Evaluation Criteria for Self-Management Features in DBMSs Technical Report FIU-SCIS: 03-24-2007

Tariq M. King, Tuan L. Cameron, Suzette V. Diaz, Joseph Cilli, and Abdul Muqueet
School of Computing and Information Sciences
Florida International University
Miami, FL 33199, USA
email: {tking003, tcame002, sdiaz009, cillij, mmuqu001}@cis.fiu.edu

Abstract

The cost and difficulty of maintaining large scale heterogeneous systems has caused a paradigm shift towards self-managing systems. Large-scale systems typically require intensive data management services and hence many database management systems now incorporate features such as self-configuration, self-optimization, self-protection, and self-healing. This paper proposes criteria for evaluating self-management features in database management systems and uses those criteria to evaluate popular databases such as DB2, SQL Server 2005, Oracle 10g, Teradata and Sybase ASE.

Keywords: Self-Management and DBMS, Autonomic Computing, DB2, SQL Server, Oracle, Sybase ASE, and Teradata.

1 Introduction

Major industry players such as IBM and Microsoft have proposed their future direction in addressing the problems relating to the continual growth in size and complexity of computing systems. IBM's vision of dealing with the software complexity crisis is the Autonomic Computing (AC) paradigm, which is based on the idea of the human autonomic nervous system. The human autonomic nervous system performs basic functions such as breathing, homeostasis and regulating the heartbeat without the need for conscious thought. Extending this concept to the computing systems of the future has been IBM's primary focus. IBM has successfully attracted members from the IT industry, research community, and academia to the area of autonomic computing through various manifestos, research papers, and technical reports [3, 2, 7, 5].

The Dynamic Systems Initiative (DSI) from Microsoft [8] addresses the complexity problem of distributed systems by shifting the focus of software development from a single application view to one which encapsulates the entire system. One of the key target areas of the DSI is the Windows platform and making it more operationally aware in order to simplify and automate how large systems are configured, deployed, and operated. Furthermore, DSI extends to self-management of both hardware and software resources that will operate in this self-aware operating environment.

The underlying motivation for the two aforementioned initiatives is to alleviate the human administrator of many of the burdensome tasks associated with configuration and maintenance of highly complex systems. The cost and difficulty of managing large scale systems could be greatly reduced through increased automation of low-level tasks, coupled with the provision of mechanisms that allow administrators to specify desired system objectives in a high-level fashion.

For decades the role of the database administrator (DBA) has been an extremely challenging one. Databases typically have thousands of tunable parameters which must be considered together with the operating environment to maximize overall database performance. Many backup and recovery solutions are not scalable and the DBA is generally faced with a continual growth in the volume of data within the enterprise. According to [11], the new innovative self-management features of database management systems (DBMSs) will lead to tasks like performance tuning becoming less challenging, requiring less effort and involving fewer technical skills. However, studies have also shown that the incorporation of such features into software involves the risk of making management harder if these features do not provide truly self-managing capabilities [4].

The major contribution of this paper is the establishment of criteria for evaluating self-management features in DBMSs. Our work provides:

1. Well-defined criteria and associated weights for use in the evaluation of self-management features in DBMSs.
2. A survey of the self-management features in five popular databases – IBM DB2, SQL Server, Oracle 10g, Sybase ASE, and Teradata.
3. An evaluation of the self-management features of the databases in (2) using our proposed criteria.

This paper is organized as follows: the next section contains background information on database management systems, and outlines the main features of self managing software. Section 3 provides a survey of the self-management features of five popular databases. Section 4 presents our proposed evaluation criteria and provides the rationale for their definition. Section 5 provides the results and discussions of our evaluation study. Section 6 presents related work, and in Section 7 we give concluding remarks and discuss future work.

2 Background

In this section we provide background information on database management systems and the traditional role played by the database administrator. We also discuss the key characteristics of self-management with respect to software systems.

2.1 Database Management Systems

A database management system (DBMS) is a software package designed to organize, store, retrieve and manage data in a database [12]. Database administrators use the various tools and functionalities provided by the DBMS to manage the environmental aspects of the database. These environmental aspects include data backup and recovery; verification of data integrity; data access control; data availability; and database performance.

