DataObjectException

Proving practical perspective on data migration – Dynamics 365

How to resolve quickly and successfully CRM errors

Whitepaper
Version 1.0, September 2019

Technical White Paper

Thierry Sinassamy with DataObjectException

V 1.0, September 2019

DataObjectException

Thierry Sinassamy (M.Sc.)
Software Architect & Developer

https://www.yuzuai.ca

DISCLAIMER

The white paper does not provide any confidential client data. It describes how we can quickly resolve CRM errors during a data migration from a SQL Server database to a SQL Server database, from Dynamics CRM 2011 on-premises to Dynamics 365 on-premises.

The purpose of the white paper is intended to show how we can deal with generic CRM errors and to show resolving generic CRM errors could a painless process.

Table of Contents

SUMMARY

The data migration process requires a methodological approach to address any errors generated by our validation tests once the data is transferred. The data migration process must be accompanied by a “Troubleshooting” process. The latter should provide appropriate, accurate and relevant solutions to help make the data migration a success.

Two types of errors are possible: a generic type error and a specific type error. Because of the content of errors generated by the CRM platform, it is natural to categorize them according to its two types.

To provide a better understanding of the content of these errors (generic and specific), tools are available (platform log, CRM trace log, server log, SQL query tracking, decompilation of .NET Framework CRM services). Thus, using these tools will generate a process (containing inclusive steps), creating inclusive levels of Troubleshooting. Once the errors are entered in a transparent manner, the solutions can be applied and thus put an end to the error generated by the platform.

Finally, the white paper shows how much CRM service contract extensions (“EntityServiceInternal<T>”) could play a key role in a problem-solving process.

KEYWORDS

Dynamics 365, Data Migration, Dynamics CRM, Troubleshooting, CRM Trace, CRM Exceptions, Event Viewer, Decompiler, Dynamics Link Library (DLL), MS SQL Profiler, Stored Procedures, OrganizationServiceFault, Database, FaultException, C#, CRM SDK365, Microsoft.Crm.ObjectModel, Asynchronous CRM services, Synchronous CRM services.

CONFIDENTIALITY

The white paper does not provide any confidential client data related to data migration. All the code in this white paper is only related to the CRM SDK365 code.

CHAPTER 1

INTRODUCTION

How to resolve a post-migration error in Dynamics 365?

CONTEXT

Describing data errors post-migration

INTRODUCTION: How to resolve a post-data migration error in Dynamics 365?

Troubleshooting is a systematic approach to problem solving that could used to find and resolve problems related to asynchronous and synchronous CRM services. The first step is to collect information related to the problem, such as undesirable behavior, a lack of expected functionality, or a crash in functionality related to the triggering of services (synchronous and asynchronous) that rely on CRM service contracts (Create Request, Update Request, SetState Request, etc.), etc.

Context: data migration phases

The migration from the CRM 2011 database to the D365 database takes place in an "unrestricted" environment from which we had access to all services and components of the Dynamics 365 platform. Our migration process has been executed in 4 steps:

- Step 0: Creating of technical requirements
- Step 1 : Data Preparation Process
  - Data related to the CRM security model (Team, Business Unit, System User and Queue)
  - Data related to CRM Entity Configuration Model (Solution, Solution Component, Localized Label)
- Step 2 : Data Transfer Process
  - Transfer by Groups and Zones
- Step 3 : Data finalization process
  - Finalize related to symmetrical indexes and keys
  - Finalization related to deploying CRM components using SDK

PROBLEM: Description of CRM Errors Post-Migration

Visualizing CRM Error or CRM Exception

When did a CRM error or exception occur during migration process?

STEP	PROCESS	DETAILS	CRM EXCEPTION
0	Creating technical requirements	N/A	**No exception**
1	Data Preparation Process	CRM Security Model	**FaultException CRM**
		CRM Configuration Model	**No exception**
2	Data Transfer Process	Groups/Zones/Entities	**FaultException CRM**
3	Data Finalization Process	Indexes/Symetric Keys	**No Exception**
		CRM Components Deployment	**FaultException CRM**

Questions asked to troubleshoot CRM errors

- How? How do I understand CRM error or exception?
- How? How do I understand CRM error or exception and find the right solution?

What type of CRM error or exception occurs when migrating data?

Types of CRM errors or exceptions

TYPE	ERROR/EXCEPTION	DLL TARGETED	ERROR EXTENSION
CRM	FaultException/OrganizationServiceFault	Microsoft.Xrm.Sdk	No extension
CRM	FaultException/OrganizationServiceFault	Microsoft.Xrm.Sdk	System.Web.HttpUnhandledException

Contents of CRM errors or exceptions

TYPE	ERROR/EXCEPTION	DLL	CONTENT
CRM	FaultException/OrganizationServiceFault	Microsoft.Xrm.Sdk	An unexpected error occurred
			Can't add members to the default business unit team
			RoleService: VerifyCallerPrivileges failed. User: ac8726fc-16e8-e311-0247-000056b80603, PrivilegeName: prv…, PrivilegeId: 2caa63a2-07d6-42dd-9262-300c6b61ee00, Depth: Global, BusinessUnitId: 2caa63a2-07d6-42dd-9262-300c6b61ee00

Depth of CRM errors or exceptions

TYPE	ERROR/EXCEPTION	DEPTH	CONTENT	ENTITIES MODEL
CRM	FaultException/OrganizationService Fault	1	An unexpected error occurred	ALL ENTITIES
		2	Can't add members to the default business unit team	SECURITY MODEL
		2	RoleService: VerifyCallerPrivileges failed.	SECURITY MODEL

Some examples

Error regarding entities « Business Unit » and « Team »

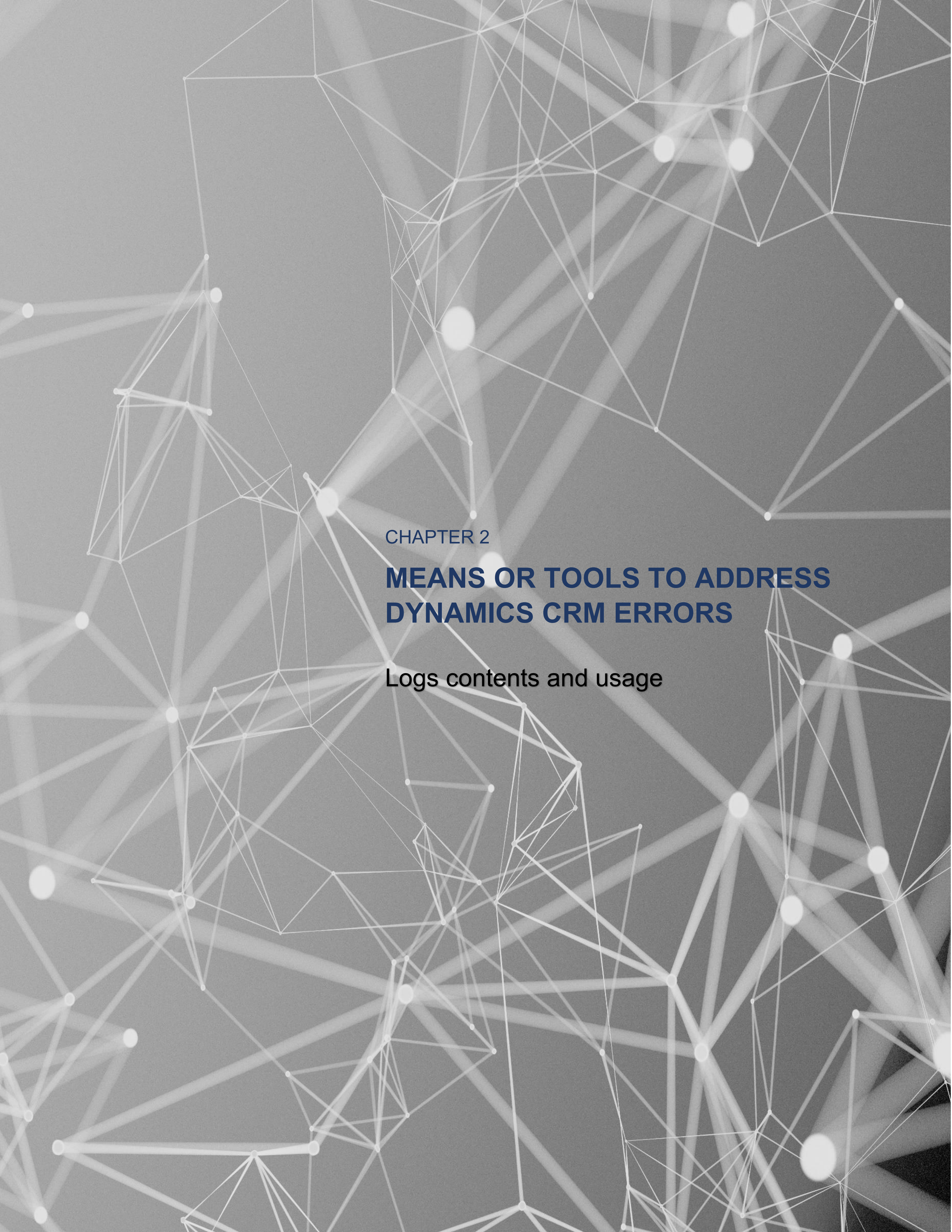
```
Unhandled Exception: System.ServiceModel.FaultException[[Microsoft.Xrm.Sdk.OrganizationServiceFault, Microsoft.Xrm.Sdk,
Version=8.0.0.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35]]:
Can't add members to the default business unit team. Detail:
<OrganizationServiceFault xmlns:i="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://schemas.microsoft.com/xrm/2011/Contracts">
  <ActivityId>8efcd1b6-5bf4-49f0-87cf-e42452c9fdc2</ActivityId>
  <ErrorCode>-2147187957</ErrorCode>
  <ErrorDetails xmlns: d2p1="http://schemas.datacontract.org/2004/07/System.Collections.Generic" />
  <Message>Can't add members to the default business unit team</Message>
  <Timestamp>2018-09-24T18:31:28.3317847Z</Timestamp>
  <ExceptionRetriable>false</ExceptionRetriable>
  <ExceptionSource i:nil="true" />
  <InnerFault i:nil="true" />
  <OriginalException i:nil="true" />
  <TraceText i:nil="true" />
</OrganizationServiceFault>[1]
```

