Chapter 2: Literature Review

This chapter covers the study on existing systems available in the market. The purpose of this study is to obtain all the features that are necessary to be included in the new system. The study seeks to discover how the old systems perform their tasks, their strengths and weaknesses.

This chapter also reviews the latest technologies that have arrived recently in the field of Information Technology. It seeks to discover the concepts of the new technologies, and how these concepts came about, built upon earlier pioneering technology. Besides, it covers tools and methods that will be useful for the project development.

2.1 Analysis Studies

2.1.1 Overview of current Examinations Systems

An examination system at the secondary school level in Malaysia consists of four main stages. Each of these components represents a challenge in organization and implementation.

![Figure 2.1 Four Stages in the Examination System](image)
The first stage in an examination system is the registration of students for an examination. In a school system, the registration involves obtaining relevant particulars of each student, and entering the data in a form of Register. Some schools are still using a large book as a Register while others have implemented some form of computerized data entry system. Most rural schools are still using the manual system, as they do not have the computer skills needed to implement a computerized system. Even in most of the large town schools, such as Sekolah Menengah King George V, Seremban, they are using Microsoft Excel spreadsheets to handle their examinations processing. In SMCW, information is entered in Microsoft Excel files, one for each Form. There are problems in combining the results of each class to process results for each form, from Form One to Form Five. So there are plans to export them to an Access Database, which can handle some of the processing needs of registration.

The Lower Secondary School Students sit for subjects in a single package. The Upper Secondary Students choose from a few packages of subjects. All students must take some core subjects. Students need to know what elective subjects are available, what their scopes are, and the syllabus of each subject they are expected to take. Each class in each Form has predefined packages of subjects that each student in that particular class must take.

The second stage involves organizing the Examination Process. This includes Setting the Questions, Preparing the Marking Scheme, Executing the examination (students sitting for the examination), Marking the examination scripts, Issuing the examination results. Each of these steps can be subjected to varying degrees of computerization, including online tests where the students enter their answers in a computer.

The third stage is the recording of marks in marks sheets, processing the results, including calculating grades, averages and class positioning.

The final stage of the Examination System is the presentation of the results. This
can take the form of displaying the results on screen, printing out the results as mark sheets, generating result slips, analyzing the results for overall performance. It also includes selection of students for various incentives, such as prize giving.

2.1.2 Current Process Flow (Manual) or Existing System

The current manual system of school examination systems is that marks are entered on paper forms that are distributed to each teacher in a particular class. Some teachers would transfer these marks into a large master mark sheet. All the master mark sheets of each class are then bound in one huge master examination book. The process flow looks like this.

![Diagram of examination process]

Figure 2.2 Classic Examinations system

Some teachers would enter marks in individual electronic spreadsheets to aid in the process. Others would use database programs such as dBase IV and Microsoft Excel to enter the marks in their home computers. As such, they use different file formats and different file structures, using different types of programs. This makes it extremely difficult or impossible to combine all the data in the various files to form a master examination marks database.
In 1995, a computer program was introduced in SMCW for entering marks in the school’s computers. This was a DOS-based program written in the C programming language. The system was called REX. It uses a file-processing method of database access, manipulating files in dBase IV format. The menu system is shown in Figure 2.4.

The data entry screen is shown in Figure 2.5. Although these screens are reconstructed from the original DOS interface, they show the layout of the DOS type of menu handling and data entry of the REX system.

### 2.1.3 Similar Systems available in the Internet.

[www.skali.com.my](http://www.skali.com.my) is a new system implemented by the Ministry of Education that uses Web Pages for the registration of students for the Sijil Pelajaran Malaysia (SPM) examination. This system is designed using Active Server Pages (ASP) running in
The modules supported by this system include:

- **Registration of Students**
  
  This module enables registration of new students, and registration of repeating students.

- **Listing of registered Students**
  
  This module provides a list of students registered for the examinations.

- **Searching for Students**
  
  This is a search facility to look for particulars of students registered in the examination.

- **Placement of students in respective Examination Centres.**
  
  Once the students are registered, they must be placed in designation Examination
Centres. For school candidates, they sit for the examination in their own school. For private candidates, they are assigned to one of the schools.

- **Generation of Index Numbers**
  Each candidate has to be assigned an index number. This number is not auto-generated by the system. Rather the Exam Secretariat in the school assigns them manually.

- **Transfer of Students between Examination Centres**
  Candidates may transfer to other another school, or even to other states in the interim period between the first registration and the actual dates of the actual examination. The system has to cater for this transfer between examination centres.

- **Modifications to Student Information.**
  This module is for changing the particulars of the students. In addition, there could also be cancellation of candidates for a variety of reasons.

- **Marks Entry.**
  There are several modules to support entry of marks, such as the screen below shows. This module is meant for entry of Trial Examinations that are examined internally by each school.

- **Various Reports**
  The reports include Report of Registered Students, List of Transfer Candidates (that is candidates transferring from one school to another), List of Payment of Examination Fees, and List of Modifications to Examination Candidate Entries.

  Figure 2.6 shows a typical screen shot for the entry of marks for the subject 1103-Bahasa Melayu. It lists the candidates by their index number (Angka Giliran) and Identity Card Numbers (No. Kad Pengenalan). There is a textbox for each candidate for entering
the marks. However, this screen is difficult to use because the candidates' names are not listed.

Figure 2.6 Marks Entry in Skali's Web Site

This system functions to serve the needs of Registration for the public examinations. There is no function for the processing of examination results for the school's internal examinations, which need examination analysis and reports. The system does not make use of Web Services. There is no function that an external program could call to perform a similar task. Furthermore, there is no class registration system available on the Internet.

