DESIGN AND IMPLEMENTATION CONSIDERATIONS FOR MONITORING AND EVALUATING INFORMATION SYSTEMS

Joseph E. Urban

Wayne D. Dominick

Computer Science Department
University of Southwestern Louisiana
USL Box 4-4330
Lafayette, Louisiana 70504

Abstract

The effective administration of information systems requires a well-defined methodology for evaluating system performance and analyzing system usage. This paper describes some of the more important design and implementation considerations relevant to monitoring and evaluating information systems.

The general objectives of information system monitoring and evaluation are discussed, with a focus on system efficiency analysis and user interaction analysis.

The potential for utilizing software monitoring mechanisms within information systems is highlighted and the detailed structure of a monitor record, generated from a representative software monitor, is presented.

Key words and phrases: system performance evaluation, software monitors, user interaction analysis, information systems.

CR Categories: 3.72, 3.73, 3.74, 4.33, 4.34, 4.6

DESIGN AND IMPLEMENTATION CONSIDERATIONS FOR MONITORING AND EVALUATING INFORMATION SYSTEMS

I. Introduction

Motivated by the need to manage and access large masses of data generated by the constantly-expanding information explosion, business, government and academic communities have placed an increasing emphasis on the use of both general-purpose and special-purpose computer-based information systems. The effective administration of these systems requires a well-defined methodology for evaluating system performance and analyzing system usage.

The intent of this paper is to describe some of the design and implementation considerations associated with the development of a methodology for monitoring and evaluating information systems. The two primary generic types of information systems to be addressed are bibliographic information storage and retrieval systems, utilized for accessing reference material, and generalized data base management systems, utilized for defining, accessing, manipulating and analyzing data bases of varying content in any number of application areas.

This discussion addresses both system efficiency evaluation and user interaction analysis and overviews the required capabilities of a software monitoring facility to support such evaluations and analyses.

Most of the concepts described within this paper are intended to be generalizable across different information systems. However, in order to provide concrete examples of how they have been implemented into an existing information system, an appendix is provided which illustrates their use within MADAM (Multics Approach to Data Access and Management) [1], an information system developed at the University of Southwestern Louisiana. MADAM is also described in a companion paper [2].

II. Objectives of Information System Monitoring and Evaluation

For the purposes of this discussion, the overall objectives of information system monitoring and evaluation efforts will be grouped into two primary categories, namely, system efficiency analysis and user interaction analysis [3]. The former category addresses the functional performance of the system, while the latter category addresses the usage of the system by the members

of its user communities. Within system efficiency analysis, emphasis is placed on the "system", whereas, within user interaction analysis, emphasis is placed on the "user" and the user interactions with the system.

It is also necessary to distinguish between the concepts of "monitoring" and "evaluation." Monitoring is the process of collecting data associated with the functioning and/or usage of a system, while evaluation is the process of analyzing the functioning and/or usage of a system in order to make decisions concerning the effectiveness of the system in satisfying its design objectives. Thus, it is possible to monitor but not evaluate (which is a waste of time) as well as to evaluate without monitoring (for example, employing analytical techniques in contrast to analyzing empirical data). The orientation of this paper is to address the interrelated processes of monitoring and evaluation by viewing monitoring as the primary mechanism for collecting the data necessary to perform evaluations of both system execution efficiency and user interaction patterns.

For a comprehensive treatment of the concepts of monitoring and evaluation, the reader is referred to Lucas [4].

III. System Efficiency Analysis

The execution efficiency of an information system depends upon numerous factors, ranging from factors relating to the implementation environment (machine and storage device characteristics, sophistication of supportive operating system features, characteristics of the system's implementation language and the efficiency of generated object code) through factors characterizing individual data bases supported (data base size, content and structuring).

During execution, an information system must interface with many levels of supportive mechanisms which may include any number of hardware-implemented machines, firmware- or software-implemented virtual machines, operating systems, physical storage devices and data bases. Evaluation objectives and performance parameters for analyzing system execution efficiency can certainly be formulated for each of these levels. The intent of this section of this paper is not to examine each of these levels in detail, but rather to identify their existence as relevant considerations within a methodology for monitoring and evaluating information systems.

