Enterprise Data Integration for Microsoft Dynamics CRM

Daniel Cai

http://danielcai.blogspot.com
About me

• Daniel Cai
  – Developer @KingswaySoft
    • a software company offering integration software and solutions

– Main interests
  • Microsoft Dynamics CRM
  • Business Intelligence
  • .NET
  • Enterprise Architecture

– Microsoft Dynamics CRM MVP – 2011, 2012
Agenda

• Challenges

• Data Migration vs. Data Integration

• Data Migration Processes

• Data Migration / Integration Approaches
  – CRM Import Wizard
  – Custom Development
  – ETL
  – Service Bus / BizTalk

• Tips, tricks and traps
The Challenges of Data Migration / Integration

• Data migration/integration is complex
  – The diversity of data and systems
  – Data integrity
  – Time-consuming for large data set
  – The complexity and intricacy when working with Microsoft Dynamics CRM web service interfaces

• Data migration/integration is often overlooked
  – Data migration may not appear as important as the application itself
  – Last minute rush often causes poor planning, which could further delay your go-live date
  – Improper implementation of data migration can cause “surprises” down the road

• Data quality is a key to user adoption
  – Lack of quality data can make the system unusable
  – New business processes often depend on quality data to support them
# Data Integration vs. Data Migration

<table>
<thead>
<tr>
<th>Data Migration</th>
<th>Data Integration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Often a “One-Off” activity</td>
<td>On-going data synchronization and replication</td>
</tr>
<tr>
<td>• Often large volume of data in initial load</td>
<td>• Managing incremental changes</td>
</tr>
<tr>
<td>• Cost to fix any data issues thereafter is high</td>
<td>• Different requirements call for different design</td>
</tr>
<tr>
<td>• Often significant data cleansing effort is required</td>
<td>• Real-time</td>
</tr>
<tr>
<td></td>
<td>• Batch</td>
</tr>
<tr>
<td></td>
<td>• Messaging</td>
</tr>
<tr>
<td></td>
<td>• Usually need to be done within a time window</td>
</tr>
</tbody>
</table>
Data Migration Process

1. **Data Extraction**
   Extra data from different data sources

2. **Data Mapping**
   Transform and map data from source to the target system

3. **Data Validation**
   Validate data

4. **Data Load**
   Load data into the target system
Data Migration/Integration Approaches

• Leverage existing technologies and tools
  – CRM Import Data Wizard
  – ETL tools
    • SSIS
    • Informatica
    • Scribe
    • Connectors for Microsoft Dynamics
    • etc.
  – BizTalk / Service Bus
  – Other tools

• Write your own
  – Program against CRM Web Service Interfaces (SDK Programming)
CRM Import Data Wizard
CRM 2011 Import Data Wizard

- **What’s it?**
  - Free utilities offered by the platform

- **Pros**
  - Works for simple and small data import scenarios
  - Works within application, available for CRM users for self-served data imports
  - Undoable
    - works well for new insert, but not update though
  - Free

- **Cons**
  - No delete
    - don’t confuse with the above undo capability
  - No transformation
  - No scheduling, no automation
  - Difficult to manage incremental changes
  - Maximum number of records is constrained by file size
  - Limited capability of handling relationship
  - Exceptions to be expected when used for some special entities, fields
CRM 2011 Import Data Wizard

Map Fields

Select the Microsoft Dynamics CRM record type and map each source field to a target Microsoft Dynamics CRM field. We suggest that you map all the required fields before you click Next.

- All the record types with fields have been successfully mapped.

CRM Record Types
- Account

Source Fields
- Required Fields
  - Account Name
- Optional Fields
  - Account Number
  - Address 1: Address Type

Export Data to Excel

Select the type of worksheet to export.

- Static worksheet with records from this page
- Dynamic PivotTable
- Dynamic worksheet

Make this data available for re-importing by including required column headings.

We recommend that you save the exported file before you open it.
Custom Integration Development
Custom Integration Development

• How does it work?
  – Write custom code against CRM web service interfaces using SDK or service references

• Pros
  – Leverage your .NET programming (C# or VB.NET) skills
  – More granular control
  – Flexible integration points
    • Plugins
    • Workflows
    • Standalone applications (Console, Windows Form, and probably third-party apps)

• Cons
  – Could be an exhaustive effort, particularly when infrastructure coding is involved
    • Scheduling
    • Threading
    • Intricacies of working with CRM web service interfaces
  – Most likely much higher maintenance cost down the road
  – Often the case, the implementation ends up with a tightly-coupled architecture style, which leads to poor maintainability
Choices – Custom Development

• Service Interfaces
  – SOAP 2011
  – SOAP 2007
    • Not support by Office 365 CRM Online
    • Could be retired by Microsoft anytime soon

