



# Address Book integration with JDE using iBOLT

A PUBLICATION BY  
Forza Consulting & Magic  
Software

September 2010

Magic Software is a trademark of Magic Software Enterprises Ltd. All other product and company names mentioned herein are for identification purposes only and are the property of, and may be trademarks of, their respective owners. Magic Software Enterprises has made every effort to ensure that the information contained in this document is accurate; however, there are no representations or warranties regarding this information, including warranties of merchantability or fitness for a particular purpose. Magic Software Enterprises assumes no responsibility for errors or omissions that may occur in this document. The information in this document is subject to change without prior notice and does not represent a commitment by Magic Software Enterprises or its representatives.



## Managing your address and contact information is challenging!

Contact and address data tends to be stored by different applications dispersed throughout your company. With this contact and address data being so fragmented across your systems, managing the integrity and validity of this data can be an enormous challenge. In addition to JD Edwards your contact and address data could be stored in a warehouse management system, a CRM system, a customer service application, a HR system, or an intranet application or web portal such as Microsoft SharePoint.

When contact data is not centrally managed, but is maintained across several applications, it becomes virtually impossible to guarantee that the contact data can be kept consistent. Ensuring that the contact data is the same in each application will typically require a lot of effort and discipline. The possibilities of inconsistencies occurring are high as manual processes are prone to error.

The implementation of an automated solution for integrating and synchronizing address book data between your various applications will reduce the costs and errors that result in attempting to keep this type of data consistent. In addition the time and effort required and the level of human intervention will be significantly less.

## Synchronizing you data

To ensure the consistency of your data across different databases and to avoid entering/maintaining the data more than once, interfaces will need to be developed between the different applications.

Normally these types of interfaces are batch driven and synchronization occurs just once a day, i.e. mainly during the night. As a result the interface is not real-time, however with the use of a web service based integration you can keep your data consistent real-time.

Often the synchronization of data is unidirectional. This means that the data is managed and maintained in one (leading) application and replicated to the other applications. A drawback to this approach is that changes inadvertently made in the 'secondary' applications are not replicated back to the lead application. For example, if JD Edwards is the leading application for address book data then in this approach any changes to address book data made in another application would not be synchronized back to JD Edwards.

where data is synchronized from one database to another, interfaces are usually point to point. These interfaces are usually developed on an 'ad hoc' basis, at a point in time where the requirement for the interface with a specific application has become a necessity.

With these types of interfaces it is quite common that the knowledge and detailed documentation of how the interface in question has been constructed is missing. As a result it becomes a major challenge to manage and maintain the contact data as well as the interfaces themselves.

If there are any possibilities of data errors occurring during the automatic synchronization of data, then these need to be caught by an error handling procedure. If this is not taken care of, then over a period of time the contact data will become inconsistent between the different databases

## Required Knowledge

Synchronizing contact data between the JD Edwards address book and other systems may seem simple. However building the integration is no easy task, given the number of tables and the referential integrity within address book in JD Edwards.

The JD Edwards Z-file functionality allows users to make changes to the address book. There are many scenarios in which Z-file process does not provide sufficient functionality.

In the customization required for integrating different data, you need to take into consideration the business rule logic and the referential integrity of the JD Edwards database. As a result this means that the building of an address book interface has become a major task requiring a significant investment in time for both consultants and end users, and of course, this requires knowledge of the JD Edwards development tools.

## The Perfect Interface

The perfect interface does exist! This is an interface where all address and contact data is seamlessly integrated and users do not need to be concerned about the validity of the data. Such an interface would have the following characteristics:

- The interface is real time. Address and contact data is immediately updated in multiple systems
- Although the database is leading, users still want to have the possibility to add/change data elements in the various databases. For example in the case when an address has been changed or a contact has been added. The synchronization needs to be executed in both directions.
- The interface can be developed and implemented quickly, taking into consideration the business rules as they are defined in JD Edwards.
- There is a clear overview of all disparate interfaces which can be easily managed using one tool. The details of each interface can simply be retrieved. It is clear which role this process has in relationship to other interfaces and processes.
- There are no inconsistencies such as incomplete and 'duplicate' records. There is an error handling procedure to prevent this from happening

With iBOLT it is possible to build the perfect interface.

## The advantages of iBOLT

The power of using iBOLT to integrate address and contact data is the speed, ease and cost effectiveness of building the interface. Each JD Edwards related scenario is realized quickly without requiring a large team of specialists. Integrations and their transactions can be managed and constantly monitored.

