# PERFORMANCE CONTROL OF RDBMS APPLICATIONS

Beatrice RUMPLER and Mario POLO and Benjamin RAZAFIMANDIMBY
INSA of LYON-Informatique
20, av A. Einstein
F 69621 VILLEURBANNE CEDEX
FRANCE

Tel : +33 72 43 83 92
Email:rum@ifhpserv.insa-lyon.fr

Abstract :

The goal of our research is to predict performance of applications using ORACLE RDBMS and then to propose tools to optimize performance.

The performance we are interesting in, is the performance as perceived by users. We then specially study user's transactions response time.

Our method is based on measurement, and the first step was to measure performance on existing applications using ORACLE RDBMS. We have developed several software tools:

- user simulator
- application generator
- workload application generator
- measurement tools to measure user transactions response time and system activity during a transaction execution.

The second step consisted in data collection and data analysis of measures. The data analysis, based on statistic methods, has permitted to extract the most influent factors and to understand how they can enhance applications performance.

We are now able to present the most part of these results.

The last step will consist in building the rules of an expert system for configuration and tuning assistance of ORACLE RDBMS applications. We also analyse the impact of operating system (UNIX) parameters on performance, and these information will complete our expert system possibilities.

The present paper will describe precisely this research with tools developed, methods used, and results.

Keywords: application development tools, performance engineering, simulation, operating system, RDBMS, data analysis, expert system.

## I - INTRODUCTION

Nowadays, we can find many tools for computer system performance analysis, but we lack of tools for application performance analysis. We can't predicate application performance and it's only when the final implementation is done that users discover applications response time (Coyette, 1987) (Rumpler, 1990) (Boudigue, 1990).

At this step, it's often too late to significantly improve performance, or it's by feeling that system and application parameters are modified.

The goal of our research is to predicate performance of ORACLE RDMBS applications under UNIX and to propose tools to optimize performance.

The performance we are interesting in, is the performance as perceived by users. We specially study users transactions response time.

Application performance depends on many factors. We have limited our study in analysing the influence of :

-operating system and ORACLE RDBMS configuration parameters
-workload generated by transactional applications and by the number of users simultaneously connected to the database.

Others points, such as tables or transactions requests structuration have not been integrated in our research, although their impact on performance are notable.

Our method is based on measurement. The first step of our research was to be able to measure performance of existing applications and the system activity during execution. Then we proceeded to analyse these results to detect the most influent parameters on application performance and their impact according to the configuration.

We have considered hardware configuration and software configuration for both of operating system (UNIX) and RDBMS ORACLE.

The results of data analysis will permit us to define the rules of the configuration assistance system expert for ORACLE RDBMS applications under UNIX

## II-METHODOLOGY

We have chosen a quantitative approach based on measurement and modelling (Ferrari, 1978) (Devarakonda and al, 1989) (Ammar and al, 1992).

Measurement permits to obtain data directly from the system and the evaluated application. These data give information about system activity and application performance, these information are used to make a model of the system (Bui and al, 1990). So, in our approach, measurement and modelling are closely coupled .

Our methodology include five steps:

- realisation of software measurements tools:
  . a ***spy*** for measurement of transactional request response time.
  . a ***monitor*** for measurement of system activity during a transactional request execution.
-realisation of software tools for experiment environment:
  . a ***user simulator*** able to memorize an reproduce an operator behaviour during ORACLE applications execution.

. a ***workload generator*** which provides several applications running simultaneously with various number of users connected to these applications.

. an ***automatic generator*** of RDBMS tables for applications.

- data collection :

The software tools developed are used for experimental design and data collection. Data are collecting from the operating system and the measured application according to various experiment context. The workload generated by running applications, by the numbers of users simultaneously connected, by the kind of transactional request measured and so on ... can be automatically modified from a data collection to another. We used factorial method for our experimental design.

- data analysis:

We proceeded to two kinds of analysis :

. First, we analysed the whole data collected by means of methods such as principal component analysis (P.C.A.). This method permitted us to verify the good correlation between a measured request response time and an estimator of the same request response time. The estimator is calculated using the all data collected. It also permitted to eliminate wrong measures.

. Then, we proceeded to individual analysis on each kind of transactional request measured. This step has permitted to detect the most influent measured factors on application performance. We also precisely analysed the system functioning according to each kind of measured request and then we have extracted for each of them, the main correlated part to response time.

- expert system realisation:

The results obtained from data analysis will be used to build the rules for the configuration assistance system expert. This is the next step of our research.

## III MEASUREMENT TOOLS

The software tools built to perform measurement in various experimental designs are presented in this paragraph.

### III-1 Tools for transactional request response time measurement:

In figure 1, we present a system architecture where performance of an ORACLE RDBMS application (more precisely a SQLFORMS application) is measured while others applications are running. Measures are done by our transactional spy and our software monitor.

The ***transactional spy :***

This tool is the first element of our measurement environment (Polo,1990) (Rumpler and al, 1991).

The spy is a chronometer, it is activated when an operator command is launched and it is stopped when the result is obtained on the screen. This tool is integrated to the system.