• Programming Styles
  – Early bound
  – Late bound

• Performance Improvement
  – Multi-threading
  – Bulk Data Load API
# Early-bound vs. Late-bound

<table>
<thead>
<tr>
<th>Pros</th>
<th>Early-bound</th>
<th>Late-bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Compile-time validation through strongly-typed entity classes and fields</td>
<td>• Slightly better performance comparing to early-bound</td>
<td></td>
</tr>
<tr>
<td>• Intellisense</td>
<td>• More flexibility</td>
<td></td>
</tr>
<tr>
<td>• CRM LINQ query APIs</td>
<td>• Smaller binary delivery</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cons</th>
<th>Early-bound</th>
<th>Late-bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Small performance overhead</td>
<td>• No compile-time validation or intellisense</td>
<td></td>
</tr>
<tr>
<td>• Dependency on command-line tool when CRM metadata has been updated</td>
<td>• Less productive CRM LINQ query APIs</td>
<td></td>
</tr>
<tr>
<td>• Larger binary delivery</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```csharp
var contact = new Contact
{
    FirstName = customer.FirstName,
    LastName = customer.LastName,
    Telephone1 = customer.Phone,
    EmailAddress1 = customer.Email,
};
xrmServiceContext.AddObject(contact);
xrmServiceContext.SaveChanges();
```

```csharp
var query = from c in svcContext.ContactSet
            join a in svcContext.AccountSet
            on c.ContactId equals a.PrimaryContactId.Id
            where c.LastName == "Smith"
            select new
            {
                ContactFirstName = c.FirstName,
                ContactLastName = c.LastName,
                AccountPhone = a.Address1_Telephone1
            };
```

```csharp
var contact = new Entity("contact");
contact["firstname"] = customer.FirstName;
contact["lastname"] = customer.LastName;
contact["telephonel"] = customer.Phone;
contact["emailaddress1"] = customer.Email;
organizationServiceProxy.Create(contact);
```