2.1.4 Web Services on the Internet

Case Study I – DS Data Systems for air fare quotes

DS Data Systems hosts Web Services to obtain live airfare quotes. Figure 2.7
shows the main screen which explain that the service searches the major airlines in real
time to find the best available prices direct from the airlines' web sites. The system then
presents a data entry screen as shown in Figure 2.8, which has entries for the origin and
destination routes, as well as the flight times. Here it is necessary to know the airport
codes. Here the Origin is entered as LHR, and the Destination as CDG.

![Figure 2.7 The Air Fare Quote Web Service Main Screen](image)

After clicking [Get Quotes], Figure 2.9 shows the screen as it searches real time
for the fares. Finally the result is shown in Figure 2.10, which indicates that Northwest
Airlines gives the best quote of £147.65, as compared with the next best quote from
Qantas of £462.10. It took about 50 seconds find the best quote, but the time varies for
other routes. This set of Web Services, however, is for testing and so do not have all the
full features expected of a user-friendly interface. The main drawback in the user-
interface is a lack of a drop-down list to select the airport codes.
Figure 2.8 Using the Air Fare Quote Flight Request Information

Search Book Go Air Fare Quotes

- **Origin**: LHR
- **Destination**: CDG
- **Departure Date**: 29/5/2003
- **Return Flight?**: Yes
- **Return Date**: 9/5/2003

Get Quotes

**Technical Details**

This JSP example demonstrates the use of the Search Book Go Air Fares web service.

Click here to view an example SOAP request for this web service.

Figure 2.9 Searching of the Air Fare Request

Searching

Please wait. The search process may take up to a minute or so because the Search Book Go service contacts airline websites directly in order to obtain up-to-the-minute fare information.
Northwest Airlines
£147.65
02/05 06:35
09/05 09:00

Qantas
£462.10
02/05 14:15
09/05 19:25

Figure 2.10 Results of the Air Fare Search

Case Study II – Financial Web Services

Financial Web Services offers Web Services that enables the user to calculate Bond Pricing, Bond Yields, Duration and Convexity, Annualized Volatility, and Zero Coupon Yields for Fixed Income calculations. It also offers Foreign Exchange Cross Rates and Swap Rates calculations. The Business News section also offers Derivatives, Fixed Income, Equity Markets, Stock Watch and Top Financial News.

Figure 2.11 shows the Zero Coupon Yields screen where there are text boxes to enter price and percentage for each bond for 2, 3 and 4 year bonds. After clicking the [Go] button, the screen refreshes, and the results are shown in the lower half of the screen. Figure 2.12 shows the Foreign Exchange Cross Rates Screen, where the cross rates of Canadian Dollars and Singapore Dollars are calculated.
The News and Features Web Service provides a listing of the top financial news in real time. Since this Web Service is in the United States, it gives the financial news that is relevant to that country, and the major world financial news.
Case Study III – GraphMagic

GraphMagic is a Web Service Site for generating graphs and charts. It needs to load in one of three types of clients: Perl, ASP.NET or PHP. The ASP.NET client requires the use of an authorization key and installation of Microsoft Visual Studio.NET. This Web Service Site enables the user to access their Web Service functions through a programming interface. Knowledge of programming and an understanding of their APIs are essential to use this interface. It is a highly rated site, but it requires a period of time to learn its methods in order to use it effectively.
Case Study IV – I-Cuisine

The site at http://icuisine.net/webservices/RecipeService.asmx has an easy to use Web Service which enables the user to enter a type of recipe, and select from a list of recipes. Figure 2.14 show the listing of chicken recipes after entering “Chicken” into the text box for “Search Foods”, and clicking [Go]. Selecting “Chicken Breast, oven-roasted, fat-free, sliced” will display a table showing the nutritional facts of the recipe, together with a pie chart showing their nutritional content. This site shows an effective use of charts for the display of information.
Figure 2.14 iCuisine Chicken Recipe Listing

Figure 2.15 iCuisine Chicken Breast Recipe
These case studies demonstrate the function and capabilities of Web Services, which show the type of components that will be developed in this project. The case studies cover Web Services which have mathematical and charting features.

Web Services need to be made available over the Internet. One of the ways is for the service to be advertised or listed in a Web Service Directory. An Internet search has resulted in several such directories. Examples are the IBM UDDI Business Registry (at https://uddi.ibm.com), as shown in Figure 2.16. Other sites include RemoteMethods (at http://www.remotemethods.com), and salcentral (at www.salcentral.com).

![IBMBusinessRegistry](image)

**Figure 2.16 IBM UDDI Business Registry**

2.2 Review on Latest Technologies

2.2.1 Client-Server Architecture

In the early history of distributed computing, huge mainframe computers did most of the processing of data and functionality, handling multiple users. Dumb terminals were
connected to access the resources on the mainframe. As desktop computers became cheaper and increased greatly in processing power, some of the processing was done on these intelligent terminals, later referred to as clients. The mainframe became the "Server".

This led to the Client-Server model of computing, that involves a central server handling the database functionality. The ‘fat-client’ handles the user interface, and some of the business logic, requesting data access from the server.

However, there are problems with this architecture, especially when maintenance of the client software involves replicating new versions over a widely distributed environment. With the advent of the World-Wide Web, ‘thin clients’ came into place, where the only software required is an Internet browser. The bulk of the processing is now distributed into a multi-tier architecture at the server level. This architecture is shown in Table 2.1.

