Technium Vol. 9, pp.26-40 (2023)
ISSN: 2668-778X
www.techniumscience.com

UACA: Unified Access Control Approach For Heterogeneous Database Based-on Service Data Object

Sadoon Hussein Abdullah[1], Nadia Maan [2]

[1] College of Science, Mosul University, Iraq.
[2] College of Computer Science and Mathematics, Mosul University, Iraq.

Abstract: The requirement for data interchange, access, and retrieval from diverse databases. This requirement is getting more and more crucial. However, retrieving this data from a heterogeneous database necessitates juggling many kinds of query languages, data models, and structure. The goal of this project is to create a unified access control approach (UACA), which will be managed by service data object (SDO) technologies and will be based on an agent mechanism for application access to business databases. To create a unified data access control, applications must be submitted through a UACA. To prevent a direct link to the database and application security threats, unified access control isolates applications from databases. Application systems submit a request for identity authentication to the database access control system using the data access protocol of the access control system. Data are finally transmitted to the application system after identity authentication, data gathering, and data transmission. This investigation led to the creation of a client-middle layer, which is also in charge of transformation and optimization, to ensure uniform data access control and access to the heterogeneous database. It also manages system communication using the TCP/IP protocol to guarantee the accuracy and reliability of the connection. SDO data retrieval offers strong adaptability and system compatibility.

***Keywords*:** Service data object; SOA; TCP/IP; Heterogeneous.

1. Introduction

In the current world of technology, the conditions in the market are changing hence requiring flexible enterprise information management basing them on mission-critical software integration. During in the recent decades, the data models (DM) and their architecture have been significantly altered to support object methods and interoperability.

Access of data from different storage is the main activity of any application service. Therefore deferent storage ways have different ways of accessing these data. The platform of the software and the programming language used in creation of this software application will give the right data access method to be used. Many data access techniques fall short of the demand for reusability in the context of service-oriented architecture, leading to the development of the service data object mechanism [1].

Technium Vol. 9, pp.26-40 (2023)
ISSN: 2668-778X
www.techniumscience.com

There are several data access methods; Microsoft has access method such as ado.net, ActiveX and ODBC, java has JDO, JDBC, and JAXB [2]. These access methods are designed for specific use only, as shown in Fig.1.

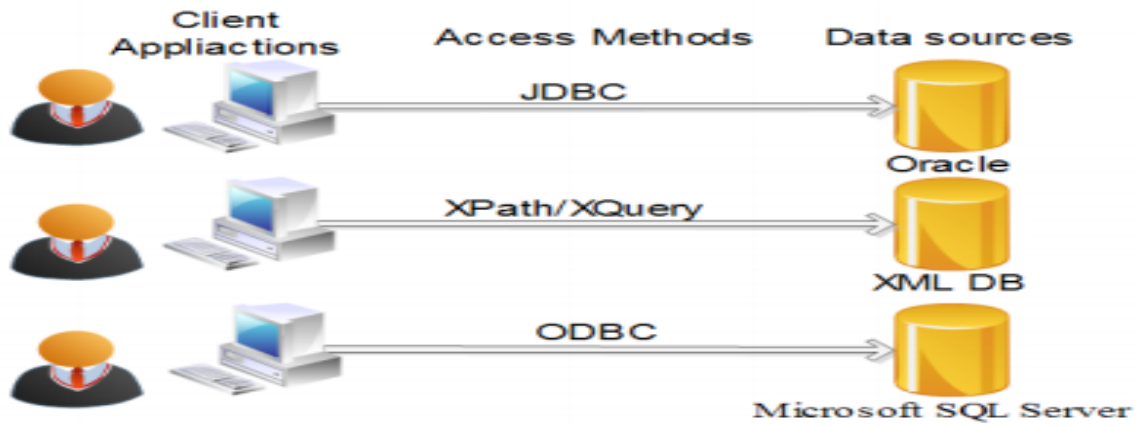

Figure 1. Data access strategies

However, there are issues and risks associated with using these data access methods when users and data are in direct contact. This means that a unified access control method database will need to be developed in order to control access to data. [3]. As shown in Fig.2..

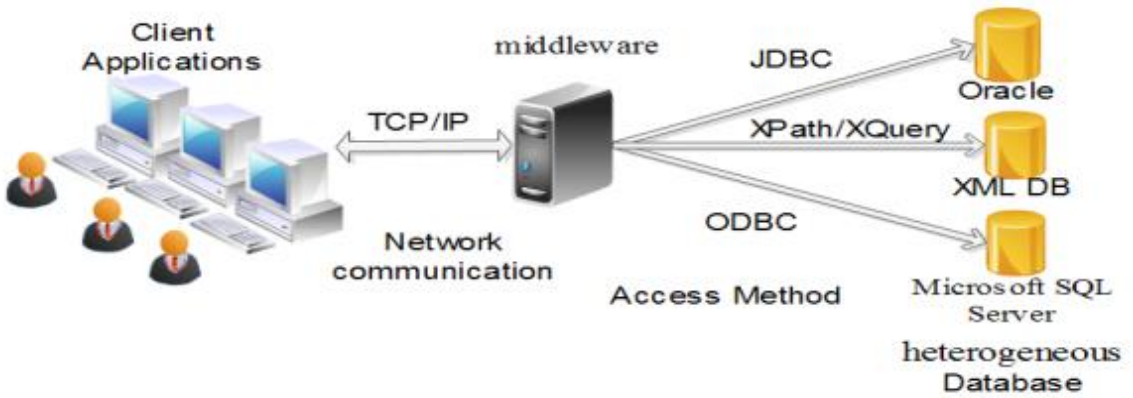

Fig.2. Three proposed construction troubleshooting and performance testing steps

In the above figure we can say that the access to heterogeneous database between the data sources and client applications is achieved. There is data integration between them.

