

# SalesLogix



---

## SalesLogix v6 Architecture, Customization and Integration

[www.saleslogix.com](http://www.saleslogix.com)

---

December 2004

**sage**  
software  
*Your business in mind.*

**TABLE OF CONTENTS**

- Introduction ..... 3**
- Tiered Architecture Concept ..... 3**
- SalesLogix Architecture ..... 4**
  - Business Rules Security Sync Logging Business Rules ..... 4*
  - Data access layer..... 5*
  - Rules layer ..... 6*
  - Presentation layer ..... 6*
  - User interface layer..... 7*
  - SalesLogix hardware configuration..... 7*
- Data Services: Synchronization..... 8**
- Data Services: Data Security..... 9**
  - Record-level security..... 10*
  - Field-level security ..... 10*
  - Sales Client feature security ..... 10*
  - Support Client feature security ..... 11*
- Data Services: User Connection Management ..... 11**
- Customization and Integration..... 11**
  - SalesLogix Client customization ..... 11*
- SalesLogix Software Development Kit (SDK) ..... 12**
  - Front office application integration..... 13*
  - Communicating with other applications from SalesLogix..... 13*
  - Controlling SalesLogix from other applications ..... 15*
  - Integration in a Web environment..... 15*
  - Real-time integration with back office data ..... 16*
  - Batch integration with back office data ..... 16*
- Summary..... 17**
- About Sage Software..... 17**
- End Notes ..... 17**

## Introduction

Businesses today demand that all their vital data work together to serve the overall needs of the company. What were once discrete “silos” of information—for example, customer data, financials, manufacturing status—all are coming together to provide managers with views of their business that they never had before. Of course, making all these disparate sources of information work as one is no easy task. It requires IT solutions that can readily share information and be customized to take advantage of it.

SalesLogix was designed and built to provide an affordable, highly reliable, easily customizable, and quickly usable customer relationship management (CRM) solution for driving sales performance in small to mid-sized businesses. With its Web-enabled architecture, SalesLogix provides sales tools to users via networked PCs, laptops, Web browsers, and handheld devices. Data synchronization supplies the same data, Microsoft® Windows® user interface, and customizations as the networked software to disconnected users. Administration and maintenance of SalesLogix does not require a dedicated IT staff.

The SalesLogix Architecture has grown from its client/server roots to a tiered architecture. Network and Web clients for all SalesLogix® modules utilize a central database. Data access is provided by the SalesLogix OLE DB (Object Linking and Embedding Database) Provider, which enforces data security and synchronization logging.

*What were once discrete “silos” of information—for example, customer data, financials, manufacturing status—all are coming together to provide managers with views of their business that they never had before.*

## Tiered Architecture Concept

In a tiered architecture, the data and presentation are separated into distinct application layers. The user interface layer uses the underlying presentation layer to provide data viewing and manipulation services to the user. The rules layer enforces business and data rules as a service to the presentation layer. The rules layer accesses the data layer (the database) through a data access layer.

<b>USER INTERFACE LAYER</b>	End user software (Windows, Web browser, handheld device, etc.)	
<b>PRESENTATION LAYER</b>	Client application that transforms the business rules and data into code that can be displayed to the user	Web server that transforms the business rules and data into HTML, DHTML, WML XML, etc. that can be displayed to the user
<b>RULES LAYER</b>	Business logic and data services to transform raw data into useful information	
<b>DATA ACCESS LAYER</b>	Data input/output to the database	
<b>DATA LAYER</b>	Data Storage (relational database) and query and performance optimization (indexing, etc.)	

**Table 1. Tiered architecture**

This model makes applications more scalable. Instead of each client application consuming resources to access the data layer directly, clients communicate with the rules layer. The rules layer can support many clients, thereby reducing resource consumption and improving scalability. Data services such as connection pooling,

translation, security, and application logging can be shared across the client population.<sup>1</sup>