The effectiveness of a DBMS with respect to alleviating the burden of the DBA should be analyzed in the context of the activities for which the DBA is responsible. A survey of 52 enterprises conducted by [11] revealed that the database administration activities that are found to be most challenging (ranked in order from most challenging to least) are: performance and tuning; patch and upgrades; change management; planning; backup and recovery; replication and synching; security issues; and resource issues.

2.2 Self-Management in Software

For a feature in a software system to be considered as self-managing, it must implement a closed-loop of control in which changes to components are sensed so that corrective actions can be taken [9]. Even though corrective actions need not be fully automated, the system should provide the administrator with some level of automation to reduce the amount of manual intervention required to solve problems that occur. The ability to specify system policies is also central to idea of self-managing software. Policies provide the means by which administrators can express the desired system objectives [5], and are therefore used by the system to determine when corrective actions should be taken. For these reasons any software system that provides self-management facilities must be aware of its own state and behaviors, as well as its operating environment.

There are four characteristics that are widely-used and accepted when classifying the different types of self-management features in software systems [5, 9]. These include: (1) *self-configuration* – automatically configuring or re-configuring existing system components, and seamlessly integrating new components; (2) *self-optimization* – automatically tuning resources and balancing workloads to improve operational efficiency; (3) *self-healing* – proactively discovering, diagnosing and repairing problems resulting from failures in hardware or software; and (4) *self-protection* – proactively safeguarding the system against malicious attacks, and preventing damage from uncorrected cascading failures.

3 Self-Management DBMS Features

In this section we provide information gathered from a survey of the self-management features in five popular database management systems. The databases surveyed were IBM DB2, MS SQL Server 2005, Oracle 10g, Teradata and Sybase ASE 12.5.1. Each feature is presented along with a short description of its self-management functionality.

3.1 IBM DB2 Universal Database

Over the years IBM has been gradually incorporating self-management technology into its products, with its DB2 Universal Database being no exception. The main self-managing features in DB2 [7] include:

- *Query Optimizer* - determines the most efficient way to execute an SQL query by using powerful query rewrite rules and a detailed cost model.

- *Configuration Advisor* - automatically configures and allocates system memory for operations such as data caching, sorting and networking.
- *Index Reorganizer* - automatically reorganizes index structures of database tables and merges pages which may have become fragmented.
- *Design Advisor* - analyzes a workload of one or more queries supplied by the user and suggests candidates for optimal indexes.
- *Query Parallelism Selector* - dynamically determines the most effective degree of parallelism to use for queries, thereby allowing complex queries to benefit from parallel processing.
- *Load Utility Tuner* - balances the load process by automatically selecting the memory consumption, I/O parallelism, and SMP parallelism degree.
- *Query Patroller* - utilizes policies to automatically accept, analyze, prioritize, and schedule database requests; and may also send notifications to users when their requests have been processed.
- *Incremental Restorer* - uses information from the databases history to perform an automated search for the backup images required to complete a restore process specified by the user.
- *Support Serviceability Utility* - automatically collects system information such as machine specification, database product levels, and configuration.

3.2 MS SQL Server 2005

Microsoft's DSI has led to the development of a component called the Microsoft Operations Manager (MOM), which provides manageability as part of the design and implementation of the Windows environment. Using MOM in conjunction with the SQL Server 2005 Management Pack allows for quick discovery of various database-related problems. This is achieved through the provision of proactive and reactive monitoring techniques, thereby allowing system administrators to fix issues before users encounter a problem. The major features of MOM and other self-management features relating to SQL server [8] include:

- *Server Agent* - monitors the state of the included services and the state of the databases.
- *Connectivity* - ensures that databases are configured correctly and that clients can connect to the SQL server.
- *Server Service* - detects blocked processes and checks for failed agent jobs, or jobs taking an excessive time to execute.
- *Health* - monitors the health of replication, alerting on failures and observing the state of database mirroring.
- *Index Tuning* - automatically selects indexes in the query and storage engines through an Index Tuning Wizard [8]. The wizard then recommends a set of indexes and materialized views that are suitable given a workload of a set of SQL statements.
- *Storage* - monitors the remaining space in database and checks for auto-grow of files and file groups.
- *Query Optimizer* - automatically resolves which columns require histograms for tuning and determines the proper amount of optimization per query.