2. Related work and Main Problems Definition

Below, we will outline the work associated with performance testing for identifying and eliminating fault bottlenecks; subsequently, we will discuss the manner in which CloudPT merges with a cloud-ecosystem service.

2.1. Service Data Object Technologies

Service Data Object (SDO) is designed to unify and simply how applications do handle data [12]. By using service data object, programmers can uniformly access and manipulate information from the

heterogeneous data sources such as, relational database, Web services, and XML data sources, and also information system from enterprise [13]. The basis of service data object is on the concept of the disconnected data graphs, which are graph structured data objects.

Service data object also provides a metadata API, which allows applications, tools, and frameworks to introspect the data model for a data graph. The service data object metadata API simplify and unify data source specific metadata APIs to enable applications to handle data from heterogeneous data sources in a uniform way. Aims of SDO are to support disconnected programming models, and unify access to data from heterogeneous data sources [13].

2.1.1. Service Data Object Architecture

Service data object is made up of composable architecture (as opposed to monolithic) [13]. As shown in Fig.3.

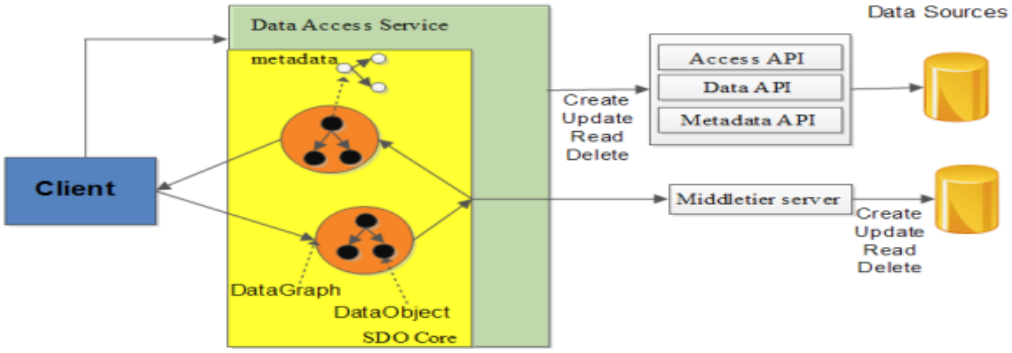

Fig.3. Service data object architecture

2.1.2. Components of Service Data Object

Data Object: Data objects contain the actual data, which have primitive values and referencing to other data objects. These data objects also do referencing to their metadata, which allows DO (data object) to be examined for information about data types, data relations and data constraints [13]. As shown in Fig.4.

Technium Vol. 9, pp.26-40 (2023)
ISSN: 2668-778X
www.techniumscience.com

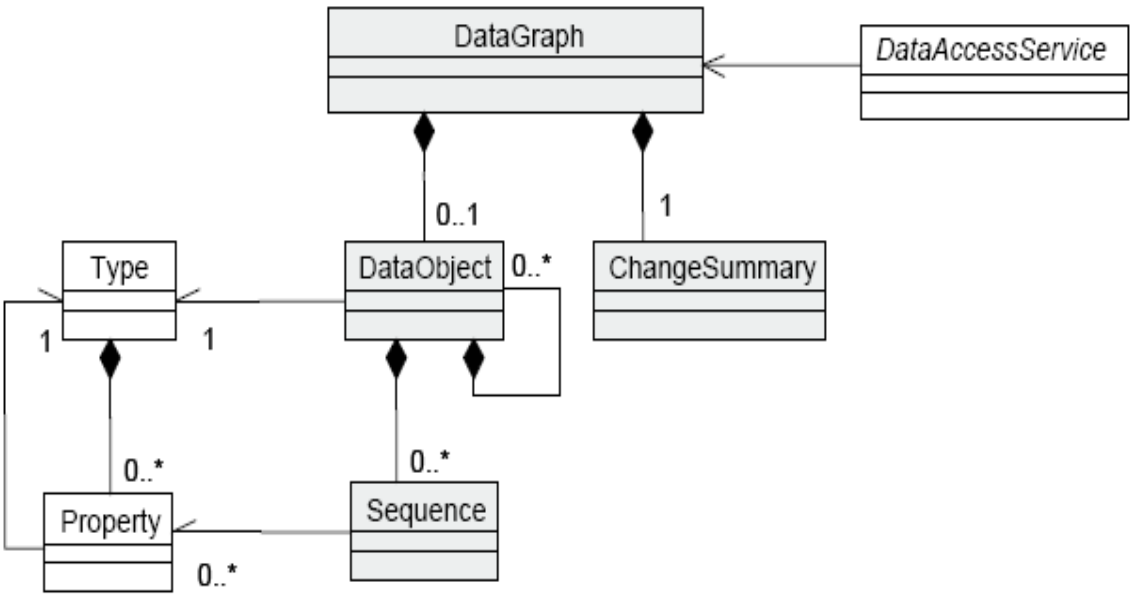

Fig.4. UML model of core SDO components

2.1.3. Virtual Data Access

Virtual data access is very important requirement in enterprise environments in where data provision functions are formally separated from data use function for several reasons including; security, skills, and governance. This separation is important in SOA (Service-oriented architecture) implementation too. Service oriented architecture gives the unified client data and metadata API that enable framework, tools and runtimes to be able to function in this environment [14]. As shown in Fig.5.

