

The OLAP and Data Warehousing Approaches for Analysis and Sharing of Results from Dependability Evaluation Experiments

Henrique Madeira
University of Coimbra, DEI-CISUC
3030 Coimbra - Portugal
henrique@dei.uc.pt

João Costa and Marco Vieira
Polytechnic Institute of Coimbra, DEIS-CISUC
3031 Coimbra - Portugal
[\[jcosta,mvieira\]@isec.pt](mailto:[jcosta,mvieira]@isec.pt)

Abstract

Two important questions on experimental dependability evaluation remain largely unanswered: 1) how to analyze the usually large amount of raw data produced in dependability evaluation experiments and 2) how to compare results from different experiments or results from similar experiments across different systems. These problems are also common to other dependability evaluation techniques such as the ones based on simulation, or even to the analysis of field data on computer faults. We propose the use of data warehousing technologies to store raw results from different experiments/setups in a common multidimensional structure where raw data can be analyzed and shared world wide by means of web-enabled OLAP (On-Line Analytical Processing) tools. This paper describes how to use the proposed approach in a concrete example of dependability evaluation experiment.

1. Introduction

Experimental dependability evaluation has been extensively used to evaluate specific fault tolerance mechanisms, validate robustness of software components, or to assess the general impact of faults in systems. However, in spite of the effort put on the development of adequate tools and the intensive research devoted to the mitigation of key problems such as experiment representativeness, intrusiveness and portability of tools, just to name a few, two important questions remain largely unanswered:

- How to analyze the usually large amount of raw data produced in dependability evaluation experiments, especially when the analysis is complex and has to take into account many aspects of the experimental setup (e.g., target systems, configurations, workload, etc)?
- How to compare results from different experiments or results from similar experiments across different systems if the tools, data formats, and the setup details are different and, often, incompatible?

Many dependability evaluation tools, such as fault injection and robustness testing tools, have been proposed and built [1, 2, 3, 4, 5] but all these tools either provide

rudimentary means to analyze data or, more frequently, just store the raw results in a spreadsheet format such as the Microsoft Excel[®]. Although this approach can be acceptable for specific (and simple) analysis, it is clearly not enough when the amount of raw data is very large, the analysis required is complex, and particularly when heterogeneous data from different experiments has to be analyzed and cross-exploited.

A recent research effort is centered on the definition of dependability benchmarks as a standard procedure to assess measures related to the behavior of a computer system or computer component in the presence of faults [6]. Given the fact that the key goal of dependability benchmarking is the comparative assessment of dependability features across different systems or system configurations, the proposal of a general approach to analyze and cross-exploit raw results obtained from different dependability evaluation experiments became a decisive issue.

The central idea of the new approach proposed in this work is to collect the raw data produced in dependability evaluation experiments and store it in a multidimensional data structure (data warehouse). The data analysis is done through the use of commercially available OLAP (On-Line Analytical Processing) tools such as the ones traditionally used in business decision support analysis [7]. That is, instead of following the usual trend of adding data analysis features to fault injectors and robustness testing tools, we propose a clear separation between the experimental setup (target specific) and the result analysis setup (general in our approach). Existing tools and experimental setups are used as they are, and the only thing required is to export the data obtained in the experiments to the data warehouse, where all the analysis and cross-exploitation of results can be done in an efficient and general way. The key advantages of the proposed approach are the following:

- It is a general approach:
 - As the experimental evaluation of dependability is a multidimensional problem, our proposal of using a multidimensional database and OLAP technologies for result analysis is applicable to all the experimental dependability evaluation scenarios (see further on).
 - Any existing tools and experimental setup can be used; the only thing we need to know is the data format of

the raw results produced during experiments, in order to read them into the data warehouse.

- The results analysis setup is based on standard OLAP technologies, which are the general-purpose tools for the analysis of multidimensional datasets.
- It is easy to compare and cross-exploit raw results from different experiments, as all the raw data is stored in a common data warehouse.
- It is easy to share raw results world wide as the data stored in a data warehouse can be explored by web-enabled versions of OLAP tools. This way, it is possible to make available to the entire dependability community the raw data of dependability evaluation experiments (data is available for analysis through a web page).

