ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)
Volume 2, Issue 2, February 2013

A Three-Layer Architecture based Approach for Data Access Layer in the Information Systems Production

Farhad Soleimanian Gharehchopogh, Esmail Amini, Behnam Zebardast

Abstract— **Software architecture as an important branch of software engineering is one of the significant issues in software production line. It makes communication between system elements and shows us the general structure of the system. Among all the existing and developing architecture, we can point out the layer-based architecture. Nowadays, information systems architecture is mostly three-layer. In this architecture, the task is spitted between different layers and each layer performs specific task. In three-layer architecture there is a specific layer for information management and storing and retrieving data called data access layer. In this paper, by presenting a model for data access layer, we want to maximize criteria for functionality, reusability and reliability. The internal structure of this model is based on a modular and object oriented design patterns. So this independent module can be added to any project that requires secure information access management as data access layer without any changing, and its interfaces can be used to communicate with data and information management.**

Index Terms— **Software Architecture, Information System, Three-Layer Architecture, Data Access Layer, Object-Oriented Design Pattern.**

I. INTRODUCTION

In software production line and designing phase, one of the issues that software designers are facing with, is software architecture that plays main role in reducing complexity and cost of software. Several architectures have been offered for designing software systems that each one of them has some advantages and disadvantages. For sure architecture designing should observe all aspects such as hardware, hosting platform, task holder demands and capabilities of software team in order to offer an architecture with optimized designing and good performance [1]. One of the architectures that have many uses in software production line, especially in information systems, is layered architecture and its aim is to reduce system complexity and also improving reliability and functionality of software development. In this architecture, system components are based on specific layers related to the tasks they do and the role they play, that is very impressive in reducing system complexity and lots of software's other characteristics[2]. One of the successful models of this architecture is three-layer architecture that divides components of system into three different layer based on its uses in a way that, in order to run each action, information should pass three different layer and there should be an appropriate circumstances for running each application, so that the application could run and effect on information [2, 3]. Advantages for this architecture including easily noticeable ease of errors management and security that has made this architecture successful one in software production [3]. In much architecture for information systems a separated and independent part to work with system data is being designed. Also in three-layer architecture; there is specific layer to work with system data and only this layer is allowed to work with system data and it's called data access layer[3,4]. This layer receives credible information from the higher layer and does the saving and retrieving of the information resource. In the internal structure of architecture, only this layer has the permission to work with information storage resource and all other modules and layers in the system have to use data access layer in order to achieve their information. Therefore, this layer acts as bottleneck for information access and has a crucial role in the security and efficiency of information system [3, 4, and 5]. In some information systems there is designed a different and specialized data layer for each separate system. It increases costs and wastes developer's time. However you can dynamically design one layer independent from another layer using object-oriented design techniques of internal structures so that the layer can communicate only through interfaces [5]. In this paper, we are about to show that the three-layer architecture has much greater potential for early changes in the acceptance of information system. We'll do this by analyzing and designing a new model for data access layer. We are also to make this architecture the most qualified by using object-oriented design models and by optimal designing of the layer components. In some parts of paper, in order to clarify the matter much better, we've used library management system as case study and have used Enterprise Architect 7.0 software for drawing diagrams. In section (2) we'll study the general structure and different layers of the three-layer architecture. We explain our proposed model for different layers and will also move toward an optimal architecture. We will focus on data access layer and its internal structure more and will identify its interaction with other layers by representing a new model for this layer. In section (3) we'll study the internal architecture of the proposed model for

Farhad Soleimanian Gharehchopogh *is with the Computer Engineering department, Science and Research Branch, Islamic Azad University, West Azerbaijan, Iran., (corresponding author's website. www.someimanian.com ,*

Esmail Amini *is with the Computer Engineering department, Science and Research Branch, Islamic Azad University, West Azerbaijan, Iran.*

Behnam Zebardast *is with the Computer Engineering department, Science and Research Branch, Islamic Azad University, West Azerbaijan, Iran.*

data access layer and its influence on the system architecture. In section (4) we'll examine the performance of the proposed model and finally in section (5) we'll estate the conclusion and future Works.

