

Architecture

Microsoft Dynamics CRM 4.0

Offline and Online Synchronization in Microsoft Dynamics CRM

White Paper: “Nuts and Bolts” Series

Date: August 2008

**Acknowledgements**

Initiated by the Microsoft Dynamics CRM *Engineering for Enterprise* (MS CRM E2) Team, this document was developed with support from across the organization and in direct collaboration with the following:

**Key Contributors**

Andrei Smertin (*Microsoft*)

Navin Thadani (*Microsoft*)

Dominic Pouzin (*Microsoft*)

**Technical Reviewers**

Roger Gilchrist (*Microsoft*)

Praveen Upadhyay (*Microsoft*)

Shashi Ranjan (*Microsoft*)

The MS CRM E2 Team recognizes their efforts in helping to ensure delivery of an accurate and comprehensive technical resource in support of the broader CRM community.

MS CRM E2 Contributors

Amir Jafri, Program Manager

Jim Toland, Content Manager

**Feedback**

Please send comments or suggestions about this document to the MS CRM E2 Team feedback alias ([entfeed@microsoft](mailto:entfeed@microsoft.com)mslogo[.com](mailto:entfeed@microsoft.com)).

Microsoft Dynamics is a line of integrated, adaptable business management solutions that enables you and your people to make business decisions with greater confidence. Microsoft Dynamics works like and with familiar Microsoft software, automating and streamlining financial, customer relationship and supply chain processes in a way that helps you drive business success.

U.S. and Canada Toll Free 1-888-477-7989

Worldwide +1-701-281-6500

[www.microsoft.com/dynamics](http://www.microsoft.com/dynamics)

**Legal Notice**

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

This White Paper is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO THE INFORMATION IN THIS DOCUMENT.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

© 2008 Microsoft Corporation. All rights reserved.

Microsoft, Microsoft Dynamics, Microsoft Dynamics Logo, and Microsoft Office Outlook are trademarks of the Microsoft group of companies.

All other trademarks are property of their respective owners.

**Table of Contents**

[Introduction 4](#_Toc206998066)

[Overview of Offline and Online Synchronization 5](#_Toc206998067)

[Overview of the Offline Client 5](#_Toc206998068)

[Offline Client Components 5](#_Toc206998069)

[Offline Client Modes 5](#_Toc206998070)

[Online Mode 5](#_Toc206998071)

[Offline Mode 6](#_Toc206998072)

[Transitioning between Modes 6](#_Toc206998073)

[The Microsoft Dynamics CRM 4.0 GoOffline Process 7](#_Toc206998074)

[Step 1: PrepareSync 8](#_Toc206998075)

[SyncEntry Table 8](#_Toc206998076)

[Sync State Tracking 8](#_Toc206998077)

[PrepareSync API 9](#_Toc206998078)

[Step 2: Propagate Schema Changes 10](#_Toc206998079)

[Step 3: Move Data 11](#_Toc206998080)

[BCP Data Files 11](#_Toc206998081)

[HTTP Handler 11](#_Toc206998082)

[PostSync API 12](#_Toc206998083)

[RecordSyncInfo API 12](#_Toc206998084)

[Maintaining Data Integrity during the GoOffline Process 12](#_Toc206998085)

[Working in Offline Mode 13](#_Toc206998086)

[The Offline Queue Table 13](#_Toc206998087)

[Recording Offline Operations 14](#_Toc206998088)

[The Microsoft Dynamics CRM 4.0 GoOnline Process 15](#_Toc206998089)

[Synchronizing Offline Changes to the Microsoft CRM Database 15](#_Toc206998090)

[Playback of the Offline Queue Table 15](#_Toc206998091)

[Maintaining Data Integrity during the GoOnline Process 16](#_Toc206998092)

[Appendix A: Additional Resources 17](#_Toc206998093)

[Appendix B: Term List 18](#_Toc206998094)

[Appendix C: Dynamic Batching 19](#_Toc206998095)

[Frequently Asked Questions 20](#_Toc206998096)

[Functionality 20](#_Toc206998097)

[Security 21](#_Toc206998098)

[Deployment 21](#_Toc206998099)

[Filtering Local Data 21](#_Toc206998100)

# Introduction

**CRM E2 Nuts and Bolts SeriesOverview**

The MS CRM Engineering for Enterprise (E2) *Nuts and Bolts* (NB) series is designed as an expanding set of topical articles, each of which provides detailed information about the internal mechanisms behind a specific area of MS CRM 4.0 functionality, such as the “GoOffline” process, Workflow architecture, or the security/authentication model. Topic selection is driven by many factors, not the least of which is the frequency of related queries to technical aliases.

