as Infosys, HP and IBM have some of the industries best experts.They have proven methodologies, tools and frameworks but like us do not always deploy these skills when needed ie up front.

What they learnt:

- Performance is critical to us
- Vendors must push back when presented with inadequate requirements
- Bring in the experts on critical applications
- Vendors will have to partner with other vendors to be successful
- Create and maintain their part of the performance model.
- This is a resource intensive activity; good planning helps avoid the resource contentions (people and hardware)
- They are going to get the blame anyway!

Questions?

WebSam Scalability Analysis & Capacity Projection

Background

- **SAM 3.x was facing performance related issues. With the data volumes and number of users bound to increase, redesign of the architecture was done with changes being made to enhance the performance of the application to support growing workload. The main objective of SAM v 4.0 was to make the application more scalable and robust with high performance.**

Objective

- **Perform an analysis of the WSAM 4.0 Prototype (POC-3) to determine the application scalability**
- **Conduct Bottleneck Analysis of the application architecture**
- **Perform a capacity projection exercise to meet the projected Operational Workload by determining the Scale Factor with respect to the infrastructure in the test environment.**

Analysis & Inferences

- **Initial Observations revealed a problem at the app server side - Web logic crashing with increase in load. Modifications were done at the code level and the heap memory size was increased to resolve the issue.**
- **Further tests revealed high disk utilization (100%), low throughputs, low utilizations indicating a bottleneck at the database side.**
  - Clear observation of the disk subsystem revealed improper distribution / fragmentation of the database data.
  - After carefully distributing the database data across the various disks, and tuning of Unix Operating System and Informix database config parameters the disk bottleneck was resolved.
- **On this tuned environment, the system began scaling up with increase in load. Although the station and country level were scaling up, the Region Level Report may not scale up to meet the Required SLAs under operational workload. This has to be critically looked into**
- **It was determined that the bottleneck device is the Database CPU**

Performance Testing Approach

- **Creating an Abstract model of the system.**
- **Perform Workload Modeling and Workload Characterization of the Current Application to determine the workload characteristics.**
- **Performing Iterative performance tests on the provided Prototype.**
- **Identifying Bottlenecks.**
- **Capture Performance Metrics in the Lab Environment**

Conclusion

- **Analysis of Performance Metrics revealed that the application is scalable, except for the Region Level Report**
- **In the absence of proper NFR for WebSAM, what-if analysis was conducted with a scale factor of 3 (with reference to Lab Infrastructure for the bottleneck device – Database CPU) and 4 using Analytical Techniques for Extrapolation .**
- **For a Scale Factor of 4 it was recommended either to go in for 290 Concurrent users with 5 sec SLA ,500 Concurrent users with around 11 sec SLA**

SCL – Scalability Analysis and Capacity Projection

Background

- **SCL is being enhanced for piece enablement**
- **Some new transactions are being added to the existing application**
- **NFR has been provided for the already existing transactions however for the new transactions, SLA has to be defined**
- **Information in terms of expected data growth, future business growth mentioned in NFR is being taken into consideration**

Application-related

- **Three transactions were selected based on the following criterion: maximum throughput, effect of adding new transactions and category of transactions**
- **Purge has been selected as one of the transactions based on the inputs received from APIS about its criticality**

Workload-related

- **To determine whether the customer requirement of given throughput would be met. The throughput targets have been mentioned in the NFR for important transactions**

Database-related

- **Database organization and disk layout needed to be reviewed, since the application was more database centric**

- **Objective**
- **Perform an analysis of SCL 6.0 prototyped transactions to determine the application scalability**
- **To identify bottlenecks in the application that will impact the performance and scalability of the application**

Recommendations

- **Code profiling**
- **Database tuning (dropping of unnecessary indexes, providing appropriate fill factors for the clustered and non-clustered indexes, re-indexing, index de-fragmentation and query optimization)**
- **Hardware upgrades were suggested**

Findings

- **The implementation of the Batch Jobs was not correct**
- **Throughput mentioned in the NFR was not reachable and was not correct**
- **Scale factor for the overall system, taking business growth into account**