

OSS ©
FEDERATED SEARCH ENGINE
Middleware

FOR Developers (ASP.NET)

Version: ASP.NET (3.5)
Updated: 4Q/200, g1d0r1
Reference: Middleware 0468/86

CONTENT DESCRIPTION

1. TECHNICAL SPECIFICATIONS

- 2.1 Programming Codes
- 2.2 Database
- 2.3 Application Architecture
- 2.4 Hardware Architecture

2. APPLICATION MIDDLEWARE- INTRODUCTION AND FEATURE DESCRIPTIONS

- 2.1 Introduction
 - 2.1.1 The Background- How to Query for Both Object Orient (OO) and Non-Object Oriented Subjects
 - 2.1.2 The LINQ – The .NET Language Integrated Query
 - 2.1.3 OSS © Federated Search Engine (ASP.NET). v3.5
- 2.2 Application Architecture and Workflow
 - 2.2.1 The Application Work-Flow
 - 2.2.2 The Application Architectural Topology
- 2.3 Business Applications and the Advantages of Using LINGQ Middleware
- 2.4 Application Middleware's Screen Shot Examples

3. HARDWARE CONFIGURATIONS

- 3.1 Hardware Deployment based on HA (High Availability)
- 3.2 Hardware Deployment for Voluminous Concurrent Log-ins
- 3.3 Overall Diagram of Combined Application and Hardware

1. TECHNICAL SPECIFICATIONS

- 1.1 **Programming Tools:** The application's source codes were written in Microsoft ASP.NET (.NET Framework 3.5) and AJAX,
- 1.2 **Database:** The application middleware supports clients to deploy any RDMS as their preferred choice, as itself is a data access layer where it can make seamless connection strings to any preferred RDMS database (Oracle 10/ 11g, MS SQL 2000/2005, IBM DB2 or My SQL);
- 1.3 **Application Software Architecture:** The application's architecture was designed in:
 - A. **SOA** (Services Orient Architecture, details are at Section 2.2 below) which its functions are exposed to the developers as a service; and
 - B. **N-Tier:** It supports the tier-based hardware (cluster-server) environment for deployment.
- 1.4 **Hardware Architecture:** As the core application's base architecture is based on SOA and N-tier, it supports the deployment of layer-based hardware (or cluster-server) in its physical deployment.

In a cluster-server environment, it can be hosted at the Application Server tier. The N-tier is inherently providing clients the great flexibility of having High Availability (HA) and Scalability features. For more details, please refer to Section 3 below.

2. APPLICATION SOFTWARE'S INTRODUCTION AND FEATURE DESCRIPTIONS

2.1 INTRODUCTION OF LINQ

2.1.1 THE ORIGINAL IDEAS & BACKGROUND- HOW TO QUERY for BOTH OBJECT-ORIENT AND NON OBJECT-ORIENTED SUBJECT

.NET Language-Integrated Query

After two decades, the industry has reached a stable point in the evolution of object-oriented (OO) programming technologies. Programmers now take for granted features like classes, objects, and methods. In looking at the current and next generation of technologies, it has become apparent that the next big challenge in programming technology is to reduce the complexity of accessing and integrating information that is not natively defined using OO technology. The two most common sources of non-OO information are relational databases and XML.

Rather than add relational or XML-specific features to the programming languages and runtime, Microsoft has initiated a new project – the **LINQ project** – with a new approach. The project takes on a more general approach and are adding general-purpose query facilities to the .NET Framework that apply to **all sources of information**, not just relational or XML data. This facility is called .NET Language-Integrated Query (LINQ).

2.1.2 THE LINQ – .NET Language Integrated Query

Microsoft purposely uses the term “language-integrated query” to indicate that query is an integrated feature of the developer's primary programming languages (for example, Visual C#, Visual Basic). Obviously, the Language-integrated query allows query expressions to benefit from the rich metadata, compile-time syntax checking, static typing and IntelliSense that was previously available only to imperative code. Now, the LINQ query also allows a single general purpose declarative query facility to be applied to all in-memory information, not just information from external sources.

2.1.3 OSS © FEDERATE SEARCH ENGINE (ASP.NET) v3.5

Our application middleware, OSS © Federated Search Engine (ASP.NET) v3.5, was formerly known as **OSS © Data Access Layer (ASP.NET) v2.0** and written as pure N-tier based codes. The former's programming tools were based on ASP.NET 2.0/AJAX.

