Choosing the Appropriate Database Development Tool
Whitepaper

Published: November 1996 *For the latest information, please see //www.microsoft.com/msdn/.*

Contents

Introduction

Microsoft® Solution Providers and MIS developers often ask the question: "Which Microsoft tool should I use to build this database solution?" In most situations, virtually all of the Microsoft tools can be used effectively. Nevertheless, each tool has its relative strengths. The purpose of this paper is to help developers choose the tool best suited to their needs. In particular, this paper will examine Microsoft® Access 97, Microsoft Visual Basic® programming system, Microsoft Visual FoxPro™ database management system[1], and SQL Server™. Mention will also be made of Microsoft Visual C++® development system and Microsoft Visual J++™ development system.

The contents of this paper are as follows:

- Database Development Tools Overview: An overview of each of the above mentioned tools
- Questions and Answers: Structured guide to help developers in their decision-making process
- Web Issues: Overview of using Microsoft tools to develop Internet database solutions
- Appendices

Database Tools Overview

Microsoft is progressively moving its development tools toward a common development environment with shared components between tools. This environment will provide the ability to create a single application by mixing and matching components developed with various languages/tools without leaving the development environment. This trend will enable developers to maximize the power and performance of their solutions by facilitating their ability to build each component of their solution with the most appropriate tool.

Microsoft Access[2]

Microsoft Access has changed the image of desktop databases from specialist applications used by dedicated professionals to standard business productivity applications used by a wide range of users. More and more developers are building easy-to-use business solutions on, or have integrated them with, desktop applications on users' desktops.

Microsoft Access has a tradition of innovation by making historically difficult database technology accessible to general business users. Whether users are connected by a LAN, the Internet, or not at all, Microsoft Access ensures that the benefits of using a database can be quickly realized. With its integrated

---

[1] This paper uses the terms "Visual FoxPro," "FoxPro 2.*x*, " "FoxPro," and the "FoxPro engine." For clarity, "Visual FoxPro" refers to Microsoft Visual FoxPro 3.0 and 5.0, unless a specific version is indicated. "FoxPro 2.*x*" refers to Microsoft FoxPro 2.6 and earlier versions. "FoxPro" refers generically to any version of either FoxPro 2.*x* or Visual FoxPro. "FoxPro engine" refers to the database engine used in Visual FoxPro.

[2] For more information on Microsoft Access, see http://www.microsoft.com/accessdev/.

technologies, Microsoft Access is designed to make it easy for all users to find answers, share timely information, and build faster solutions.

At the same time, Microsoft Access has a powerful database engine and a robust programming language, making it suitable for many types of complex database applications.

Data Engine

Microsoft Access ships with the Microsoft Jet database engine[3]. This is the same engine that ships with Visual Basic and with Microsoft Office. Microsoft Jet is a 32-bit, multithreaded database engine that is optimized for decision-support applications and is an excellent workgroup engine.

Microsoft Jet has advanced capabilities that have typically been unavailable on desktop databases. These include:

- **Access to heterogeneous data sources.** Microsoft Jet provides transparent access, via industry-standard Open Database Connectivity (ODBC) drivers, to over 170 different data formats, including dBASE, Paradox, ORACLE, Microsoft SQL Server, and IBM DB2. Developers can build applications in which users read and update data simultaneously in virtually any data format.
- **Engine-level referential integrity and data validation.** Microsoft Jet has built-in support for primary and foreign keys, database-specific rules, and cascading updates and deletes. This means that a developer is freed from having to create rules using procedural code to implement data integrity. Also, the engine itself consistently enforces these rules, so they are available to all application programs.
- **Advanced workgroup security features.** Microsoft Jet stores User and Group accounts in a separate database, typically located on the network. Object permissions for database objects (such as tables and queries) are stored in each database. By separating account information from permission information, Microsoft Jet makes it much easier for system administrators to manage one set of accounts for all databases on a network.
- **Updateable dynasets.** As opposed to many database engines that return query results in temporary views or *snapshots*, Microsoft Jet returns a *dynaset* that automatically propagates any changes users make back to the original tables. This means that the results of a query, even those based on multiple tables, can be treated as tables themselves. Queries can even be based on other queries.
- **Query optimization using Rushmore™ query technology.** Microsoft Jet has incorporated this innovative technology from Microsoft FoxPro to enhance query performance.

Users of Microsoft Access can work remotely with replicated copies of a database and then merge changes back from one replica to another (and to the original database, called the "design master"). This can be accomplished with the Windows® 95 Briefcase. It can also be accomplished through code.

---

[3] For additional information on the Jet database engine, please refer to *Microsoft Jet Database Engine Programmer's Guide*, published by Microsoft Press.

Microsoft Access supports both full database replication as well as partial replication of particular tables or rows within tables. In addition, changes made by the developer in the design master are propagated to all the replicas. Replication can also be used for managing remote changes to data, distributing Microsoft Access applications, and network-load balancing.

With the Microsoft Replication Manager, available through the Microsoft Office 97, Developer Edition, replication can be scheduled and accomplished over the Internet.

Design Tools

The Database window is the central hub in Microsoft Access and is used to organize and manage files in a database. The Database window has tabs for Tables, Queries, Forms, Reports, Macros, and Modules. This allows users to easily access all objects in the database.

From the Database window the developer can view the Relationships window. It allows users to create database schemas graphically by adding tables to the Relationships window and then relating those tables by dragging and dropping related fields. Relationships can be defined to enforce referential integrity and to cascade updates and deletes. The Relationships window also supplies access to the Table Design view, which allows the developer to modify the structure of tables.

The Form Designer is used to create user forms. All objects on a form have a full complement of properties, methods, and events. Code for validation and other form operations is written in Visual Basic for Applications and can be stored with the form or in a separate code module.

Binding objects and data is easy with Microsoft Access. Complex data-management forms can be created easily by dragging and dropping fields and controls onto the form design surface. If a form is bound to a parent table, dragging a child table onto the form creates a subform, which will automatically display all child records for the parent.

Microsoft Access has a variety of wizards to ease application development for both users and developers. These include:

- The Database Wizard, which includes more than 20 customizable templates to create full-featured applications with a few mouse clicks.
- The Table Analyzer Wizard, which can decipher flat-file data intelligently from a wide variety of data formats and create a relational database.
- Several form and report wizards, which allow users great flexibility in creating the exact view of data required, regardless of underlying tables or queries.
- The Application Splitter Wizard, which separates a Microsoft Access application from its tables and creates a shared database containing the tables for a multiuser application.
- The PivotTable® Wizard, which walks users through the creation of Microsoft Excel PivotTables based on a Microsoft Access table or query.
- The Performance Analyzer Wizard, which examines existing databases and recommends changes to improve application performance.

In addition to the wizards just listed, Microsoft Access provides a number of ease-of-use features, in keeping with its goal of providing easy access to data for users. These include:

- Filter by Form, which allows users to type the information they seek and have Microsoft Access build the underlying query to deliver just that data, in a form view.
- Filter by Input, which allows users to simply right-click on any field, in any view, and then type the criteria they are looking for into an input box on a pop-up menu. Upon pressing ENTER, the filter is applied and the user then sees only the information they are looking for.
- Filter by Selection, which allows users to locate information quickly on forms or datasheets by highlighting a selection and filtering the underlying data based on that selection.

Programming Language and Development Environment

The programming language of Microsoft Access is Visual Basic for Applications. This is the same Visual Basic for Applications that is in the other Microsoft Office applications as well as in Visual Basic. This affords the Microsoft Access developer an impressive amount of leverage. Data access routines, form validations, complex calculations, and much more can be written for a Microsoft Access application and then reused in a Microsoft Excel spreadsheet or Microsoft Word document. In addition, the Visual Basic for Applications language is very similar to the Visual Basic Scripting Edition language that is used in Microsoft Internet Explorer and the Outlook™ 97 Desktop Information Manager .

Programmatic control of the Microsoft Jet database engine is provided through Data Access Objects (DAO). DAO organizes tables, queries, fields, and so forth, into a hierarchy of objects and collections of objects. For instance, in a database there is a collection of tables. Each table is an object and has a collection of fields and indexes. Objects have properties that define their characteristics and methods that are used to perform operations on the objects. DAO provides a full complement of methods to access data. These can be as simple as opening tables and running queries, locating records, or modifying data. They can be as advanced as modifying a table's structure or replicating data.

Programmatic access to data is also provided by ODBCDirect, which uses the same object model as DAO. ODBCDirect provides high-speed, optimized access to ODBC data, without the need to load the Microsoft Jet engine. This provides increased performance and a smaller memory footprint.

Both DAO and ODBCDirect are also available in the other Microsoft Office applications. Data access code written for a Microsoft Access application can be reused in a Microsoft Office application.

The debugging environment in Microsoft Access provides the developer with breakpoints, watch variables, an "immediate" pane, and a Calls window to trace open procedures. The code window features two particularly useful developer aids. QuickTips display the syntax for a function, method, or procedure based on the cursor position. Each parameter is displayed in bold when it is time for the user to type it. An additional developer aid is a drop-down

menu that appears after the user types the name of an object and a period. The menu contains a list of the properties and methods available for the object.

Microsoft Access includes a sophisticated Object Browser, which allows developers to quickly explore an object hierarchy to find specific information. The browser differentiates between built-in properties, custom properties, methods, event handlers, and user-defined procedures.

Class modules in Microsoft Access are modules that can contain the definition for a new object. When a new instance of a class is created, the new object is created. Any procedures defined in the module become the properties and methods of the object. This functionality gives developers greater control of their application code and gives them the ability to create reusable components more easily.

Microsoft Access provides complete integration with the Microsoft Visual SourceSafe™ (VSS) version control system. Note that this integration is supplied through the Microsoft Office 97, Developer Edition and is not included with the base Microsoft Access product.

ActiveX Extensibility and Interoperability

Microsoft Access has excellent support for ActiveX™ controls, including controls that bind to a row of data. It does not, however, support controls that act as containers or that bind to more than one row of data.

Microsoft Access can control Automation servers and, because Microsoft Access itself is an Automation server, it can be controlled by other applications. This would allow a Microsoft Excel application, for instance, to use the Microsoft Access report writer to display information. Unlike Visual Basic and Visual FoxPro, Microsoft Access cannot create custom Automation servers.

Internet Features

Microsoft Access 97 is one of the first desktop databases to support the storage of hyperlinks as a native data type. A hyperlink, when clicked, opens a linked Microsoft Office document or URL address, whether it is located on the World Wide Web, an intranet, corporate LAN, or local machine.

The Internet Assistant for Microsoft Access for Windows 95 has been integrated into Microsoft Access. The Publish to the Web Wizard is a flexible tool that allows users to publish any object in their database either statically or dynamically. It allows for custom HTML formatting using templates and remembers all of the settings used to output the objects in the form of a configuration. The Publish to the Web Wizard integrates with the Microsoft WebPost Wizard to automatically move the published objects to the Web server of choice, whether it is on the Internet or corporate intranet.

The Publish to the Web Wizard allows users to publish data in a table, query, form, or report. Data can be published in three formats:

- Static HTML, which builds a static page of data.
- Dynamic HTX/IDC, which builds IDC and HTX files used by the Internet Database Connector.

- Dynamic ASP, which builds ASP files used by Active Server Pages.[4]

Microsoft Visual Basic[5]

Visual Basic programming system has always enabled developers to quickly and efficiently produce applications for the Microsoft Windows operating system. Visual Basic allows developers to create everything from simple programs to advanced, enterprise-wide client/server applications and take advantage of the latest three-tier client/server capabilities.

