Server Architectures: System Performance and Estimation Techniques

January 2005

René J. Chevance

Foreword

- **This presentation is an introduction to a set of presentations about server architectures.**
  They are based on the following book:

  Serveurs Architectures: Multiprocessors, Clusters, Parallel Systems, Web Servers, Storage Solutions
  René J. Chevance
  Digital Press December 2004 ISBN 1-55558-333-4
  http://books.elsevier.com/

  This book has been derived from the following one:

  Serveurs multiprocesseurs, clusters et architectures parallèles
  René J. Chevance
  Eyrolles Avril 2000 ISBN 2-212-09114-1
  http://www.eyrolles.com/

  The English version integrates a lot of updates as well as a new chapter on Storage Solutions.

© RJ Chevance

Contact: www.chevance.com **rjc@chevance.com**

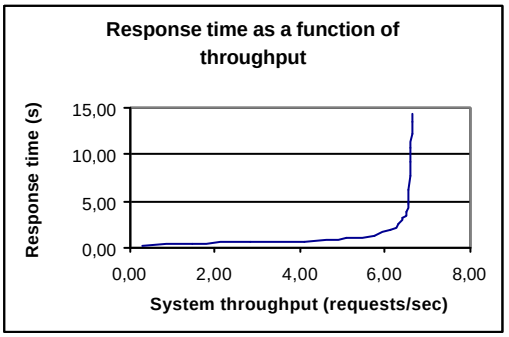
Organization of the Presentations

- Introduction
- Processors and Memories
- Input/Output
- Evolution of Software Technologies
- Symmetric Multi-Processors
- Cluster and Massively Parallel Machines
- Data Storage
- ➔ **System Performance and Estimation Techniques (this presentation)**
  - Dimensions of System Performance
  - Few Facts
  - Approaches to Performance
  - Benchmarks
    - Processor Level Benchmarks
    - System Level Benchmarks
    - Benchmarks: Final words
    - Comparing the Performance of Architectural Approaches
  - Taking performance into account in projects through modelling
    - Operational Analysis
    - Modelling Process
    - Modelling Strategy
    - Empirical rules of systems sizing
    - Capacity Planning
- DBMS and Server Architectures
- High Availability Systems
- Selection Criteria and Total Cost of Possession
- Conclusion and Prospects

© RJ Chevance

Dimensions of System Performance

- **System's performance can be expressed in two related dimensions:**
  - **Response time**
  - **Throughput**
- **These two dimensions are not independent**
- **Response time as a function of throughput - (Example of a system supporting a growing request rate)**



© RJ Chevance

Few Facts

- **Often, considerations of performance are left until last minute and are only taken into account when problems have already surfaced**
- **This kind of approach generally results in the panic measures that yield mediocre or even disastrous results for system architecture**
- **Proposed approach:**
  - **Identify application characteristics**
  - **Characterization of the proposed architecture:**
    - **System architecture**
    - **Application architecture**
  - **Sizing of the components**
  - **Performance estimation**
  - **Iterate if necessary**
- **We will get back on this approach later on**

© RJ Chevance

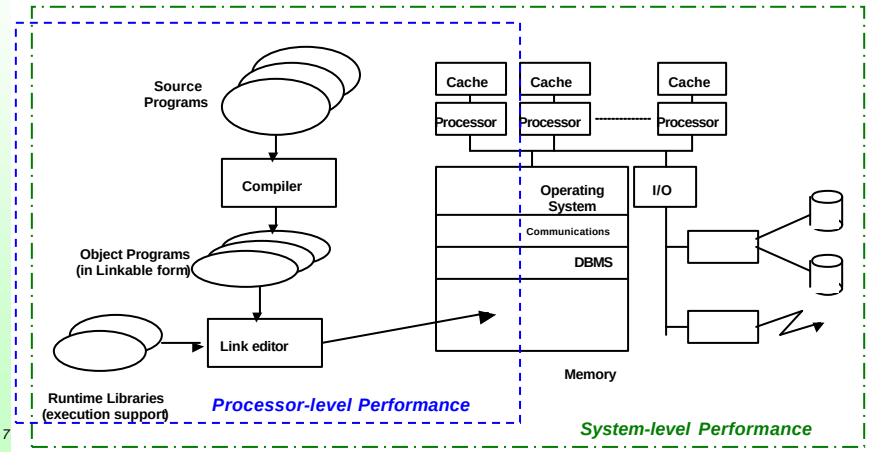
Approaches to Performance

- **Approches**
  - **Benchmarks**
    - ↗ **Performance comparison between different systems**
    - ↘ The use case illustrated by the benchmark may not correspond to the intended use of the system
  - **Intuition**
    - ↗ Simple, cheap, limited
    - ↗ Based upon experience
    - ↘ Lack of reliability
  - **Measurements**
    - ↗ Precision
    - ↘ Difficult to set up the environment
    - ↘ Cost
    - ↘ Interpretation not very easy
    - ↘ The system must exist
  - **Modeling**
    - ↗ Does not require the system to exist
    - ↗ Return on investment
    - ↘ Need a specific expertise

© RJ Chevance

System Level and Processor Level Performance

- With benchmarks, performance may be expressed at two levels:
  - Processor Level Performance
  - System Level Performance
- Characterizing performance at the levels of processor and system