Error regarding entities « Role » et « Privilege »

```
Unhandled Exception: System.ServiceModel.FaultException[[Microsoft.Xrm.Sdk.OrganizationServiceFault, Microsoft.Xrm.Sdk,
Version=8.0.0.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35]]: System.Web.HttpUnhandledException: Microsoft
Dynamics CRM has experienced an error. Reference number for administrators or support: #9DA12221Detail:
<OrganizationServiceFault xmlns:i="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://schemas.microsoft.com/xrm/2011/Contracts">
  <ActivityId>bd4af73c-a11c-474d-8cff-a076416208aa</ActivityId>
  <ErrorCode>-2147220970</ErrorCode>
  <ErrorDetails xmlns:d2p1="http://schemas.datacontract.org/2004/07/System.Collections.Generic" />
  <Message>System.Web.HttpUnhandledException: Microsoft Dynamics CRM has experienced an error. Reference number for administrators or support: #9DA12221</Message>
  <Timestamp>2019-05-11T23:50:33.5128137Z</Timestamp>
  <ExceptionRetriable>false</ExceptionRetriable>
  <ExceptionSource i:nil="true" />
  <InnerFault>
    <ActivityId>bd4af73c-a11c-474d-8cff-a076416208aa</ActivityId>
    <ErrorCode>-2147220960</ErrorCode>
    <ErrorDetails xmlns:d3p1="http://schemas.datacontract.org/2004/07/System.Collections.Generic" />
    <Message>RoleService::VerifyCallerPrivileges failed. User: ac8026fc-16e8-e311-9247-005056b80603, PrivilegeName: prv..., PrivilegeId: 2caa63a2-00d6-42dd-9262-058c6b61ee01, Depth: Global, BusinessUnitId: db002ef6-16e8-e311-9247-005056b80603</Message>
    <Timestamp>2019-05-11T23:50:33.5138137Z</Timestamp>
              …
  </InnerFault>
  <OriginalException i:nil="true" />
  <TraceText i:nil="true" />
</OrganizationServiceFault>[2]
```

Error as generic error

```
Unhandled Exception: System.ServiceModel.FaultException[[Microsoft.Xrm.Sdk.OrganizationServiceFault, Microsoft.Xrm.Sdk, Version=8.0.0.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35]]: An unexpected error occurred. Detail:
<OrganizationServiceFault xmlns:i="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://schemas.microsoft.com/xrm/2011/Contracts">
  <ActivityId>425234c8-811f-4484-a2ff-2f7a3bf98487</ActivityId>
  <ErrorCode>-2147220970</ErrorCode>
  <ErrorDetails xmlns:d2p1="http://schemas.datacontract.org/2004/07/System.Collections.Generic" />
  <Message>An unexpected error occurred</Message>
  <Timestamp>2018-09-24T18:15:17.2530019Z</Timestamp>
              ...
  <TraceText i:nil="true" />
</OrganizationServiceFault>[3]
```



CHAPTER 2

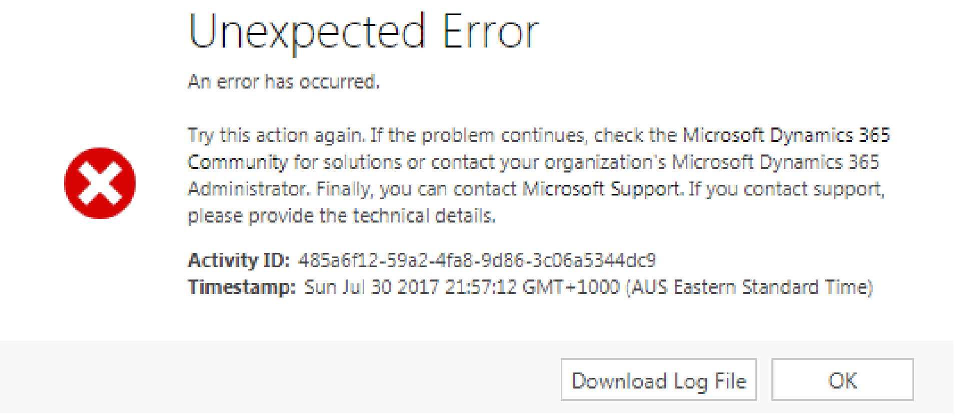
MEANS OR TOOLS TO ADDRESS DYNAMICS CRM ERRORS

Logs contents and usage

Means or tools to address CRM errors or exceptions: Logs contents

Log generated by CRM platform

When the platform executes the user's request for an action through the interface using the service of the CRM (OrganizationService), that action gets an error or a CRM exception that appears in a "Pop-up" window. From this window, the user can download the error in question (see points on "Concrete examples").



Unexpected Error

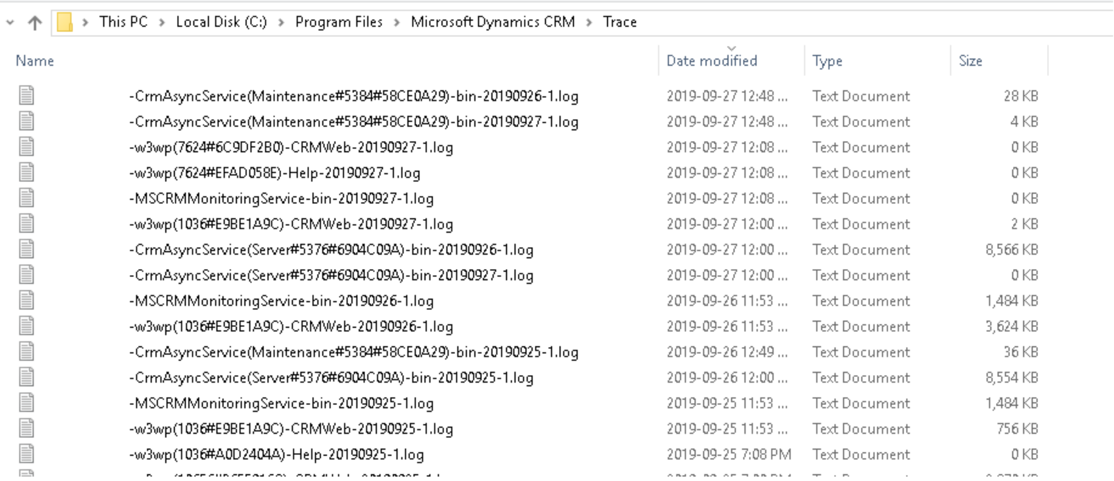
An error has occurred.

Try this action again. If the problem continues, check the Microsoft Dynamics 365 Community for solutions or contact your organization's Microsoft Dynamics 365 Administrator. Finally, you can contact Microsoft Support. If you contact support, please provide the technical details.

Activity ID: 485a6f12-59a2-4fa8-9d86-3c06a5344dc9
Timestamp: Sun Jul 30 2017 21:57:12 GMT+1000 (AUS Eastern Standard Time)

Log generated by the server through "Tracing CRM" feature

In general, the "Tracing CRM"[4] function is not enabled. However, during a data migration, its activation becomes essential to understand and understand a CRM error or exception generated through the CRM interface. Once the error has occurred, it will have to be repeated once the "Tracing CRM" has been activated. In doing so, the log will be placed in the directory defined in the server registry keys.



