Implementation of Agent Oriented Database Model for Accelerating DSS Query

Shivani A Trivedi¹ Dr. N. N. Jani² Dr. Jagdish Pandya³
¹ Assistant Professor ² Director ³ Executive Manager
1,2,3 Department of Computer Engineering

Abstract—The cited research summarized in this paper is targeted to implement the proposed agent oriented database model to enhance DSS query-response performance. The implementation of data has taken from student data repository of a university system. Three data sets are taken with the volume of hundreds, thousands and lacks records. Data sets of 400 records have been implemented in proposed AODB model to compare query-response performance with the previous experimental review data based on RDBs and MDBs. Query response in terms of execution time is considered as comparison parameter. The results of this experiment have been found which indicates a shift of paradigm from RDBs to MDBs to AODBs data model paradigm justifying improvement in query performance in DSS.

Key words: Connectivity, Resiliency, WSN, constraints

I. INTRODUCTION

Application of DSSs is in the areas like health, finance, sales, education, bank etc. And database management system plays vital role in development framework of DSS. This research paper is to implement AODB model in the database and to observe the DSS query response. And further to compare it with previous review of comparative study of DSS query-response performance in two data model RDBS and MDDBs. This work is part of the undertaken research intended to develop intelligent agent oriented data model to accelerate performance of DSS. To examine and analyse the DSS query execution time using proposed AODB database model. In the experimentation, data sets of 400 records of the student data from the university system. This model with specified data has been implemented using Oracle 11g database system.

The Multi-Agent DSS systems so far developed on its established architecture have focused more on the design aspects at front-end level. That can be identified on bases of study findings. Database Management System is primary component to carry up to date and comprehensive data for intelligent decision support system said [1]. Using JADE platform, resource allocation problem in multi-agent environment can be solved using integrated system to present role of database. [2, 3 4]. The multi-agent oriented paradigm used in dynamic decision making in the area like stock market to manage portfolio, risk management based on the preferences in the opinion of group decision making [2,4,5,6].

An agent assisted DSS is suggested to be useful for high degree of cooperative problem solving capability [7]. Multi-Agent paradigm enhance the two characteristics in DSS like interoperability and reuse of underlying data files in heterogeneous database system using Object Modelling Technique (OMT) and Multi-Agent Software Engineering (MaSE) [8].

II. AGENT ORIENTED DSS-DATABASE MODEL DEVELOPMENT

In this research, AOR modelling techniques are used to design proposed agent oriented database model (AODB model) which on implementation is anticipated to accelerate the performance level of data access. This model is based on two sub-models an external AOR model corresponding to environment analysis to communicate among agents (as database agent object interface), and an Internal AOR model, corresponding to a base object and resource object data design model figure (1).

Fig. 1: AORDB Model

Three phases are suggested in this research to develop AODB model represented in figure (2). In the phase I. Decision support database environment analysis is done to identify the decision maker and their information need based on the data. The second phase AODB design is prepared using AOR modelling technique. And last in third phase AODB model is implemented using object oriented database and object relational database support.

Fig. 2: AODB development Phases

Fig. 3: represent external model of agent which consisting of base object and resource object and communication of an agent with its environment. Figure (4)
represents internal communication between base object and resource object. Table (1) represents PEAS description of data as agent for in-data intelligence architecture for generating and automating decision making data in the database.

![Diagram](image)

**Fig. 3: External AODB Model for Agent [STUDENT]**

**Fig. 4: Internal communication model for base data and resource object**

<table>
<thead>
<tr>
<th><strong>Architectural Components Of Agent</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Student</td>
<td>Rollno, Name, address, city, contact no</td>
</tr>
<tr>
<td>Performance Measure</td>
<td>Attendance, Promoted, Grade, Placed, Library due, Accounts due, Unclear papers</td>
</tr>
<tr>
<td>Environment</td>
<td>Student Information System (database)</td>
</tr>
<tr>
<td>Actuators</td>
<td>Faculty, Student, Librarian, accountant, Director</td>
</tr>
<tr>
<td>Percepts</td>
<td>Attendance Rule, Promotion Rule, Library Rule</td>
</tr>
</tbody>
</table>

**Table 1: PEAS description of the task environment for Student agent**

**III. IMPLEMENTATION OF DATA MODELS AND TEST DATA GENERATION**

We implemented the agent oriented data models for student (schemas) using the ORACLE 11g R2. We created trigger on the existing relational schema to automate the base data value generation. The we have created student type object and student resource type object. After that we have implemented student_agent as relational embedded. Which is based on base object and resource object as mentioned above. Then we created SQL and PLSQL scripts for identified decision requirement from the power users. Then we executed the end user required queries to take the result data. Figure (5) represents some sample script code used in this experiment.