With several million units users, Visual Basic is a very popular Windows-based development tool and has a large third party following. Its popularity is due to its rapid application development abilities plus its flexibility, allowing development of a wide range of projects. Whether it's for large-scale team development of three-tiered, distributed enterprise applications, workgroup and workflow projects, or departmental solution building, Visual Basic is a highly effective development environment.

Data Engine

Visual Basic ships with the Microsoft Jet database engine. This is the same engine that ships with Microsoft Access and Microsoft Office. See the "Data Engine" section under Microsoft Access for more information on Microsoft Jet.

Design Tools

The Project window is the central hub in Visual Basic and is used to manage the files that make up a project. These include forms, code modules, class modules, resource files, and ActiveX controls.

The Data Manager application allows the developer to create new Microsoft Jet databases and to examine or map databases in a variety of other formats, such as FoxPro, dBASE, Paradox, or any ODBC data source.

The Form Designer is used to create user forms. All objects on a form have a full complement of properties, methods, and events. Code for validation and other form operations is written in Visual Basic and can be stored with the form or in a separate code module.

Binding objects and data is accomplished through the use of the data control. Properties of the data control are set to identify the data to be used. When the form is opened, the data is retrieved and controls on the form can be populated with data. The Remote Data Control, available in the Enterprise Edition, is designed for high-performance remote data access from Microsoft SQL Server 6.*x*, Oracle7.*x* database servers, or any other ODBC-compliant database servers and is used to populate controls with data from a remote database.

---

[4] For more information on the Internet Database Connector and Active Server Pages, see the Web Issues section of this white paper.

[5] For more information on Microsoft Visual Basic, see http://www.microsoft.com/vbasic/.

Programming Language and Development Environment

The Visual Basic programming language is, obviously, the programming language of Visual Basic. As mentioned in the Microsoft Access section above, it is the same language that is used in Microsoft Access and Microsoft Office. The Visual Basic Scripting Edition language used in the Microsoft Internet Explorer and Outlook is a subset of the Visual Basic language. This provides the developer with an impressive amount of leverage and the ability to write code in one environment and reuse it in another.

Visual Basic ships with the Microsoft Jet database engine and therefore fully supports DAO (Data Access Objects) as the programmatic interface to the Microsoft Jet engine. In addition, the Visual Basic Enterprise Edition provides Remote Data Objects (RDO). Performance tuned for SQL Server 6.*x* and Oracle7.*x* database servers. The ODBC-based Remote Data Objects offer server-specific features that are not accessible using the Microsoft Jet database engine in Visual Basic. The Remote Data Objects require a very small memory footprint, since they are simply a thin "wrapper" on the ODBC-API.

RDO includes the following capabilities:

- **Programmatic control over server cursors,** giving developers precise control over the type of cursor created (key set, static, dynamic, forward, or scroll only) and the type of concurrency used during updates (pessimistic, optimistic on row ID, or optimistic on row values).
- **The ability to return multiple result sets.** Rather than forcing a developer to execute multiple queries to return multiple result sets, Remote Data Objects allow the client/server developer to easily execute a single query that returns multiple result sets.
- **Limit the number of rows returned from a query.** Client/server developers can easily limit the maximum number of records returned from a query.
- **Fatal Error Threshold.** Client/server developers rely on error codes returned from stored procedures to determine both fatal errors and informational status from the execution of stored procedures. For example, an error message returned from a stored procedure might be used to display the message "Quantity on part 1234 is getting low" during an insert of a new order item. When the fatal error threshold is exceeded, the stored procedure will return fatal error or informational status.
- **Asynchronous queries.** Using Remote Data Objects, client/server developers now have the ability to perform queries in the background or to cancel large queries that are in progress.
- **Expose underlying ODBC handles.** By exposing the underlying ODBC handles, client/server developers have complete freedom to gain access to any back-end features not fully exploited by the Remote Data Objects. This is analogous to exposing window handles for forms and controls in Visual Basic.

Class modules in Visual Basic are modules that can contain the definition for a new object. When a new instance of a class is created, the new object is created. Any procedures defined in the module become the properties and

methods of the object. This functionality gives developers greater control of their application code and gives them the ability to create reusable components more easily.

The debugging environment in Visual Basic provides the developer with breakpoints, watch variables, an "immediate" pane, and a Calls window to trace open procedures. The most popular debugging tools—stepping through code while displaying watch variables or executing statements in the immediate window—are combined and visible in a single window. The two panes of the Debug window closely mirror the information that developers need while debugging. The bottom half is a fully functional Code Editor, while the top half toggles between a watch window and an immediate pane.

Visual Basic includes a sophisticated Object Browser, which allows developers to quickly explore an object hierarchy to find specific information. The browser differentiates between built-in properties, custom properties, methods, event handlers, and user-defined procedures. The Object Browser allows developers to easily identify and manage all of the Automation servers currently referenced by their project. Once developers locate the desired objects within the Object Browser, they can easily paste the correct syntax directly into their code or even jump to a detailed help topic that describes the use of the objects.

The Enterprise Edition includes an integrated tool for analyzing the performance of a Visual Basic-based application. Implemented as an add-in, the profiler examines two important performance characteristics of programs: code coverage and code optimization. Code coverage is the process of determining whether or not a particular function or line of code has been executed. Code optimization is the process of determining which lines of code are executed most often.

Visual Basic provides complete integration with the Microsoft Visual SourceSafe (VSS) project-oriented team development system. This integration is included with the Professional and Enterprise Editions of Visual Basic.

The Visual Basic development environment has been exposed as an OLE object that can be manipulated programmatically with Automation. Microsoft exposed the Visual Basic development environment to encourage development of extensibility enhancements in this new direction. Many third-party tools now can be modified easily to allow them to integrate with the Visual Basic development environment. The new Add-In menu item and Add-In Manager have been added specifically to support this functionality. Using the new exposed interfaces, third parties can do the following:

- Add menu items to the Visual Basic menu structure, integrating the two tools interfaces
- Modify Visual Basic projects and add forms, controls, and code
- Support two-way communication between the third-party tool and Visual Basic

The following areas provide some of the first examples of tools utilizing these capabilities.

- Source code control.
- CASE and data modeling tools.
- Design aids, form design, and code generation add-ins

ActiveX Extensibility and Interoperability

Visual Basic has complete support for ActiveX controls, including controls that act as containers and controls that bind to more than one row of data.

Visual Basic can control Automation servers, and the Professional and Enterprise Editions include the ability to create Automation servers (formerly known as OLE Automation servers or OLE servers). An Automation server is an application that exposes functionality that can be used and reused by other applications, including Web sites, through Automation. Using Visual Basic programming system a developer can create an Automation server that displays reusable forms, implements business rules, or packages a complex routine into a simple component that other programmers can use.

Remote Automation allows solution developers to create true distributed multitier client/server applications. It enables developers to run an Automation server on one machine and the client application on a separate machine on the network. This feature is extremely useful for implementing three-tier client/server applications, asynchronous processing, offloading work to underused machines, and reducing client maintenance.

Visual Basic Enterprise Edition includes the Component Manager, a tool that allows a developer to catalog, identify, and locate Automation servers in a networked environment. With full searching and sorting capabilities, the Component Manager simplifies the identification of existing Automation servers that can be incorporated into a developer's solution.

In addition to maintaining references to components that may be located anywhere in an enterprise network, the Component Manager also provides access to associated files that contain detailed information about using the component, sample client code, registration files, functional specifications, registration files, performance data, run-time dependencies, licensing, and so forth. The Component database can reside on the user's desktop or in any remote ODBC database.

Internet Features

Visual Basic supports the Internet in several key ways:

- **ActiveX controls.** Using ActiveX controls, developers can add Internet functionality to Visual Basic applications. Developers can build Web browsing and FTP file transfer right into their Visual Basic applications.
- **Deploying Visual Basic Applications on Internet Servers.** Visual Basic applications can be built as Automation servers and can be deployed on an intranet or Internet server, providing database services to Web browser-based clients.
- **Visual Basic Scripting Edition.** VBScript is a high-performance scripting language designed to create active, online content on the World Wide Web. It allows developers to link and automate a wide variety of objects in Web pages, including ActiveX Controls and "applets" created using the Java™ language. The Microsoft Internet Explorer and Active Server Pages both fully support VBScript.

Microsoft Visual FoxPro[6]

Microsoft Visual FoxPro database management system version 5.0 is a powerful object-oriented environment for database construction and application development. Visual FoxPro 5.0 provides the tools developers need to manage data—whether they are organizing large tables of information, running queries, or programming a fully developed data-management application for users.

Visual FoxPro brings several unique strengths to the Visual Tools family, including:

- An object-oriented tool set for creating reusable components
- A data-centric tool set with a strong local database engine and client/server connectivity
- A powerful interactive data manipulation environment

Microsoft Visual FoxPro 5.0 is a professional developer's database tool, at the high-end of the desktop database market and in the rapid application development (RAD) and object-oriented programming (OOP) tools markets. The strength of Visual FoxPro lies in its maturity, power, and performance relative to the competition, as well as in the large number of FoxPro developers who have years of experience building high performance database applications.

Data Engine

The Visual FoxPro database engine is a full-featured file-server relational database engine that is optimized for large datasets. Database applications can never be fast enough. This is precisely the area where FoxPro has become legendary for consistently outperforming the competition and for offering blazingly fast access to large amounts of data.

Visual FoxPro's data dictionary eliminates the need for the developer to write code to enforce various types of data integrity. With the Table Designer the developer can attach validation rules and defaults to individual fields in a table. Row level validation rules can be supplied as well. The developer can also write Insert, Update, and Delete triggers to be automatically run when any of these operations are performed.

Defaults, rules, and triggers can be simple expressions, such as *price must be greater than cost*, or they can be developer written functions. These functions can be stored in a separate procedure library or as stored procedures in the database. This allows the developer to write any amount of code required to enforce data integrity and to have that code automatically run when the user modifies data.

Although Visual FoxPro does not provide engine-level declarative referential integrity, it has a Referential Integrity Builder, which can build the required triggers to not only enforce referential integrity but also to supply cascading deletes (automatically delete child records when a parent record is deleted) and cascading updates (automatically change the foreign key in child tables if the primary key is changed in a parent table).

---

[6] For more information on Microsoft Visual FoxPro, see http://www.microsoft.com/vfoxpro/.

Local, Remote, and Offline Views

A view in Visual FoxPro is a query result set that can be updated. It is similar to a Microsoft Access dynaset. The view's definition consists of a Select statement that retrieves data. Parameterized views are used to limit the amount of data in the result set. For example, an Orders view may consist of a Select statement that joins the Orders table to the Order Details table and may be parameterized on the order number. The user may enter the order ID on a form and then press a button to rerun the view's Select statement and populate the form with the resulting data.

Remote views are the same as local views except that they are based on ODBC (Open Database Connectivity) data such as that held in a Microsoft SQL Server database. They are created and designed with the View Designer in the same manner as local views.

Visual FoxPro allows developers to build prototypes of their client/server applications using local views that use local FoxPro tables. When it is time to deploy the application using remote data, the developer can use the Upsizing Wizard to create a SQL Server or Oracle version of the FoxPro data. If the application's forms and reports are based on views, the data source can be changed from local to remote in a handful of lines of code.

There are times when the developer might want to display, collect, or modify data independently of the host database. By using the new Offline View features in Visual FoxPro, views can be used to connect to a host database (which could be either a FoxPro database or an ODBC-connected server) and create a subset of data for use offline. Then, working offline, the view can be used directly or through an application. Any changes stored in the view can then be uploaded back to the host database.

A key benefit of offline views in Visual FoxPro 5.0 is that minimal code changes are required to use them. The application is simply pointed at the offline view, instead of the normal view of the data, and then all fields, indexes, tables, forms, and so forth, are accessible using the exact same names. Developers implementing offline views can then write custom code to handle conflicts that may have occurred if the same rows in both views have been updated.