© RJ Chevance

Benchmarks

- Processor Level
  - Types of benchmarks:
    - Synthetic pprograms (e.g. Whetstone, Dhrystone) resulting from the analysis of the behaviour of real programs
    - Kernels (e.g. Linpack) corresponding to the isolation of the most frequently executed part of programs
    - Toy programs: extremely small programs (e.g. Hanoï towers)
    - Real programs (e.g. SPEC)
  - Only, benchmarks based on real programs must be considered
  - *Note about MIPS: For a long time, mainframe and minicomputer performance was expressed in MIPS (millions of instructions per second), a measure that is about as useful (given the diversity of instruction set architectures) as comparing vehicles by noting how fast their wheels (which, even in a limited subset of vehicles such as automobiles, vary in diameter form 10" to 18") turned, over undefined routes with undefined road conditions*
  - For embedded systems: EEMBC (Embedded Microprocessor Benchmark Consortium). Visit www.eembc.com

© RJ Chevance

Processor Level Benchmarks

- SPEC (System Performance Evaluation Cooperative)
  - Evolution of processor level benchmarks:
    - SPEC89, SPEC92, SPEC95, SPEC2000
  - SPECCPU2000: collection of reference programs:
    - CINT2000: integer workload (12 programs - 11 in C and one in C++)
    - CFP2000: floating point workload (14 programs - 6 in Fortran77 and 4 in C)
  - SPECCPU2000 evaluation procedure:
    - measure the execution time of each program on your system (calculate the mean of three executions)
    - calculate the SPEC ratio for each program
    - SPEC_ratio = (execution time on Sun Ultra 5_10/execution time on system) x100
    - calculate the following values:
      - SPECint2000: the geometric mean of the SPEC_ratios for the integer programs, with any compiler options for any program
      - SPECint_base2000: the geometric mean of the SPEC_ratios for the integer programs, with identical compiler options for all programs
      - SPECfp2000: the geometric mean of the SPEC_ratios for the floating point programs, with any compiler options for any program
      - SPECfp_base2000: the geometric mean of the SPEC_ratios for the floating point programs, with identical compiler options for all programs
  - Information and results: www.spec.org

© RJ Chevance

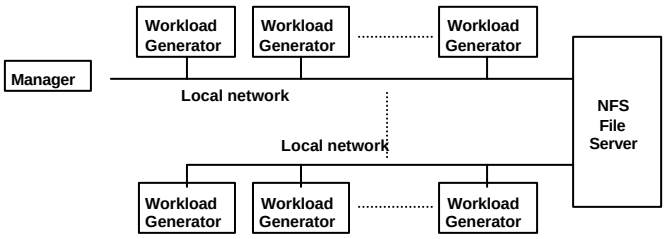
System Level Benchmarks

- SPEC System Benchmarks:
  - SPEC SFS97: System-level file server
  - SPECjbb2000: server portion of Java applications
  - SPECweb99: Web server
  - SPECjvm98: Java Virtual Machine
  - SPECjAppServer: Java Enterprise Application Servers (using a subset of J2EE)
  - SPEC_HPC2002: High Performance Computing
  - SPEC OMP2001: scalability of SMP systems (using the OpenMP standard)
  - SPECmail2001: email servers
  - …..

© RJ Chevance

System Level Benchmarks(2)

- **SFS System-level file server**
  - NFS File server
  - Workload based on the observation of NFS file servers
  - Test configuration:



  - Server response <40 ms
  - Results:
    - Average number of operations per second
    - Average response time per operation

© RJ Chevance

System Level Benchmarks(3)

- **SPECweb99**
  - Characterization of the performance of Web servers
  - Benchmark derived from the observation of real-world usage:
    - 35% of the requests concern files smaller than 1 KB
    - 50% of the requests concern files between 1 KB and 10 KB
    - 14% of the requests concern files between 10 KB and 100 KB
    - 1% of the requests concern files between 100 KB and 1 MB
  - Number of file repertories proportional to the server performance
  - Request distribution
    - Static get 70%
    - Standard dynamic get 12.45%
    - Standard dynamic get (cgi) 0.15%
    - Specific dynamic get 12.6%
    - Dynamic post 4.8%
  - Server throughput must be between 40 KB/s and 50 KB/s
  - Result: maximum number of connections the server can support

© RJ Chevance

System Level Benchmarks(4)

- TPC creation (Transaction Processing Council) follows up the first attempt to standardize an OLTP benchmark in the second half of the 80's (TP1 benchmark)
- TPC is a group of computer systems vendors and DBMS vendors
- Several types of benchmarks:
  - TPC-A (1989): simple and unique transaction updating a bank account (obsolet);
  - TPC-B (1990): TPC-A reduced to the Database part (obsolete);
  - TPC-C (1992): multiple and complex transactions;
  - TPC-D (1995): decision support (obsolete);
  - TPC-H (1999) : decision support
  - TPC-R (1999) : decision support
  - TPC-W (2000) : transactions over the (e-commerce, B-to-B,...)
  - TPC-E (not approved): enterprise server environment
- Metrics:
  - Performance
  - Price/performance
- Specifications and results available on http://www.tpc.org

© RJ Chevance

System Level Benchmarks(5)

- TPC-C (1992)
  - Transactions in business data processing;
  - 5 transactions types and 9 tables ;
  - Exercises all system's dimensions (processing subsystem, database, communications) ;
  - Generic specification to be implemented;
  - Obeys ACID properties (Atomicity, Consistency, Isolation and Durability) ;
  - Two metrics :
    - Performance : tpm-C (TPC-C transactions per minute)
    - Price/Performance : $/tpm-C (cost of ownership over a 3 years period divided par the number of transactions per minute)
  - Size of database and communication subsystem proportional to system performance (scalability) e.g. for each tpm : 1 terminal and 8 MB measured;
  - Response time constraints (90%) depending on each transaction type
  - TPC-C limitations as compared with real world applications:
    - Very low complexity
    - Too much locality