Articles in the NB Series are designed to provide detailed technical resources that:

* Address often repeated queries to Technical aliases
* Consolidate answers, links, etc., that are generated in response to those queries
* Offer multiple levels of complementary information to support a broader, multi-perspective understanding of the topic
* Convey the baseline “principles” users require to begin to address related but tangential technical queries
* Present content using a consistent structure and “look and feel”

**Audience**

The target audience of the NB Series includes (but is not limited to):

* Solution Architects
* Application Architects
* Infrastructure Architects
* Consultants
* Developers

**NB Article Content and Structure**

Articles in the NB Series are designed to accommodate three independent but complementary levels (or “tiers”) of information:

* *Core*. High-level, architectural information; “schematic-level ” view of functionality; provides contextual overview / baseline
* *FAQ*. Frequently asked questions/answers; common queries (from Field/Partners) about details of functionality
* *Scenario-based*. Detailed explanations about how to address unique scenarios; practical details about resolving issues or accomplishing specific “real-world” tasks

Each article’s initial release will ideally include Core and FAQ information, as well content that supports some portion of the Scenario-based tier. However, because each tier targets an independent information need, NB articles offer the flexibility of releasing information as it become available, rather than delaying release in anticipation of any remaining components.

**Important**: This release of the NB article *Offline and Online Synchronization in Microsoft Dynamics CRM* provides:

* *High-level, architectural information*
* *Frequently asked4 questions/answers*

Detailed explanations about how to address unique scenarios regarding offline and online synchronization will be released as they become available.

# Overview of Offline and Online Synchronization

Microsoft Dynamics CRM 4.0 uses two processes to manage replication of information between the central Microsoft Dynamics CRM database and the local data store on a computer running Microsoft Dynamics CRM for Microsoft Office Outlook with Offline Access (the “offline client”).

The *offline synchronization* (GoOffline) process manages replication of information from the Microsoft Dynamics CRM database to the offline client. On the other hand, the *online synchronization* (GoOnline) process, manages replication of changes in the local data store from the offline client to the Microsoft Dynamics CRM database when the offline client reconnects to the network.

**Note**: The offline client also supports synchronizing contacts and activities to Outlook folders, or Outlook Synchronization, which enables Microsoft Office Outlook users to view CRM information including contacts, tasks, phone calls, letters, faxes, appointments, and E-mails.

# Overview of the Offline Client

The offline client is designed for mobile professionals who often travel with their computers and who require access to CRM data while they are disconnected from the Microsoft Dynamics CRM server. These users also need to ensure that local CRM data is synchronized with the Microsoft Dynamics CRM server when they subsequently reconnect.

## Offline Client Components

To provide this functionality, the offline client requires components in addition to those installed with Microsoft Dynamics CRM for Microsoft Office Outlook (the “online client”).

**Important**: The online client is outside the scope of this article.

Installing the offline client also provides local instances of the Microsoft Dynamics CRM platform logic, a Web server, and SQL Server 2005 Express Edition.

**Note**: SQL SERVER 2005 EXPRESS limits the size of the database to four gigabytes.

## Offline Client Modes

The offline client can function in *online mode*, when the offline client is connected to the CRM server, or in *offline mode*, when the offline client is disconnected from the CRM server.

**Important**: The offline client can transition between online and offline modes only if the associated user’s security role provides the “Go Offline” privilege. Without this permission, offline client users are limited to working in online mode.

**Note**: In offline mode, the client computer can be running an instance of Outlook that is still actively connected to Exchange.

### Online Mode

The following conditions apply when the offline client operates in online mode:

* All business and application logic processing occurs on the Microsoft Dynamics CRM server.
* All database activity is performed against the Microsoft Dynamics CRM database; the local SQL Server 2005 Express database is not used in online mode.
* If the offline client loses connectivity to the server, the user can work offline on previously synchronized data.
* Users can configure a background process to periodically synchronize the offline client’s local data store, which will improve the efficiency of the GoOffline process.
* Administrators can use a Microsoft Dynamics CRM server global setting to specify the frequency with which automatic synchronization updates the offline client’s local database.

### Offline Mode

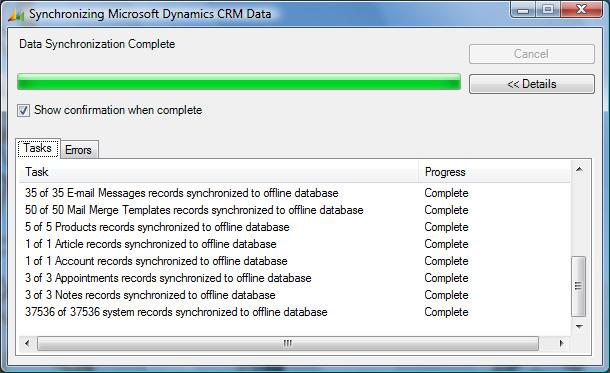
The following conditions apply when the offline client operates in offline mode:

* All business and application logic processing is performed on the client computer; the application logic uses a local instance of the SQL Server 2005 Express database, platform code, and business logic.
* All Microsoft Dynamics CRM forms are rendered by the client's local web server.
* All database activity is performed against the local SQL Server 2005 Express database.
* All “write” requests are accompanied by a second update to a local data store that is used to synchronize the local SQL Server 2005 Express database with the Microsoft CRM Dynamics database when the offline client goes online i.e. is synchronized with the Microsoft CRM database.