The latest version, based on ASP.NET v3.5, is taking a new approach as it is now offered as **SOA (Service Orient Architecture)** and based on the principals of **LINQ**. The SOA platform basically is to expose the search capability (LINQ) as services where the developers can easily embed them in their applications where it helps eliminate the duplicate effort of writing N number of search functions if their applications have N number of modules that call for customized Search Engines.

In addition, we have added the **FEDERATED SEARCH** as optional feature to the LINQ middleware platform. What does it mean? It means that when the users intend to search for a particular thing at their application software, they can search for the sources which would match the key search criteria, from both internal sources (hosted at in-house Ethernet network's) and from the

external world-wide-web (internet)'s, at the single click. For the external search, we provide the option of choosing either Google or Yahoo's. For details, please see the Screen Shots at Section 2.4.

2.2 APPLICATION ARCHITECTURE & WORKFLOW

LINQ (Language Integrated Query) works as a middle tier between data store and the language environment. From a developer's point of view, it is just a new pattern for querying data from multiple data structures directly in the IDE. Behind the scenes it does a whole lot of tasks like expression processing, validation and calling the right routine to fetch data or build a query to run in SQL Server. In short, LINQ stands as common query gateway between the language and the data store.

2.2.1 THE APPLICATION'S WORKFLOW

Below the diagram (**Figure-1**) depicts the general work flow of how LINQ works in crawling out the selected data between the Applications (represented as "C# / VB") and Database (represented as "Data Store").

Figure 1: LINQ workflow (from language to data store)



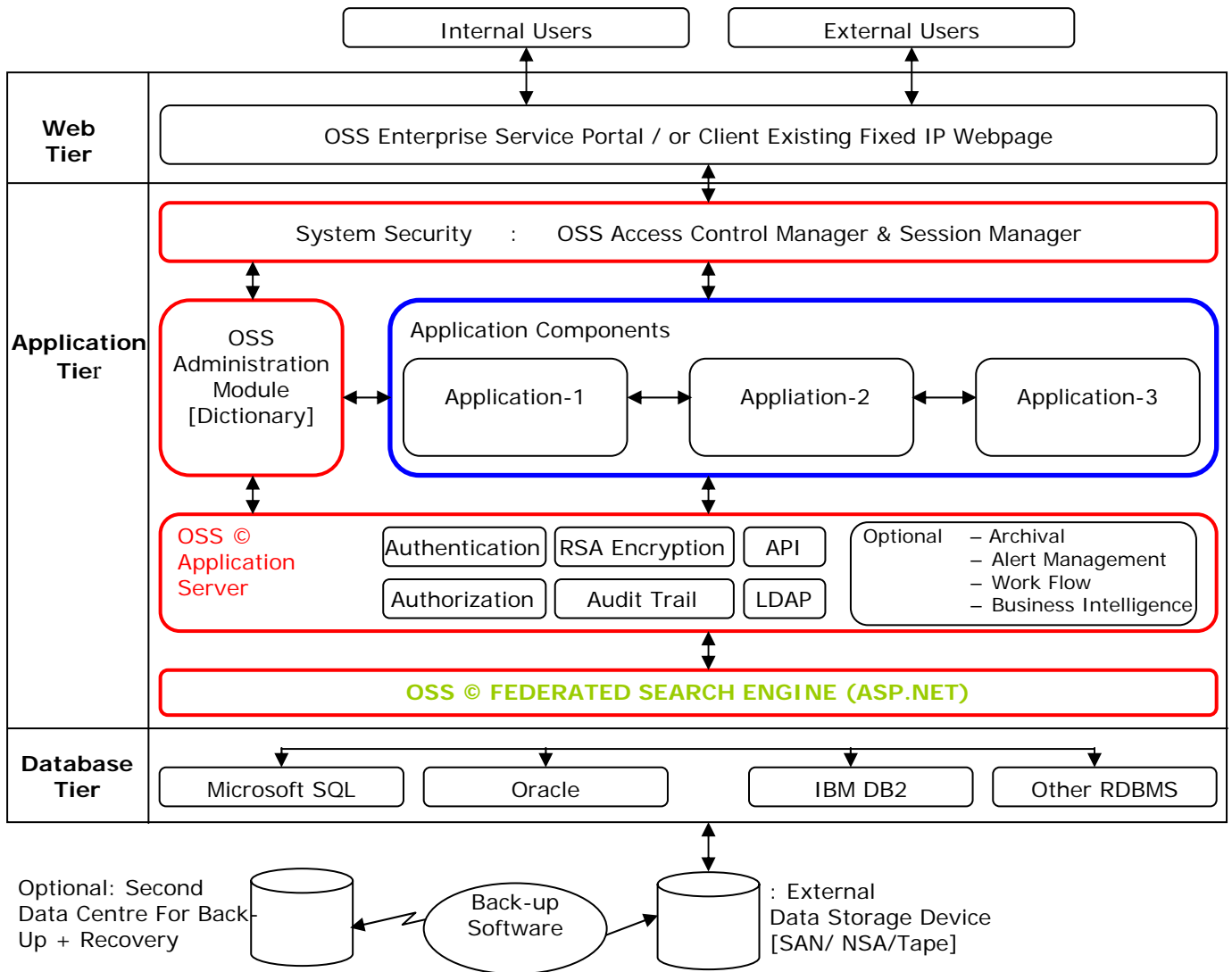
[Note: LINQ basically extends the Microsoft programming tools, C# and Visual Basic, with native language syntax for query operation. In addition, it provides class libraries to take advantage of these capabilities.]