**A. SQL-ODL Script 1:**

**Public interface:** Specification Type student_type as object:

```
student_type enrollmentno number,
first_name varchar2(20)
middle_name varchar2(20)
last_name varchar2(20)
mother_name varchar2(20)
student_category varchar2(4)
birth_date date
academic_year varchar2(7)
course varchar2(5)
brach varchar2(20)
enrollmentdate date
```

**Method Specification:** Map member function get_enrollmentno return number, member procedure display_details(self in out nocopy student_type)

**B. SQL-ODL Script 2:**

**Private Implementation:** Body create or replace type body student_type is map member function get_enrollmentno return number is begin return enrollmentno; end get_enrollmentno;

**Method Specification:** Map member function get_enrollmentno return number, member procedure display_details(self in out nocopy student_type) is begin

```sql
dbms_output.put_line(self.enrollmentno||' '||self.first_name||' '||self.middle_name||' '||self.last_name||' '||self.birth_date);
```

end;
C. SQL-ODL Script 3:
Public interface Specification: Type

student_resource_type as object
attendance_per number,
promotedsem varchar2(20),
cpi number(4,2),
library_due number,
fees_due number, placed yn char(1), placed_company varchar2(256),
placed_salary number,
meritscholarship number
Method Specification:
map member function get_academic_fact return number,
member function get_account_fact return number,
member function get_library_fact return number,
member function get_placement_fact return number,
member function get_scholarship_fact return number

D. SQL-ODL Script 4:
Create Type Student_Resource_Tab_Type as Table Of Student_Resource_Type:
create table student_agent
(student_info student_type,
student_resource student_resource_tab_type)
nested table student_resource store as srtable;

E. SQL- Script 5:
select p.student_info.brach,
p.student_info.course,
count(p.student_info.enrollmentno)
from student_agent p
where p.student_info.enrollmentdate='10-jan-2014'
group by p.StUDENT_INFO.brach, p.student_info.course;

IV. RESULTS AND ANALYSIS
The statics and its graphical results to identification of suitability of existing database model to accelerate DSS query performance is out lined from execution time, memory blocks used, logical complexity and volume of data. These observations are shown in figure (5). Figure (6) depicts the execution time of DSS query on the student_agent embedded object table.

Time also require 10%less than the RDBs. Data shows that when the volume of records increases that is in thousands and lacks there is very minor reduction in execution of query in both RDBs and MDDBs.

Table: 2 Summary of review research experiment and proposed AODB model experiment

<table>
<thead>
<tr>
<th>Schema</th>
<th>Execution Time in seconds (in %)</th>
<th>Memory Blocks (KB)</th>
<th>Data Volume In Number of records</th>
<th>Aggregat Data Volume In Number of records</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relational [9]</td>
<td>0.0334</td>
<td>5</td>
<td>400</td>
<td>0</td>
</tr>
<tr>
<td>Star schema[8]</td>
<td>0.0301</td>
<td>5</td>
<td>400</td>
<td>16</td>
</tr>
<tr>
<td>Proposed Agent Oriented Schema</td>
<td>0.016</td>
<td>5</td>
<td>400</td>
<td>0</td>
</tr>
</tbody>
</table>

V. CONCLUSION AND FUTURE EXTENTION
The analysis of the results obtained on implementation of proposed AODB model has led to conclude that the query execution response time is comparatively reduced indicating appreciable reduction in query execution response time from RDBs to MDDBs and future to AODBs. This impacts the enhancement of performance level of DSS. Quantitatively the execution time has found to be decreased by 9% in case of MDDBs compared to RDBs and overall decrease of 51% in case of RDBs to AODBs, as interpreted from graph. These
results justify the research targets of enhancement of performance levels of DSS query. The undertaken work can be extended for further validation with use of more diversified and complex data sets.

REFERENCES