## SalesLogix Architecture

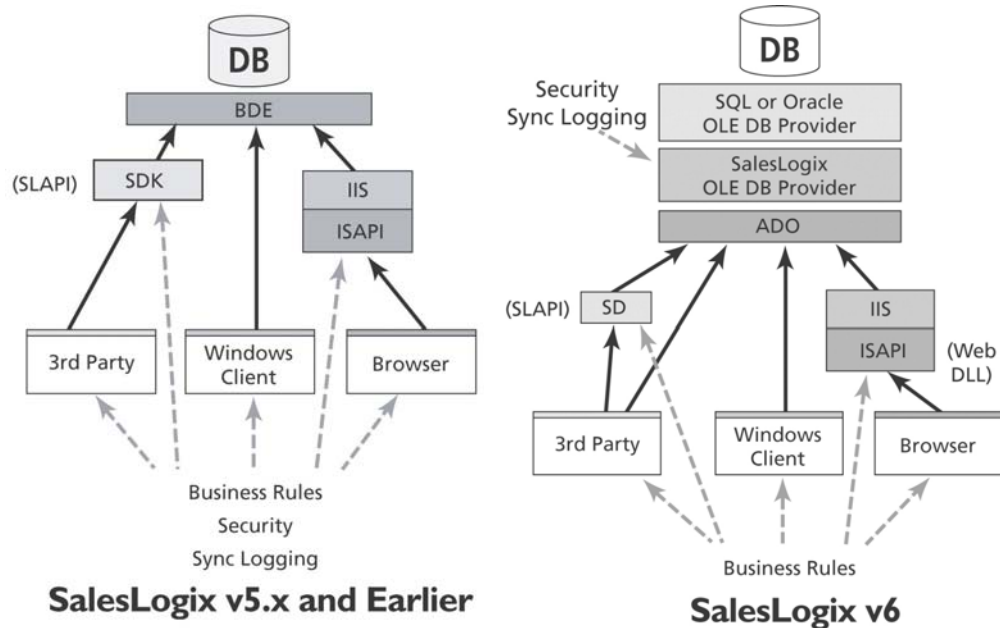
Before version 6.0, the SalesLogix application was a two-tiered client/server application. With two-tiered applications, the client process handles the presentation and business logic, including business rules, data logging for synchronization, and user security. Data can be stored on a separate database server or on the same machine as the client for offline (remote) access. The client application connects directly to the data source for the lifetime of the application. Data access is through the Borland Database Engine (BDE) or Open Database Connectivity (ODBC). Customization requires an Application Programming Interface (API) layer to enforce the business logic, security, and sync logging for remotes.

The v6 release brings a tiered architecture to SalesLogix.

*The v6 release brings a tiered architecture to SalesLogix.*

### Business Rules Security Sync Logging Business Rules

Figure 1. Before and After



**Table 2. The SalesLogix Tiered Architecture**

<b>USER INTERFACE LAYER</b>		SalesLogix Windows Client, Web Clients, Web Phones, Handheld Devices		Custom Interfaces
<b>PRESENTATION LAYER</b>		SalesLogix Windows Clients Client application that transforms the business rules and data into code that can be displayed to the user	SalesLogix Web Host Web server that transforms the business rules and data into HTML, DHTML, WML, XML, etc. that can be displayed to the user	Third-Party applications utilizing ADO
<b>RULES LAYER</b>	BUSINESS RULES	SalesLogix OLE DB Provider (Data rules for synchronization, security, and user connection services)		
	DATA RULES			
<b>DATA ACCESS LAYER</b>		SQL Server or Oracle® OLE DB Provider		
<b>DATA LAYER</b>		Microsoft SQL Server or MSDE database Oracle database		

The data layer is the relational database layer. SalesLogix supports Microsoft SQL Server including Microsoft Data Engine (MSDE) and Oracle relational databases.

### Data access layer

The basic data access layer is the native database OLE DB Provider. The Microsoft OLE DB Provider for SQL Server allows ActiveX® Data Object (ADO) to access Microsoft SQL Server. The OLE DB Provider for Oracle allows ADO to access an Oracle database.

The SalesLogix OLE DB Provider extends the functionality of the native database OLE DB Provider by implementing extended interfaces that are not natively supported by the database. The SalesLogix OLE DB Provider allows ADO to access SalesLogix data without the need for a proprietary API.

The SalesLogix OLE DB Provider (technically, a service provider) adds three services to the native database OLE DB (Data) Provider, which are not natively supported. These services are internal to the SalesLogix OLE DB Provider and used automatically via the standard ADO interface—no additional or proprietary SalesLogix ADO methods are required. These services bridge the data and presentation layers, and contain some rules layer functions. The SalesLogix OLE DB Provider automatically:

- Performs data logging, and creates transaction exchange files (TEFs) for synchronization of data to remote users;
- Enforces field- and record-level data security on all queries passed to the database; and
- Manages user connections to the database and enforces licensing.

The services are implemented as separately compiled components that can be swapped in and out in the future. This offers great flexibility, enabling changes to the configuration by adding and removing services or by replacing a service with one implemented by an authorized third party, for example. This also helps move more data logic out of the presentation layer (the clients) and into the rules layer. Services can be located on the same machine or distributed across different machines.

The SalesLogix OLE DB Provider is not limited to SalesLogix Clients. Microsoft Visual Basic®, Application Service Provider (ASP) pages, Crystal Reports®, or any other

*The SalesLogix OLE DB Provider allows ADO to access SalesLogix data without the need for a proprietary API.*

application that uses a standard OLE DB connection may access SalesLogix data as a client. The interface is standard Microsoft ADO, requiring no proprietary API calls to access the data.