Our measurement environment software first starts the SQLFORMS application to be measured, then it gets the SQLFORMS application process PID number (Process Identificator). After this, it starts the spy, giving it the SQLFORMS application's PID number. Using the PID number of the SQLFORMS application to be measured, the spy finds in the processes table the SHADOW's process PID number (The SHADOW process is started by the SQLFORMS application execution).

We also studied in details the mechanisms launched between the database, the operating system and the SQLFORMS application. We noticed that, for every transaction, a dialogue is installed between the SQLFORMS process and the ORACLE SHADOW process (Razafimandimby, 1992).

The SQLFORMS process retrieves the transaction started by the operator before to transmit it to the SHADOW process which handles it, then SQLFORMS gets results from the SHADOW process and displays the result.

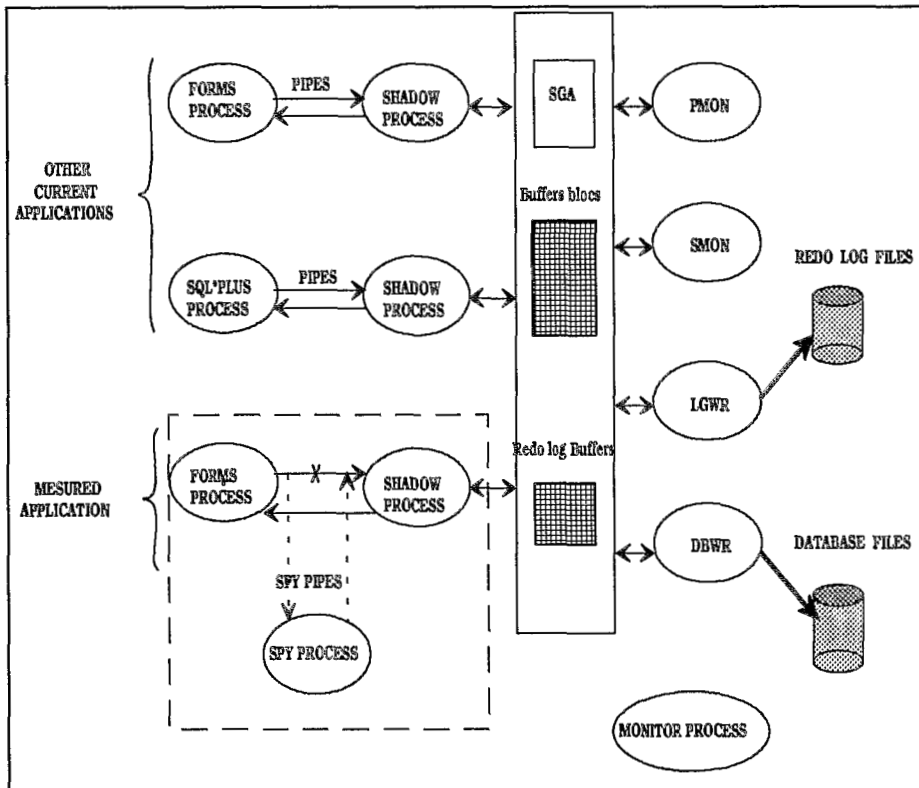


Figure 1 : System architecture

The communication between the SQLFORMS and SHADOW processes is performed by two UNIX pipes. It consists in a frame set that we have to control in order to determine the starting and stopping instant of response time measurement. From the UNIX processes table we got the PID processes numbers and the system pipes descriptions.

We decided to reroute the exchanges of the two pipes in our spy process to control the frames sent between the SQLFORMS process and the SHADOW process. The pipe rerouting is performed by handling i-node pointers in the processes table. This new structure is presented in figure 2

So we successfully control the exchange between the SQLFORMS process and the SHADOW process.

By analysing the frames head during applications execution, we got the necessary information to detect the beginning of an operator transaction and the beginning of the database response display.

We can start and stop the spy at the best time to measure precisely a transaction response time.

This architecture lightly modifies the normal functioning of ORACLE, but without disturbance, because of a good control of system calls from UNIX core.

The ***software monitor*** :

This tool is a process which is started by the spy. It collects system information about the measured application while a transaction is executed. By regular intervals, the monitor memorizes system data of the measured application, such as CPU time, number of input/output, waiting time ....

### III-2 Commands set generator and user simulator tools

The commands set generator is a tool able to memorize operator behaviour during a SQLFORMS application execution. This tool scans database manipulation sequences of an operator and stores the commands with their keyboard input times. The principle consists in redirecting standard input toward the tool which store user commands in a file.

In other hand the generator can combine two and more operator behaviours. In this way, the tool extracts sequences from input commands set files, calculates new keyboard input time according to the last command keyboard input time and stores each command in output file. So different behaviours can be led automatically.

The user simulator reproduces user behaviour stored in commands set file. This tools determines time execution of each command with respect to the keyboard input times.

These tools are used by the automatic workload generator.

