



invisibleCRM

113 Barksdale Professional Center Newark, DE 19711 Avora Holdings, Ltd. www.invisiblecrm.com

Outlook Integration for Amdocs CRM 6

Technical White Paper



Table of Contents

1	OVERVIEW.....	3
2	PRODUCT DESCRIPTION	3
2.1	FEATURES	3
2.2	OBJECT SYNCHRONIZATION SCENARIOS	10

1 Overview

This document is a technical description of the software product, aimed to create bi-directional integration between Amdocs CRM (hereinafter Amdocs CRM) and Microsoft Outlook (hereinafter Outlook).

The document describes key features of the Outlook Integration for Amdocs CRM (hereinafter The Solution), provides a high-level architectural overview of the solution and its components, and exhibits the functional and implementation design of each component. The Solution is developed for two versions of Amdocs CRM - 12.5 and 6. Architecture, laid down in this document, is common for both versions of The Solution.

Using this document IT professional experts can review The Solution and, based on this, provide their opinion for decision makers.

2 Product Description

2.1 Features

The Solution is designed to maintain the consistency of data collected inside of Outlook (Personal Storage as well as Mailbox) and the corporate CRM database while the user works with Amdocs CRM online, or with Outlook either online or offline.

It is assumed that the user owns appropriate Amdocs software licenses including Amdocs CRM user licenses, e.g. Amdocs Sales Web Client, and server-side licenses, including BEA WebLogic, Amdocs CRM and Amdocs CRM Integration Gateway. The integration works mainly with Amdocs CRM server components and it updates Amdocs' database, but it also provides UI integration and additional features such as updated forms and a Calendar view for the Amdocs Sales Web Client. Amdocs Sales Classic Client (Windows Client) will work with data synchronized with Outlook via the server-side components, but there is no integration on the client side.

The following objects can be synchronized either manually or automatically:

Outlook Object	Amdocs CRM Business Object
Contacts	Contacts
Calendar Items	Action Items
Tasks	Action Items
Email Messages	Action Items

The Outlook personal contacts folder is mapped to Amdocs CRM's Contacts business object. All Outlook calendar items (Appointment, Meeting Request) and Tasks (Task Request also) are mapped to Amdocs CRM's Action Items.

The Solution also allows pushing of incoming and outgoing email messages from Outlook to Amdocs CRM in form of logging Action items in Amdocs CRM according to Email type.

The Solution extends Outlook's standard forms to manage additional CRM-specific details using additional Tabs and optional fields. Also calendar items and messages can be linked to Amdocs CRM's objects such as Account, Contact, Opportunity, Lead.

The Solution does not impinge on Outlook functionality, the user continues to work with Outlook as he or she did before. All additional features are optional; moreover all extra fields are available in offline mode, so the user can fill in CRM details using cached master data to define links to Amdocs CRM's objects or references to dictionaries.



The user is able to configure The Solution behavior on his or her desktop using settings such as synchronization rules, synchronization schedule, etc. These options are available as an extra Tab inside Outlook ‘Tools\Options’.

From the Amdocs CRM web client prospective, the user uses new buttons and navigation links to push information and synchronization with Outlook. The Calendar View improves the overall usability of Amdocs CRM with daily, weekly, and monthly views of action items laid-out in the familiar Outlook style.

Architecture

The Solution consists of components of two major types:

- **Server-side components**, which are located on the application server (BEA WebLogic) on top of Amdocs CRM server;
- and **Client-side components**, which are incorporated into Outlook as Add-ins and separate SyncService processes.

The server-side components are (Java, JSP, EJB):

- **Amdocs CRM Web Services for Outlook Sync** – EJB objects wrapping Amdocs CRM objects and exposing web services for client-side components to interact over HTTP(S);
- **Amdocs CRM Web Services configuration file** – java property file, exposing Amdocs CRM and WebLogic installation settings;
- **Amdocs CRM Web Application for Customized Forms** – modified JSP pages of Amdocs CRM changing UI of Amdocs CRM web client;
- **Amdocs CRM Web Application for Calendar View** – modified and added JSP pages of Amdocs CRM extending UI with Calendar View feature;
- **Amdocs CRM-Outlook Object Mapper** – set of XMLfiles describing mapping of Amdocs CRM and Outlook objects and fields;