These data services enforce the data rules and can be considered part of the rules layer, as described in the next section. Each of these services will be discussed in more detail in later sections.

Connection management is an important aspect of the new architecture. In older versions of SalesLogix, each instance of the client created a connection to the database upon user login and held that connection, whether it was being used or not, until the user logged out. This restricted scalability because only a limited number of connections to the database could be made at any one time.

With SalesLogix v6.0, a pool of connections is available for use on the SalesLogix Server. For each client query, a connection is used from the pool and released when execution of the query is complete. Clients do not hold multiple database connections open while the user is logged in. This improves scalability because typical use involves a salesperson logging in, executing one or two queries to retrieve needed information, and then staying logged in while performing some other activity, such as talking with a client on the phone while referring to the information just retrieved, but not making any more queries.<sup>2</sup>

## Rules layer

Windows clients and the Web host handle business rules. They also control all logic that deals with the “flow” of data, for example, “If A happens, then do B, C, and D.”

Data rules are handled by the SalesLogix OLE DB Provider, which spans the rules and data access layers. Data rules include the logic required for synchronization, field and record security, and user connections and licensing. The SalesLogix OLE DB Provider controls access to the database and uses the native database data providers (SQL Server or Oracle) to access the data layer.

## Presentation layer

The SalesLogix Sales and Support Clients have a unique presentation layer. The user interface views, basic scripts, reports, document templates, and all other customizable aspects of the clients are stored as objects in the database and retrieved as needed. The clients retrieve these objects from the database or from cache and display them as the user interface. A key benefit of this approach is that remote, disconnected users have exactly the same user interface as network users.

Security, data logging for synchronization, and connection management are moved out of the presentation layer, into the SalesLogix OLE DB Provider. This gives SalesLogix a more flexible and organized architecture.

Using SalesLogix Architect, the objects, or plug-ins, can be easily customized to fit the needs of the customer or individual user. Additionally, the plug-ins are part of the database so they synchronize to remote users automatically—no need to deploy and install each customization update to the clients. See “Customization and Integration” section of this paper for more information about customization.

*Using SalesLogix Architect, the objects, or plug-ins, can be easily customized to fit the needs of the customer or individual user.*

The presentation layer of all SalesLogix Web products has two parts: the template entry in the database and the actual DHTML or HTML pages. The Web Manager stores a list of templates in the database that are essentially pointers to the actual Web pages. The templates contain special SalesLogix tags and are referenced through the SalesLogix Web Host to enable security and hypertext link resolution, as well as to manage previous versions.

Because templates are HTML files, they contain tags that serve as commands specifying how part or all of an HTML page should be structured. Normal HTML tags can be used to perform simple tasks, such as italicizing a word or phrase. More complex actions, such as some database interactions, require custom tags. SalesLogix supports not only standard HTML tags, but also 13 custom tags that request information for the Web Client or other Web products. Custom tags are compatible with HTML 1.0 and higher.

When a Web client calls a URL, the Web Host uses the URL to find the raw template page, reads the SalesLogix tags in the template, and replaces the tags with live data, which is sent to the browser as DHTML and HTML pages. The Web Host is implemented as an Internet Information Server (IIS) Internet Server Application Programming Interface (ISAPI) interface (Web DLL).

## User interface layer

SalesLogix has four primary clients that serve as the user interface.

- **SalesLogix Sales Client** is a Windows application that provides a customizable interface for sales and marketing functions.
- **SalesLogix Support Client** is a Windows application that provides a customizable interface for customer service and technical support functions.
- **SalesLogix Sales Web Client** is a customizable browser-based interface for sales functions.
- **SalesLogix WebTicket** is a customizable browser-based interface for customer support functions.