II. VARIOUS LAYERS IN THREE-LAYER ARCHITECTURE BASED ON THE PROPOSED MODEL

Three-layer architecture can exist in different layers and various structures and the basic foundation of the system is consisted of three main layers which are responsible for core tasks. User interface layer, business layer and data access layer are core layers of three-layer architecture [6]. Components in the system perform tasks according to their duty but in our proposed model in addition to these layers, there is another layer which acts mostly as an information transfer between the other three layers and thus, it does not affect the internal structure of other layers and architecture of the system. The mentioned layer is called common layer and we'll broadly talk about this layer and its internal structure in the proposed model. Figure (1) shows the interaction between the different layers of three-layer architecture of the proposed model.

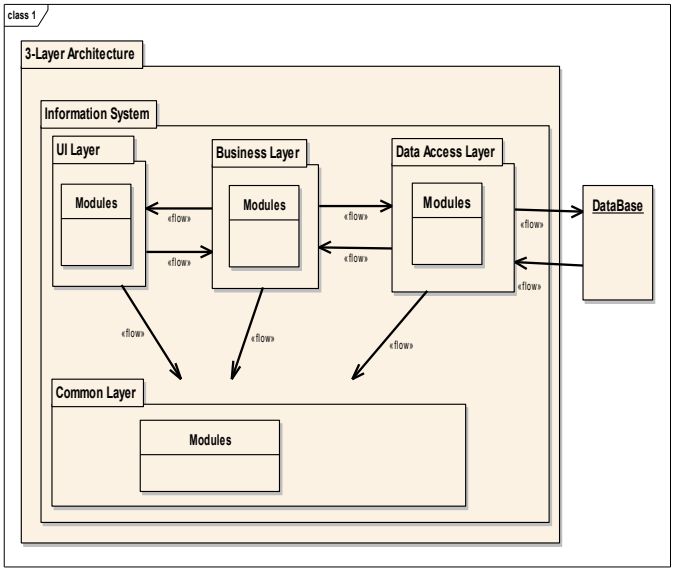

Fig .1: Existing layers in the proposed model based on three-layer architecture and the connection between these layers.

A. Common Layer

This layer is accessible by any other layer in the system and is consisted of components that transfer the information between main layers. In fact, the objects of these components are created and initialized in the user interface layer and these components, then, transmit the information to the business layer and business layer, after some checking, transforms the information to data access layer and this layer then layer after evaluating them, applies the information and performs the required actions on them[6,7].

B. Internal Structure of Common Layer Components

In order to design internal structure of common layer components, it should be kept in mind that these components should observe all information that system's main layers require, and also main information that user requires, actually raw information. Also in this layer entering some invalid information by user can be forbidden [6]. Figure (2) shows internal structure of this layer's components based on proposed model.

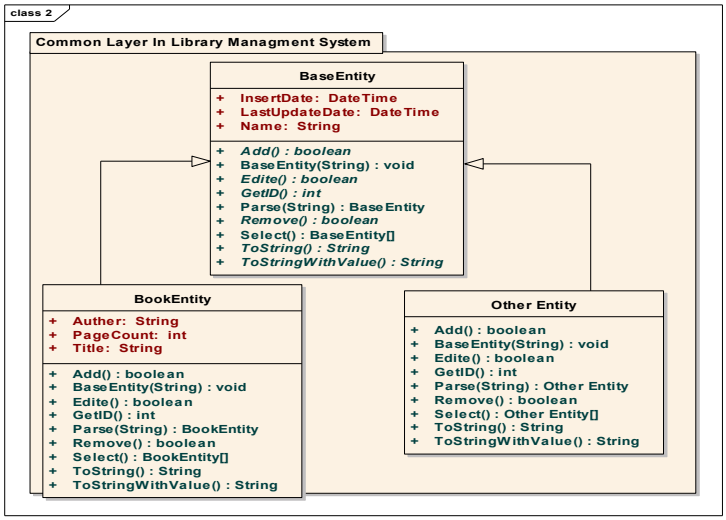

Fig .2: internal structure of components of common layer based on suggested model.

As figure (2) represents we defined a basic class called BaseEntity and we put IDs and actions that are mutual between all components of this layer in this basic class and then all the common layer's classes inherited this class and if there be necessity, they edit some their special components. As you see in figure (2) BookEntity class that could be placed in common layer as book entity class, has inherited BaseEntity class.