The client-side components are (VC++, ATL, WTL, COM, SOAP):

- **Outlook Add-In Component** – a component (implemented using MS Add-In Designer, MS Outlook Object Model type-library and Collaboration Data Objects/CDO library) consist of in-proc COM-component and resources DLL. The component contains the following subcomponents:
 - **Login dialog** – a dialog to request user credentials to login to Amdocs CRM server;
 - **Customized Toolbars** – additional buttons on the toolbar;
 - **Customized Forms** – extended forms of Contact, Calendar Items, Tasks (inherited from standard ones);
 - **Options Dialog Extension** – extra Tabs on the standard Outlook Tools\Options dialog;
 - **Event Processor** – code responsible for capturing events inside Outlook and launching synchronization service.
- **Outlook Sync Service** – a set of modules implemented by the synchronization scheduler, synchronization manager, SOAP/web services access support, logging, and local storage (cache);
 - **Scheduler** – Orders and prioritizes synchronization calls (scheduled background synchronization processes as well as those manually initiated by Outlook and web application users). Also replicates auxiliary/locally cached objects;
 - **Synchronization manager** – performs the actual synchronization between Outlook and Amdocs CRM;
 - **SOAP support** – based on WinInet library, it’s a module with custom modifications providing compatibility between Microsoft SOAP and Java SOAP; provides HTTPS secure transmission ability;
 - **Log manager** – records and manages synchronization logs and process including error logs and data conflicts etc.
- **Local Storage (Cache)** – is organized as a set of files that are saved in a special directory under the user’s profile. Thus each user works with his own storage only. It is also possible to maintain a single storage location for all users.
- **Local HTTP server** is used to leverage the soft launch of Outlook forms within Amdocs CRM web pages as well as when implementing background Outlook-targeted actions from within Amdocs CRM web applications (e.g. Push to Outlook). For this, a special application level protocol over HTTP has been created. Because of this, the major concern is maintaining security while implementing this network service; the component is hard coded to accept only TCP-packets from the loop back interface



- **Administration Manager** – settings, local storage parameters, and other administrative data stored on the client side;

There are two installers, one for the server-side components. The second is a self-installing program to be run on the user's desktop. These installers come with default settings for synchronization and other functions. Default settings are based on out-of-the-box Amdocs CRM and Outlook functionality and data structures whenever there are Amdocs CRM or Outlook data structures, UI parts, or logic that have been updated. However, client side applications remain virtually untouched, with the customization being performed on the server side.

The following auxiliary objects are to be stored in the local cache (local storage):

- Site
- Account
- Opportunity
- Lead
- Contact's preferred contact mode
- Contact's preferred contact time
- Contact Status
- Action Item's type
- Action Item's Priority
- Action Item's Status

Local storage files are located in the folder `${service data folder}/Data/`.

List of files:

- sites.xml;
- accounts.xml;
- leads.xml
- opportunities.xml
- guides.xml – contains different primitive objects (countries, states, statuses, etc.)

Configuration files are located in the folder `${service data folder}/Settings/`.

List of files with stored options:

- system.config
 - primary database;
 - form types (Outlook/Amdocs CRM);
 - Amdocs CRM user;
 - user's password in encrypted form (if stored);
 - background synchronization parameters (synchronization time intervals in seconds);
 - other user defined synchronization parameters;
- webservices.config
 - Amdocs CRM server url;
 - relative url's of web services;
 - version of Amdocs CRM server;
- mapping.config
 - Amdocs CRM <-> Outlook field mapping rules;
- dynamic.config
 - background synchronization timestamps;

Log files are located in the folder `${service data folder}/Log/`.

In order to integrate, the client and server need to be able to transport value objects representing business data, SOAP messaging extension of HTTP protocol allows for transport of arbitrary objects, transformed into XML format over HTTP. The use of web-services is a common way to provide interoperability across various applications, developed and deployed using different platforms, which is the case for Amdocs CRM and Outlook Integration solution.



invisibleCRM
www.invisiblecrm.com

A Web service can make itself available to potential clients by describing itself in a Web Services Description Language (WSDL) document. A WSDL description is an XML document that provides all the pertinent information about a Web service, including its name, the operations that can be listed and the parameters for those operations, and the location of where to send requests. A consumer (client-side synchronization service) can use the WSDL document to discover what the service offers and how to access it.