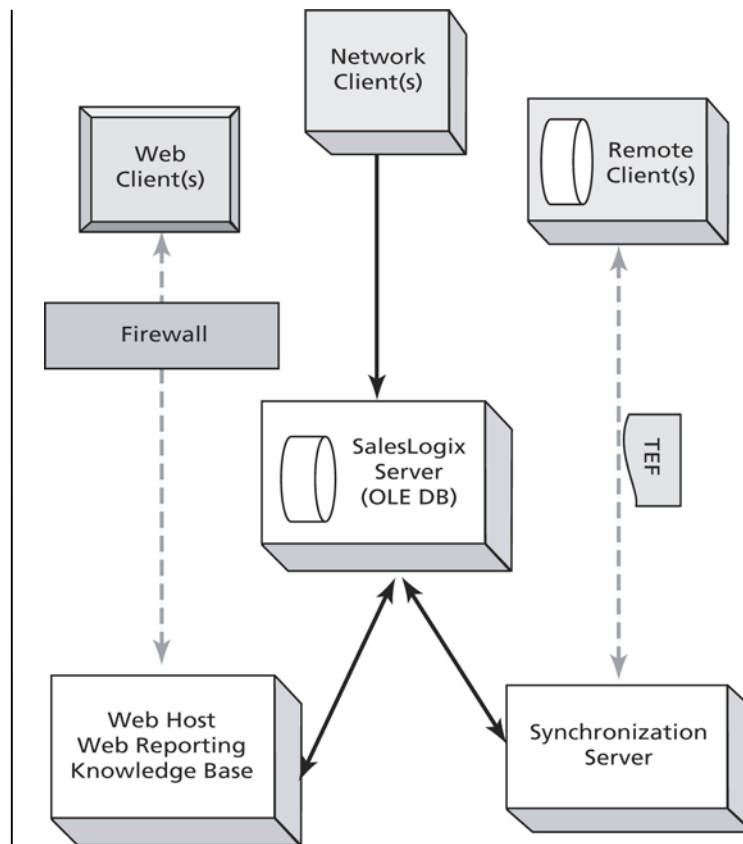
In addition, there are several browser-based interfaces to specific functions. These include a SalesLogix LeadCapture form to submit contact information from a public website directly to SalesLogix; and a Web phone interface that provides contact, calendar, and activity information on an Internet-enabled phone.

Using SalesLogix Architect and Web Manager, most of the user interface can be easily customized to fit the needs of the customer or individual user. Read more information about customization in the “Customization and Integration” section of this paper.

## SalesLogix hardware configuration

The SalesLogix architecture works with many hardware configurations. For example, the various SalesLogix servers can be housed on separate computers or combined as required by the customer. The following diagram represents a basic network configuration to support several users and all SalesLogix components. Refer to the SalesLogix Planning Guide and SalesLogix Implementation Guide for hardware details and recommendations based on specific requirements.

*Using SalesLogix Architect and Web Manager, most of the user interface can be easily customized to fit the needs of the customer or individual user.*



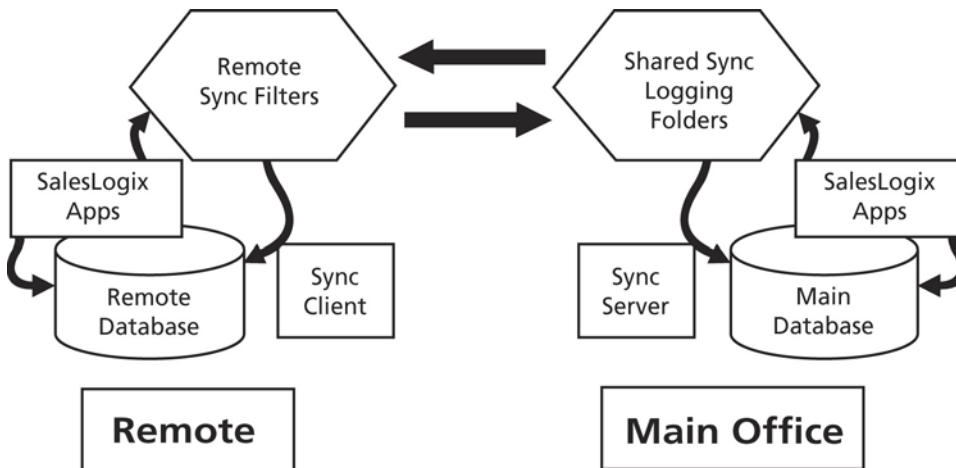
**Figure 2. A typical SalesLogix hardware configuration.**

*Remote users work with their individual databases on their own machines. Remote Offices can support network users who connect directly to the Remote Office database.*

## Data Services: Synchronization

Data synchronization enables remote salespeople to access up-to-date corporate data on their "disconnected" mobile devices. The main office contains the host database, the central set of shared synchronization folders, and the SalesLogix Synchronization Server. It supports all users who connect to the host database.

SalesLogix remotes include both Remote users and Remote Offices. Remote users work with their individual databases on their own machines. Remote Offices can support network users who connect directly to the Remote Office database. All remotes (users and offices) have a subset of synchronization folders and either the Synchronization Client or the Remote Office Synchronization Server.



**Figure 3. The Synchronization Process**

As each user makes changes to his or her database, SalesLogix tracks the field-level changes and stores them in transaction exchange files (TEFs). SalesLogix tracks the time and date of the change, the user and site that performed the change, and other details. Tracking field-level details about each transaction makes possible sophisticated conflict resolution, efficient processing, and data rollback so erroneous updates can be undone.

Unlike synchronization systems based on index scanning or stored procedure schemes, SalesLogix logging makes it possible for updates to be applied across different database platforms. Transaction log files generated by a SQL Server or MSDE database can be applied to an Oracle database with no conversion required.