© RJ Chevance

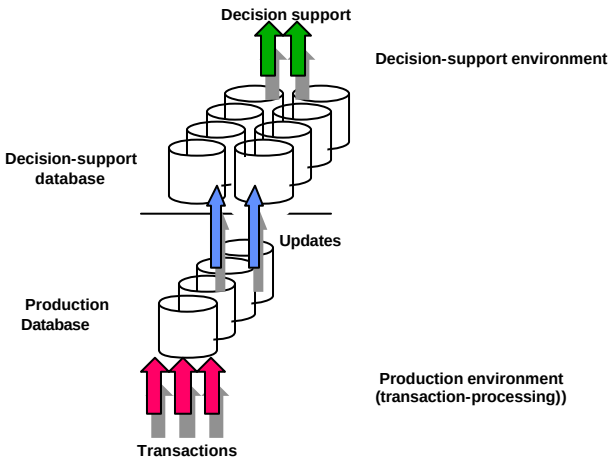
System Level Benchmarks(6)

- **TPC-H (1999) Decision support applications**
  - **Database available 24x24 and 7x7**
  - **Online update of the decision support database (from the production database) without interrupting decision support applications**
  - **Several database sizes possible e.g. 100 GB, 300 GB, 1 TB, 3 TB,…**
  - **Set of queries to be considered as unknown in advance (no optimization possible at database level)**
  - **Metrics (expressed relatively to a database size)**
    - **QphD@<database_size> : Composite Query-per-Hour characterize the throughput of the system**
    - **$/QphD@< database_size > : ownership cost for a five year period, divided by the performance of the system for a specified database size**
- **TPC-R (1999) Decision support applications**
  - **Similar to TPC-H except that requests are considered to be known in advance thus allowing optimizations at database level**
  - **Metrics**
    - **QphR and $/QphR**

© RJ Chevance

System Level Benchmarks(7)

- **TPC-H Application Environment**



© RJ Chevance

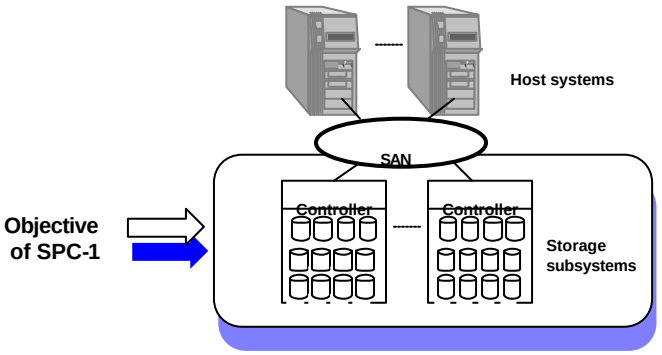
System Level Benchmarks(8)

- **TPC-W: Commercial Web Applications**
  - Three types of client requests :
    - information search, followed by purchase - B to C, for Business to Consumer
    - information search
    - web-based transactions - B to B, for Business to Business
  - 8 tables, 5 interaction types: Browse -information search, Shopping cart - item selection, Purchase, Recording of customer information, Searching for information on an item
  - Respect of the ACID properties and response time constraints
  - Size of the benchmark specified by the Scale Factor (i.e. the number of items for sale : 1,000, 10,000, 100,000, 1 million, or ten million)
  - Data concerning purchases must be maintained on line (i.e., on disk) for 180 days
  - Results :
    - Performance, expressed in WIPS@<scale_factor> (Web Interactions Per Second) measures the number of requests per second, ignoring network time (time is measured from entry of the request into the server to the time the response exits the server).
    - price/performance, expressed as $/WIPS@<scale_factor>, where the price is the cost of acquisition and of three years ownership of the system.
    - transactional performance, expressed as WIPSo@<scale_factor>; the number of transactions per second
    - browsing performance, expressed as WIPSb@<scale_factor>; the number of browse requests per second.

© RJ Chevance

Storage Subsystems Performance

- **Storage Performance Council (SPC - 2001)**
  - SPC-1 (SPC Benchmark-1) – SAN environment
  - Specification and results available on http://www.StoragePerformance.org
- **SPC-1 Environment**



© RJ Chevance

Storage Subsystems Performance(2)

- SPC-1 id composed of 8 streams, each stream representing the profile of an application. The set of 8 streams is called a BSU (Business Scaling Unit). To reach the saturation state, the number of BSU is increased.
- Streams:
  - 5 Read/Write random streams
  - 2 Read sequential streams
  - 1 Write sequential stream
  - A stream generates a total of 50 I/O requests per second
- **SPC-1 I/O streams:**

	Reads/s	Writes/s	Total (operations/s)
Random	14.5	16.2	30.7
Sequential	5.3	14.0	19.3
Total (operations/s)	19.8	30.2	50

- **With SPC-1, subsystem performance is characterized by two dimensions:**
  - SPC-1 LRT: response time under light load (10% of the maximum throughput)
  - SPC-1 IOPS: maximum throughput

© RJ Chevance

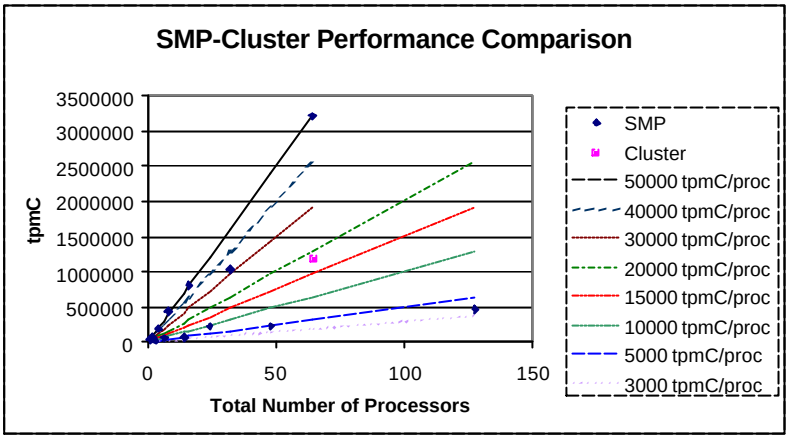
Benchmarks: Final Words

- ↗ Relatively good coverage of the domain
- ↗ Robustness of the definition and evolution processes
- ↗ Reasonably representative (within given limits)
- ↗ Provide comparison basis for various systems
- ↗ Since the investment necessary to perform measurements is large, publishing results is a sign of good financial health for Companies and confidence in the competitiveness of their systems
- ↗ Foster performance improvement process
- ↘ At limits, not really representative (e.g. at lowest $/tpmC or at highest tpmC)
- ↘ The cost of benchmarking activity may be dissuasive for some actors