Web-services are actually specially developed Java interfaces and classes plus a WSDL document.

To enable the optimum interoperability between the synchronization service and Amdocs CRM, web-services, being in-between, should serve as EJB-clients. When the client side request is received the application server that hosts the web-service maps XML-objects to Amdocs CRM XV Objects and initiates the requested web-service operation. The next step is to call EJB's deployed under the same application server. EJB's are either standard Amdocs CRM EJBs or custom EJB wrappers to Amdocs CRM IGBean components that implement integration business logic calls for save operations. The latter type of EJB's may be generated atop IGBeans by using Amdocs CRM *ejbgen* utility (with `-beans` switch on Amdocs CRM 6 to keep solution compatible with Clarify 12.5). Thus, in order to implement this function (a remote procedure call to operate on CRM data) we need to construct a server-side component chain comprised of the following parts : IGBean, EJB, Web-Service. We will refer to those chains as the I/E/W chains. Fig.1 reveals high-level design of OI4A Server components, their deployment and interoperability with Amdocs CRM components.

AmdocsCRM and OI4A Server components

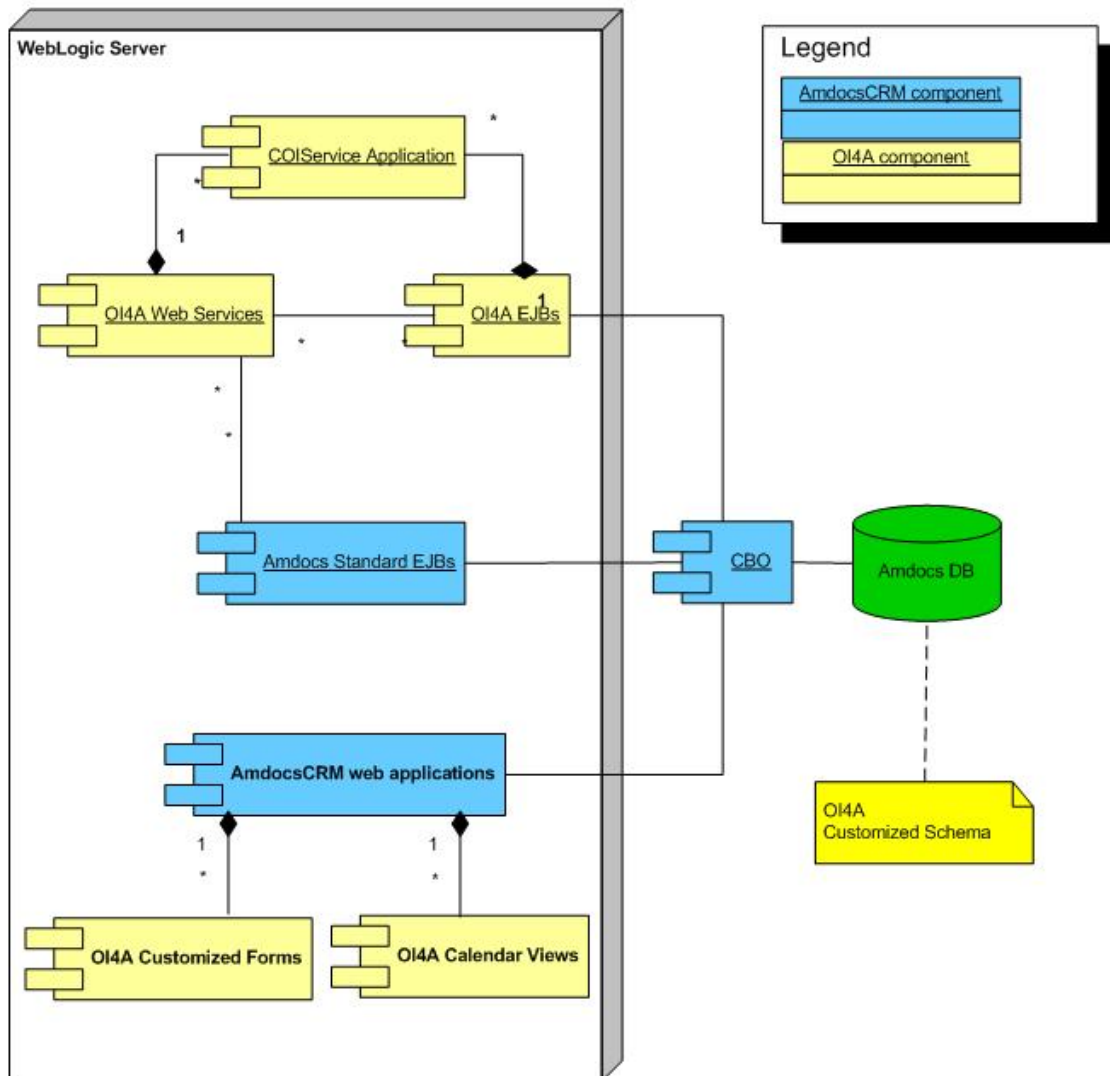


Fig.1 Amdocs CRM and OI4A Server components: architecture and interoperability

OI4A adds a custom table x_contact2user, which is related to table_contact table, to the generic Amdocs database. Customization is to be done via general use of DD Editor utility.

In general the data and control flow between components is as follows:

- The login dialog is used to allow the user interface to request credentials from the user in order to login to Amdocs CRM. After being acquired, credentials are passed to the Sync Service so that a persistent synchronization session can be established.
- Settings defined by default are stored in the local storage. The user can change these settings using a Tab in the Options dialog of Outlook. The Add-In component updates the local storage.
- The Add-In component tracks all user actions with fields and buttons on the customized toolbar and forms. Based on the command received, the component initiates login/logout, automatic or manually forced synchronization, or updates records in the Outlook's internal storage and local storage.