The SalesLogix Synchronization Client for the Remote user or Remote Office transfers TEFs to the shared Sync Logging folders at the Main Office. It also moves files from the shared Sync Logging folders to the remote synchronization folders. These TEFs are then applied to the remote database, so it contains the same information as the host database. Note that the remotes never connect directly to the host database during this process, but only to the shared Sync Logging folders at the Main Office.

The SalesLogix Synchronization Server at the main office processes the TEFs received from the Remote user or Remote Office and applies them to the host database. If other remotes also need this updated information, it generates TEFs for them. Additionally, the Synchronization Server processes the TEFs generated by users at the main office, creating TEFs for the Remote users or Remote Offices that require those updates. This ensures that all the databases eventually contain the same information as the host database.

## Data Services: Data Security

SalesLogix provides a level of data security not available in the underlying database. The SalesLogix OLE DB Provider enforces this level of security.

*Unlike synchronization systems based on index scanning or stored procedure schemes, SalesLogix logging makes it possible for updates to be applied across different database platforms.*

*Account ownership defines which users have access to an account and the contacts, opportunities, and other information associated with that account.*

## Record-level security

SalesLogix record-level security is based on the concept of account ownership. Account ownership defines which users have access to an account and the contacts, opportunities, and other information associated with that account. There are four types of account ownership:

- **Everyone ownership.** No security profile prevails. As a result, any user in the system may view, edit, or delete information pertaining to the account, or even delete the account itself.
- **Everyone (View Only) ownership.** All users have read-only privileges to the account.
- **Individual user ownership.** Security is based on individuals' profile security settings. Personal contacts are private and cannot be accessed by others. Personal contacts can be made public, but public contacts cannot be made personal.
- **Team ownership.** A team is a group of users who have access to the same accounts. Some team members are designated as team owners and can change account ownerships. All other users on the team have designated configurable security profiles. Each security profile specifies what kind of access each user has to the data, such as read/write authority or read-only. Users can belong to more than one team.

## Field-level security

Field-level security controls data access at the table or field level. Field-level security can be enabled when a table is created through the SalesLogix Database Manager. One-to-many tables do not allow for field-level security.

Security profiles are managed through the Security Profile Manager. When a user is added to a team, the user's default field-level security profile is used. Once added to a team, the user's field-level security profile for the team-owned accounts can be changed. Changing a user's field-level security profile in one team does not affect that user's default field-level security profile in any other team to which they belong. Members of the same team may have different profiles. The no access, read only, or read/write access rights may be applied to the fields in a table that has field-level security enabled. One table can contain the same access for all fields. Individual fields can also be assigned different access rights.

## Sales Client feature security

The SalesLogix Sales Client permits the administrator to further restrict users' ability to employ specific features of the software:

- Addition and deletion of accounts, contacts, and opportunities can be restricted. By default, addition and deletion are permitted.
- Users must have security rights to accounts, contacts, or opportunities to have add or delete rights.
- The add-and-delete rights set in Feature Security control menu access rights.
- Users can restrict menu and toolbar functions.

SalesLogix enables users to manage access and calendar permissions to other users' calendars. By default, all users have access to all other users' calendars and can view activities in single-user or multi-user views. The following rights can be set:

- **Add.** The user can schedule calendar items.
- **Edit.** The user can modify scheduled calendar items.
- **Delete.** The user can delete scheduled calendar items.
- **Sync.** Sends the user all activities for the user selected on the Calendar Control list. If cleared, the user can still send activities but won't receive all activities for that user.

### Support Client feature security

The SalesLogix Support Client permits the administrator to further restrict functions the user can perform. Support Client feature security rights and restrictions include:

- The ability to restrict whether a user can add, delete, and edit accounts, contacts, tickets, defects, return merchandise authorizations (RMAs), and procedures. The default setting is to allow insertion and deletion. If the user can add records, editing and viewing is permitted.
- Users can restrict lookup functions.
- Users can restrict menu and toolbar functions.

*The SalesLogix Support Client permits the administrator to further restrict functions the user can perform.*

## Data Services: User Connection Management

User connection management is implemented as a data service as part of the SalesLogix OLE DB Provider. This service enforces licensing to determine who can log on to SalesLogix and who is actively connected to SalesLogix.

This service improves product licensing flexibility and management. One new feature enables the administrator to view the list of individuals logged onto SalesLogix.

## Customization and Integration

### SalesLogix Client customization

SalesLogix Architect is a Visual Basic-like development environment. It is used to build, edit, and manage plug-ins. Plug-ins are objects for customizing standard functionality. They are also used to add custom functionality to SalesLogix. Plug-ins are used to create detailed procedures and to customize the appearance of screens in SalesLogix, as follows:

- **Create forms.** A form is a user interface that captures or displays information. Existing forms can be modified and custom forms created.
- **Create Basic scripts.** Basic scripts can be used to string together a series of Basic functions and commands used by forms and reports, or

called from menu or toolbar items. Basic scripts can be created to perform specific functions in SalesLogix.