C. business Layer

Business Layer is one of the most important layers of the three-layer architecture. It applies the highest complexity on the system (it makes the system the most complex). Therefore, by optimal designing we can reduce the complexity of the system. This layer handles the computations and work logic as well. This layer controls the information received from adjacent layers and then if the information is valid it will be sent to the next layer. These controls may consist of syntax and semantic control [7, 8, and 9]. In most of the proposed models for this layer, this layer is divided into two separate layers which are based on syntax and semantic controls and this task is done in order to reduce the complexity of the layer [8, 9]. According to the things mentioned above, we can have a better understanding of the layer and we can make the layer have a better performance by representing an optimal model for this layer based on object-oriented design patterns. In figure (3), there is shown a structure of the components in this layer based on the proposed model.

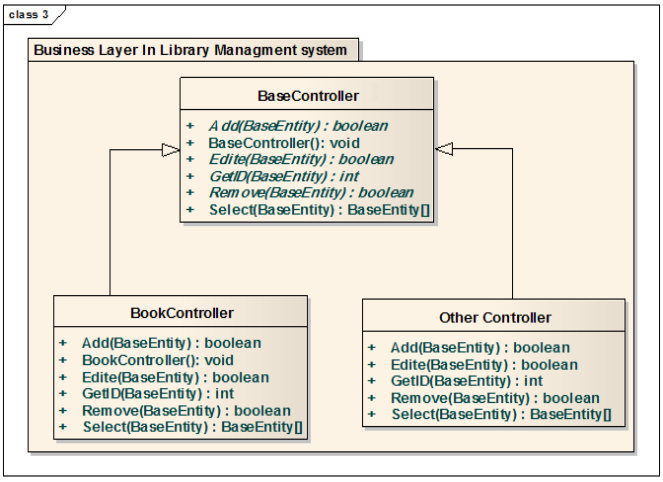

Fig .3: internal structure of existing components in the business layer and their connection based on the proposed model.

As you can see in figure (3), the internal structure of the classes of business layer is represented in the form of BaseController in a way that, the other components of business layer inherit from BaseController and they rewrite its actions. The aim of this task is to establish more and more harmony between the components in the business layer. For instance, BookController class, which is located in the business layer and works as class controller of the book entity in library management system, inherits from BaseController and performs its actions.

D. User Interface Layer (UI Layer)

UI layer which is also is known as presentation layer, is an interface between system ad users. Components of his layer are connected with user directly or indirectly and they transfer information to the next layer by accepting valid information from users. In the view of users this is the highest layer in system. Components of this layer are strongly connected with user, because of that they must be designed based on user's requirements. So it's impossible to offer a model for components of this layer and mostly it depends on user's requirements and manners. In order to offer an optimized structure and standard model without any redundancy and complexity for these kinds of components, the system designer should consist on user basic requirements and should have interaction with them in order to be able to design optimized model to components of this layer [10].

E. Data Access Layer

In information systems, most of system processes are associated with storing and recovering information, because of that in architecting these kinds of systems, the module that is associated with storing and recovering information, gets much attention. Also there is specific layer in three-layer architecture that is responsible for accessing information and its name is data access layer. All processes related to system's information management including insert and remove, edit and recovery of information are responsibilities of this layer. A lot of models have been used and deliberated to data access layer that all of them had some advantages and disadvantages [10, 11]. In some of these models by changing system's information structure, data access layer model must be changed. In fact data access layer is associated with system's information structure and this is one of the serious problems of these kinds of models because it has not capability of reusing. According to the importance of data access layer in information system's architecture, internal architecture of this layer should be designed in a way that it could be used in all systems with any kind of information as data access module and this matter makes the speed of software production process faster[11,12]. Because of hat in our suggested model, in addition to observing capability of reusing, we have considered lots of quality characteristics of software such as capability of development reliance and security capability by designing data access layer and we apply them in our designing [13]. Figure (4) represents general structure of data access layer and its engagement method with an adjacent layer based on suggested model.