© RJ Chevance

Comparing the Performance of Architectural Approaches

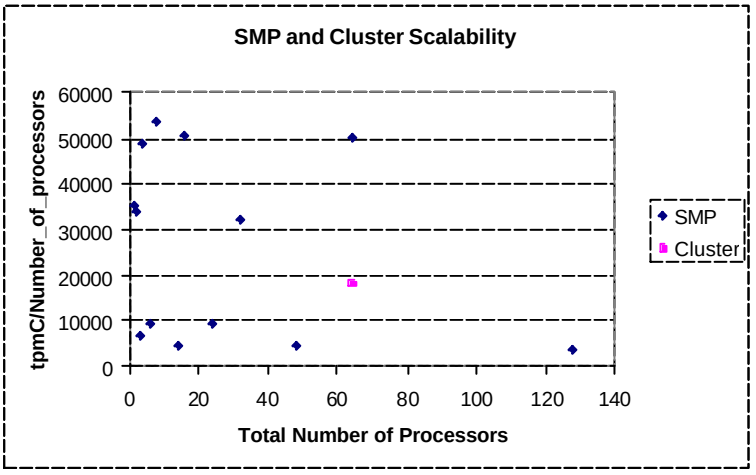
- **SMP-Cluster TPC-C Performance Comparison (source data TPC)**



© RJ Chevance

Comparing the Performance of Architectural Approaches(2)

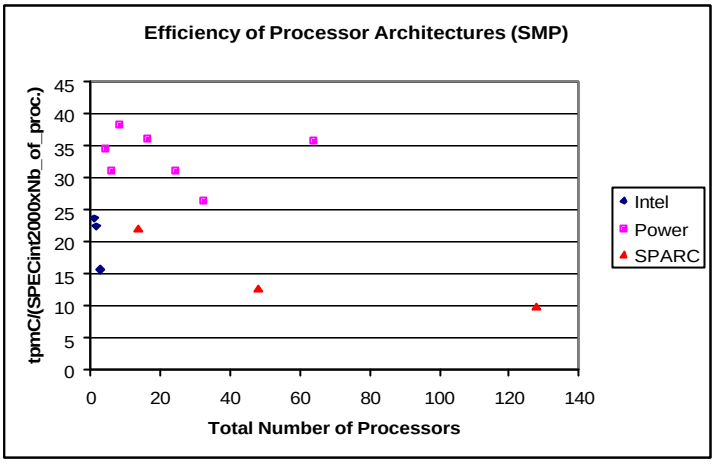
- **SMP and Cluster Scalability (source data TPC)**



© RJ Chevance

Comparing the Performance of Architectural Approaches(3)

- **Architectural Efficiency for TPC-C (source data TPC and SPEC)**

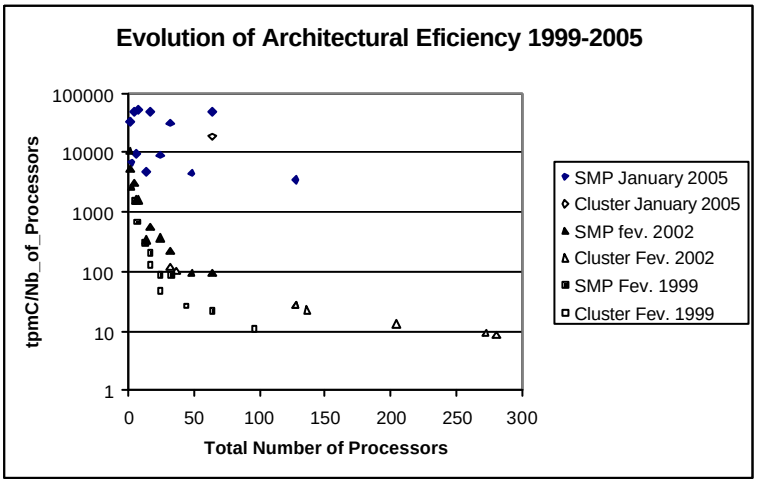


Page 23

© RJ Chevance

Comparing the Performance of Architectural Approaches(4)

- **Evolution of Architectural SMP and Cluster Efficiency with TPC-C (source data TPC)**



Page 24

© RJ Chevance

Taking performance into account in projects trhough modelling

© RJ Chevance

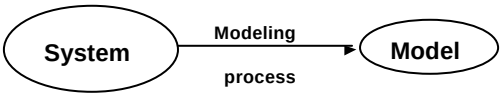
Modelling

- **Modeling is a good trade-off between intuition and measurements**
  - **More reliable than intuition**
    - **Reflects system's dynamics**
    - **Side effects can be detected**
  - **More flexible than measurements**
    - **Significant changes require only changes to the model or its parameters**
    - **Does not require the system to exist**
- **Advantages**
  - **Reliable results**
  - **Rigorous formalism**
  - **A global view of the system, its architecture and its behaviour**
  - **Improved communication between teams**
- **Obstacles**
  - **Little-known area**
  - **Knowledge of tools is not widespread**
  - **The necessity of understanding how the system works**
  - **Difficulty in getting needed data**
  - **The temptation to go measure instead**