The system presented in this paper is currently being used in the DBench project for the analysis and sharing of dependability benchmark results in several scenarios: analysis of raw data of single experiments, analysis and comparison of benchmark results obtained in different systems, and sharing and cross-exploitation of results among project partners. Some experimental examples and the corresponding raw results that can be analyzed through a web based OLAP tool can be found here: www.dei.uc.pt/~henrique/DBenchDW/examples. We see this possibility of making the raw results available to the dependability community, together with a tool to analyze them, as an important step to increase the exchange of results among researchers and practitioners. This way, people can not only read the traditional paper presenting the final results, but they will also have access to the raw data and they will be able to use that data for other purposes or to compare with their own results.

This paper is organized as follows: section 2 provides some background on data warehousing and OLAP and presents the proposed approach in more detail. Section 3 illustrates the approach with a concrete example. Section 4 explains how to use the proposed approach and section 5 proposes some possible scenarios to use this idea. Section 6 concludes the paper.

2. DBench-OLTP specification outline

Data warehousing refers to “a collection of decision support technologies aimed at enabling the executives and managers to make better and faster decisions” [8]. A data warehouse is a global repository that stores large amounts of data that has been extracted from heterogeneous systems. OLAP (On-Line Analytical Processing) is the technique of performing complex analysis over the information stored in a data warehouse [7]. The data warehouse coupled with OLAP enable decision makers to analyze and understand business trends and to transform raw data into strategic decision making information.

In data warehousing the data is organized according to the multidimensional model, which includes two kinds of data: facts and dimensions. Facts are numeric or factual data that represent a specific business or process activity and each dimension represents a different perspective for the analysis of the facts. Each dimension is described by a set of attributes. For instance, in the classical example of a chain of stores [7] represented in figure 1, some of the dimensions are products, stores, and time while the facts (the small cubes) contain the total sales, profit, etc for a given product in a given store on a single day. Note that the facts are just numerical quantities and only acquire meaning when referenced to the dimensions. Normally, there are more than three dimensions and the dimension attributes represent a detailed description of the dimensional data. The OLAP analysis over a multidimensional cube consists of dicing and slicing the data in order to compute the desired measures.

Normally, data warehouses store the data in a relational database. That is, the multidimensional model is implemented as one or more star scheme [8] formed by a large central fact table surrounded by several dimensional tables related to the fact table by foreign keys.

Typical data warehouses are periodically loaded with new data that represents the activity of the business since the last load. For example, at the end of the day, the data

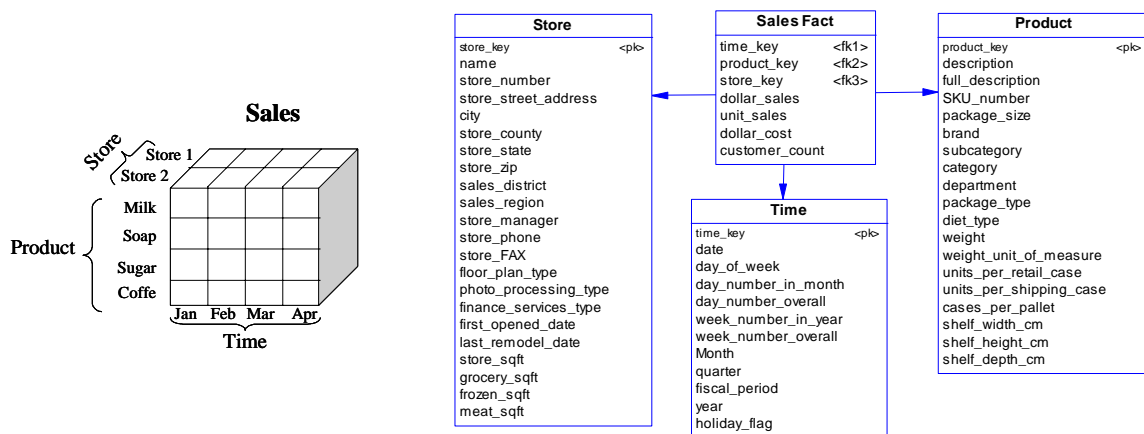


Figure 1 – Simple example of multidimensional model: logical view (left) and a physical star schema (right).