Design Tools

The Project Manager is the central hub for application development in Visual FoxPro and is used to organize and manage files in projects. The items in the Project Manager are organized in an outline view that users can expand or collapse. This allows users to access all files associated with the project very quickly and easily.

The Database Designer displays all tables, views, and relationships contained in the database. It allows users to create database schemes graphically by adding tables to the Database Designer and then relating those tables by dragging and dropping related fields. The Database Designer also supplies access to the Table Designer, which allows the developer to modify the structure of tables.

The Database Container in Visual FoxPro is a repository for all related tables, local views, remote views, and connections. When a user creates a connection to back-end data, it is stored within the Database Container. When

relationships are set within the Database Container, they are persistent throughout the entire application.

The Form Designer is used to create user forms and toolbars. All objects on forms have a full complement of properties, methods, and events. The developer can create custom form level properties and methods as well. Code for validation and other form operations is written in the Visual FoxPro language and can be stored with the form, in a separate program, in a class definition, or in a stored procedure in the database.

Visual FoxPro has also made binding objects and data easy with the graphical Data Environment Designer. Complex data-management forms can be created easily by dragging and dropping fields and controls onto the form design surface. If a form is bound to a parent table, dragging a child table onto the form creates a grid control, which will automatically display all child records for the parent.

Field mapping enables the developer to specify what type of control Visual FoxPro creates when a table or field is dragged to a form. The control can be one of the built-in controls or a custom class. This would allow the developer to build a custom state or country combo box and have that placed on the form when a state or country field is dragged from a table.

While designing a form, users can save a group of objects as a class from within the Form Designer. The class can then be reused on other forms not only in the current application but in others as well. As an example, suppose the developer places controls for name, address, and phone number on a customer form. Realizing that this is a common group of controls, the developer saves it as a visual class. When creating an employee form, the developer can reuse the class and does not need to create another group of controls to handle the same type of information. This can be a huge timesaving in development, especially over time as the class is used in other applications.

The Visual Class Designer is also used to create classes. It is the only way to visually modify classes. It shares the same look and feel of the Form Designer. The developer can create custom properties and methods of a class, which complement the built-in properties and methods. Custom classes can be created from any of the built-in Visual FoxPro base classes, including forms, command buttons, text boxes, and grids, to name a few.

Visual FoxPro includes several wizards to automate the process of creating database objects. These include wizards for creating tables, queries, forms, reports, mail merges, and pivot tables. In addition, there are wizards to help the developer import data into FoxPro tables, distribute an application, and upsize an application to SQL Server or Oracle databases.

Programming Language and Development Environment

While Visual FoxPro still supports standard xBASE procedural programming, new extensions to the language give users the power and flexibility of object-oriented programming. Object-oriented design and object-oriented programming represent a change in focus from standard procedural programming. Instead of thinking about program flow from the first line of code, users need to think about creating objects—self-contained components of an application that have private functionality as well as functionality that developers can expose to the user. Visual FoxPro allows users to create

classes visually (using the Class Designer and Form Designer) and through code (with the DEFINE CLASS command).

The Visual FoxPro object-oriented programming model includes:

- **Inheritance,** which allows a newly created object to inherit characteristics (that is, properties and methods) from its parent. Although objects can inherit characteristics from their parents, they can also override the inheritance. Visual FoxPro fully supports inheritance and overrides, both through code and visually.
- **Subclassing,** which takes inheritance one step further by extending the feature to custom-built classes. Classes can be build based on other classes. The subclasses then inherit from their parent class.
- **Encapsulation,** which allows developers to encapsulate attributes (properties) and behaviors (methods) into objects. Even though the internal implementation and management of an object may be completely hidden, the object knows everything it needs to know in order to behave accordingly.
- **Polymorphism,** which allows a specific function or method to behave differently depending on the object it is associated with. Polymorphism allows users to create structured class hierarchies, where the basic skeletons of objects are defined in the base class and specialized code is actually defined in the derived subclasses.

The Class Browser is used to browse, track, and modify classes. It also allows the developer to create subclasses, copy classes from one class library to another, remove classes and library files, rename a class, or redefine the relationship of a class. The Class Browser further allows the developer to view a hierarchy of classes and to track the inheritance of members between related classes. The Class Browser also allows the developer to view properties and methods of any Automation server.

Visual FoxPro includes a full-featured Debugger that runs in its own separate window and has five windowpanes: Trace, Watch, Locals, Call Stack, and Output. The Debugger includes a number of advanced features, including the ability to save and load a debugging session, persistence of settings between sessions, event tracking, coverage logging, and complex breakpoint support. Drag-and-drop is fully enabled between all debug surfaces.

Visual FoxPro 5.0 provides complete integration with the Microsoft Visual SourceSafe (VSS) project-oriented team development system.

ActiveX Extensibility and Interoperability

Visual FoxPro 5.0 has excellent support for ActiveX controls, including controls that act as containers and controls that are bound to a row of data. It does not, however, support controls that bind to more than one row of data. Visual FoxPro is unique in its ability to Vtable bind to ActiveX controls and Automation servers. This provides considerably quicker access to properties and methods and speeds overall performance.

Visual FoxPro can control Automation servers and, because Visual FoxPro itself is an Automation server, it can be controlled by other applications. This would allow a Microsoft Excel application, for instance, to run a Visual FoxPro routine to mine and collate data.

As discussed above, one of the most powerful features of Visual FoxPro is the ability to create and reuse classes that provide common functions. With Visual FoxPro 5.0, this paradigm was extended to include ActiveX controls. A class definition can include ActiveX controls. The class definition can set properties and methods for the ActiveX control, modifying its default behaviors so that each instance of the class then provides the exact behavior the developer desires. This provides Visual FoxPro developers with fine-grained control over reusable objects. It also allows the developer to build data binding into controls that don't inherently support data binding, for instance, a class based on a calendar control can be bound to a date field.

In the same manner as Visual Basic, Visual FoxPro 5.0 includes the ability to create Automation servers to expose functionality that can be used and reused by other applications. A Visual FoxPro developer could wrap existing data manipulation code into an Automation server that can be called from any of the Microsoft Office applications or by the Internet Information Server, to list two examples. This allows the Visual FoxPro developer to maintain his or her investment in existing code while taking advantage of the latest advances in component technologies.

Also in the same manner as Visual Basic, a Visual FoxPro Automation server supports Remote Automation to create true distributed multitier client/server applications.

Internet Features

Visual FoxPro 5.0 supports the Internet in several key ways:

- **ActiveX controls.** Using ActiveX controls, developers can add Internet functionality to Visual FoxPro applications. Developers can build Web browsing and FTP file transfer right into their Visual FoxPro 5.0 applications and even create subclasses of ActiveX controls.
- **Deploying Visual FoxPro Applications on Internet Servers.** With this release, Visual FoxPro applications can be built as Automation servers and can be deployed on an intranet or Internet server, providing database services to Web browser-based clients.
- **FOXISAPI.** The Foxisapi library allows developers to create forms in Visual FoxPro and run them from a Web page. The Foxisapi library and sample code to demonstrate its capabilities ship with Visual FoxPro.

Visual FoxPro also supplies a WWW Search Page Wizard, which creates a Web page that allows visitors to search and retrieve records from a Visual FoxPro table. The wizard asks for the table to be searched, the search field, and information about how to format the Web pages it creates. The wizard creates an HTML page for entering search information, a parameterized query for retrieving the records from the table, and a results page for displaying the records the query retrieved.

Microsoft SQL Server

Microsoft SQL Server 6.5 combines the best of traditional mainframe computing—centralized security, data integrity, and control—with the best of today's PCs—ease of use, rich user interfaces, and a variety of off-the-shelf productivity tools. It makes it possible for multiple front-ends to share information, enabling the developer to choose the most appropriate tool for the

job. SQL Server makes efficient use of networks; because database queries are processed at a centralized server, network traffic is reduced.

Microsoft SQL Server incorporates a world-class feature set for distributed client/server computing. Customers using SQL Server will see benefits in the following key areas:

- Reliable distributed data and transactions
- Centralized control of distributed servers
- Very high performance and scalability
- Support for very large databases
- Full programmability and standards support
- Rich desktop integration
- Open interoperability

SQL Enterprise Manager is a graphical Windows-based tool that lets the administrator manage all of an organization's distributed SQL Servers from a single point of control, from a Windows NT® or Windows 95 operating system environment. It simplifies complex administration tasks such as storage allocation and permissions management through a Windows-based user-interface. SQL Enterprise Manager provides graphical management of database objects such as tables, views, stored procedures, and triggers. Visual Basic–based scripting can extend these capabilities to automate remote operations across multiple servers. Other graphical utilities are included for managing security, tape volumes, and client configuration.

SQL Server administration architecture is based on an open, Automation architecture, called SQL Distributed Management Objects (SQL-DMO). This makes it extensible with a wide array of tools that can control Automation servers. SQL Server 6.5 adds built-in data and schema transfer capabilities and a new DBA Assistant that automates routine maintenance operations.

Unlike database servers, which run only on proprietary operating systems or support only proprietary network protocols, SQL Server is network independent. Because SQL Server relies on open industry standards, it can run on most popular networks, including IBM LAN Server, Microsoft LAN Manager, Novell NetWare, Banyan VINES, DEC PATHWORKS, Apple AppleTalk, and Windows NT Server-based networks.

SQL Server's Transact-SQL language provides compatibility with the industry standard data access language. Transact-SQL includes important extensions necessary to support mission-critical applications such as stored procedures and triggers. SQL Server cursor support simplifies development of rich data browsing applications with capabilities such as forward/backward scrolling, positioned updates and deletes, and flexible concurrency control options.

Consistency and recoverability of a database are guaranteed in case of system failure—even in the middle of complex updates by more than one user. SQL Server treats all database changes inside a transaction as a single unit of work. By definition, either an entire transaction is completed safely and all resulting changes are reflected in the database, or the transaction is rolled back—and all changes to the database are undone.

Transaction processing is a vital requirement for a system designed to support mission-critical applications, such as accounting, inventory management, and

online order entry. Using a two-phase commit protocol, SQL Server even supports synchronized transactions that span more than one server—helping to guarantee that all servers on the network will be maintained in a consistent state.

Another benefit of the SQL Server transaction processing design is implicit concurrency control. SQL Server employs Dynamic Locking, a locking architecture that keeps concurrent users from interfering with each other during queries and updates. Page-level locking is the default, with optional insert row-level locking now available in version 6.5.

All SQL Server locking is implicit—the programmer doesn't have to worry about locking commands. The process of obtaining a lock is exceptionally fast—a matter of microseconds—since lock information is stored in a memory-resident table. Multiple levels of locking are supported, and SQL Server always picks the least restrictive lock needed to support the operation.

SQL Server has server-enforced data integrity support for data integrity, and complex business policies can be incorporated in the database itself, where it can be maintained centrally and enforced uniformly for all applications. SQL Server 6.5 offers ANSI standard referential integrity and column level constraints, and also has advanced data integrity features such as rules, defaults, stored procedures, triggers, and user-defined data types to enforce data integrity.

SQL Server allows precompiled collections of SQL commands (stored procedures) to be stored in the database, a feature that, in some cases, can decrease query processing times by as much as 80%. Stored procedures can also be used to centralize application logic in the database, allowing the developer to program the server to perform complex queries or updates in response to a simple procedure call. Because stored procedures can accept passed parameters, their behavior can be flexible and dynamic. Stored procedures can be written once and shared among applications, greatly simplifying the application development process.

