

Tech Talk

D365 and Dataverse - Integration Overview

Presenter(s):

Corina Balan, Sr. Solution Architect

Michele Mazzucco, Sr. Solution Architect

Co-Presenter(s)/Q&A:

Ali Adamu, Sr. Solution Architect



TechTalk Series

- ✓ Session 1: Dynamics 365 Integration – General Guidance – **Completed**, see QR code for the link



- **Session 2: Integration patterns for Dataverse**
- *Session 3: Integration patterns for F&O – Planned for 13th Nov*
- *Session 4 – Complex and mixed scenarios – Planned for 4th Dec*

Integration components - Overview



Event Grid



Service Bus



Web Jobs



Data flow



Azure Function



Data Factory



Logic Apps



API management



Power Automate

Inventory
Visibility

MES
Integration

Demand
planning*

Sensor Data
Intelligence

Pricing
service*

Connected
Field Service

Finance and Operations Apps

PunchOut
e-Procurement

Electronic
Reporting

Invoice
capture

Data entities

Data Management
Framework

OData

Custom classes
External endpoints

Custom services

Dual Write

Business Events
Data Events

Virtual Table

Synapse Link

Office 365

Dataverse

OData/Web API

TDS Endpoint

Plug-in (Synch)

WebHook

Plug-in (Asynch)

Microsoft Teams

Near real-time

Asynchronous

Synchronous

AAD Authentication

RESTful services

JSON message format



Outbound

Inbound

*Preview

Agenda

- Dataverse inbound integration patterns
 - Overview of integration methods
 - Message processing with Azure Service Bus
 - Batch import with Azure Data Factory
 - Embed real-time data with UI controls
 - Scaling and security remarks
- Dataverse outbound integration patterns
 - Plugins to external web services and Azure
 - Webhooks
 - Business Events
 - Batch export with Azure Synapse Link for Dataverse
- Resources
- Q&A



Dataverse inbound integrations

Dataverse inbound integrations

Different approaches

Dataverse APIs

Organization Service – defines the operations supported by the platform as messages, provides an SDK for .NET development.

Web API – can be used across a wide variety of programming languages, platforms, and devices. All the same operations as the SDK for .NET but presented in RESTful style.

Dataverse connectors

Power Automate – SaaS, low-code/no-code, using prebuilt or custom connectors.

Logic Apps – PaaS, low-code/no-code or pro development, using prebuilt or custom connectors.

Data Factory – managed cloud service for complex ETL and data integration projects.

UI integration

Virtual Tables – integrate data from external systems, represented as Dataverse tables.

Custom Controls/Pages – add full pages, dialogs, or panes in a model-driven app with the flexibility of canvas designer.

Reports - embed Power BI reports in model-driven apps.

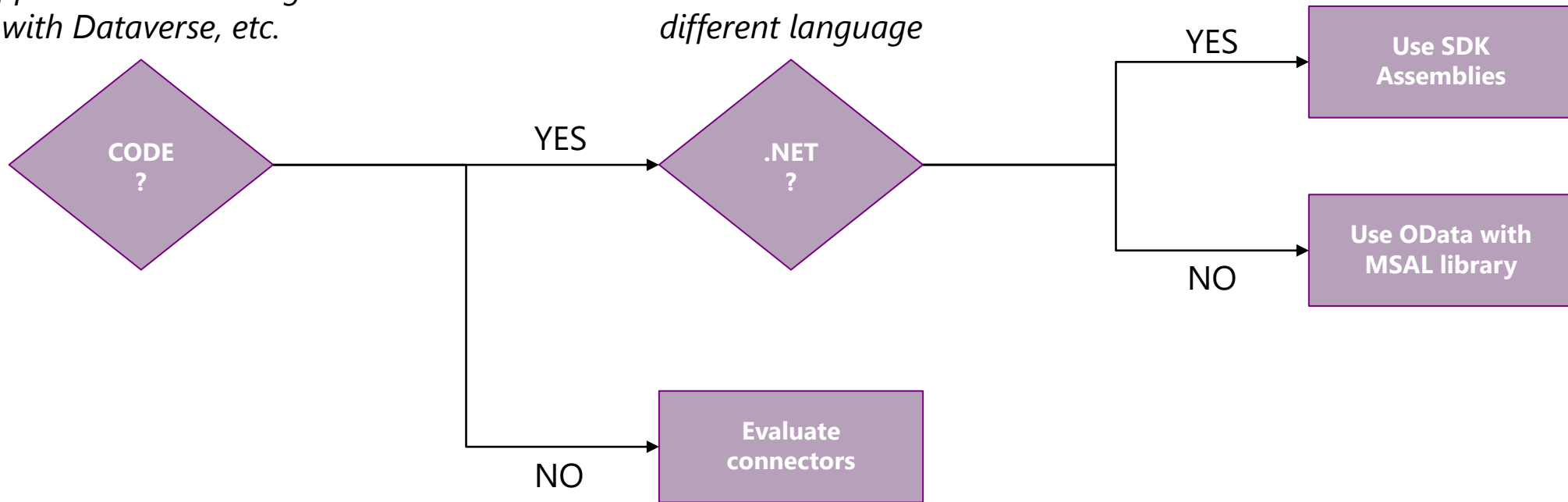
Define your approach

Do you **need** to write code to interact with Dataverse?

e.g.: you are implementing a web service, or custom application interacting with Dataverse, etc.

Can your code be implemented with .NET or are you in a different context?

e.g.: extending an existing application which is written in a different language



Define your approach

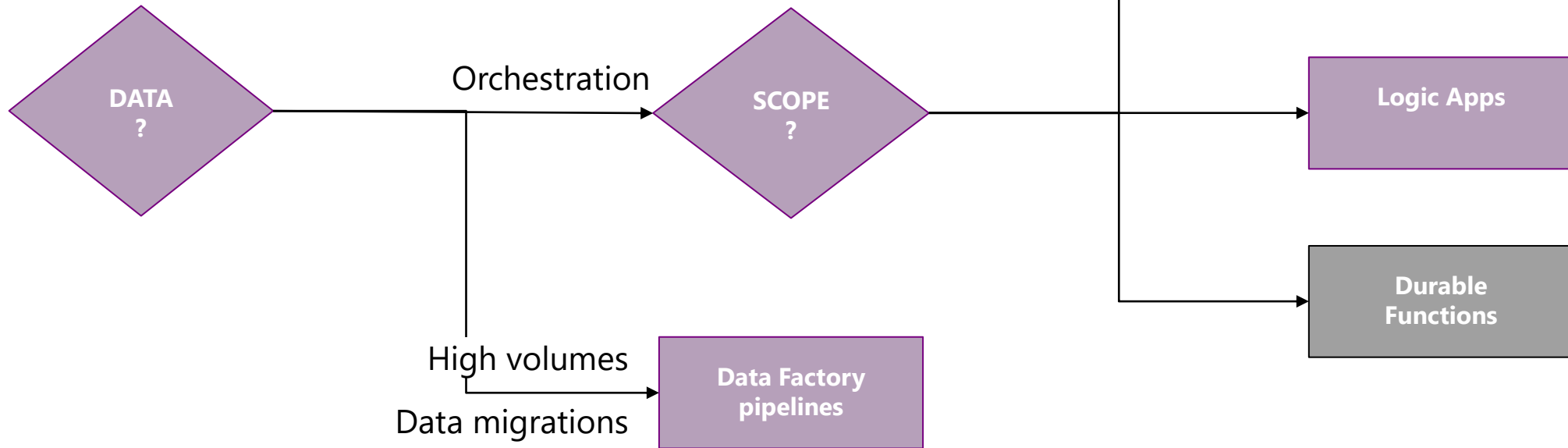
Are you implementing a data-centric approach to move huge amount of data?