2.2.2 THE APPLICATION ARCHITECTURAL TOPOLOGY

The diagram below illustrates the LINQ's architectural topology where virtually it is located in between the application modules and Database/ storage devices.

For application (or code) development environment, the developers can insert it as one of the syntax in their application and it can be invoked by a simple end-user's mouse click.

Physically, it can be hosted at any of the Application Servers in a cluster environment.



2.3 BUSINESS APPLICATIONS AND THE ADVANTAGES OF USING LINQ MIDDLEWARE

1. **EASY AND VERSATILE:** As the application is written as SOA based middleware where its search services can be easily embedded to the application that the developers are developing at.
2. **WORK with CLR (Common Language Runtime):** The core query capability of LINQ (Standard Query Operators) works with plain old CLR (Common Language Runtime) objects. Then there is XLinQ (for Xml) and DLinq (for databases). DLinq is the ORM piece of LINQ. This means that you (the developers) can query many types of data formats, such as XML, data in a database, and plain old CLR objects all at once. In short, you can do few tasks together at one shot by either just selectively select, then compile the matches (among many other things) XML, relational data, and in memory objects into a single result.
3. **Creating Customized LINQ Query:** As LINQ is a new language that supports querying virtually any data source - such as [SQL](#), XML and in-memory data structures (such as Dictionary, List and more). It gives us (the developers) the opportunity to create custom LINQ providers to give querying capabilities to our API, services and data access layer.
4. **Working with XML/SQL**

4.1 LINQ to SQL

LINQ to SQL, a component of Visual Studio, provides a run-time infrastructure for managing relational data as objects without losing the ability to query. It does this by translating language-integrated queries into SQL for execution by the database, and then translating the tabular results back into objects you (the developers) define. Your application is then free to manipulate the objects while LINQ to SQL stays in the background tracking your changes automatically.

LINQ to SQL is designed to be non-intrusive to your application.

It is possible to migrate current ADO.NET solutions to LINQ to SQL in a piecemeal fashion (sharing the same connections and transactions) since LINQ to SQL is simply another component in the ADO.NET family. LINQ to SQL also has extensive support for stored procedures, allowing reuse of the existing enterprise assets.

LINQ to SQL applications are easy to get started.

Objects linked to relational data can be defined just like normal objects, only decorated with attributes to identify how properties correspond to columns. A design-time tool is provided to automate translating pre-existing relational database schemas into object definitions for you.

4.2. LINQ to XML: XML Integration

.NET Language-Integrated Query for XML (LINQ to XML) allows XML data to be queried by using the standard query operators as well as tree-specific operators that provide XPath-like navigation through descendants, ancestors, and siblings. It provides an efficient in-memory representation for XML that integrates with the existing System.Xml reader/writer infrastructure and is easier to use than W3C DOM.