SQL Server also supports extended stored procedures that call external C-based DLLs for even more flexibility. Version 6.5 introduces Automation stored procedures, which let the developer extend SQL Server with virtually any Automation-compatible programming environment.

Triggers prevent incorrect, unauthorized, or inconsistent changes to data. Triggers are specialized forms of stored procedures that are executed automatically by SQL Server whenever any attempts to modify associated data are made. Triggers are incredibly powerful. They allow the developer to enforce data integrity and complex business policies centrally in the server, instead of having to program this logic into individual applications. The following are some examples:

- Not accepting a new customer credit order when outstanding debts for that customer exist.
- Automatic updates to the total product cost whenever a component's part cost is updated.
- Not allowing updates that attempt to increase a credit limit by more than 25%.

Triggers are a crucial feature in a production DBMS; SQL Server exemplifies an implicit understanding of this crucial requirement.

Rules are another SQL Server tool to enforce data integrity centrally, but at the individual column level. Rules can automatically require that a value fall within a particular range, appear within an explicit list, or even match a string pattern. For example, a database containing information on senior citizens could have a rule that *the age field must contain a value between 65 and 120 years.*

Default settings specify that SQL Server will automatically insert a standard value in a column if the user does not supply one. For example, a *state_code* field might have a default of "CA" for a company that does business primarily in California. Rules and defaults can be connected to a particular column, a number of columns, or all columns in the database that have a given user-defined data type.

SQL Server allows developers to create their own data types (user-defined data types) to supplement those provided by SQL Server. The advantage of user-defined data types is that rules and defaults can be bound to them for use in multiple tables, as well as tailor them to specific applications. For example, a *state_code* data type might be defined as two characters (char[2]) with an associated rule that requires all values to appear in a list of 50 valid state codes. This behavior is automatically applied whenever a new column having a *state_code* data type is added to a table.

SQL Server incorporates a built-in data replication architecture that offers reliable, high-performance distribution of data in an enterprise while fully preserving transaction semantics. With an intuitive "publisher/subscriber" metaphor and "drag/drop" management interface, SQL Server's replication is easier to use and more flexible than other database implementations. It also supports both log-based replication and "point-in-time" synchronization (snapshots), giving administrators a choice of replication strategies. SQL Server offers distributed security integrated with the Windows NT operating system, with central control of passwords across a domain and new encryption services for logon and data stream. Data being replicated can also be encrypted over the wire.

SQL Server 6.5 includes new OLAP style query operators, CUBE and ROLLUP, that return a multidimensional result set from a single query, simplifying the retrieval of information for analytical purposes. A new "data pipe" capability, using INSERT….EXEC, lets SQL Server collect data from other SQL Servers and data sources for consolidation into a data warehouse or data mart.

Microsoft SQL Server provides consistently high performance in a client/server DBMS. Conventional DBMSs accommodate concurrent users in two basic ways: by creating a separate process for each user (considerably increasing system overhead and memory usage) or by using a single process to service all user requests—meaning that queries are processed in serial order instead of parallel order.

SQL Server is optimized for Windows NT and uses a very efficient design that incorporates multiple native threads within a single process to handle user requests—allowing queries to be processed in parallel with very little overhead and no run-time memory allocation. This architecture is also memory efficient—SQL Server needs only about 40K of memory for each additional user. That compares to the 400K or more needed by other database servers. The major advantage of the SQL Server multithread design is throughput. SQL Server doesn't slow down as multiple users are added to the network.

SQL Server avoids costly downtime for routine maintenance tasks. Nothing is more detrimental to productivity than network resources that become periodically unavailable. SQL Server dynamic backup allows the administrator to back up databases even while users are actively reading and writing to them—a fundamental requirement for mission-critical applications.

In case of system failure (operating system crashes, power outages, and so forth), SQL Server's automatic recovery mechanism recovers all databases to the last state of consistency in a matter of minutes, with no administrator intervention. Applications can be up and running again right away.

The SQL Server high availability design even allows the administrator to perform database design changes or diagnostics while the system is online. New features in SQL Server 6.5 include backup/restore of individual tables, making disaster recovery easier, and point-in-time recovery that lets a database be restored to a certain point in time. SQL Server 6.5 also supports Compaq's Online Recovery Server, for automated fail-over in the event of a system failure.

SQL Server's device mirroring capability provides fault tolerant operation to meet critical business application requirements. A high availability design allows any SQL Server database device to be mirrored, unmirrored, or remirrored on the fly, and automatic cutover provides nonstop operation in case of I/O failures that require OS recovery. SQL Server device mirroring allows an administrator to establish a very granular fault-tolerant environment (mirroring only transaction logs, for example). Microsoft SQL Server also fully supports Windows NT Server RAID 5 or hardware mirroring for fault-tolerant operation.

SQL Server provides the utmost in protection for sensitive data. SQL Server implements comprehensive user-level permissions on tables, views, stored procedures, and SQL commands. It also supports field-level security. And, because permissions can also be applied to groups, security is easy to implement and administer. Since security is handled centrally by SQL Server, elaborate user validation logic doesn't have to be coded into client applications.

SQL Sever also directly integrates with the Windows NT operating system security to provide a single log on to both network and database resources, with advanced password aging and user account lockout control. New encryption services provide secure over-the-wire data encryption between clients and servers for a high level of distributed security.

SQL Server offers open (documented and published) application programming interfaces, DB-Library and ODBC, that can enable any software developer to build client applications for SQL Server. Hundreds of available applications, tools, and turnkey business solutions support SQL Server already. Microsoft is committed to continued enhancement of these interfaces to preserve customer investments.

SQL Server supports large TEXT and IMAGE data types, with a limit of 2GB per field, making it an ideal platform for document and image management applications. These fields are also very useful for managing Web content.

SQL Server distributed data management allows workstations to access and perform transactions to several SQL Servers at the same time, making true distributed applications possible. SQL Server allows the creation of a single distributed transaction that updates multiple servers while enforcing data integrity and consistency. If the workstation or any server were to crash, the partial transactions on all other servers would be rolled back automatically.

SQL Server 6.5 makes this process transparent to the application, with a new component called Distributed Transaction Coordinator (DTC). DTC supports popular programming interfaces such as ODBC, DB-Library, Transact-SQL, XA, and OLE transactions.

Support is a very important component of a product such as SQL Server. Microsoft offers fee-based, toll-free technical support 24 hours a day, seven days a week. Mission-critical support includes: immediate, server-down support; technical and management issue escalation; remote diagnostics, problem replication labs; hot fix and service pack releases; detailed bug lists and fix lists; and online services.

Q & A

The questions in this section should be used in conjunction with the product overviews in the sections above. These questions highlight some of the differences between the database tools and between various database strategies.

Database Engine Questions

1. Which is faster, the FoxPro engine or the Microsoft Jet engine?

   Traditionally, the FoxPro engine has been the fastest desktop database engine. The legendary data access performance in FoxPro stems from its Rushmore query optimization and use of fixed length storage of data. Rushmore creates bit-mapped indexes that allow the engine to very quickly determine whether each row in a table meets a criteria and should therefore be included in a query's result set. Fixed width storage of data ensures that a field will be in the same location from row to row, making that field easier to locate.

   Early versions of the Microsoft Jet database engine were not optimized for multiuser performance with large data sets. The emphasis in Microsoft Access versions 1.0 and 2.0 was more on ease of use, not on engine performance. However, the 3.0 version of the Microsoft Jet database engine, which appeared in Microsoft Access 95, was completely rewritten as a 32-bit multithreaded process. The Microsoft Jet team incorporated much of the performance work, including Rushmore technology, that made the FoxPro engine fast. Additional performance work was done in the 3.5 version of Microsoft Jet, which ships with Microsoft Access 97 and Microsoft Office 97.

   Both the FoxPro and the Microsoft Jet engines have been engineered to provide excellent performance with a range of database sizes. In many situations, there might not be a significant performance difference between them.

   In querying situations, in particular, the difference between the two engines may be small, due to the fact that the Microsoft Jet engine was rewritten to use many of the performance techniques originally found in the FoxPro engine.

   The two engines differ in how they lock data. FoxPro locks records whereas Microsoft Jet locks 2K pages of data. In earlier versions of Microsoft Jet (version 2.5 and earlier), this caused contention when adding rows because only one user at a time could have the last page of data

locked. This issue has been addressed in the latest version of the Microsoft Jet engine, which will allow data at the end of a table to be spread out among several pages, thus allowing multiple users to add rows.

Microsoft Jet still locks the entire page when a row is updated, whereas FoxPro would only lock that row. Many users updating records that are physically near each other could lead to a performance advantage for FoxPro. However, there are more similarities in the latest versions of the engines (Microsoft Jet 3.5 and Visual FoxPro 5.0) than there are differences.

2. Is a file-server or client/server based solution needed?

   The Microsoft Jet and FoxPro engines are file-server based. The data and indexes reside either on the user's local machine or on a network file server. In either case, most processing is performed locally on the client machines. The Microsoft Jet engine and FoxPro engine are tuned to be as efficient as possible in sending indexes and/or data over the network to the client. To locate a single record, only those pages of the index that are required to locate the record are sent from the server. At that point, either engine can request from the file server the page (or pages) that contains the record's actual data.

   SQL Server is a server-based DBMS. The data and indexes reside on a single machine that is accessed over the network. Clients send SQL statements over the network to SQL Server, which processes the statements and then sends results and/or messages to the clients.

3. When should an application be moved to SQL Server?

   Customers often ask for a general rule of thumb as to when they should move from using the Microsoft Jet or FoxPro engines to the SQL Server engine. This is a difficult question to answer.

   A rule of thumb that is expressed in terms of number of users or size of database runs the risk of inaccuracy and is also subject to becoming outdated very quickly, as products are improved. For example, with proper tuning, the current version of Microsoft Access can certainly handle 50 concurrent users and 100 megabytes of data, whereas previous versions could not.

   With a small number of users, good hardware, and proper indexes, there is no reason to expect a significant performance difference between the FoxPro, Microsoft Jet, and SQL Server engines in many types of applications. The more users in the system and the less capable the client machines and LAN infrastructure, the more likely SQL Server would outperform the Microsoft Jet and FoxPro engines. One reason SQL Server may outperform those engines is that its processing power resides on a high-end machine that is not restricted by bottlenecks such as slow network infrastructure (that is, NIC cards, slow protocol, and transport).

   The decision to migrate to SQL Server will often not be based on performance alone. The developer might decide to take advantage of SQL Server features that are not available in the FoxPro or Microsoft Jet engines, such as dynamic backup and automatic recovery. The answer to the question, then, is "if you are dissatisfied with performance or are seeking the benefits of using a client/server database, you should investigate Microsoft SQL Server." Both Visual FoxPro and Microsoft

Access offer upsizing tools to migrate data to SQL Server.

4. What is the expected byte size of the database?

   A Microsoft Jet database can hold up to 1.2 gigabytes of data. A Visual FoxPro database has no size limitations, although each table can hold up to 2 gigabytes of data. Although it is certainly possible to split the data into more than one database or table, there is a loss of functionality (particularly referential integrity) as well as an administrative price to pay. SQL Server can comfortably handle databases of 200 gigabytes of data today, and that number is expected to grow to between 500 gigabytes and 1 terabyte with upcoming versions of SQL Server.

5. Will the solution be a transaction[7]-based database with a high cost of downtime?

   SQL Server is the best choice when the solution is based around a high volume of mission-critical transactions (for example, banking, 911 calls, and so forth). In SQL Server, every transaction is logged; thus, if for any reason the system should fail, all committed changes are completed and uncommitted changes are rolled back automatically, thereby helping reduce the expense of downtime. Although Microsoft Jet and Visual FoxPro have the ability to handle transactions, they do not support automatic recovery in the case of a system crash. Data can be lost if the network goes down.

