

# Integrating Windows<sup>TM</sup> Applications for the Enterprise

---

An AppSwing<sup>TM</sup> White Paper

**AppSwing**   
Business where you want <sup>it</sup>

AppSwing Limited  
Atlantic House  
Imperial Way  
Reading  
RG2 0TD UK

[info@appswing.com](mailto:info@appswing.com)

## Copyright

---

Copyright © 2003 AppSwing Limited. All rights reserved. No part of this publication may be reproduced, transmitted, transcribed, stored in a retrieval system, or translated into any computer language, in any form or by any means, electronic, mechanical, magnetic, optical, chemical, manual, or otherwise, without prior written permission. You have limited permission to make hardcopy or other reproductions of any machine-readable documentation for your own use, provide that each such reproduction shall carry this copyright notice. No other rights under copyright are granted without prior written permission. The document is not intended for production and is furnished “as is” without warranty of any kind. All warranties on this document are hereby disclaimed including the warranties of merchantability and fitness for a particular purpose.

## Trademarks

---

AppSwing, AppServer, AppStudio and AppManager are either registered trademarks or trademarks of AppSwing Limited. All other trademarks are held by their respective companies.

Technical specifications and availability are subject to change without prior notice.

## Overview

---

Enterprises are increasingly recognising the need to integrate applications into portal and integration frameworks that are accessible to a wide range of staff, customers and partners. Existing Windows applications require back-end machine-to-machine integration so that they can be sliced appropriately for their intended audience, and incorporated with other existing solutions.

Read this document if:

- You need to connect an existing Windows application to your enterprise portal
- You want to enable the exchange of information between internally-developed Windows applications and your enterprise's line-of-business applications, such as customer relationship management (CRM) or enterprise resource planning (ERP) systems
- You must provide a subset of an internal application as a Web Service.

## Enterprise Application Integration

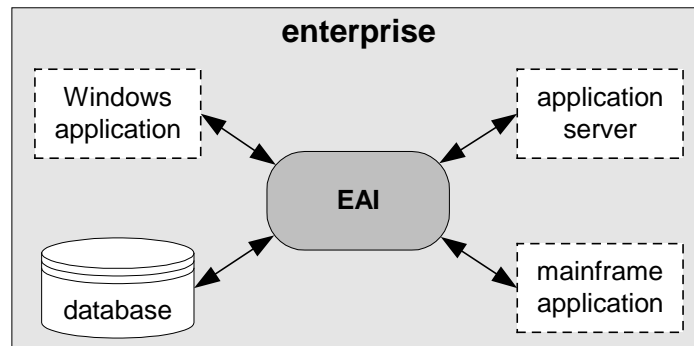
---

Any typical enterprise will have a variety of IT Systems that have evolved over time. In the early '80s, with the advent of the PC, businesses slowly replaced dumb thin terminals with PCs to improve performance and gain end user productivity with personal applications and office tools. In the mid-'80s, client/server systems began to rapidly emerge.

In the '90s, the trend toward commercial off-the-shelf software dominated, firstly with the adoption of Enterprise Resource Planning (ERP) applications. These packages were intended to provide the back office applications not adequately addressed by the legacy host systems. Later came the introduction of Customer Relationship Management (CRM) applications to provide front office management applications.

With the advent of the Internet revolution and the dot-com boom, companies were forced to respond by rapidly deploying Web infrastructures of their own. With the widespread acceptance of Java as the object-oriented development tool of choice, Web infrastructures evolved separately from the rest of the organisation's applications.

However, all these platforms and applications needed to share common data, so Enterprise Application Integration (EAI) platforms emerged to reduce the growing programming effort required for integration. The EAI infrastructure in an enterprise acts as a central negotiator that manages interactions among applications. The central node provides a useful uncoupling function: using an intermediate communication format prevents having to integrate an application several times with several other applications, and simply performs a single integration on the central node. See Figure 1.