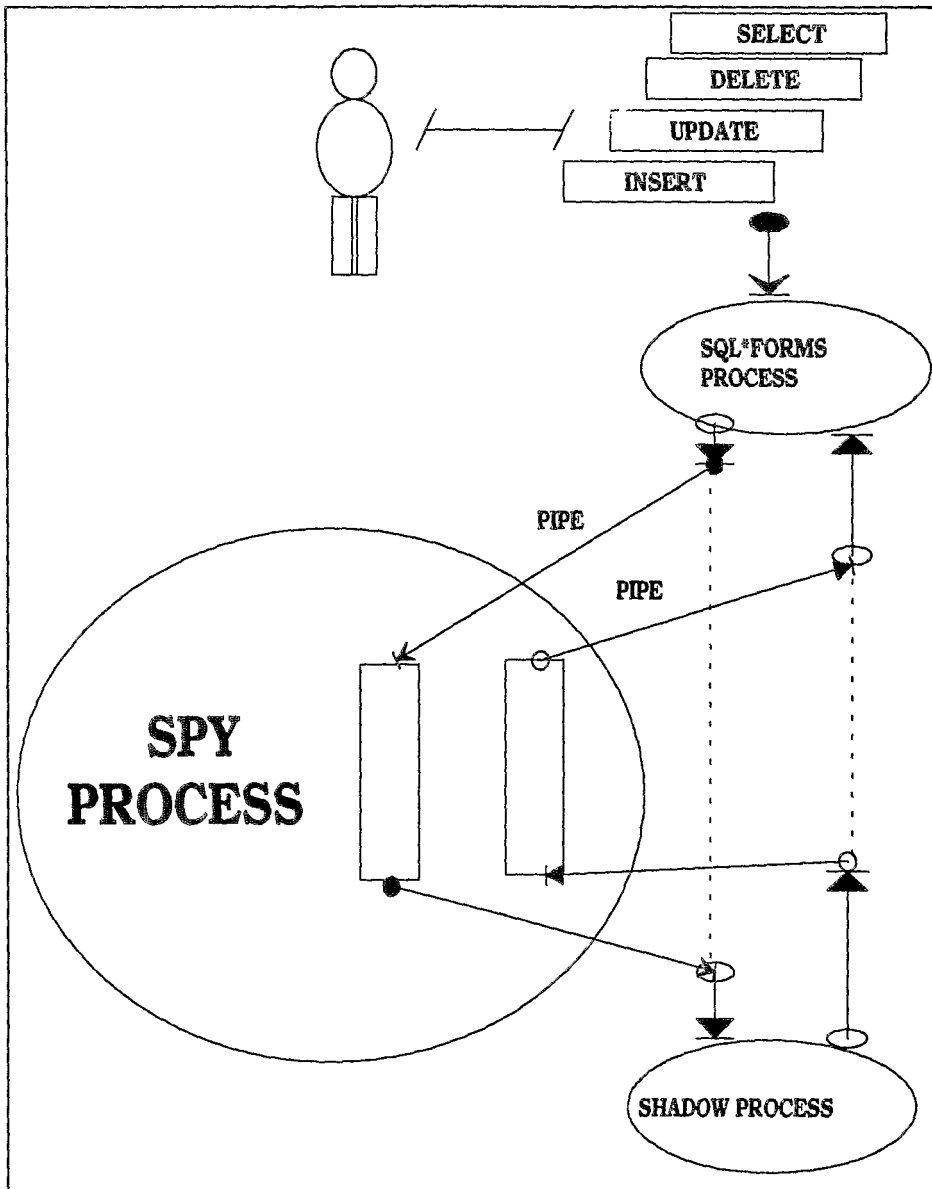


Figure 2 : Software architecture diagram

### III-3 Workload generator

In this study, we define how the operating system and ORACLE RDBMS workload is generated by the set of applications which are simultaneously running in real execution.

The workload is specified by the number of users which are connected to ORACLE RDBMS and the kind of application executed by each of them.

So, the automation of workload generation requires the giving of these two set of parameters which are the input data for the tool.

The tool manages connection of every user specified.

For every operator connected, the generator calls the user simulator tools in order to generate behaviours and, so, one part of workload.

## IV- DATA COLLECTION

### IV - 1 Experimental design

We decided to use a factorial design for data collection.

A performance study with k factors, with the $i^{th}$ factor having $n_i$ levels, requires n experiments, where :

$$n = \prod_{i=1}^{k} n_i$$

In our approach, we first selected the factors and the levels of those factors to be investigated (Rumpler and al, 1993).

Collections are made out in various operating environment. Each of these environments is characterised by three sets of factors :

- Factors which define UNIX operating system and ORACLE RDBMS
- Workload characteristics
- Database physical architecture

In another hand, to start the collection we have built a software application used for measurement. This application extracted from a real situation is about a sale management software in which customer, stock and invoicing are managed.

Then we have designed the different applications making up the all software.

This approach make us sure that usual database transactions such as :

- table operations : projection or join,
- data manipulation : insertion, modification and deletion operations,
- aggregate operations : sum, average, maximal value, ...

will be considered in the study.

At last, each application transaction is stored by the generator tool as a commands set file.

### IV - 2 UNIX operating system and ORACLE RDBMS configuration

There are more than a hundred configuration factors for the operating system and for the RDBMS. However, all these factors don't have influence on performance (Bach, 1986) (AT&T, 1988). So we have classified them in three sets :

- *static factors :* This kind of factors specifies critical resources. Bad configuration of memory size implies bad performance (swapping and

pagination phenomenon) (Elhardt and al, 1984). The DB_BLOCK_BUFFERS determines, as a static factor, the granted memory structure size (number of database cache buffers of the System Global Area).