6. Will the system need to be available 24 hours a day, seven days a week?

   SQL Server supports dynamic backup and does not require users to leave the system to back up data. The data engine itself can create backup files by reading data and copying it to disk or tape. On relatively modest hardware, SQL Server can achieve backup rates of up to 20 gigabytes per hour. In addition, the transaction log can be backed up separately from the database, thus allowing a mix of complete and incremental backups.

   The Microsoft Jet and FoxPro engines do not have built-in backup capabilities. They require all users to be out of the system, at which point the files containing the data can be backed up using standard network backup procedures.

7. What degree of security will be required?

   In order to have tight database security, security must be provided at the level of the network, servers, databases, tables, and SQL statements. SQL Server and Microsoft Jet both offer a high degree of security at each of the data levels, offering logon IDs and passwords, user permissions, and encryption. SQL Server also allows integration of its security with network security. While the FoxPro engine has no native security, Win32® encryption API and third-party solutions are available to allow developers to customize security to their needs.

[7] For more information on transactions, see Appendix C: Glossary.

8. Does the application require replication?

   Replication allows data to be copied automatically from one database to another. Typical uses include distributed processing, load balancing, and dynamic backup. Each of the database engines provides support for replication, though their goals are somewhat different. Replication in SQL Server is designed for high-volume, mission-critical use. Microsoft Jet replication is designed for ease-of-use and flexibility. Replication in Visual FoxPro is designed for performance and flexibility.

   SQL Server supports the publish-and-subscribe model of replication to other SQL Server-based systems and to ODBC data sources. Updates to each table are restricted to a single site, to enforce guaranteed logical consistency in the database. The Microsoft Jet engine supports the multimaster model of replication with other Microsoft Jet databases. Updates at multiple sites may be permitted, which provides flexibility at the cost of potential conflicts or inconsistencies. Conflict resolution mechanisms are provided, but at any given time, the database is not guaranteed to be consistent, as is the case with SQL Server.

   Partial replication of Microsoft Jet databases, where entire tables or just those rows that meet specified criteria, such as Western region sales or New York customers, was introduced in Microsoft Jet 3.5. SQL Server 6.5 supports partial replication both at the row and column level. Both Microsoft Jet and SQL Server support scheduled synchronization (Microsoft Jet requires Microsoft Replication Manager, provided in Microsoft Office 97, Developer Edition, in order to support scheduled synchronization).

   The FoxPro engine supports full and partial table replication in Visual FoxPro 5.0, through the Offline View feature. Offline Views are local, updateable snapshots of FoxPro engine views and can be based on local or remote data. Replication in the FoxPro engine is based on the publish-and-subscribe model. Scheduled synchronization requires additional coding.

9. Will technical support be required seven days a week, 24 hours a day?

   Round-the-clock support can be purchased from Microsoft for each Microsoft database engine. In addition, SQL Server offers Quick Fix Engineering, offering customer-specific fixes for identified problems.

Visual Tool Questions

1. What platform(s) will the database clients be on?[8]

   For solutions that require client implementations on the Macintosh, the only Visual Tools available are Visual C++ and Visual FoxPro. To develop solutions for the Windows 16-bit operating system, it is necessary to either use earlier versions of the Microsoft Visual Tools or to develop a Web strategy.[9] Microsoft continues to sell 16-bit versions of each of the Visual Tools. Finally, for MS-DOS® and UNIX solutions, developers must use FoxPro 2.*x* for the UNIX or MS-DOS operating systems or choose a Web strategy.

---

[8] For more information on cross-platform development, see the "Web Issues" section.

[9] For more information on Web database solutions, see the "Web Issues" section.

2. Will a Microsoft Office component be a significant part of the solution?

   From the user's perspective, Microsoft Access has the best visual integration with Microsoft Office. With one menu choice a user can take a query and send its results to Word or Microsoft Excel. From within Microsoft Excel, the user can take a worksheet and save it as a Microsoft Access form, report, or table.

   All of the Visual Tools allow programmatic integration with the Microsoft Office suite through their ability to call Automation servers. Each of the Microsoft Office components can read and write FoxPro 2.*x*, Visual FoxPro, Microsoft Jet, and SQL Server data by using the respective ODBC drivers. In addition, the Microsoft Office applications can read and write Microsoft Jet data directly via DAO.

3. Will the developers need to create ActiveX controls?[10]

   ActiveX controls are custom user-interface components that can be shared among multiple tools (for example, a tabbed page frame or calendar control). Visual C++ and the Visual Basic Control Creation Edition can be used to create ActiveX controls. Since ActiveX controls can be used by any of the Visual Tools, it may be useful to build ActiveX controls with one tool and develop the remainder of the application with another.

4. Will the developers need to be able to create Automation servers?[11]

   Visual C++, Visual Basic Enterprise Edition, and Visual FoxPro can be used to create Automation servers. All of the Visual Tools, as well as Microsoft Office products, can call Automation servers.

5. Will there be any speed or size requirements for the client solution?

   An important factor in determining the size and speed of an application is whether or not the solution is compiled.[12] Visual C++ allows developers to write extremely tight code and has a powerful compiler, thus allowing it to create the fastest applications. Visual Basic, Visual FoxPro, and Microsoft Access can be used to create fast applications, but since solutions created with these tools are not truly compiled, the user's computer must have enough memory to support both the custom solution and the run-time library or Visual Tool.[13]

6. How fine a degree of control will be required by the developer?

   Visual C++ development system offers the finest degree of control for solution development. This is quite understandable when one considers that the Visual Tools are themselves written in Visual C++. Of course, the more control required, the longer the solution will take to develop. Visual Basic offers the next highest degree of control, as it is designed to be a general purpose Windows development tool and not just a database tool.

   Microsoft Access and Visual FoxPro do not offer the same degree of detail

---

[10] ActiveX technologies are a critical component in the growing use of three-tiered client/server business solutions. For more information on this topic, see http://www.microsoft.com/oledev/olerem/3tierwp1.htm

[11] For more information on Automation servers, see Appendix C: Glossary.

[12] For more information on compiling, see Appendix C: Glossary.

[13] For solutions that are not compiled to run, the appropriate visual tool's run-time library must be in active memory.

as Visual C++ and Visual Basic. This is because they are first and foremost database development tools and are not optimized for general-purpose application building.

7. How much time will be available for development?

   If there is only a short amount of time available for development, Microsoft Access and Visual FoxPro are the most logical solutions. Both are data-centric tools with a number of wizards and other built-in features to help developers build database solutions. Visual Basic often requires more code to be written because it has fewer built-in features. This can be mitigated to a certain extent by using third-party ActiveX controls.

   Using Visual C++ requires the highest time commitment. Although it does have a number of helper applications and wizards, the C++ language is significantly more difficult than Visual Basic for Applications or the Visual FoxPro language.

8. Which Visual Tools are currently in use?

   Perhaps the most important rule to remember in selecting the appropriate Visual Tool is this: "Experience counts." Developers using previous versions of Microsoft Visual Tools should upgrade to the current version to benefit from the latest advances and productivity enhancements.

   If a developer is new to the Microsoft family of Visual Tools, past experience with other developer tools may point the developer towards certain tools. For example, Basic programmers will likely prefer Visual Basic or Microsoft Access, while xBASE developers will likely prefer Visual FoxPro.

   The Visual Tool with the shortest learning curve is Microsoft Access. Visual FoxPro is the next easiest to learn, followed by Visual Basic. Learning Visual C++ requires a large time investment, due to the fine degree of control it provides and the complexity of the language.

   Another important factor in choosing a Visual Tool is leverage. A developer who knows Visual Basic for Applications can develop solutions using Microsoft Access, Visual Basic, and Microsoft Office. In addition, the developer would be very familiar with Visual Basic Scripting Edition. Much code can be shared and reused among these products.

   A Visual FoxPro developer may not able to reuse her code; however, her experience with the object-oriented programming features of Visual FoxPro would lessen the learning curve required to learn object-oriented programming in Visual C++ or Visual J++.

9. Will support be required seven days a week, 24 hours a day?

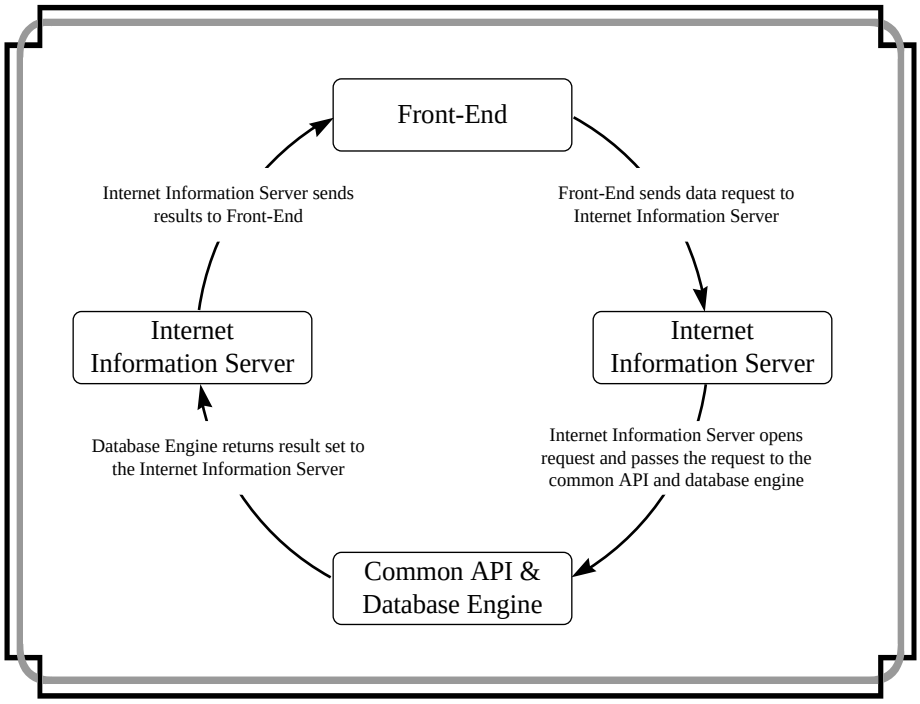
   Round-the-clock support can be purchased from Microsoft for each Microsoft Visual Tool.

Web Issues

While there are many similarities between developing database applications for traditional networks (LANs and WANs) and developing database applications for the Web, the latter requires the developer to consider additional issues when choosing the appropriate database development tools. LAN-based database tools rely on consistent connections; however, with Web database

solutions, HTTP connections are stateless.

The illustration below provides a simplified model of Internet database applications and shows how the traditional model has been altered to provide functionality over the Web. The remainder of this section will provide an overview of this model and then discuss how various Microsoft database development tools fit into this model. Finally, there will be a brief discussion of two component technologies, ActiveX and Java, and their role in database development. Several additional Microsoft tools are referred to in this section. Overviews of each of these tools are provided in Appendix B.



Model Overview

As outlined above, there are three basic tools in Web database solutions: Internet servers, database engines, and front-end tools. Database engines and front-end tools were both discussed earlier in this paper (although front-end tools earlier in this paper were limited to Visual Tools). However, Internet servers are an additional consideration for developers when creating Web database solutions.

Internet Information Server (IIS)

An essential layer of any Web database solution is an Internet server. The Internet server transfers information sent between the front-end and the database engine. Microsoft IIS provides many features to facilitate database solutions. In particular, it can use the Internet Database Connector (IDC) as an easy-to-use method of passing requests to ODBC drivers, which in turn send information to the database engine.