3.3 Oracle 10g

Oracle 10g introduces various self-management capabilities to simplify administration, increase efficiency and lower the total cost associated with systems management. These new capabilities include [6]:

- *Server-based Advisors(SBA)* - responsible for analyzing information related to self-management functionality and making recommendations to resolve issues.
- *Automatic SQL Tuner(AST)* - performs statistical analysis related to SQL query optimization, SQL profiling, access path analysis and SQL structure analysis.
- *Automatic Storage Manager(ASM)* - simplifies how datafiles, controlfiles and logfiles are stored.
- *Automatic Workload Repository (AWR)* - stores and manages information for all self-tuning functionality.
- *Automatic Database Diagnostic Monitor(ADDM)* - analyzes data in the AWR, identifies possible performance bottlenecks and provides recommendations for solving discovered problems.
- *Optimizer Statistics Collector(OSC)* - gathers statistics on optimizations.
- *Active Sessions History(ASH)* - maintains a representation of active sessions which includes wait events and SQL information.
- *Manageability Monitor (MMON)* - coordinates all the autonomic management within the server by taking snapshots of the database and storing the information in the AWR.
- *Server-Generated Alerts(SGA)* - configures the system to automatically generate an alert whenever an event is triggered.

- *Oracle Universal Installer (OUI)* - automates all pre and post installation tasks. OUI automatically checks the system prior to guarantee the success of the installation process and recommends changes.
- *Automatic Shared Memory Manager (ASMM)* - automates the management of shared memory used by an Oracle Database instance and liberates administrators from having to configure the shared memory components manually.

3.4 Sybase ASE

Adaptive Server Enterprise (ASE) is relational database management software that is manufactured and sold by Sybase Inc. It is used in several industries including the financial world and e-commerce, and runs on a variety of platforms, including UNIX, Linux, Windows, and MacOS. The main self-managing features in the Sybase ASE include [10]:

- *Job Scheduler (JS)* - automatically schedules routine jobs such as scheduled backups, index management, and reports, and can be extended to automatically do repetitive tasks.
- *Database Space Management(SM)* - helps run ASE for extended periods without human intervention by pre-setting growth or threshold requirements, therefore allowing the database, as well as the logs, to grow automatically.
- *Resource Management Module(RMM)* - responds to changing conditions in a mixed-workload environment with no human intervention by monitoring and dynamically adjusting database resources such as connections, locks, and the like in order to avoid system resource bottlenecks that can occur under increased loads.
- *ASE Replicator (REP)* - implements basic replication from a primary Adaptive Server to one or more remote Adaptive Servers.

3.5 Teradata

Teradata provides several self-management features which reduce both the complexity involved in maintenance of the data warehouse and the cost of ownership by simplifying the administration required. These self-management features [1] include:

- *Optimizer* - takes SQL from tool and/or user and runs without hints or tuning. Removes DBA time required to capture and analyze tool-constructed SQL and optimize physical model for the queries. Reduces DBA involvement in collection, analysis, and resolution of performance problems.
- *Data Management* - Reduces analysis time for data demographics, query demographics, and distribution options. Removes planning and execution of index data management, makes indexed data available in real-time with base table.
- *Space Management* - Removes time necessary to evaluate, allocate, and monitor table and work spaces.
- *Fail-over* - Removes set up effort for MPP systems to handle outages or lost nodes. Reduces time for planning and execution, shortens downtime caused by growth, eliminates DBA recalculation after growth process.
- *Workload Management* - Requires less DBA time for scheduling and monitoring of conflicting tasks.

4 Evaluation Criteria

This section presents the proposed evaluation criteria for self-management features in database management systems, and provides the rationale for the relative importance of each criterion. We also present the weights associated with each criterion for use in the evaluation.