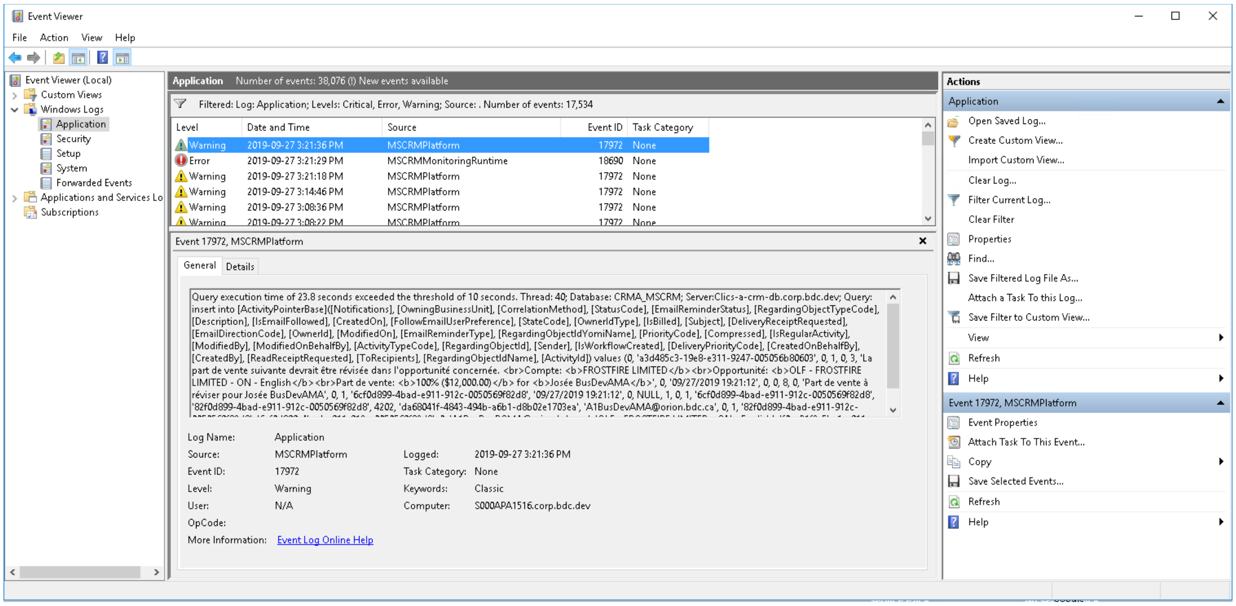
This PC > Local Disk (C:) > Program Files > Microsoft Dynamics CRM > Trace

Name	Date modified	Type	Size
-CrmAsyncService(Maintenance#5384#58CE0A29)-bin-20190926-1.log	2019-09-27 12:48 ...	Text Document	28 KB
-CrmAsyncService(Maintenance#5384#58CE0A29)-bin-20190927-1.log	2019-09-27 12:48 ...	Text Document	4 KB
-w3wp(7624#6C9DF2B0)-CRMWeb-20190927-1.log	2019-09-27 12:08 ...	Text Document	0 KB
-w3wp(7624#EFAD058E)-Help-20190927-1.log	2019-09-27 12:08 ...	Text Document	0 KB
-MSCRMMonitoringService-bin-20190927-1.log	2019-09-27 12:08 ...	Text Document	0 KB
-w3wp(1036#E9BE1A9C)-CRMWeb-20190927-1.log	2019-09-27 12:00 ...	Text Document	2 KB
-CrmAsyncService(Server#5376#6904C09A)-bin-20190926-1.log	2019-09-27 12:00 ...	Text Document	8,566 KB
-CrmAsyncService(Server#5376#6904C09A)-bin-20190927-1.log	2019-09-27 12:00 ...	Text Document	0 KB
-MSCRMMonitoringService-bin-20190926-1.log	2019-09-26 11:53 ...	Text Document	1,484 KB
-w3wp(1036#E9BE1A9C)-CRMWeb-20190926-1.log	2019-09-26 11:53 ...	Text Document	3,624 KB
-CrmAsyncService(Maintenance#5384#58CE0A29)-bin-20190925-1.log	2019-09-26 12:49 ...	Text Document	36 KB
-CrmAsyncService(Server#5376#6904C09A)-bin-20190925-1.log	2019-09-26 12:00 ...	Text Document	8,554 KB
-MSCRMMonitoringService-bin-20190925-1.log	2019-09-25 11:53 ...	Text Document	1,484 KB
-w3wp(1036#E9BE1A9C)-CRMWeb-20190925-1.log	2019-09-25 11:53 ...	Text Document	756 KB
-w3wp(1036#A0D2404A)-Help-20190925-1.log	2019-09-25 7:08 PM	Text Document	0 KB

Log generated in the server via "Event Viewer" feature

The function of Event Viewer in a server (installation of the platform in full server mode) could play a key role in the "Troubleshooting" in general. Therefore, it was clear that all applications and behaviors of the latter would be logged in the server and categorized by the system (error, warning, information).

In the D365 platform, all "warning" logs must be considered in connection with an exception or an error generated during a query via the CRM interface. In fact, this type of error ("warning") in the server can hide a CRM error.



Application Number of events: 38,076 (!) New events available

Filtered: Log: Application; Levels: Critical, Error, Warning; Source: . Number of events: 17,534

Level	Date and Time	Source	Event ID	Task Category
Warning	2019-09-27 3:21:36 PM	MSCRMPlatform	17972	None
Error	2019-09-27 3:21:29 PM	MSCRMMonitoringRuntime	18690	None
Warning	2019-09-27 3:21:18 PM	MSCRMPlatform	17972	None
Warning	2019-09-27 3:14:46 PM	MSCRMPlatform	17972	None
Warning	2019-09-27 3:08:36 PM	MSCRMPlatform	17972	None
Warning	2019-09-27 3:08:22 PM	MSCRMPlatform	17972	None

Event 17972, MSCRMPlatform

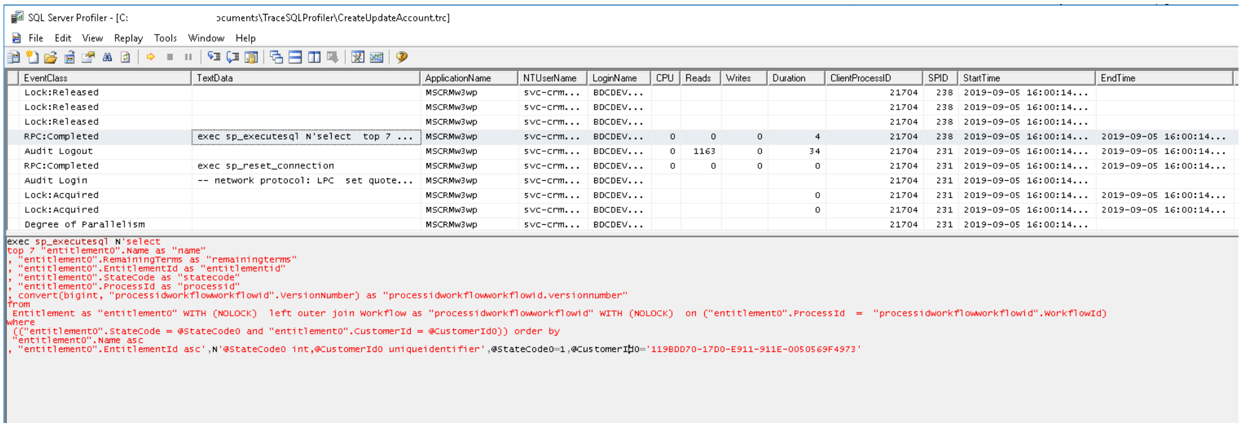
General | Details

```
Query execution time of 23.8 seconds exceeded the threshold of 10 seconds. Thread: 40; Database: CRMA_MSCRM; Server:Clics-a-crm-db.corp.bdc.dev; Query:
insert into [ActivityPointerBase]([Notifications], [OwningBusinessUnit], [CorrelationMethod], [StatusCode], [EmailReminderStatus], [RegardingObjectTypeCode],
[Description], [IsEmailFollowed], [CreatedOn], [FollowEmailUserPreference], [StateCode], [OwnerIdType], [IsBilled], [Subject], [DeliveryReceiptRequested],
[EmailDirectionCode], [OwnerId], [ModifiedOn], [EmailReminderType], [RegardingObjectIdYomiName], [PriorityCode], [Compressed], [IsRegularActivity],
[ModifiedBy], [ModifiedOnBehalfBy], [ActivityTypeCode], [RegardingObjectId], [Sender], [IsWorkflowCreated], [DeliveryPriorityCode], [CreatedOnBehalfBy],
[CreatedBy], [ReadReceiptRequested], [ToRecipients], [RegardingObjectIdName], [ActivityId]) values (0, 'a3d485c3-19e8-e311-9247-005056b80603', 0, 1, 0, 3, 'La
part de vente suivante devrait être révisée dans l'opportunité concernée. <br>Compte: <b>FROSTFIRE LIMITED</b><br>Opportunité: <b>OLF - FROSTFIRE
LIMITED - ON - English</b><br>Part de vente: <b>100% ($12,000.00)</b> for <b>Josée BusDevAMA</b>', 0, '09/27/2019 19:21:12', 0, 0, 8, 0, 'Part de vente à
réviser pour Josée BusDevAMA', 0, 1, '6cf0d899-4bad-e911-912c-0050569f82d8', '09/27/2019 19:21:12', 0, NULL, 1, 0, 1, '6cf0d899-4bad-e911-912c-0050569f82d8',
'82f0d899-4bad-e911-912c-0050569f82d8', 4202, 'da68041f-4843-494b-a6b1-d8b02e1703ea', 'A1BusDevAMA@orion.bdc.ca', 0, 1, '82f0d899-4bad-e911-912c-
```

