

Connecting the Desktop with the Enterprise Stores

Sangeeta Patni, Founder and VP, Engineering, Extensio Software, Inc.

e-mail: spatni@extensio.com

The inability to penetrate the personal world of business users has frustrated many enterprise software vendors. Ruled by the ubiquitous Microsoft desktop applications, the personal desktop world of business users consists of spreadsheets, word documents, e-mails and presentations, all built using Microsoft Desktop applications, sitting on the desktop but unconnected to the enterprise data stores. This personal world along with their users has traditionally remained out of reach by enterprise software.

Multiple initiatives have been taken by vendors over the years to penetrate the personal desktop world of business users. The recent announcement of Mendocino (now called Duet), by SAP® and Microsoft®, hailed by SAP executives as one of the most far reaching technology initiatives in the recent years, proves the value of the desktop integration to their software businesses. It is easy to see why vendors are working on their desktop connectivity story. Not only do customers love their desktop interfaces and are delighted by anything that allows them to work in them, the ability to serve customers in these interfaces provides vendors the opportunity to increase their user base dramatically. This paper starts with the business driver for desktop connectivity, discusses the various desktop connectivity approaches, why they have not succeeded, and then proposes an alternative SOA-based approach that is flexible and extensible.

Connect up the Desktops: The Business Imperative

Businesses find that it is increasingly difficult for them to take the risks of decision making based on the data contained within the user's unconnected personal desktop for the following reasons:

1. **Data Accuracy:** Businesses have no means of knowing whether the data contained the spreadsheets, word, and e-mails is accurate, real-time, and free of copy-paste errors.
2. **User Productivity:** Businesses are concerned with the large number of hours spent in assembling the Office documents from myriad data sources. Users are also forced to change their context and swap their desktop windows, causing user fatigue.
3. **Data Security and Accountability:** CFOs and CIOs worry about the absence of enterprise security and data accountability in Office documents.

Enterprises need ubiquitous Desktop Connectivity technology that lets their business users:

1. Create and assemble Office documents with data from multiple enterprise sources, with minimal man-hours and without cut-copy-paste human errors
2. Create documents that reflect the current information from the enterprise information stores
3. Get secure access to enterprise data stores within their context and Office interfaces, within their information tools, for faster and better decisions

The gap between the desktop and the enterprise data sources has created a huge opportunity for enterprise software vendors, but also a challenge to create technology that is extensible, flexible, personal, easy-to-use, yet secure and connected, and works within the favorite Microsoft Office interfaces. There indeed are various point-solutions available for connecting the Microsoft Office Desktop with back-end enterprise data stores, but none of them are generic enough to be used by all users for multiple tasks.

The challenge is large not because of technology alone but also because of the must-have requirement of seamless user experience aligned with the way people work within these

interfaces. The solution also has to be deployable within the enterprises meeting their security and extensibility requirements.

Connecting the Desktop: Key User Requirements

The desktop world of business users is characterized by two key attributes - Flexibility and Ease-of-Use. For any desktop solution to succeed, it must ensure that it does not jettison this essential value, while maintaining the following:

1. **Flexibility of creation:** Ability to create personalized documents with data assembled from different back-end data stores, with minimum efforts and minimum errors
2. **Ease of interaction:** Ability to transparently view, update and initiate transactions with the back-end data store in multiple ways without having to know the complex back-end details
3. **Flexibility of format and presentation:** Present the information in a personalized front-end specific manner
4. **Flexibility of sharing:** Ability to share the document with internal and external users

Connecting the Desktop: Key IT Requirements

On the other hand, IT requires a desktop connectivity solution that meets the following criteria:

1. Connects to all existing and future data sources, within and outside the firewall
2. Extensible platform that supports multiple interfaces and interaction models
3. Avoids task based piece meal point solutions by specific applications
4. Controls access and authorization for data accessible within Office

Further, like most IT investments, the solution must leverage existing IT investments, be cost effective, be easy to install, maintain, and deploy, and should offer a choice of infrastructure

Connecting the Desktops: Typical Approaches

Over the years, enterprise vendors have come up with multiple desktop integration strategies that vary from a simple generation of report in Word or Excel to refreshing the application specific data. Some work on pre-defined templates, while some carry out specific tasks and some work only on specific Office interfaces.