```csharp
var query = from c in orgContext.CreateQuery("contact")
            join a in orgContext.CreateQuery("account")
            on c["contactid"] equals a["primarycontactid"]
            where (String)c["lastname"] == "Smith"
            select new
            {
                ContactFirstName = c["firstname"],
                ContactLastName = c["lastname"],
                AccountPhone = a["address1_telephonel"]
            };
```
Data Integration through CRM Web Service

Demo
ETL
ETL Tools

• What’s ETL
  – Extract, Transform, Load

• Pros
  – Development Productivity
  – Visual development environment for data flows and control flow tasks
  – Scheduling engine
  – Performance
  – Scalability
  – Extensibility
  – Can be part of your overall BI and data warehousing strategies

• Cons
  – Learning curve of the ETL tool
  – Extra license cost of the ETL tool and/or the adapters
  – Probably not ideal solution for real-time requirements

• Options
  – SSIS (SQL Server Integration Services)
  – Informatica
  – Scribe
  – Connectors for Microsoft Dynamics
  – ...
SSIS Integration Toolkit for Microsoft Dynamics CRM

• Why SSIS?
  – SSIS is Microsoft’s answer to enterprise data integration
  – Scalability
  – Performance
  – Extensibility
  – Easy to work with
  – Accessible technical resources
  – Fits your overall business intelligence strategies

• What’s SSIS Integration Toolkit?
  – A cost effective and easy-to-use SSIS adapter
    • Support for Microsoft Dynamics CRM 2011, 4.0 and 3.0
    • Support for all deployments, including Office 365

• Free developer edition available at www.kingswaysoft.com
SSIS Integration Toolkit for Microsoft Dynamics CRM (Cont.)

- **CRM Connection Manager**
  - Support for all deployment types (On-premise, IFD, Office 365 and Online)
  - Support for SOAP 2011, 2007 and 2006 Service Endpoints

- **CRM Source Component**
  - Support for using CRM entity or FetchXML as data source
    - Any complex FetchXML query, including full metadata from linked entities
    - Support for parameterized FetchXML query

- **CRM Destination Component**
  - Five actions
    - Create
    - Update
    - Delete
    - Upsert
    - ExecuteWorkflow
  - Support of Bulk API
  - Support for CRM many-to-many relationship without requiring you to a single line of code
  - Unique Text Lookup feature, and many more ...

- **CRM OptionSet Mapping Component**
  - Translation of input values to valid CRM option set values
  - Ability to create new option when no match is found
SSIS Integration Toolkit for Microsoft Dynamics CRM

Demo
Connectors for Microsoft Dynamics

- A small footprint ETL engine

- Support the integration between Microsoft Dynamics CRM and most of Microsoft Dynamics ERP applications (AX, NAV, GP, SL)

- SDK is available to develop your own adapters
Connectors for Microsoft Dynamics (cont.)

Service Bus / BizTalk
What’s service bus?
- a software architecture model used for designing and implementing the interaction and communication between mutually interacting software applications in service-oriented architecture (SOA)

Pros
- Messaging based approach
  - Optimized to move single transactions between different systems or processes in near real time or real time
- Decoupled integration architectural model
  - Best suited for decoupled heterogeneous systems by using a middleware

Cons
- Probably not best fit with large volume data load
- Performance overhead due to serialization and deserialization
Service Bus Implementation Patterns

• Various Implementation Patterns (Azure Service Bus)
  – Queue
    • No active listener is required
    • destructive read vs. non-destructive read
    • two types of queues
      – message buffer queue
      – persistent queue (new)
  – One Way
    • requires an active listener
    • retries through asynchronous system job
  – Two Way
    • requires an active listener
    • a string value can be returned
  – REST
    • essentially a two-way listener in REST style
  – Topic (new in UR12 and December 2012 Service Update)
    • Similar to a queue, except that listener(s) can subscribe to receive messages from the topic
Windows Azure

Exchange messages between loosely coupled, composite applications.

Direct connection created if possible.
CRM + Azure Service Bus

- Integration points
  - Plugin
  - Workflow

CRM + Azure Service Bus

Demo

Walkthrough available at http://www.youtube.com/watch?v=qXPEFZXgasE
Some Limitations – CRM + Azure Integration

• There is no way to use custom messages
  – You publish the entire execution context, which could contain unnecessary information for other parties
  – Remove sensitive information from the context object if necessary

• Although you can host Windows Service Bus on-premise, you can’t use the service endpoint offered by the platform to talk to your service bus on-premise from CRM plug-in or Workflow

• If you need to push data (messages) in to CRM when they arrive in Azure Service Bus, you would need to write a listener service to do so, this is not currently provided by the platform

• There will be some technical challenges if you want to utilize third-party service bus solution for CRM Online
CRM On-Premise + Service Bus for Windows Server

• Scenarios
  – You don’t want a service bus that’s hosted on the cloud

• Benefits
  – It is a service bus on-premise
  – Better network connectivity if the service bus is for internal integration purposes only

• Caveats / tricks
  – CRM on-premise only, cannot be run in a sandbox runtime
  – We use plugin class static members to avoid constant initialization of message factory, which is expensive

Demo
Tips, Tricks and Treats
Common Technical Issues

• Microsoft Dynamics CRM as target
  – References (Lookup fields)
    • Design your program or data flow based upon the dependency
  – OptionSet (Picklist)
    • Integer values vs. text values
    • Translation of Option Set values between the source and the target
  – Special Entities
    • connection
    • principleobjectaccess
    • ...
  – Special Fields
    • statecode
    • statuscode
    • ownerid
    • activityparty fields
    • ...
  – Entity Type Code
    • It could change across environments for custom entities
  – Mind the performance impact with different cascading behavior
Common Technical Issues (cont.)

• Microsoft Dynamics CRM as source
  – Incremental changes
    • CreatedOn, ModifiedOn, VersionNumber
    • Use custom field

  – ActivityParty fields
    • What you get is an entity collection

  – Virtual Fields
    • Read from FormattedValues
How to maximize your data load performance?

- Minimize the number of fields you select when reading from or writing to CRM

- Utilize Bulk Data Load API introduced in UR13 and December 2012 Service Update
  - 500%-900% performance improvement for CRM online

- Use multi-threading to write to CRM in parallel
  - BDD in SSIS

- Write to multi-node in parallel if you have a cluster

- Mind your network latency between your integration client and CRM server

- Mind the impact that your plugins or workflows may have on data load performance
  - Disable them if you don’t need them in initial load
  - Use attribute filtering
  - Watch out the growth of CRM workflow log table (AsyncOperationBase), and delete completed workflow jobs as necessary

- SQL Performance Optimization
  - Disk IO, Memory
  - DB maintenance jobs to REBUILD or REORGANIZE indexes on a regular basis
  - Consider adding custom indexes if needed

- Many more tips from CRM whitepapers and the community
Rules of Thumb

• Know your data
  – both the source and target

• Know your tool
  – There are often more than one way to get a job done using one tool

• Define a proper error handling strategy and probably a retry mechanism should intermittent error happens

• Possibly use data migration/integration as the venue to clean up your data

• Define proper strategies that can help you avoid infinite loops when you need to do two-way integration

• Plan ahead, expect changes, particularly metadata changes
Q & A

Daniel Cai
KingswaySoft, [http://www.kingswaysoft.com](http://www.kingswaysoft.com)
Personal Blog: [http://danielcai.blogspot.com](http://danielcai.blogspot.com)
Email: daniel.cai@kingswaysoft.com