Log Name: Application
Source: MSCRMPlatform   Logged: 2019-09-27 3:21:36 PM
Event ID: 17972   Task Category: None
Level: Warning   Keywords: Classic
User: N/A   Computer: S000APA1516.corp.bdc.dev
OpCode:
More Information: Event Log Online Help

"Tracking" tool of SQL queries generated through CRM platform

Because data migration is a database migration to a database in an "on-premises" context (on-site), it became undeniable that the SQL Server Profiler tool could play a crucial role in the "Troubleshooting" of our CRM errors because we knew that queries executed via the platform (CRM web services contracts) turned into SQL queries and used procedures stored or native views D365.


SQL Server Profiler - [C: ocuments\TraceSQLProfiler\CreateUpdateAccount.trc]
File
Edit
View
Replay
Tools
Window
Help

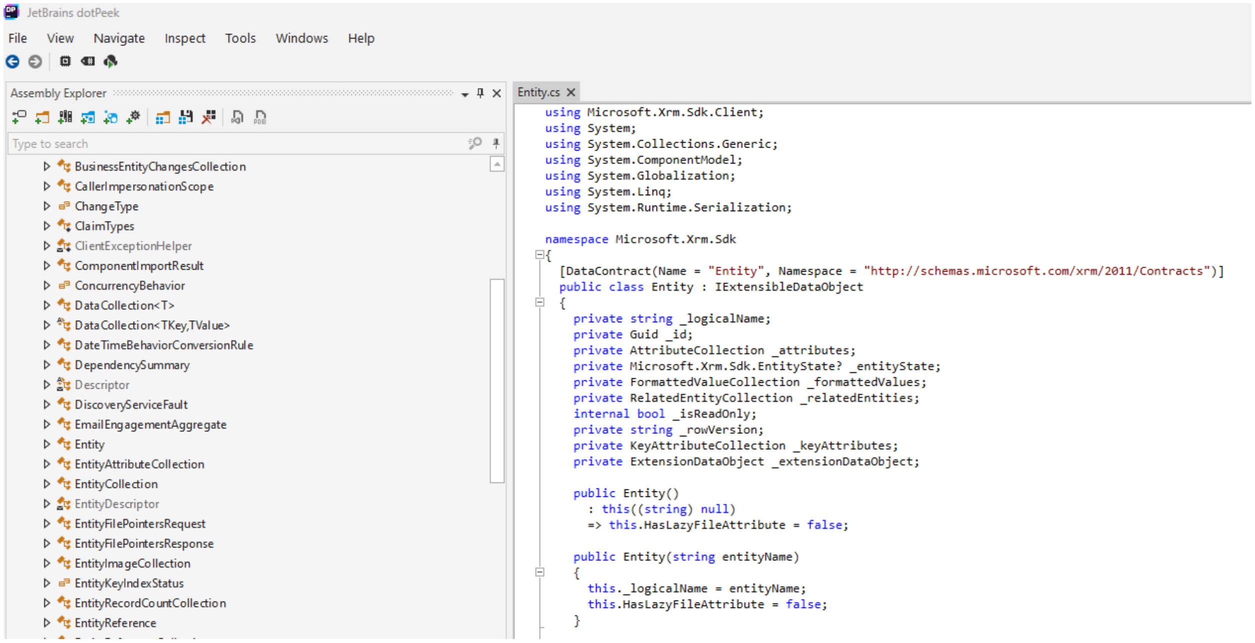
EventClass	TextData	ApplicationName	NTUserName	LoginName	CPU	Reads	Writes	Duration	ClientProcessID	SPID	StartTime	EndTime
Lock:Released		MSCRMw3wp	svc-crm...	BDCDEV...					21704	238	2019-09-05 16:00:14...	
Lock:Released		MSCRMw3wp	svc-crm...	BDCDEV...					21704	238	2019-09-05 16:00:14...	
Lock:Released		MSCRMw3wp	svc-crm...	BDCDEV...					21704	238	2019-09-05 16:00:14...	
RPC:Completed	exec sp_executesql N'select top 7 ...	MSCRMw3wp	svc-crm...	BDCDEV...	0	0	0	4	21704	238	2019-09-05 16:00:14...	2019-09-05 16:00:14...
Audit Logout		MSCRMw3wp	svc-crm...	BDCDEV...	0	1163	0	34	21704	231	2019-09-05 16:00:14...	2019-09-05 16:00:14...
RPC:Completed	exec sp_reset_connection	MSCRMw3wp	svc-crm...	BDCDEV...	0	0	0	0	21704	231	2019-09-05 16:00:14...	2019-09-05 16:00:14...
Audit Login	-- network protocol: LPC set quote...	MSCRMw3wp	svc-crm...	BDCDEV...					21704	231	2019-09-05 16:00:14...	
Lock:Acquired		MSCRMw3wp	svc-crm...	BDCDEV...				0	21704	231	2019-09-05 16:00:14...	2019-09-05 16:00:14...
Lock:Acquired		MSCRMw3wp	svc-crm...	BDCDEV...				0	21704	231	2019-09-05 16:00:14...	2019-09-05 16:00:14...
Degree of Parallelism		MSCRMw3wp	svc-crm...	BDCDEV...					21704	231	2019-09-05 16:00:14...	

```
exec sp_executesql N'select
top 7 "entitlement0".Name as "name"
, "entitlement0".RemainingTerms as "remainingterms"
, "entitlement0".EntitlementId as "entitlementid"
, "entitlement0".StateCode as "statecode"
, "entitlement0".ProcessId as "processid"
, convert(bigint, "processidworkflowworkflowid".VersionNumber) as "processidworkflowworkflowid.versionnumber"
from
 Entitlement as "entitlement0" WITH (NOLOCK)  left outer join Workflow as "processidworkflowworkflowid" WITH (NOLOCK)  on ("entitlement0".ProcessId  =  "processidworkflowworkflowid".WorkflowId)
where
 (("entitlement0".StateCode = @StateCode0 and "entitlement0".CustomerId = @CustomerId0)) order by
 "entitlement0".Name asc
, "entitlement0".EntitlementId asc',N'@StateCode0 int,@CustomerId0 uniqueidentifier',@StateCode0=1,@CustomerId0='119BDD70-17D0-E911-911E-0050569F4973'
```

Tool to decompile synchronous and asynchronous CRM services (.NET libraries)

While there are recognized Dynamics CRM errors caused by asynchronous and synchronous service behaviors during a data migration, there have been no errors in this migration directly related to asynchronous CRM services. Errors appear at first glance as errors from the OrganizationService (CRM) synchronous services.

The free decompiling tool referred to is provided by Jet Brains dotPeek (.NET Decompiler). This allows to decompile the .NET libraries used by the D365 platform on both the client (synchronous services) and server (asynchronous service) sides.