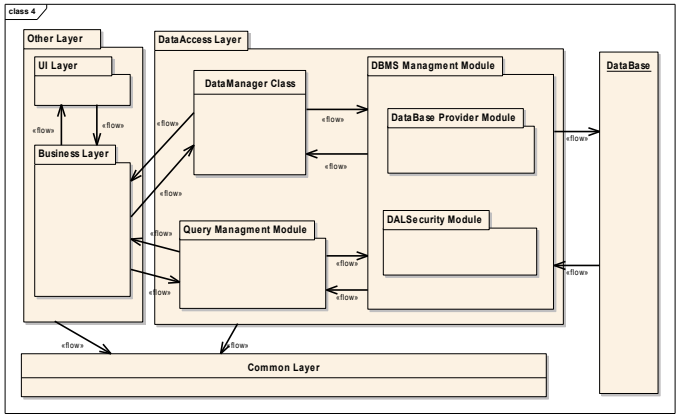

Fig .4: overall structure of internal architecture of data access layer based on the proposed model.

III. Internal Structure Of Data Access Layer

As you observe in figure (4), all system layers and modules are associated with data access layer directly and indirectly. In order to design a model for data access layer we should keep it in mind that what the other modules and layers of system expect from this layer [14]. After recognizing other requirements of system layers, we can offer an appropriate model for that by observing object-oriented design patterns. A data access layer must have the ability to control the information access, to perform the commands related to information edition, to control faults and to manage transactions. A layer which leads us to data must ensure that the information of the system will be comprehensive at any situation [13, 14]. According to the previous needs, from now on, based on figure (4), we'll explain the internal structure of the each of the modules existing in the data access layer.

A. Data Manager Module

Data manager module is one of the major modules in our proposed model for data access layer. This module performs all the main jobs of the system including saving and

retrieving of the information. All system entities can use this module to have connection to their information [15]. Figure (5) shows the internal structure of the Data Manager module.

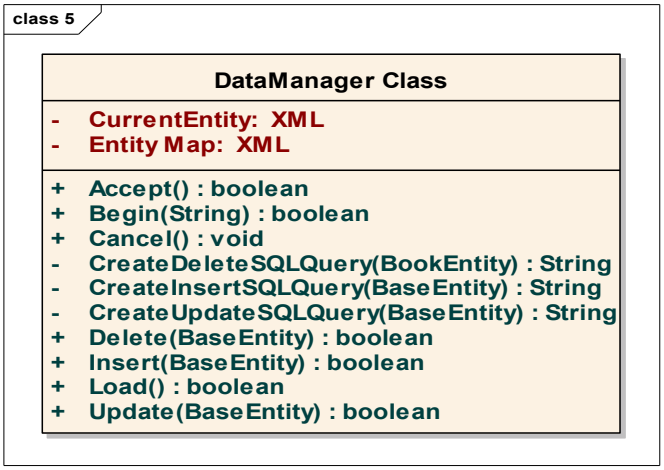

Fig .5: Internal structure of the Data Manager module and its IDs and actions.

According to figure (5) this class consists of some IDs and several actions that each of are used according to their role in the class. Whenever we want to perform any operation on an entity, we transfer the whole previous information from that entity to the CurrentEntity ID. This operation is done by the Begin action of this entity and then we perform all the essential operations on this ID. After noticing that we are done with CurrentEntity ID, we update its information using Accept in the data resource. the general information concerning with data resource, including the number of entities and their names and fields and anything needed for checking the accuracy of the information, are placed in EntityMap ID. For only one time the information is read by Load action and is placed at EntityMap ID at the startup of the system. Actions such as CreateInsertSQLQuery, CreateUpdateSQLQuery and CreateDeleteSQLQuery receive a kind of BaseEntity and produce the SQL command due to that entity and finally turn it back. These actions use EntityMap ID in order to produce the SQL command due to the entity. some actions such as insert, delete, update and select, by receiving objects of BaseEntity type, produce the appropriate SQL command due to the entity using local actions existing in data manager module. We have already talked about these happenings. After producing the appropriate command, the actions perform the command on any entity existing in the memory which is recognized by CurrentEntity ID. This operation will increase the running speed of the commands, especially in a networked environment. After performing all the required operations on an entity, the information of the CurrentEntity will be applied on the actual entity in the data resource by the action called Accept. By offering such a structure, in addition to speeding up the calculation, before applying them on the data resource we'll make sure that the information is valid and thus, it will increase the system reliability. Data manager module possessing these capabilities supports the abilities of the transaction [15].

