Volume 2, Issue 1, January 2012 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Information Management System using LINQ over ADO.Net

Dr. Samrat O Khanna
Institute of Sci. & Tech. for Adv Std & Res.

Mijal A Mistry
Institute of Sci. & Tech. for Adv Std & Res.

Abstract: This paper represents the usefulness of LINQ over the classical ADO.Net. LINQ is the new methodology that is available into .Net framework which provides us the capability to write the queries without knowing the syntax for the query. It provides the flexibility and significance to the development process. Several tasks can be easily achieved using LINQ. We can even also write the complex queries much easily and it also provides the different ways to manipulate the results. In today's world there are several applications existing in which it is require to obtain such kind of easiness and flexibility which LINQ provides and also it has some good performance result. This paper also represents how to write the LINQ queries and what are the prerequisite to use it. This paper also shows how to write the aggregate function into LINQ.

Keywords: LINQ, ADO.Net, .Net, SQL, Query

I. INTRODUCTION

LINQ (Language Integrated Query**)** is a Microsoft programming model that is used to add the feature of formal **QUERY** capability in Microsoft .Net-based programming languages. [1] Language-Integrated Query (**LINQ**) is a new feature introduced in visual studio 2008 and .NET Framework 3.5. Developers can write the queries in the code to retrieve the data from various types of data sources like SQL, XML and XQuery. [2]

It offers an easy and expressive way to manipulate with the data. The main advantage of **LINQ** comes from its ability to apply the same query to an SQL database, a DataSet, an array of objects in memory and to many other types of data as well. LINQ requires the presence of specific language extensions. [1] Earlier developers have to learn the syntax for writing queries for each data source. LINQ simplifies by offering a model to work with the data across various kinds of data sources. [2]

II. ARCHITECTURE OF LINQ

The following figure 1 shows the architecture of LINQ, which can query different data sources using different programming languages:

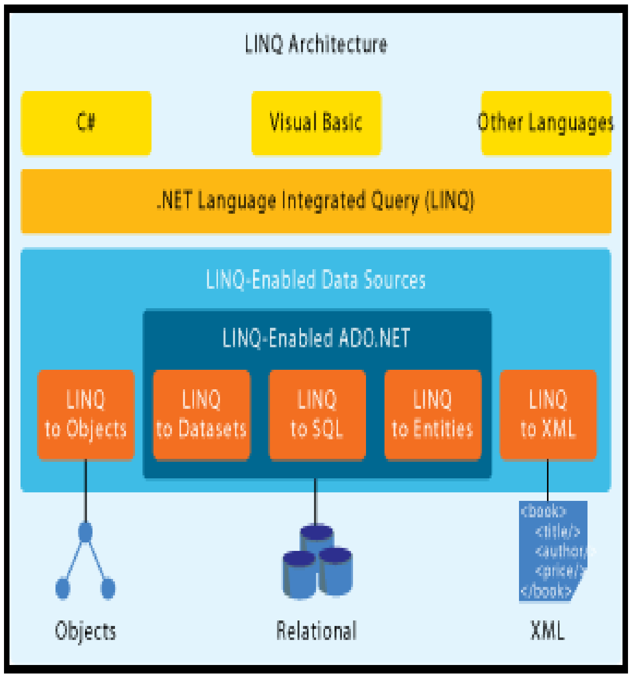

Fig. 1 Architecture of LINQ

1) **LINQ to Objects:** Refers to the use of LINQ queries to access in-memory data-structures. For example we can Query to Array, Lists, Objects

2) **LINQ to SQL:** It is part of the ADO.NET family of technologies. LINQ to SQL translates the Language Integrated Queries in the object model to SQL and sends them to the database for execution. When the database returns the result, LINQ to SQL translates them back to objects that we can work with. LINQ to SQL also supports stored procedures and user-defined functions in the database. For Example we can query the database table directly by using Entity Framework.

3) **LINQ to Datasets:** It is used to query the data cached in Datasets. A Dataset is disconnected, consolidated data from different data sources

4) **LINQ to Entities:** The Entity Data Model is a conceptual data model that can be used to model the data so that applications can interact with data as entities or objects. Through the Entity Data Model, ADO.NET exposes the entities as objects.