For example, if the primary objective of an information system evaluation was to evaluate the efficiency of the storage structures that were currently being employed to structure various data bases, the following general types of performance

variables and performance measures would be appropriate:

Performance Variables
- Data base structuring techniques employed
  - for data values
  - for indices
  - for directories
- Data base storage space requirements
  - for data values
  - for indices
  - for directories
- Type and complexity of operations performed
  - retrieval operations
  - insertion operations
  - deletion operations
  - update operations
- Operating system loading during operation execution

Performance Measures
- Response time (real time) for operation execution
- CPU time for operation execution
- I/O time for operation execution
- Paging activity (in a paged memory environment) for operation execution

For a detailed specification of the system execution efficiency parameters currently being monitored within the MADAM system, the reader is referred to the appendix.

IV. User Interaction Analysis

The two primary objectives of user interaction analysis are the identification of system usage profiles and the evaluation of user satisfaction with information system features and capabilities.

Traditionally, user interaction analyses of this nature have been performed using primarily manual data collection methods such as questionnaires, interviews and the like. The disadvantages of manual data collection methods are apparent: they are time consuming to administer; they usually collect data only after-the-fact, i.e., after the user has completed his or her sessions with the system, rather than during the sessions; they are prone to biases on the part of the administrator, for example, the interviewer; and they still require a data reduction effort to convert the results into machine readable form if any type of computerized analysis is to be performed on the data.

Realizing the deficiencies inherent in manual data collection methods, the authors contend that the process of collecting user interaction data should be viewed as an additional capability to be incorporated into the design of any automatic monitor for an information system.

Using an automatic monitor, the identification of system usage profiles can be accomplished in a straightforward manner via the definition of monitor parameters such as the following:

System Usage Profile Parameters
- Frequencies of data base usage
- Frequencies of command type usage
- Frequencies of user error occurrences
- Context of user error occurrences
- Duration (real time) distributions
  - per session with the system
  - per access of a particular data base
  - per command usage

System usage profile data similar to the above are currently being collected via automatic monitors incorporated into several information systems, including RIQS (Remote Information Query System) [5], BASIS (Battelle's Automated Search Information System) [6] and MADAM (see the appendix).

The evaluation of user satisfaction with information system features and capabilities is considerably less deterministic than is either the analysis of system execution efficiency or the identification of system usage profiles. Due to the inherently subjective nature of the concept of "user satisfaction", it is quite difficult to define quantitative parameters which accurately measure various levels of user satisfaction or dissatisfaction with information system processing.

Within this context, user comments and user ratings of system capabilities can certainly be employed, however, even these simple measures must be used with care. For example, system-prompted user ratings can become a self-defeating measure if the user becomes dissatisfied with the system merely because the system is asking the user to rate it too often. As our research in this area is in its infancy, we can provide no deterministic answers to these questions at this time. Via joint efforts with Northwestern University, we are in the process of identifying and subsequently validating various direct and surrogate measures which may be used to ascertain user satisfaction as a function of numerous parameters which can be collected by an automatic monitor incorporated into an information system.

V. Summary

Most of the system monitoring concepts described within this paper have been implemented into a bibliographic information storage and retrieval system at the University of Southwestern Louisiana and are in the process of being implemented into a generalized data base management system also at that University. To provide the framework for conducting a detailed examination of the design and implementation of the monitoring facility currently implemented in the bibliographic system (MADAM), the monitor record generated from each session with this system is illustrated in the Appendix to this paper.

Analyses and evaluations of the system using the monitored data has already resulted in several major enhancements to the system, including a redesign of the data base structures for minimizing storage requirements and a redesign of the inverted lists for optimized access time and paging. Similarly, analyses of user interaction data have highlighted the need for several query language enhancements which are currently in the design and implementation phase.

Acknowledgements

Portions of this work were supported by the National Science Foundation and Northwestern University, Evanston, Illinois under NSF Grant No. DSI76-19481.

The authors would also like to acknowledge Sharad K. Agarwal for his assistance in the implementation of the monitoring mechanisms and Howard G. Berg for his continuing assistance in the statistical analysis of the monitored data.