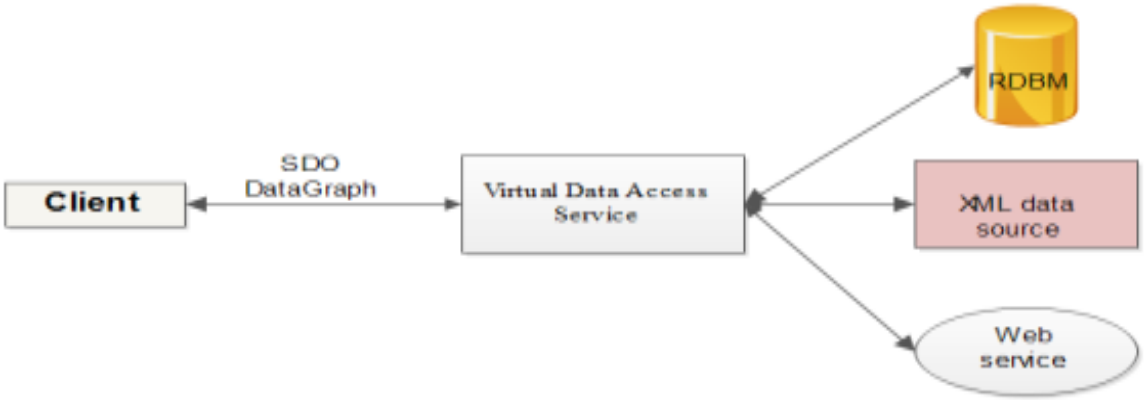

Fig.5. Virtual data access based on SDO

3. Technologies of Systems Architecture

The control system which deals with the access of data from multiple data sources specifically concentrates on a central database which controls the business access database through use of program's applications. The technology involves unified access control server software and unified

data access server software the unified data access server determines if it shall access by the users after asking the unified access control server. Similarly, a unified data access server is in control of transforming and optimizing the access to database. Based on this, the unified data access shall divide these systems into various subsystems like the unified access control service [15].

At the time a user logs in, he is able to analyze and extract the permission of the user after reviewing the amount of information's then decides if it shall allow the user to access the system. Once the user is given permission, the function shall then he shall access the system including the database after judging the request and the system then redistributed in consuming the execution space and the access of control services [16], as shown in Fig.6.

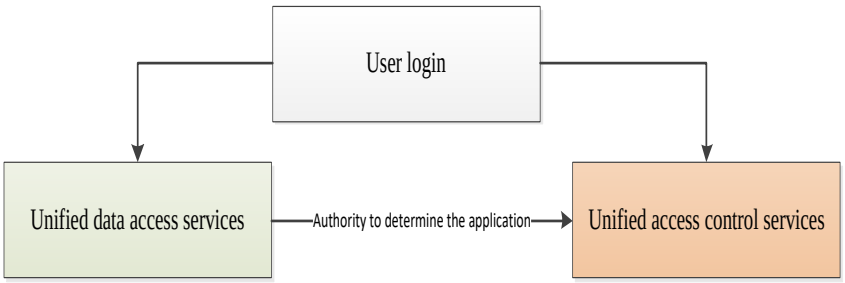

Fig.6. System architecture

3.1. Design and Structure of UACA

UACA is client-side server approach which was developed and designed to work with the Host server (Central database server system) to achieve unified data access interface to a heterogeneous database and data integration between the client applications and data sources. From an operation and maintenance standpoint, it offers convenience and greatly reduces the costs of operation, maintenance, management, making it easier to develop application software. System communication uses TCP/IP protocol [17], data uses XML format [18][19], and data objects are an SDO representation of structure data.

In heterogeneous data integration based on XML, it is a key approach implementation to achieve local database and XML mapping. In order to realize the data amendment of the XML generated by SQL to XML method, use the JDOM to convert XML file to the Document object in the application. You can then traverse the structure to find, display and modify the appropriate information.

3.2. UACA Structures and Functions

The Unified Access Control Approach (UACA) consists of two subsystems: the unified data access interface system and the unified access control approach. Complete control of unified data access services to business database applications that access a central database is required, as this ensures that all business access to the data access service goes through the middle layer, achieving access to the database and centralized data access control. Additionally, it is in charge of optimization and transformation. Unified access controls services to complete the user authentication, access control, unify access interface services and integration services, and integrate data access with clients' applications using SDO, XML technologies. Additionally, it manages system communication using

the TCP/IP protocol to guarantee the correctness and stability of communication. Data utilizing XML is adaptable and compatible. , as shown in Fig. 7.

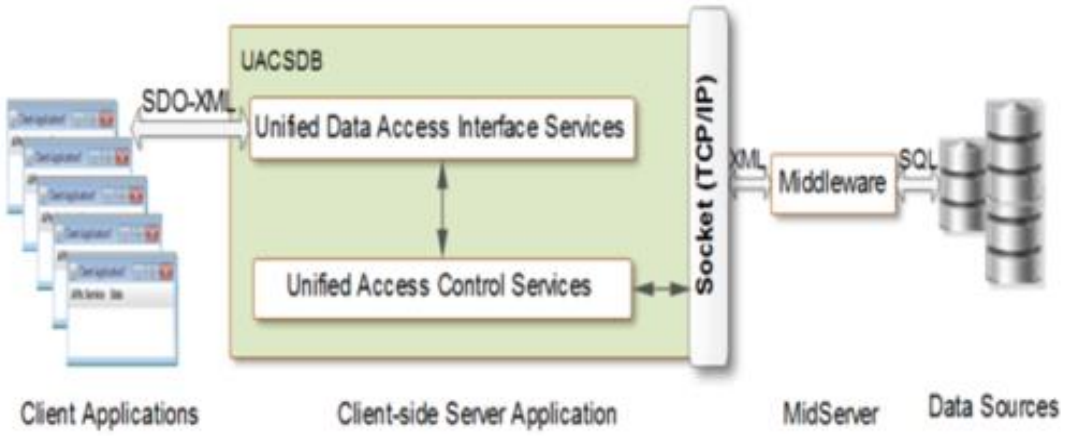

Fig.7. UACA structures and functions

3.3. Implementation of Centralized Access Control System Process

Here, the session ID (SID) and the User Login that applies the log on details from the centralized database site that is consolidated and integrated from the system access control. The use from the user like password, name and login details from the Web site of the organization.