<table>
<thead>
<tr>
<th>Thin Client</th>
<th>Web Server</th>
<th>Application Servers</th>
<th>Database Servers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Browser</td>
<td>Processes HTTP requests</td>
<td>Manages middle-tier functions</td>
<td>Databases</td>
</tr>
<tr>
<td>Fat Client</td>
<td></td>
<td></td>
<td>Legacy Mainframes</td>
</tr>
<tr>
<td>VB, Java, C++</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Presentation</td>
<td>Business Logic</td>
<td></td>
<td>Database</td>
</tr>
</tbody>
</table>

Table 2.1 Multi-tier architecture

2.2.2 Component Technology

Code reuse began with sharing common functions and subroutines. The problem was that these codes were not portable. When the object-oriented programming model was introduced, it brought about the concept of hiding the implementation, while exposing only the header. However, portability was still a problem.

Then component technology arrived, which uses interface-based programming. COM, DCOM and JavaBeans worked well in an Intranet environment. However, the
problem lies in distributing components across the Internet.

2.2.3 Web Services

A new programming paradigm was introduced when Web Services appeared. A Web Service can be defined as a self-describing, modular application that can be published, located and invoked from anywhere on the Web.

Web Services are the new generation of distributed computing, using loosely coupled services that communicate via standard Internet protocols such as HTTP and XML. The use of XML has given programmers a standard format for describing information in a structured and meaningful way. Clients can access and share data with a Web Service regardless of the application, platform, or language.

With Web Services, there is no need to install a component from a third party. Instead, a third party exposes a Web Service with the required functionality over the Internet. Then the company’s custom application accesses it, and uses the functionality to do its own processing.

2.2.3.1 The benefits of Web Services

Web Services provide for loosely coupled communication between enterprise systems. They are ideal for integrating disparate enterprise systems, particularly those systems that have evolved over time. Some of the benefits of Web Services are as follows:

- Platform independence – Any application, running in any operating system, or written in any language can send and receive SOAP messages.
- Ubiquitous communication channel – Web Services depend on the Internet to communicate and operate. The Internet is built on open, standardized protocols such as TCP/IP and HTTP.
• Enterprise interoperability – The growth of Business-to-Business (B2B) communications involve applications that involve merging or complementing existing hardware and software systems with trading partners and suppliers. Although Electronic Document Interchange (EDI) has been in use, implementing them has been costly, time-consuming and inflexible.

• Functional reuse – Developers can take advantage of functionality implemented by external partners by consuming the Web Services they expose. There is no need for rebuilding what has already been built.

• Extension of business – Web Services allows a business to extend its relationships and its reach to customers. By allowing third parties to access internal functionality through Web Services, businesses allow their customers to access them in a more integrated manner.

• Server neutrality – Web Services can be developed using any programming language and any server software.

• Secure communications – Web Services are as secure as any other web application. They use the authentication methods of HTTP.

2.3.3.2 When to use Web Services

Web Services should be used when we need interoperability across heterogeneous platforms. That is, when we need to expose all or part of our application to other applications on different platforms. Web Services are chosen when the application needs to send messages through a firewall. Web Services are most beneficial when the goals are achieving interoperability and integrating disparate software systems.

2.2.3.3 The Web Services core technologies include:

(a) HTTP (HyperText Transport Protocol) for transport of messages.

(b) XML (Extensible Markup Language) for a standard way of encoding data and messages across all Web Services.
(c) SOAP (Simple Object Access Protocol) enables accessing of Web Services using well-known calls and responses.

(d) WSDL (Web Service Description Language) specifies the interface of each method, of what parameters are accepted, and of what it returns.

(e) UDDI (Universal Description, Discovery and Integration) acts as a central repository of what Web Services are available.

- **Transport Mechanism**
  The HTTP protocol is a simple request/response mechanism and is synchronous. Calls to a service waits until the service returns. There is a HTTP/S protocol that provides an encrypted channel between the HTTP server and the client. Asynchronous calls, in which the caller does not wait for a response before proceeding to other processes, is provided by other channels, such as Java Messaging Service (JMS) and the Java API for XML (JAX-M).

- **Discovery using UDDI**
  UDDI allows for the registration of an entity as a service provider as well as the registration of specific services the company is willing to provide. It is the yellow pages of Web Services. Java provides the Java API for XML Registries (JAXR) to access a UDDI registry, query its contents, and make entries.

- **Description using WSDL**
  WSDL is the language used for specifying detailed information about a service, so that an application knows how to invoke the service with the required parameters. The WSDL document is a XML document and is complex and includes elements such as imports, types, schemas, messages, ports, bindings, and services.

- **Invocation using SOAP**
  SOAP allows a Web Service to be invoked by a client application irrespective of the language. The SOAP message consists of an envelope that contains a Header
and a Body.

![SOAP Message Diagram]

Figure 2.17 The SOAP Message

The Java API for XML RPC (JAX-RPC) implements a pure Java version of SOAP, and allows developers to execute Remote Procedure Calls (RPC).

- **Data/Document Processing using XML**
  XML documents consist of customized tags that are well formed.

- **DISCO (Discovery)**
  This is a Microsoft proprietary mechanism to locate and use a Web Service. An XML file with an extension of .disco is created when publishing a Web Service in Visual Studio .NET. While UDDI acts as a central repository to locate a Web Service, DISCO uses a browse and a find approach.
2.2.3.4 Web Services Architecture

The overall Web Service Architecture consists of a Web Service Server which makes a set of Web Services available over the Internet. A user discovers the existence of a Web Service through a Web Service Register, and obtains the methods, parameters and return values through the Web Services Definition Language (WSDL). The user, that is the client, then sends its request over the Internet using XML, specifically using SOAP over HTTP. When the request reaches the Web Service Provider, it then returns its message to the client, again as SOAP over HTTP. The overall picture is shown in Figure 2.18.