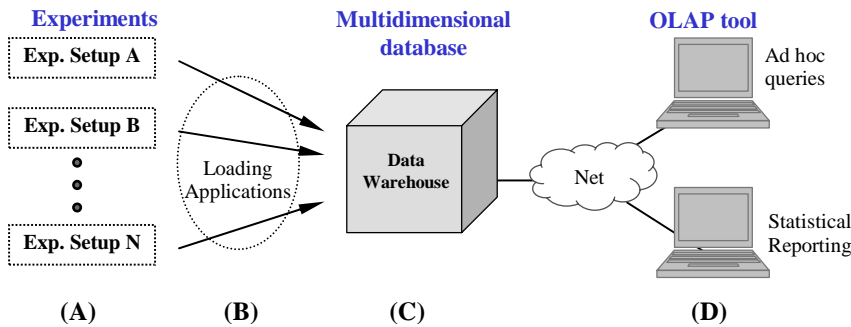


Figure 2 - Basic elements of the proposed approach.

that represents the daily business activity is extracted from the specific systems where it is produced and loaded into the data warehouse. In practice, the raw data came from several sources and it is necessary to introduce some transformations to assure data consistency, before loading that data into the data warehouse. The interested reader can find a complete course in [7].

Our proposal consists of adapting the data warehousing approach to the experimental evaluation of dependability. The readouts collected during the experiments represent the data facts of the multidimensional model, while key features of the experiment represent the dimensions. For example, raw data representing things such as error detection efficiency or error recovery time are facts, while sets of attributes describing the target systems, the different configurations, the workloads, the faultloads, etc represent the dimensions of the model. Figure 2 shows the key elements of the proposed approach:

Experimental setups (A area in fig. 2) – They are used as they are. That is, you can use your favorite tool and do the experiments in the usual way. What is needed is to have access to the raw results produced during the experiments and know the format of those results.

Loading applications (B area in fig. 2) – Normally, the loading applications are specific programs that read the raw data into standard database tables for transformation and final load into the data warehouse. However, we have developed a general purpose loading application because it is relatively easy to interpret the format of the files/tables normally used by fault injectors and other dependability evaluation tools.

Data warehouse (C area in fig. 2) – After the necessary multidimensional analysis in order to identify the required star schema (facts and dimensions), the raw data from a given experimental setup can be stored in the star schema. If the results from different experimental setups are compatible and can be compared, then they will be stored in the same star schema or in scheme that share at least one dimension. If results from different setups are unrelated and cannot be compared, then they will be stored in separate scheme and analyzed independently

from each other. In other words, this approach also imposes that the comparison and cross-exploitation of results from different systems be meaningful, as if the results cannot be compared they will end up in different scheme and will have to be analyzed independently.

The OLAP tools (D area in fig. 2) – The OLAP tools are used to analyze the data and compute the measures. As most of the tools allow analysis through the web, this is the natural way to share the raw

data.

The steps needed to put our approach into practice are:

- a. Multidimensional analysis of the experimental setup (or setups) and definition of the adequate star schema. Once the schema is defined, all the tables are created in the data warehouse and the star schema is prepared to receive the raw data.
- b. Use the general-purpose loading application to define the loading plans for each table in the star schema. A loading plan is a kind of script that tells the loading application where to read the data to be loaded into each table of the star schema (it is necessary to define a loading plan per table in the star schema). If the data sources are too specific or have idiosyncrasies that cannot be handled by the general-purpose loading application, then it is necessary to write a specific program to get the data and load it in the star schema table. Once all the loading plans are prepared, the data warehouse can be loaded with raw data. Every time a new experiment is run in the experimental setup the corresponding loading plans are run again to add the new data to the data warehouse.
- c. Analyze the data with the OLAP tool. This includes not only the normal calculation of measures (e.g., error detection coverage, latency, recovery efficiency, etc) but also detailed analysis to find interesting things in the results. This is in fact the main advantage of using OLAP analysis, as finding unexpected results and analyzing data trends is the typical job done when OLAP is applied to its classical business field.

It is worth noting that data warehousing applications always require some administration, in addition to the three main steps mentioned above. Although this requires database administration skills, it is just normal routine to a database administrator [7, 9].

3. Example: a data warehouse to analyze results of benchmarking DBMS recovery mechanisms

The raw data analyzed in this data warehouse example has resulted from a set of dependability evaluation

experiments meant to benchmark different configurations of the recovery mechanisms of the Oracle 8i DBMS [10]. The data warehouse itself runs over Oracle 9i database management system (DBMS) and the OLAP tool is the Discoverer 4.0, also from Oracle.

3.1. Experimental setup

The basic platform used in the experiments consists of two Intel Pentium III machines with 256MB of memory, four 20GB hard disks, running the Windows 2000 operating system and connected through a dedicated fast-Ethernet network. Figure 3 shows the key components of the experimental setup.

The workload for this recovery benchmarking example is the standard TPC-C performance benchmark [11] running on top of Oracle 8i database server. This benchmark represents a typical database installation and simulates the activities found in many complex OLTP (On-Line Transaction Processing) application environments. The performance metric for this benchmark is expressed in transactions-per-minute-C (tpmC).