The verification of the user logging enterprise or company notifies the central database like the access control system and the user offering the SID of the user login details. Besides, there is also the web client calls program that offers the SID call information and the name through the user's computer and the browser like calls where the user computer is contained in the client's programs. By the client accessing the central database, there involves the data access of the control system that makes the port proxy request that receives in the specific process including the details involved.

The User's client program is able to access the central database from the central system database and the centralized access aimed at controlling the process from the client program that sends the requested data to the client. This process is known as SOCKET programming that applies TCP communication [21][22], as shown in Fig. 8.

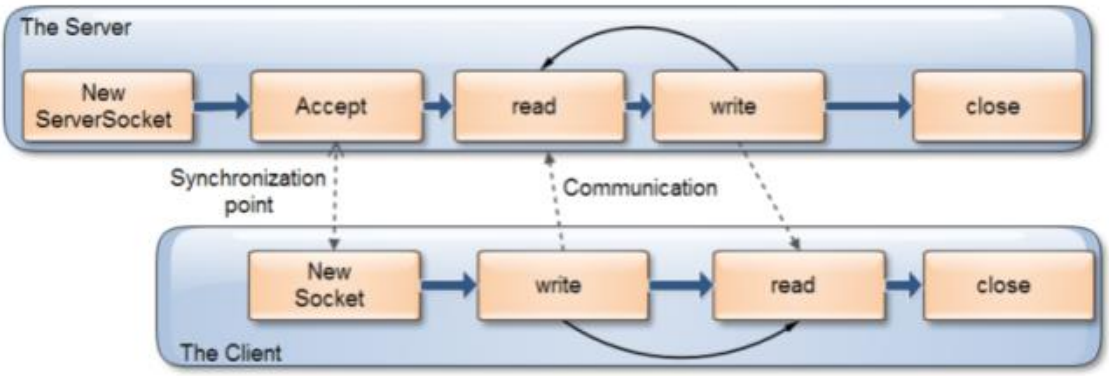

Fig.8 A SOCKET communication programming interface

3.4. User Login and Access to SID

Users logging services to log on into the central database site in host server. Using user name and password, users log on to the logging company's web site. They are verified by the user logging company's web site, which notifies the central database site control access system user login object that the user has logged in and access to the central database system of centralized access control object provides the user login session ID (SID) [23].

3.4.1. Web Client program calls

The web site for the client program provides the SID, well name and other call information via the browser of user's computer. The client program send a data request of a client access to the central database focused on data access control system made the port proxy object data request service. The format of data request followed by details. Client data requests sent and received the results of the specific process, followed by details.

3.4.2. Dealing with Client Data Sets Obtained

The clients focus from the central database access control system for data access used a port proxy object to receive the results of the client's requests. If the query is 'SELECT', then get the XML format of the result data set. If the query is 'INSERT/DELETE/UPDATE' get the result code as successful or failed operations. The format of the data request returns information, followed by details. Client data requests sent and received the results of the specific process, followed by details. The specific content of each tag, according to the actual situation changes return value conventions, as shown in Table 1. Here some request Format technique, as example:

- Query action (select):

```
<?xml version="1.0" encoding="UTF-8" standalone='yes' ?>
<dbmcall>
    <sid>SID</sid>
    <action>Select</action>
    <fields>Column-Name </fields>
    <data></data>
    <tables>Table-Name</tables>
    <where></where>
    <option></option>
</dbmcall>
```

Return Message Format:

```
<?xml version="1.0" encoding="UTF-8" standalone='yes' ?>
<result>
<ROW>
<Column-name>Data sets</ Column-name >
</ROW>
</result>
```

- Query action (Delete)

Request Format:

Technium Vol. 9, pp.26-40 (2023)
ISSN: 2668-778X
www.techniumscience.com

```
<?xml version="1.0" encoding="UTF-8" standalone='yes' ?>
<dbmcall>
 <sid>186a5c2a-8690-40d4-99b0-fd1d9f65affc</sid>
 <aid>a001</aid>
 <action>delete</action>
<tables>TableTest</tables>
 <where>EMPID=221</where>
</dbmcall>
```

The Insert/update/delete request the return message format is:

```
<?xml version="1.0" encoding="UTF-8" standalone='yes' ?>
<result>
        <resultcode>1/-3</resultcode>
                                        </result>
```

Table 1 Return value conventions

Result code	The code value meaning
Ready	After a successful connection, one makes an inquiry from the SQL requests. At the phase of socket connection
Busy	Connecting the Socket pool that is busy. Here, one tries again later. At the phase of socket connection.
1	Update/Insert or Delete Success.
-1	Denied Access. The access of the user is banned since the user does not have access
-2	No Free DB Connection. The database connection is not idle
-3	Insert/Update/Delete Failed.
OK	A Successful Operation

The implementation process steps in UACS with centralized access control system process see Fig. 9.