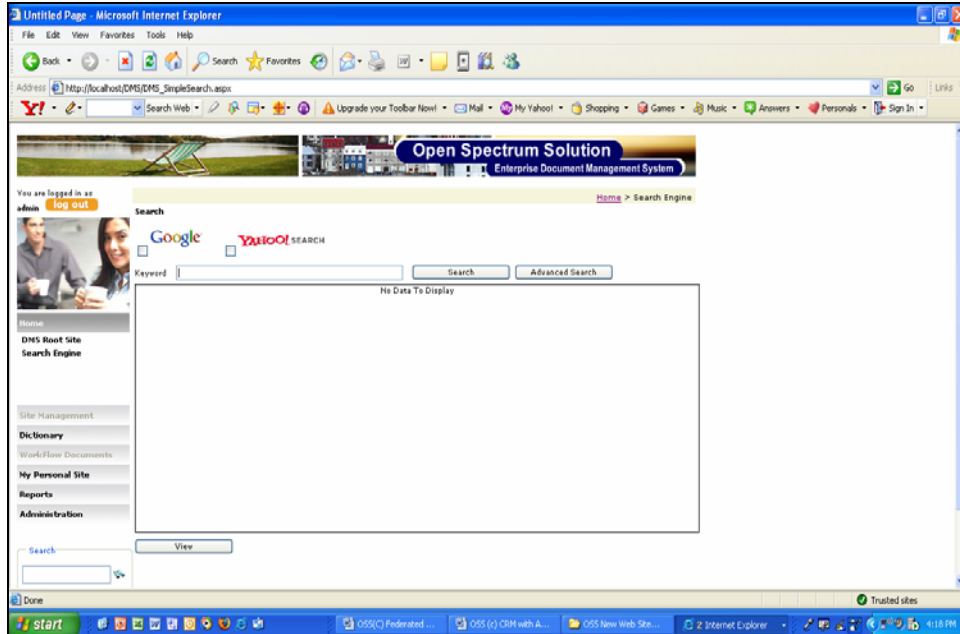
5. Developers can use LINQ with any data source: Developers not only can express efficient query behavior in their programming language of choice; but the middleware also provides great capability to transform the data-query's results into the format they prefer, and this lead to easy end-result's manipulations. In addition, it provides many other development tools' add-on features, such as the LINQ-enabled languages provides type-safety, compile-time checking of query expressions, as well as debugging and rich re-factoring support.

6. Language Features Supporting the LINQ Project

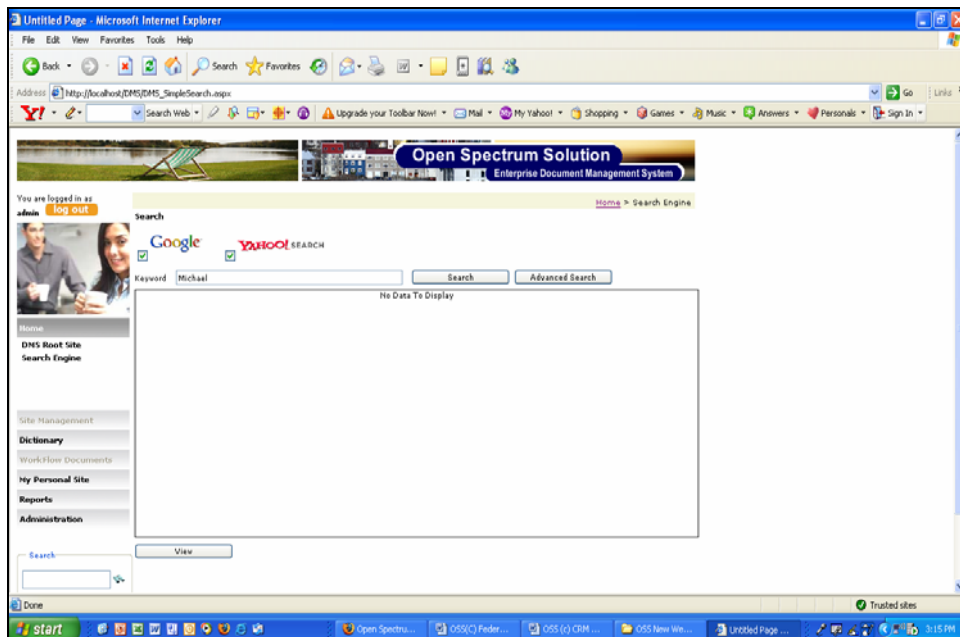
LINQ is built entirely on general purpose language features, some of which are new to C# 3.0 and Visual Basic 9.0. Each of these features has utility on its own, yet collectively they provide an extensible way to define queries and query-able APIs.

2.4 APPLICATION MIDDLEWARE'S SCREEN SHOT EXAMPLES

1. **Federated Search Engine:** This page illustrates the end-user of the application is presented with a Federated Search option to search for particular item in his/her application.