References

1. Dominick, Wayne D. and Sharad K. Agarwal, MADAM: Multics Approach to Data Access and Management Users Guide, Technical Report CMPS-77-6-1, Computer Science Department, University of Southwestern Louisiana, Lafayette, Louisiana, January, 1977, 90p.

2. Dominick, Wayne D. and Sharad K. Agarwal, "MADAM: Multics Approach to Data Access and Management," to be presented at the 15th Annual Southeastern Regional ACM Conference, to be held in Biloxi, Mississippi, April 18-20, 1977.

3. Mittman, Benjamin and Wayne D. Dominick, "Developing Monitoring Techniques for an On-Line Information Retrieval System," Information Storage and Retrieval: Theory and Practice, Vol. 9, No. 6, June, 1973, pp. 297-307.

4. Lucas, Henry C., Jr., "Performance Evaluation and Monitoring," Computing Surveys, Vol. 3, No. 3, September, 1971, pp. 79-91.

5. Dominick, Wayne D., "System Performance Evaluation of Interactive Retrieval," in Personalized Data Base Systems, B. Mittman, L. Borman (editors), John Wiley & Sons, Inc., 1975, Chapter 12, pp. 216-227.

6. Penniman, W. David and J. Chris Perry, "Tempo of On-Line User Interaction," Proc. 5th Mid-Year Meeting of the American Society for Information Science, May 20-22, 1976, pp. 34-49.

APPENDIX

The MADAM Monitor Record

This appendix defines the current MADAM monitor record. The first part of the appendix illustrates the equivalent of the declarations of the monitor record structures (in a PL/I-like notation) and the second part describes the function of each monitor record data item.

MONITOR RECORD DECLARATIONS

```
1 monitor_record_length unaligned controlled,
  2 dbn_len fixed bin(7),
  2 nam_len fixed bin(7),
  2 maf_len fixed bin(7),
  2 search_stat_len(no_of_searches),
    3 ser_len fixed bin(20),
  2 output_stat_len,
    3 display_stat_len(no_of_each_type_command(display_command_no)),
      4 dis_len fixed bin(20),
    3 print_stat_len(no_of_each_type_command(print_command_no)),
      4 prt_len fixed bin(20),
  2 multics_stat_len(no_of_each_type_command(multics_command_no)),
    3 mul_len fixed bin(20),
  2 ceo_len fixed bin(20),
  2 com_len fixed bin(15);


1 monitor_record unaligned controlled,
  2 login_stat,
    3 date_login fixed bin(20),
    3 real_time_on fixed bin(42),
    3 cpu_time_on fixed bin(30),
    3 amount_of_paging fixed bin(15),
    3 system_load fixed bin(7),
    3 name_of_database char(dbn_len) varying,
    3 size_of_database,
      4 data_file,
        5 no_of_records fixed bin(12),
        5 no_of_pages fixed bin(10),
      4 text_inverted_file,
        5 no_of_keys fixed bin(15),
        5 no_of_pages fixed bin(10),
      4 date_inverted_file,
```

```
      5 no_of_keys fixed bin(12),
      5 no_of_pages fixed bin(10),
    3 user_info,
      4 name char(nam_len) varying,
      4 affiliation char(maf_len) varying,
    3 system_info,
      4 system_version_number fixed bin(5),
      4 system_compilation_date fixed bin(20),
    3 os_system_version_number char(8),
  2 search_stat(no_of_searches),
    3 real_time_for_search_start fixed bin(42),
    3 real_time_for_search_finish fixed bin(42),
    3 cpu_time_for_search fixed bin(30),
    3 amount_of_paging fixed bin(15),
    3 system_load fixed bin(7),
    3 no_of_records_searched fixed bin(12),
    3 no_of_records_retrieved fixed bin(12),
    3 no_of_search_terms fixed bin(6),
    3 search_text char(ser_len) varying,
  2 output_stat,
    3 display_stat(no_of_each_type_command(display_command_no)),
      4 no_of_records  fixed bin(12),
      4 no_of_words    fixed bin(20),
      4 display_text char(dis_len) varying,
    3 print_stat(no_of_each_type_command(print_command_no)),
      4 no_of_records  fixed bin(12),
      4 no_of_words    fixed bin(20),
      4 print_text char(pri_len) varying,
  2 maintenance_stat,
    3 add_stat,
      4 no_of_records_added fixed bin(12),
      4 no_of_items_added fixed bin(15),
      4 real_time_for_add fixed bin(36),
      4 cpu_time_for_add  fixed bin(30),
      4 amount_of_paging   fixed bin(15),
    3 delete_stat,
      4 no_of_records_deleted fixed bin(12),
      4 no_of_items_deleted   fixed bin(15),
      4 real_time_for_delete  fixed bin(36),
      4 cpu_time_for_delete   fixed bin(30),
      4 amount_of_paging      fixed bin(15),
    3 update_stat,
      4 no_of_records_updated fixed bin(12),
      4 no_of_items_updated   fixed bin(15),
      4 real_time_for_update  fixed bin(36),
      4 cpu_time_for_update   fixed bin(30),
      4 amount_of_paging      fixed bin(15),
  2 multics_stat(no_of_each_type_command(multics_command_no)),
    3 real_time_for_multics fixed bin(36),
    3 cpu_time_for_multics fixed bin(30),
    3 amount_of_paging fixed bin(15),
    3 multics_text char(mul_len) varying,
  2 logout_stat,
```

```
    3 no_of_searches fixed bin(10),
    3 no_of_free_text_searches fixed bin(10),
    3 no_of_inverted_file_searches fixed bin(10),
    3 no_of_boolean_combination_searches fixed bin(10),
    3 total_no_of_errors fixed bin(10),
    3 no_of_each_type_errors(100) fixed bin(7),
    3 no_of_each_type_command(50) fixed bin(7),
    3 command_execution_order char(ceo_len) varying,
    3 real_time_off fixed bin(42),
    3 cpu_time_off fixed bin(30),
    3 amount_of_paging fixed bin(15),
    3 system_load fixed  bin(7),
    3 user_rating(3) fixed bin(3),
    3 user_comment char(com_len) varying,
    3 session_code  bit(10),
    3 session_cost fixed dec(7,2);
```

