Chapter 15

Database Connectivity and Web Technologies

Learning Objectives

- In this chapter, you will learn:
  - About database connectivity fundamentals
  - About various database connectivity technologies
  - How web-to-database middleware is used to integrate databases with the Internet
  - What services are provided by web application servers
  - What Extensible Markup Language (XML) is and why it is important for Web database development
  - About cloud computing and how it enables the database-as-a-service model

Database Connectivity

- **Database middleware**: Provides an interface between the application program and the database
- Data repository - Data management application used to store data generated by an application program
- **Universal Data Access (UDA)**: Collection of technologies used to access any type of data source and manage the data through a common interface
  - ODBC, OLE-DB, ADO.NET form the backbone of MS UDA architecture

Native SQL Connectivity

- Connection interface provided by database vendors, which is unique to each vendor
- Interfaces are optimized for particular vendor's DBMS
  - Maintenance is a burden for the programmer

ODBC, DAO, and RDO (1 of 3)

- **Open Database Connectivity (ODBC)**:Microsoft's implementation of a superset of SQL Access Group
- **Call Level Interface (CLI)** standard for database access
  - Widely supported database connectivity interface
  - Allows Windows application to access relational data sources by using SQL via standard **application programming interface (API)**

ODBC, DAO, and RDO (2 of 3)

- **Data Access Objects (DAO)**: Object-oriented API used to access desktop databases such as MS Access and FileMaker Pro
  - Provides an optimized interface that expose functionality of Jet data engine to programmers
  - DAO interface can be used to access other relational style data sources

ODBC, DAO, and RDO (3 of 3)

- **Remote Data Objects (RDO)**
  - Higher-level object-oriented application interface used to access remote database servers
- **Dynamic-link libraries (DLLs)**
  - Implements ODBC, DAO, and RDO as shared code that is dynamically linked to the Windows operating environment

Components of ODBC Architecture

- High-level ODBC API through which application programs access ODBC functionality
- Driver manager that is in charge of managing all database connections
- ODBC driver that communicates directly to DBMS