### Simplicity

Simplifying the development process has always been a priority for Magic Software. This is also the case for their iBOLT integration suite. The following points are examples which emphasize the simplicity of iBOLT:

- iBOLT integrations are developed 'code free', this means without using a programming language. You do not need to be a programmer to develop in iBOLT. Interfaces can be developed using "drag and drop" in a graphical environment.
- iBOLT has a standard JD Edwards connector. This connector gives you direct access to JD Edwards's business logic (business functions) both standard and customized. This makes iBOLT the ideal integration tool for JD Edwards. The development of web services in iBOLT is even easier than using the JD Edwards development toolset to build business services.
- iBOLT is a comprehensive integration framework which connects JD Edwards to other applications.

The simplified development process within iBOLT ensures that you can develop interfaces faster. In addition you do not require as many specialists as you typically would need when building an interface between systems or creating a new workflow. The simplified framework makes it easy for Application Managers to design and develop new interfaces.

## Completeness

The integration processes are built using a visual design tool which is the iBOLT studio. The integrations are built using a range of standard components, wizard and connectors. In addition to the JDE Connector, iBOLT has a large number of connectors allowing you to integrate too many other applications. By using these components, connectors and wizards, the development process of building interfaces is simplified and shortened. Next to the JDE Connector are many other standard connectors. For example Microsoft SharePoint, Microsoft CRM, Java/RPG/.NET, JMS en MSM queues, MS Office, Outlook en SAP.

The diagram below gives you a view of the standard components that are available in the iBOLT Studio and can be used to develop interfaces.

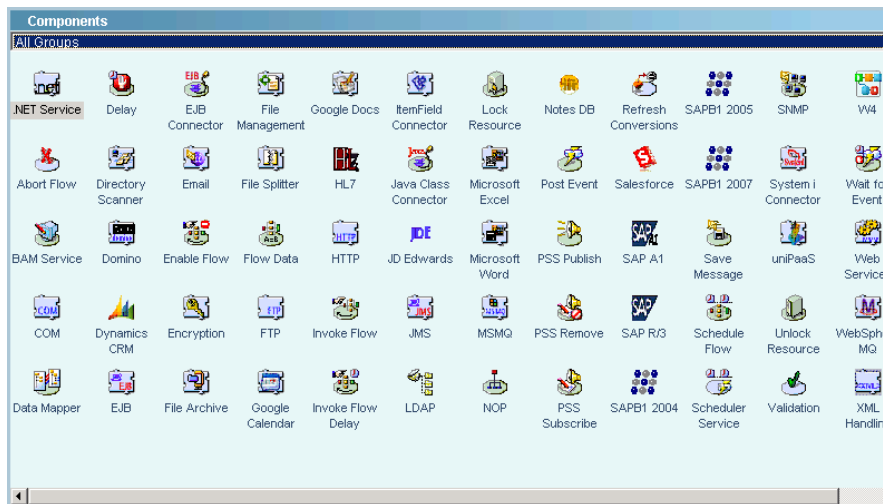


Figure 1. De iBOLT Component Library

It is also possible to develop custom connectors using the Magic UniPaaS platform which is 'code free'. These can then be added to the iBOLT component library.

The iBOLT monitor is a standard component of the iBOLT integration suite. It allows you to monitor the status of interface transactions as they are processed by the iBOLT server and provides you with detailed logging functionality.

In relation to error handling it is possible to generate warnings within iBOLT. For example within the JD Edwards work center application, in a separate online (web) application or in an email/SMS message. Transactions which fail to be processed for whatever reason can be placed on hold and resubmitted for processing at a later stage ensuring that they are not lost.

The iBOLT suite has also built in Business Process Design and Workflow functionality. Using a graphic design tool, process flows can be built. Typically this would be the first step in developing a specific interface such as the address book synchronization. With this functionality it is possible to design a workflow containing a number of interfaces and decision points. For example if we look at the process of changing and authorizing payment terms. The synchronization of the change in the payment term with other systems would be just one step in the total process flow as defined in iBOLT

In iBOLT it is easy to analyze, manage and change existing interfaces. The graphical presentation of the business process and the details of the interfaces provide you with a sound basis for documentation of your current interfaces.

## **Return on Investment**

The iBOLT suit is scalable. The platform can grow as the number of interfaces increases. The license structure for iBOLT is based on this concept. Above all, the reduced need for specialists, the ability to rapidly develop and deploy interfaces and the pricing model of iBOLT ensures that a return on investment for each iBOLT interface is economically favorable. This is valid for even the relatively small interface requirements such as the address book interface or a web shop interface.

## Case: SharePoint Integration

To give you an idea of how an address book interface would work, an example of an address book interface with Microsoft SharePoint is further explained in this section,

Microsoft SharePoint is a practical example where there is a requirement to synchronize address book data with JD Edwards.

SharePoint is a commonly used portal with a large number of standard business templates including address book forms. SharePoint is usually deployed as a front-end solution.

In many cases there is a requirement to integrate the data processed in a SharePoint form at the front-end with a back-end ERP system such as JD Edwards.