Are you implementing event orchestration between applications?

Licensing/cost management?

ALM strategy (Power Platform solutions vs Azure resources)?

Design tools/skills?



Define your approach



[Web API and the Organization service | Microsoft Learn](#)



[Learn about MSAL | Microsoft Learn](#)



[Get started with virtual tables | Microsoft Learn](#)



[Converge model-driven and canvas apps using the custom page | Microsoft Learn](#)



[Embed a Power BI report in a model-driven app main form | Microsoft Learn](#)



[Integration and automation platform options in Azure | Microsoft Learn](#)



[Power Automate connectors | Microsoft Learn](#) - [Logic Apps connectors | Microsoft Learn](#)



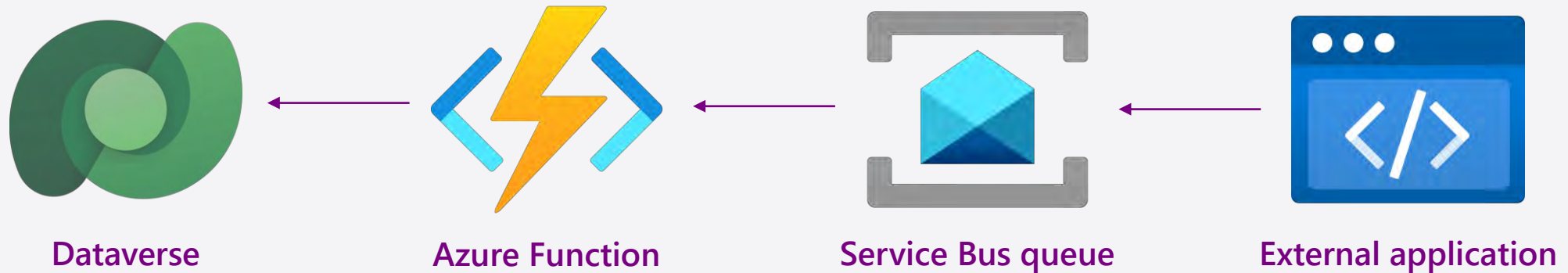
[Introduction to Azure Data Factory - Azure Data Factory | Microsoft Learn](#)

Processing an incoming message

Example scenario




Inbound example – message from Service Bus queue



Integrate data with Azure service bus

Service Bus allows the management of simple queues (Send-Receive scenario) or more complex topic/subscription definition (Publish-Subscribe scenario). External applications can add a message to a queue by performing HTTP POST operations. Azure Functions can be triggered by the event of a message being added to a queue. Inside the Azure Function, .NET code can interact with the Organization Service to perform operations against Dataverse.

Inbound example – message from Service Bus queue




Service Bus

Microsoft

Azure Service

Connect apps on an enterprise message queueing and pub-sub platform using open protocols including JMS.


Create 






integrationworkshop


Service Bus Namespace


Create queue 


Service Bus

Name * 

Max queue size

1 GB 

Max delivery count * 

Message time to live 

Days

Hours

Minutes

Seconds


 Search to filter items by name...

Name	Status	Message count	Active messages	Dead-letter messages	Scheduled messages	Max size
crm-importcontacts	Active	0	0	0	0	1024 MB

Queues

1

Inbound example – message from Service Bus queue





Function App

Microsoft

Azure Service

Write any function in minutes – whether to run a simple job that cleans up a database or build a more complex architecture. Creating functions is easier than ever before;

Create 



Azure Functions

C#

Azure

Cloud

Functions worker ⓘ

.NET 6.0 (Long Term Support) ▾

Function ⓘ

Service Bus Queue trigger ▾

☒ Use Azurite for runtime storage account (AzureWebJobsStorage) ⓘ

☐ Enable Docker ⓘ

Connection string setting name ⓘ

integrationworkshop

Queue name ⓘ

crm-importcontacts

Name	Trigger	Status	Monitor
ReceiveServiceBusQueueMessage	Service Bus	 Enabled	Invocations and more

Inbound example – message from Service Bus queue

```
public class ReceiveServiceBusQueueMessage
{
    private readonly ILogger _logger;

    public ReceiveServiceBusQueueMessage(ILoggerFactory loggerFactory)
    {
        _logger = loggerFactory.CreateLogger<ReceiveServiceBusQueueMessage>();
    }

    [Function(nameof(ReceiveServiceBusQueueMessage))]
    public void Run([ServiceBusTrigger("crm-importcontacts", Connection = "integrationworkshop")]
        ServiceBusReceivedMessage message)
    {
        // log message details
        _logger.LogInformation("Message ID: {id}", message.MessageId);
        _logger.LogInformation("Message Body: {body}", message.Body);
        _logger.LogInformation("Message Content-Type: {contentType}", message.ContentType);

        // deserialize message body
        ContactMessage? contact = JsonSerializer.Deserialize<ContactMessage>(message.Body.ToString());

        // call Dataverse custom API
        CreateContact(contact);
    }

    public void CreateContact(ContactMessage? contact)
    {
    }
}
```

```
public class ContactMessage
{
    public string FirstName { get; set; }
    public string LastName { get; set; }
    public string Email { get; set; }
    public string CompanyCode { get; set; }
    public string CompanyName { get; set; }
}
```