![Figure 2.18 Web Services Message Passing](image)

Not all Web Service Applications involve the whole overall architecture. Web Services can be developed based upon two possible architectures:

(a) Exposing Web Services
(b) Consuming Web Services
• **Exposing Web Services**

  This involves developing new Web Services or exposing current COM applications as Web Services. Microsoft has provided the SOAP Toolkit (currently version 2, but version 3 is now available), which provides an interface to expose the COM object as a Web Service.

  Developing new Web Services involve deploying a Web Service on a Web Server that can be accessed over an intranet or the Internet.

• **Consuming Web Services**

  Consuming Web Services could be as simple as just a single desktop application accessing the functionality exposed by a Web Service over the Internet. Web Service clients need not be client-based. It can be a Web Application, or even another Web Service.

  A more complicated way of consuming Web Services involves integrating various Web Services from various sources to complete a whole application, or using some Web Services exposed by business partners to be combined with an internal application. The architecture could look like the diagram in Figure 2.19. Web services brokers, such as IONA's Orbix E2A XMLBus Edition, are built to provide this type of service.

  In the .NET Framework, in order to access an XML Web Service, the following basic steps must be taken:

  • Locate the XML Web Service that gives the required functionality.
  • Create a proxy class for the XML Web Service by adding a Web reference to the project.
  • Reference the proxy class in the client code by including its namespace.
  • Create an instance of the XML Web Service proxy class in the client code.
  • Access the XML Web Service using the methods of the proxy.
2.2.3.5 Types of Web Services

The literature turned up two types of Web Services:

(a) XML Web Services

(b) COM+ Web Services

• XML Web Services

XML Web Services enables programs to access application logic using standard Web protocols, such as XML and HTTP. This is the most interoperable type of Web Services as the server and clients can run on different platforms, be it open
source or Microsoft-only. They can be implemented in Java or in the .NET Framework (as ASP.NET Web Services).

XML Web Services can be accessed by many types of clients, be it Windows Applications, Linux Applications and other Web Services, as long as they can send, receive, and process XML messages to and from the XML Web Service. However, XML Web Services are stateless. Each incoming request is handled independently.

- **COM+ Web Services**
  COM+ Web Services adds features that integrate with Microsoft .NET remoting to provide check box activation of XML Web Service publication via SOAP for COM+ components. .NET remoting supports a more tightly coupled, object-based programming model between client and server, which provides remote access to server-side objects with full data type fidelity. Clients can also obtain references to server-side objects and control the lifetime of those objects for stateful interaction. However, the client applications must be implemented using .NET remoting.

2.2.3.6 Web Service Communication Styles

There are two principal architectures for Web Service interfaces:

(a) Synchronous Web Services
(b) Asynchronous Web Services

- **Synchronous Web Services**
  The client invokes a service and then waits for a response to the request. Typically, the Web Service must respond within a short time. Examples are credit card approval and user authentication. It uses RPC calls.

- **Asynchronous Web Services**
  The client invokes the service but does not or cannot wait for the response. It may
take a significant amount of time for the service to process the request. The client can continue with some other processing rather than wait for the response. Later, when it does receive the response, it resumes whatever processing initiated the service request. This is a document-centric architecture.

A typical Web Service may mix and match these two architectures.

2.2.3.7 The future of Web Services

Web Services will lead to many new applications on the Web, and change the way existing web applications will operate. Among the new concepts emerging is the Programmable Web, where programmers can build entire systems by integrating the services and functionality exposed via Web Services.

In addition, Web Services will enable many applications to be developed for mobile devices. Instead of being confined to desktop applications, Web Services can be accessed using Web Applications, or even other Web Services.

With the .NET and the J2EE initiative establishing the standards and the core technologies as widely accepted specifications, the next step is to define a comprehensive architecture for additional technologies, such as security, process flow, reliable messaging, and transactions. This goal is being pursued by the W3C Web Services Architecture Working Group.

2.2.4 Platforms

2.2.4.1 SunONE, J2EE, Java and Open Source

In the Java environment, there is a large variety and mix of Java technologies to study and to choose. There are various players in the market delivering various toolkits for the development of Web Services. They all focus their implementations within the Java 2 Platform, Enterprise Edition (J2EE) initiative. Sun Microsystems has published its Open Network Environment architecture (SunONE) as a blueprint for the evolution of Web services. The Java Language is the only programming language available.
• Apache SOAP 2.2 – the most mature tool for Java developers. It does not support WSDL.

• Axis (Apache eXtensible Interaction System) – a vendor-neutral offering. It provides an implementation of JAX-RPC.

• International Business Systems offers the IBM WebSphere SDK for Web Services (WSDK). It comes with Axis, Xerces, a private UDDI registry, a test database, WebSphere web server. This was a replacement of the earlier IBM Web Services Toolkit (WSTK), which used Axis as a SOAP engine and Websphere as a web server as well.

• Sun Microsystems has recently provided the Java Web Services Developer Pack (WSDP). They all use various implementations of the Java 2 Enterprise Edition (J2EE) specifications. J2EE is the server side application model for Java applications. It uses Tomcat as the servlet engine instead of WebSphere. SunONE is an attempt to combine all the disparate Java technologies to assist developers in building Web Services.