MONITOR RECORD DATA ITEM FUNCTIONS

login_stat.date_login
: Stores the date of the MADAM session.

login_stat.real_time_on
: Stores the real time at MADAM login time.

login_stat.cpu_time_on
: Stores the cpu time used prior to MADAM login time.

login_stat.amount_of_paging
: Stores the amount of paging used prior to MADAM login time.

login_stat.system_load
: Stores the operating system load at MADAM login time.

login_stat.name_of_database
: Stores the name of the data base being accessed.

login_stat.size_of_database.data_file.no_of_records
: Stores the number of logical records in the data file portion of the data base being accessed.

login_stat.size_of_database.data_file.no_of_pages
: Stores the number of pages in the data file portion of the data base being accessed.

login_stat.size_of_database.text_inverted_file.no_of_keys
: Stores the number of keywords defined within the text inverted file portion of the data base being accessed.

login_stat.size_of_database.text_inverted_file.no_of_pages
Stores the number of pages in the text inverted file portion of the data base being accessed.

login_stat.size_of_database.date_inverted_file.no_of_keys
Stores the number of keys defined within the date inverted file portion of the data base being accessed.

login_stat.size_of_database.date_inverted_file.no_of_pages
Stores the number of pages in the date inverted file portion of the data base being accessed.

login_stat.user_info.name
Stores the name of the user of MADAM.

login_stat.user_info.affiliation
Stores the affiliation of the user of MADAM.

login_stat.system_info.system_version_number
Stores the version number of the currently executing version of MADAM.

login_stat.system_info.system_compilation_date
Stores the date on which the currently executing version of MADAM was compiled.

login_stat.os_system_version_number
Stores the version number of the currently executing version of the Multics operating system.

search_stat.real_time_for_search_start
Stores the real time at which the execution of the search against the data base started.

search_stat.real_time_for_search_finish
Stores the real time at which the search finished, i.e., when the system displays the results of the search to the user.

search_stat.cpu_time_for_search
Stores the cpu time duration for the execution of a search.

search_stat.amount of paging
Stores the amount of paging used by the search.

search_stat.system_load
Stores the operating system load at the time of the execution of a search.

search_stat.no_of_records_searched
Stores the number of data base records searched in executing a search. A value of 0 is stored for the execution of an

inverted list search or a boolean combination select search, since such searches access no data records, merely indexes.