**Note**: Some operations that are available online mode, such as Convert Order to Invoice, are not available to users in offline mode. In addition, most customization features, as well as security operations such as changing roles, privileges, or organization settings, are available only in online mode. Other features, such as workflow, run upon reconnection to the server.

### Transitioning between Modes

Offline client users can initiate the GoOffline process and transition from online mode to offline mode either by clicking on the **Go Offline** button or by selecting **Go Offline** from the **CRM** menu. The progress and eventual completion of the GoOffline process is displayed in the **Synchronizing Microsoft Dynamics CRM Data** dialog box, as shown below:



To initiate the GoOnline process and transition from offline to online mode, offline client users must connect to the Microsoft Dynamics CRM server and then click on the **Go Online** button.

# The Microsoft Dynamics CRM 4.0 GoOffline Process

The Microsoft Dynamics CRM 4.0 GoOffline process involves three primary steps:

1. **PrepareSync**. In response to a client request, the server prepares a subset of Microsoft Dynamics CRM database records for transfer to the offline client.
2. **Propagate schema changes**. The offline client assesses and applies any customizations that have been applied to the server.
3. **Move data**. In response to a client request, the server sends a “batch” of records to the offline client. After the records are added to the local database, the client issues another request for data, and the process repeats until all data is successfully downloaded.

Each primary step in this process involves one or more sub-steps, as illustrated in the following graphic:



The following sections provide additional detail about the Microsoft Dynamics CRM 4.0 GoOffline process.

## Step 1: PrepareSync

The PrepareSync step leverages database-driven security and user filter settings to evaluate the Microsoft Dynamics CRM database and determine which records are subject to a GoOffline request from a specific instance of the offline client. To generate the offline dataset, Microsoft Dynamics CRM 4.0 uses several internal components and processes.

### SyncEntry Table

A SyncEntry table maintains all the information about replication data that is associated with a specific offline client subscription. An *offline client subscription* is associated with each instance of the offline client and allows syncing of data from CRM to the local SQL Express database. The CRM database maintains a separate SyncEntry table each for offline client subscription; table data varies based on security and the offline data filters associated with a specific user.

**Note**: The CRM database also maintains a separate SyncEntry table for each of the other types of client subscription, which are explained in the following table:

|  |  |
| --- | --- |
| Client Subscription Type | Purpose |
| **Outlook client** | Allows online or offline client to sync data (task, phone call, letter, fax, contacts, appointments) changes from CRM to the local instance of Outlook |
| **ABP (AdressBookProvider)** | Allows online or offline client’s CRM address book provider to sync data (contacts, accounts, lead, system user, equipment/facility) changes from CRM to the Outlook ABP |
| **DM (DataMigration)** | Allows the data migration wizard to sync data during the data migration process |

The Outlook client, ABP, and DM client subscription types are outside the scope of this article.

To accommodate the information associated with each replicated object, the SyncEntry table includes four columns, which are described in the following table:

|  |  |  |
| --- | --- | --- |
| Column Heading | Data Type | Description |
| **ObjectId** | guid | ID |
| **ObjectTypeCode** | int | CRM entity type |
| **SyncState** | int | State (synched, to be synched, etc.) |
| **VersionNumber** | timestamp | The version of the record that has synched to the client (used to detect changes) |

### Sync State Tracking

While the SyncEntry table uniquely identifies a replicated object and its CRM type, capturing timestamp / sync state data is vital to accurately tracking row-level changes. In each replicated table, the **VersionNumber** column captures the object’s SQL timestamp and the **SyncState** column captures a numerical value representing a replicated object’s sync state:

* **0** - Successfully moved to client
* **1** - Needs to be moved to client; object might already exist on client computer
* **2** - Needs to be deleted from client; object might not exist on a client

### PrepareSync API

After evaluating CRM security and user-defined filters, the PrepareSync API populates a temporary #SyncEntry table and returns information about the number of records to be synced, on a per entity basis, in xml format. For example:

<result>

<entity name=”account” delete\_count=”5” insert\_count=”20”>

<entity name=”contact” delete\_count=”1” insert\_count=”12”>

<entity name=”lead” insert\_count=”1200”>

</result>

The logic used by the PrepareSync API is illustrated in the following diagram:



For each entity:

1. *Populate #SyncEntry*. Evaluate offline filters based on user’s security and insert IDs into a temporary #SyncEntry table
2. *Merge deleted records*. For records that are in the SyncEntry table but not in the temporary #SyncEntry table, change to SyncState = 2 (to be deleted)
3. *Merge updated records*. For records that are in both the SyncEntry and #SyncEntry tables, change to SyncState = 1
4. *Merge inserted records*. For records that are in the #SyncEntry table but not in the SyncEntry table, insert records with SyncState = 1

After these changes, information about inserted, updated, and deleted records becomes available on the server for a subscription at the time of PrepareSync execution.