4.1 Closed Loop Controller

As mentioned in Subsection 2.2, closed-loop controllers are central to the notion of self-management. We consider two aspects of this criterion – the sensory behavior that monitors the managed resource, and the effectory behavior that applies the change to the managed resource. Although a feature is not truly self-managing unless it implements both sensor and effector behavior, we make partial provisions for features that may only implement one of these aspects on the grounds that they are extensible to truly self-managing features. The weights defined for the closed loop controller criterion are shown in Table 1. Equal weights of value 1 were assigned to features that implement soley the sensor or effector behavior. However, if both behaviors were implemented a combined weight of 2 was assigned to that feature.

Table 1. Weights for Closed-Loop Control

Characteristic	*Weights*
Sensor only	+1
Effector only	+1
Sensor & Effector	+2

4.2 User-Defined Policies

It is essential for administrators to be able to specify the desired system behavior of self-managing systems as high-level objectives. We use the term "'policy"' in a broad sense to refer to any mechanism that allows user-defined values to be set for the bounds of desirable system behavior. Furthermore, we give more weight to the policies that use a standardized format for their specification and representation (e.g. XML), noting that such policies facilitate and promote automation. The weights defined for the user-defined policies criterion are shown in Table 2. Non-standard policies are given a weight of 1, while standardized policies are assigned a weight of 2.

Table 2. Weights for User-Defined Policies

Characteristic	*Weights*
Non-Standard Policy	+1
Standardized Policy	+2

4.3 Feature Diversity

This criterion relates to both the number of features provided by the DBMS and whether or not those features cover a cross-section of the categories of self-management characteristics (i.e. self-healing, self-configuration, etc.) discussed in Subsection 2.2. The rationale for the latter is that we allocate extra weighting to DBMSs that offer well-rounded self-management facilities rather than focusing on one category of self-management. The weights defined for the user-defined policies criterion are shown in Table 3. Weights were defined based on the number of features as well as the variety of the categories of features implemented.

The weights defined for the features diverseness criterion are shown in Table 3. A value of 1 was given for each self-management feature implemented in the database. In addition, a value of 1 was assigned for each category covered from the four self-management characteristics outlined in Subsection 2.2.

Table 3. Weights for Feature Diversness

Characteristic	*Weights*
Number of features	(per feature) +1
Variety of features	(per category) +1

4.4 Automation Granularity

The usefulness of self-management depends on the level of granularity of the automation and the flexibility of the facilities available to the administrator with respect to controlling that automation. This criterion assigns weights based on the level of automation from simple error reporting to fully automated sensor/effector behavior. Logging of self-management behavior is also considered as it may also be extensible to more comprehensive self-management behaviors.

The weights defined for the automation granularity criterion are shown in Table 4. A weight of 1 was assigned to features that implemented only a general report or alert feature. A value of 2 for features that implemented alerts but also made suggestions to the administrator for diagnosing the problem found. Features that implemented full automation, with respect to monitoring problems and automatically adjusting the system to maintain the specified behavior, but offered no logging facilities were given a weight of 3. If logging facilities were also available a weight of 4 was used, and a value of 5 was assigned if the adminstrator was provided with options that allowed switching between the level of automation.

Table 4. Weights for Automation Granularity

Characteristic	*Weights*
General Error Reporting	+1
Reporting w/ Recommendations	+2
Automation Control, no Logging	+3
Automation Control w/ Logging	+4
Automation Control, Logging, Options	+5

5 Evaluation Study

Our study involved using the proposed criteria to evaluate the self-managment features in IBM DB2 Universal Database, MS SQL Server 2005, Oracle 10g, Sybase ASE 12.5.1, and Teradata. The primary objective was to determine how well these popular DBMS solutions compare against each other in providing a well-rounded self-management solution. In this seciton we present the results of our evaluation study, and provide discussions of those results.