Figure 15.2 - Using ODBC, DAO, and RDO to Access Databases

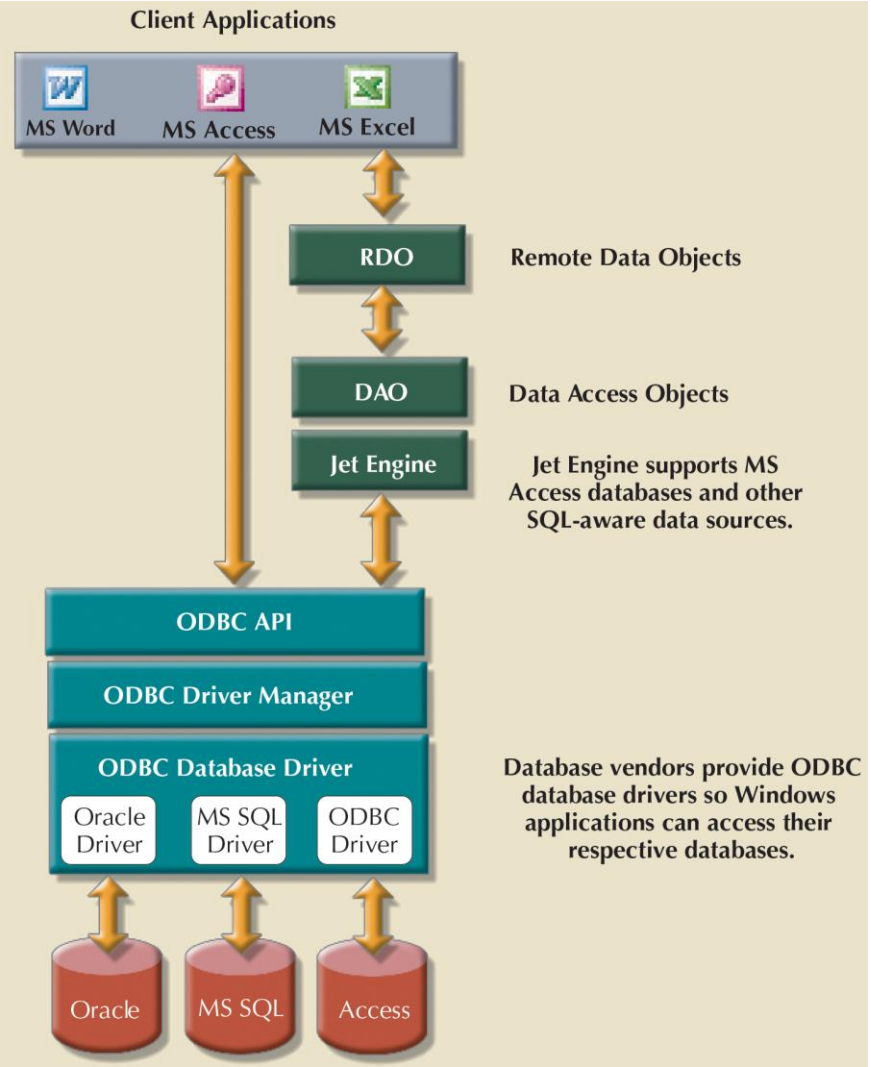
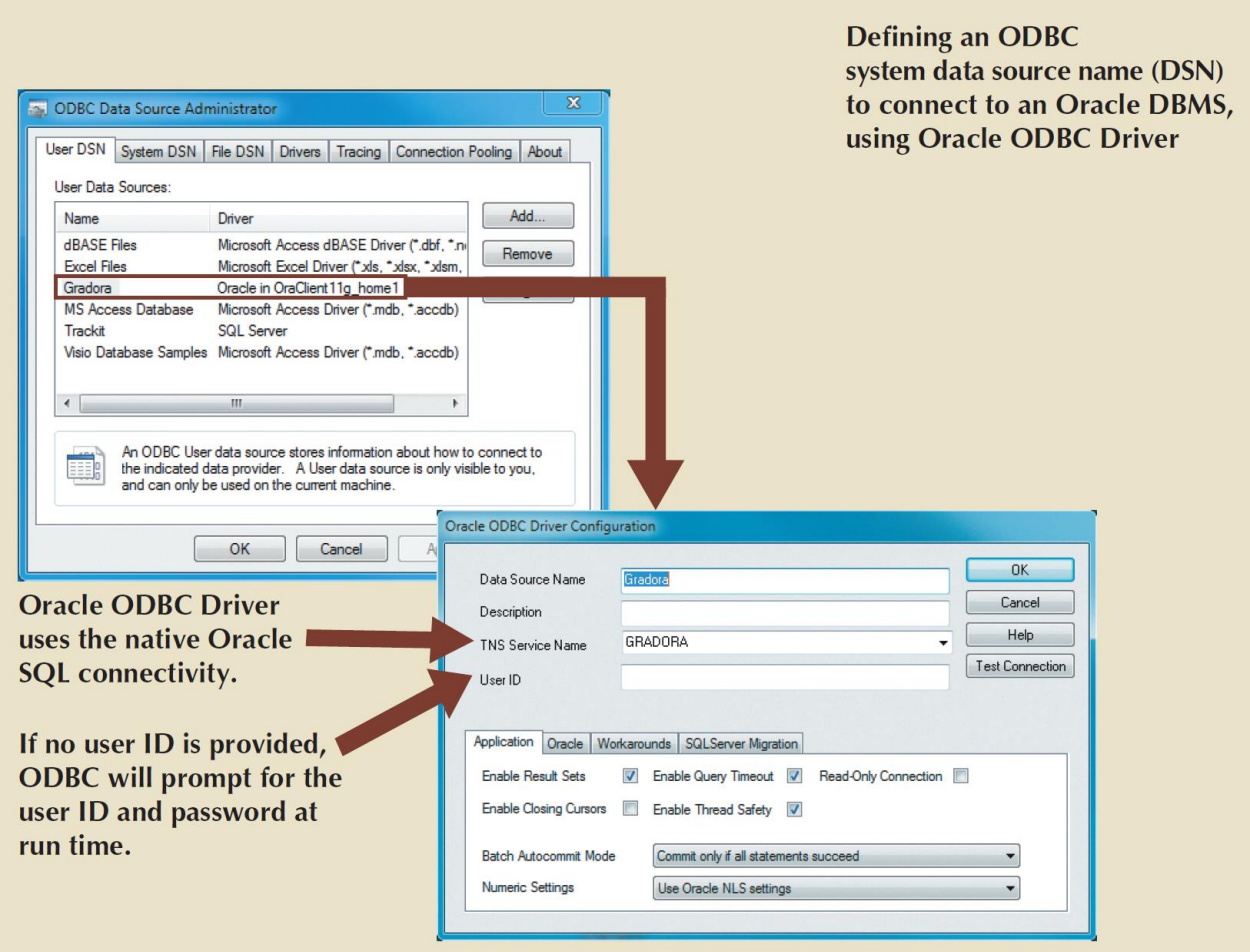