Here is a brief overview of the popular desktop strategies implemented:

1. **Generate desktop documents:** Many vendors create reports in pre-formatted Word, Excel or e-mail alerts with data from only that application. Some have refresh capabilities to ensure that the data stays current.

Examples:

- a. **Oracle WebADI** using VB Macros, that creates Excel and Word documents with data contained in Oracle eBusiness applications
- b. **Excel Report and PDF generator** from Oracle XML Developer, and other reporting vendors
- c. **Export to Excel** option by Report Modules of almost all enterprise applications

These documents are mostly static, useful for analysis and information gathering and meant for as-is usage. Users cannot use these documents to create and assemble their own documents without following their old cut-copy-paste methods, which comes with its attendant risks. Further, there is no flexibility to connect these documents with other sources,

other than the application source that created it. In addition, the export step is repeated every time the latest information is needed.

2. **Upload from Desktop Documents:** Some allow data to be uploaded in their environment for pre-defined transactions in pre-defined Excel, Word or Info-path forms.

Examples:

- a. **Oracle Web ADI** using VB Macros with specific Excel templates that upload data into specific Oracle eBusiness Application transactions
- b. **Peoplesoft (now Oracle)** for select master and transaction data

While this meets the requirements of batch uploads in most cases, and perhaps an occasional update using forms, this is limited to just one kind of pre-defined interaction with the enterprise data source. Further, this pre-defined task has to be performed using pre-created rigid templates and document structures, imposing limits on flexibility of creation and its usage itself.

3. **Embed live elements in spreadsheets:** Some vendors enable users to create custom spreadsheets along with live spreadsheet elements that can be refreshed.

Examples:

- a. **BI Add-ins for Excel:** Oracle BI, Hyperion, and Business Objects
- b. **Juice** (Bought by Proclarity, which in turn was bought by Microsoft)
- c. **Gurunet** created a virtual hyperlink but required a data warehouse of relevant data

While this meets the user's flexibility needs of document creation, the data source accessible is typically limited to a single one. Further, it is restricted to only Excel and not on other Office interfaces.

4. **Provide a data access folder/portlet/web part in Outlook:** Some vendors enable users to view enterprise data stores as Outlook folders that are essentially browser links.

Examples:

- a. **Salesforce** with Outlook Plug-in

This works only against a specific data source, with limited interactivity options. Further, it works only with Outlook and not other applications.

5. **Predefined Synchronization of Outlook with Information Stores:** Many vendors provide address book and contact synchronization with back-end systems.

Examples:

- a. **Mendocino/Duet** from SAP and Microsoft to sync up Outlook calendar with specific SAP transaction within SAP Project Management module.
- b. **Blackberry, other PDA vendors:** Contacts and Calendar sync up
- c. **Outlook connector** from Oracle Collaboration Suite

This works for a small set of pre-defined tasks and connects desktop with repositories of certain type of data with a select few transactions at the back-end. Users can use it only for limited purposes. Further, this is typically offered only against Outlook and does not extend to other Office interfaces.

Why Existing Approaches Have Not Worked?

None of the above approaches have managed to garner support from a large community of end users. Some approaches have failed because they needed data to be collected in one huge store before generating one desktop document, an impractical deployment hurdle. Some have attempted to offer limited tasks, such as synchronize a part of the data, while others have provided limited interactivity with select transactions on select Office interfaces. Some still require the users to login into the enterprise application before the data can be exported to their desktop applications. Some provide only static data, and some provide read-only access.

The existing approaches have all ended up providing point solutions that work for a specific task or a specific desktop application that works against a specific data source. They do not extend to other data sources, or other Office applications. Plus the IT department gets called in to create point solution for each such enterprise application, and the users have to learn the myriad ways of each such extension.