Inbound example – message from Service Bus queue

```
// connect Dataverse client using managed identity
using
(
    ServiceClient client = new ServiceClient
    (
        new Uri(environment),
        tokenProviderFunction: async (par) =>
        {
            var managedIdentity = new DefaultAzureCredential();
            return
            (
                await managedIdentity.GetTokenAsync
                (
                    new Azure.Core.TokenRequestContext(new[] { $"{environment}/.default" })
                )
            ).Token;
        }
    )
)
```

```
// init new request for the custom API
OrganizationRequest req =
    new OrganizationRequest("mike_CreateContact");
req["FirstName"] = contact.FirstName;
req["LastName"] = contact.LastName;
req["Email"] = contact.Email;
req["CompanyCode"] = contact.CompanyCode;
req["CompanyName"] = contact.CompanyName;

// try to execute the request and get the result
try
{
    OrganizationResponse res = client.Execute(req);
    _logger.LogInformation
        ($"Created new contact {res["ContactId"]}");
}
catch (Exception ex)
{
    _logger.LogInformation
        ($"Error processing message: {ex.Message}");
}
```

Inbound example – message from Service Bus queue

API

Custom API

Edit API

Delete API

Unique Name

mike_CreateContact

Name

CreateContact

Display Name

CreateContact

Description

CreateContact

Allowed Custom Processing Step Type

Sync and Async

Binding type

Global

Bound Entity

PluginType

Execute Privilege Name

prvCreateContact

IsFunction

No

Enabled For Workflow

No

IsPrivate

No

IsCustomizable

True

IsManaged

Unmanaged

Request Parameters (Input)

New Request Param

	Unique Name	Type	Is Optional
▶	FirstName	String	No
	LastName	String	No
	Email	String	No
	CompanyCode	String	Yes
	CompanyName	String	Yes

Response Properties (Output)

New Response Prop

	Unique Name	Type
▶	ContactId	Guid

Inbound example – message from Service Bus queue

```
// check if name is passed
if (String.IsNullOrEmpty(firstname) || String.IsNullOrEmpty(lastname))...

// check if email is passed
if (String.IsNullOrEmpty(email))...

// init query to check for duplicates
QueryExpression contactquery = new QueryExpression("contact");

[DUPLICATE DETECTION]

// check for duplicates
if (contacts.Entities.Count > 0)...

// init new contact
Entity contact = new Entity("contact");
contact["firstname"] = firstname;
contact["lastname"] = lastname;
contact["emailaddress1"] = email;

// init entityreference for account
EntityReference accountref = null;

// check if company code is passed
if (!String.IsNullOrEmpty(companycode))
{
    // use alternate key to set entityreference
    accountref = new EntityReference("account", "accountnumber", companycode);
}

// check if company name is passed
else if (!String.IsNullOrEmpty(companyname))...
```

```
// set account lookup on contact
contact["parentcustomerid"] = accountref;

// create new contact and return GUID as output
Guid newcontactid = service.Create(contact);
context.OutputParameters["ContactId"] = newcontactid;
```

General Configuration Information

Message:	mike_CreateContact
Primary Entity:	none
Secondary Entity:	none
Filtering Attributes:	Message/Entity does not support Filtered Attributes
Event Handler:	(Plugin) CustomAPI.CreateContact
Name:	CustomAPI.CreateContact: mike_CreateContact of any Entity
Run in User's Context:	Calling User
Execution Order:	1

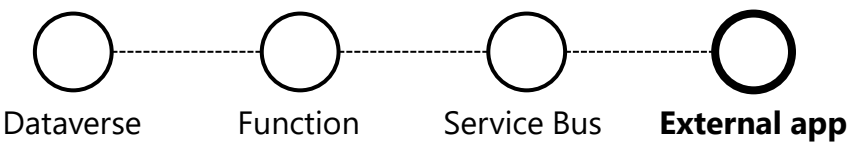
Eventing Pipeline Stage of Execution	Execution Mode	Deployment
<input type="radio"/> Pre-validation	<input type="radio"/> Asynchronous	<input checked="" type="checkbox"/> Server
<input type="radio"/> Pre-operation	<input checked="" type="radio"/> Synchronous	<input type="checkbox"/> Offline
<input checked="" type="radio"/> Post-operation		

Processing an incoming message

Demo



Demo – message from Service Bus queue



Azure Service Bus / Send Message

Save

POST

https://integrationworkshop.servicebus.windows.net/crm-importcontacts/messages

Send

Params

Authorization

Headers (10)

Body

Pre-request Script

Tests

Settings

Cookies

☐ none

☐ form-data

☐ x-www-form-urlencoded

☒ raw

☐ binary

☐ GraphQL

JSON

Beautify

1

{

2

"FirstName": "Michele",

3

"LastName": "Mazzucco",

4

"Email": "michele.mazzucco@microsoft.com",

5

"CompanyCode": "MSFT",

6

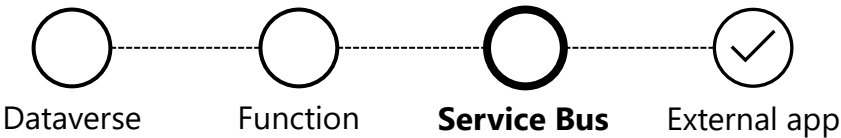
"CompanyName": "Microsoft Corporation"

7

}

Response

Demo – message from Service Bus queue



Message Counts

Active
0 MESSAGES

Scheduled
0 MESSAGES

Peek Mode ▾ | [Send messages](#) | [Refresh](#) | [Show message body ▾](#) | [Settings](#) | [Learn more](#) | [Give feedback](#)

[Queue \(1\)](#) | [Dead-letter \(0\)](#)

[Peek from start](#) | [Peek next messages](#) | [Peek with options](#) | [Re-send selected messages](#) | [Download selected message body](#)

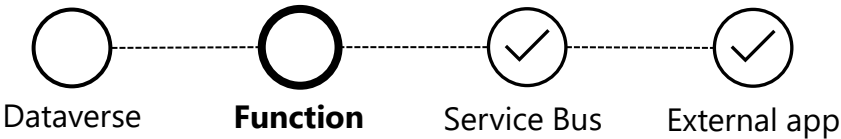
Showing 0 of 1 messages

<input type="checkbox"/>	Sequence Number	Message ID	Enqueued Time	Delivery Count	State	Body ...
--------------------------	-----------------	------------	---------------	----------------	-------	----------

Peek or receive messages to view them here.



Demo – message from Service Bus queue



Invocations Logs

Success Count

✓ 6

Last 30 Days

Error Count

✗ 0

Last 30 Days

Invocation Traces

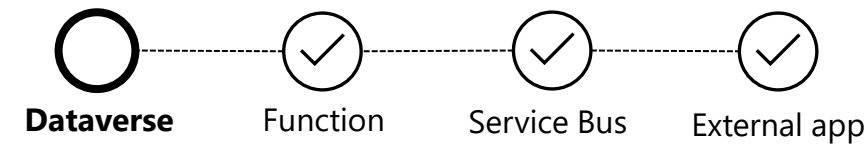
The twenty most recent function invocation traces. For more advanced analysis, run the query in Application Insights.