- The Customized Forms are available in both online and offline mode. Requirements for link-fields reference data are extracted from the local cache; this is also called auxiliary data. The synchronization of locally stored objects is simple because they are not changed on the client side.
- The Scheduler, which is active while Outlook is running, initiates the Sync Service according to the synchronization schedule.
- The Sync Service routine is invoked by the Scheduler according to the determined interval. It is designed to keep objects marked as “to be synchronized” in both Outlook and Amdocs CRM and to update the Contacts list according to the installed filters.
- The Sync Service uses the local storage to retrieve data about changes to be synchronized to the server.
- The Sync Service uses SOAP (XML, HTTP(S)) calls to the remote web services to send changes from the desktop and to receive updates from the corporate database. The component uses the SOAP over HTTP methods to contact those deployed on the Amdocs CRM server web services; each object is then serialized and de-serialized
- CRM-specific data is stored inside Outlook’s pst-files and local storage. Customized Outlook Forms enable the user to view and edit this data.
- Sync Service synchronizes standard and added fields of Outlook objects to the server using mapping rules.
- All exceptions and collisions and errors are logged by the Sync Service.
- All requests from the client components (mostly Sync Service) are intercepted by web services wrapper objects. They handle interaction with Amdocs CRM by calling IGBs (Integration Gateway Beans) for save operations or standard Amdocs EJBs and CBOs for get operations.
- Customized Amdocs CRM web pages activate web services to push contacts to the client-side Outlook so that information is returned from the client-side web pages to the Outlook Add-In component. The web page and Outlook can be opened in different boxes using the same user id.
- When the user clicks on the ‘Push to Outlook’ button in Amdocs CRM’s Contact web form, the local web-server should be POST-requested. It parses the input parameters and calls the local synchronization service, which saves the newly created object to Outlook database and triggers one-way single object synchronization from Outlook to Amdocs CRM.
- The “calendar view” button extends user interface to CRM Action Items The UI area of the Calendar control is divided into three areas: navigation areas, events view area, and discreteness area (fig.3).