None of the approaches have looked comprehensively from the end-user perspective in how they want to interact with enterprise applications from within Office documents and what would meet their usage needs. End users do not want partial solutions that work against select applications, or sources and transactions, and tie them down to rigid templates, or force them to use pre-formatted Office documents. What end-users really need is a technology that retains their flexibility and ease-of-use in full measure. They need a desktop connectivity solution that permeates enterprise data within the interface of their choice, from the data store that they want, use it the way they want and present it in the format they want, with centralized control being exercised by their IT department.

The Mendocino/Duet Juggernaut

Mendocino/Duet from SAP and Microsoft promises connectivity to SAP from within Office interfaces. While the entire implementation is seriously handicapped in terms of transactions and data sources covered, accompanied by serious issues on TCO and platforms, Mendocino offers interoperability in a manner that tries to keep end-user flexibility intact, thereby making it potentially endearing. With its in-context menus on Outlook that offer choice of interaction options and menu, end-users see future possibilities of extending such interactivity modes to multiple sources, multiple interfaces and more transactions.

For Mendocino, Microsoft and SAP have used the Information Bridge Framework and have created in-context menus and context sensitive smart tags to create interesting user demos for specific SAP transactions. Its extensibility remains to be proven, especially given the fact that it has taken 100+ engineers toiling over 15 months to create Outlook connectivity for just 5 SAP transactions out of hundreds.

Assuming that the approach succeeds, customers should note that this comes with a heavy price tag. Enterprises would need to upgrade their desktops to Office Professional 2003+ and their SAP to mySAP ERP 2004+, in addition to investing in a heavy middleware stack, consisting of multiple technologies from SAP and Microsoft.

Despite its serious limitations and its obvious high TCO, Mendocino has fired the imagination of CIOs and end-users alike, not with what it can do now, but what it could do in the future. Further, given the marketing machinery of the two software giants, they certainly would be able to seed the market. Whether the Mendocino/Duet brand of desktop connectivity will be successful would still depend upon how useful it really is, and whether its competitors let Duet control the market.

Do customers really have no other way to get desired desktop connectivity, or are there some alternatives?

Overview of an Alternative Approach for Desktop Connectivity

Instead of a piece meal approach, such as offering Excel reports or Outlook based sync ups, and creating a custom desktop integration solution for every enterprise application and task, what is needed is a generic solution that works with all enterprise applications and all desktop applications, and is flexible enough to perform a variety of tasks.

A generic solution needs technology to be able to connect with multiple data sources using the native protocol and formats. It also needs support for understanding the various end-user Office applications, their interaction model, their multiple presentation formats and their multiple versions.

The only way to abstract multiple data sources, formats, protocols, and their security, is through a middleware that can faithfully abstract the back-end systems from the front-end user applications, and deliver data from back-end systems in the presentation format required by the Office application. This is akin to the typical 3-tier web architecture where the middleware sits between the presentation/display layer and the data layer. Just as the business layer sits in the middleware, a similar integration and presentation layer can solve the desktop integration problem also. In addition, some basic tools or wizards would be needed to create the business logic to extract, filter, and aggregate the data from the back-end systems.

This end goal can be achieved with a combination of a SOA based Information Delivery Server and an end-user interface coupling technology such as an Office Add-in at the user's desktop. The SOA architecture provides the mechanism to abstract data from multiple back-ends as information services that can be picked and selected for read/write/embed tasks within Office documents, while the Office Add-in implements the user interface, presentations and inclusion of information elements in the Office documents.

The SOA based server can provide back-end connectivity to multiple enterprise applications, including SAP, Oracle, Databases, portals, XML sources, etc. and abstract information contained in them as Information Services to end-users, shielding them from complex back-end details. The SOA based server, in addition to serving up data within the desktop application, can also provide them as secure web services. The SOA-based server can be implemented on top of both a J2EE and .net based system leveraging the inherent functionality of the application server such as EAI, ETL, data connectivity, XML transformation, security, and so on. The development tools to build the services can also leverage the middleware platform.

Office Add-in interfaces can work with technologies such as COM/ATL to deliver information elements along-side or inside Office documents.. The custom Office Add-ins can have the logic to connect to the Information Delivery server, fetch the services list, select the service, and execute the service request. The data returned can be displayed in the front-end Office documents using specific presentation format. They can also implement a variety of user interfaces such as toolbars and window panes within the Office application to request for information service and receive desired information on them. .As Office Add-in interfaces are supported on all versions of Office 2000+, customers would not require any MS Office upgrades.