[Run query in Application Insights](#) [Refresh](#)

Filter invocations

Date (UTC)	Success	Result Code	Duration (ms)	Operation Id
2023-11-01 17:04:38.511	✓ Success	0	3445	f03cb175c151cb61040be850ba0ffc6
2023-11-01 17:03:09.659	✓ Success	0	8991	e646452234392140e51a9e7251931ed5
2023-10-29 15:24:05.147	✓ Success	0	423	740e13f7937c08c70e16ddd21df9a7d2
2023-10-29 15:23:34.291	✓ Success	0	375	149269797e7303ea22269d41161300d3
2023-10-29 15:21:38.144	✓ Success	0	4525	672e2bd92636175eee1fef10907b250f

Demo – message from Service Bus queue



Active Contacts ▼				Edit columns	Edit filters	<input type="text" value="Filter by keyword"/>
<input type="checkbox"/>	Full Name ↑ ▼	Email ▼	Company Name ▼	Business Phone ▼		
<input type="checkbox"/>	Jim Glynn (sample)	someone_j@example.com	Coho Winery (sample)	555-0109		
<input type="checkbox"/>	Maria Campbell (sample)	someone_d@example.com	Fabrikam, Inc. (sample)	555-0103		
<input type="checkbox"/>	Nancy Anderson (sample)	someone_c@example.com	Adventure Works (sample)	555-0102		
<input type="checkbox"/>	Patrick Sands (sample)	someone_k@example.com	Alpine Ski House (sample)	555-0110		
<input type="checkbox"/>	Paul Cannon (sample)	someone_h@example.com	Alpine Ski House (sample)	555-0107		
<input type="checkbox"/>	Rene Valdes (sample)	someone_i@example.com	A. Datum Corporation (sample)	555-0108		
<input type="checkbox"/>	Robert Lyon (sample)	someone_g@example.com	Contoso Pharmaceuticals (sample)	555-0106		
<input type="checkbox"/>	Scott Konersmann (sample)	someone_f@example.com	City Power & Light (sample)	555-0105		
<input type="checkbox"/>	Sidney Higa (sample)	someone_e@example.com	Blue Yonder Airlines (sample)	555-0104		
<input type="checkbox"/>	Susan Burk (sample)	someone_l@example.com	A. Datum Corporation (sample)	555-0111		
<input type="checkbox"/>	Susanna Stubberod (sample)	someone_b@example.com	Litware, Inc. (sample)	555-0101		
<input type="checkbox"/>	Thomas Andersen (sample)	someone_m@example.com	Coho Winery (sample)	555-0112		
<input type="checkbox"/>	Yvonne McKay (sample)	someone_a@example.com	Fourth Coffee (sample)	555-0100		

Rows: 13

Recommended topics



SCALING AND PERFORMANCE



SECURITY



BEST PRACTICE AND GUIDANCE



[Manage application users in the Power Platform admin center - Power Platform | Microsoft Learn](#)



[API limits overview \(Microsoft Dataverse\) - Power Apps | Microsoft Learn](#)



[Service protection API limits \(Microsoft Dataverse\) - Power Apps | Microsoft Learn](#)



[Managed identities - Azure App Service | Microsoft Learn](#)



[Create and use custom APIs \(Microsoft Dataverse\) - Power Apps | Microsoft Learn](#)



[Work with alternate keys \(Microsoft Dataverse\) - Power Apps | Microsoft Learn](#)



[Azure Service Bus messaging - queues, topics, and subscriptions - Azure Service Bus | Microsoft Learn](#)



[Service Bus dead-letter queues - Azure Service Bus | Microsoft Learn](#)



[Azure Service Bus trigger for Azure Functions | Microsoft Learn](#)



[Azure Functions error handling and retry guidance | Microsoft Learn](#)



[Target-based scaling in Azure Functions | Microsoft Learn](#)



[Expose APIs from functions using Azure API Management | Microsoft Learn](#)

Alternative – consume message with Power Automate



Dataverse



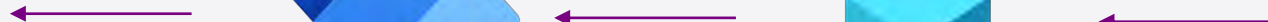
Power Automate



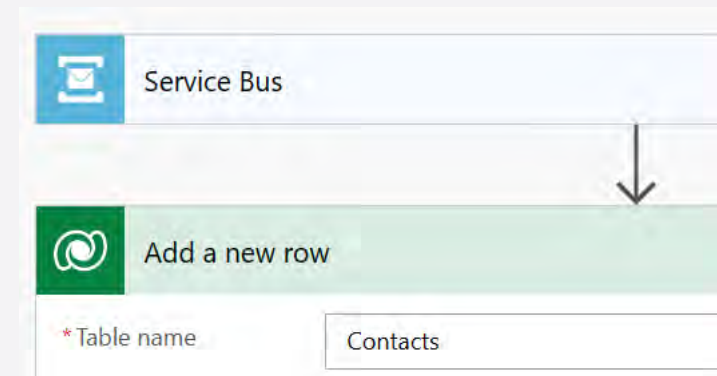
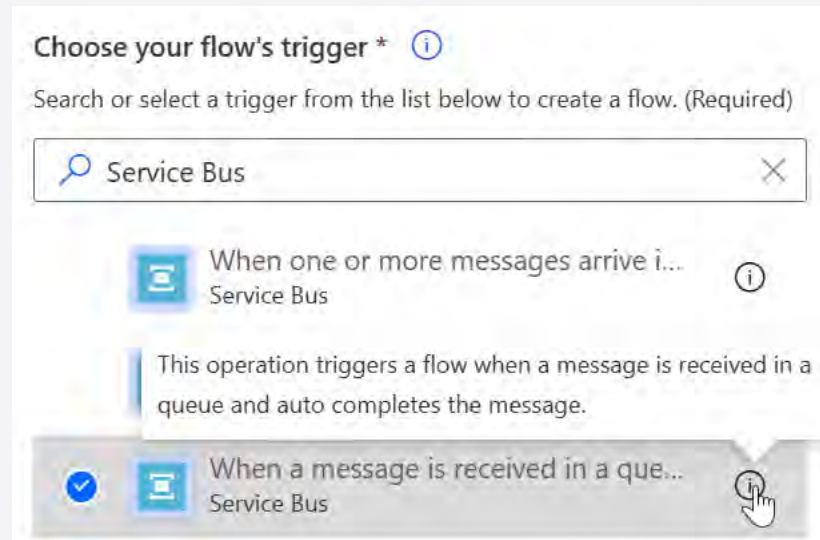
Service Bus queue