The interface with SharePoint uses a web service API which is typically not easy to develop. However, iBOLT has a SharePoint connector for the web service API which simplifies the integration with JD Edwards.

## Example: An address book change from SharePoint to JD Edwards

This process shows how a change made in SharePoint is synchronized with the Address Book in JD Edwards. Described below is what happens within the iBOLT flow which performs the synchronization.

- In SharePoint a 'List Web Service' for contact data is used to trigger a notification of a change, an addition or a deletion of a specific contact.
- This information is picked up and processed by iBOLT. iBOLT uses the action code to determine which activity needs to be performed i.e. an addition, change or deletion of a record
- As a result a record is added, changed or deleted in the JD Edwards address book. The JD Edwards Connector is used to call the required JD Edwards business function to perform this activity. When a record is added to the Address Book, the JD Edwards address book number is passed back to SharePoint establishing the referential integrity between both systems.

The diagram below depicts the flow as previously described.

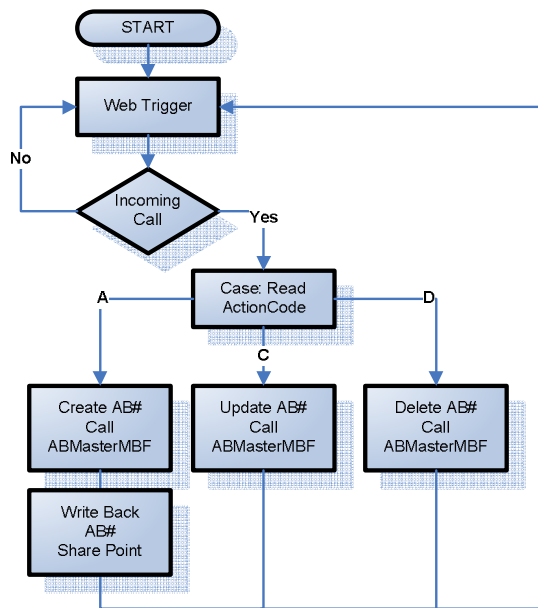


Figure 2. Flow – SharePoint to JDE

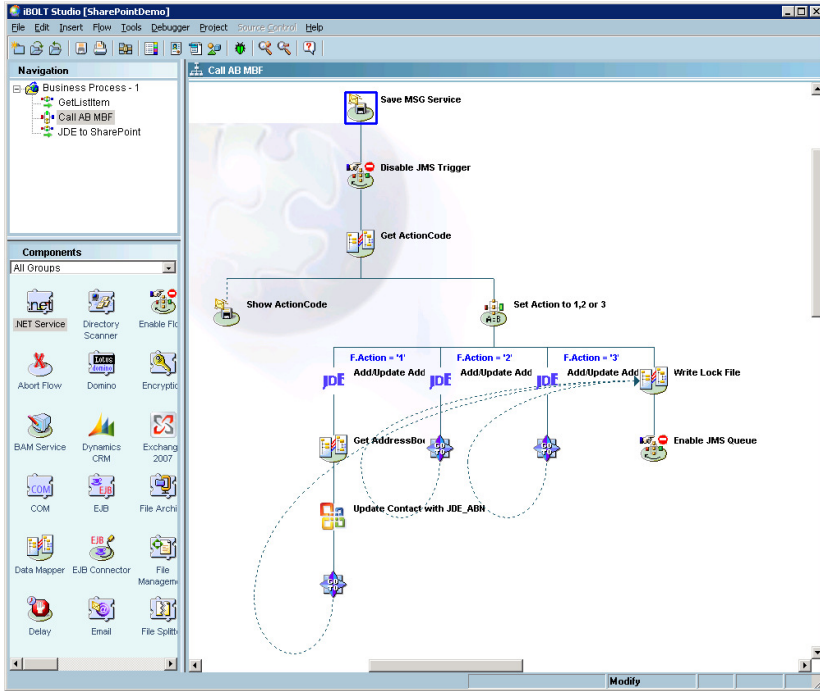


Figure 3. iBOLT Flow - Call AB MBF

The iBOLT data mapper allows you to create associations between fields from a source application (SharePoint) and a target application (JD Edwards). In this case the relationship is created by simply dragging a line from the SharePoint field to the related JD Edwards field.