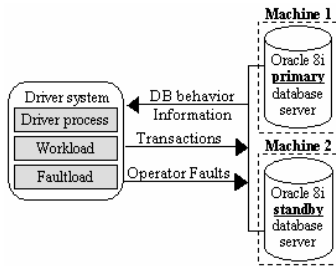


Figure 3. Experimental setup layout.

The TPC-C specification includes an external driver system that emulates all the client applications and respective users during the benchmark run. This driver system has been extended to handle the insertion of the faults. Additionally, the driver system also records the raw data needed to get the recovery and integrity measures, which consists mainly of recovery time, lost transactions, and detection of integrity violations.

The faultload consists of operator faults [12] injected through a set of scripts following the steps represented in Figure 4. In order to make it easy to reproduce the experiments we have decided to inject the faults at three specific moments, respectively after 150, 300, and 600 seconds from the TPC-C start. The duration of each individual experiment is 20 minutes and the execution of all the experiments is fully automatic.

The following main types of operator faults have been used: Shutdown abort, Delete a datafile, Delete a tablespace, Set a datafile offline, and Delete user's object. These faults have been chosen based on their ability to emulate the effects of other types of faults, diversity of impact in the system, and complexity of recovery. It is

worth noting that most of these types of fault actually include a large number of possible individual faults.

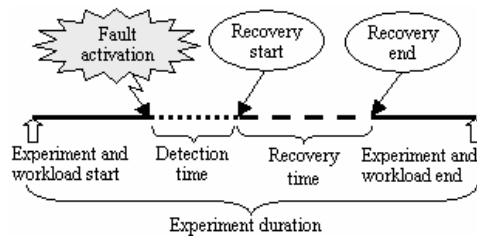


Figure 4. Steps in the injection of an operator fault.

The readouts collected by the driver system after the injection of each fault include the following data:

- TPC-C: includes information about the submitted transactions. Some examples of the readouts collected for each transaction submitted are: the identification of the client that submitted the transaction, the type of transaction, a timestamp of the executed transaction, the transaction sequence number, the server reply, and a timestamp of the server reply.
- Injection of faults: includes data on the fault type, fault activation time, and time needed to inject the fault.
- Impact of faults:
 - Data about the recovery process (e.g., type of recovery performed, the recovery time, etc).
 - Data integrity violations.
 - Failure modes (e.g., database crash, impact on the transaction execution, etc).

All these readouts are stored in several tables in the driver system. The total size of these tables at the end of the experiments was about 5 GB (about 95% of this space correspond to the TPC-C logs created by the driver systems to log the activity of the TPC-C clients), which gives an idea of the amount of data that is necessary to compute all the measures.

The measures consist of:

- Transactions-per-minute-C (tpmC). This is the main TPC-C measure (the other TPC-C measure is price-per-transaction but we have not considered this one).
- Recovery time.
- Number of lost transactions.
- Number of integrity violations.

It is worth noting that these measures are taken from the end-user point of view, which means that, for example, the recovery time includes all the time needed to recover the Oracle server plus the time needed to reestablish the transaction execution at the client application level (i.e., as it would be seen by the end-user).

3.2. The star schema

The definition of a data warehouse star schema is a three steps process (but the process is quite iterative and steps are revisited several times for refinement) [7]:

- a. Identification of the facts that characterize the problem under analysis.
- b. Identification of the dimensions that may influence the facts.
- c. Definition of the granularity of the data stored in the star schema (i.e., the level of detail).

Figure 5 shows the star schema used in the data warehouse built for the example of benchmarking DBMS recovery mechanisms. As usual in data warehousing, the star schema is quite intuitive. The fact table contains the basic numerical facts obtained from the readouts collected in the experiments and the dimensions include all the different perspectives that may be used to analyze those numerical facts. Some dimensions include many rows (e.g., the dimension “Faultload” stores all the individual faults injected) and some others have just few rows (e.g., the dimension “SUT” has only two rows, corresponding to the two systems tested). Each row of the fact table is linked to one row in each dimension, using the normal referential mechanisms of relational databases [9].