Figure 15.3 - Configuring an Oracle Data Source

FIGURE 15.3 CONFIGURING AN ORACLE ODBC DATA SOURCE



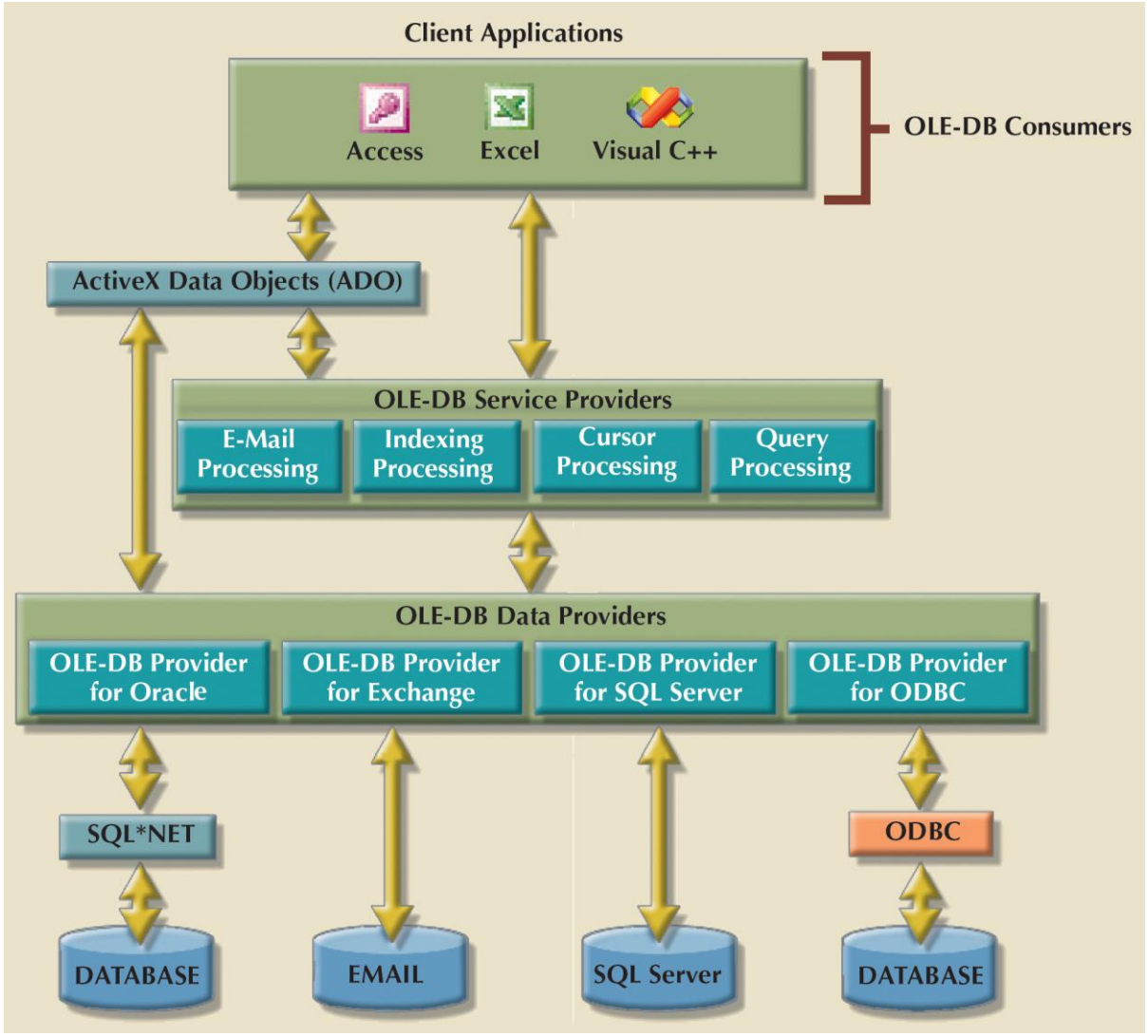
Object Linking and Embedding for Database (OLE-DB) (1 of 2)

- Database middleware that adds object-oriented functionality for access to data
- Series of COM objects provides low-level database connectivity for applications
- Types of objects based on functionality
  - Consumers (applications or processes)
  - Providers (data or service)

Object Linking and Embedding for Database (OLE-DB) (2 of 2)

- Does not provide support for scripting languages
- **ActiveX Data Objects (ADO)** provides:
  - High-level application-oriented interface to interact with OLE-DB, DAO, and RDO
  - Unified interface to access data from any programming language that uses the underlying OLE-DB objects