- **Build Sales Client processes.** Processes are used to execute a sequence of tasks over a set time period. Processes can be built to perform repetitive tasks or to implement sales processes.
- **Customize Sales Client menus and toolbars.** Custom menus and toolbars can be created and new items added to existing menus and toolbars.
- **Create or modify reports.** The reports included with SalesLogix can be modified or new reports can be created using Crystal Reports. Reports are based on the standard tables (Contact, Account, Ticket, Defect, and so on) or on any custom table in the database.
- **Build templates.** Mail Merge templates for letters, memos, and fax cover sheets are included with SalesLogix. New templates can be created or the existing ones edited.
- **Create SQL scripts.** SQL scripts can be created to perform specific functions used by views, and reports, or called from menu or toolbar items.

The Architect also includes features for building components used by plug-ins, such as pick lists and calculated fields. The Database Manager enables you to extend the SalesLogix database schema to incorporate custom tables that are automatically synchronized to remote users. The Join Manager and Lookup Manager help manage table relationships and searches.

## SalesLogix Software Development Kit (SDK)

To facilitate interaction between SalesLogix and third-party applications, developers can use the standard ADO interface to retrieve and update SalesLogix data.

ADO accesses the database via the SalesLogix OLE DB Provider, which performs data logging and security functions that required a proprietary API with older versions of SalesLogix.

The SalesLogix application programming interface (SLAPI) is also available. In most cases, proprietary SLAPI is no longer required, but it remains for backwards compatibility. SLAPI is a set of high-level functions that enable other applications to interact with the SalesLogix database. These functions include database access, table access, table functions, row functions, logging, and miscellaneous functions that provide information from the system.

With the SDK you can easily add seamless integration from outside the SalesLogix clients. The SDK includes a set of sample programs that aid access to the underlying SalesLogix database from various languages outside the system.

Part of the SDK, the SalesLogix Developer's Reference manual provides functions used to customize SalesLogix to meet specific needs. It includes information about using Visual Basic (VB) and SQL scripts to customize and modify SalesLogix. Basic functions that work with SalesLogix, with examples, are included for reference. The SalesLogix Architect is used to build and save your customized scripts and functions.

*To facilitate interaction between SalesLogix and third-party applications, developers can use the standard ADO interface to retrieve and update SalesLogix data.*

The SalesLogix Web application programming interface (WAPI) is a set of routines, protocols, and tools for building or manipulating software applications. SLAPI is not supported on the Web, thus WAPI is used to provide quick and easy access to data, to create Web business rules for Web Client users, and to perform data validation or manipulation. SalesLogix offers 75 WAPI functions to aid in customization of Web products. Knowledge of basic HTML and Basic scripting are prerequisites for Web application customization.

Actions are Basic scripts that provide quick and easy access to SalesLogix data. They are server-side scripting tools using a Basic-like language that includes special extensions for interactions with the SalesLogix database and Web Client. Use actions to perform data validation or manipulation against information from the SalesLogix database or posted via HTML forms. You can also use actions to create rules for fields and forms and to serve as business rules for Web Client users.

Additionally, Active Server Pages (ASP, ASP.Net) and similar Web technology that supports ADO may use the SalesLogix OLE DB Provider without the need for the proprietary API.

### Front office application integration

Because SalesLogix is used as the primary front office solution in many implementations, integrating SalesLogix data with other systems is a frequent necessity. As systems are more prevalently used to aid interactions with customers, full-featured integrations between these systems is needed to avoid data inconsistencies and other anomalies that can affect the overall customer service experience.

Real-time integration involves passing data from another client application or database to the SalesLogix application, and vice versa, while user interactions with the system are taking place.

To facilitate the most efficient use of the systems, this type of integration can also require the manipulation of the SalesLogix and/or OEM custom applications in response to user actions.

Disconnected users require data synchronization. SalesLogix provides a robust synchronization scheme to handle this data transfer, as described in "Data Services: Synchronization" section. It uses ADO to interface to the SalesLogix OLE DB Provider, handling synchronization data logging automatically, with no additional development effort.

### Communicating with other applications from SalesLogix

Depending on the availability and feature set of an open interface, integration from SalesLogix to another application is straightforward.

If the interfaces in the application support Component Object Model (COM), then object manipulation can be developed and executed with VBA extensions (VBScripts) in SalesLogix, enabling communication with the application. These scripts are executed without any user interruption.

Alternatively, if the application interfaces support a scripting environment that can read the Windows registry, then registry keys and values can be manipulated to pass data from SalesLogix to the application.