5) **LINQ to XML:** Using LINQ to XML, we can query, modify, navigate, and save the changes of an XML document. It enables us to write queries to navigate and retrieve a collection of elements and attributes

LINQ Query operation contains three parts[b]

1. Obtain the data source.
2. Create the Query.
3. Execute the Query.

Important thing to notice is when write a query it do not retrieve any data. In order to get the data from data source you need to execute the query. [2]

Query expression contains three clauses **from, where and select.** The **from** clause specifies the data source, **where** clause applies the filter and **select** clause specifies the types of returned elements. Query executes in foreach statement in the example. The actual execution of the query starts when iterate over the query variable in foreach statement. [2] Queries that perform aggregate operations must first iterate over those elements. For e.g. **Count**, **Max** and **Average**. These execute without an explicit foreach statement.

```
int prodCount = prods.Count();
```

III. DEVELOPED SYSTEM

For convey our point we have developed our own system which uses the LINQ and we have analyzed the performance of it and how it is benefited to more complex systems. We have developed system for electronic products management. The system provides the list of different products and its features, its description, its images, prices etc. based on the query supplied by the user. Earlier people go for traditional database model for writing and executing queries. Here we have used LINQ for writing the queries and it has several advantages over the traditional model of the database. For using LINQ into the application, we can not directly writes the queries into application, we have to add the DataContext into the application which we can connect to a database, retrieve objects from it, and submit changes back to it. We can use the DataContext just as you would use an ADO.NET SqlConnection. In fact, the DataContext is initialized with a connection or connection string that you supply.

The purpose of the DataContext is to translate your requests for objects into SQL queries to be made against the database, and then to assemble objects out of the results. The DataContext enables Language-Integrated Query (LINQ) by implementing the same operator pattern as the Standard Query Operators, such as Where and Select. [4]

The DataContext is the source of all entities mapped over a database connection. It tracks changes that you made to all retrieved entities and maintains an "identity cache" that guarantees that entities retrieved more than one time are represented by using the same object instance. [5] A DataContext instance is designed to last for one "unit of work" however your application defines that term. A DataContext is lightweight and is not expensive to create. A typical LINQ to SQL application creates DataContext instances at method scope or as a member of short-lived classes that represent a logical set of related database operations. [5]

Once we have added the DataContext into our application, then we have to assign the connection string to the DataContext object. Following code snippet shows it.

```
DataContext db = new
DataContext(@"E:\ProductsCollections.m
df");
```

Each database table is represented as a Table collection available by way of the GetTable method, by using the entity class to identify it. Following code snippet shows it.

```
Table<products_master> products =
db.GetTable<products_master>();
```

Now let us write the LINQ query for retrieving the all products from the products_master table.

```
var products =
        from prod in products
        select products_name;
```

If we want to retrieve products say for mobiles then the LINQ can be like this.

```
var prodmobile =
   from prod in products
   where product_cat == "Mobile"
   select products_name;
```

IV. ADVANTAGES OF LINQ

- No magic strings, like you have in SQL queries
- Intellisense
- Compile check when database changes
- Faster development
- Unit of work pattern (context)
- Auto-generated domain objects that are usable small projects
- Lazy loading.
- Learning to write linq queries/lambdas is a must learn for .NET developers. [3]

Following shows the more details comprehensive analysis for the LINQ queries.

Regarding performance:

- Most likely the performance is not going to be a problem in most solutions. To pre-optimize is an anti-pattern. If you later see that some areas of the application are to slow, you can analyze these parts, and in some cases even swap some linq queries with stored procedures or ADO.NET.
- In many cases the lazy loading feature can speed up performance, or at least simplify the code a lot. [3]

Regarding debugging:

- In my opinion debuging Linq2Sql is much easier than both stored procedures and ADO.NET. Linq2Sql Debug Visualizer, which enables you to see the query, and even trigger an execute to see the result when debugging. [3]
- You can also configure the context to write all sql queries to the console window,

Regarding another layer:

- Linq2Sql can be seen as another layer, but it is a purely data access layer. A stored procedure is also another layer of code, and I have seen many cases where part of the business logic has been implemented into stored procedures. This is much worse in my opinion because you are then splitting the business layer into two places, and it will be harder for developers to get a clear view of the business domain. [3]

LINQ in general

- Query in-memory collections and out-of-process data stores with the same syntax and operators
- A declarative style works very well for queries - it's easier to both read and write in very many cases
- Neat language integration allows new providers (both in and out of process) to be written and take advantage of the same query expression syntax

You get a lot of flexibility, as you can limit the columns of what you are getting and it will actually only retrieve that. On the stored procedure solution you have to define a procedure for each column set you are getting, even if the underlying queries are the same.

V. CONCLUSION

By developing the above system using LINQ we came across several of its advantages over traditional one and it also provides the flexibility for writing the queries. Hence it is worth to use it and also we can deal several new components of LINQ which has several capabilities to make the work easier for developers. It also provides the debugging facilities which help the developer to get the idea about the execution.

REFERENCES

[1]http://microsoftmentalist.com/2011/07/06/what-is-linq-introduction-to-linq-importance-of-linq/
[2] http://www.techbubbles.com/net-framework/linq-architecture/
[3]http://stackoverflow.com/questions/593808/what-are-the-advantages-of-linq-to-sql
[4] http://msdn.microsoft.com/en-us/library/bb399375.aspx
[5] http://msdn.microsoft.com/en-us/library/system.data.linq.datacontext.aspx