Figure 15.5 – OLE-DB Architecture



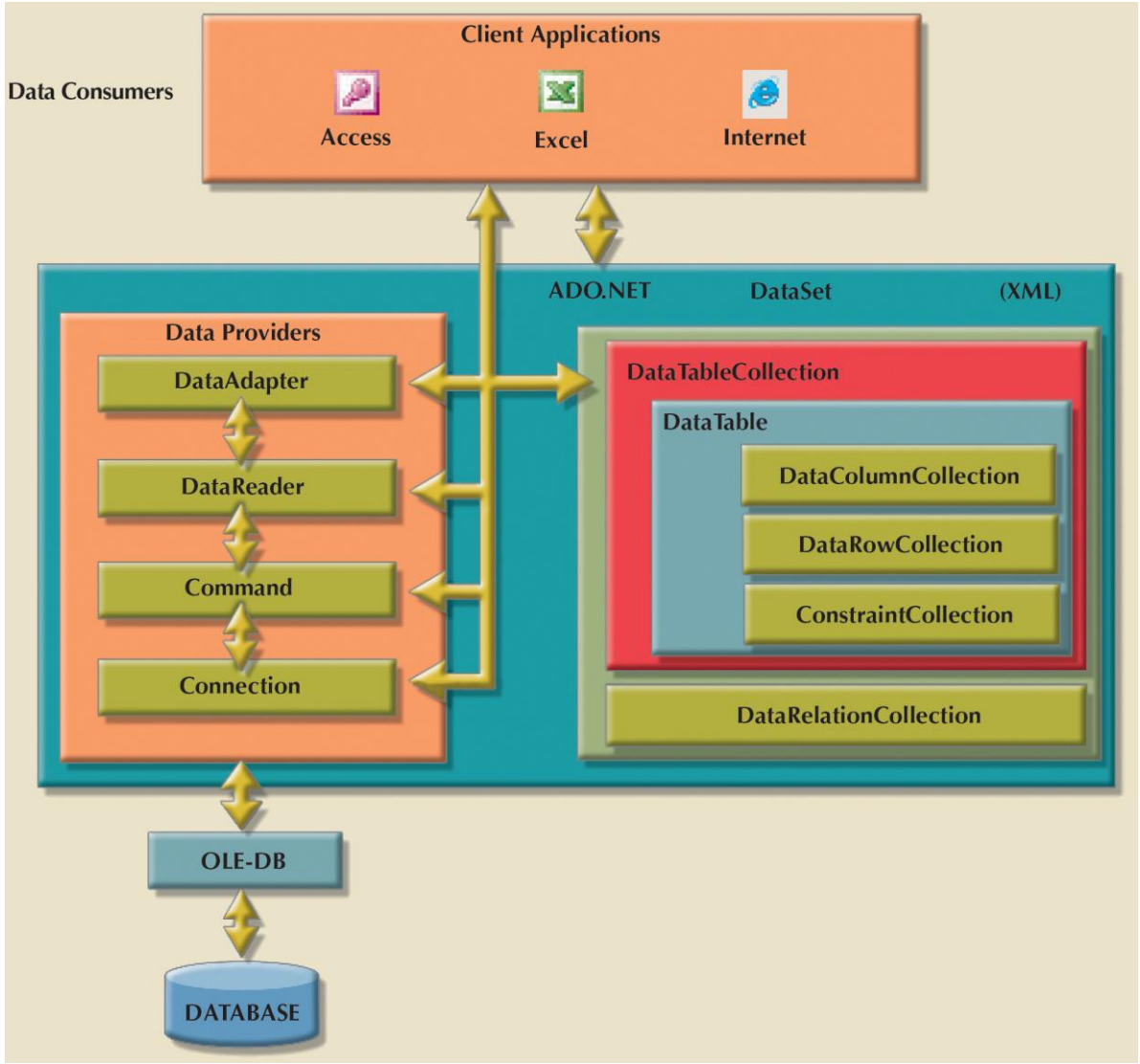
ADO.NET (1 of 2)

- Data access component of Microsoft's .NET application development framework
- **Microsoft's .NET framework**
  - Component-based platform for developing distributed, heterogeneous, interoperable applications
  - Manipulates any type of data using any combination of network, operating system, and programming language
  - Extends and enhances functionality critical for the development of distributed applications

ADO.NET (2 of 2)

- **DataSet**: Disconnected memory-resident representation of the database
  - Contains tables, columns, rows, relationships and constraints
  - Internally stored in XML format
  - Data in DataSet is made persistent as XML documents