- *dynamic factors :* Factors (like LOG_CHECKPOINT_INTERVAL) are used in dynamic system processing. Even if they have no impact in resource size, the choice of their values can generate bad functioning and bad performance.
- *other factors* which have no impact on performance but are required by the system.

We only study the two first sets and some factors can belong to both sets.

During the collection we have combined different values of these factors in order to obtain more than one operating system and RDBMS configurations.

| % Consultation Vs % Modification | NB users | Workload type |
|---|---|---|
| 100/0 | 20 | 1 |
| | 15 | 2 |
| | 10 | 3 |
| | 05 | 4 |
| 75/25 | 20 | 5 |
| | 15 | 6 |
| | 10 | 7 |
| | 05 | 8 |
| 50/50 | 20 | 9 |
| | 15 | 10 |
| | 10 | 11 |
| | 05 | 12 |
| 25/75 | 20 | 13 |
| | 15 | 14 |
| | 10 | 15 |
| | 05 | 16 |
| No workload | 00 | 17 |

Table 1: tested workloads

### IV - 3 Workload factors

Workload is characterized by:

- the number of users connected to the database and the number of applications simultaneously running.
- the type of transactions generated by each user. There are four elementary types of transactions : "SELECT", "UPDATE", "INSERT", "DELETE" operations which can be grouped in consultation request ("SELECT") and modification request ("UPDATE", "INSERT", "DELETE").

To study the impact of these two characteristics, we built various kinds of workload where :

- the ratio of users executing consultation request Vs users running modification request are (100%-0%), (75%-25%), (50%-50%), (25%-75%),
- the number of users simultaneously connected is changed from 5 to 20.

We present in table 1 the different workload tested during the data collection.

### IV - 4 Database physical architecture

Database architecture, characterised by tables size, is an important criteria for the data collection (Bing and al, 1987). We have built a *table generator* tool which carries out, automatically, the tables creation.

By this tool, during the data collection, we can load various tables automatically in order to collect information with different numbers of rows for each table (1000, 10000, 30000 and 50000 rows).

### IV - 5 Data collection procedure

For a given RDBMS and operating system configuration, for a fixed database physical architecture and a defined workload we proceeded :

- to memorize global system activity during the execution of the evaluated application. The software monitor tool samples, by regular intervals, statistics tables managed by UNIX. So, we get without disturbing system such information as :
  - number of running processes,
  - number of swapped processes,
  - CPU activity,
  - number of input/output,
  - ...
- to measure the impact of specific requests on database kernel performance. The transactional spy as explained in III-1 extracts and memorizes :
  - transaction response time,
  - consummated CPU time,
  - number of input/output,
  - ...

A collection tool is built to start workload generator with given characteristics and to activate transactional spy and software monitor.

For an easier result exploitation, we decided to register every collection with its own context in a descriptive data sheet.This file contains :

- operating system configuration factors,
- ORACLE RDBMS configuration factors,
- database physical architecture description,
- system workload characteristics,
- evaluated application specification and its commands set,
- collection results for each evaluated application transaction,
- activity system information collected during transactions execution..

Data collection make us repeating the same procedure for each environment specified, as shown in figure 3.