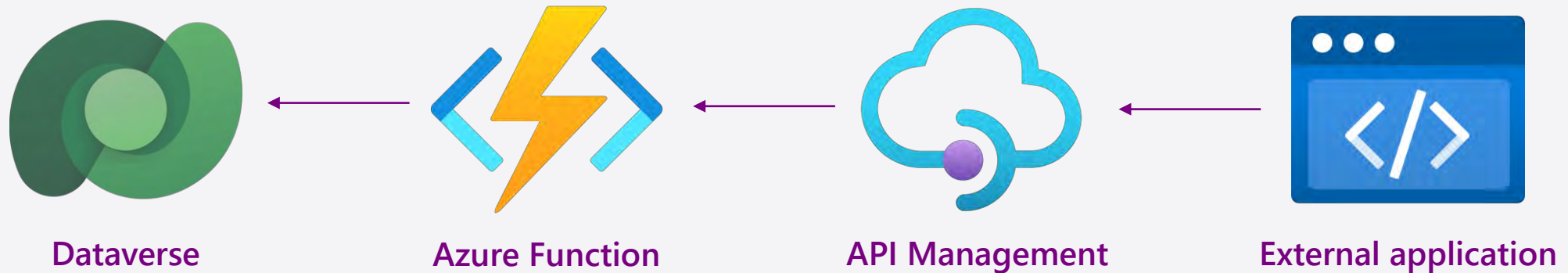
External application



Service Bus and
Dataverse connectors
for Power Automate



Alternative – expose Azure Function as API



Expose an Azure Function with API Management

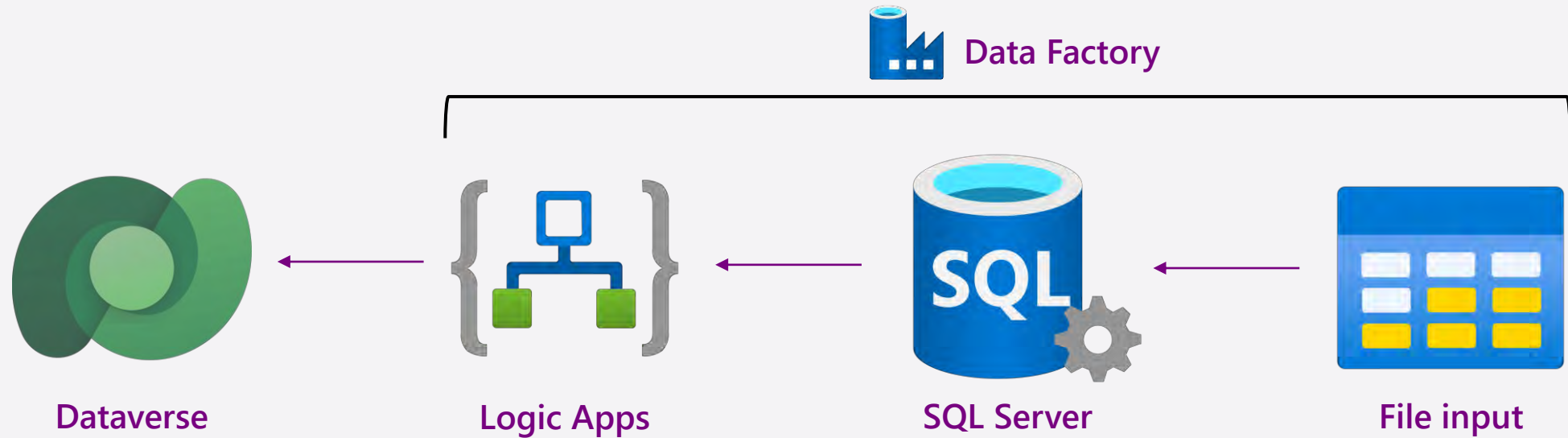
Azure Functions integrates with Azure API Management in the portal to let you expose your HTTP trigger function endpoints as REST APIs. These APIs are described using an OpenAPI definition. This JSON (or YAML) file contains information about what operations are available in an API. It includes details about how the request and response data for the API should be structured. By integrating your function app, you can have API Management generate these OpenAPI definitions.

Bulk import of records from a database table

Example scenario



Inbound example – batch import of database rows



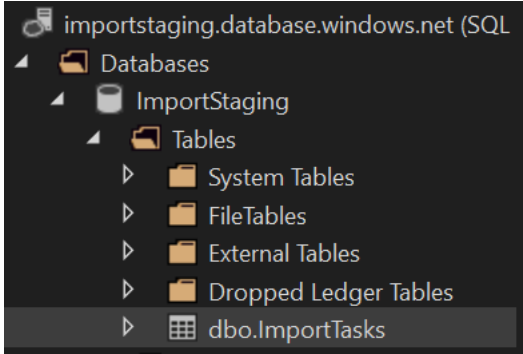
Azure Data Factory

A Data Factory can have one or more pipelines. A pipeline is a logical grouping of activities that together perform a task. The pipeline allows you to manage the activities as a set instead of each one individually. You deploy and schedule the pipeline instead of the activities independently. The activities in a pipeline define actions to perform on your data. For example, you can use a copy activity to copy data from a file into SQL Server. Then, use a data flow activity to process and transform data.