B. Query Manager Module

In the proposed model for data access layer, there has been considered a module which is specialized in retrieving the information from data resource. Query manager module is responsible for querying the data resource and retrieving required information and then transferring that to the application. Most of the languages for querying data resource do not have an object-oriented structure and thus, the existence of such structures in the data access layer causes the architecture and the structure of the module to be apart from the object-oriented principle [13, 14, and 15]. In the offered model for data access layer, using object-oriented design patterns we've represented a module with object-oriented internal structure called query manager in order to produce and develop the commands for retrieving the information and querying the data resource with object-oriented structure. Figure (6) shows the internal structure of the Query Manager module.

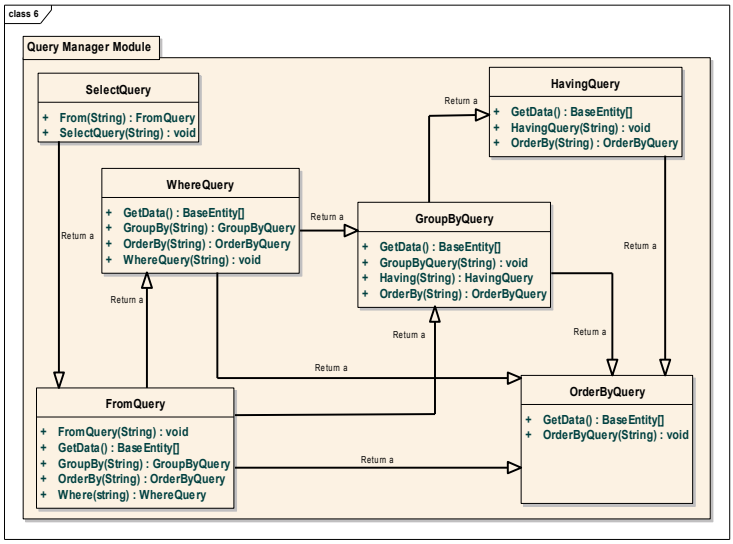

Fig .6: The internal structure of the Query Manager module and the relation between its components.

According to figure (6), query manager module has several components which we've recognized according to the hierarchical structure of the commands of the query languages and then we've encapsulated each command in a component according to its responsibilities. according to the internal structure of the each of the components of figure(6), we have to mention that these structures can be developed when needed but what is certain is that, query manager module must have the minimum components for querying the data resource and withdrawing the required information[14,15]. According to the represented structure for query manager module we were able to represent the required data resource query, using object-oriented design principles. Query manager module has met all the software developer's expectations in the field of retrieving information from data resource and also this module has been represented as a part of data access layer in the offered model [11].

C. DBMS Management module

This module is responsible for performing all the

commands related to data resource. Indeed, all the modules in the data access layer send all their commands and requests to DBMS management module and then DBMS management module performs the commands in the form of a transaction and returns the result of the operations to the other modules. When this module supports transaction-based processing, then we can make sure of information integrity [16]. Since DBMS management module operates in data access layer, it must be able to support any kind of data resource. For this purpose we should enable it, in its internal structure to work with any data resource. In addition to things mentioned above, the information security of input and output from data resource is very important and critical. Data access layer should make sure of the accuracy of the input and also prevent the banned users from accessing the information [15, 16]. Figure (7) shows the internal structure of DBMS management module with its internal modules.

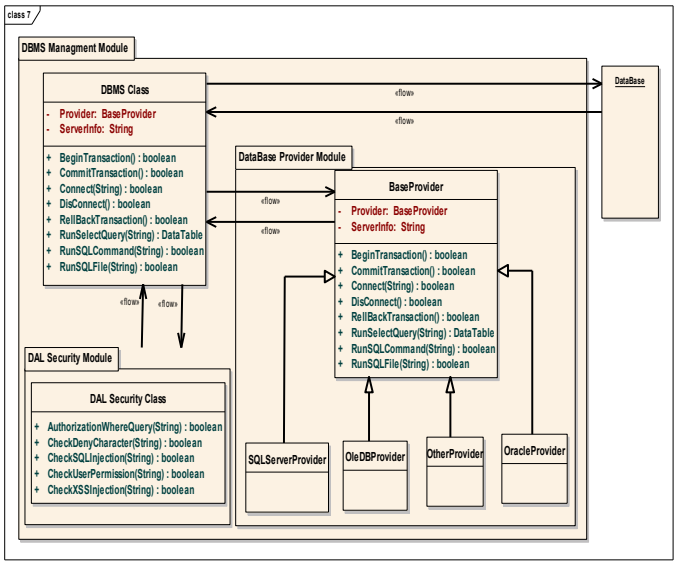

Fig .7: Internal architecture of DBMS Management module and its internal modules.