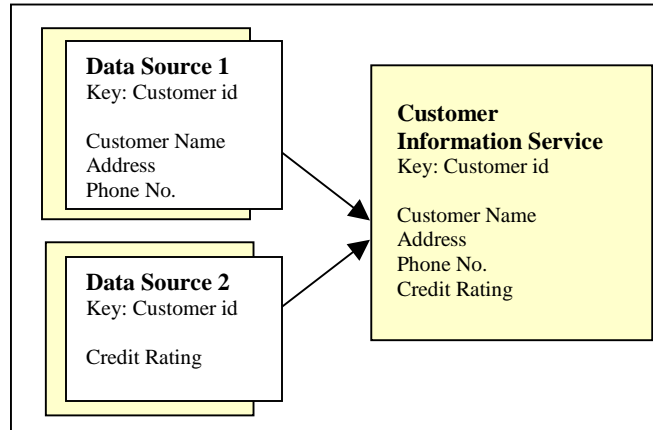
With an extensible, cost effective and lean mechanism for connecting desktops with enterprise applications, enterprises will be able to offer desktop connectivity for all enterprise data, with multiple interaction modes to their users in a consistent and generalized manner.

Information Services Overview

The remainder of this document describes the proposed architecture and its components.

Information Services, broadly speaking, are a logically grouped set of information elements, extracted from data sources about an information entity. An information service may have information elements combined

from multiple data source, or could be the result of some transform operation on the original data. For example, a Customer Information service may take a customer ID and return the Customer name, address, phone number, and credit rating. End users view and consume this service without knowing that the customer name, address and phone numbers have come from their back-end CRM



application, while the credit rating has come from their custom database application or a SOAP service. These information services are typically hosted on a server within the data-center.

By moving to a service-based model, the users just specify the service they want (e.g. customer rating service), and they get the related data from the service directly into their Office document cells, without having to know the where, the how, and the when of the data. It is the service now that knows the details of getting the data and not the user.

Service-oriented Architecture (SOA) based Information Server

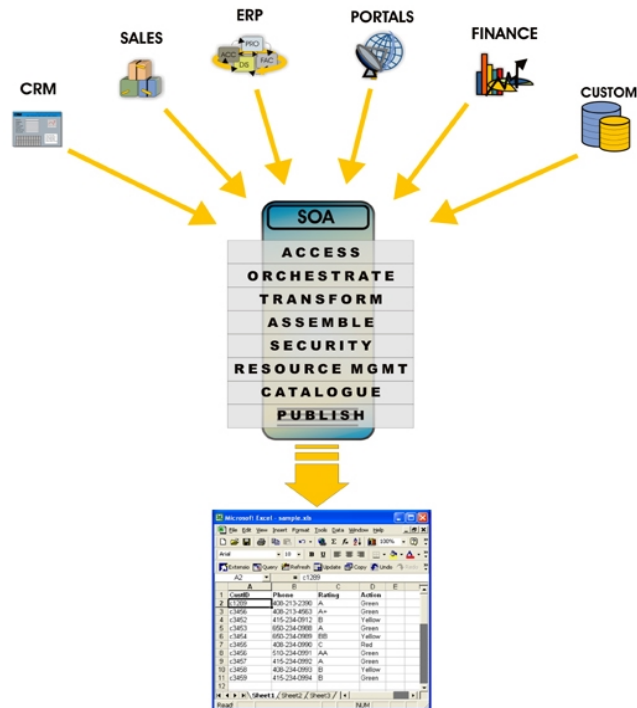
An information server using service-oriented architecture enables a loosely coupled integration of the back-end resources such that it masks the application consuming this data from the underlying IT complexity. This complexity includes the back-end resource structure, data-format, session management, security, connection pooling, and caching.