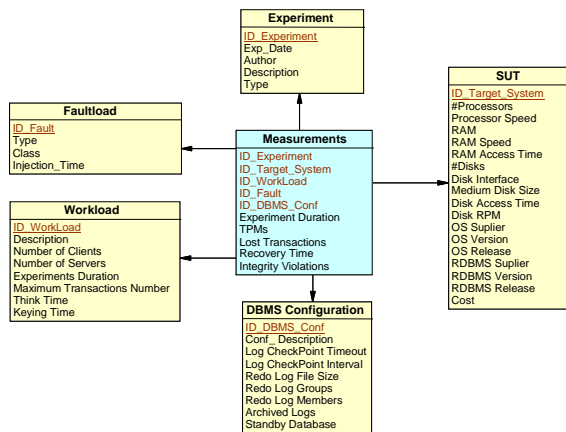


Figure 5. The star schema of the example.

One important aspect (not evident in figure 5) is that the data stored in the fact table has resulted from a preliminary processing of the readouts (pre-aggregation). This is quite common in data warehousing and means that the granularity chosen for the fact data is higher than the raw data. The pre-aggregation is done during the loading of the star schema.

Once the star schema is defined, the corresponding tables are created and the data warehouse is ready to be loaded with the data collected in the experiments.

3.3. Analyzing the data

This section gives an idea of the interface provided by the Discoverer 4.0 to analyze the data stored in the star schema. For space reasons only two screens can be shown.

Figure 6 shows the typical screen used for query construction. All that is needed is to drag the selected attributes from the frame on the left to the frame on the

right and click OK to process the query (the tab menu at the top of the screen allows the refinement of the query). For example, the query in figure 6 calculates the recovery time for each type of fault. Figure 7 shows the answer obtained with this query.

4. Cross-analyzing data from different types of experiments

The example of data warehouse shown in section 3 can be used in (or adapted to) many types of dependability evaluation experiments. Normally, when the experimental setup changes only the dimensions of the problem are affected. For example, a different target system can be accommodated in the model by inserting a new row in the dimension “SUT”; a new type of workload is a new register in the dimension “Workload”; a new faultload is a complete set of new rows (one row per fault) in the dimension “Faultload”, and so on and so forth.

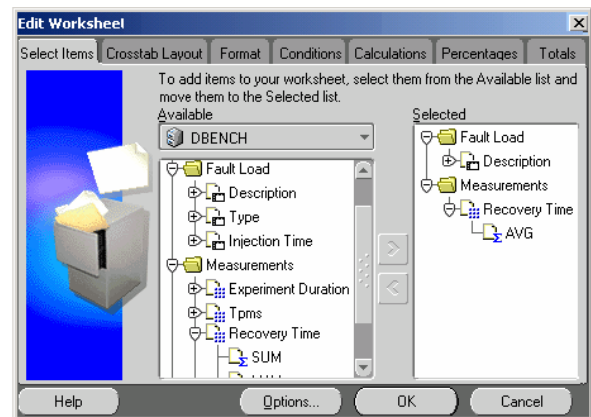


Figure 6. Query construction.

Whenever data from several experiments in different setups can be stored in the same star schema, it means that all the experiments are described by the same type of numerical facts and all of them share the same dimensions. In this case, it is very easy to compare results across different experimental setups. Furthermore, all the possible comparisons are meaningful, as all the experimental setups share a common data model.

When the only way to add data from a new setup is to define a new fact table (because the experiments in the new setup are characterized by a completely different set of measures), then the data warehouse will have a constellation of star scheme linked by the common dimensions. For example, if the same faults can be used in two different setups that are represented by two different star scheme, this means that the “Faultload” dimension is common to both star scheme.

It is easy to realize that the degree of freedom in cross-analyzing data from more than one star scheme is determined by the number of common elements

(dimensions) of the scheme. If the readouts from two different setups are stored in two completely independent star scheme it means that results cannot be compared (it would be an apples with oranges comparison). This is guaranteed by the intrinsic nature of the relational model [9], which does not allow joining data from tables that are not related somehow.

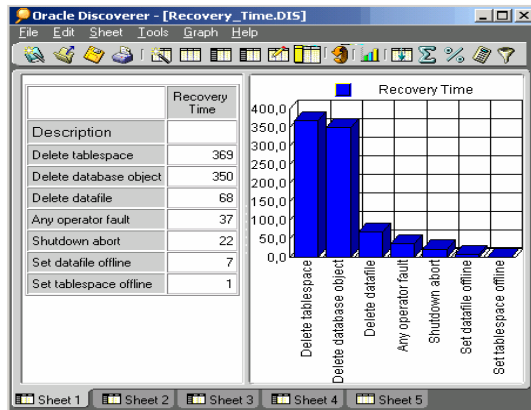


Figure 7. Example of query answer.