Inbound example – batch import of database rows

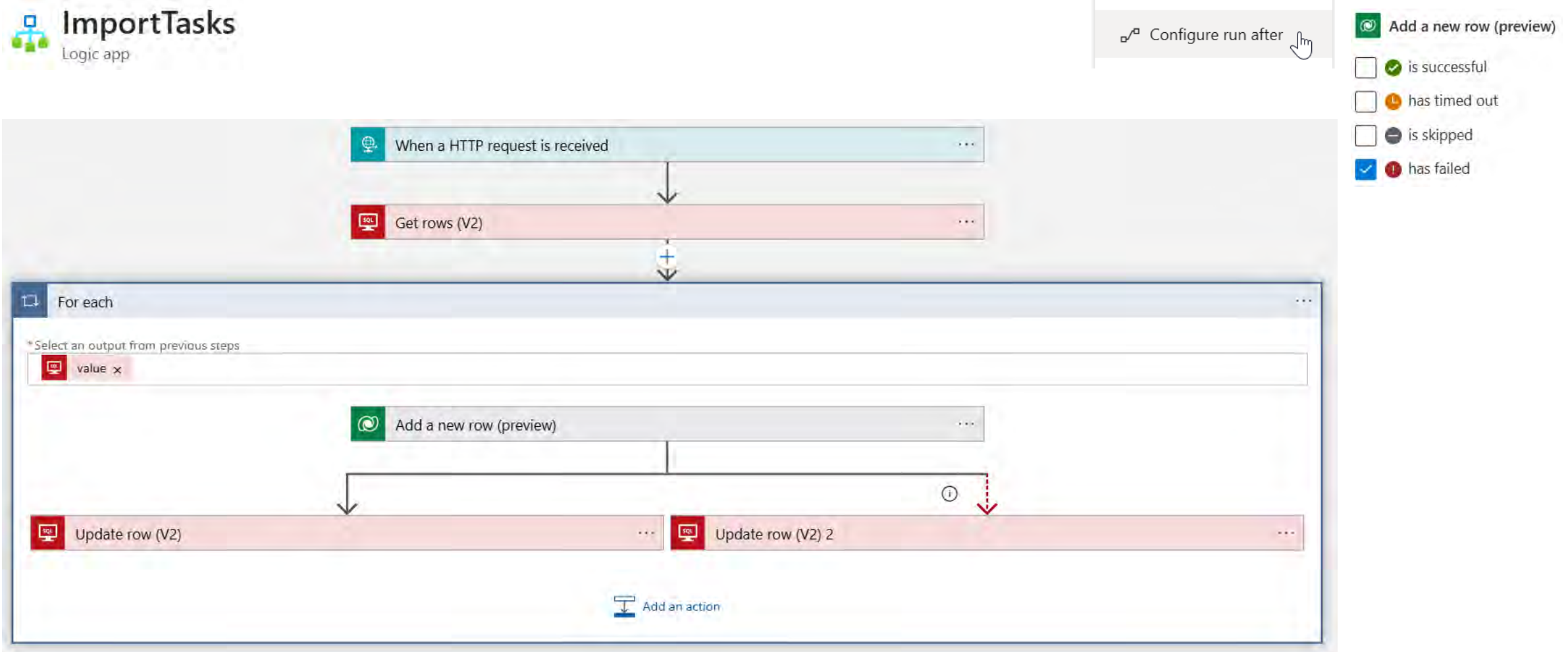
	A	B	C	D	E	F
1	Title	Description	Category	Duration	DueDate	CreatedOn
2	Aenean fermentum.	Sed sagittis. Nam congue, risus semper porta volutpat, quam pede lobortis ligula, sit amet eleifend pede libero quis orci. Nullam molestie	Human Resources	75	2/22/2024 18:20	10/20/2023 1:21
3	Phasellus in felis.	Maecenas ut massa quis augue luctus tincidunt. Nulla mollis molestie lorem. Quisque ut erat. Curabitur gravida nisi at nibh. In hac	Sales	120	1/12/2024 20:46	9/21/2023 23:58
4	Donec dapibus.	Donec diam neque, vestibulum eget, vulputate ut, ultrices vel, augue. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices	Human Resources	30	2/24/2024 14:23	9/22/2023 4:13
5	In est risus, auctor sed, tristique in, tempus sit amet, sem.	Maecenas ut massa quis augue luctus tincidunt. Nulla mollis molestie lorem. Quisque ut erat. Curabitur gravida nisi at nibh. In hac	Finance	90	3/1/2024 9:43	9/8/2023 4:56
6	Fusce consequat.	Duis bibendum. Morbi non quam nec dui luctus rutrum. Nulla tellus.	Finance	120	2/14/2024 14:10	10/18/2023 10:18
7	Integer tincidunt ante vel ipsum.	Nulla ut erat id mauris vulputate elementum. Nullam varius. Nulla facilisi. Cras non velit nec nisi vulputate nonummy. Maecenas tincidunt	Sales	15	2/24/2024 15:34	9/10/2023 15:06
8	Maecenas ut massa quis augue luctus tincidunt.	Etiam vel augue. Vestibulum rutrum rutrum neque. Aenean auctor gravida sem.	Marketing	90	2/6/2024 2:37	10/26/2023 21:59
9	In blandit ultrices enim.	Pellentesque at nulla. Suspendisse potenti. Cras in purus eu magna vulputate luctus. Cum sociis natoque penatibus et magnis dis parturient	Sales	120	11/29/2023 16:36	10/16/2023 17:53
10	Vivamus vel nulla eget eros elementum pellentesque.	Lorem ipsum dolor sit amet, consectetur adipiscing elit. Proin risus. Praesent lectus. Vestibulum quam sapien, varius ut, blandit non, nulla	Marketing	30	11/9/2023 13:13	10/8/2023 16:36
11	Nullam molestie nibh in lectus.	Maecenas leo odio, condimentum id, luctus nec, molestie sed, justo. Pellentesque viverra pede ac diam. Cras pellentesque volutpat dui	Marketing	120	1/23/2024 22:48	10/21/2023 14:20
12	Morbi a ipsum.	Fusce consequat. Nulla nisl. Nunc nisl. Duis bibendum, felis sed interdum venenatis, turpis enim blandit mi, in porttitor pede justo eu	Operations	15	11/19/2023 18:36	9/15/2023 1:29
13	Sed ante.	Sed ante. Vivamus tortor. Duis mattis egestas metus. Aenean fermentum. Donec ut mauris eget massa tempor convallis. Nulla neque	Operations	30	11/9/2023 8:19	10/15/2023 1:32
14	Nulla ut erat id mauris vulputate elementum.	Donec diam neque, vestibulum eget, vulputate ut, ultrices vel, augue. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices	Operations	30	11/4/2023 16:43	10/7/2023 10:19
15	Mauris ullamcorper purus sit amet nulla.	Pellentesque at nulla. Suspendisse potenti. Cras in purus eu magna vulputate luctus. Cum sociis natoque penatibus et magnis dis parturient	Human Resources	60	11/1/2023 22:02	10/21/2023 21:30
16	Curabitur convallis.	In hac habitasse platea dictumst. Morbi vestibulum, velit id pretium iaculis, diam erat fermentum justo, nec condimentum neque sapien	Marketing	60	1/31/2024 9:58	9/22/2023 9:30
17	Cras in purus eu magna vulputate luctus.	Aenean fermentum. Donec ut mauris eget massa tempor convallis. Nulla neque libero, convallis eget, eleifend luctus, ultrices eu, nibh	Sales	75	12/18/2023 8:24	10/30/2023 23:57
18	Cum sociis natoque penatibus et magnis dis parturient montes, mauris eget tempus imperdiet.	Proin eu mi. Nulla ac enim. In tempor, turpis nec euismod scelerisque, quam turpis adipiscing lorem, vitae mattis nibh ligula nec sem. In	Marketing	60	11/21/2023 6:57	10/3/2023 5:05
19	Etiam justo.	Sed ante. Vivamus tortor. Duis mattis egestas metus. Aenean fermentum. Donec ut mauris eget massa tempor convallis. Nulla neque	Sales	90	2/4/2024 3:38	9/1/2023 12:13
20	Duis bibendum.	Sed ante. Vivamus tortor. Duis mattis egestas metus.	Marketing	60	1/21/2024 13:37	9/27/2023 18:50
21	Praesent lectus.	Phasellus in felis. Donec semper sapien a libero. Nam dui. Proin leo odio, porttitor id, consequat in, consequat ut, nulla. Sed accumsan	Finance	120	3/10/2024 8:48	9/10/2023 19:13
22	Phasellus in felis.	Suspendisse potenti. In eleifend quam a odio. In hac habitasse platea dictumst.	Sales	75	12/5/2023 5:39	9/25/2023 20:10
23	Vivamus tortor.	Morbi porttitor lorem id ligula. Suspendisse ornare consequat lectus. In est risus, auctor sed, tristique in, tempus sit amet, sem.	Human Resources	120	11/26/2023 19:31	9/23/2023 23:31
24	Nulla mollis molestie lorem.	In sagittis dui vel nisl. Duis ac nibh. Fusce lacus purus, aliquet at, feugiat non, pretium quis, lectus.	Finance	120	3/7/2024 3:19	10/16/2023 21:30
25	Proin eu mi.	Donec diam neque, vestibulum eget, vulputate ut, ultrices vel, augue. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices	Operations	60	1/5/2024 1:11	9/20/2023 19:36
26	Sed ante.	Duis consequat dui nec nisi volutpat eleifend. Donec ut dolor. Morbi vel lectus in quam fringilla rhoncus. Mauris enim leo, rhoncus sed, vestibulum	Marketing	90	1/29/2024 0:08	9/1/2023 2:43
27	In sagittis dui vel nisl.	Phasellus sit amet erat. Nulla tempus. Vivamus in felis eu sapien cursus vestibulum. Proin eu mi. Nulla ac enim. In tempor, turpis nec euismod	Operations	30	2/23/2024 17:34	9/17/2023 17:29
28	Etiam faucibus cursus urna.	Lorem ipsum dolor sit amet, consectetur adipiscing elit. Proin risus. Praesent lectus. Vestibulum quam sapien, varius ut, blandit non, nulla	Finance	30	3/29/2024 21:57	9/14/2023 9:01
29	Nam dui.	Quisque id justo sit amet sapien dignissim vestibulum. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia	Marketing	15	11/16/2023 1:32	9/27/2023 3:41
30	Quisque id justo sit amet sapien dignissim vestibulum.	Integer tincidunt ante vel ipsum. Praesent blandit lacinia erat. Vestibulum sed magna at nunc commodo placerat.	Sales	75	11/1/2023 17:43	9/3/2023 20:18
31	Morbi non lectus.	Maecenas leo odio, condimentum id, luctus nec, molestie sed, justo. Pellentesque viverra pede ac diam. Cras pellentesque volutpat dui	Human Resources	120	3/7/2024 16:53	10/2/2023 8:38
32	Phasellus id sapien in sapien iaculis congue.	Fusce posuere felis sed lacus. Morbi sem mauris, laoreet ut, rhoncus aliquet, pulvinar sed, nisl. Nunc rhoncus dui vel sem. Sed sagittis	Marketing	45	12/8/2023 5:51	9/23/2023 10:25
33	Integer ac neque.	Proin leo odio, porttitor id, consequat in, consequat ut, nulla. Sed accumsan felis. Ut at dolor quis odio consequat varius. Integer ac leo	Marketing	105	2/15/2024 20:18	9/17/2023 16:42
34	Praesent blandit.	Maecenas ut massa quis augue luctus tincidunt. Nulla mollis molestie lorem. Quisque ut erat.	Finance	60	2/2/2024 1:22	10/21/2023 22:05
35	Fusce lacus purus, aliquet at, feugiat non, pretium quis, lectus.	Vestibulum ac est lacinia nisi venenatis tristique. Fusce congue, diam id ornare imperdiet, sapien urna pretium nisl, ut volutpat sapien	Marketing	75	2/11/2024 0:12	10/27/2023 13:17