**Figure 1: Integration**

Middleware products such as the following assist in building interrelated systems:

|  |   |
|--|---|
| <b>Enterprise portal software</b>                    | <p>Integrate different business logic units to form a single presentation unit.</p> <p>Examples: IBM WebSphere, Plumtree, SAP Portals, and Vignette.</p>  |
| <b>Enterprise application integration frameworks</b> | <p>Enable applications to talk to each other. These frameworks are focused especially on business-to-business (B2B) systems such as supply chain management and order management systems, in which heterogeneous systems must communicate automatically.</p> <p>Examples: Tibco, WebMethods, Vitria, IBM (Crossworld division), and Sybase (NEON division).</p> |
| <b>Infrastructure products</b>                       | <p>Facilitate communication between Internet and Windows applications.</p> <p>Examples: BEA (Tuxedo), IBM (MQ division), Microsoft (biztalk), &amp; Sun (iPlanet division).</p>   |

## The Problem: Connecting to Existing Applications

EAI addresses the pressing need of enterprises to leverage their investments in the diverse range of IT systems to meet changing business requirements. The diversity of the platforms, interfaces, and programming environments of these systems makes it difficult to retrieve and combine data from multiple systems, or to interface with external systems.

Most EAI platforms do not address the majority of applications in an enterprise. These are the custom applications developed without a clean interface. Incorporating these applications involves expensive and high-risk development, either to access data directly from underlying databases, or to intervene in the business logic of the legacy application. In many cases, the legacy systems are not sufficiently documented, make use of

technologies no longer readily found in the marketplace, and the original developers are no longer available.

Integration is a necessary response to changing business environments. It may be driven by a strategic initiative (such as a merger or acquisition), or the need to streamline operations (e.g., supply chain management interfacing with a supplier's inventory system or customer's database). However, connecting to existing applications, particularly when heavily customised, can be a complex task, requiring a substantial commitment of time and resources. Existing applications in general and Windows applications in particular, do not generally provide an application-programming interface (API). When an API does exist, it is typically very specific and non-standard.

Thus, companies today are faced with some difficult choices, as presented in the following table:

|   |  |
|---|--|
| <p><b>Avoid integration</b></p>   | <p>Duplicate effort in maintaining IT systems.</p> <p>Inefficiencies and inaccuracies in coordinating information.</p> <p>Lack of timely information due to manual integration processes.</p> <p>Increased costs arising from inefficiencies.</p> <p>Forfeit of new business opportunities because of inability to integrate with customers and suppliers, or failure to identify new opportunities because of insufficient or inaccurate information.</p> <p>Disadvantage vis-à-vis competitors due to absence of business agility.</p> |
| <p><b>Custom modify legacy systems to create integration interfaces</b></p>                   | <p>IT staff may not be familiar with the legacy systems and/or the technologies used.</p> <p>Legacy systems may not be well documented.</p> <p>Directly accessing the systems' underlying databases risks bypassing important business logic.</p> <p>Coding change to legacy systems can endanger existing functionality.</p> <p>The system design may not support addition of integration interfaces, thus requiring extensive modifications.</p> <p>Commitment of IT resources = immediate costs + opportunity costs.</p>              |
| <p><b>Completely redevelop legacy systems for new technologies supporting integration</b></p> | <p>High cost option, often costlier than initially estimated.</p> <p>Risky. The success of large development projects depends on many factors: personnel availability and skills, planning, management commitment, etc.</p> <p>Time-to-market for a large-scale project is long, thus delaying crucial business benefits.</p> <p>Is a one-system solution. Additional legacy systems may also require redevelopment.</p>   |

Many system architectures distinguish between the following layers:

- Presentation, which formats data for a specific user terminal
- Business logic, which performs the processes and applies business rules to the processes
- Data store, which holds the underlying information

When an application is built with this kind of architecture, some application integration methods bypass the presentation layer and directly access the business logic and data store layers. Bypassing this business logic in the presentation layer is not just complex; it is very high risk for customers. Not only are calculations skipped, but also input checks and security checks. In Windows applications, it is not uncommon for at least some of the business logic to be implemented in the presentation layer (fat clients). Actually, most of the information that exists in Windows applications is accessible only via a Windows user interface for client/server applications.