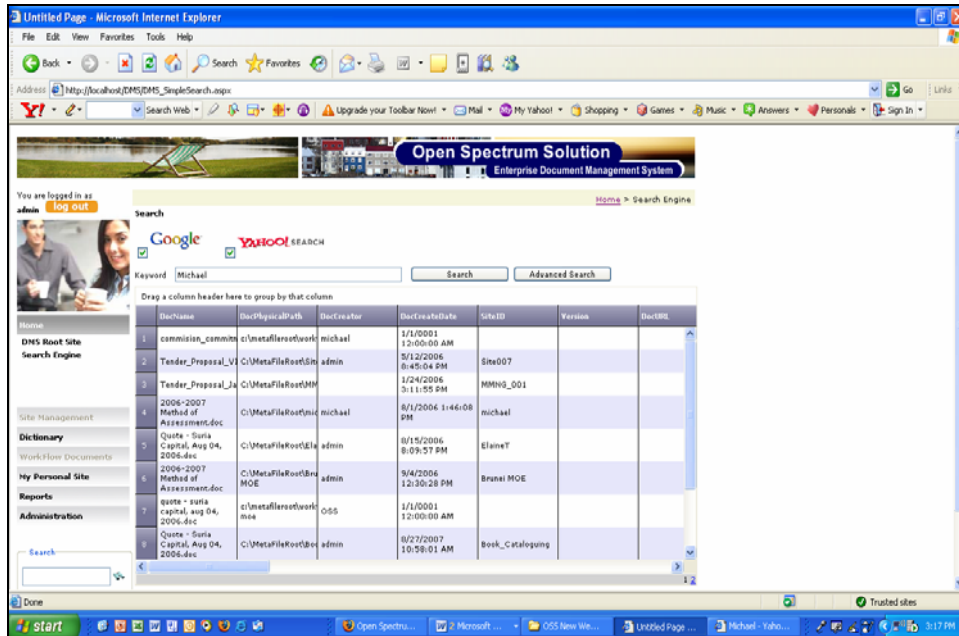


2. **Invocation of simultaneous Searches:** This page illustrates the end-user keyed in a search word as "Michael" at the application's basic search engine; at the same time he/she also checked the Yahoo Search Engine to search for the similar articles in the internet.

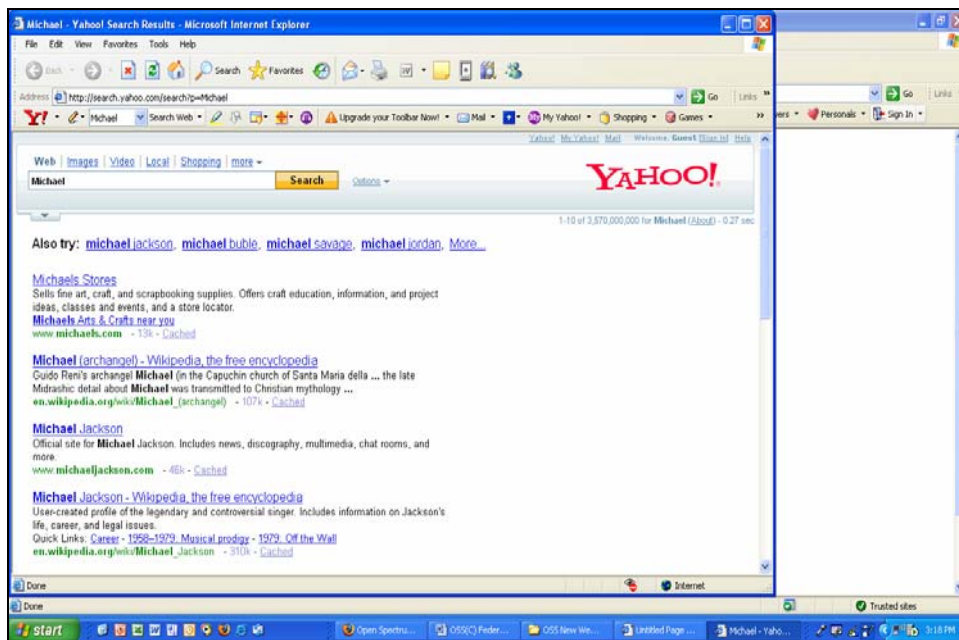


3. **The Federated Search Results:** By invoking the Federated Search Engine, the application will return two (2) types of the search results as show below:

A. **Internal Search Result:** This page illustrates the search result from the application's internal Ethernet's data sources that matched the key search criteria of "Michael"