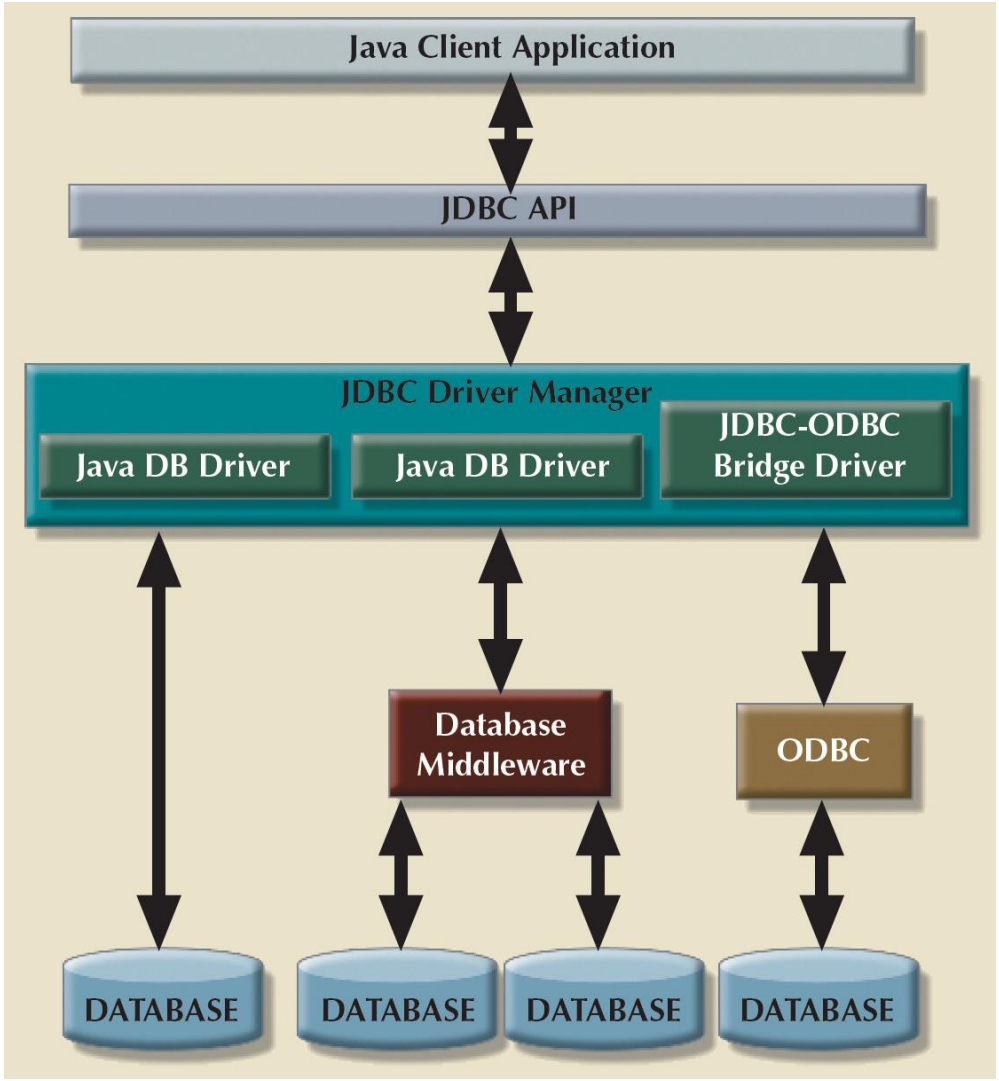
Figure 15.6 – ADO.NET Framework



Java Database Connectivity (JDBC)

- Application programming interface that allows a Java program to interact with a wide range of data sources
- Advantages of JDBC:
  - Company can leverage existing technology and personnel training
  - Allows direct access to database server or access via database middleware
  - Allows programmers to use their SQL skills to manipulate the data in the company's databases
  - Provides a way to connect to databases through an ODBC driver

Figure 15.7 – JDBC Architecture



Database Internet Connectivity

- Allows new innovative services that:
  - Permit rapid response by bringing new services and products to market quickly
  - Increase customer satisfaction through creation of web-based support services
  - Allow anywhere, anytime data access using mobile smart devices via the Internet
  - Yield fast and effective information dissemination through universal access

Table 15.3 - Characteristics and Benefits of Internet Technologies (1 of 2)

INTERNET CHARACTERISTIC	BENEFIT
Hardware and software independence	Savings in equipment and software acquisition Ability to run on most existing equipment Platform independence and portability No need for multiple platform development
Common and simple user interface	Reduced training time and cost Reduced end-user support cost No need for multiple platform development
Location independence	Global access through Internet infrastructure and mobile smart devices Creation of new location-aware services Reduced requirements (and costs!) for dedicated connections

Table 15.3 - Characteristics and Benefits of Internet Technologies (2 of 2)

INTERNET CHARACTERISTIC	BENEFIT
Rapid development at manageable costs	Availability of multiple development tools Plug-and-play development tools (open standards) More interactive development Reduced development times Relatively inexpensive tools Free client access tools (web browsers) Low entry costs; frequent availability of free web servers Reduced costs of maintaining private networks Distributed processing and scalability using multiple servers

Web-to-Database Middleware

- Web server is the main hub through which Internet services are accessed
- **Server-side extension**: Program that interacts directly with the web server
  - Provides its services to the web server in a way that is totally transparent to the client browser
  - Known as **web-to-database middleware**