The following schemas summarize this overview by visualizing the idea of components decomposition and their relationships:

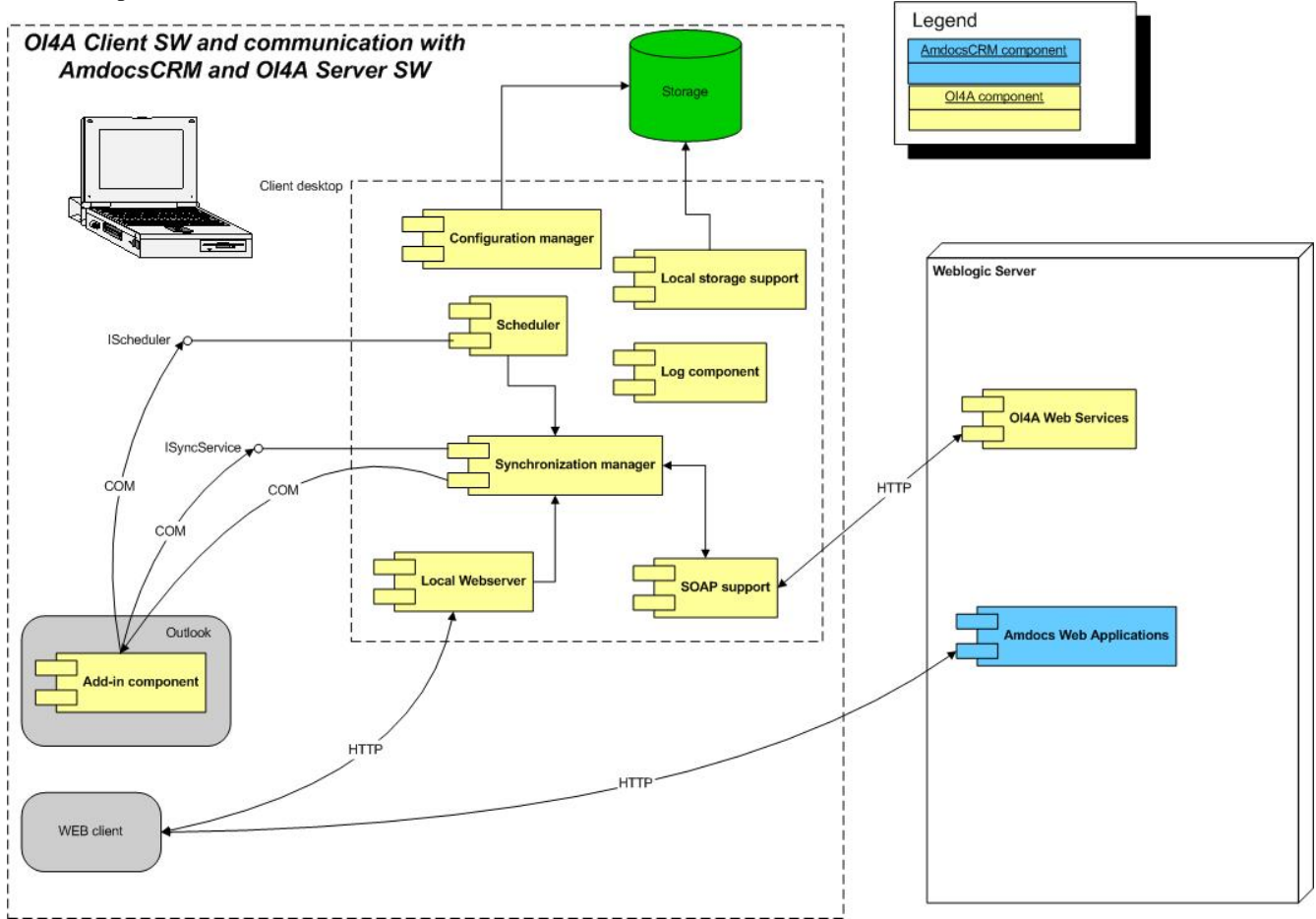


Fig. 1. OI4A client SW architecture and interoperability with OI4A Web Services



Fig. 3. Calendar views UI

2.2 Object synchronization scenarios

The major task when integrating two independent complex software units, such as Amdocs CRM and Outlook, is to create a precise object mapping scheme that respects the independence of each system and allows for bi-directional data exchange. We present the following object mapping scheme below; Common issues and ways of conflicts resolving are later outlined.