Inbound example – batch import of database rows



	Name	Data Type	Allow Nulls	Default	
	Title	nvarchar(200)	<input checked="" type="checkbox"/>		
	Description	nvarchar(MAX)	<input checked="" type="checkbox"/>		
	Category	nvarchar(20)	<input checked="" type="checkbox"/>		
	Duration	int	<input checked="" type="checkbox"/>		
	DueDate	datetime	<input checked="" type="checkbox"/>		
	CreatedOn	datetime	<input checked="" type="checkbox"/>		
	ImportStatus	smallint	<input checked="" type="checkbox"/>	((0))	
	Id	int	<input type="checkbox"/>		

Inbound example – batch import of database rows



Inbound example – batch import of database rows



Linked service *

AzureBlobStorage

Test connection

Edit

+ New

[Learn more](#)

File path *

importdata

Directory

FT-ImportTasks.xlsx

Browse

<input type="checkbox"/>	Source	Type		Destination	Type
<input type="checkbox"/>	Title	abc String	→	Title	abc nchar
<input type="checkbox"/>	Description	abc String	→	Description	abc nvarchar
<input type="checkbox"/>	Category	abc String	→	Category	abc nchar
<input type="checkbox"/>	Duration	abc String	→	Duration	123 int
<input type="checkbox"/>	DueDate	abc String	→	DueDate	datetime

URL * ⓘ

https://[redacted].logic.azure...

Method * ⓘ

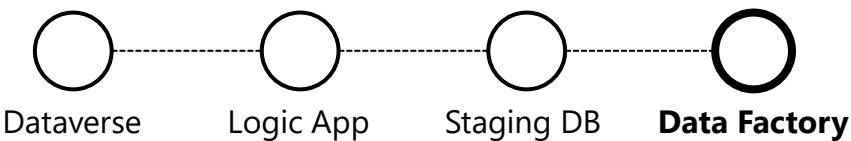
POST

Bulk import of records from a database table




Demo




Inbound example – batch import of database rows



Parameters Variables Settings Output









Pipeline run ID: f0bbf649-f9c7-4c4b-9a72-ad34bbfc4366   

Pipeline status  In progress

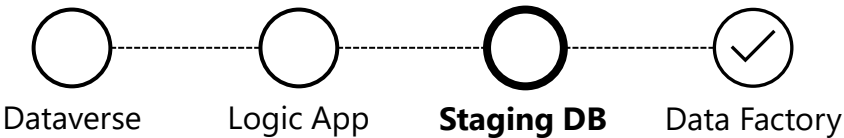
All status 

[Monitor in Azure Metrics](#)  [Export to CSV](#) 

Showing 1 - 1 of 1 items

Activity name 	Activity status 	Activity type 	Run start 	Duration 	Integration runtime	User properties 	Activi 
Excel into Staging	 Queued	Copy data	11/1/2023, 7:23:30 PM	6s			3ad7a

Inbound example – batch import of database rows



```
select * from ImportTasks
```