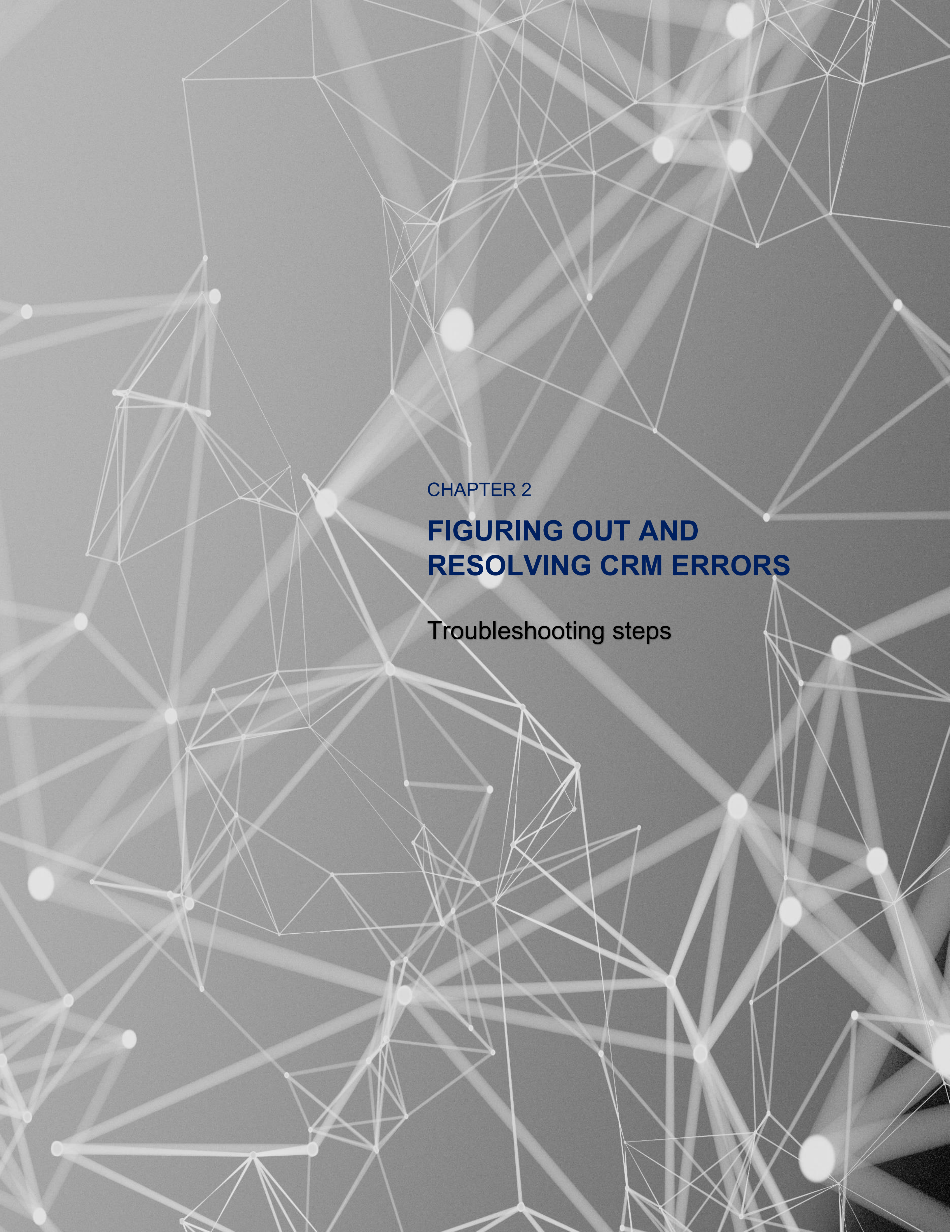


```
using Microsoft.Xrm.Sdk.Client;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Globalization;
using System.Linq;
using System.Runtime.Serialization;

namespace Microsoft.Xrm.Sdk
{
  [DataContract(Name = "Entity", Namespace = "http://schemas.microsoft.com/xrm/2011/Contracts")]
  public class Entity : IExtensibleDataObject
  {
    private string _logicalName;
    private Guid _id;
    private AttributeCollection _attributes;
    private Microsoft.Xrm.Sdk.EntityState? _entityState;
    private FormattedValueCollection _formattedValues;
    private RelatedEntityCollection _relatedEntities;
    internal bool _isReadOnly;
    private string _rowVersion;
    private KeyAttributeCollection _keyAttributes;
    private ExtensionDataObject _extensionDataObject;

    public Entity()
      : this((string) null)
      => this.HasLazyFileAttribute = false;

    public Entity(string entityName)
    {
      this._logicalName = entityName;
      this.HasLazyFileAttribute = false;
    }
```



CHAPTER 2

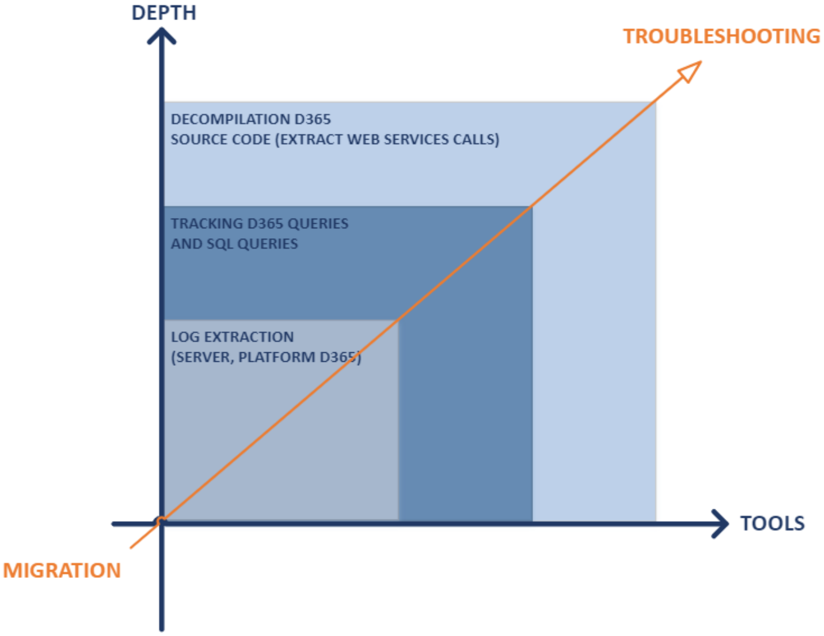
FIGURING OUT AND RESOLVING CRM ERRORS

Troubleshooting steps

Figuring out and resolving CRM errors

3 inclusive levels of "Troubleshooting"

CRM error handling and resolution process could be characterized by 3 levels of "Troubleshooting": extracting logs, tracking SQL Server queries, and finally, decompiling the D365 platform's native code source. These 3 levels of analysis should not be separated from each other but should instead be included in a comprehensive approach and considered complementary to each other.



« Troubleshooting » steps

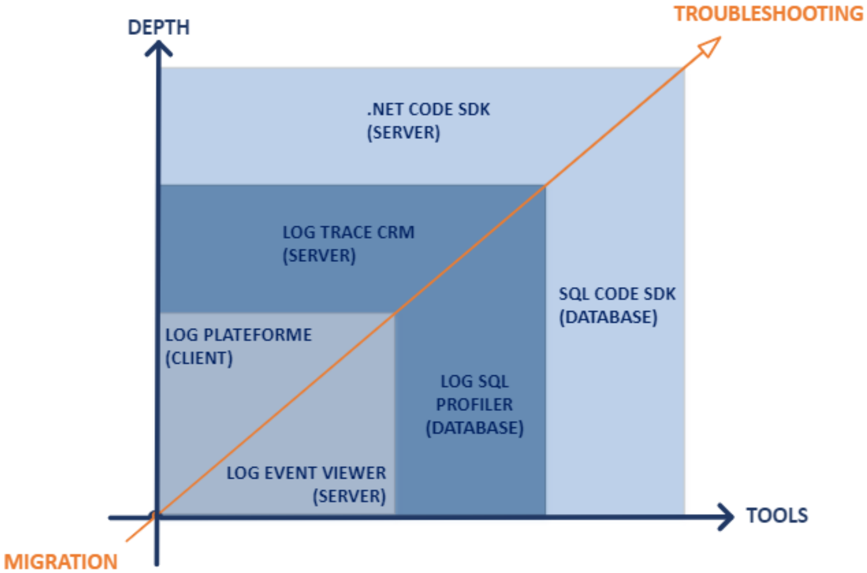
- Preliminary Step**:**
  - ❖ Exécution du code de migration (SSIS package).
  - ❖ Importer l'organisation CRM via le « Deployment Manager » (SDK Deployment)
  - ❖ Exécuter les tests fonctionnels de type « CRUD » (Create, Read, Update et Delete) et de type « Activation ».

- **Step 1**: Retrieving the CRM log generated after the CRM error is displayed via the CRM interface.
- **Step 2**: Extraction of CRM log generated in the server by D365 platform.
  - ❖ Extracting ERROR errors if needed
  - ❖ Extracting WARNING errors if needed
- **Step 3**: Activate the CRM function "Trace" and restart the test.
- **Step 4**: Extraction of CRM log from CRM Trace.
- **Step 5**: Comparison and validation of the contents of the CRM "Trace" with previous extractions.
- **Step 6**: Activate SQL Profiler and restart the test.
- **Step 7**: Extracting SQL queries for the CRM database (CRM_MSCRM).
- **Step 8**: Comparing and validating the contents of the "SQL Profiler" with previous extractions.
- **Step 9**: Retrieving .NET (C#) code from the D365 Software Development Kit (SDK) via the Jet Brains decompiling tool.

- **Step 10**: Extracting CRM native stored procedures from the CRM Organization (CRM_MSCRM) database.