© RJ Chevance

Modelling(2)

- **Concept of a model**

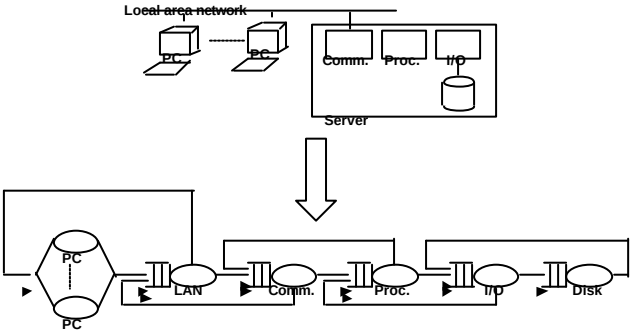


- **Several techniques for modelling:**
  - **Operational analysis**
  - **Queueing networks**
  - **Stochastic Petri nets**
  - **Discrete event simulation**
- **In this presentation, we will concentrate on queueing networks and operational analysis**

© RJ Chevance

Modelling(3)

- **Concept of a queueing nework**

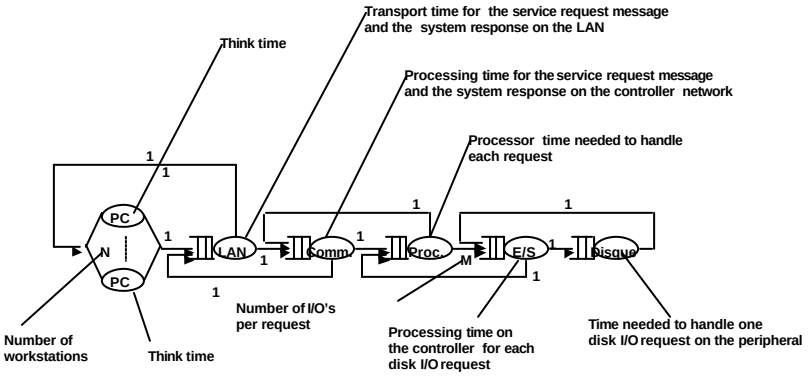


  - **Basic concepts**
    - **Customers and classes of customers**
    - **Stations (server + queue) and service**
    - **Transitions**

© RJ Chevance

Modelling(4)

- **Queueing Network Parameters**

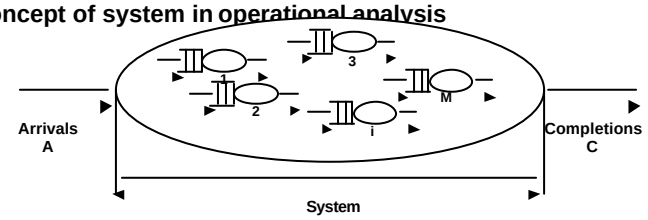


  - Solving methods for queueing networks: exact methods (analytical methods, Markov chains); approximate mathematical methods; or discrete event simulation e.g. MODLINE (www.simulog.fr)
  - Résults :
    - Average response time
    - Throughput
    - Busy factors for the stations
    - ....

© RJ Chevance

Operational Analysis

- **Operational Analysis**
  - A first approach to modelling and a means to verify homogeneity of measurements
  - Introduced by J.P. Buzen (1976) and improved by P.J. Denning and J. P. Buzen (1978)
- **The term operational simply means that the values used in the analysis are directly measurable**
  - with a single experiment
  - with a finite time duration
- **Operational analysis establishes relationships between the values measured**
- **The values used in operational analysis are those produced by tools such as Performance Monitor (Windows) or vmstat and sar (UNIX)**
- **Concept of system in operational analysis**



© RJ Chevance

Operational Analysis(2)

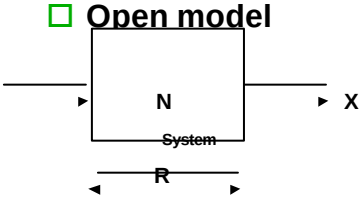
- **Operational analysis assumes that the following assumptions are true:**
  - **Assumption 1: arrival/completion equilibrium - the total number of arrivals is matched by the number of completions (this assumption is known as the job flow balance or Flow Equilibrium Assumption)**
  - **Assumption 2: stepwise change - the state of the system changes only as result of an event at a time, arrival or completion**
  - **Assumption 3: resource homogeneity – the completion rate of a resource depends only on the arrival rate (provided the resource is not saturated). At saturation, the completion rate depends on the service time of the saturated resource.**
  - **Assumption 4: homogeneous routing – the routing of requests inside the system is independent of the state of the system**

Page 31

© RJ Chevance

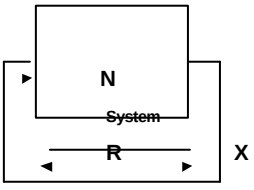
Operational Analysis(3)

- **Types of Models**
  - **Open model**

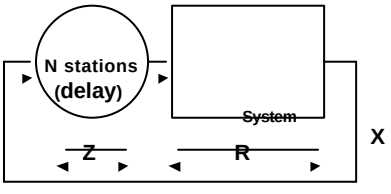


Notations :
N : Number of clients in the system or number of stations
X : System throughput (clients/s)
R : Response time
Z : Think time

  - **Closed model**



b) - Base closed model



c) - Closed model with a delay-type station

Page 32

© RJ Chevance

Operational Analysis(4)