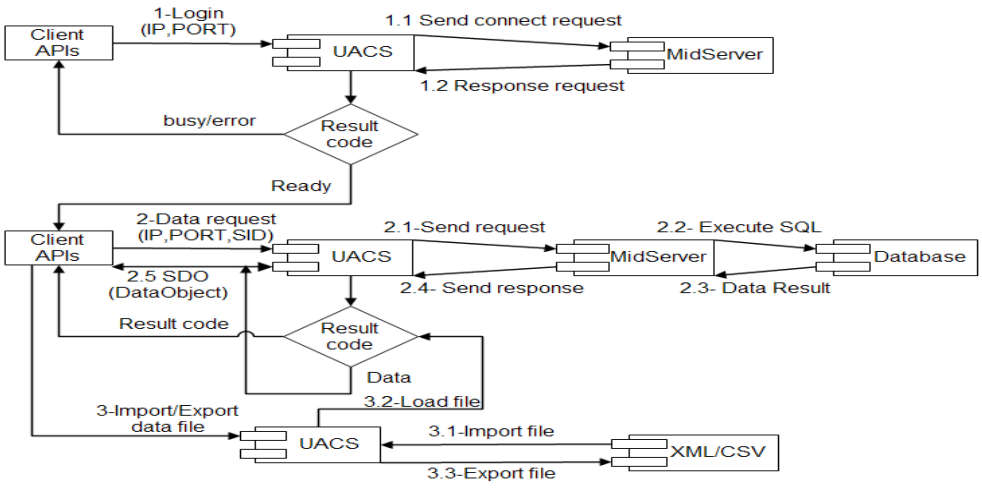

Fig.9. Implementation process steps for UACA model

4. Users Login

To achieve easy connect (login) with UACA a special interface was designed, as shown in Fig.12 and Fig.13 in Appendix A. The client user can used this interface to enter the initial connect such as IP address, PORT, and SID, more details as shown in Appendix A:

- Precondition:
  - Connect details of host server (IP, PORT).
  - Generating a unique session ID (SID).
  - Activation network.
- Trigger:
  - Users decide connect to server database.
- Main success scenario:
  - User selects connect service from client application interface.
  - User enters the connect information (IP, PORT-SID).
  - User select connect button.
  - UACS send request to contact the database host server (Mid-Server) through socket server programming.
  - Host server sends response to UACA.
  - UACA view success message to user (Host severer is ready).
- Exceptions
  - Problem in the network.
  - Host server is not known (IP or PORT) error.
  - State host server is busy or unknown.
  - User does not have the authority to contact to data source or SID is error.

Sequence diagram of request connect to host server with UACA , see Figure 10.

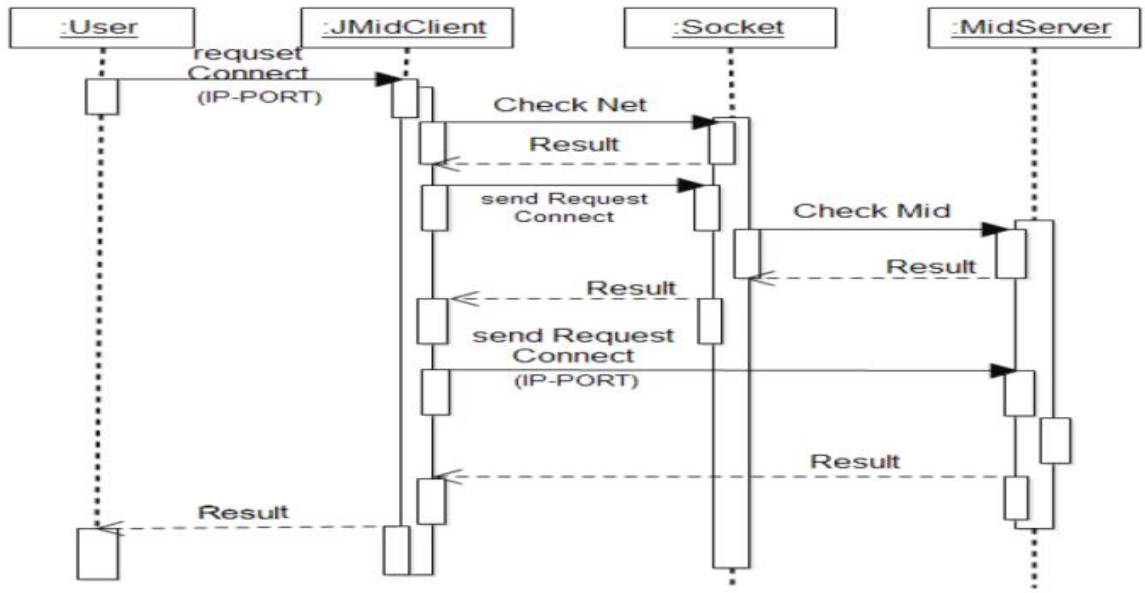

Fig. 10. A sequence steps for request connection to host server based on UACA.

5. Data Access Process, Analysis and Decision

The Case study is to design a unified interface to access the employee's data through UACA and achieve unified access, integration of data between the client application and source data, as shown in Figure 11. We have implemented queries on data processing and storage and get values depending on the size of data and execution time and analyzed the results. The costs query for data access processing and analysis, as shown in Table 2. We have decision evaluation this model with new technologies such as Apache Spark [24][25] and Hadoop MapReduce [25] in the feature work with issues of a time costs for large datasets analysis and process.