In order for companies to create new business functions and processes, they either need to rewrite their existing business functions or do expensive and invasive hand-coding. The ability to integrate Windows applications in a standard, straightforward way without rebuilding them does not exist. Connecting to existing applications in the enterprise is one of the most acute problems facing EAI vendors today.

To integrate Windows applications, traditional EAI products require either a well-defined and well-documented COM API into the application or an adapter written to one specific application. Not all Windows applications have COM APIs. In fact, most custom-written Windows applications do not have an API.

The current EAI offerings have no good way to integrate custom-developed Windows applications. If an enterprise wants to integrate information from a custom-developed Windows application into another application or business process, the only way to do it is to resort to the old hand-coded point-to-point style of integration, a style that has proven so expensive and untenable that the whole category of EAI originally emerged.

AppSwing provides a solution for the missing piece of the integration puzzle.

## The Solution: AppServer

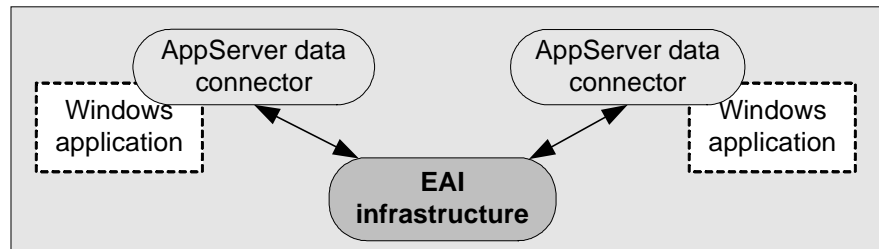
---

AppServer takes advantage of standards-based Web services to bring custom-developed Windows applications into the broader integration context. AppServer is a software solution that provides a generic interface to Windows enterprise applications. Accomplishing this integration will finally enable companies to combine business logic and data from their Windows applications with business logic and data from a range of other applications, databases, and technologies. AppServer interacts with the presentation layer of the Windows application. It presents to the integrator a simple view of the application that includes all the values required by a user, while maintaining input and security checks. Using scripts to advance the application, AppServer relies on the existing business logic to extract the relevant data for the EAI framework. In this way, it acts as a universal adapter, extracting data as required.

## AppServer data connectors

---

AppServer provides data connectors to extend Windows applications as Web services. See Figure 2.



**Figure 2: Data connectors**

AppServer can provide the following types of data connectors:

|  |  |
|--|--|
| <b>XML export and import interface</b> | The EAI infrastructure creates detailed scripts as XML input, which AppServer activates to obtain and return the requested information.  |
| <b>XML Web services</b>                | The EAI infrastructure provides values that AppServer inserts into predefined scripts, one for each query type. The script accesses the desired information and returns data to the EAI infrastructure. The XML input and output of the Web service define the query and results syntaxes, respectively. The product includes a script-building tool for recording and debugging predefined queries. |

## Product highlights

---

AppServer features include:

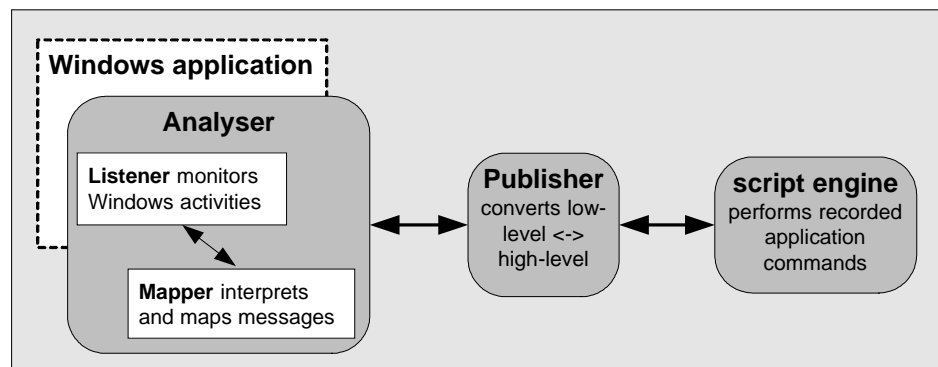
- Rapid data connector development, utilising scripts
- No reprogramming
- No requirement for the original application source code
- Multi-tier flexible system architecture with load-balancing and fail-over
- Visual script generation through a recording and editing tool
- Script debugger for testing recorded scripts
- Integration with .Net Web services tools such as BizTalk

## AppServer Architecture

AppServer is installed on the same Microsoft Terminal Server (TS) as the Windows application client. AppServer includes:

- A script engine
- Publisher
- Analyser, composed of the Listener and the Mapper

See Figure 3.



**Figure 3: AppServer architecture**

When the user of the EAI infrastructure attempts to access an application, the system activates that application's data connector, which consists of a number of predetermined scripts. These scripts simulate paths through the application and create the required XML. The requested application must already be running in the background.

The script for the requested action is activated, together with any data. The steps through to the application are as follows:

1. The script engine analyses the script and translates it into logical high-level commands.
2. The Publisher translates the stream into low-level commands.
3. The Mapper translates the commands into Window commands, which the accessed application processes.

The steps from the application are as follows:

1. The Listener executes alongside the accessed application and monitors the resulting graphical user interface (GUI) activities.
2. The Mapper and Analyser work together to send an updated representation of the GUI to the Publisher.
3. The Publisher combines the data and sends the XML back to the script engine.
4. The script engine translates the XML into an application-compatible XML schema. The EAI infrastructure receives the data.

## Competitive Advantages

AppSwing offers a unique and unmatched solution to the problem of connectivity in the enterprise application integration field. AppServer connects Windows-based applications (or slices of those applications) to the EAI infrastructure, thus facilitating full-scale integration of applications at the back-end.

The following table compares AppServer to alternative techniques for integrating existing applications into EAI frameworks.

| <b>Integration solution</b>                     | <b>Why AppServer?</b>  |
|---|--|
| Connector supplied by EAI vendor                | <p>Often, applications do not have an API, especially when they are internally developed. When they exist, each API is different and must be learnt individually, whereas AppServer requires learning a single tool.</p> <p>While ready-made connectors promise very fast implementation time, the choice of connectors provided by any particular EAI vendor is very limited and rarely meets the full range of client needs.</p> <p>Additionally, vendor-provided connectors cannot support applications developed in the organisation, or customisations carried out by the organisation on standard software.</p> <p>AppServer enables the creation of connectors for every application, be it internal or external, original or customised.</p>   |
| Connecting through the original application API | <p>By turning the application into a component, AppServer basically converts the application's user interface into an API, creating a universally available, simple integration interface.</p>   |
| Connecting through the database                 | <p>Since most applications today rely on a relational database, organisations are tempted to implement integration connectors directly through the database. The database schema of most applications is complex and often undocumented, making this solution difficult to implement. In many cases, information required for integration is computed by the application during runtime. Connecting directly to the database forces the EAI implementer to reprogram parts of the application in order to retrieve these values.</p> <p>Bypassing business logic is not just complex; it is also dangerous. Not only are calculations skipped, but also input checks and security checks. AppServer, on the other hand, works on the user interface level. It presents to the integrator a very simple view of the application that includes all the values required by a user, while maintaining input and security checks.</p> |

## Conclusions

Implementing EAI solutions with AppServer gives the enterprise a single, cost-effective, solution for integrating with all their Windows systems. This brings with it a number of advantages:

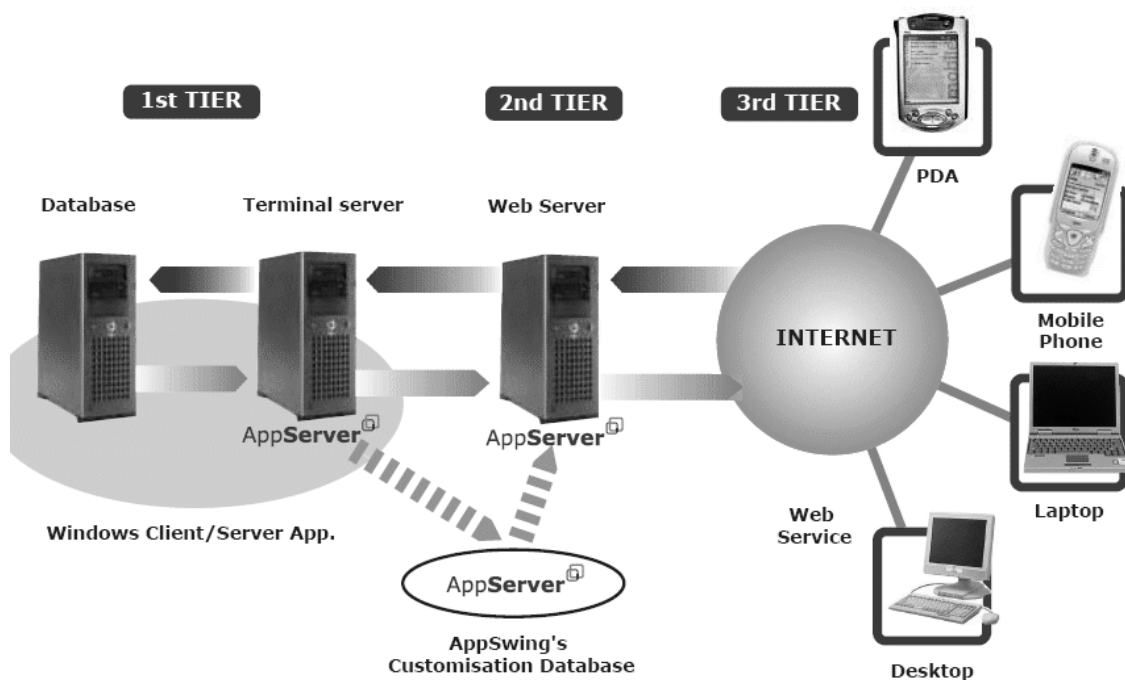
- Only one solution to implement and work with
- Opens up Windows applications that may not have been accessible via other methods
- Greater return-on-investment possible with the rollout of other applications.

## Furthering AppServer's ROI

Introducing AppServer to the enterprise not only allows Windows applications to be seamlessly extended to corporate portals but also extends the reach and value of existing business applications to the web and mobile devices.

With AppServer there is:

- No development overhead
- No client software to install, works with any web browser
- No interruption to the business
- No additional retraining costs for support and service teams



## AppServer solutions

---

### Web-Enablement

Deliver an HTML translation of existing Windows applications to a desktop browser without changing a single line of code.

- § Provide remote workers with a truly web-enabled experience through the desktop browser on their home PC.
- § Deliver sliced versions of complete applications to broaden market opportunities or control access.

### Application Mobility

Rearrange the Windows application interface to fit the screen of any mobile phone or PDA and drive backend applications securely without installing any client software or any synchronisation.

- § Connect mobile teams to existing office-based applications
- § Choose the tablet, PDA or phone to suit your requirements.

### Application Integration

Pass data from Windows applications as XML or as a web service for integration into a portal or a third party application.

- § Enable staff, customers and partners to read and update core business data on-line through the company portal or web site.
- § Integrate data from multiple applications for delivery in a single user interface.