The technologies delivered by Sun’s Java Web Services Developer Pack (WSDP) version 1.1 include:

• Java API for XML Binding (JAXB)
• Java API for XML Messaging (JAXM)
• Soap with Attachments API for Java (SAAJ)
• Java API for XML Processing (JAXP)
• Java API for XML Registries (JAXR)
• Java API for XML-based RPC (JAX-RPC)
• JavaServer Pages™ Standard Tag Library (JSTL)
• Tomcat (Java servlet and JavaServer Pages container)
• Ant build tool
• Registry Server
A typical usage of these technologies is shown in the Figure 2.20 below.

![Figure 2.20 Java Web Services technologies](image)

The open source methods involve making use of various independent software technologies that are not entirely integrated. This makes developing a Web Service and consuming a Web Service rather complicated and time consuming. Sun Microsystems has made the effort more integrated by providing the Java Web Services Developer Pack.

The core of Sun's WSDP is JAXP. These APIs enable working with XML documents in a variety of ways from Java programs. Two XML-based communication models complement JAXP's XML-processing capability: JAXM and JAX/RPC (Java APIs for XML-based Remote Procedure Calls). JAXM allows arranging message exchanges between Web Services, based on XML document formats. The message exchange can take place synchronously—causing a client program to wait for a reply message before proceeding—or asynchronously, meaning that a Web Service's client does not expect an immediate response. With JAX/RPC, an object's interface in Java is defined, providing an implementation of that interface, and then mapping the Java interface type to an XML definition. To deploy Web Services, the JWSDP bundles the latest Apache Tomcat release (4.0.3), with Apache Ant-based task templates to register and manage Web Services inside Tomcat's servlet containers. Sun has also prepared an excellent tutorial to give programmers a head start in using this technology.

IBM's Websphere SDK for Web Services V5.0 (WSDK) is another tool in this Java platform. It, however, uses the WebSphere Application Server. WSDK comes with
the following tools:

- Embedded version of WebSphere Application Server V5.0.
- Java 2 Standard Edition SDK 1.3.1
- Support for SOAP 1.1, WSDL 1.1, and UDDI version 2.
- WSDL4J, and UDDI4J.
- JAX-RPC (JSR-101) support.
- Support for the JSR-109 architecture for J2EE based Web Services.
- Tools to create, describe, deploy, access, publish, and discover Web Services.
- Security support as defined in the WS-Security specification.
- Support for Web Service implementation that follows the profiles and scenarios defined by the Web Services Interoperability Organization (WS-I).
- A private UDDI registry that is UDDI version 2 compliant.

IBM has also prepared a tutorial to use this WSDK, but it assumes a working knowledge of Java programming language and XML. Knowledge of J2EE and Ant are helpful but not required.

Figure 2.21 shows a typical architecture using a J2EE application server to host a set of Java Web Services. Web services can be integrated with application servers, such as WebLogic, WebSphere, and Orbix E2A J2EE Edition, to provide access to a variety of back-end systems, including .NET classes and COM objects, Enterprise JavaBeans, CORBA objects, MQSeries, MSMQ, and Java Messaging System queues.

2.2.4.2 .NET Framework and Microsoft Technology

This is an integrated environment that makes the chore of developing Web Services much easier. The .NET platform consists of the following parts:

- The .NET framework
- Visual Studio .NET, which creates .NET solutions using VB.NET, Managed C++, and C#
Figure 2.21 Web Services in J2EE application server implementation

- A server infrastructure that includes MSMQ (a message queuing similar to JMS or MQ Series), COM+ (a component model similar to EJBs), Active Directory (an LDAP implementation) and IIS (a web server).

- Building block XML Web Services called .NET My Services (Hailstorm) and includes a 'Passport service'

- A set of .NET servers such as Microsoft SQL Server 2000, Microsoft Biztalk server 2000.

Figure 2.22 The Microsoft .NET platform
The .NET Framework is a Microsoft technology that consists of five concepts:

- The Common Language Runtime (CLR) – hosts all .NET code and provides core services. It is a managed environment that executes all .NET programs and functions as the execution engine. This is similar to the Java Virtual Machine (VM). Code that runs within a CLR is referred to as ‘managed’ code.

- The Common Type System (CTS) and the Common Language Specification (CLS) – CTS fully defines all possible data types supported by the runtime. CLS is a subset of CTS that must be supported by all .NET languages.

- The Base Class Libraries – a comprehensive, object-oriented collection of reusable classes. These include ASP.NET, ADO.NET, web applications and Web Services classes.

- Microsoft Intermediate Language (MSIL) – all code written in .NET languages compiles to a platform-neutral code called MSIL. It is a binary intermediate form for a .NET program. This is similar to Java byte code.

- Assemblies – the fundamental unit of deployment, version control, reuse, activation and security.

The components of interest for developing Web Services include

- ASP.NET
- ADO.NET
- .NET Remoting
- ATL Server
- Web Forms
- Windows Forms
ASP.NET

ASP.NET is a set of components and services layered on top of the .NET Framework. There is an extensive class library that provides all of the base functionality for working with SOAP messages and processing WSDL files. The environment makes it very convenient to set up and deploy Web Service applications.

.NET Remoting

.NET represents the next generation of distributed component technologies. It supercedes DCOM. It enables .NET components to be exposed and utilized from a client machine as if the component was local. It provides a richer interface than ASP.NET Web Services. Web Services enable only single method calls. However, .NET Remoting is a Microsoft-only implementation.

ATL Server

ATL Server is a tool designed to allow C++ developers to expose and to consume
Web Services. It allows developers to build a Web Service in C++ as an ISAPI filter, giving the developer maximum control over performance and scalability. However it is a very complicated technology, requiring very experienced C++ programming skills.