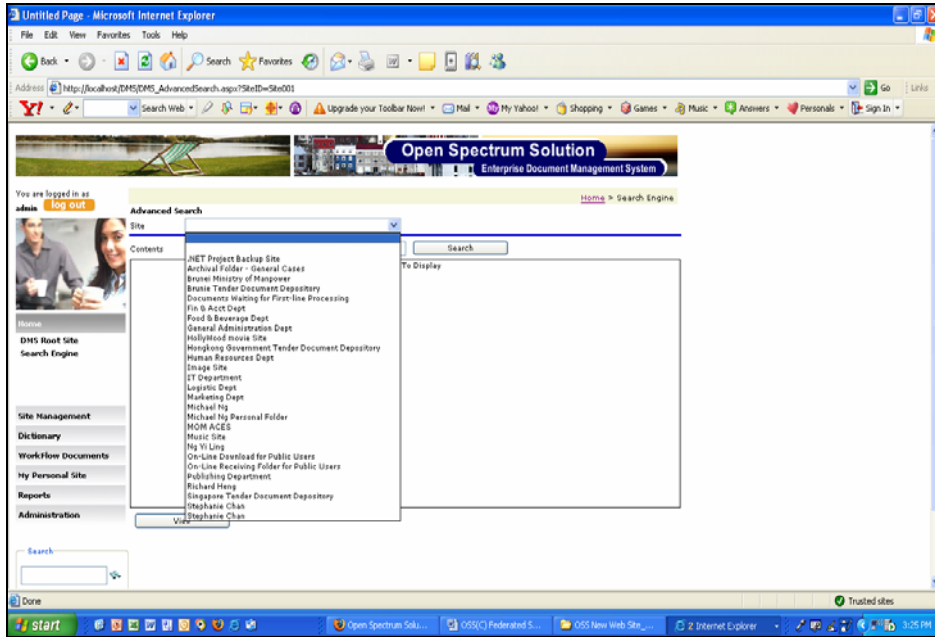


B. The diagram illustrates another search result page from the external sources (from Yahoo's) that displayed articles whose contents matched the criteria search word "Michael", which was open as a separate web browser.



4. **Advanced Search:** The middleware not only supports the basic search engine that the developers can built on, but it also supports Advanced Search capability at their applications as show below.

The Advanced Search's fields can be customized as the middleware permits the developers to add as many as back-end connection as the application's fields are requiring for.



3. HARDWARE CONFIGURATIONS

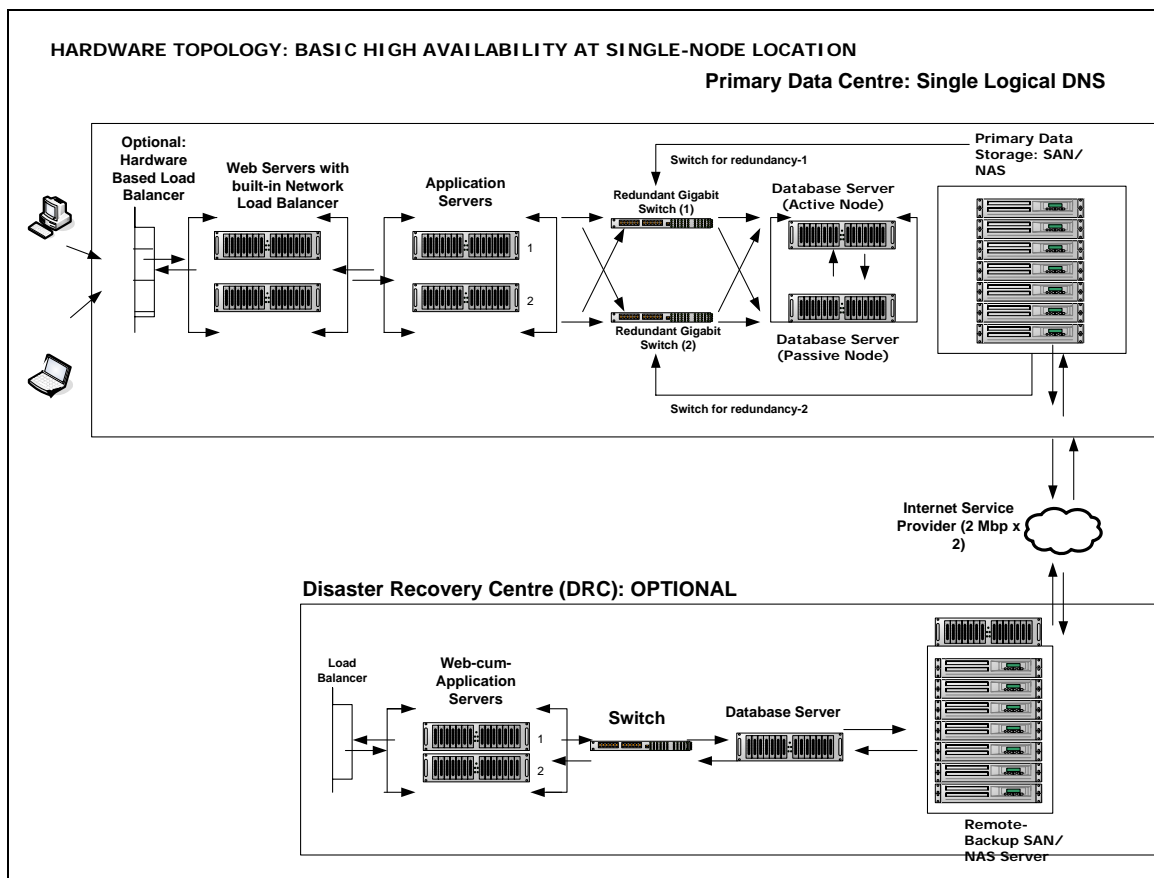
3.1 HARDWARE DEPLOYMENT BASED ON HA (High Availability)