## Step 2: Propagate Schema Changes

A major advantage of Microsoft CRM is the ability to customize the application to meet the needs of a specific business scenario. During offline synchronization, customizations that have been applied to the server must be propagated to the offline client. To manage this process, Microsoft Dynamics CRM 4.0 leverages the following internal components:

* **Metadata Helper**. Responsible for any changes in CRM schema; component also used for schema changes during upgrade and import/export.
* **DiffBuilder**. Compares two metadata caches, resulting in a sequence of requests to Metadata Helper.
* **Metadata Cache**. Serves as a common denominator by providing different ways to load the CRM metadata via Web Service, Database, and XML files.

The following diagram illustrates the logic used during propagation of schema changes:



To propagate schema changes, the DiffBuilder component:

1. Loads server metadata through Web Service into one instance of the Metadata Cache
2. Loads current client metadata from the CRM database into a second instance of the Metadata Cache
3. Compares the two sets of cached data to determine differences in Entities, Attributes, and Relationships
4. Submits requests based on these differences to Metadata Helper to ensure that the local database includes the same level of information as does the CRM database

## Step 3: Move Data

After PrepareSync and downloading server customizations to the offline client, the last step in the offline synchronization process is to move the data to the offline client’s local data store. The Move Data step uses the following logic:

1. Server receives from offline client a web request to download a BCP data file
2. Internal handler calls the GenerateSyncData() API and inserts header information into the appropriate BCP file
3. HTTP handler streams the BCP data file to the offline client
4. Offline client uses BULK INSERT command to insert relevant information into the appropriate tables in the client’s local database

### BCP Data Files

#### BCP data files contain information (in native SQL format) that is transferred to the offline client and inserted into tables in the client database. To generate BCP files, entity tables are joined with the SyncEntry table (which was populated correctly during the Prepare Sync step) to retrieve the data that needs to be transferred to the offline client. Note that only records marked as SyncState = 1 are retrieved from Entity tables; for records marked as SyncState = 2, the server only returns IDs. After the information contained in a BCP data file is transferred to the offline client database, the file is deleted from server.

**Note**: BCP data files are read-accessible only to the owner of the associated subscription. They are stored on the CRM Web server in the install folder Server\OfflineData and use the naming convention <*SubscriptionId*>.bin (where *SubscriptionId* is the GUID assigned to the user subscription). On the client computer, BCP data files are stored in the folder AppData\Microsoft\MSCRM\BCP.

### HTTP Handler

#### The HTTP handler is responsible for transporting BCP data files over the network. The CRM Web server implements a Web service that the offline client can use to download BCP data files from the server.

#### The offline client generates web requests to the server regarding downloading BCP data files. For example an offline client’s web request might use the following syntax:

[http://server/MSCRMServices/OfflineSync.ashx?SubscriptionId={7D4F38B5-BC58-46CD-B3AB-721DEE532F86}&EntityName=Account&Action=2&BatchSize=0](http://server/MSCRMServices/OfflineSync.ashx?SubscriptionId=%7b7D4F38B5-BC58-46CD-B3AB-721DEE532F86%7d&EntityName=Account&Action=2&BatchSize=0)

When the server receives the web request from the offline client, an internal handler generates the BCP data file (as described earlier) and then returns the file to the offline client in the WebResponse. Before sending the BCP data file, however, the internal handler inserts a header in the following format:

// BCP file header

// ----------------------------------------------

// | hr | file\_size | metadata\_version |

// ----------------------------------------------

// hr - 4 bytes. Indicates result code of request. S\_OK - successful.

// file\_size - 8 bytes. File size.

// metadata\_version - 4 bytes. Metadata version

The offline client uses the information in this header to determine possible server errors and to validate that the client metadata version matches that on the server, which helps to avoid potential failures during the Bulk Insert process.

### PostSync API

The PostSync API manages the SyncState after data has been moved to the local data store on an offline client. The PostSync API modifies the SyncEntry table by deleting rows with SyncState value of 2 (since those rows are now deleted from the client) and by changing to 0 the SyncState value of rows with a SyncState = 1 (since they are now synched).

The PostSync API supports batching, or moving data and marking it synched in batches. Batching provides a number of benefits, including:

* Preventing timeouts for large amounts of moved data
* Making move data step configurable and more resilient to network connectivity failures
* Increasing server scalability

**Note**: Rather than a single, larger operation, batching uses several smaller operations. As a result, the Move Data step takes longer to complete.

Batch sizes are determined by the number of rows to be deleted from/inserted to the client database; for each client computer, batch size is configurable, the configuration being saved in client registry as dword OfflineRowsBatchSize.

Successful completion of a batch step calls the platform PostSync() API. If there is an intermittent failure during data transfer, the batched PostSync ensures that the same data is not downloaded again, effectively providing for error recovery.

The number of retries the client computer attempts is also configurable, with the information being stored in the client registry as dword OfflineMaxRetryCount.