search_stat.no_of_records_retrieved
Stores the number of data base records retrieved by the search, i.e., the number of hits from the search.

search_stat.no_of_search_terms
Stores the number of search terms entered by the user as part of the search.

search_stat.search_text
Stores the full text of the users search.

output_stat.display_stat.no_of_records
Stores the number of logical records displayed by the current display command.

output_stat.display_stat.no_of_words
Stores the number of words (physical Multics 36-bit words) displayed by the current display command.

output_stat.display_stat.display_text
Stores the full text of the user specified display command options.

output_stat.print_stat.no_of_records
Stores the number of logical records printed by the current print command.

output_stat.print_stat.no_of_words
Stores the number of words (physical Multics 36-bit words) printed by the current print command.

output_stat.print_stat.print_text
Stores the full text of the user specified print command options.

maintenance_stat.add_stat.no_of_records_added
Stores the number of logical records added to the data base during the current session.

maintenance_stat.add_stat.no_of_items_added
Stores the number of logical items added to the data base during the current session.

maintenance_stat.add_stat.real_time_for_add
Stores the real time duration of all add processing during the current session.

maintenance_stat.add_stat.cpu_time_for_add
Stores the cpu time duration of all add processing during the current session.

maintenance_stat.add_stat.amount_of_paging
Stores the amount of paging of all add processing during the current session.

maintenance_stat.delete_stat.no_of_records_deleted
Stores the number of logical records deleted from the data base during the current session.

maintenance_stat.delete_stat.no_of_items_deleted
Stores the number of logical items deleted from the data base during the current session.

maintenance_stat.delete_stat.real_time_for_delete
Stores the real time duration of all delete processing during the current session.

maintenance_stat.delete_stat.cpu_time_for_delete
Stores the cpu time duration of all delete processing during the current session.

maintenance_stat.delete_stat.amount_of_paging
Stores the amount of paging of all delete processing during the current session.

maintenance_stat.update_stat.no_of_records_updated
Stores the number of logical records updated in the data base during the current session.

maintenance_stat.update_stat.no_of_items_updated
Stores the number of logical items updated in the data base during the current session.

maintenance_stat.update_stat.real_time_for_update
Stores the real time duration of all update processing during the current session.

maintenance_stat.update_stat.cpu_time_for_update
Stores the cpu time duration of all update processing during the current session.

maintenance_stat.update_stat.amount_of_paging
Stores the amount of paging of all update processing during the current session.

multics_stat.real_time_for_multics
Stores the real time duration of all MULTICS processing performed from each MULTICS mode usage from MADAM.

multics_stat.cpu_time_for_multics
Stores the cpu time duration of all MULTICS processing performed from each MULTICS mode usage from MADAM.

multics_stat.amount_of_paging
Stores the amount of paging of all MULTICS processing performed from each MULTICS mode usage from MADAM.

multics_stat.multics_text
Stores the full text of all MULTICS processing performed from each MULTICS mode usage from MADAM.

logout_stat.no_of_searches
Stores the total number of searches performed within the MADAM session.

logout_stat.no_of_free_text_searches
Stores the total number of free text searches performed within the MADAM session.

logout_stat.no_of_inverted_file_searches
Stores the total number of inverted file searches performed within the MADAM session.

logout_stat.no_of_boolean_combination_searches
Stores the total number of boolean combination searches performed within the MADAM session.

logout_stat.total_no_of_errors
Stores the total number of errors (both user errors and system special conditions) occurring within the MADAM session.

logout_stat.no_of_each_type_errors
Stores the number of errors of each type which occurred within the MADAM session. Each error type has a unique error number, which is used as the index into this error structure.

logout_stat.no_of_each_type_command
Stores the number of times each MADAM command was executed within the MADAM session. Each MADAM command has a unique command number which is used as the index into this command structure.

logout_stat.command_execution_order
Stores the order in which MADAM commands were executed within the MADAM session by storing the command name abbreviations in order of use.

logout_stat.real_time_off
Stores the real time at MADAM termination time.

logout_stat.cpu_time_off
Stores the cpu time used at MADAM termination time.

logout_stat.amount_of_paging