5.1 Results

The results of our study were first collected in a tabular spreadsheet format and then various graphical

	JS	SM	RMM	REP	Totals
Closed Loop controller					
Monitor	1	1	1	1	1
Effector	1	0	1	1	3/4
Total CLC	**2**	**1**	**2**	**2**	**88%**
User Defined Policies					
Standard Format	2	2	2	2	**100%**
Feature Diversity					
Covered	1	1	1	1	**100%**
Class of characteristics:					
--Self-Healing	0	1	0	0	
--Self-Optimizing	1	0	1	0	
--Self-Protecting	0	1	0	1	
--Self-Configuring	0	0	0	0	
--Total Classes	**1**	**2**	**1**	**1**	**31%**
Automation Granularity					
General Reporting/Alerts	1	1	0	1	
Recommendations	0	0	0	0	
Full automation	0	0	0	0	
Full automation with Logs	0	**0**	4	0	
Full automation with Logs and capable of changing Feedback Levels	0	0	0	0	
Total AG	1/5	1/5	4/5	1/5	**35%**
Grand Total					**63%**

Figure 1. Spreadsheet results for Sybase.

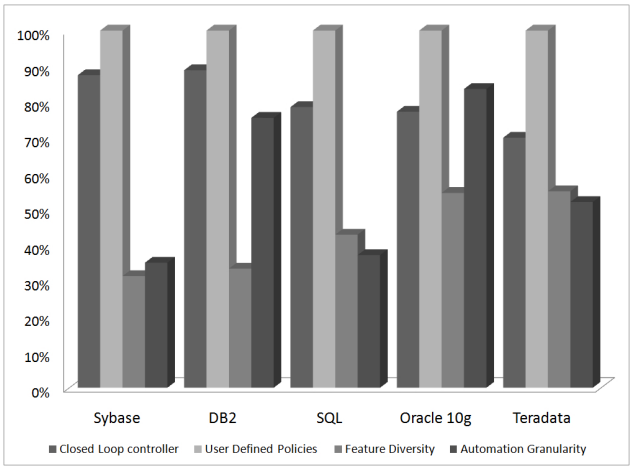

Figure 2. Bar chart self-* feature comparison.

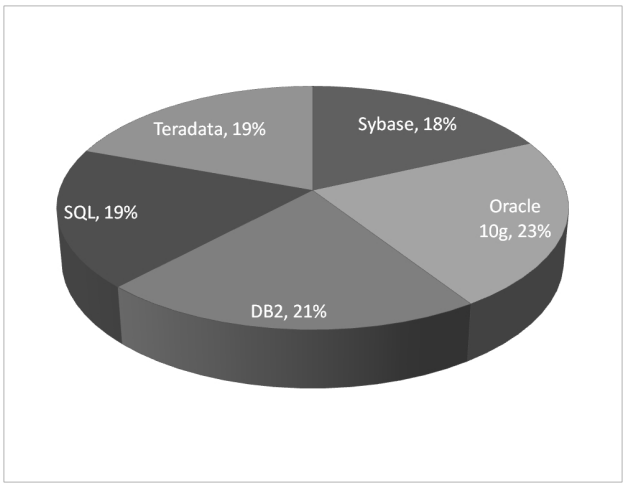

Figure 3. Overall self-* pie chart comparision.

representations were automatically generated. Most of the tabular results have been omitted (with the exception of a single sample) because of space limitations but are available from the authors on request.

Figure 1 shows a sample spreadsheet used to collect the data resulting from the evaluation of a single database. In the sample, Sybase ASE's features were ranked and the scores were charted against the other databases. It shows an important characteristic of our weighted approach from a cost of ownership perspective. Higher value is placed on automation as opposed to feature diversity, so that self-management features are deemed less valuable if they are not automated. Figure 2 shows a bar chart comparison of the individual self-management features of all five evaluated databases based on the four proposed criteria. Figure 3 shows a pie chart depicting an overall comparison of the five DBMSs. The percentages were calculated based on the average of the detailed comparison categories. The total percentage was then divided by the sum of all five DBMSs.

5.2 Discussion

Although different features of various DBMSs were evaluated, the functionality of the feature had no bearing on the results. Our goal was to measure the effectiveness of the stated features with respect to their self-management capabilities. Furthermore, no credence was given to the underlying programming of the stated features or the efficiency of differing algorithms.