*Real-time integration involves passing data from another client application or database to the SalesLogix application, and vice versa, while user interactions with the system are taking place.*

*SalesLogix uses Microsoft ADO for data access and has the ability to read and write data to many back-end platforms (such as Oracle, Sybase, DB2) through custom VBScripts when the appropriate driver is present.*

If the application has neither a COM interface nor a scripting environment, manipulation can be accomplished through command line parameters. This is a limited method of communication, however, because communication to the called application must be done by loading an executable and data cannot be passed to the application while it is open.

If the application does not support any external interfaces or command line parameters, or if direct database manipulation is desired or required (for example, in cases where there is no client application), a few options must be considered:

- SalesLogix uses Microsoft ADO for data access and has the ability to read and write data to many back-end platforms (such as Oracle, Sybase®, DB2) through custom VBScripts when the appropriate driver is present. This functionality allows programmers to design “behind the scenes” application integrations that will manipulate data in another database based on actions the user takes in SalesLogix. The opposite can also be accomplished. For example:
- A user adds customer data with a certain flag to SalesLogix. A custom script can be triggered in the SalesLogix client to verify the data against another back-end database. If verification is obtained, then another call can be made to the database to find additional information about the newly added customer, such as his or her credit limit. This additional data can be written to the SalesLogix database. If verification is not obtained, the user can be halted from adding the data.

The limitation of this example is that the bound data objects in SalesLogix cannot be bound to data that resides outside of the SalesLogix database. Integration must occur completely through ADO and custom scripts, and, although it is possible to populate some controls (data grids are a notable exception) in the SalesLogix user interface with data obtained by scripts that query another database, this adds more complexity to the project.

- A custom application can be developed to act as an interface to the application database in lieu of the SalesLogix Client. The manipulation of this application can occur through COM object manipulation, registry value settings, or command line parameters as described above. If this custom application is going to write data to the SalesLogix database, it must use ADO to read and write data. If ADO is not supported, SLAPI must be used. This is required to notify the synchronization engine that database changes have occurred. Otherwise, Remote users will not receive data changes made by the custom application.
- A Web-based application can be developed to act as an interface to the application database. A browser control can be displayed in the SalesLogix user interface to display the information. This requires an Internet connection and does not serve disconnected users.
- A custom ActiveX control can be developed to act as an interface to the application database. The control, when properly implemented, can be used much as a native control within the SalesLogix Client user interface. This gives extreme flexibility but is more complex to develop.

## Controlling SalesLogix from other applications

In addition to communication, application integration can also involve the control of one application by another. When invoked by another application, the SLX Windows Clients support this type of integration by two methods: COM and registry value passing.

**Control via COM.** The SalesLogix Sales and Support Clients each support a limited COM interface. The following functionality is supported in the COM interface, with slight differences between each client interface:

- System menu functions can be executed.
- Custom basic scripts, which can trigger several events to occur, can be executed. This, in essence, can extend the functionality supported by the COM interface.
- Data for the global variable area can be set and retrieved.
- System information can be retrieved.

**Control via the Windows Registry.** Because SalesLogix includes a scripting environment capable of reading from and writing to the Windows registry, the registry can be used to pass information to the SalesLogix Client. Events can be used to trigger custom scripts that read these values, thus creating the illusion of seamless real-time integration.

Any manipulation of SalesLogix data by another application must use the SalesLogix OLE DB Provider or the SalesLogix API. This is required to notify the synchronization engine that database changes have occurred. Otherwise, Remote users will not receive changes to data made by this application.

## Integration in a Web environment

Because the SalesLogix Web Client is accessed from a Web browser, integration with other client applications through SalesLogix Web customizations is limited. As a result, SalesLogix is most often integrated with Web-enabled applications or applications that can be Web-enabled via Active Server Pages (ASP, ASP.Net). Depending on the availability and feature set of an open interface, integration from SalesLogix to another application is relatively straightforward.

If the interfaces in the application support COM and can be embedded in a Web browser, then object manipulation can be developed and executed using active server page files that are called from within the HTML interfaces of the SalesLogix Web Client.

If the application being integrated supports URL parameter passing, then the SalesLogix Web Client can be customized to call the application with the appropriate parameters embedded in the URL. In this scenario, the SalesLogix application should invoke the client application using an embedded URL call within JavaScript or VBScript. This allows the called application to manipulate the SalesLogix application through embedded URL statements, enabling bi-directional integration.

The SalesLogix Web Client supports embedded parameters in URL statements as a method of integration at the workstation. Web-based custom applications can also be written with active server pages using ADO to read and write to the SalesLogix

*In addition to communication, application integration can also involve the control of one application by another.*

*Real-time back office integration involves the movement of data between two databases on a real-time (or near real-time) basis. As data is added to one system, a mechanism is invoked that causes this data to be inserted or moved into another database.*

database via the OLE DB Provider. The SalesLogix OLE DB Provider informs the synchronization engine that database changes have occurred so that changes to SalesLogix data made by the application are communicated to the host database.