Another feature of IIS that helps developers to streamline their solutions is the use of the Internet Server Application Programming Interface (ISAPI). ISAPI enables Web application developers to bypass the coding procedures required by the widely used Common Gateway Interface (CGI) standard. ISAPI offers

the most efficient way to access external applications from a Web server. Rather than depend on CGI to link browsers to applications, ISAPI employs software drivers to frequently accessed BackOffice™ applications within the Web server. Visual C++ can be used to increase the power of ISAPI and provides wizards for creating ISAPI server extensions and filters. ISAPI server extensions and filters are DLLs that intercept specific server events and perform appropriate actions.

Additionally, through the use of ActiveX Server Framework[14], developers can further extend their Web database solutions. ActiveX Server Framework includes server scripting, a technology that enables IIS to invoke Automation servers, which can be written using Visual C++, Visual Basic, and Visual FoxPro.

ActiveX Server Framework is leveraged by "Internet Studio" and by Microsoft Access (through the Publish to the Web Wizard). It ships with a Database Connector server side object that enables solutions to access ODBC data sources. Furthermore, the ActiveX Server Framework Database Connector allows users to retrieve and update records on a remote database. This powerful feature can be used to greatly enhance Web database solutions and can easily be integrated into a Web application using Visual Basic Script or another active server scripting engine.

Database Engine

Under the Web paradigm, there is still the need for both a user interface and a database engine. In fact, the functionality of the database engine remains largely unaltered. The database engine still receives queries for data and returns results to the client. Nevertheless, each of the three Microsoft database engines (FoxPro, Microsoft Jet, and SQL Sever) has strengths that make it particularly strong for certain types of Web database solutions. The strengths of each engine in a traditional setting carry over to the Web database paradigm. Below, three basic Web scenarios are addressed.

- High Volume Internet Database Solutions

  As with a solution on a traditional network, SQL Server is the best tool for a solution to handle a high volume of transactions or queries over the Internet, due to its abilities in logging and rolling back transactions. Furthermore, if the developer is creating an electronic commerce solution, the SQL Server security features will make it the best database engine for the job.

- Specialized Internet Solutions

  Specialized Internet solutions are solutions that experience reduced traffic; even though a specialized Internet solution is available to the world, it is only of interest to a small audience (for example, a Web site for a local chamber of commerce). The Microsoft Jet database engine and the FoxPro engine will generally be the best choice for these solutions. They are relatively easy to administer and provide excellent performance with low to moderate numbers of users.

[14] For more information on ActiveX Server Scripting, see Appendix A: Microsoft Internet Technologies and Tools.

- Intranet Database Solutions

  Intranet database solutions are similar to LAN-based database solutions. The corporate intranet can be though of as being just another type of network. Decisions as to the appropriate engine for intranets should be made based on criteria from the "Database Engine Questions" section above.

Front-End

Microsoft provides a number of tools to create front-ends for Web database solutions. Developers will select their tools based on the needs of their users and the nature of the solution. Web-based front-end solutions can be placed into three categories: pure HTML solutions, extended network solutions, and hybrid solutions. Each of these categories is discussed briefly below, as are the tools that can be used to implement such solutions.

Pure HTML Solutions

Overview

On the Web, the user interface frequently consists of an HTML document, which is then viewed through a Web browser[15] (for example, Microsoft Internet Explorer, Netscape Navigator, and so forth). HTML documents can be created with some of the Microsoft Visual Tools or through a Web authoring tool (for example, Microsoft FrontPage™ Web authoring and management tool, Microsoft “Internet Studio”, and so on).

In a pure HTML solution, the user interface (that is, the HTML document) sends a query parameter to the Internet server (for example, Microsoft Internet Information Server); subsequently, the Internet server uses a database connector to submit the query to the database engine.[16] The database engine returns the results through the database connector to the Internet server, which, in turn, creates a new HTML document with the formatted results of the query and sends the dynamically constructed HTML document to the Web browser.

Strengths and Weaknesses

One key advantage of pure HTML solutions is that they can be accessed by anyone who has an HTML browser, and, thus, they are browser and platform independent. Since these solutions are platform independent, developers only need to develop one version of their solution, thereby greatly reducing costs. One weakness of pure HTML solutions is that HTML cannot generally duplicate all of the functionality present in traditional front-end solutions developed with any of the Visual Tools. Thus, database developers seeking to maximize reach while minimizing solution costs should use pure HTML solutions.

Choosing the Appropriate Tool

If a developer is seeking to build a basic Web solution quickly and cheaply and is already familiar with Microsoft Access or Visual FoxPro, both of these Visual Tools would be good choices for building the HTML documents. Likewise, if a

---

[15] For more information on Microsoft Internet Explorer, see Appendix A: Microsoft Internet Technologies and Tools.

[16] For more information on ActiveX Server Scripting, see Appendix A: Microsoft Internet Technologies and Tools.

developer is seeking to expand the functionality of an existing Visual FoxPro or Microsoft Access solution to the Web, in most circumstances these two Visual Tools should be used. Microsoft Access provides the Publish to the Web Wizard, which allows developers to convert their Microsoft Access solutions to HTML documents. Through this Microsoft Access wizard, database developers can either provide static HTML solutions or create dynamic solutions, using either the IDC or the Active Server Framework. Visual FoxPro converts its solutions to HTML through its Internet Search Wizard.[17]

However, if additional functionality and/or controls (for example, buttons, drop-down lists, and so forth) are required in the HTML document or the solution is not associated with an existing Microsoft Access or Visual FoxPro solution, "Internet Studio" or Microsoft FrontPage, or some combination of the two, will be the best tool for the solution. If the database developer is unfamiliar with building advanced Web solutions, as with any development tool, some time will be required to come up to speed on "Internet Studio". "Internet Studio" is especially well tuned for Web database solutions and provides an intuitive means of building solutions that access the ODBC data access component of the Active Server Framework.

Extended Network Solutions

Overview

In an extended network solution, the front-end of the solution is not associated with a browser; thus, the Web is used for two purposes: to send requests for data and to return results. The user interface is placed on the user's desktop, and query results are merely incorporated into this interface. Extended network solutions also communicate with the database engine through an Internet server.

Strengths and Weaknesses

Extended network solutions are an excellent alternative for developers who want to provide database access to a small audience, dispersed over multiple networks. Furthermore, extended network solutions offer an excellent alternative for database developers who want to expand the reach of their existing database solutions. For these types of solutions, the user requires a full, functional copy of the front-end application on their desktop computer; thus, extended network solutions are especially appropriate when users will need to use the solution on a regular basis. The drawback, of course, is that solutions cannot be constantly updated, as they can in both pure HTML and hybrid solutions, due to issues such as download time and client coordination.

Nevertheless, once on a user's desktop computer, the front-end of a extended network solution can access information extremely quickly over the Internet, since there is generally no wait time for objects such as graphics files and controls. This type of solution is an excellent alternative for intranet solutions, where the corporation controls the applications present on a user's computer. In short, extended network solutions should be used in Web database solutions

[17] Visual FoxPro also provides its own Internet server, a Visual FoxPro executable application that runs in the background. When it receives a query from the Web page through the CGI script, the Visual FoxPro Information Server runs the transmitted query and sends the results back in an HTML document.

for limited-size audiences who require data access speed and who will use the solution multiple times.

Choosing the Appropriate Tool

The choice of the appropriate tool for an extended network solution can be determined by using the "Visual Tool Questions" section above.

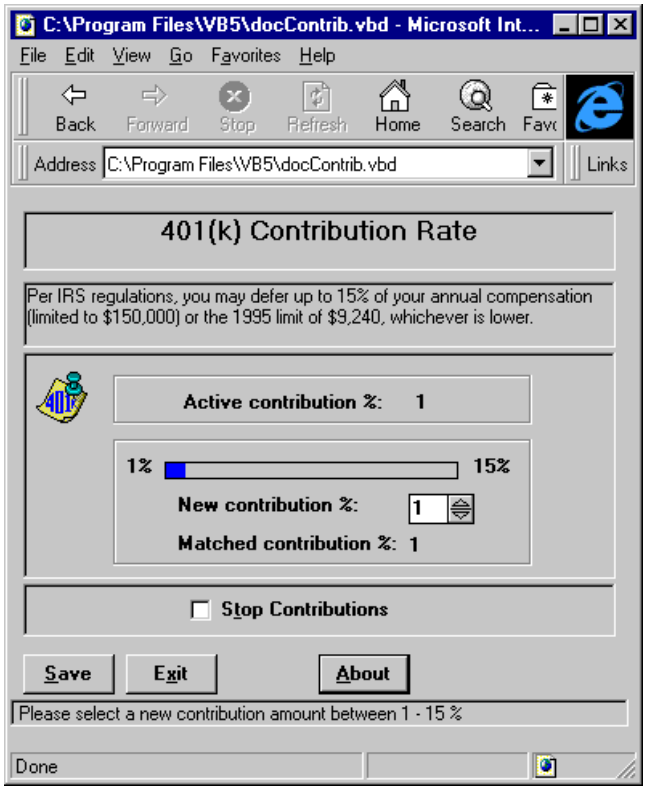
ActiveX Document Solutions

Overview

Through the use of ActiveX technologies, Microsoft has created a third powerful way to create Web solutions: ActiveX documents. ActiveX documents enable users to open files, complete with native toolbars, menus, and all other user interface elements, from within Microsoft Internet Explorer. The browser simply becomes the frame in which users can open and use richly formatted documents (such as Microsoft Excel spreadsheets and Word documents) or custom applications (such as Visual Basic solutions) right inside their Web browser.

In order for an ActiveX document such as a Microsoft Excel or Word document to run, a copy of the respective parent application must be present on the user's desktop computer. Likewise, for an ActiveX document such as a Visual Basic executable file to run, a copy of the Visual Basic run-time library must be available on the user's computer.

The screen shot below shows how the ActiveX document technology could support a Visual Basic database solution. In this solution, a Visual Basic front-end is embedded in the Microsoft Internet Explorer browser. Links are maintained with an employee database, which tracks each employee's 401K contributions.



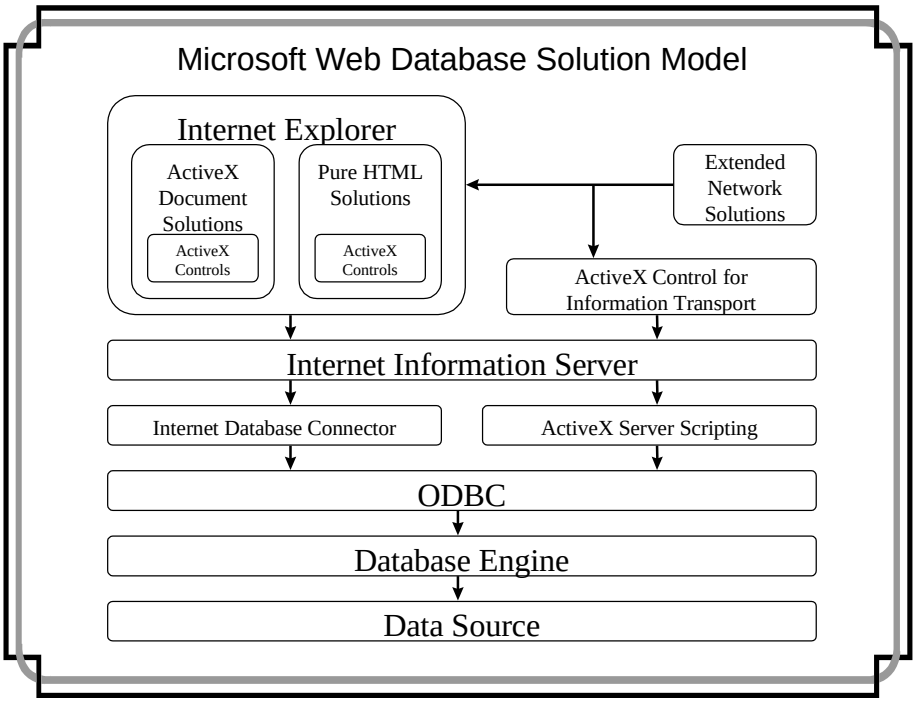
Strengths and Weaknesses

By using ActiveX document solutions, developers can create Web database solutions with all the power and flexibility of traditional network solutions. This type of solution is particularly practical on corporate intranets; in this situation, the corporation knows which applications are available to its employees and can thus ensure that everyone can utilize the solution. Furthermore, ActiveX document solutions can often offer a low cost means of providing content on an intranet, as it allows employees to post intranet solutions with tools they already know (for example, Visual Basic, Visual FoxPro, Microsoft Excel, Microsoft Word, and so forth).