OI4A: Outlook <-> AmdocsCRM object mapping scheme

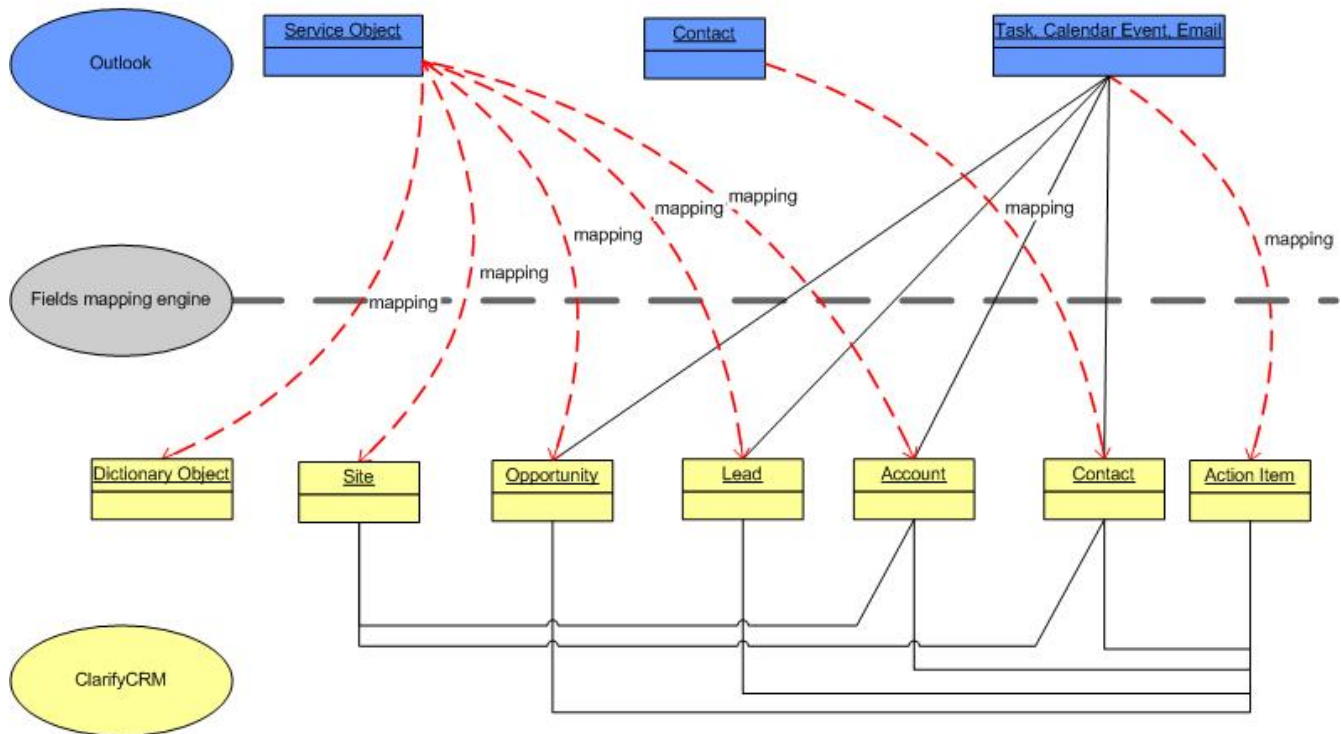


Fig. 4. Objects mapping scheme

The common issue when synchronizing objects across different systems is to provide the correct field mapping engine. The current approach is to keep all operations, related to field mapping on the client-side. That is why this function is implemented inside the Sync Service.

The following pairs of objects are used:

- Outlook Contact <-> Amdocs CRM Contact

There is CRM-to-Outlook contacts synchronization initially launched via export operations. Afterwards bi-directional synchronization goes on.

- Outlook Task <-> Amdocs CRM Action Item

The bi-directional synchronization starts from an established “synchronization date”. Action Items created with the type “To Do” are supposed to become Outlook tasks.

- Outlook Calendar Event <-> Amdocs CRM Action Item

The bi-directional synchronization starts from an established “synchronization date”. Action Items created by types, other than “To Do”, are supposed to become Outlook Calendar Items.

- Outlook Email <-> Amdocs CRM Action Item

Manual export of inbound / outbound messages is launched via Export (Send and Export) functionality. Opportunities and Accounts may be attached to the messages before synchronization with CRM. Contacts, leads known to the CRM, mentioned in To & From, are automatically linked to created Action Items.

Posted and accepted Outlook Meeting requests / Task requests are also synchronized with Amdocs CRM. They can be associated with Contact, Lead, Account or Opportunity in the same way as tasks and appointments.