As mentioned above, the steps are not exclusive, but complementary to one another. In addition, numbering highlights the order in which they are executed, even if it is not always necessary to extract the CRM native code from the database (step 10).

The further you go through the steps, the more you use "Troubleshooting" tools, and the more you advance in the resolution in depth.



Why such a leveling of "Troubleshooting" tools? Because simply, one level of resolution could bring to another level of resolution. So, here are the possible crossings:

We know that level 1 of "troubleshooting" is not enough to make sense of the error and even more so to link to SQL queries executed or queries executed by .NET code (via SDK) during the transaction. That said, level 1 will at least be our starting point for the "troubleshooting" process. 2 scenarios are possible:

<u>Scenario 1 - from Level 1 to Level 3: We can go from level 1 to level 3 directly.</u>

- An unexpected error has occurred: It may be better to move immediately to level 3 by searching the services mentioned in the .NET library (Microsoft.Crm.ObjectModel) because the transaction executed through the CRM interface is related to the entity involved in the transaction at first sight. In addition, it would be better to "tracker" all SQL queries (views, stored procedures..." via the "SQL Profiler" tool. However, the error does not always allow us to focus our search on the CRM native stored procedures.
- Case of a specific error "Can't add members to the default business unit team": It is also preferable to take the error message (descriptive content) and search the .NET library (Microsoft.Crm.ObjectModel) and the native SQL queries of the CRM. We can access source code: search by keywords does not give anything and therefore, the focus should be on methods in services related to the following entities: Team ("TeamServiceInternal", Business Unit ("BusinessUnitServiceInternal") and System User ("SystemUserServiceInternal").

- Case of a specific error "RoleService::VerifyCallerPrivileges failed": It is also preferable to search the SDK's .NET library (Microsoft.Crm.ObjectModel) by keywords. Also, since the contents of the error refer to concepts such as "Role" and "Privilege", the latter can use the search for native SQL queries in the MSCRM database.

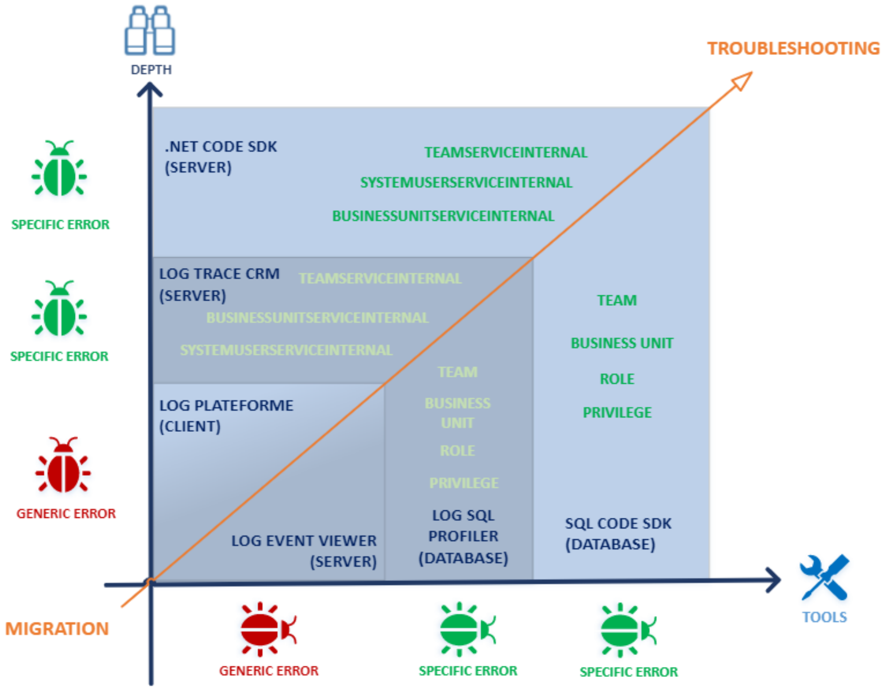
Scenario 2 - from level 1 to level 2, then level 2 to level 3: Level 2 is an intermediate passage

- Case of a generic error "An unexpected error has occurred" and Case of a specific error "Can't add members to the default business unit team" and "RoleService::VerifyCallerPrivileges failed": It may be better to move to level 2 in other words:
  - Activate the CRM trace: This allows to extract the information that relates to the .NET bookstore taken into account in the transaction, but also, the log generated by the "Trace CRM" refers to the methods in which the queries of the CRM service contracts are mentioned (e.g. UpdateRequest).
  - And/or Activate the SQL profiler: this allows you to extract information that relates to SQL queries and stored procedures considered in the transaction.

Regardless of the type of error, the CRM trace and the SQL profiler play a decisive role:

The CRM trace provides access to the method of execution from the so-called "StackTrace" and the type of exception (e.g. CrmException); of course, the error code (e.g. -2147187957) can also be found there. As for the SQL profiler, it gives access to the SQL request executed but possibly to the stored procedure performed by the CRM service contracts (updaterequest, createrequest, etc.). In addition, the SQL profiler goes further by providing access to the SQL views used in the transaction.

As a result, the further you go in the level, the more generic the error can become. It is from level 2 that the generic error becomes a specific error.



Error resolutions and CRM exceptions

Here, the resolution process answers the question: how can the errors or CRM exceptions mentioned above be resolved: "Can't add members to the default business unit team"; "RoleService::VerifyCallerPrivileges failed" and finally, "An unexpected error has occurred"?

How to fix the generic error: an unexpected error has occurred?

- **Step 1**: The server log ("Event Viewer") looks at the request executed considering the exact timing of execution: the warning and error types are considered. In the CRM, the "warning" type is to be considered an error-type error.
- **Step 2**: The error code is extracted from the XML from the platform-generated log and searched in the CRM server's decompiled bookstores.
- **Step 3**: The transaction is restarted and with the sql profiler, the request is captured (select request, "update" type, "insert" type, "delete" type, stored procedure call, call of views). Thus, extract the query if possible from the database of the organization CRM (MSCRM).
- **Step 4**: CRM Trace can give content but not always guaranteed. Based on the error code, we could determine and confirm the type of error: here is the C# code below.

```
 Public enum ErrorType
{
 SystemFailure,
 Timeout
 CustomerError
 }
```

The CRM's SDK contains an error list that the platform categorizes with an enum-type object (see next list). The generic error can be found in the "ErrorCode" class in the .NET bookstore "Microsoft.Crm.Constants":

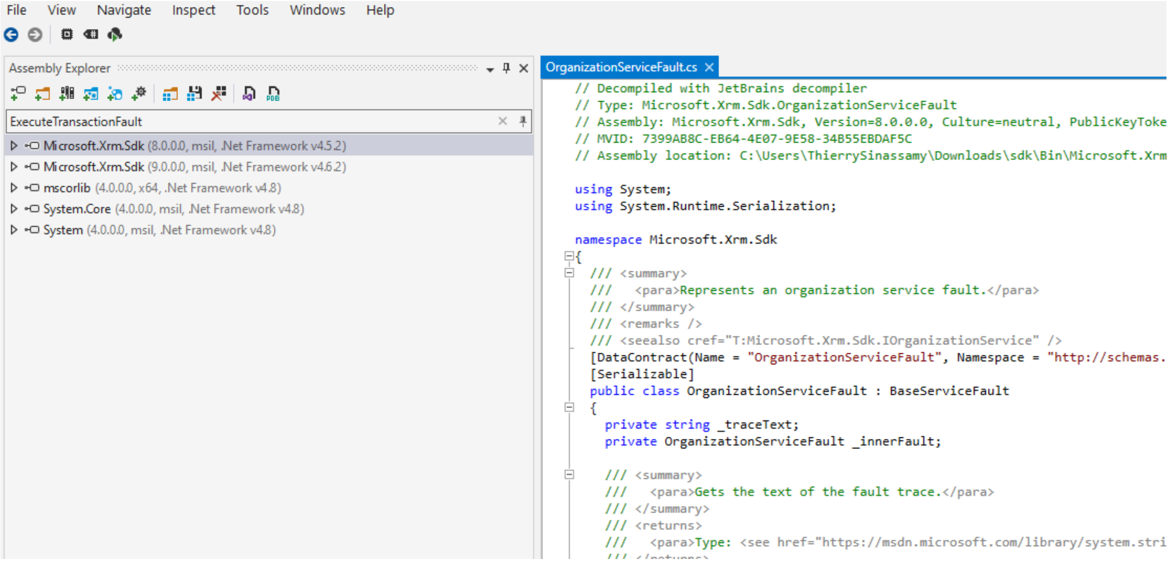
```
ErrorCodes.ErrorMessages.Add((object) -2147220970, (object) -"An unexpected error occurred.");
ErrorCodes.ErrorTypes.Add((object)-2147220970,(object) ErrorCodes.ErrorType.SystemFailure);
```

In the case of migration, the CRM error will be conducive to being system-type more often even if we migrate data related to the CRM's basic security model.