### RecordSyncInfo API

The RecordSyncInfo API populates the SubscriptionSyncInfo table with basic statistics about the results of the GoOffline process.

## Maintaining Data Integrity during the GoOffline Process

Should a connection failure occur during the GoOffline process, the process fails. When a connection is restored and the user initiates the GoOffline process again, the offline client will re-attempt the process.

However, there is little chance of data corruption, because during the GoOffline process, Microsoft Dynamics CRM uses existing web services to create the offline data set and copy it to the client's local SQL SERVER 2005 EXPRESS database. Should a connection failure occur during this process, the client fails to go offline. After the connection is restored and the user re-attempts to go offline, the replication process starts over and any data that has already been replicated (indicated by the batched PostSync) is not replicated again.

# Working in Offline Mode

To maintain the fidelity of security roles, etc. during the GoOnline process, the offline client requires a mechanism for:

* Capturing write operations that change data when the client operates in offline mode
* Playing those operations back to the server without bypassing the platform (pipeline) when the offline client transitions to online mode.

## The Offline Queue Table

To provide for this functionality, in offline mode the client’s local platform layer makes an additional database write for each successful modification of the data in the local SQL Server 2005 Express database. This additional write is captured and saved as an entry in the local Offline Queue table, which tracks all modifications, including Create, Update, and Delete operations, against each object.

**Note**: The Offline Queue table does not record rejected transactions and query transactions.

**Important**: Because the CRM SDK is SOAP-based, the offline client captures each SOAP request to the offline platform that results in a modification to the database. The SOAP request is played back to the CRM server as an SDK request when going online. This guarantees that the request will go through the CRM server pipeline, enforcing the security policies, business logic, workflow rules, plug-ins, etc. that are in effect at the time the GoOnline process occurs.

The schema of the Offline Queue table is presented in the following table:

| Column Name | Data Type | Description |
| --- | --- | --- |
| ObjectId | UniqueIdentifier | An identifier indicating the entity associated with the SOAP packet; column may not always contain valid data |
| ActionDate | Datetime | The timestamp when the SOAP packet was inserted into the table |
| CommandId | Smallint | An identifier indicating the type of operation of the SOAP packet ex:- Create, Update, Delete etc. |
| Data | Ntext | The SOAP packet (xml) |
| ParentId | Nvarchar(100) | An identifier used to match up start and stop markers in the offline queue |
| HttpHeader | Ntext | The SOAP Action header |
| Url | Nvarchar(500) | The name of the file (srf or asmx) to which the SOAP packet was sent |
| RowNumber | Bigint | Identity column uniquely identifying each row in the offline queue |
| ObjectTypeCode | Int | The object type code of the entity on which the operation was carried out |
| State | Int | An identifier the sync queue uses to track which SOAP packets were played back successfully and which need to be cleaned up in the next go offline |

## Recording Offline Operations

In Microsoft Dynamics CRM 4.0, recording offline operations involves the following process.

1. The offline client connects to the local web service that is hosting ASP.NET inside the Microsoft.Crm.Application.Hoster process.

**Note**: For more information about hosting ASP.NET by using the ASP.NET hosting APIs (System.Web.Hosting), see the ASP.NET Cassini Web Server Sample (a sample web server written using the Microsoft® .NET® Framework) at  
<http://www.asp.net/downloads/archived/cassini/>)

1. The platform registers with ASP.NET a SOAP extension to be called each time the application makes a SOAP request of the offline platform. The SOAP extension captures the incoming SOAP packet, URL, and Http Header, and stores the information in memory.

**Note**: For additional information about SOAP, in the .Net Framework Developer Center, see *SoapExtension Class* at  
(<http://msdn.microsoft.com/en-us/library/system.web.services.protocols.soapextension.aspx>)

1. The offline client also registers with the CRM SDK plug-in infrastructure an offline-only plug-in that is responsible for injecting primary keys into the Entity (when Create\Update calls are made) and regenerating the SOAP packet.

This enables the creation of entities with the same primary key when the SOAP packet is played back to the server and, and it avoids the need to resolve SOAP packets that update the same entity during the same offline session. The plug-in uses metadata to determine the primary key fields to inject. It can also handle nested entities such as activity parties.

1. If additional information is required for successful playback, the business logic layer (running inside the offline platform) uses its own mechanism to inject that information in the SOAP packet so that the SOAP packet plays back appropriately.

For example, consider the Email promote APIs, which do not provide primary keys for activity parties, attachments etc. Since the SOAP packet does not contain a parameter that can be populated by the plug-in referenced in step 3, the business logic layer uses the OptionalParameters section of the CRM SDK Request to inject such data. This data is read by the online platform when the request is played back. Internally, the platform uses an optional parameter (OfflineDataOptionalParameter) for this purpose.

**Important**: Misuse of the OfflineDataOptionalParameter parameter by ISVs or others can result in undefined behavior.

1. The platform inserts into the Offline Queue table the SOAP packet and other information injected by the business logic layer, together with the appropriate operation type, every time a modification is made to the offline database.