- **Operational quantities (observed at resource i during the observation period T):**
  - **Arrival rate λ_i = number of arrivals/time = A_i/ T**
  - **Average throughput X_i = number of completions/time = C_i / T**
  - **Utilization U_i = total utilization busy time/time = B_i / T**
  - **Average service time S_i = total service time/number of clients served = B_i /C_i**
  - **Service demand a resource i: $D_i = S_i \ V_i$ (client service requests multiplied by the visit ratio to the resource), V_i is defined as the ratio of the number of completions by resource i and the total number of completions by the system, denoted C_0 (thus $V_i = C_i / C_0$). In other words, V_i is the number of visits to the resource i within the framework of the execution of a request**
  - **Total service demand $D = \sum_{i=1}^{M} D_i$**
  - **X, which is defined as C_0/T, is the average throughput of the system**

© RJ Chevance

Operational Analysis(5)

- **Operational Laws**
  - **Utilization Law**
    - **$U_i = X_i S_i = X D_i$**
  - **Forced Flow Law**
    - **$X_i = X V_i$**
  - **Little's Law defines the average number of clients Q_i in station i with response time R_i and throughput X_i**
    - **For a closed system:**
      - **$Q_i = X_i \ R_i$**
    - **For an open system**
      - **N = X R**
  - **General Response -Time Law**
    - **$R = \sum_{i=1}^{M} R_i V_i$ where R is the global system response time**
  - **Interactive Response Time Law (for networks with a delay-type station)**
    - **R = (N/X) - Z where N is the number of users and Z is the think time (at the delay-type station)**

© RJ Chevance

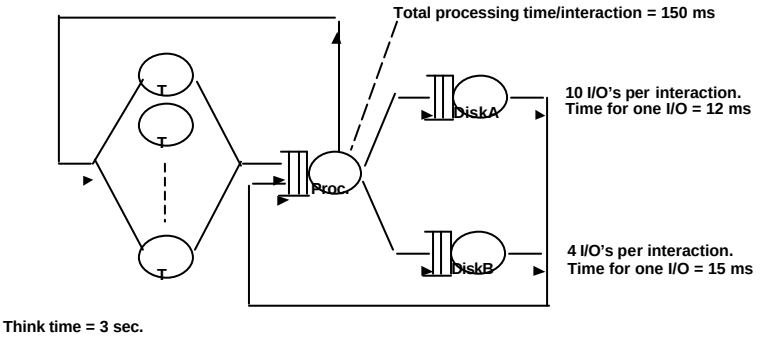
Operational Analysis(6)

- **Bottleneck Analysis**
  - Let D_{max}, the service demand on the most heavily-loaded resource
    - $D_{max} = \max_i \{D_i\}$
  - Limits
    - For a open system
      - Throughput: $X <= 1/ D_{max}$
    - For a closed system (with a delay-type station)
      - Throughput: $X(N) <= \text{Min} \{ 1/ D_{max} , N/(D + Z)\}$
      - Response time: $R(N) >= \text{Max} \{D, N D_{max} - Z\}$
    - Maximum number of users (in a closed system with a delay-type station) before saturation starts (saturation knee):
      - $N^* = (D + Z) / D_{max}$

© RJ Chevance

Operational Analysis(7)

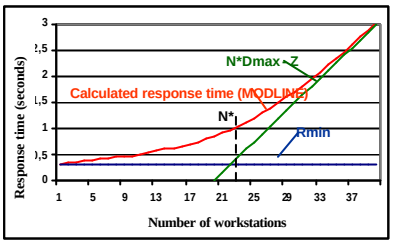
- **Example of an interactive system**

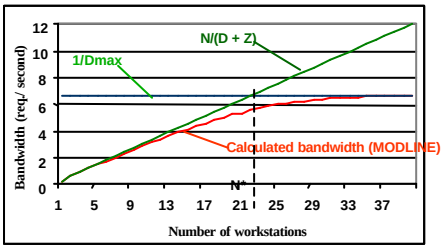


© RJ Chevance

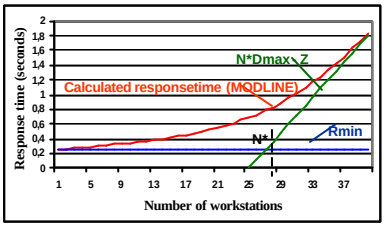
Operational Analysis(8)

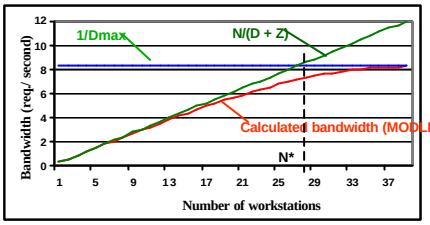
- **Interactive System Behaviour**





- **Interactive System Behaviour After Doubling Processor Performance**





Page 37

© RJ Chevance

Modelling Process

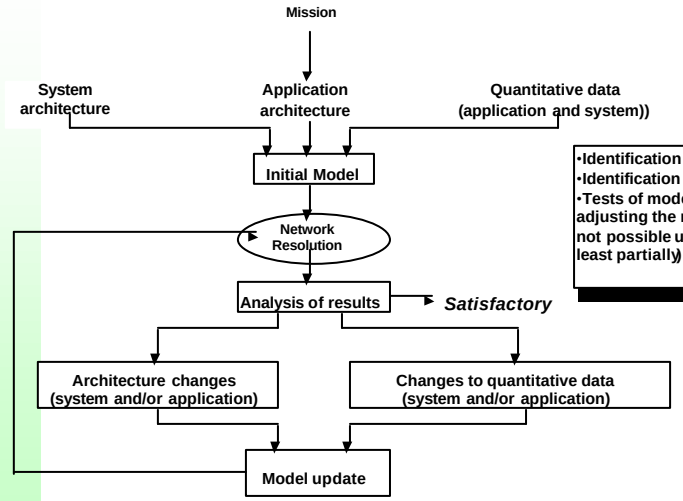
- **From a schematic point of view, two extreme cases can be considered (real world situations are closer ot the second one):**
  - **Blank sheet of paper development**
    - **define the system's mission and select an architecture for it**
    - **develop a model of the system**
    - **study the behavior and properties of this model**
    - **validate the application structure and system architecture using the model**
    - **validate the model as soon as possible against the actual system**
    - **Keep adjusting the model as needed until implementation is complete**
  - **Development starting from a set of existing components**
    - **having obtained a first-level understanding of the system, built a first model**
    - **adjusted the model to match the observed behavior of the system**
    - **once there is a good match, it is possible to study the effect of changes by introducing them into the model rather than the system.**
    - **as the system evolves, the model must be kept up to date (for example, to match changes made through maintenance activities, or through the effect of replacing obsolete components)**