The main value of the SOA-based architecture is in deriving value out of the existing applications and resources without duplicating data, business logic or security efforts. The SOA based framework is typically composed of the following elements:

- Native interfaces to connect, authenticate, and access back-end sources. A SOA based system can access heterogeneous resources using SOAP or resource native interfaces such as SQL, HTTP, SAP BAPI, or custom protocol.
- Logic to combine and unify the data from multiple sources. This can be a simple JOIN statement to complex scripting (with BPEL) depending upon the user requirements.
- Framework to expose the combined data elements as an information service that can then be consumed by end-users, devices or other applications.
- Framework to publish service directory and service schema

For a SOA based framework to provide data for end-user consumption, the technology needs the following additional components:

- Modules to map the result presentation layer to interfaces such as Excel
- Protocols to connect and authenticate various end-user interfaces
- Interface specific service publication and registry
- Interface specific update process



The SOA based information delivery server contains XML metadata about the services such as the data keys, connection details, transformation, assembly, resource information, and security. The service specific

meta-data would typically be stored in a database and used at run-time for request processing. If there are any changes to the request process, they can be made without any user impact.

The server publishes a directory containing the services and attributes. The user chooses the service and input parameters, and sends the request to the server. The server responds by calling the enterprise application using the meta-data, getting the result in the native format, extracting the relevant data, converting the data format and finally sending the data to the end-user.

Office Add-in Client for End-user Desktops

Microsoft Office provides well-defined means to add custom commands, interfaces and features to Office using Add-ins. Examples include Excel based statistical and financial packages. Office Add-in interfaces are supported on all versions of Office 2000+ and do not require any MS Office upgrades. Office Add-ins are stable, popular and have been used by hundreds of developers for years now. End users themselves can install office Add-ins easily.

While Add-ins are typically used to manipulate data within the existing Office document, they can do a lot more including contacting an external server and fetching external data within the Office document. They can connect with middleware servers and perform user authentication, data request and data retrieval tasks. They can work with technologies such as COM/ATL to interface with information objects inside Office documents. They can also embed information elements in documents that can be refreshed on demand by end-users.

In addition, Office Add-ins can implement application specific user interfaces on Office documents including:

- a. An Office tool bar that hosts information services from SOA based Information Server

- b. Input and Output Wizards that enable users to choose the information service of choice, provide inputs and receive responses from back-end data stores in multiple interfaces
- c. A browser based pop-up window that presents the data view on the current Office screen
- d. A desktop application bar, like the Office help bar, containing application menus, buttons and data window to show the data response from the Information Server

The Office Add-in consists of two components – the Server Component that connects with the SOA based Information Server, and the Desktop Application Component that provides the end-user interface and the delivery component in the desktop application specific format and interface. The Desktop component sends the parameterized service request to the server using XML/HTTP. It then interprets the XML results sent back by the server and passes them on to the Desktop Application Component. The Desktop component either reads/writes the response in the desktop document in its specific format, or serves it up for end-user presentation.

By communicating in XML, the Information Delivery Server and the Office Add-in retain the ability to send any other meta-data attached to the information service data. Examples of future extensibility include suggested refresh period for embedded information elements, associated services for the information element in questions, and any special format related information.

Defining and Building Information Services on a SOA Server

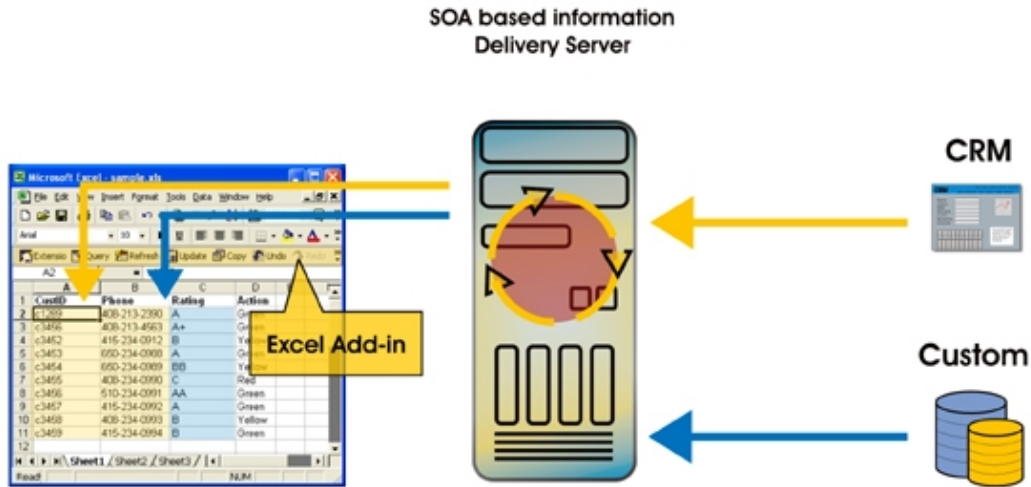
The IT department in conjunction with the business analysts and users typically develops the information services. Alternately, the application vendors can ship prepackaged information services that access their own applications, and then the IT department can create information services that combine data from those services. Once the services are defined, the business analysts can select the presentation and the fields required by the users.