2.2.4.3 Comparing J2EE and .NET Platform

.NET shines in its tool and language support. J2EE is a winner in industry maturity and interoperability support. Both platforms are good in legacy and standards support. J2EE offers only Java as a language of development, and consists of many disparate technologies. However, .NET has made a first attempt at writing Web Services easier to begin. So this project begins by writing Web Services using the .NET Platform.

J2EE in this context is merely a specification, with multiple product implementations. Among the various implementations are the following:

- JBuilder from Borland
- IBM Web Application Server
- Oracle 9i Application Server

J2EE requires an application server for its server implementation of the J2EE specification. The J2EE concept views Java classes and beans as building blocks, with Web services as a type of interface into and out of them. J2EE supports Web Services loosely as additional Java APIs. J2EE vendors essentially view Web services as another type of application server client. On the other hand, Web Services can be designed as building blocks in the .NET Framework, which just uses the Windows operating system for many of its services.

J2EE and .NET are both based on the concept of a 3-tier architecture. The aim is that the construction and delivery of the user interface is separate from the construction of the business logic, which in turn is separate from the back-end data or infrastructure being accessed.

The conceptual Architectural comparison between the J2EE platform and .NET
platform can be divided into Presentation and Access, Business Logic, Connectivity and Runtime, as shown in Figure 2.24 and 2.25.

![Diagram of the Conceptual Java Architecture](image)

Figure 2.24 The Conceptual Java Architecture

![Diagram of the Conceptual .NET Architecture](image)

Figure 2.25 The Conceptual .NET Architecture

### 2.2.5 Web Server

The following Web Servers were reviewed:

#### 2.2.5.1 Internet Information Service (I IS)

This is a Microsoft-specific Web server and runs only on Windows platforms. It is
an enterprise-level Web Server that is included with Windows 2000. This is the server of choice when selecting the .NET platform. In order to use IIS as a Web Server for development, it is necessary to have VS Developer rights. Also, to debug applications, it is necessary to have Debugger rights. It supports the Windows GUI and provides integration with other Windows mechanisms, such as Active Directory, MS SQL Server and Windows Security services. IIS also supports the benefits of COM+. Windows XP Home Edition does not come with IIS. See Appendix D on how to overcome this problem.

2.2.5.2 Microsoft Personal Web Server (PWS)

PWS is a scaled-down version of IIS for personal computers. It is ideal for educational institutions, small businesses and individuals who only have access to Windows 98.

2.2.5.3 Apache Web Server

The Apache Software Foundation maintains Apache Web Server. It is currently the most popular Web server because it is stable, reliable, efficient and portable. It is an open source product and runs on UNIX, Linux and Windows platforms. The source code for Apache is freely available. This has spurred the development of Apache functions from many volunteer programmers (and recently from IBM). Apache is modular in structure, so users can pick and choose modules to fit their requirements. However, there is no GUI available yet from Apache to manage this server.

2.2.5.4 TomCat

This is an open source Web Server developed by the Jakarta Project. It is a servlet engine that serves as the official Reference Implementation of the Java Servlet and Java Server Pages technologies. Tomcat is developed in an open and participatory environment and released under the Apache Software License.

2.2.5.5 Other servlet engines encountered in the literature include

(a) Sun Application Server 7
(b) Oracle 9i Application Server
(c) IBM Websphere Application Server
(d) iPlanet from Netscape

2.2.6 Operating System

The following Operating Systems were reviewed

2.2.6.1 Microsoft Windows

This is a set of commercial operating systems offered by Microsoft Corporation, and the latest products include Windows 2000 and Windows XP. Windows 2000 is a menu-driven operating system that uses a graphical user interface (GUI) as its main method of communication with the user. Its user interface is very user friendly. It is the most popular operating system.

2.2.6.2 UNIX

UNIX has three major advantages:

- It is portable from large systems to small systems
- It has powerful utilities
- It is device independent

The UNIX operating system is written using the C programming language. However it has some distinct disadvantages:

- There is no single standardized version of the software.
- Its commands are so terse that it is not user-friendly to novices

2.2.6.3 Linux

This is a free, open source operating system that is gaining widespread acceptance in the industry. Its commands are similar to the UNIX operating system. This operating system is more scalable than the Windows operating system. It consists of a Linux kernel that is totally different from UNIX. The Linux system includes a multitude of components, borrowed from other development projects.
2.2.7 Database Server

A "database engine" is the core process that a database management system, such as Access or SQL Server, uses to store and maintain data. It serves two functions. The first is to store information, and the second is to process requests for stored information. It does not have a graphical user interface. The user interface is generally by applications, or by visual compilers. The following Database Servers were reviewed:

2.2.7.1 Microsoft Access and Microsoft Jet

Jet 4.0 is the default database engine used in Microsoft Access 2000, a desktop database product that is available as a part of Microsoft Office in its latest releases. Jet is a file-server database management system, and is scaled for desktop and shared database solutions. All the database information is contained in a single .MDB file. Although Jet technically can support up to 255 users, in real world applications, it is safer to support up to 10 users. While Jet is limited to a 2-GB file size, we can link multiple files together to extend the size limit accordingly. Jet-based solutions are a good choice for desktop or shared applications with their ease of use and low cost. However, as the system grows, many code changes are required before the Jet applications can be migrated to the performance and scalability of SQL Server.

File-based databases limit performance of applications because database processing occurs on the client machine. When multiple users are using a file-based database, each must open and read information from the same file. This can create a bottleneck, as each application user waits for their turn to read from, or write to, the database file.

However, Jet databases are easier to backup and distribute because the database is contained in a single file.