ActiveX document solutions are limited by the tools available to their users. For example, without a Visual Basic run-time library on the user's computer, Visual Basic-based ActiveX documents will not run. Thus, this type of solution is not the best approach for database developers who want to reach an extremely wide, uncontrolled audience. However, ActiveX documents are an excellent choice when the audience is well defined and the database developer wants to create a highly interactive solution.

Choosing the Appropriate Tool

The choice of the appropriate tool for this type of solution can be determined by using the "Visual Tool Questions" section above.

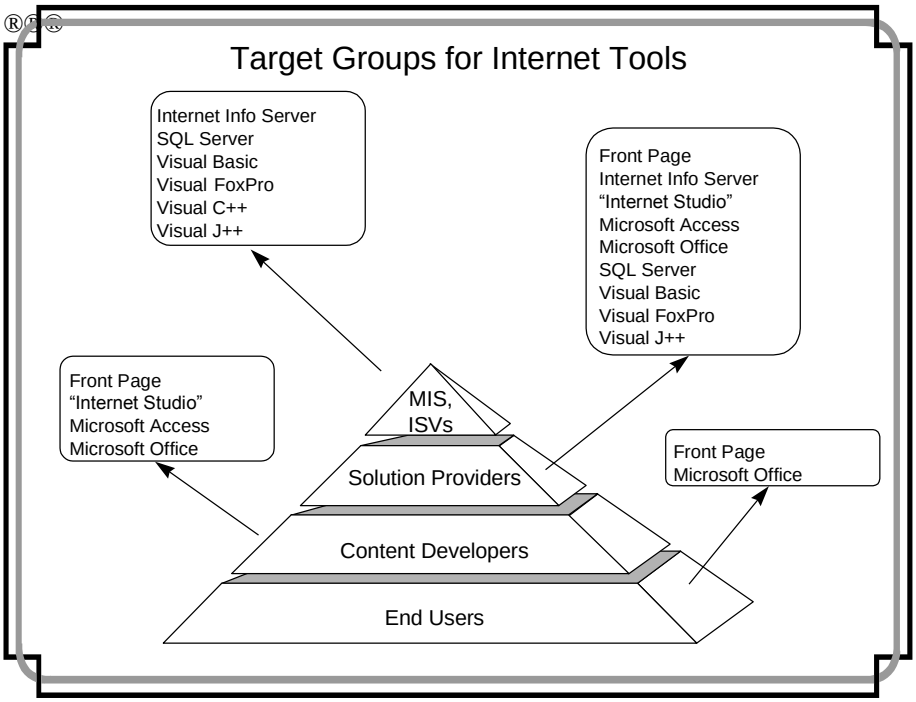


Components in Database Solutions

Adding components to Web database solutions, such as ActiveX controls and Java applets, is a means by which developers can enhance their solutions. Components can be accessed by both the front-end and Internet server in Web database applications, performing functions such as database table navigation, mathematical computation, and report design.

ActiveX controls are discussed briefly in the "Visual Tool Questions" section above. These powerful controls can be built with Visual J++ development software, Visual C++, and Visual Basic.

In addition to creating ActiveX controls, Visual J++ can be used to create Java applets. A Java applet is a Java program that can be included in an HTML page. When a user browses an HTML document with an attached Java applet, the applet's code is transferred to the user's system, compiled, and executed by the browser. Java applets will run on any platform where there is a Java Virtual Machine (VM) exposed through the browser.



Conclusion

Choosing the appropriate database developer tools can be a daunting task without taking an organized approach to the decision-making process. This paper is intended to assist the developer in that process by breaking it down into a series of smaller-scale decisions. While there are some solutions that will require features that are only supplied by one or two of the developer tools, for most solutions, choosing the appropriate tool will require the developer to weigh the importance of several solution criteria. Perhaps the most important factor is the experience and preferences of the developer. A tool is as good as the person using it, and in the hands of the experienced developer, any of the Microsoft database developer tools can be used to create powerful solutions.

Appendix A

Microsoft Internet Technologies and Tools

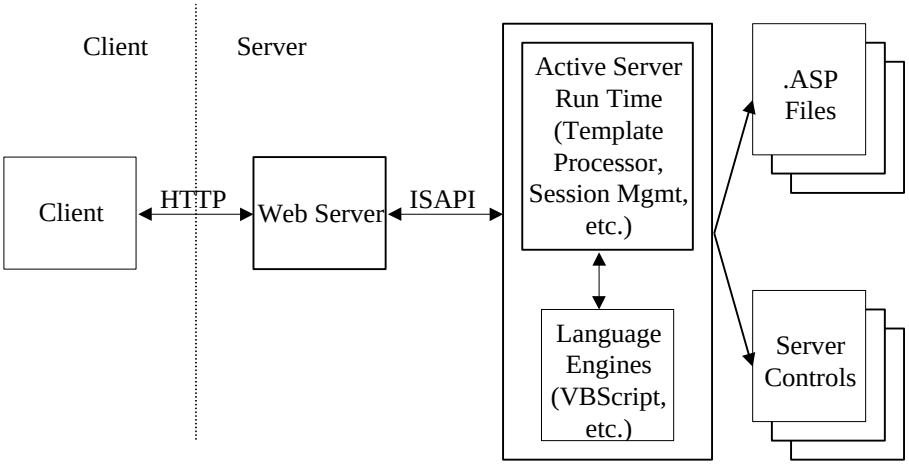
ActiveX

ActiveX is a set of technologies that enables software components to interact with one another in a networked environment, regardless of the language in which they were created. ActiveX controls are built on the Component Object Model (COM) and leverages five+ years of Microsoft investment.

ActiveX Server Framework

Microsoft ActiveX Server Framework defines a language (and its interpreter) that enhances HTML to support a limited subset of the Visual Basic Scripting language. It makes the language dynamic by adding the ability to invoke ActiveX Automation objects implemented in DLLs. A growing set of ActiveX Server objects have been defined to do many useful tasks, including handling user preferences, user authentication, database access, and conducting full-text searches.

The ActiveX Server Framework run time is built as an ISAPI application that runs on top of IIS (or on its peer Web server relative).



Client requests come in over HTTP and are dispatched to the ActiveX Server Framework run time through ISAPI. The run time knows how to process template files, including operations such as callouts to server controls, to create the resulting HTTP response (typically an HTML page) that is shipped back to the client.

FrontPage

The Web-publishing application created for nonprogrammers, the Microsoft FrontPage Web authoring and management tool gives a Webmaster everything he or she needs to get an Internet or intranet Web site up and running quickly. Its user-friendly functionality (for example, WYSIWYG editing, wizards to step through the creation of Web sites, integration with Microsoft Office, and so forth) makes it an excellent tool to create the framework for an Internet database application. The combination of flexible client/server architecture, passwords, user authentication, and other security features in FrontPage

enables contributors in different locations to securely update different pages simultaneously on the same site!

Microsoft Internet Explorer 3.0

Microsoft Internet Explorer 3.0 is the cross-platform client that delivers ActiveX technologies. Embracing both Java and Microsoft COM technology, ActiveX helps preserve existing investments in applications, tools, and source code while letting the developer create innovative Internet applications. The Microsoft Internet Explorer client is built with standard Microsoft components that have been revised to communicate to and from the Internet. These components make writing Internet applications as easy as writing applications for the operating system. Plus, these components can be reused in existing networking and stand-alone applications, making them Internet-ready.

Internet Information Server

Optimized for Web performance, Internet Information Server takes advantage of the high-performance architecture of Windows NT. Microsoft benchmark tests show IIS outperforming all other popular Windows NT-hosted Web servers. The Internet Server API (ISAPI) is designed for custom server extensions like content indexing, log analysis, data input forms, and database access. For database applications, IIS comes with the Internet Database Connector, which uses Open Database Connectivity APIs to send and retrieve information between remote data sources and the Internet. Using IDC and IIS, developers can provide anyone who is using a standard Web browser with real-time access to ODBC data sources.

ISAPI

The Internet Server Application Programming Interface is a high-performance interface to back-end applications running on a Web server. Based on its own DLL, ensuring significant performance improvements over CGI, ISAPI is easy to use, well documented, and does not require complex programming.

This API, available on Web servers from an increasing number of vendors, is quickly becoming the leading choice for Web server application development. Co-developed by Process Software and Microsoft Corporation, the Internet Server API is being endorsed by an increasing number of server developers, allowing application developers to write for a single specification and to deliver on multiple platforms.

“Internet Studio”

“Internet Studio” is a powerful development tool for building dynamic Web applications for corporate intranets and the Internet. Since “Internet Studio” is built on top of IIS, developers will be able to construct HTML "templates" that are filled dynamically with database information as users navigate the site. With its powerful database connectivity options, developers will be able to manage multiple database connections and integrate data from multiple data sources in a single page. Finally, “Internet Studio” will be tightly integrated with the other Microsoft Visual Tools so that it will be easy to integrate client and server components built with these tools within an “Internet Studio”-based Web application. For example, using “Internet Studio” and Active Server Scripting, a developer can integrate a Visual Basic or Visual FoxPro based Active Server Component that performs business logic.

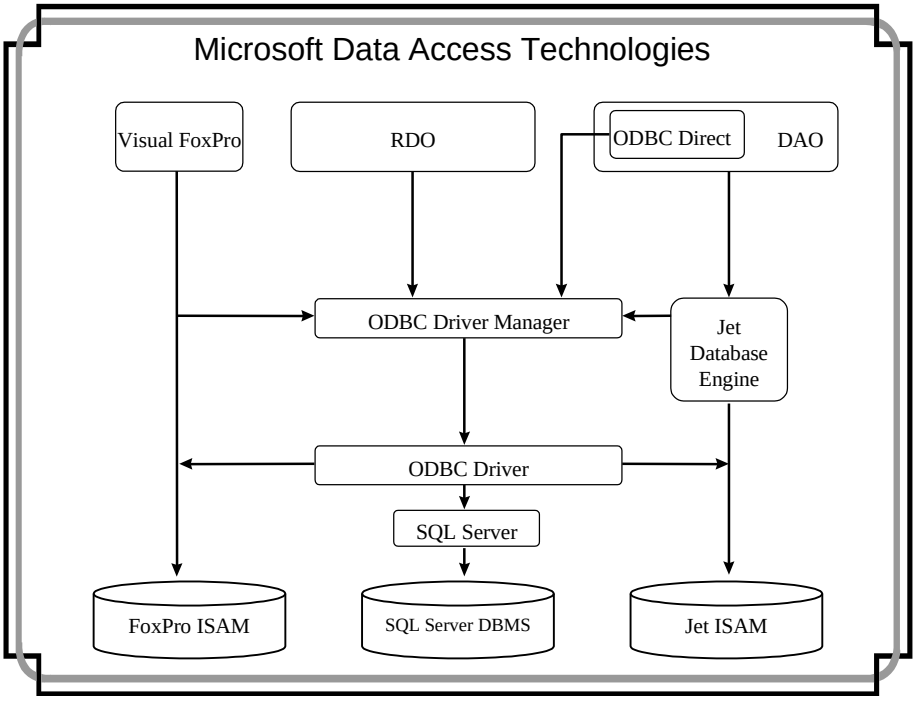
Visual J++

Visual J++ development software is the complete solution to power Java programming. With the Java compiler, developers can compile code at over a million lines a minute. A line of wizards helps developers create baseline applications in no time. In addition to its ability to create platform independent applets, Visual J++ has extensive COM support and enables developers to embed ActiveX controls in applets. Furthermore, Visual J++ provides extensive support for DAO.