IT Developers take the following steps to build the services:

1. Identify information entities such as customer, including validation and mapping
2. Identify the data sources that contain information elements for the entity
3. Define the rules to fetch/update data from data store using the development tools. The rules can be written using resource specific wizards such as the SQL query wizard, SOAP wizard and the XML/HTML filter
4. If required, define the rules for transformation, mapping, and assembly of data
5. Define the presentation layer as appropriate for Outlook, Excel, or Word
6. Publish the information service on the services directory for the end-user
7. Assign user/group permissions for information services

Depending upon the complexity of the service, it may take from a few minutes to a few hours or more to build a set of information services, whether for lookup, reports, update, or alerts. SOA based architecture provides the required infrastructure for IT to serve, manage and control end-user access to information services.

An Example: SOA and Excel Add-In Integration Architecture



The Information Delivery Server and Excel Add-in work in tandem in the following manner to provide connectivity to back-end information sources on Excel spreadsheets:

| User Requirements | SOA Contribution | Excel Add-in Contribution |
|---|---|--|
| Easily select data from multiple data stores, and refresh them later. | Provides information services containing elements spanning multiple data-sources. | Provides the ability to link and refresh Excel cells using information services. |
| Eliminate copy-paste, and use of SQL/XML to get data in Excel | Information services that hide the technical details of accessing data-sources | Provides menu based interface to get data directly within Excel |
| Update/Insert data in back-end from Excel | Provides the update interface if supported by applications | Consistent mechanism to specify and send updates |
| Have a replicable data collection process | Centrally available services allow reuse across users | Have persistent links to services in Excel |
| Maintain “single version of truth” and cell-integrity | Live on-demand connection to data-sources ensures latest data | Provides refresh on-demand, audit, version control and cell protection, for cell integrity |
| Ability to add internal and external data sources | Allows addition of information elements from new data sources with the tools | The newly published services are accessible from the service directory |

Benefits of SOA enabled Information Delivery for Desktop Connectivity

The benefits of the SOA based Middleware with Office Add-ins are:

1. Choice of Office Add-in technology as a desktop application that works on Office 2000+ installations and can connect to back-end SOA server over XML/HTTP
2. Add-ins can be extended to build rich user interfaces such as toolbars, side panes and floating e-window, using a combination of COM/ATL and command bar interfaces
3. Office Add-in offers full flexibility to end users to create, edit, share and build live Office documents that can be refreshed on-demand

4. Sophisticated SOA based three-tier architecture that enables information services to be created that can be extracted, transformed, unified and presented to multiple end-user consumer points from multiple sources
5. XML/HTTP based client server communication enables access across firewall
6. Centralized service development, authorization and deployment
7. Better choices for the middleware, including tools and infrastructure

Summary

With a combination of SOA based Information Delivery server and an end-user interface coupling technology using Office Add-ins, software providers can build rich desktop connectivity solutions that retain end-user flexibility and ease-of-use on Office interfaces, with sophisticated read/write modes of interactivity with multiple back-end data stores. This solution is extensible, cost effective and a lean mechanism for connecting desktops with enterprise applications. Further, such desktop connectivity solutions work across multiple versions of Office and can connect with multiple back-ends, with a choice of middleware infrastructure.