Page 38

© RJ Chevance

Modelling Process: Ideal Case

Design Phase

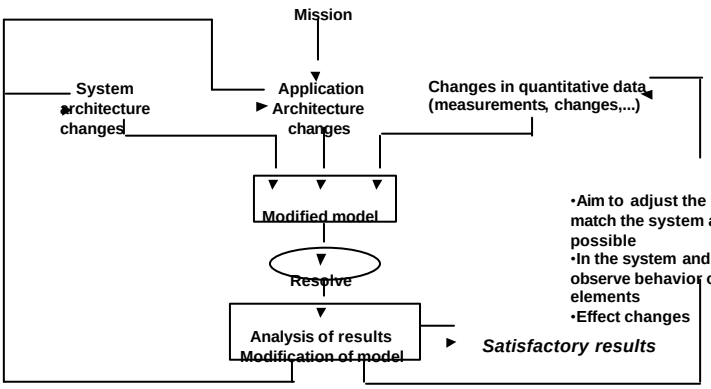


- Identification of the critical elements
- Identification of what to measure
- Tests of model robustness (since adjusting the model to match reality is not possible until the system exists, at least partially)

© RJ Chevance

Modelling Process: Ideal Case

Implementation Phase

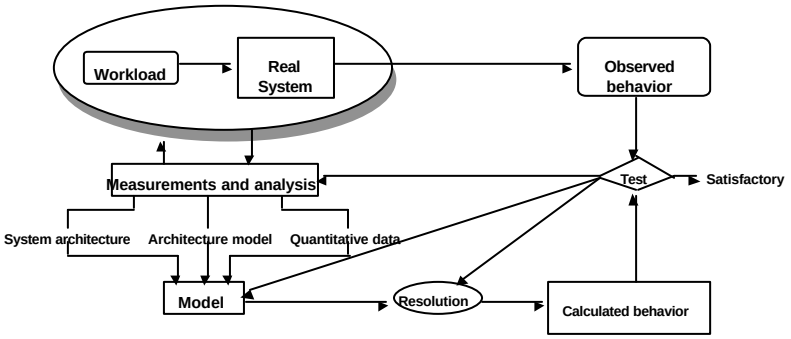


- Aim to adjust the model to match the system as soon as possible
- In the system and the model, observe behavior of the critical elements
- Effect changes

© RJ Chevance

Modelling Process: Existing System

- **Modeling an Existing System**
  - The objective is to obtain a fitted model for an existing system whose architecture, mission and associated quantitative data are not totally known at the starting point)

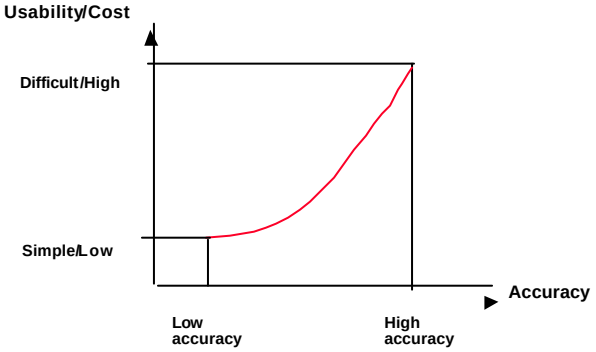


  - Don't expect to obtain a model of a system without an understanding of the system architecture and its dynamics. This can be accomplished by a variety of means, including:
    - interviews
    - code analysis
    - execution and I/O traces

© RJ Chevance

Modelling Strategy

- **Search Space for an Appropriate Model**



Properties of a « good » model:

- Utility
- Homogeneity
- Simplicity
- Comprehensibility

© RJ Chevance

Modelling Strategy(2)

- Steps for defining a modelling strategy:
  - Looking at each system component and deciding whether it should be modeled analytically or with simulation
  - Defining an appropriate hierarchy of models, from simulation (if necessary) to analytic at the highest levels.
  - Defining the models and the parameters they require for characterization
  - Obtaining numerical values for the parameters through either measurement or estimation (any estimates should be checked later)
  - Evaluation and adjustment of the models, progressing from lowest level up to the most complete
  - Evaluation and adjustment of the whole model
- At the end of the process, the resulting model represents the system well enough, and changes to mission, configuration, sizing, hardware components etc can be studied on the model in confidence that the behavior it shows is a good prediction of what will happen on the system

© RJ Chevance

Empirical Rules of Systems Sizing

- This presentation is based on the work of Jim Gray and Prashant Shenoy « Rules of Thumb in Data Engineering »
- Consequences of the Moore's law (density of integrated circuits doubles every 18 months):
  - Every 10 years, the storage capacity of a DRAM memory increases 100 fold
  - A system will need an extra bit of physical address space every 18 months
    - 16 bits of physical address in 1970
    - 40 bits in 2000
    - 64 bits in 2020/2030?
- Evolution of magnetic disks:
  - Capacity: x 100 between 1990 and 2000
  - Over a 15 years period:
    - speed x 3
    - throughput: x 30
    - form factor: / 5
    - Average access time: /3 (about 6ms by 2002)
  - Internal throughput: 80 MB/s
  - Disk scanning time (e.g. Unix *fsck*): ~45 minutes
  - Price : $0.011/MB (2005)