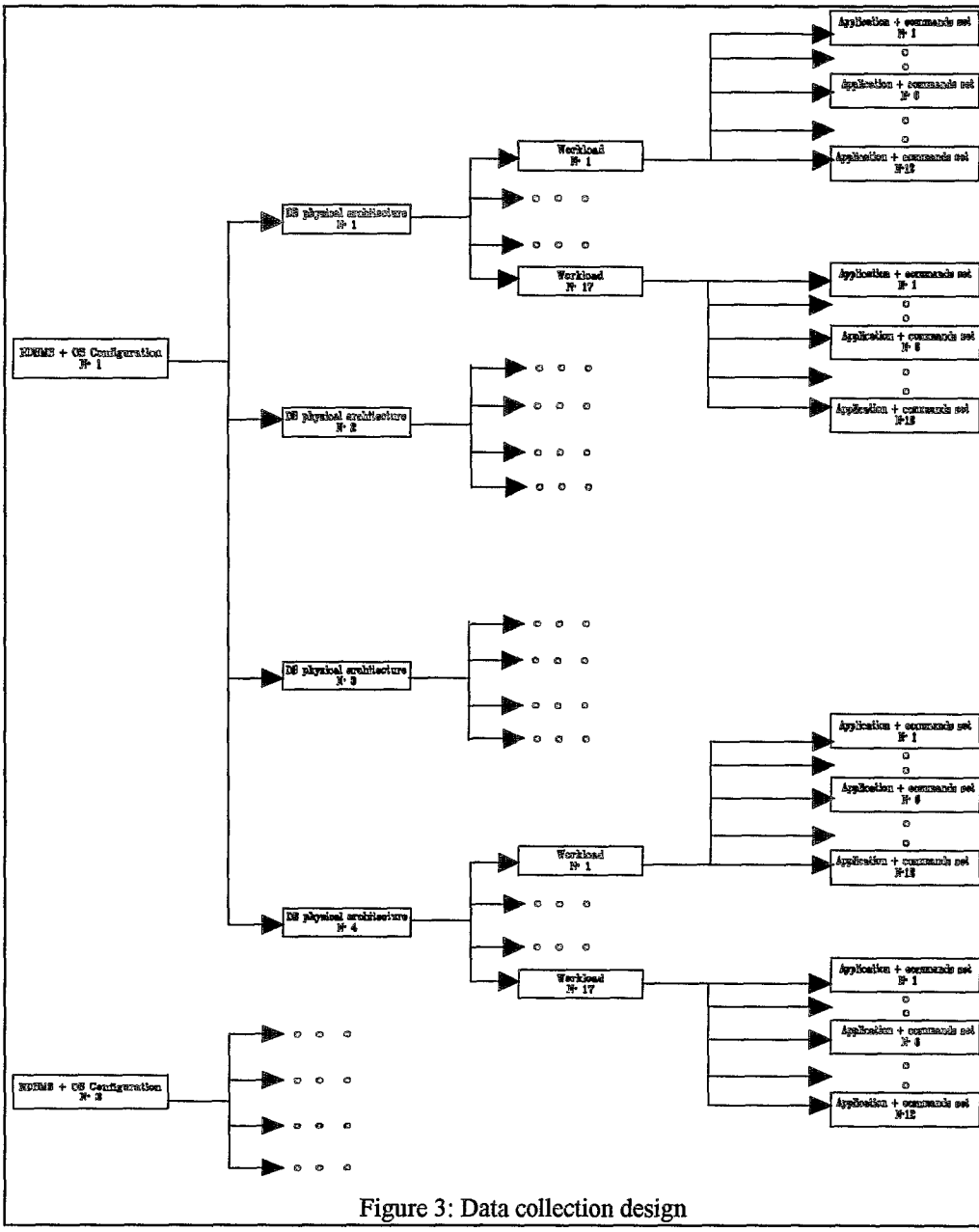

Figure 3: Data collection design

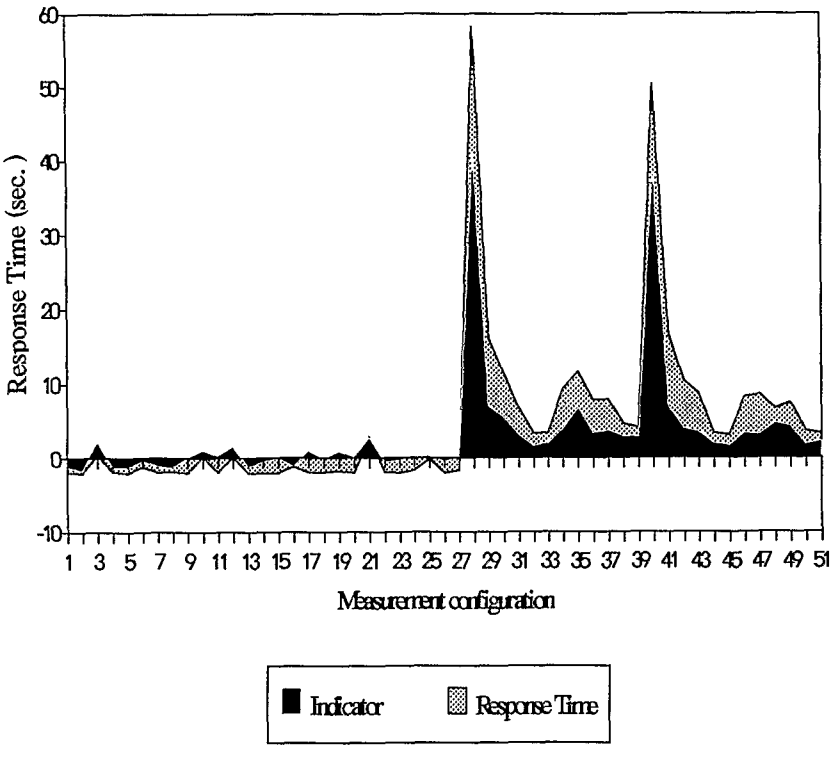


Figure 4: Response time estimator

## V - DATA ANALYSIS

### V-1 Principal Component Analysis

Principal Component Analysis (P.C.A.) has been applied on the whole measured parameters for each kind of SQL request type used in the measured transactions.

By this method we have calculated an estimator of a user request response time using the measured parameters. The figure 4 shows the good correlation between this estimator and the SQL request response time measured by the spy.

PCA also gives information about variation explained by the main axes.

Figures 5 and figures 6 respectively show the measured data plotted along the two first axes these figures represent. SQL INSERT and SQL SELECT request transactions.

In figure 5-a and figure 6-a, X axe and Y axe represents the axes the most correlated with the response time. In figure 5-b and figure 6-b, X axe is the most correlated with the response time and Y axe is the axe which has the best variation explained.