As the application is architected and designed in N-tier, it supports tier based hardware deployment which in turn supports High Availability, Failover and Scalability.

In its most basic hardware deployment, clients can deploy the application with web server, application server and database servers as depicted below.

The design, as described in **Diagram 3.1** below, offers High Availability (Failover), is capable to support approximate 600 concurrent users at any given time, given certain hardware specifications' set-up.

Diagram 3.1



In the event clients intend to scale up the hardware to accommodating more concurrent users (exceeding 600 concurrent users), the design can be scaled up as depicted at **Diagram 3.2's** below.

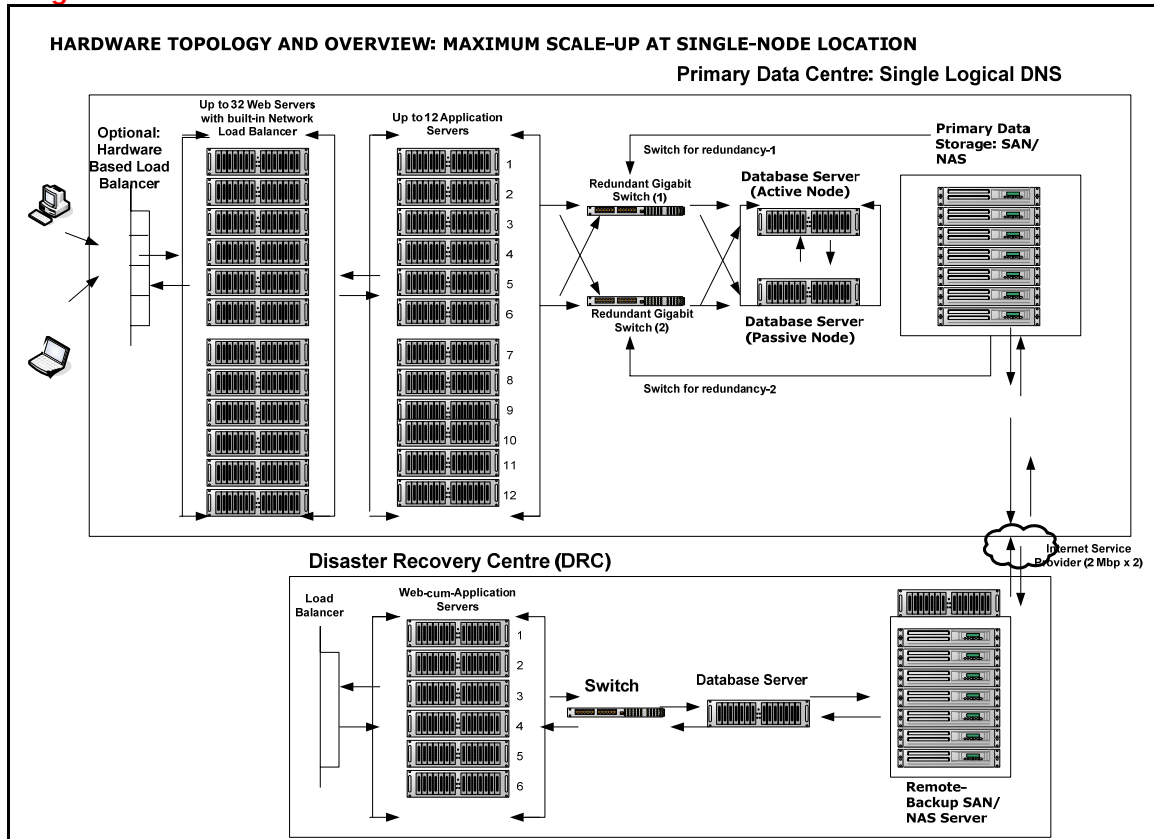
3.2 HARDWARE DEPLOYMENT FOR VOLUMINOUS CONCURRENT LOG-INS

The High Availability (HA) can be achieved thru hardware deployment of up to 32 Web servers, 12 Application Servers and 2 Database Servers at single location as depicted below.