Figure 15.8 - Web-to-Database Middleware

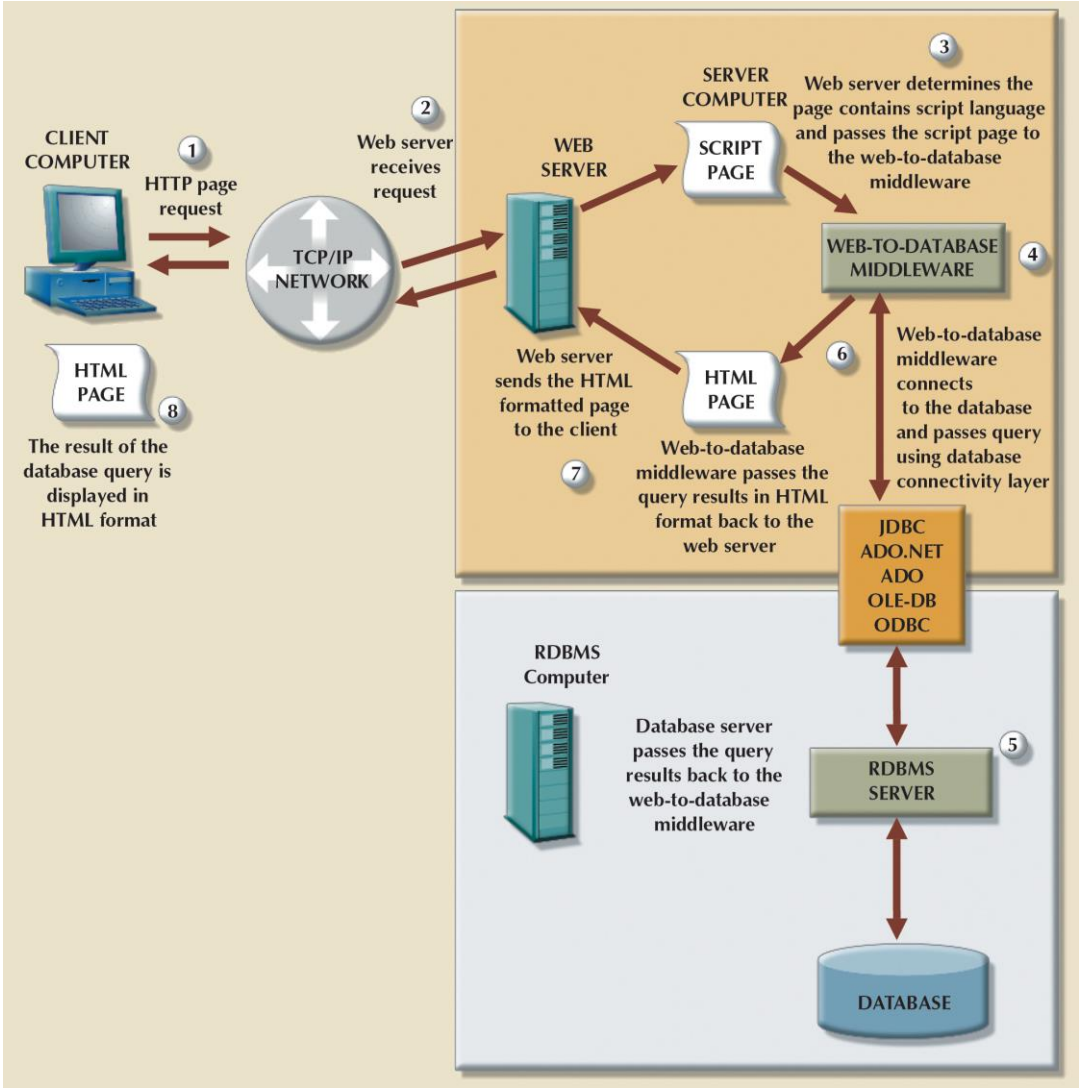
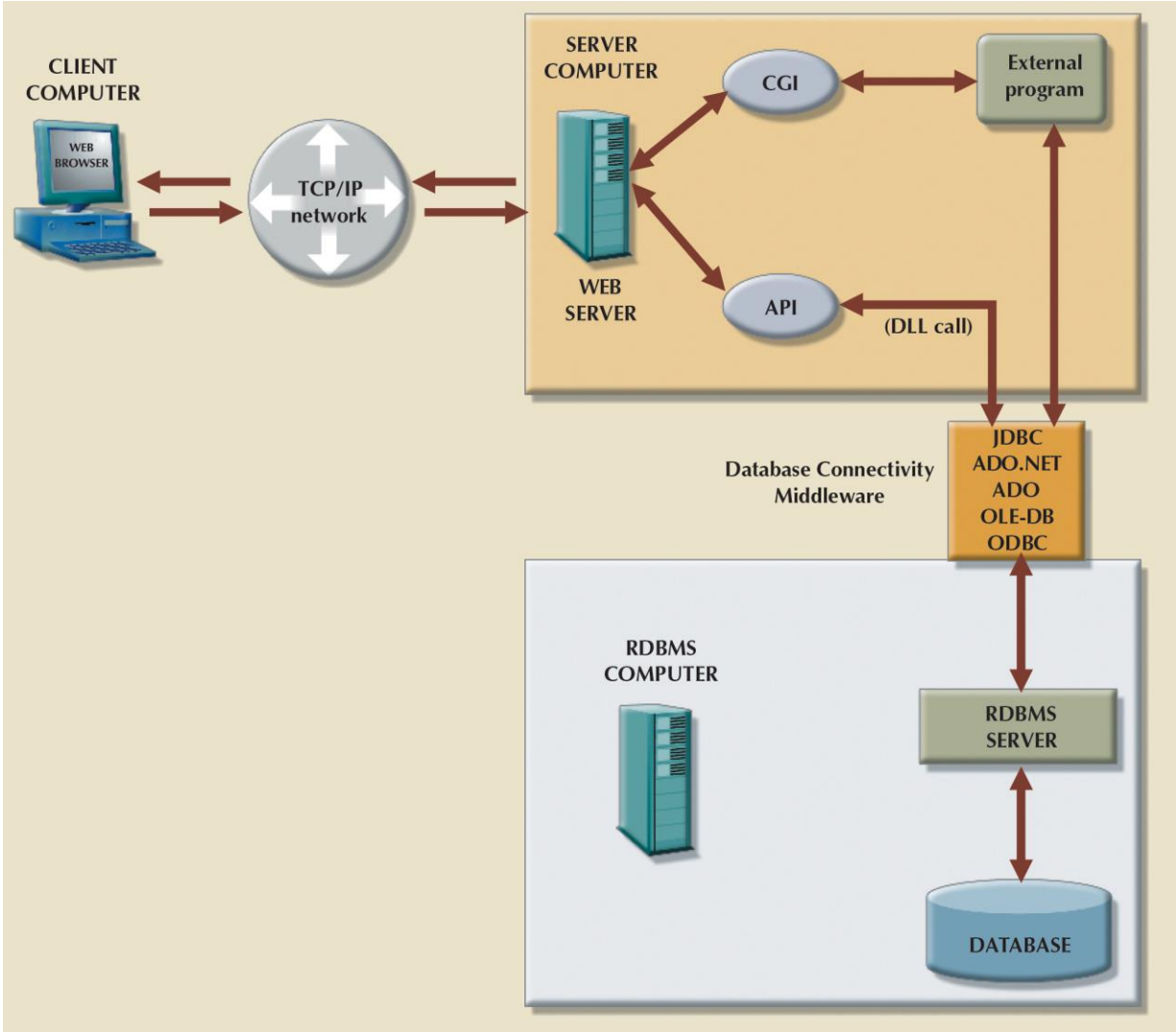