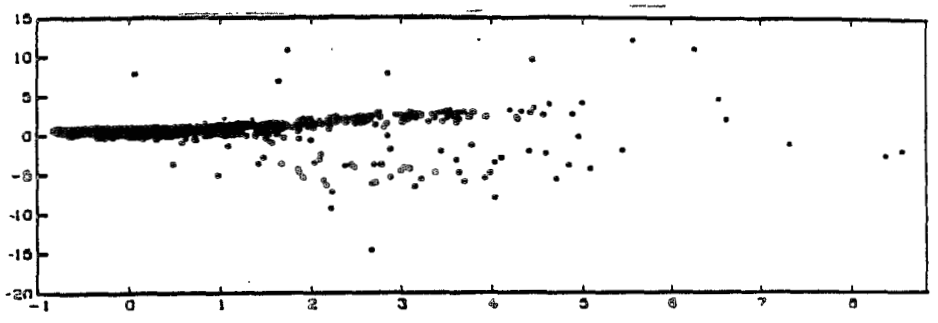

Figure 5-a : Most correlated axes with response time of "INSERT" SQL COMMAND

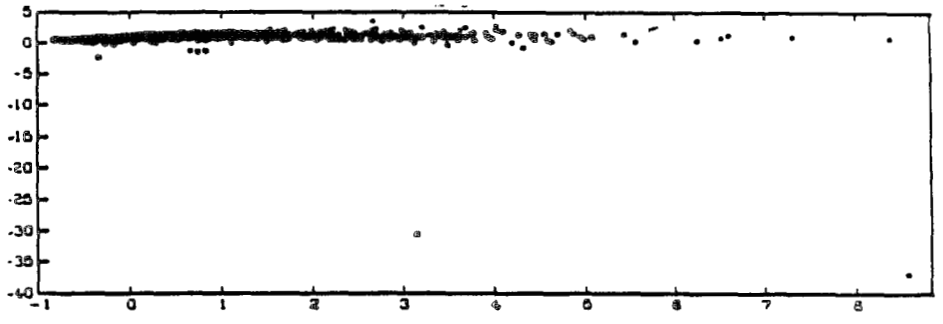

Figure 5-b : Axes of most variation explained of "INSERT" SQL COMMAND

These results have permit to eliminate wrong measures which only represent 0.08% among the all data collected. These representations show that the measured parameters and the factors used in experimental design are highly correlated with user request response time (Ahituv and al, 1988a) (Ahituv, 1988b).

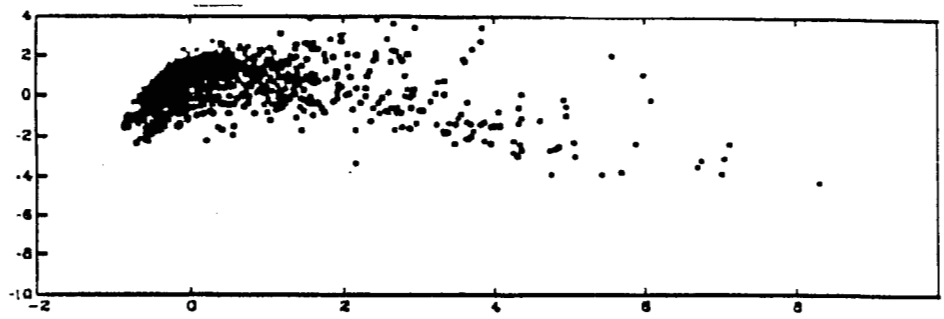

Figure 6-a : Most correlated axes with response time of "SELECT" SQL COMMAND

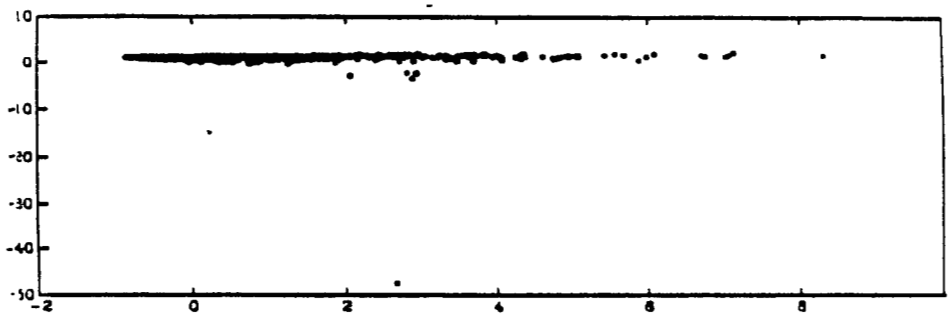

Figure 6-b : Axes of most variation explained of "SELECT" SQL COMMAND

### V-2 Impact of individual factors on specific user request response time

In figures 7, 8 and 9 the impact of four factors:

- SGA buffers numbers
- Numbers of users simultaneously connected
- Tables size
- Software workload characteristics

is represented for a SQL SELECT request response time.

In figure 7 and 8 we notice that user request response time increases with the number of users connected to the database and with the database tables size.

In every figures we can see that performance is worst when the SGA uses more of memory buffers. This can be explained by the increasing memory swapping operations.The SGA buffers number must be chosen according to the system memory global size and the number of applications.