2.2.7.2 Microsoft SQL Server 2000 Desktop Engine (MSDE)

Microsoft SQL Server 2000 Desktop Engine is a data store based on Microsoft
SQL Server technology, but it is designed and optimized for use on smaller computer systems, such as a single user computer or small workgroup server. Most workflow projects or client/server applications run unchanged on either the desktop engine or SQL Server.

MSDE enables developers to build desktop and shared database solutions that easily migrate to SQL Server when the solution must scale. Code written in MSDE will also run without changes in SQL Server. However, unlike SQL Server, the desktop engine has a two-gigabyte database size limit, does not support symmetrical multiprocessing (SMP), and cannot be a publisher for a transactional publication. MSDE, like SQL Server, runs as a client-server service, much like a Web server. MSDE runs best in an environment of 5 or fewer concurrent users. However, Jet will probably support more concurrent users than MSDE. Database size for MSDE is limited to 2 gigabytes. MSDE, like SQL Server, supports more industrial-strength database features such as transaction-logging, stored procedures, triggers, distributed transactions, dynamic backup and restore, and automatic recovery. MSDE provides a more seamless upsize path to SQL Server. In MSDE, relationships are called database diagrams. In addition, there are new database objects, such as views, stored procedures, and data access pages.

MSDE operates as a windows service. Windows services are applications that run as background processes. They are started when the operating system starts, and ended when the operating system ends.

To create a MSDE database in Microsoft Access, select New, Project. This creates an .adp file. This file is linked to the .mdf file created by MSDE.

2.2.7.3 Microsoft SQL Server

SQL Server is a client-server database management system. Unlike file-based databases, client-server database engines manage read-write operations to the database. Because of this, client-server databases such as SQL Server can handle many more concurrent users and vastly greater amounts of data. MSDE and SQL Server both conform to Transact-SQL (T-SQL) language guidelines. Jet does not.
SQL Server and the MSDE are server-based database systems. Requests for information are processed on the server and only the resultant data is sent back to the client application. This scenario greatly reduces network traffic; it also allows users to access data and experience acceptable application performance without a high-end client workstation.

SQL Server runs only on the Windows platform, which severely restricts its portability. SQL Server 2000 comes in seven editions: Enterprise, Standard, Personal, Developer, Desktop Engine, and SQL Server CE. The first two editions are the choices if a developer is building a database or analysis application that accesses SQL Server 2000. They can only be installed on server operating systems, namely Microsoft Windows 2000 Server, Windows 2000 Advanced Server, Windows 2000 Datacenter Server, and Windows NT Server 4.0. Between these two editions, the features are as follows:

- **SQL Server 2000 Enterprise Edition**
  It includes the complete set of SQL Server database and analysis features. It supports the largest Web sites and enterprise online transaction processing (OLTP) and data warehousing systems. It supports up to 64 gigabytes of memory and up to 32 processors. It can divide database workload across independent servers. It supports System Area Networking (SAN) connection technologies between servers. For availability, this edition includes features for mission critical applications such as failover clustering to ensure that applications stay up and running even when disaster strikes. In terms of performance, this edition speed up an application by taking full advantage of symmetric multiprocessor (SMP) computers, faster queries and transaction applications.

- **SQL Server 2000 Standard Edition**
  This is a more affordable option for small-sized and medium-sized organizations. Standard Edition can be used on symmetric multiprocessing systems with up to 4 CPUs and 2 GB of RAM. It includes the core functionality needed for non-
mission-critical e-commerce, data warehousing, and line-of-business solutions. It has all the data mining features and the core OLAP functionality. Apart from these, it also includes Data Transformation Services (DTS), Replication, Full-Text Search, English Query, Stored procedure development and debugging tools, SQL Profiling and performance analysis tools.

- **SQL Server 2000 Personal Edition**
  SQL Server 2000 Personal Edition is ideal for mobile users who spend some of their time disconnected from the network but run applications that require SQL Server data storage, and for stand-alone applications that require local SQL Server data storage on a client computer.

- **SQL Server 2000 Developer Edition**
  This edition allows developers to build any type of application on top of SQL Server. It includes all of the functionality of Enterprise Edition but with a special development and test end-user license agreement (EULA) that prohibits production deployment.

- **Microsoft SQL Server 2000 Windows CE Edition**
  Microsoft SQL Server 2000 Windows CE Edition (SQL Server CE) is the compact database for rapidly developing applications that extend enterprise data management capabilities to devices.

- **SQL Server 2000 Desktop Engine**
  SQL Server 2000 Desktop Engine is the successor to Microsoft Data Engine (MSDE) 1.0, which was based on SQL Server version 7.0. As such, SQL Server 2000 Desktop Engine is often referred to as MSDE 2000. The most significant characteristic of the Desktop Engine is that it is a redistributable version of the SQL Server relational database engine. The Desktop Engine has the smallest footprint of any edition of SQL Server 2000. The SQL Server 2000 Desktop
Engine does not include graphical management tools; the application distributing the engine is usually coded to perform any needed database administration.

2.2.7.4 IBM DB2

This is a desktop version of IBM's mainframe database product running on OS/390. It can be programmed using the Call Language Interface (CLI). In terms of performance, this is the top DBMS product. There are more administrative products on the market for DB2 on OS/390 than for the others, making it easier to manage. However, there is little difference on the desktop versions.

2.2.7.5 MySQL

MySQL is a multi-user, multithreaded RDBMS server that uses SQL to interact with and manipulate data. This is an open source relational database product. It can be downloaded for free at www.mysql.com. It is a high performance and scalable DBMS. It has an in-memory query results cache that contributes to its performance. Most agree, however, that it works best when managing content and not executing transactions. Its important features are:

• Multithreaded capability enables the database to perform multiple tasks concurrently.
• Support for various programming languages: C, C++, Java, Python, Perl, PHP.
• It runs on just about any operating system. Specifically, it runs on Windows, Linux and UNIX
• The ability to handle large databases.
• The ability to access tables from different databases by using a single query.