© RJ Chevance

Empirical Rules of Systems Sizing(2)

- **Evolution of magnetic disks(ctd):**
  - ratio of disk capacity to the number of accesses per second increases by a factor of 10 every 10 years
  - the ratio of disk capacity to the disk throughput increases by a factor of 10 every 10 years
- **Implication: disk access is a critical resource**
- **Possible actions to counteract the problem:**
  - intensive use of caching techniques
  - the use of log-structured file systems, which replace multiple updates to portions of a file with the writing of large, contiguous, sequential blocks of data at the end of a file
  - the use of disk mirroring (RAID 1) rather than RAID 5 - an update on a RAID 5 storage subsystem requires the reading of data distributed across several disks to allow parity calculation and then the writing of both the data and the parity information

© RJ Chevance

Empirical Rules of Systems Sizing(3)

- **Historically, there has been a price per megabyte ratio of 1:10:1000 between memory, disk and tape ($1 of tape holds $10 of disk data and $1000 of memory data)**
- **Now, with tape robots, these ratio are 1:3:300**
- **Consequences:**
  - the preferred storage place for information is moving from tape to disk, and to some extent from disk to memory (although the use of memory presents the problem of memory loss when electrical power is lost, as well as the problem of providing access to the memory by another system in the case of failure of the system containing the memory)
  - use of a disk mirror (RAID 1) as backup instead of tape, with the mirror being remote to avoid failure in the event of a catastrophe.
  - In 1980, it was reasonable to have one person allocated per gigabyte of data. By 2005/2006, we can generally expect a ratio of around 20 TB per administrator. Note: this ratio is very application-dependent

© RJ Chevance

Empirical Rules of Systems Sizing(4)

- **Derived from the rules stated by Gene Amdahl (1965):**
  - **Parallelism: if a task has a sequential portion taking S to execute and a parallelizable portion which takes P to execute, then the best possible speedup is (S + P)/S**
  - **System balance: a system needs about 8 MIPS of delivered processor performance for each MB/sec of I/O throughput. (the profile of the workloads must be taken into account e.g. a TPC-C workload will cause a processor to average around 2.1 clocks per instruction executed, while TPC-H will let it run at about 1.2 clocks per instruction)**
  - **Memory/Processor: a system should have about 4MB of memory for every MIP delivered by the processor (this used to be 1 MB; the trend to increasing MB/ MIPS is expected to continue)**
  - **I/O: a system will execute about 50,000 instructions per I/O operation for a random I/O profile, and about 200,000 instructions per I/O operation for a sequential I/O profile**

© RJ Chevance

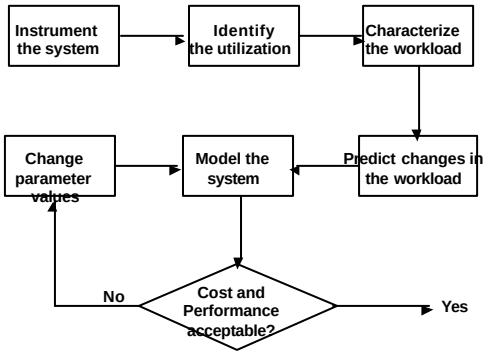
Empirical Rules of Systems Sizing(5)

- **Gilder's Law (1995)**
  - **Installed throughput triples every year**
  - **Interconnect throughput quadruples every 3 years**
- **I/O**
  - **A network message costs 10,000 instructions plus 10 instructions per byte**
  - **A disk I/O costs 5.000 instructions plus 0.1 instructions per byte**
  - **The cost of a SAN message is 3000 clocks plus 1 clock per byte**
- **Management of RAM-based disk caches (Jim Gray)**
  - ***Five-Minute Rule for random I/O*: for a disk whose pages are accessed randomly, cache the pages which are re-used within a 5-minute window**
  - ***One Minute Rule for sequential I/O*: for a disk whose pages are accessed sequentially, cache the pages which are re-used within a 1-minute window**

© RJ Chevance

Capacity Planning

- **Objective: satisfaction of the needs of future use by predicting the extensions needed by the system, the approach being based on measurements of the existing system**
- **Two possible approaches:**
  - **based on of a model of the system**
  - **based on system measurements and their expected changes**



© RJ Chevance

References

- **Tim Browning « Capacity Planning for Computer Systems » AP Professional 1995**
- **René J. Chevance « Serveurs Architectures: Multiprocessors, Clusters, Parallel Systems, Web Servers, Storage Solutions » Digital Press December 2004**
- **René J. Chevance « Serveurs multiprocesseurs, clusters et architectures parallèles » Eyrolles 2000**
- **Jim Gray, Prashant Shenoy « Rules of Thumb in Data Engineering » IEEE International Conference on Data Engineering San Diego April 2000**
- **Raj Jain « The Art of Computer Systems Performance Analysis » John Wiley & Sons 1991**
- **Daniel A. Menascé, Virgilio A. Almeida « Capacity Planning for Web Services » Prentice Hall 2002**

- **Web Sites :**
  - **http://www.spec.org (System Performance Evaluation Corporation - SPEC)**
  - **http://www.tpc.org (Transaction Processing Council - TPC)**
  - **http://www.StoragePerformance.org (StoragePerformance Council - SPC)**
  - **Outil de modélisation http://www.simulog.fr (Simulog)**
  - **For performances of workstations:**
    - **http://www.bapco.com**
    - **http://www.madonion.com**
    - **http://www.zdnet.com**

© RJ Chevance