- Step 5: We try to cross the request launched with the decompiled code and always considering the object model that we test (entity and therefore, for example, "RoleServiceInternal" if our transaction concerns the CRM security model). It is enough to reach the method executed by the SDK of the CRM and thus analyze its content.

How to resolve generic error: an unexpected error has occurred? - Additional point

By default, all CRM transactions from the client (interface) use the .NET library "Microsoft.Crm.Core.WebServices", more precisely, uses a base class of type "sealed" and therefore cannot be instantiated.



ExecuteTransactionFault

```
Microsoft.Xrm.Sdk (8.0.0.0, msil, .Net Framework v4.5.2)
Microsoft.Xrm.Sdk (9.0.0.0, msil, .Net Framework v4.6.2)
mscorlib (4.0.0.0, x64, .Net Framework v4.8)
System.Core (4.0.0.0, msil, .Net Framework v4.8)
System (4.0.0.0, msil, .Net Framework v4.8)
```

```
// Decompiled with JetBrains decompiler
// Type: Microsoft.Xrm.Sdk.OrganizationServiceFault
// Assembly: Microsoft.Xrm.Sdk, Version=8.0.0.0, Culture=neutral, PublicKeyToke
// MVID: 7399AB8C-EB64-4E07-9E58-34B55EBDAF5C
// Assembly location: C:\Users\ThierrySinassamy\Downloads\sdk\Bin\Microsoft.Xrm

using System;
using System.Runtime.Serialization;

namespace Microsoft.Xrm.Sdk
{
  /// <summary>
  ///   <para>Represents an organization service fault.</para>
  /// </summary>
  /// <remarks />
  /// <seealso cref="T:Microsoft.Xrm.Sdk.IOrganizationService" />
  [DataContract(Name = "OrganizationServiceFault", Namespace = "http://schemas.
  [Serializable]
  public class OrganizationServiceFault : BaseServiceFault
  {
    private string _traceText;
    private OrganizationServiceFault _innerFault;

    /// <summary>
    ///   <para>Gets the text of the fault trace.</para>
    /// </summary>
    /// <returns>
    ///   <para>Type: <see href="https://msdn.microsoft.com/library/system.stri
```

In the log generated by the platform, the contents of the error refer to a method of type "ExecuteTransasctionFault", an object that inherits from a base class "OrganizationServiceFault". The latter is referenced in generic type errors:

```
[DataContract(Name = "ExecuteTransactionFault", Namespace =
"http://schemas.microsoft.com/xrm/7.1/Contracts")]
[Serializable]
public sealed class ExecuteTransactionFault : OrganizationServiceFault
{
  private int _faultedRequestIndex;

  [DataMember]
  public int FaultedRequestIndex
  {
   get
   {
     return this._faultedRequestIndex;
   }
   set
   {
     this._faultedRequestIndex = value;
   }
  }
}
```

As for the BaseServiceFault error - we know that it is a CRM service contract that is called (createrequest, updaterequest...) and therefore a request is made to the server via the CRM SDK. This is why CRM tracing and SQL profiling must be enabled in order to accurately retrieve the execution method during the transaction.

The method that generates the generic error is "ConvertToExecuteTransactionFault", which sets the parameter to "BaseServiceFault" creating the error in XML format.

```
public static ExecuteTransactionFault ConvertToExecuteTransactionFault(BaseServiceFault serviceFault)
{
    ExecuteTransactionFault transactionFault = new ExecuteTransactionFault();
    transactionFault.ErrorCode = serviceFault.ErrorCode;
    transactionFault.ErrorDetails = serviceFault.ErrorDetails;
    transactionFault.ExtensionData = serviceFault.ExtensionData;
    transactionFault.InnerServiceFault = serviceFault.InnerServiceFault;
    transactionFault.Message = serviceFault.Message;
    transactionFault.Timestamp = serviceFault.Timestamp;
    transactionFault.ActivityId = serviceFault.ActivityId;
    OrganizationServiceFault organizationServiceFault = serviceFault as OrganizationServiceFault;
    if (organizationServiceFault != null)
    {
      transactionFault.InnerFault = organizationServiceFault.InnerFault;
      transactionFault.TraceText = organizationServiceFault.TraceText;
    }
    return transactionFault;
}
```

How to resolve the specific error: RoleService::VerifyCallerPrivileges failed?

The error is of type CrmException not defined in the "ErrorCode" class of the .NET library "Microsoft.Crm. Constants", but defined in the .NET library "Microsoft.Crm.ObjectModel":

- **Step 1**: Extract the method associated with the error on transaction from the "CRM Trace": "VerifyCallerPrivileges". Retrieving should allow us to determine the service generates the error, in this case
  "**RoleServiceInternal**".

- **Step 2**: Extract the SQL execution query into the SQL Profiler (views, stored procedures if applicable):

  The stored procedure "p_RetrievePrivileges" is the stored procedure that searches for the privileges associated with the user that is logged in.

  Filtered views "**FilteredPrivilege**" and "**FilteredRole**" are not insignificant.

- **Step 3**: Extract the "**VerifyCallerPrivileges**" method from the CRM SDK .NET library (Microsoft.Crm.ObjectModel). This method validates the privileges of the transaction target user for each role associated with the Business Unit associated with the transaction.

```
private void VerifyCallerPrivileges(Guid roleId, RolePrivilege[] privileges, ExecutionContext context)
{
    Guid guid = this.RetrieveRoleBusinessUnitId(roleId, context);
    foreach (RolePrivilege privilege in privileges)
      privilege.BusinessUnitId = guid;
    this.VerifyCallerPrivileges(privileges, context);
}
```

Specifically, the contents of the CRM error or exception will be launched in a specific method "**ThrowPrivilegeNotFoundException**" if the privileges of the calling user are not comparable with those of the "Business Unit" and the "Team":

```
private static void ThrowPrivilegeNotFoundException(Guid userId, RolePrivilege rolePrivilege,
ExecutionContext context)
{
    Microsoft.Crm.Metadata.Privilege privilege = MetadataCache.GetInstance((IOrganizationContext)
context).Privileges[(object) rolePrivilege.PrivilegeId];
    throw new CrmSecurityException(string.Format((IFormatProvider) CultureInfo.InvariantCulture,
"RoleService::VerifyCallerPrivileges failed. User: {0}, PrivilegeName: {1}, PrivilegeId: {2}, Depth: {3},
BusinessUnitId: {4}", (object) userId, privilege != null ? (object) privilege.Name : (object) "Privilege not
found in cache", (object) rolePrivilege.PrivilegeId, (object) rolePrivilege.Depth, (object)
rolePrivilege.BusinessUnitId), -2147220960);
}
```

How to resolve the specific error: Can't add members to the default business unit team?

The error is of type CrmException and type "SystemFailure":

```
ErrorCodes.ErrorMessages.Add((object) - 2147187957, (object) "Can't add members to the default business unit team");
ErrorCodes.ErrorTypes.Add((object) - 2147187957, (object) ErrorCodes.ErrorType.SystemFailure);
```

- **Step 1**: Extract the method associated with the error on transaction from the "CRM Trace": "**CheckPrivilegesAndAccess**". Retrieving should allow us to determine the service generates the error, in this case "TeamServiceInternal".

- **Step 2**: Extract the SQL execution query into the **SQL Profiler** (views, stored procedures if applicable):

  The stored procedure "p_... " is the stored procedure that searches for the privileges associated with the user that is logged in.
  The filtered views "..." and "..." are not negligible.

- **Step 3**: Extract the "**CheckPrivilegesAndAccess**" method from the CRM SDK .NET library (Microsoft.Crm.ObjectModel). This method validates the privileges of the transaction user for each privilege associated with the "Team" associated with the transaction.

```
private bool CheckPrivilegesAndAccess(Guid teamId, int errorCode, bool allowDefaultBuTeam,
ExecutionContext context)
{
    bool flag = this.AllowDefaultBuTeamMembership(allowDefaultBuTeam, context);
    if (!this.Provisioning &&
!context.Caller.Equals(SecurityLibrary.GetOrganizationSystemUser(context.OrganizationId,
(IOrganizationContext) context)))
    {
      if (!flag && this.IsDefaultTeam(teamId, context))
        throw new CrmException(errorCode, new object[0]);
      this.CheckPrivilegeAndAccess(teamId, PrivilegeType.Write, AccessRights.WriteAccess, context);
      this.ValidateCallerPrivileges(teamId, context);
      this.ValidateCallerAttributePrivileges(teamId, context);
    }
    return flag;
}
```

```
private void CheckPrivilegeAndAccess(Guid teamId, PrivilegeType privType, AccessRights accessrights,
ExecutionContext context)
   {
    EntityMetadata entity = MetadataCache.GetInstance((IOrganizationContext)
context).GetEntity(this.PlatformName);
    SecurityLibrary.CheckPrivilege(context.Caller, entity.GetPrivilege(privType).Id, context);
    BusinessManagementUtility.CheckAccess(new BusinessEntityMoniker(teamId, this.PlatformName,
(IOrganizationContext) context), accessrights, context);
   }
```

CONCLUSION

GENERIC AND SPECIFIC CRM ERRORS

ENTITYSERVICEINTERNAL<T>

CONCLUSION: GENERIC AND SPECIFIC CRM ERROR

"Troubleshooter" a CRM error during a data migration is mainly related to the D365 security model.