**Note**: The insert into the Offline Queue table occurs on the same transaction as the rest of the operations in the platform to ensure fidelity of the offline queue in the event that an error occurs during an operation, causing the transaction to be aborted.

1. After inserting the SOAP packet in the Offline Queue table, the SOAP packet is marked as captured to prevent multiple insertions of the same SOAP packet into the Offline Queue table should a specific operation require multiple modifications to the database against a single web request.

# The Microsoft Dynamics CRM 4.0 GoOnline Process

At some point, offline users must synchronize changes made to their local SQL SERVER 2005 EXPRESS database with the Microsoft Dynamics CRM database.

## Synchronizing Offline Changes to the Microsoft CRM Database

During synchronization, offline database changes are replayed to the Microsoft CRM server. However, unlike the process that occurs when disconnecting from the server, synchronizing an offline client to the server also requires that Microsoft Dynamics CRM:

* Guarantees that the latest Microsoft CRM business logic is performed on every offline action submitted to the Microsoft CRM Server.
* Uses a method of “playing back” the changes as they had occurred offline. Each update transaction is submitted from the Offline Queue table to the Microsoft CRM Server as if the user entered the transaction in the web browser at that moment.

This play back method guarantees that all business logic is performed on every offline transaction submitted to the Microsoft CRM server. The data must be validated by the Microsoft CRM server's business logic even though it was validated by the client's business logic when the data was originally entered. This is because a user's security privileges might have changed after taking the client offline. Therefore, objects to which they originally had access might have been modified and rendered the access obsolete. Additionally, performing the Microsoft CRM Server's business logic on each played back transaction also guarantees that all objects work within current workflow and plug-in process rules.

## Playback of the Offline Queue Table

When the offline client reconnects to the network and goes online, SOAP packets in the Offline Queue table are played back to the server. The application uses start and stop markers to transact certain operations in the application that perform multiple operations in the platform (for example, reopening a case, which requires copying the case line items to the new case). In case of a failure in playing back any operation between a start-stop marker, all the remaining operations are aborted and the sync queue moves to the next stop marker.

Attachments are also captured as part of the SOAP packet that is stored in the offline queue. As a result, each attachment occupies twice the space in the database when created offline. As part of the process of SOAP packet playback, prior to playback the SyncQueue component modifies the SOAP XML to inject:

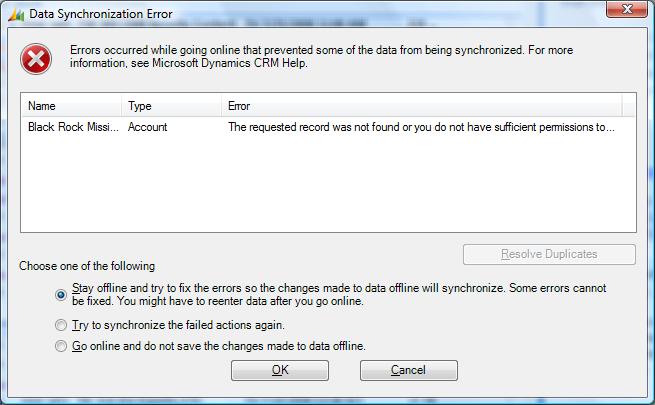
* The Authentication token, which is required of all SDK requests. For CRM Online and SPLA/IFD, the Authentication ticket is time sensitive, expiring after a fixed period, so the Authentication ticket is generated and injected into the SOAP packet during playback.
* A duplicate detection optional parameter, which allows the server’s Duplicate detection infrastructure to take appropriate action based on user settings and upon the user’s selection on the client side (if the user chooses to create the duplicate anyways).
* A CallerOrigin SOAP header, which alerts the server that a particular SDK request is coming from an offline queue playback. The header also contains a timestamp of when the SOAP packet was added to the offline queue, which allows plug-ins on the server side to use this information to implement custom conflict resolution mechanisms. Out of the box, Microsoft Dynamics CRM 4.0 does not include built in conflict resolution mechanisms for offline queue playback, and as a result, the last one to playback always wins.

## Maintaining Data Integrity during the GoOnline Process

Should a connection failure occur during the GoOnline process, the process fails. When a connection is restored and the user initiates the GoOnline process, the offline client will re-attempt the process.

However, there is little chance of data corruption, because during synchronization, no transaction is removed from the Offline Queue table until it is successfully replayed to the server (the transaction is either updated on the server or rejected by the server due to changes in security or business processing logic that occurred after the client went offline). When connection is restored following a network failure and synchronization is initiated, the process starts where it left off. This occurs because the transactions remaining in the offline queue are limited to those not processed prior to the network failure.