This scaled-up design is capable to support up to 30,000 concurrent users at any given time.

In the event of **Oracle RAC** (Real Application Cluster) is deployed, the number of concurrent users can be extended to 100,000 with 100 database servers.

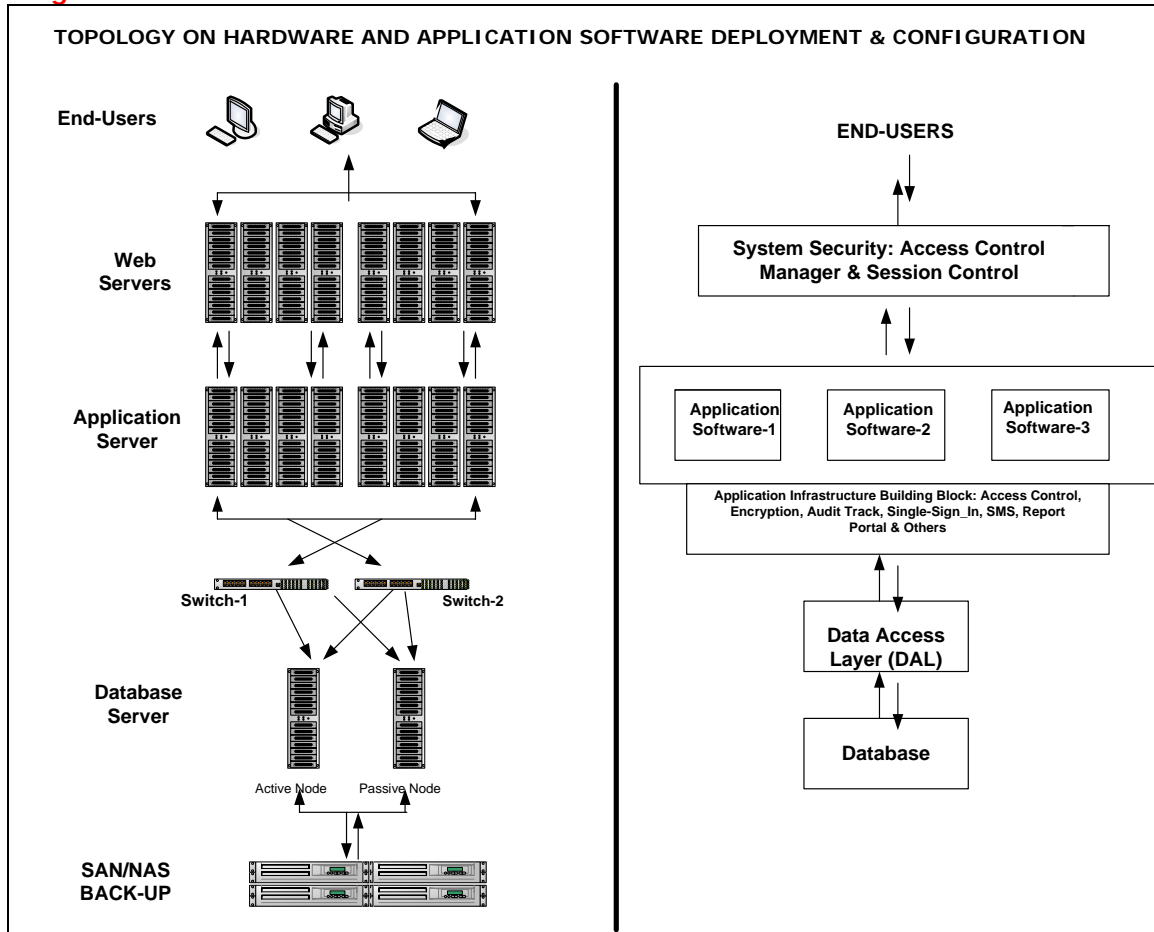
Diagram 3.2



3.3 OVERALL DIAGRAM OF COMBINED APPLICATION SOFTWARE AND HARDWARE

The following diagram (**Diagram 3.3's**) depicts the location based co-relationship between hardware and application software, when they are deployed together.

Diagram 3.3



[Open Spectrum Solution]

1. The above information is correct at the time of this article went to print and release on the OSS website. OSS reserves the absolute right to alter and change any of them at any given time without notifying the installed clienteles;
2. For most updated information on the said application, please contract your nearest OSS authorized resellers or logon to www.open-spec.com for contact; and
3. For reporting of error and mistakes at the above article's, please send your message to Documentation@open-spec.com.