Even if an error is more generic, the error becomes with the use of tools such as "Jet Brains", "SQL Profiler" a specific error that will allow to target the solution to be implemented in our migration code.

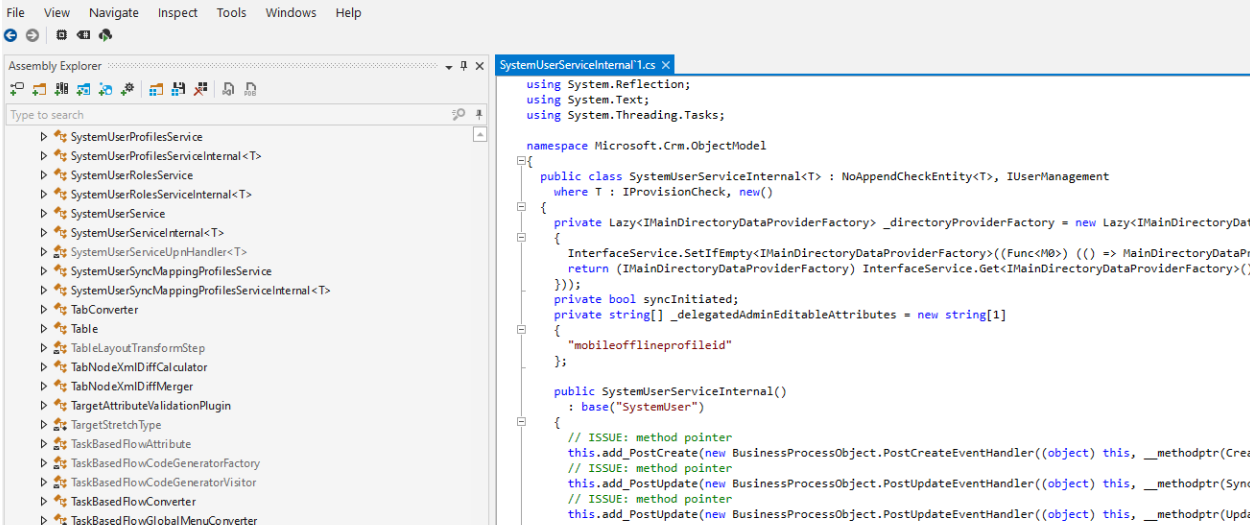
Finally, successful data migration requires a correction of data related to the CRM security model: System User, Team, Business Unit, Role, and Privilege.

The extension of service contracts related to CRM entities: "EntityServiceInternal<T>".

Regardless of whether an error is generic or specific and regardless of which entity the exception generated by the CRM platform is concerned, the dynamic library (DLL) "**Microsoft.Crm.ObjectModel**" plays a key role in the "troubleshooting" process.

The .NET library in question contains an extension for each native entity to the system (System User, Business Unit, Annotation...) for all service contracts executed by the platform and thus initiated by user action via the client interface (CreateRequest, UpdateRequest, SetStateRequest...).The execution of each extension contains validation actions that, in the case of failure, generate the "CRM Exception" type exception making the error a general or specific error (as emphasizing a specific exception).

For example, the System User entity has its own extension: "SystemUserServiceInternal<T>". This extension is called by all functionalities accessible through the CRM interface ("create", "disable", "enable", etc.).
In terms of development (C#), the "SystemUserServiceInternal" class contains 23 public methods that are accessible by other .NET libraries: some perform "CRUD" actions (create, read, update and delete), others perform "validation" actions (i.e. "IsSystemAdministrator", "WhoAmI") and others perform "synchronization" actions (i.e. "SynchronizeMailboxEntity").



SystemUserProfilesService
SystemUserProfilesServiceInternal<T>
SystemUserRolesService
SystemUserRolesServiceInternal<T>
SystemUserService
SystemUserServiceInternal<T>
SystemUserServiceUpnHandler<T>
SystemUserSyncMappingProfilesService
SystemUserSyncMappingProfilesServiceInternal<T>
TabConverter
Table
TableLayoutTransformStep
TabNodeXmlDiffCalculator
TabNodeXmlDiffMerger
TargetAttributeValidationPlugin
TargetStretchType
TaskBasedFlowAttribute
TaskBasedFlowCodeGeneratorFactory
TaskBasedFlowCodeGeneratorVisitor
TaskBasedFlowConverter
TaskBasedFlowGlobalMenuConverter

```
using System.Reflection;
using System.Text;
using System.Threading.Tasks;

namespace Microsoft.Crm.ObjectModel
{
  public class SystemUserServiceInternal<T> : NoAppendCheckEntity<T>, IUserManagement
    where T : IProvisionCheck, new()
  {
    private Lazy<IMainDirectoryDataProviderFactory> _directoryProviderFactory = new Lazy<IMainDirectoryDat
    {
      InterfaceService.SetIfEmpty<IMainDirectoryDataProviderFactory>((Func<M0>) (() => MainDirectoryDataPr
      return (IMainDirectoryDataProviderFactory) InterfaceService.Get<IMainDirectoryDataProviderFactory>()
    }));
    private bool syncInitiated;
    private string[] _delegatedAdminEditableAttributes = new string[1]
    {
      "mobileofflineprofileid"
    };

    public SystemUserServiceInternal()
      : base("SystemUser")
    {
      // ISSUE: method pointer
      this.add_PostCreate(new BusinessProcessObject.PostCreateEventHandler((object) this, __methodptr(Crea
      // ISSUE: method pointer
      this.add_PostUpdate(new BusinessProcessObject.PostUpdateEventHandler((object) this, __methodptr(Sync
      // ISSUE: method pointer
      this.add_PostUpdate(new BusinessProcessObject.PostUpdateEventHandler((object) this, __methodptr(Upda
```

The extension of the service contracts of the CRM "EntityServiceInternal<T>" generates various types of exceptions: "CRM Exception", "DataServiceRequestException", "Exception" to generate specific exceptions ("CrmArgumentException", "CrmArgumentNullException"...).

Typically, the error or exception generated by the CRM platform contains an error code that can be traced into the "**ErrorCodes**" class that can be retrieved from the .NET (DLL) "**Microsoft.Crm**" library.

References

[1] CRM Error or Exception of type “Microsoft.Xrm.Sdk.OrganizationServiceFault”
[2] CRM Error or Exception of type “Microsoft.Xrm.Sdk.OrganizationServiceFault”
[3] CRM Error or Exception of type “Microsoft.Xrm.Sdk.OrganizationServiceFault”
[4] Dynamics CRM service provides way to trace the details and display it whenever plug-in fails and throws an exception. Log file is created with technical details. See references [1] to [3] at the page 8 and 9.

Legal Note

This document may be circulated without prior written consent. The information is not intended as an offer or solicitation. You may quote or use the white paper or part of it for any type of document, research or presentation, citation and referencing rules apply to this white paper. For more information, please do not hesitate to contact the author.

Glossary

Jet Brains dotPeek: Free .NET Decompiler and Assembly Browser tool (https://www.jetbrains.com/decompiler/)

OrganizationService : “A wrapper class around the IOrganizationService that manages the details of setting up an OrganizationServiceProxy or other arbitrary IOrganizationService objects. When paired with an OrganizationServiceCache object, becomes a cached service.” (https://docs.microsoft.com/en-us/previous-versions/dynamicscrm-2015/developers-guide/gg670587(v=crm.7))

StackTrace : “a stack trace is a representation of a call stack at a certain point in time, with each element representing a method invocation. The stack trace contains all invocations from the start of a thread until the point it’s generated. This is usually a position at which an exception takes place.” (https://stackify.com/java-stack-trace/)

Troubleshooting : “Troubleshooting is a process that helps people identify issues or problems occurring in a system.” (https://stackify.com/troubleshooting-vs-debugging-whats-the-difference-best-practices/)

About Us and philosophy

Thierry Sinassamy - I strongly advocate a methodology of troubleshooting an issue. The approach could be the key to a resolution of a specific problem.

Contacts

DataObjectException Inc.
Thierry Sinassamy
1-514-883-6068
thierry.sinassamy@dataobjectexception.com
https://www.yuzuai.ca