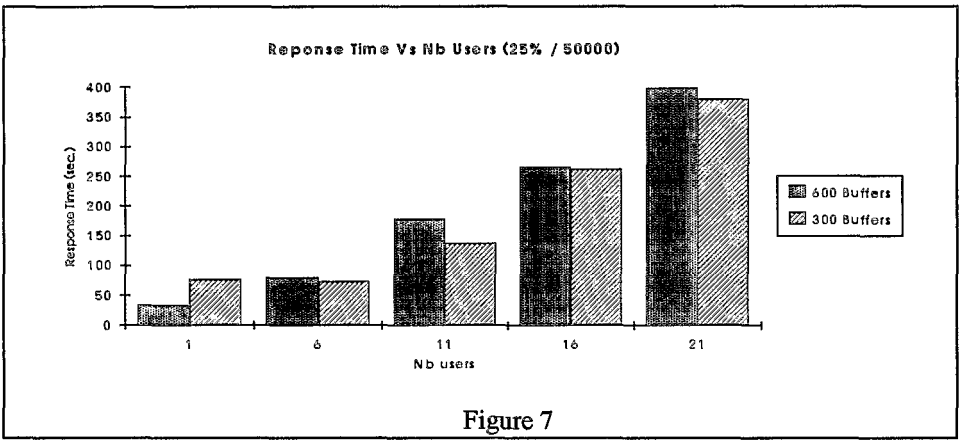


Figure 7

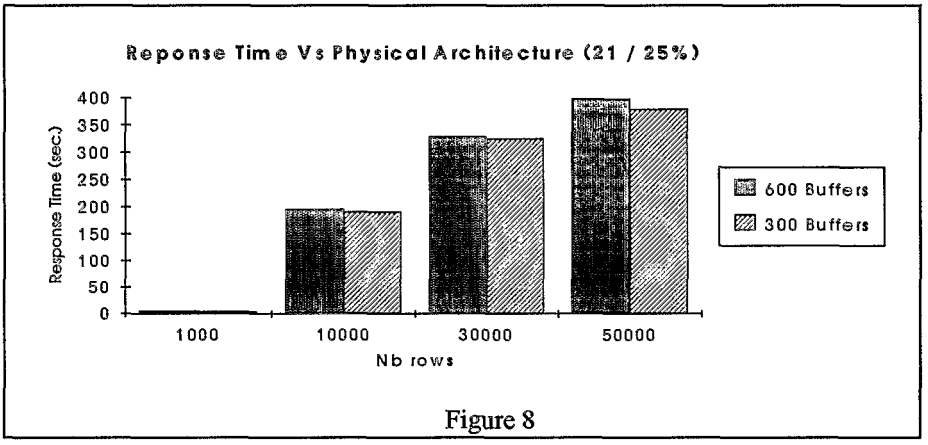


Figure 8

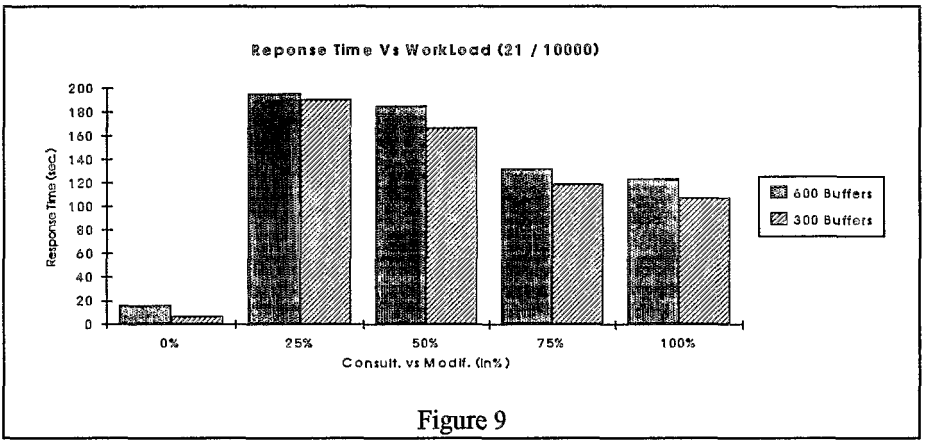


Figure 9

Figure 9 shows the impact of the application workload. By this representation, we can see that modification operations produce a more important workload than consultation operations.

In next part, we presents the results of analysis of the system activity during requests transaction execution.

### V-3 The response time Analysis

The response time perceived by the user is based on four elements :

- Physical I/O delay : this is the time consumed by the hardware part of the system (disk, controller card,...) to provide to the operating system the data blocks from the archive. This time is proportional to the number of blocks transferred from the disk and to the physical performance characteristics generally called "Access Time".

- System-CPU or Kernel-CPU time : this is the time consumed by the operating system to transfer the data blocks from the disk to the main memory. In the case of an ORACLE transaction, the System-CPU time represents the time spent in kernel mode by the ORACLE shadow process for the data block retrieving operation from disk to the process memory area. The System-CPU time is proportional to the number of logical blocks transferred ( NB logical blocks = NB physical blocks read + NB blocks cached ).

- User-CPU time : this is the time spent in user mode by the ORACLE shadow process during the SQL statement execution to load data in memory area. For example, the User-CPU time used for the ***"SELECT sum(SAL) from EMP"*** transaction depends on the *"EMP"* table characteristics ( NB blocks, NB rows, ....) and also depends on the SQL statement nature (sum operation).