According to figure (7), DBMS class provides all the requirements necessary for working with data resource. This module provides the required services for higher levels using two internal modules called DataBase provider and DAL Security. DataBase Provider module, according to data resource, produces a special provider using BaseProvider class and then delivers it to DBMS class. DAL security module is responsible for establishing information security for the output and input from data resource and is also responsible for controlling the user's access. After DBMS has recognized the type of data resource, before any other action like saving or retrieving information, it checks the accuracy of the information by the help of DAL Security module [16, 17].

IV. Evaluate the Performance of the Proposed Model

According to the structure offered in previous parts, it can be used as a framework to produce and develop software systems [14]. If we want to give an example of method of using above model in implementation phase in library management system by using C# programming language, we can use suggested model in book registering form in UI layer as below:

1) *BookEntity Book=New BookEntity();*
2) *Book.Namber=12;*
3) *Book.Title="C#";*
4) *Book.PageCount=250;*
5) *Book.Auther="D&D";*
6) *Book.Insert();*

In the first line by defining an object from BookEntity class which inherits from Base Entity class itself, all the information of the Book entity which is located in the data resource is loaded in CurrentEntity ID in data manager component. All these are done by an action in the data manager class which is called Begin, that is time includes connection with database (Tconnect) and reading information related to entity of CurrentEntity (Tc) and it can be presented as this: Tb=Tc+TConnect. In 2,3,4,5 lines entity information are being received and in same time entering amount are being deliberated according to syntax error. As any entity has limited IDs and we represent it by Ca and also times of deliberating according to each ID is limited and it is being represented by Ce, so we will have Tm=Ca*Ce duration average according to 2, 3, 4, and 5 lines that Ce is a number of checking SyntaxErrors according to any ID from any entity and it can be static and Ca is a number of any entity's IDs and because of its limitation we can say that it is static. In line 6 entrance information after being checked by DAL security module, is being registered in virtual entity CurrentEntity . So no time will be spent to connect to database. Required time to DAL security module operation could be represented by Ca, and Ca is a number of any entity's IDs and its deal is static. Required time to read EntityMap information is Te and in startup he system, once it will be performed for whole the system that is negligible. So the total time that his model spends to register some data is such as below:

TTotal=Tb+Tm+Ca

As Ca and Ce are static, Tm time can be passed up and TTotal time will be equal with O(Tb). And because Tb increases according to number of CRUD operations taken on one entity and its increasing deal will be O(1). So by taking n action on one entity O(Tb) will be equal with O(n). As all the other permissions related to work with database, including update, delete, and select are using this method, it can be proved that maximum time based on O(n) will be spent by this model. A point that is very important in this model and improves its performance is that information are being stored temporally in local memory and after doing all responsibilities and user's last confirm, new information in local memory are being transferred to data source and the information is updated by accept action related to data manager component. in fact in this model by creating a local memory for information, in addition to increasing speed of taking actions in network environments, we have reached trust capability and information collectivity to maximum level, because after taking all the actions in one field and

user's last confirm, information are being updated and, they also stop the successive and consecutive trip to the server. From the view of reusing and development capability because of designing independent modules and using necessary interfaces for data access layer, it can be used in lots of information systems and necessary changes can be taken in these modules [15-16-17].

V. Conclusion and Future Work

In this paper, by offering architecture and internal structure of modules presented in previous parts, we proved that our suggested architecture in addition to creating framework to produce and develop information systems based on three-layer architecture, has observed lots of quality characteristics of software such as reusing capability, development capability, trust capability which has been offered in McCall software's quality model. by deliberating reliance of our suggested model we concluded that becoming each UseCase needs o(n) period of time that shows importance of is reliance. By using the model offered in this paper, we can develop module-based and independent systems based on object-oriented patterns. So that software developers can work on a single project in different groups. In this paper we reached to a new framework in data access layer in which by developing other layers of the system simultaneously, we can find a new architecture and framework in producing software in which various and independent modules a work together as a framework and connect with each other, but it will happen in a case that developing this model be based on requirements and principles of information systems in growing trend of software industry and also it should be based on quality principles of software and in future by developing our suggested model by using technologies independent of platform as a part of a module-based architecture, we can develop this model in order to have a secure saving and recovering information and also access to various saving sources in addition to Databases.