2.2.7.6 Oracle 9i Database

This is a very reliable database product from Oracle Corporation, which is the current industry leader in database technology. There are more programmers who can program Oracle DBMS than any other. Oracle 9i Database comes in three editions: Enterprise, Standard and Personal. Using this database requires a lot of configurations,
which are beyond the scope of this dissertation.

2.2.8 Database Access Technology

Databases could be accessed using the following technologies:

2.2.8.1 Open Database Connectivity (ODBC)

A consortium of companies designed ODBC, including Microsoft, Lotus, Oracle and IBM. It was one of the first attempts to create a generic way of talking to different database engines. However, it was built to access relational data from relational databases. It is accessed via a straight-C API.

2.2.8.2 Microsoft JET and Data Access Objects (DAO)

Jet and DAO were built to support file-based databases, and uses ODBC to access relational databases such as Access. Thus it tends to be slow. It was built for small databases used in local deployments and single-system applications.

2.2.8.3 Remote Data Objects (RDO)

RDO was created to allow faster access, via ODBC, to relational database engines, such as SQL Server and Oracle. It provides the objects, properties, and methods needed to access stored procedures and complex resultsets.

2.2.8.4 OLE DB

OLE DB was designed to access virtually any kind of data, whether it is stored relationally in tables, directories or free form. As a result, it is very large and complicated.

2.2.8.5 ActiveX Data Objects (ADO)

ADO is a wrapper to OLE DB, but is focused on relational data. It is fairly fast,
simple to use and supports multi-tier architectures via the use of disconnected Recordsets. Microsoft ActiveX Data Objects Multidimensional (ADO MD) was an extension to include objects specific to multidimensional data from languages such as Microsoft Visual Basic and Microsoft Visual C++. Microsoft ActiveX Data Objects Extensions for Data Definition Language and Security (ADOX) includes objects for schema creation and modification, as well as security.

2.2.8.6 Remote Data Service (RDS)

RDS enables data to be moved from a server to a client application or Web page, manipulate the data on the client, and return updates to the server in a single round trip.

2.2.8.7 Microsoft Data Access Components (MDAC)

MDAC is not an access mechanism; rather it includes a single version of ODBC, OLE DB, and ADO, together with drivers for SQL Server, Access and Oracle.

2.2.8.8 ADO.NET

ADO.NET is the database access method of the .NET platform. It is multi-tier (code could be easily moved from tier to tier), disconnected, XML-based, scalable and fast.

There is no Recordset. Instead, data is manipulated using DataSets. There are four classes that are used to read and write data from data sources:

1. Connection - Connect to data source
2. Command - Execute stored procedures
3. DataAdapter - Connects DataSet to database
4. DataReader - Forward/only, read/only cursor

2.2.9 Programming Language

The following languages were reviewed:
2.2.9.1 C#

This is the language of choice for the .NET platform, as it is closely tied to the .NET CLR. It is an object-oriented language that closely resembles Java and C++. C# was designed in response to the strengths and weaknesses of other languages, in particular Java and C++. There are a few new syntax elements in C# that are not available in C++ or Java, namely properties, indexers and attributes. Properties are syntactic support for get() and set() methods. Indexers are a way to treat an object as an array. Attributes support declarative programming.

Key features of C# are:

- **Component orientation.** This makes component building easy, with language constructs such as properties, events and attributes.
- **Automatic and manual memory management.** The programmer does not have to dispose objects, eliminating memory leaks and circular references.

2.2.9.2 Java

Java is an object-oriented language that is well suited to designing software for use over the Internet. It was developed at Sun Microsystems in 1991. It is a cross-platform language that can be used in many platforms, including Windows, Macintosh, Linux and most versions of UNIX, including Solaris. The language is quite complex and includes many disparate open source technologies. It is often run as an applet in Internet browsers, which can also act as a SOAP Client.

2.2.9.3 C++

Programming Web Services in this language offers very high performance, but it requires very high programming skills. In the .NET platform, it is used to program .NET remoting, which performs better than XML Web Services. This is the language used in developing high performance ATL Server Web Services. The ATL Server uses C++ code that is compiled at development time before being deployed on the server. The ATL Server architecture runs on top of Internet Information Server (I I S).
2.2.10 Authoring Tools

Various authoring environments were reviewed:

2.2.10.1 Microsoft Visual Studio.Net

This is a commercial product that abstracts away many of the technologies used in developing Web Services. The tools for developing Web Services are well integrated and provide a fast entry into developing Web Applications and Web Services. The Integrated Development Environment provides excellent debugging tools.

2.2.10.2 .NET Framework Software Development Kit

This software development kit is freely downloadable from Microsoft's Web Site. It includes tools for programming Web Services in C#. However there is no IDE, and programmers have to use NotePad to write the source and compile using a command-line interface.

2.2.10.3 SunONE Studio

SunONE Studio 4 is an integrated development environment designed to create, test, and deploy new Web applications, servlets, and JSPs from JSP templates. It can be used to develop JavaServer Pages (JSP) web applications using prepackaged tag libraries, to meet a client's domain-specific needs. It is designed to develop dynamic Java Web Applications using one complete toolset. It includes templates, GUI layout wizard, database development support, a Web server, a Web browser, and specialty browsers. It includes the PointBase Server, which is a full featured 100% Java relational database.