- General activity delay time (overhead): this is the time generated by the operating system to manage the running processes. This delay depends on of the workload characteristic at the moment of the transaction processing.

The sum of the physical I/O delay, System-CPU time, and User-CPUtime gives the minimal theoretical request response time.

Our request response time estimation methodology is based on two steps :

- the first step consists in computing the minimal theoretical response time. The transaction we are interested in is considered as the only workload of the system during the execution time. The measures provide the basic performance of the system (like disk access time, processing time of different kind of SQL statement and so on ...). From this measures we extract the parameters of the system model to compute the minimal theoretical response time.

- the second step consists in estimating the influence of the global workload on the minimal theoretical response time found in the first step. From the data collection we have quantified the "overhead" phenomenon according to four factors presented in the paragraph V-2 and a model has been generated. From this model of "overhead", we compute the delay time introduced by the global workload. The sum of this delay time and the minimal theoretical response time gives an estimation of the final response time perceived by the user.

## VI- CONCLUSION

Our present research work has permitted to extract the main influent factors on transactional request performance in the working context and with the restrictions defined in paragraph I of this paper. These factors are: number of users connected to a database, database tables size, workload type and SGA size.

The analysis of the system activity during the execution of a transactional request has been quantified for several request types. Now we are going on with data analysis and modelling main parts of the system activity.

The next step of our research consists in building the rules for the configuration assistance system expert.

The final aims of this study are first to be able to propose the best RDBMS and Operating System configuration for ORACLE transactional application, then to propose tools to diagnostic and tune implemented applications.

## VII - REFERENCES


AHITUV, N., and IGBARIA, M, 1988a, "A model for predicting and evaluating computer resource consumption," *ACM: COMPUTING PRACTICES*, Vol. 31, N° 12, Décembre 1988, pp. 1467-1473.

AHITUV, N., et al, 1988b, "A compumetrical approach for analysis and clustering of computer system performance variables," *Comput. Opns. Res.*,Vol. 15, N°6, 1988, pp. 489-496.

AMMAR, R. A., and KRZYCH, M. J., 1992, "Computer aided performance engineering : a survey," *COMPUTER SYSTEMS SCIENCE & ENGINEERING*, Vol. 7, N°3, juillet 1992, pp. 170-189.

AT&T, 1988, "AT&T UNIX system V:Programmer's reference manual,"Prentice Hall International U.K.

BACH, J. M.,1986, "The design of the UNIX operating system" Prentice Hall International U.K..

BING YAO, S., and HEVNER, A., 1987, "Analysis of database Systeme Architectures using Benchmarks," *IEEE TRANSACTIONS ON SOFTWARE ENGENIERING*, Vol. SE-13, N° 6, Juin 1987, pp. 709-725

BOUDIGUE,D.,1990, "Un système d'évaluation de performances pour des applications relationnelles ," *PROCEEDINGS: TOULOUSE'90 , 3rd INTERNATIONAL WORKSHOP*, Vol. 1, pp. 321-333.

BUI, N., and SHIHAB, K., 1990, "Un outil d'aide a la decision pour l'évaluation de performances ," *PROCEEDINGS: 10 th WORKSHOP EXPERT SYSTEMS AND THEIR APPLICATIONS SPECIALIZED*, pp. 105-116.

COYETTE, L. et al., 1987,"Evolution of performance evaluation packages," *MODELLING TECHNIQUES AND PERFORMANCE EVALUATION*, FDIDA-PUJOLLE EDITION, Paris, pp. 311-321.

DEVARAKONDA, M. V. , and IYER, R. K., 1989, "Predictability of process resource usage: A mesurement-based study on unix," *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING*, Vol. 15, N° 12, pages 1579-1586

ELHARDT, K., BAYER, R., 1984, "A database cache for high performance and fast restart in database systems," *ACM Transactions on Database Systems*, Vol.9, N° 4, Décembre 1984, pp. 503-525.
FERRARI, D. et al., 1978, "Measurement and tuning of computer systems," PRENTICE-HALL, Englewood Cliffs,pp. 660.
POLO, M., 1990, "Outils de mesure d'une application développées sous SQL*FORMS," Mémoire de DEA en Ingénierie Informatique, INSA de LYON, LYON
RAZAFIMANDIMBY, B, 1992, "Environnement de mesure des temps de réponse des applications transactionnelles ORACLE sous UNIX", Mémoire de DEA en Ingénierie Informatique, INSA de LYON, LYON
RUMPLER, B., 1990, "Un exemple de prototypage avec ORACLE," *Revue d'automatique et de productique appliquées*, Vol 3, n° 3-4, pp. 105-122.
RUMPLER, B., POLO, M., 1991, "Outils d'aide à l'évaluation de performances d'applications générées à partir de SQL*FORMS (ORACLE)," *Revue d'automatique et de productique appliquées*, Vol 4, n° 4, pp. 467-482.
RUMPLER, B., POLO, M., 1993, "Configuration of transactional applications under UNIX", *PROCEEDINGS: ASME WAM NEW ORLEANS USA*, November 28 - December 3, 1993.