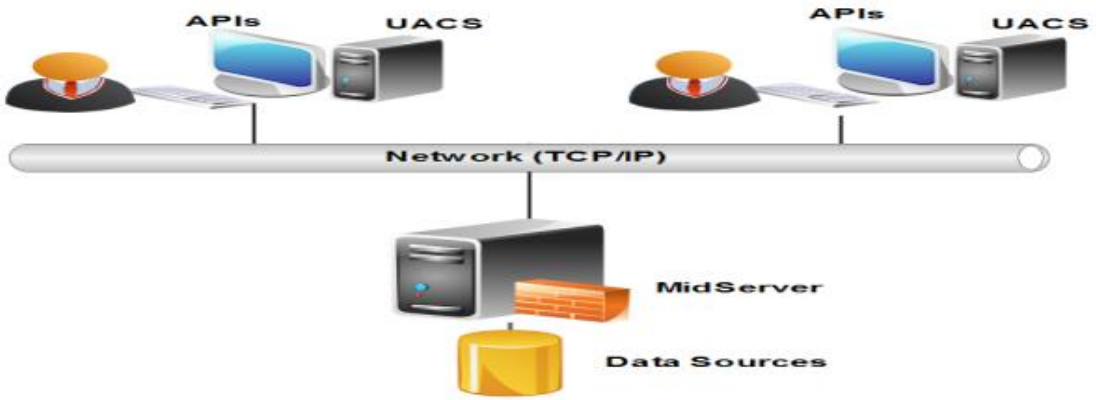

Fig. 11. Case Study of UACA

Table 2 A cost query for data access process and analysis

Data size (GB)	Start	End	Time cost
0.0015	10:27:53	10:28:14	0:00:21
0.0027	10:35:39	10:36:00	0:00:21
0.94	10:40:21	10:41:22	0:01:01
1.88	10:43:25	10:47:14	0:03:49
3.76	10:52:25	11:09:59	0:17:34
7.52	11:11:52	12:42:03	1:30:11

6. Conclusion

Designing software that can search through heterogeneous data and access data sources is a challenging endeavor since it necessitates managing a variety of issues, including query language support, data heterogeneity, and various access techniques. In this study, we suggested a SOA framework based on SDO that uses UACA to exchange and access diverse data sources. The framework's architecture components and the duties that agents should do were then represented. Additionally, restricted processes of user input, schema matching, metadata mapping, query creation, and execution have been defined for implementation. Without altering the existing model components, the UACA design allows for the addition of new model components as well as the development and improvement of SDO. In the future work to develop new service models to the

UACA to be able to complex query and work with big data based on Apache Spark and Hadoop. This development depends on parallel with development of host server system.

References

[1] Brodersen K, Rothwein TM, Malden MS, Chen MJ, Annadata A, inventors; Siebel Systems Inc, assignee. Database access method and system for user role defined access. United States patent US 6,732,100. (2004).
[2] Thomas TM, Books MT. Java data access: JDBC, JNDI, and JAXP. M&T Books; (2002).
[3] Ausanka-Crues R. Methods for access control: advances and limitations. Harvey Mudd College;301:20, (2001).
[4] Sheth AP, Larson JA. Federated database systems for managing distributed, heterogeneous, and autonomous databases. ACM Computing Surveys (CSUR);22(3):183-236, (1990).
[5] Nong RH. Database Grid: the Multi-database Built on the Grid [J]. Computer Engineering and Applications.;2:050,(2006).
[6] Programming language. Retrieval from: https://wikivisually.com/wiki/ Programming_language, (2017).
[7] Volkoff O, Strong DM, Elmes MB. Understanding enterprise systems-enabled integration. European Journal of Information Systems. 14(2):110-20, (2005).
[8] Yan, Z., Li, X., Wang, M. and Vasilakos, A.. Flexible data access control based on trust and reputation in cloud computing. IEEE Transactions on Cloud Computing, (2017).
[9] Labrinidis A, Jagadish HV. Challenges and opportunities with big data. Proceedings of the VLDB Endowment.5(12):2032-3, (2012).
[10] MacLennan, E. and Van Belle, J.P.. Factors affecting the organizational adoption of service-oriented architecture (SOA). Information Systems and e-Business Management, 12(1), pp.71-100, (2014).
[11] Laskey KB, Laskey K. Service oriented architecture. Wiley Interdisciplinary Reviews: Computational Statistics.1(1):101-5, (2009).
[12] Berg DC, Makin N, Rich LS, Schacher RL, inventors; International Business Machines Corp, assignee. Method and apparatus for generating a service data object based service pattern for an enterprise Java beans model. United States patent US 7,769,747. 2010 Aug 3.
[13] Ferguson DF, Stockton ML. Service-oriented architecture: Programming model and product architecture. IBM Systems Journal.44(4):753-80, (2005).
[14] Yan, Z., Li, X., Wang, M. and Vasilakos, A., 2017. Flexible data access control based on trust and reputation in cloud computing. IEEE Transactions on Cloud Computing
[15] van de Riet RP, Kersten ML, de Jonge W, Wasserman AI. Privacy and security in information systems using programming language features. Information Systems. 8(2):95-103, (1983).
[16] Retrieval form: https://msdn.microsoft.com/en-us/library/cc505882.aspx, (2017).
[17] Badshah F, Shah ST, Jan SR, Rahman IU. Communication between multiple processes on same device using TCP/IP suite. InCommunication, Computing and Digital Systems (C-CODE), International Conference on 2017 Mar 8 (pp. 148-151). IEEE, (2017).
[18] Khan AS, inventor; Oracle America Inc, assignee. System and method for automatically generating XML schema for validating XML input documents. United States patent US 9,286,275. (2016).