Appendix B

Microsoft Data Access Technologies

Part of any database solution with Microsoft tools involves the use of one or two middle layers to translate commands from the Visual Tools to the database engine. There are several different methods for this translation, and the method used depends primarily upon the data source[18] being accessed. The following figure illustrates how the various tools and data object models talk to the three data engines discussed in this paper.



The following paragraphs will briefly discuss these methods for each of the three Microsoft database engines.

ODBC Data Sources

SQL is currently the standard for accessing data. While the SQL standard is open and supported by ANSI and ISO, there are many variations on the SQL language (for example, SQL Server, Oracle, and Visual FoxPro all have their own versions of SQL). To solve this problem, Microsoft created Open Database Connectivity. ODBC is a common API that allows applications to talk to different SQL data sources with the same syntax and has been the standard for accessing SQL databases since 1992.

For the ODBC API to talk with a vendor-specific SQL data source, an ODBC driver must be created. The ODBC driver is a library that translates ODBC commands to commands understood by the vendor-specific SQL. Currently, there are more than 170 ODBC drivers available from third-party vendors.

[18] For more information on data sources, see Appendix C: Glossary.

ODBC is a very detailed API. Because of this, Microsoft Access, Visual Basic, and Visual FoxPro can shield the user and developer from the complexities of ODBC. Visual FoxPro allows the developer to connect to an ODBC data source and then use standard Visual FoxPro data access commands. Microsoft Access and Visual Basic provide several object models[19] to hide the complexity of ODBC. Both can use Data Access Objects, which uses Microsoft Jet to talk to ODBC. In addition, Visual Basic developers can use Remote Data Objects, while Microsoft Access and Microsoft Office developers can use ODBCDirect. Each of these bypasses Microsoft Jet to talk directly to ODBC.

Microsoft Jet ISAM

The Microsoft Jet ISAM (.mdb files) can be accessed in two ways:

- Through DAO: DAO was originally created to provide access to the Microsoft Jet database engine. Currently, DAO is available through Microsoft Office (including Microsoft Access), Visual Basic, Visual C++, and Visual J++.
- Through the Microsoft Jet ODBC driver: Microsoft has created an ODBC driver that allows developers to access tables in .mdb files through ODBC. Thus, all of the methods outlined in the "SQL Data Sources" section are available to access the Microsoft Jet ISAM.

FoxPro ISAM

The FoxPro ISAM (.dbf files) can also be accessed in two ways:

- Through the SQL in Visual FoxPro: Visual FoxPro has its own language, which is used to process all of its queries and also provides its own version of SQL with the product.
- Through the Visual FoxPro ODBC driver: Microsoft has created an ODBC driver that allows developers to access tables in .dbf files through ODBC. As with Microsoft Jet, all of the methods outlined in the "SQL Data Sources" section are available to access the FoxPro ISAM.

[19] For more information on object models, see Appendix C: Glossary.

Appendix C

Glossary

Term	Definition
Automation	Formerly OLE Automation. A term for the process that enables developers to reuse objects from off-the-shelf applications in their custom solutions. This process involves three things: a series of instructions, an application sending instructions (Automation controller), and an application that responds to those instructions (Automation server).
Automation Controller	An application or development tool that can drive other applications via Automation. An Automation controller, which can be either an application or a development tool, is used to send instructions. For example, Microsoft Excel is an Automation controller that can use Visual Basic for Applications to reuse forms and reports in Microsoft Access, an Automation server.
Automation Server	An application that can be driven by other applications via Automation. An Automation server application has exposed its functionality so that other applications can access its functionality in custom solutions. For example, Microsoft Access is an Automation server that exposes its forms and reports so that they can be used from other applications.
Client/Server Database	In this model, the client sends a request to the server, and the server makes decisions as to what data will be returned to the client (that is, the intelligence of the database resides in the server).
Compile	With some Microsoft Visual Tools, it is possible to compile an application as part of development. This means the code the application was written with is converted to machine code (that is, binary-coded machine instructions in the native language of the computer). Using a compiler decreases the space used by an application and the increases the speed at which it will run.
Data Source	A data source includes the data a user wants to access and the information needed to get to that data.
Database Client	Databases consist of two parts, servers and clients. The database client makes requests to the server for data and presents and manipulates that data.
Database Engine	That part of the database system that retrieves data from and stores data in user and system databases.
Database Server	Databases consist of two parts, servers and clients. A database server contains and manages a central repository of data.

File-Server Database	In this database model, the client decides which data to retrieve and process (that is, the intelligence of the database resides in the client). In this case, the server exists for a single purpose: to store data and provide locking support.
HyperText Markup Language (HTML)	HTML is the industry standard for creating documents for the Web. HTML is simply made up of ASCII text, which is interpreted by Web browsers. Each vendor's Web browser interprets HTML slightly differently, although the interpretation among different browsers basic HTML is generally similar.
HyperText Transfer Protocol (HTTP)	HTTP is the protocol used by Web servers and Web browsers to move documents and information over the Internet.
Object Model	A database object model is a hierarchy of collections and objects. All tables, fields, indexes, queries, and so on, are represented by objects organized into collections. Each object has a set of properties that define its characteristics and one or more methods that are used to perform various operations on other objects. RDO and DAO are examples of object models.
Query	Searches through a database to retrieve or update data are referred to as queries.
Query Optimization	In some databases, when a query is sent to look for any record that has both of two text strings, "String A" and "String B," the query first goes and looks for all the records that contain String A. Next, it looks for all the records that contain Sting B. Finally, it compares the two "sets" of records and discovers which records are in both sets. Clearly, this is not the most efficient way to find records. A more efficient way to query the database is to find all the records that contain String A and then search through those records to find String B. This is what a query optimizer does. Both Microsoft Access and Visual FoxPro make use of the Rushmore query optimization technology. This technology simplifies queries by translating the query result into a bitmap. If a record matches the given query, the bit is set to 1. If the record does not match the query, the bit is set to 0.
Run-Time Library	Without a compiler, applications distributed to users must be accompanied with a run-time interpreter or library to translate commands for the computer. One such library file is a dynamic-link library, or DLL file, which contains one or more functions compiled, linked, and stored separately from the processes that use them. Once development of an application is completed, the application is usually stored as an executable file, an application that can run outside the development environment.

Scripting	A scripting language is a language that is interpreted during run time, rather than compiled. Examples of scripting languages are Visual Basic Script and Visual Basic for Applications.
Transaction	A transaction is a group of operations that cannot be partially completed without causing missing or incorrect data. (For example, at the bank ATM machine, a request simple transaction is begun. Between the beginning and the end of that transaction, there are three steps, A, B, and C. In step A, the user sends a request for $100. In step B, the bank checks the person's account and removes $100 from the user's bank account. In step C, the user is given $100. If any one step fails, the transaction will not be completed and there will be an error in the data. If the database has a logging feature, the transaction will be rolled back; that is, the database will be restored to its state before the failed transaction began.)
Visual Basic for Applications	A shared development environment that enables programmers to create solutions with one or more Microsoft Office family applications or with any third-party product that has licensed Visual Basic for Applications. Visual Basic for Applications includes the Visual Basic for Applications language engine (compiler), a powerful editor (syntax checking, color-coded syntax), Microsoft Forms, and debugging tools.

Appendix D

Database Engine Features Comparison[20]

Feature	Microsoft Jet	FoxPro	SQL Server
Heterogeneous joins	X	X	
Rushmore query optimization	X	X	
Top n and top n% queries	X	X	
Validation rules	X	X	X
Validation rules can be custom code		X	
Default values	X	X	X
Default values can be custom code		X	
Triggers and stored procedures		X	X
Referential integrity through triggers		X	X
Declarative referential integrity	X		X
Engine level cascading updates and deletes	X		
Basic locking unit	Page	Row	Page
Row locking on insert	X	X	X
Row locking on update and delete		X	
Table-level replication	X	X	X
Row-level replication	X	X	X
Field-level replication		X	X
Custom code for replication conflict resolution	X	X	X
Scheduled replication	X[21]		X
Replication over the Internet	X		
Built-in security	X		X
Built-in encryption	X		X
Transaction processing	X	X	X
Distributed transactions			X

[20] This comparison is not an exhaustive list of the features of each database engine. Its intention is to highlight some of the similarities and differences between the engines. For more information, see the sources listed earlier in the white paper.

[21] This capability requires the Microsoft Office 97, Developer Edition.

Database Engine Features Comparison (continued)

Feature	Microsoft Jet	FoxPro	SQL Server
Dynamic backup and restore			X
Back up and restore a single table			X
Transaction log backups			X
Automatic recovery			X
32-bit engine	X	X	X
Data capacity	1.2 gigabytes per database	2.1 gigabytes per table	1 terabyte per database
Maximum number of users	255	Unlimited	32,767 user connections
Hyperlink data type	X		

Visual Tool Features Comparison[22]

Feature	Microsoft Access	Microsoft Visual Basic[23]	Microsoft Visual FoxPro
Cross platform support			X
Full complement of object based design tools	X	X	X
Drag-and-drop to set relationships between tables	X		X
Highest number of ease of use features	X		
Table, Form, Report, and Query Wizards	X		X
Import Wizard	X		X
Upsizing Wizard	X		X
Create SDI and MDI forms		X	X
Drag-and-drop fields onto forms	X		X
Field mapping to specify type of control or class used when field is dropped onto form			X
Updateable query result sets	X	X	X
Color coded editor	X	X	X
Debugger includes Locals, Watch, Call Stack window	X	X	X
Watch values in debugging	X	X	X
Debugger settings can be saved for later use			X
Best visual integration with Microsoft Office	X		
Programmatic integration with Microsoft Office	X	X	X
Excellent support for ActiveX controls	X	X	X
Support for ActiveX controls that act as containers		X	X
Support for ActiveX controls that bind to a row of data	X	X	X
Support for ActiveX controls that bind to multiple rows of data		X	
Vtable binding for improved ActiveX control performance			X

[22] This comparison is not an exhaustive list of the features of each product. Its intention is to highlight some of the similarities and differences between the Visual Tools. For more information, see the sources listed earlier in the white paper.

[23] Visual Basic Enterprise Edition

Visual Tool Features Comparison (continued)

Feature	Microsoft Access	Microsoft Visual Basic	Microsoft Visual FoxPro
Early binding in code for improved Automation server performance	X	X	
Can control other Automation servers	X	X	X
Can be called natively as an Automation server	X		X
Can create custom Automation server		X	X
Remote Automation support		X	X
Component Manager to manage Automation servers		X	
Internet wizard	X		X
Object Browser	X	X	
Integration with Visual SourceSafe	X[24]	X	X
Full support for object oriented programming, including inheritance and subclassing			X
Object programming through class modules	X	X	
Create reusable code	X	X	X
Create reusable interface elements			X
Subclass ActiveX controls			X
Event tracking tool			X
Coverage logging tool		X	X
Code profiling tool		X	X

[24] This capability requires the Microsoft Office 97, Developer Edition.