5. Possible exploitation scenarios

The data warehousing approach applied to the analysis of data obtained from dependability evaluation experiments can be used in different scenarios:

- At research team level, to perform the analysis of experimental data in a very efficient way. At the same time, both the data and the OLAP tool needed to analyze the data can be available at the web, allowing a very efficient dissemination of the research results.
- At project level (several research teams), to allow sharing and cross-exploitation of results.
- World wide in the form of common repositories to store and share experimental dependability evaluation results. In fact this is probably the only way to change the current situation, in which many teams are performing experimental dependability evaluation and there are no results currently available at the web. The data warehousing approach can change this situation drastically, as it proposes a common format to share the data (the star schema) and a standard type of tool to analyze the results (the OLAP tools) stored in the data warehouses available at the web.

6. Conclusion

This paper proposes the use of data warehousing and OLAP technologies to store raw results from different experiments/setup. The central idea is to collect the raw data produced in dependability evaluation experiments and store it in a multidimensional data structure (data warehouse). The data analysis is done through the use of commercially available OLAP tools such as the ones

traditionally used in business decision support analysis. Existing dependability evaluation tools and experimental setups are used as they are. The results obtained in the experiments are exported to a data warehouse, where all the analysis and cross-exploitation of the results can be done in an efficient and general way.

The proposed approach is particularly effective to cross-analyze data from different types of experiments, as the multidimensional schema guarantees that only meaningful comparisons are made.

References

- [1] M. Rodríguez, F. Salles, J.-C. Fabre, and J. Arlat, "MAFALDA: Microkernel Assessment by Fault Injection and Design Aid", Proc. 3rd Euro. Dependable Comp. Conf. Prague, Czech Republic, LNCS, 1667, pp.143-60, Springer, 1999.
- [2] J. Carreira, H. Madeira, and J. G. Silva, "Xception: Software Fault Injection and Monitoring in Processor Functional Units", IEEE Trans. on Software Engineering, vol. 24, no. 2, Feb. 1998.
- [3] P. J. Koopman, J. Sung, C. Dingman, D. P. Siewiorek and T. Marz, "Comparing Operating Systems using Robustness Benchmarks", in Proc. 16th Int. Symp. on Reliable Distributed Systems, SRDS-16, Durham, NC, USA, 1997.
- [4] D. T. Stott, B. Floering, Z. Kalbarczyk, R.K. Iyer, "Dependability Assessment in Distributed Systems with Lightweight Fault Injectors in NFTAPE," Proc. Int'l Comp. Performance and Dependability Symp., pp.91-100, March 2000.
- [5] J. Aidemark, J. Viter, P. Folkesson, and J. Karlsson, "GOOFI : Generic Object-Oriented Fault Injection Tool", Int. Conf. on Dependable Systems and Networks (DSN-2001), pp. B102-B103, Göteborg, Sweden, July 1-4, 2001.
- [6] K. Kanoun, J. Arlat, D. Costa, M. Dal Cin, P. Gil, J.-C. Laprie, H. Madeira, N. Suri, "DBench: Dependability Benchmarking", Supplement of DSN-2001, Chalmers Univ. of Technology, Göteborg, Sweden, 2001, pp. D12-D15.
- [7] R. Kimball et. al, "The Data Warehouse Lifecycle Toolkit", Ralph Kimball, Ed. J. Wiley & Sons, Inc, 1998.
- [8] S. Chauduri and U. Dayal. "An overview of data warehousing and OLAP technology". SIGMOD Record, 26(1):65-74, March 1997.
- [9] R. Ramakrishnan, "Database Management Systems" second edition, McGraw Hill, ISBN 0-07-232206-3.
- [10] M. Vieira and H. Madeira, "Recovery and Performance Balance of a COTS DBMS in the Presence of Operator Faults", Proc. of the Intl Performance and Depend. Symposium, IPDS2002, Bethesda, Maryland, USA, June, 2002.
- [11] TPC, "TPC Benchmark C, Standard Specification 5.0," 2001, <http://www.tpc.org/tpcc>.
- [12] M. Vieira and H. Madeira, "Definition of Faultloads Based on Operator Faults for DMBS Recovery Benchmarking", 2002 Pacific Rim International Symp. on Dependable Computing, PRDC2002, Tsukuba, Japan, December 16-18, 2002.

Acknowledgements

Funding for this paper was provided, in part, by Portuguese Government/European Union through R&D Unit 326/94 (CISUC) and by DBench project, IST 2000 - 25425 DBENCH, funded by the European Union.