[19] Lok SH, Dimantha R, inventors; Protel Communications Limited, assignee. Unified integration management—contact center portal. United States patent US 9,521,207. 2016 Dec 13.
[20] Jirkovský, V., Obitko, M. and Mařík, V.. Understanding Data Heterogeneity in the Context of Cyber-Physical Systems Integration. IEEE Transactions on Industrial Informatics, 13(2), pp.660-667, (2017).
[21] Haritha K, Jawaharlal M, Prasad PM, Chakravathi PK. Design and development of controller software for MAS receiver using socket programming. InComputational Intelligence in Data Mining—Volume 1 2016 (pp. 113-120). Springer, New Delhi.
[22] Binkley, J. "TCP/IP-Socket Programming," (2016).
[23] Burch LL, Earl DG, inventors; Micro Focus Software Inc, assignee. Multiple access authentication. United States patent US 9,391,978, (2016).
[24] Zaharia M, Xin RS, Wendell P, Das T, Armbrust M, Dave A, Meng X, Rosen J, Venkataraman S, Franklin MJ, Ghodsi A. Apache spark: a unified engine for big data processing. Communications of the ACM. 59(11):56-65, (2016).
[25] Alkasem A, Liu H, Zuo D, Algarash B. Cloud Computing: A model Construct of Real-Time Monitoring for Big Dataset Analytics Using Apache Spark. InJournal of Physics: Conference Series 2018 (Vol. 933, No. 1, p. 012018). IOP Publishing, (2018).

Appendix A

A.1 Connect to Server (User Login)

The client user can used this interface to enter the initial connect such as IP address, PORT, and SID. User as well as click on the connect button and wait for the responses from the serve (ready/ busy/ unknown),s shown in Figure 12.

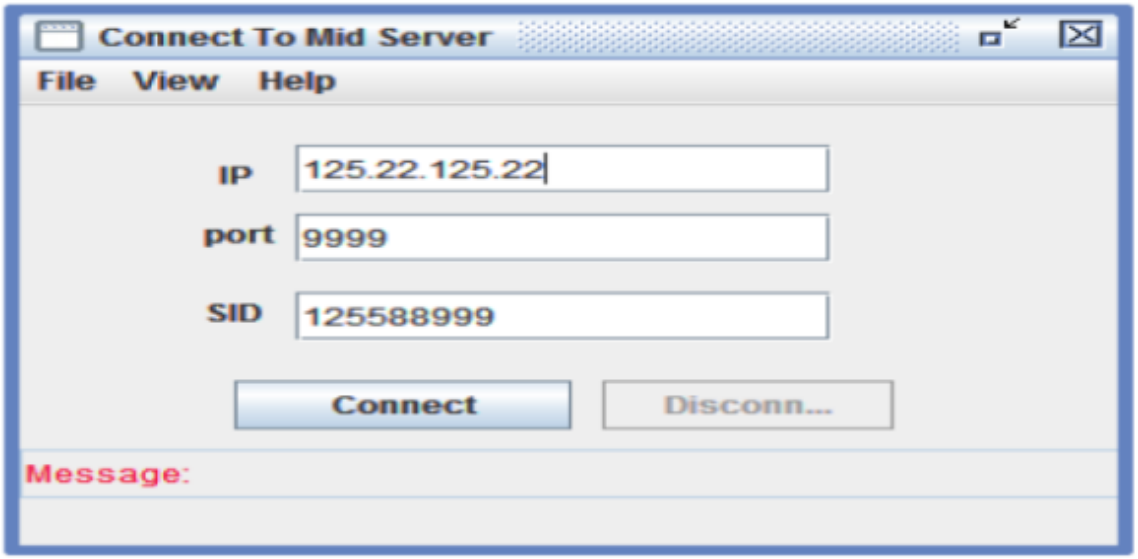

Fig. 12. Case study for connection to host server

Technium Vol. 9, pp.26-40 (2023)
ISSN: 2668-778X
www.techniumscience.com

A.2 View Data from Database (Request Data)

Here, the end users want access to all employees' data from tables and make all data manipulation processing such as select, update, delete, and insert. So, the user has a SQL statement query such as: "*Select empid, empname, empbrithday, empaddress, emptelphon, empemail, empstate, empsalary from testable*". The UACA convert this query into a XML format:

```
<?xml version="1.0" encoding="UTF-8" standalone='yes' ?>
<dbmcall>
  <sid>186a5c2a-8690-40d4-99b0-fd1d9f65affc</sid>
  <aid>a001</aid>
  <action>select</action>
<fields>empid,empname,empbrithday,empaddreess,empjob,empt
elphone,empemail,empstatu,empsalary</fields>
<data></data>
  <tables>TestTable</tables>
    <where></where>
  <option></option>
</dbmcall>
```

After that UACA send this request by socket to host server to execute this query and wait the result data, the data return format is:

```
<?xml version="1.0" encoding="utf-8" standalone='yes' ?>
<result>
    <row>
      <empid>1</empid>
      <empname>ameen</empname>
      <empbrithday>1/1/1977</empbrithday>
      <empaddreess>yemen</empaddreess>
      <emptelphone>student</emptelphone>
      <empjob>0</empjob>
      <empemail>ameen@yahoo.com</empemail>
      <empstatu>0</empstatu>
      <empsalary>1000</empsalary>
    </row>
    <row>
      <empid>2</empid>
      <empname>aymen</empname>
      <empbrithday>1/1/2005</empbrithday>
      <empaddreess>yemen</empaddreess>
      <emptelphone>student</emptelphone>
      <empjob>0</empjob>
      <empemail>150000</empemail>
<empstatu>1</empstatu>
      <empsalary>12000</empsalary>
    </row>
 </result>
```