### **Real-time integration with back office data**

Real-time back office integration involves the movement of data between two databases on a real-time (or near real-time) basis. As data is added to one system, a mechanism is invoked that causes this data to be inserted or moved into another database. Batch back office integration involves passing data to another database from the SalesLogix database and vice versa on a scheduled basis. This type of integration is transparent to users because it does not affect the application environment.

Real-time data migration between a SalesLogix host database and most other databases can be accomplished by creating a custom application to move data. The custom application runs constantly, watching for changes in data. The changes can most easily be trapped at the database level by writing changes to shadow tables that the custom application constantly queries. The actual data tables should not be constantly queried because this adds tremendous overhead to the database server. When the custom application finds new data, it transfers the data between the systems.

Any manipulation of SalesLogix data by a custom data migration application must use the SalesLogix OLE DB Provider or the SalesLogix API. This is required to notify the synchronization engine that database changes have occurred. Otherwise, Remote users will not receive changes to SalesLogix data made by the custom application.

### **Batch integration with back office data**

Data migration between SalesLogix and most other databases can be accomplished as a batch process by creating a custom application to move data on a scheduled basis. Because Remote users are present, all database changes in SalesLogix must be written through the SalesLogix OLE DB Provider or the SalesLogix API. This is required to notify the synchronization engine that database changes have occurred. Otherwise, Remote users will not receive changes to SalesLogix data made by the custom application. The custom application can be written in any language that supports function calls in a 32-bit Windows DLL. SalesLogix provides headers for the DLL functions for Visual Basic, C++, and Delphi.

## Summary

Increased demand for business applications to work together has spurred many solution providers to find ways for their products to share and manipulate one another's data. SalesLogix has long been recognized as a leader in such integration and customization. With the release of SalesLogix v6.0, the product's ability to communicate with other applications and to be controlled by them has been enhanced.

This has been accomplished through:

- The adoption of a tiered architecture;
- The lack of a requirement for proprietary APIs to access data;
- Flexible security settings;
- Efficient handling of user connections; and
- A variety of communication and control methods.

Virtually every major system implementation requires customization. The ease with which customizations can be made is an important consideration when choosing which business solution to purchase. Generally speaking, the more adaptable the solution is to the business's overall operation, the better it will meet the organization's needs.

*The ease with which customizations can be made is an important consideration when choosing which business solution to purchase.*

## About Sage Software

Sage Software offers leading business management products and services that give more than 2.3 million small and mid-sized customers in North America the insight for success throughout the lives of their businesses. Its parent company, The Sage Group plc (London: SGE.L), supports 4.3 million customers worldwide. For more than 25 years, Sage Software has delivered easy-to-use, scalable and customizable applications through its portfolio of leading brands, including Abra, ACCPAC, ACT!, BusinessVision, CPASoftware, FAS, MAS 90, MAS 200, MAS 500, MIP, Peachtree, SalesLogix, Timberline, among many others. For more information, please visit the Web site at [www.sagesoftware.com/moreinfo](http://www.sagesoftware.com/moreinfo) or call (866) 308-2378.

## End Notes

For more on tiered architecture, see the article *Application Architecture: An N-Tier Approach*, by Robert Chartier found on [www.15seconds.com](http://www.15seconds.com). Some of the information presented here is based on this article.

For background information on ADO and OLE DB Providers, visit: <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/ado270/hm/mdconserviceproviders.asp>

For details on synchronization, refer to the SalesLogix technical white paper *SalesLogix Data Synchronization Technology for Mobile CRM Users*.



## SalesLogix

8800 North Gainey Center Drive, Suite 200  
Scottsdale, AZ 85258  
800-643-6400  
[www.saleslogix.com](http://www.saleslogix.com)

The information contained in this document represents the current view of Sage Software, Inc. on the issues discussed as of the date this document was prepared. Sage Software cannot guarantee the accuracy of any information presented after the date of publication. The capabilities, system requirements and/or compatibility with third-party products described herein are subject to change without notice. Contact Sage Software for the most current information. Always consult a network specialist to discuss the security risks involved before implementing any Internet solution. Sage Software is not responsible for the content or maintenance of third-party Web sites referred to herein. This document is for informational purposes only and may not be distributed to third parties. SAGE SOFTWARE MAKES NO WARRANTIES, EXPRESSED OR IMPLIED, IN THIS DOCUMENT.

© 2004 Sage Software, Inc. All rights reserved. Reproduction in whole or in part without permission is prohibited. Sage Software and the Sage Software product and service names mentioned herein are registered trademarks or trademarks of Sage Software, Inc. and/or its affiliated entities. Platinum is a registered trademark licensed from Platinum Technology International, Inc. BatchMaster is a trademark licensed from eWorkplace Solutions, Inc. All other trademarks are the property of their respective owners.

12/04 04-3000