Figure 15.9 - Web Server CGI and API Interfaces



The Web Browser

- Software that lets users navigate the web from their client computer
- Interprets HTML code received from web server
- Presents different page components in standard way
- Web is a stateless system
  - **Stateless system**: Web server does not know the status of any clients

Client-Side Extensions

- Add functionality to Web browser
- Types
  - **Plug-in**: External application automatically invoked by the browser when needed
  - **Java and JavaScript**: Embedded in web page
    - Downloaded with the Web page and activated by an event
  - **ActiveX and VBScript**: Embedded in web page
    - Downloaded with page and activated by event
    - Oriented to Windows applications

Web Application Servers

- Middleware application that expands the functionality of web servers by linking them to a wide range of services
- Used to:
  - Connect to and query database from web page
  - Present database data in a webpage using various formats
  - Create dynamic web search pages
  - Create webpages to insert, update and delete data
  - Enforce referential integrity
  - Use simple and nested queries and program logic to represent business rules

Web Application Server Features

- Integrated development environment
- Security and user authentication
- Computational languages
- Automation generation of HTML pages
- Performance and fault -tolerant features
- Database access with transaction management capabilities
- Access to multiple services

Web Database Development

- Process of interfacing databases with the web browser
- Code examples
  - ColdFusion
  - PHP

Extensible Markup Language (XML)

- Represents and manipulates data elements
- Facilitates the exchange of structured documents over the Web
- Characteristics:
  - Allows definition of new tags
  - Case sensitive
  - Must be well-formed and properly nested
  - Comments indicated with <- and ->
  - XML and xml prefixes reserved for XML tags only

Document Type Definitions (DTD)

- File with .dtd extension that describes elements
- Provides composition of database's logical model
- Defines the syntax rules or valid tags for each type of XML document
- Companies engaging in e-commerce transaction must develop and share DTDs
- DTD referenced from inside XML document

XML Schemas

- Advanced data definition language
- Describes the structure of XML data documents
- Advantage
  - More closely maps to database terminology and features
- **XML schema definition (XSD)**: File uses syntax similar to XML document

XML Presentation

- XML separates data structure from presentation and processing
- Extensible Style Language (XSL) displays XML data
  - Defines the rules by which XML data are formatted and displayed
  - Parts:
    - Extensible Style Language Transformations (XSLT)
    - XSL style sheets