The UACA receive a result data file and convert the data into a new xml format such as:

```
<?xml version="1.0" encoding="ASCII"?>
<UDBC:Result xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:UDBC="http://www.jclient.com/UDBC"
xmlns:udbc="http://www.jclient.com/udbc" xsi:type="udbc:resultType">
<result>
   <row>
     <empid>1</empid
     <empname>ameen</empname>
     <empbrithday>1/1/1977</empbrithday>
     <empaddreess>yemen</empaddreess>
     <emptelphone>student</emptelphone>
     <empjob>0</empjob>
     <empemail>ameen@yahoo.com</empemail>
     <empstatu>0</empstatu>
     <empsalary>1000</empsalary>
   </row>
   <row>
     <empid>2</empid>
     <empname>aymen</empname>
     <empbrithday>1/1/2005</empbrithday>
     <empaddreess>yemen</empaddreess>
     <emptelphone>student</emptelphone>
     <empjob>0</empjob>
     <empemail>150000</empemail>
<empstatu>1</empstatu>
     <empsalary>12000</empsalary>
    </row>
</UDBC:Result>
```

After that a UACA load this data format with metadata file XSD as following examples:

```
 <?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns="http://www.jclient.com/udbc"
   xmlns:sdo="commonj.sdo"
   xmlns:sdoxml="commonj.sdo/xml"
   targetNamespace="http://www.jclient.com/udbc">
   <xsd:element name="Result" type="resultType"/>
    <xsd:element name="status" type="StatusType"/>
   <xsd:complexType name="resultType">
      <xsd:sequence>
         <xsd:elementname="result"  type="RESULT"/>
         <xsd:element name="changes" type="sdo:ChangeSummaryType"/>
         </xsd:sequence>
       </xsd:complexType>
 <xsd:complexType name="RESULT">
```

Technium Vol. 9, pp.26-40 (2023)
ISSN: 2668-778X
www.techniumscience.com

```
    <xsd:sequence>
      <xsd:element name="ROW" minOccurs="0" maxOccurs="unbounded">
      <xsd:complexType>
      <xsd:sequence>
      <xsd:element name="EMPID" type="xsd:int"/>
      <xsd:element name="EMPNAME" type="xsd:string"/>
      <xsd:element name="EMPBRITHDAY" type="xsd:string"/>
      <xsd:element name="EMPADDREESS" type="xsd:string"/>
      <xsd:element name="EMPTELPHONE" type="xsd:string"/>
      <xsd:element name="EMPJOB" type="xsd:string"/>
     <xsd:element name="EMPEMAIL" type="xsd:string"/>
      <xsd:element name="EMPSTATU" type="xsd:string"/>
      <xsd:element name="EMPSALARY" type="xsd:int"/>
       </xsd:sequence>
      </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:schema>
```

Finally, a DataObject display the data results to the end user through Data Grid interfaces, as shown in Figure 13.

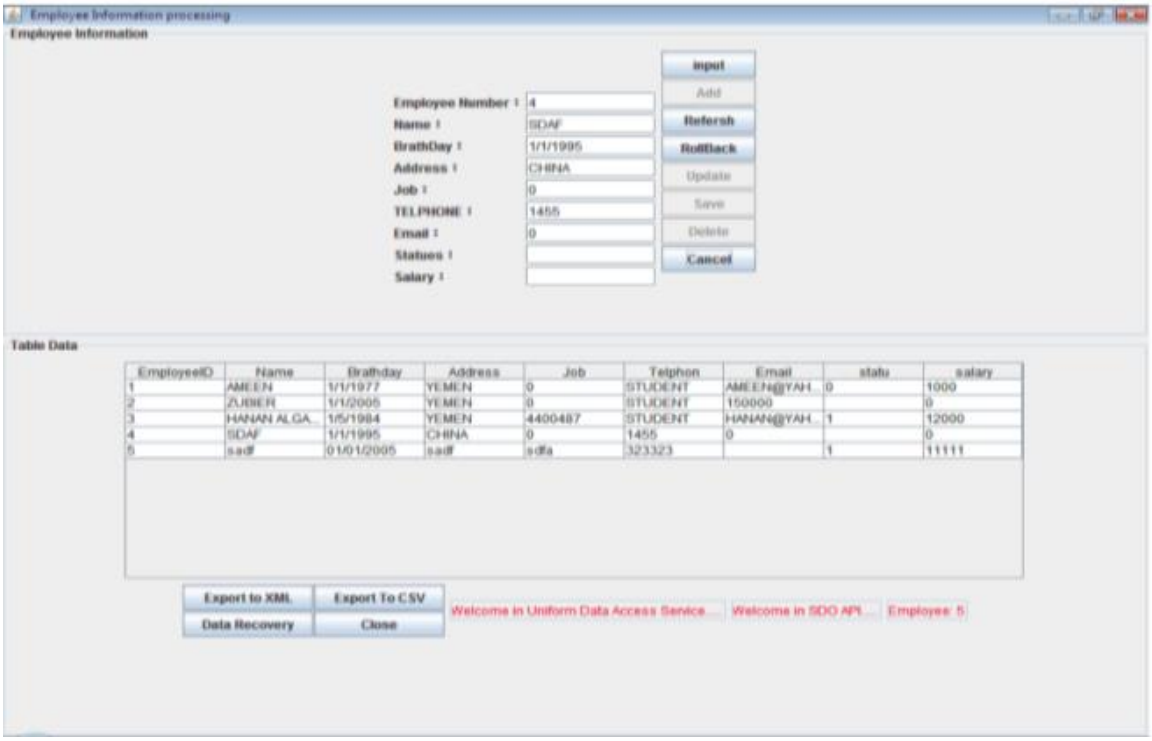

Fig. 13. Uniform data interface service

Submitted: 11.03.2023
Revised: 19.03.2023
Accepted: 22.03.2023