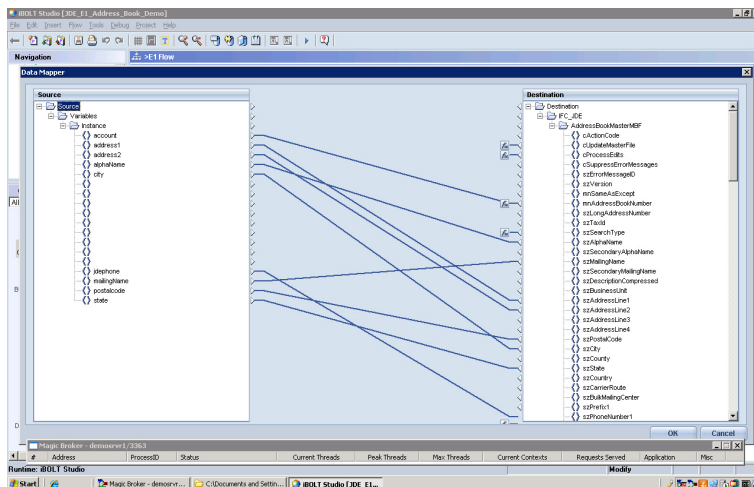


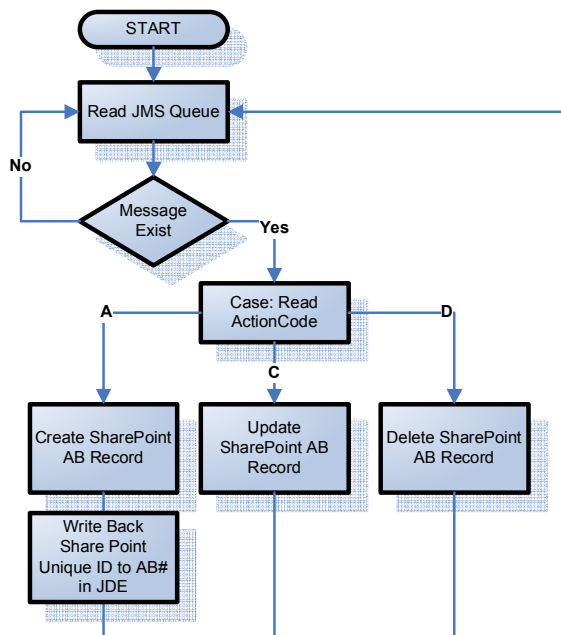
Figure 4. Address book MBF data mapper

## Example: an Address Book change from JD Edwards to SharePoint

This process shows how a change made in the JD Edwards Address Book is synchronized with an Address Book record in SharePoint. Described below is what happens within the iBOLT flow which performs the synchronization.

- A change in the JD Edwards address book triggers a Real Time Event (RTE). The real time event (an XML message) is then placed on a JMS Queue.
- The JMS Queue is scanned for inbound messages (effectively address book changes).
- When a message is found in the JMS Queue. A check is performed to verify if the address book record is locked. If the record is not locked the message is processed.
- The SharePoint record is changed added or deleted using the SharePoint connector. When a record is added, a SharePoint unique ID is passed back to the JD Edwards address book maintaining the referential integrity between the two systems.

The diagram below depicts the flow as previously described.



Figuer 6. Flow – from JD Edwards to SharePoint

This is how the above flow looks in iBOLT Studio:

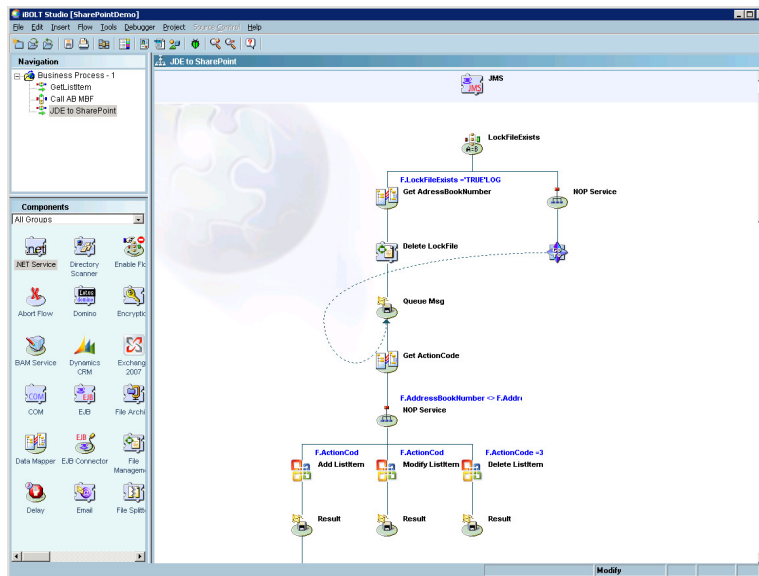


Figure 7. iBOLT Flow – JDE to Sharepoint

## Conclusion

The options for developing robust integrations with JD Edwards using iBOLT are very extensive. The address book integration with Microsoft SharePoint is just one example.

Whenever there is a requirement to make JD Edwards data available to other applications or within a business process there is a need to use JD Edwards business logic, iBOLT is the ideal choice to design, build and deploy the integration. As a result expensive, complex and time consuming development projects will be a thing of the past.