The results of IBM's DB2 Universal Database indicate that the product provides a plethora of feature-rich self-management characteristics. The high score of DB2 was attributed to both the abundance of self-management features, and the high level of automation of those features. It is evident that IBM is using its DB2 product as an integral part of their vision of autonomic computing. The product's flexibility allows the DBA to choose which autonomic attributes will be used. However, although DB2 contains many automated self-management facilities, the relatively low score with respect to self-protection indicate that this is an area that should be improved. Oracle 10g provides an impressive suite of self-management features and has reasonable coverage of the various self-management characteristics. However, like DB2, there is room for improvement in the self-protection category.

Microsoft's use of MOM in conjunction with SQL Server 2005 Management Pack has shown to have strong monitoring characteristics. Although the re-

porting of the monitors is also strong, the product falls short with regard to automation. Necessary improvements to the self-management components of SQL Server would include alerts with recommendations, and full or partial automation of the stated self-management features.

The evaluation of Sybase's ASE has shown to have a good overall result. We evaluated four different automation features and found them to implement all but self-configuration characteristics. Our initial evaluation shows it's strongest feature to be space management. Teradata has implemented many features of self-optimization and self-healing which may be due to its parallel data processing system. However, in terms of automation it can improve in areas of data management and workload management.

A key observation of the DBMS products evaluated in our study was the lack of self-protection features. This was not surprising as a 2005 survey estimated that 80% of enterprises did not even have a security plan [11]. However, this has changed within the past two years and security has become a primary focus of many enterprises. Therefore, DBMS vendors should be careful not to neglect self-protection features as they can greatly assist administrators with the challenge of maintaining data privacy and data integrity.

6 Conclusions and Future Work

In this paper we proposed evaluation criteria for self-management characteristics in DBMSs. The criteria were defined with associated weights that can be used in the assessment of any self-management characteristic. Furthermore, we provided a survey of the self-management features currently available in IBM DB2, MS SQL Server, Oracle 10g, Sybase ASE, and Teradata. The features of the aforementioned databases were evaluated using the proposed criteria in the context of four aspects of self-management; namely self-configuration, self-optimization, self-healing, and self-protection.

Future work calls for the establishment of additional criteria and associated weights, as well as further refinement of the proposed criteria. In addition, we plan incorporate additional weights related to the ranked difficulty of DBA administrator challenges into our approach. Such a strategy would allow for a better estimation of the degree to which the self-management features fulfill their goal of alleviating the burden of management from the DBA.

7. Acknowledgements

The authors would like to thank Dr. Peter Clarke for his contributions to this work.

References

[1] R. Armstrong and C. Ballinger. Teradata database ease of management a total cost of ownership advantage. Technical report, Teradata, December 2003.

[2] P. Horn. Autonomic computing: IBM's perspective on the state of information technology. Technical report, IBM Corporation, Armonk, NY, USA, 2001.

[3] IBM. An architectural blueprint for autonomic computing. Technical report, IBM Corporation, June 2005.

[4] E. Kandogan and J. Bailey. Usable autonomic computing systems: The administrator's perspective. In *ICAC '04: Proceedings of the First International Conference on Autonomic Computing (ICAC'04)*, pages 18–26, Washington, DC, USA, 2004. IEEE Computer Society.

[5] J. Kephart and D. Chess. The vision of autonomic computing. *Computer*, 36(1):41–52, January 2003.

[6] S. Kumar. Oracle database 10g: The self-managing database. November 2003.

[7] S. S. Lightstone, G. Lohman, and D. Zilio. Toward autonomic computing with db2 universal database. *SIGMOD Rec.*, 31(3):55–61, 2002.

[8] Microsoft. Dynamic systems initiative overview. Technical report, Microsoft Corporation, March 2004.

[9] H. A. Muller, L. O'Brien, M. Klein, and B. Wood. Autonomic computing. Technical report, Carnegie Mellon Univeristy and Software Engineering Institute, April 2006.

[10] Sybase. Internet: http://www.sybase.com.

[11] N. Yuhanna. Database administration challenges are shifting. Technical report, Forrester Research Inc, Cambridge, MA, USA, 2005.

[12] O. R. Zaane. Database systems and structures, chapter1, 1995.