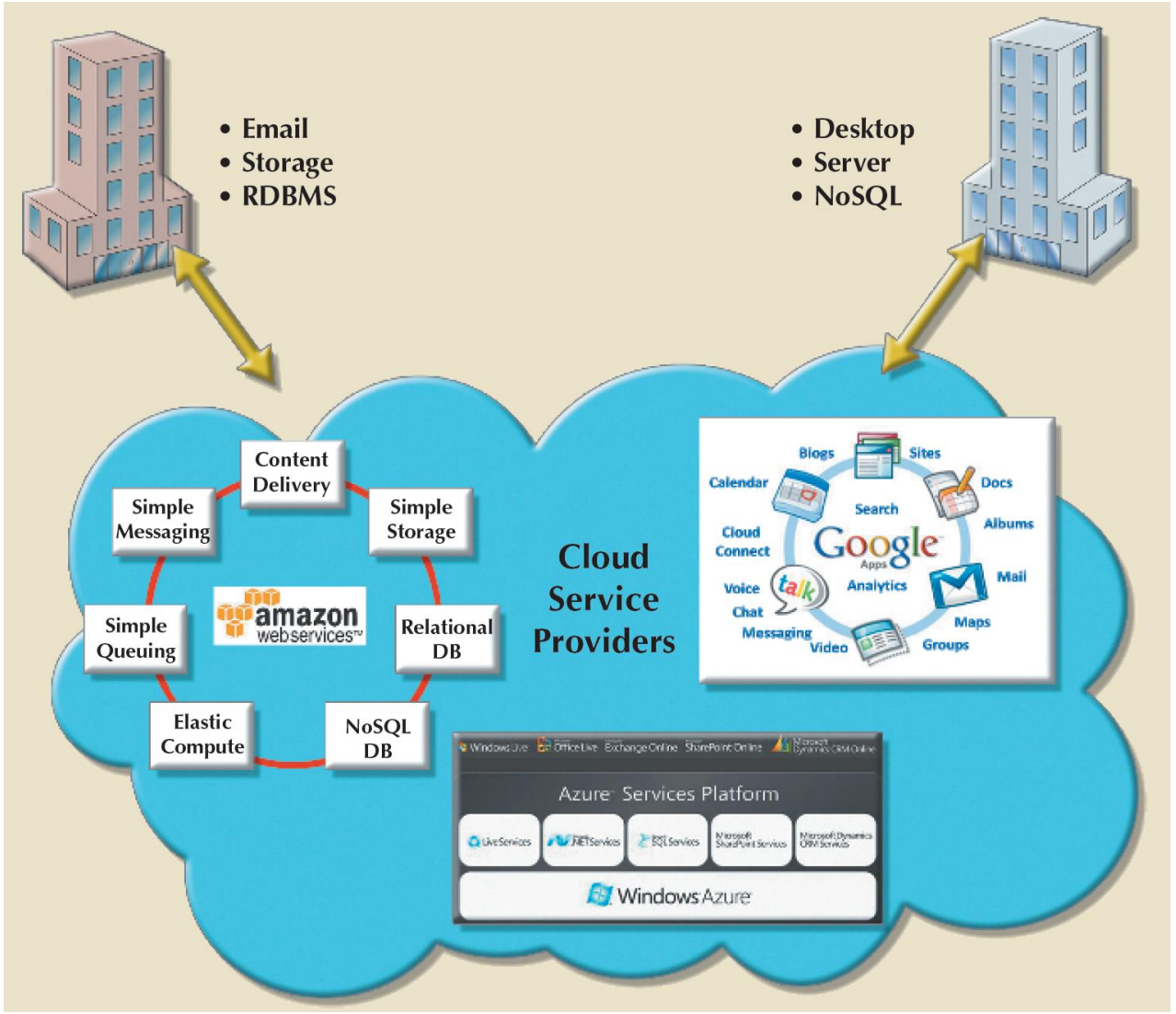
XML Applications

- B2B exchanges
- Legacy systems integration
- Web page development
- Database support
- Database meta-dictionaries
- XML databases
- XML services

Cloud Computing Services

- Computing model that enables access to a shared pool of configurable computer resources that can be:
  - Rapidly provisioned
  - Released with minimal management effort or service provider interaction
- Potential to become a game changer
- Eliminates financial and technological barriers

Figure 15.20 - Cloud Services



Cloud Implementation Types

- Public cloud
  - Built by a third-party organization to sell cloud services to the general public
- Private cloud
  - Built by an organization for the sole purpose of servicing its own needs
- Community cloud
  - Built by and for a specific group of organizations that share a common trade

Characteristics of Cloud Services

- Ubiquitous access via Internet technologies
- Shared infrastructure
- Lower costs and variable pricing
- Flexible and scalable services
- Dynamic provisioning
- Service orientation
- Managed operations

Figure 15.22 - Types of Cloud Services

Table 15.4 – Advantages and Disadvantages of Cloud Computing (1 of 2)

ADVANTAGE	DISADVANTAGE
Low initial cost of entry. Cloud computing has lower costs of entry when compared with the alternative of building in house.	*Issues of security, privacy, and compliance*. Trusting sensitive company data to external entities is difficult for most data-cautious organizations.
Scalability/elasticity. It is easy to add and remove resources on demand.	*Hidden costs of implementation and operation*. It is hard to estimate bandwidth and data migration costs.
Support for mobile computing. Cloud computing providers support multiple types of mobile computing devices.	*Data migration is a difficult and lengthy process*. Migrating large amounts of data to and from the cloud infrastructure can be difficult and time-consuming.
Ubiquitous access. Consumers can access the cloud resources from anywhere at any time, as long as they have Internet access.	*Complex licensing schemes*. Organizations that implement cloud services are faced with complex licensing schemes and complicated service-level agreements.

Table 15.4 – Advantages and Disadvantages of Cloud Computing (2 of 2)

ADVANTAGE	DISADVANTAGE
High reliability and performance. Cloud providers build solid infrastructures that otherwise are difficult for the average organization to leverage.	*Loss of ownership and control*. Companies that use cloud services are no longer in complete control of their data. What is the responsibility of the cloud provider if data are breached? Can the vendor use your data without your consent?
Fast provisioning. Resources can be provisioned on demand in a matter of minutes with minimal effort.	*Organization culture*. End users tend to be resistant to change. Do the savings justify being dependent on a single provider? Will the cloud provider be around in 10 years?
Managed infrastructure. Most cloud implementations are managed by dedicated internal or external staff. This allows the organization's IT staff to focus on other areas.	*Difficult integration with internal IT system*. Configuring the cloud services to integrate transparently with internal authentication and other internal services could be a daunting task.

SQL Data Services

- Cloud computing-based data management service
  - Provides relational data management to companies
  - Hosted data management and standard protocols
  - Common programming interface
- Advantages:
  - Reliable and scalable at a lower cost than in-house systems
  - High level of fault tolerance
  - Dynamic and automatic load balancing, automated data backup and disaster recovery are included
  - Dynamic creation and allocation of processes and storage