**Important**: If the offline client encounters errors that prevent synchronization of any Microsoft Dynamics CRM data during the GoOnline process, the **Data Synchronization Error** dialog box appears, as shown in the following graphic:



The dialog box lists the errors encountered during the synchronization process and presents users with the following options:

* Stay offline and try to fix the errors so all the data will synchronize
* Reattempt synchronizing the subject data
* Finish going online and do not save the data changes made offline

# Appendix A: Additional Resources

For additional information related to online and offline synchronization in Microsoft Dynamics CRM 4.0, see the following additional resources:

* Microsoft Dynamics CRM Server Programming Guide: *CrmOutlookService Class (Outlook Sdk Assembly)*  
  <http://msdn.microsoft.com/en-us/library/cc156644.aspx>
* Microsoft Dynamics CRM Server Programming Guide: *CRMRichClientSync Class*<http://msdn.microsoft.com/en-us/library/ms935544.aspx>
* Microsoft Dynamics CRM SDK, *Writing Custom Code for Microsoft Dynamics CRM 4.0  
  for Outlook*  
  <http://msdn.microsoft.com/en-us/library/cc151058.aspx>
* KB 951179: *Error message when you try to go offline in the Microsoft Dynamics CRM 4.0 client for Outlook with Offline Access: "Failed move data for entity"*  
  <https://mbs.microsoft.com/knowledgebase/KBDisplay.aspx?scid=kb$en-us$951179&wa=wsignin1.0>
* KB 951179: *Outlook may stop responding when you exit Outlook, go offline in the Microsoft Dynamics CRM 4.0 client for Outlook, or go online in the Microsoft Dynamics CRM 4.0 client for Outlook*  
  <https://mbs.microsoft.com/knowledgebase/KBDisplay.aspx?scid=kb;en-us;948121>

# 

# Appendix B: Term List

Descriptions of key terms associated with offline and online synchronization in Microsoft Dynamics CRM 4.0 are listed in the following table.

|  |  |
| --- | --- |
| **Term** | **Description** |
| *BCP data files* | During Move Data, responsible for transferring information (in native SQL format) to the offline client and inserting it into the client’s local database. |
| *DiffBuilder* | During Propagate Schema Changes, compares two metadata caches, resulting in a sequence of requests to Metadata Helper |
| *HTTP handler* | During Move Data, responsible for transporting BCP data files over the network; the CRM Web server implements a Web service that the offline client can use to download BCP data files from the server |
| *Metadata Cache* | During Propagate Schema Changes, serves as a common denominator by providing different ways to load the CRM metadata via Web Service, Database, and XML files |
| *Metadata Helper* | During Propagate Schema Changes, responsible for any changes in CRM schema; component also used for schema changes during upgrade and import/export |
| *Offline client subscription* | Associated with a specific instance of the offline client; allows syncing of data from CRM to the local SQL Express database on the client |
| *Offline mode* | State in which the offline client is disconnected from the CRM server |
| *Offline Queue Table* | In offline mode, tracks all modifications, including Create, Update, and Delete operations, against each object; does not record rejected transactions and query transactions |
| *Offline synchronization* | Also known as the GoOffline process; manages replication of information from the Microsoft Dynamics CRM database to the offline client |
| *Online mode* | State in which the offline client is connected to the CRM server |
| *Online synchronization* | Also known as the GoOnline process; manages replication of changes in the local data store from the offline client to the Microsoft Dynamics CRM database when the offline client reconnects to the network |
| *PostSync API* | During Move Data, manages the SyncState after data has been moved to the local data store on an offline client |
| *PrepareSync API* | During PrepareSync, populates the SyncEntry table and returns information about the number of records to be synchronized, on a per entity basis, in xml format |
| *RecordSyncInfo API* | During Move Data, populates the SubscriptionSyncInfo table with basic statistics about the results of the GoOffline process |
| SubscriptionSyncInfo *table* | During Move Data, maintains basic statistics about the results of the GoOffline process |
| *Sync State Tracking* | During PrepareSync, the process of capturing timestamp / sync state data to the SyncEntry Table to accurately track changes at the row level |
| *SyncEntry Table* | During PrepareSync, maintains all the information about replication data for a subscription. A SyncEntry table is created in the server database for each type of client subscription |

# Appendix C: Dynamic Batching

From a performance and scalability perspective, defining “optimal” batch sizes in any scenario can depend on a variety of factors:

* Downloaded dataset and different real row sizes in same table
* SQL/CRM Server load
* Network latency, bandwidth, and reliability
* Client hardware and load

Because many of these factors are dynamic, defining batch sizes statically (by row) is not optimal. Batch sizes require dynamic adjustment, increasing and decreasing as necessary to perform the data download process most efficiently.

The dynamic batching strategy uses two batch threshold times (minBatchTime, maxBatchTime) and two rates (IncreaseRate, DecreaseRate). Each batch step calculates batch size depending on the time of the previous batch step for same table. The initial batch size is the smaller of either the requested countRows or the configuration setting (by default, the value of the registry parameter OfflineRowsBatchSize).