REFERENCES

[1] Gomma, H., "Designing Software Product Lines with UML: From Use Cases to Pattern-Based Software Architectures ", Addison Wesley, 2004.
[2] Greenfield, J., Short K., "Software Factories: Assembling Applications with Patterns, Models, Frameworks, and Tools", John Wiley & Sons, 2004.
[3] Nock C., "Data Access Patterns: Database Interactions in Object-Oriented Applications", Addison Wesley, 2003.
[4] A. Neto, H. Fernandes, D. Alves, D.F. Valc´arcel, B.B. Carvalho, J. Ferreira, et al, "A standard data access layer for fusion devices R&D programs", Fusion Engineering and Design 82 (2007) ,pp.1315–1320.
[5] M. Fowler, D. Rice, M. Foemmel, E. Hieatt, R. Mee, R. Stafford. "Patterns of Enterprise Application Architecture" , Addison Wesley, 2002.
[6] F. Irmert, M. Daum, K. Meyer-Wegener, "A New Approach to Modular Database Systems" , SETMDM '08 Proceedings of the 2008 EDBT workshop on Software engineering for tailor-made data management ACM New York, NY, USA ©2008.
[7] R.Barcia, G.Hambrick, K.Brown, R.Peterson, K.S.Bhogal, Persistence in the Enterprise: A Guide to Persistence Technologies, IBM Press, 2008.
[8] A. H. Beg, A. Noraziah A. N. Abdalla, A novel design and development of persistence layer for heterogeneous synchronous replication in data grid, Scientific Research and Essays, Vol. 6(28), pp. 5966-5975, 23 November, 2011.
[9] Yong Chen, Surendra Byna ,Xian-He Sun," Data Access History Cache and Associated Data Prefetching Mechanisms", Supercomputing,2007.SC'07.Proceeding of the 2007 ACM/IEEE Conference on, pp. 1-12,2007.
[10] Lee Lueking , Heidi Schellman , Igor Terekhov, Julie Trumbo , Sini˘sa Veseli , Matthew Vranicar , Richard Wellner , Stephen White , Victoria White," The Data Access Layer for D0 Run II: Design and Features of SAM",pp.1-5,2001.
[11] Mohammadreza Jooyandeh, S. Mehdi Hashemi," Designing An Aspect-Oriented Persistence Layer Supporting Object-Oriented Query Using The .Net Framework 3.5",Computing and Informatics, Vol. 30 ,pp. 621–637,Jun 2011.
[12] Chris Jones, Simon Patton, Martin Lohner, Paul Avery," Design and Implementation of the CLEO III Data Access Syste", Preprint submitted to Elsevier Preprint, pp.1-4,April 1997.
[13] Jun Feng, Lingling Cui, Glenn Wasson, and Marty Humphrey," Toward Seamless Grid Data Access: Design and Implementation of GridFTP on .NET", Proceeding GRID '05 Proceedings of the 6th IEEE/ACM International Workshop on Grid Computing. Pp.164-171.2005.
[14] GholamAli Nejad HajAli Irani," Decentralized Principles: New Modular Software Development Principles, a Robust Object Oriented Approach", International Journal of Computer Applications, Volume 44– No13, April 2012.
[15] Lina Lubyte and Sergio Tessaris, "Supporting the Design of Ontologies for Data Access A Preliminary Investigation",pp.1-11,
[16] Scott W. Ambler," The Design of a Robust Persistence Layer For Relational Databases", June 21, 2005.
[17] Divyesh Jadav, Chutimet Srinilta, Alok Choudhary ,P. Bruce Berra ,"Design and Evaluation of Data Access Strategies in a High Performance Multimedia-on-Demand Server", Proceedings of the International Conference on Multimedia Computing and Systems (ICMCS '95),pp. 286-291, 1995.