Registry values define configuration parameters in decimal values as follows:

* "OfflineRowsBatchSize"=dword:100000
* "OfflineMinBatchTimeMilliSec"=dword:10000 10 sec
* "OfflineMaxBatchTimeMilliSec"=dword:60000 60 sec
* "OfflineIncreaseRate"=dword:2
* "OfflineDecreaseRate"=dword:2
* "OfflineMaxRetryCount"=dword:3

As an example, for a total of 6296 rows, using this strategy yields the following results:

|  |  |  |
| --- | --- | --- |
| Initial Batch Size | Time of Initial GoOffline - Without Batch-Size Adjustments | Time of Initial GoOffline -  With Batch Size Adjustments |
| 1 | 00:04:59.9179820 | 00:00:37.1709234 |
| 16 | 00:00:49.2503152 | 00:00:28.3904433 |
| 256 | 00:00:27.6554168 | 00:00:26.6245983 |
| 512 | 00:00:26.4057430 | 00:00:26.4682418 |
| 1024 | 00:00:29.9521665 | 00:00:28.3897165 |

# Frequently Asked Questions

## Functionality

Q: Is the PrepareSync process in Microsoft Dynamics CRM 4.0 more efficient than in previous versions of the product?

A: Yes. In the majority of typical scenarios, incremental PrepareSync is proportional to the data changes that occur between synchronizations. As a result, incremental PrepareSync in Microsoft Dynamics CRM 4.0 is between two and ten times faster, and it provides for greater scalability because many times less work is performed in database.

Q: Can synchronization occur without using Outlook? If not, what binds the two?

A: This is not currently supported. Synchronization requires the Outlook process to manage a wide variety of user interactions, such as resolving detection of duplicates or resolving synchronization errors / conflicts.

Q: Why doesn’t CRM work against the offline database even when a user is online?

A: SQL Server Express is a memory intensive process and not using the process when online reduces the memory required on the client computer. In addition, since the background sync process occurs every 15 minutes, the fidelity of the data available in SQL Server Express database cannot be guaranteed at all times.

Q: How do plug-ins run in offline mode?

A: Only plug-ins that have been marked to run while offline are available in offline mode. The plug-in assembly has to be stored in the database and will be synchronized to the SQL Express database as part of the go offline process. Users need to create an entry in the registry to authorize the loading of the assembly by the offline platform.

Q: How is the size of the offline database maintained to prevent the infinite growth?

A: Only subset of server data is replicated into the offline database. Users can define filters to limit the data to be replicated. Local database clean-up occurs automatically to remove all data changed in offline (changes in offline data will be replicated from server again).

Q: How is the offline synchronization processes affected by a connection failure?

A: If network failure occurs when the offline client is going offline or is synchronizing offline database changes with the Microsoft CRM database, the process fails. However, Microsoft Dynamics CRM 4.0 synchronizes data in batches. Batches that synchronize successfully are marked and will not be resynchronized. This functionality is particularly useful for scenarios in which a large data set is being synchronized over an unreliable connection.

Q: Can the offline client be viewed via the web UI instead of Outlook?

A: Yes, but this requires connecting to the correct localhost URL. Note that the port is dynamically allocated; the port number usually is 2525, but it may differ. The value of the port number is tracked in the registry.

## Security

Q: What are the effects of the “Sync to Outlook” and the “Go Offline” privileges?

A: The privileges are not complex and have few repercussions. Removing the Sync to Outlook privilege prevents synchronization of tasks / appointments / contacts to Outlook (no effect on the CRM address book provider, email tagging, etc.). Removing the Go Offline privilege prevents users from going offline / synchronizing their offline store.

Q: Is the GoOnline process secure, and if so, by what mechanisms?

A: Each SOAP packet from Offline Queue played back to server is evaluated as regular SOAP (SDK) calls with all security checks on server.

## Deployment

Q: Can the offline client be deployed via an unattended deployment?

A: Yes. With Microsoft Dynamics CRM 4.0, deploying the offline client by using SMS or Group Policy does not require administrator permissions on the local computers. Note that when deploying the offline client via SMS, the SMS installation must be configured in full UI mode with user interaction to ensure that the installation occurs only when the designated user (without local administrator privileges in this case) logs on.

Q: Can the configuration of the offline client be scripted?

A: Yes. Regardless of which CRM client version is being installed, you can use a login script to run the Configuration Wizard silently assuming one of the following conditions exist:

* The user performing the installation is the user configuring for the offline client
* The user has local administrator privileges
* The offline client is being deployed via SMS or Group Policy

Q: Can the offline client be installed by user A, who has local administrator privileges, and then subsequently be configured for user B, who is a valid CRM user lacking local admin privileges?

A: This is not supported.

## Filtering Local Data

Q: Can an administrator define the “Local Data Groups” for all offline clients to limit the amount of data being transmitted across the network and then and then distribute this synchronization configuration to all existing offline clients?

A: Several organizations have already implemented application-specific routines to pre-create default offline filters, and several example routines are available via the Web.

Q: Is it possible to prevent users from defining Local Data Groups for specific entities? What about all entities generally?

A: Yes. An administrator can use a callout to block specific modifications to the Local Data Groups on a specific user’s computer.