100 %

1

0

Ln: 1

Ch: 26

TABS

CRLF

T-SQL

Results

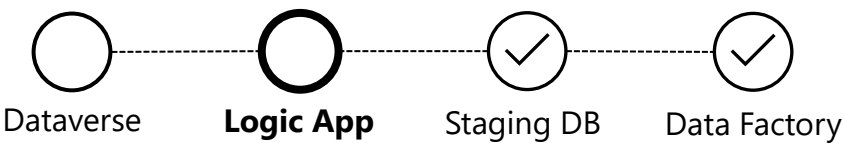
Message

Title	Description	Category	Duration	DueDate	CreatedOn	ImportStatus	Id
-------	-------------	----------	----------	---------	-----------	--------------	----

Query executed successfully at 7:25:33 PM

importstaging.database.wind... | ftadmin (89) | ImportStaging | 00:00:00 | 0 rows

Inbound example – batch import of database rows



All

▼

▼

Pick a date

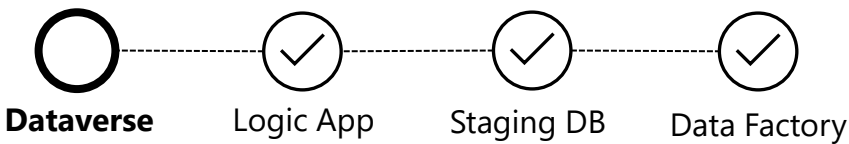
Pick a time

Search to filter items by identifier

Status	Start time	Identifier	Duration	Static Results
✔ Succeeded	11/1/2023, 7:23 PM	08585027438577580600533201212CU23	16.99 Seconds	
✔ Succeeded	11/1/2023, 7:22 PM	08585027439374041434137248393CU17	18.18 Seconds	
❗ Failed	10/29/2023, 7:39 PM	08585030021267794679701149422CU18	19.46 Seconds	
❗ Failed	10/29/2023, 7:32 PM	08585030025288012479959852146CU11	26.79 Seconds	



Inbound example – batch import of database rows



Imported Tasks ▾

Edit columns

Edit filters

Filter by keyword

Due

All ▾

Created On ▾

Subject ▾

Description ▾

Category ▾

Due Date ↑ ▾

Duration ▾

Status Reason ▾

We didn't find anything to show here

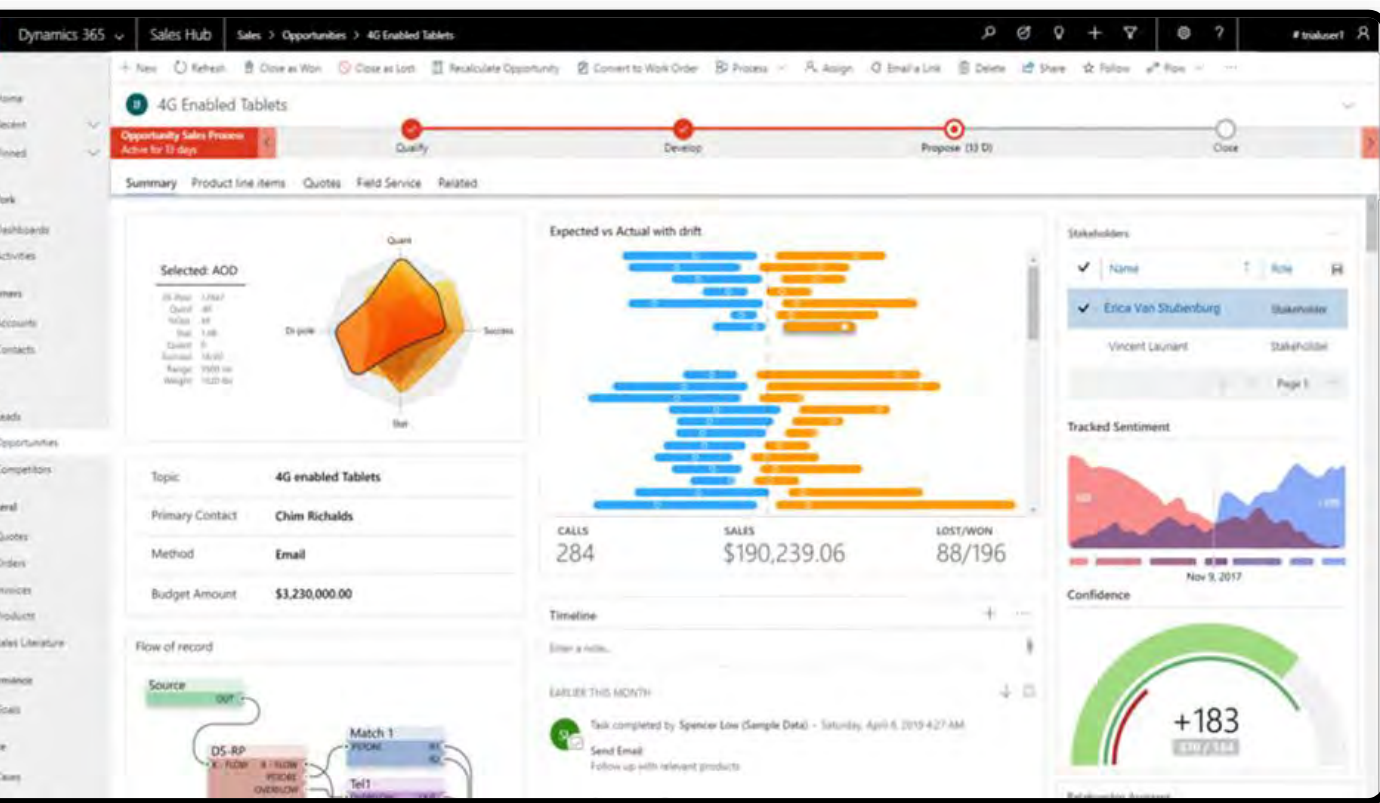
Rows: 0

Embed external data in model-driven apps

Example scenario



UI Integration - Build Custom Components with the Power Apps Component Framework



Developers can build compelling visual components for Power Apps

Custom components are responsive, reusable and support multiple form factors

Reuse your current IP and skills—framework is based on standard web technologies TypeScript/JS, CSS and HTML5

React and fabric support available in preview

Dataverse outbound integration

Endpoints and common tools

Dataverse outbound integration

Endpoints and common tools

Event driven via Dataverse event framework

Plug-ins – integrate with external web services and Azure Services.

Webhooks – HTTP pattern, posts execution context to web service.

Business Events – use for internal or external events to notify publishers.

Event driven & scheduled with Dataverse connectors

Power Automate – SaaS automation for low-code/no-code integration with 3rd party apps using 1000+ prebuilt connectors or custom connectors for cloud and on-premises connectivity.

Logic Apps – iPaaS automation for advanced integration to automate complex business processes. Tight integration with Azure. Benefit from the same connectors ecosystem as Power Automate.

Batch

Azure Synapse Link for Dataverse – continuously export Dataverse data for reporting and analytics use cases.

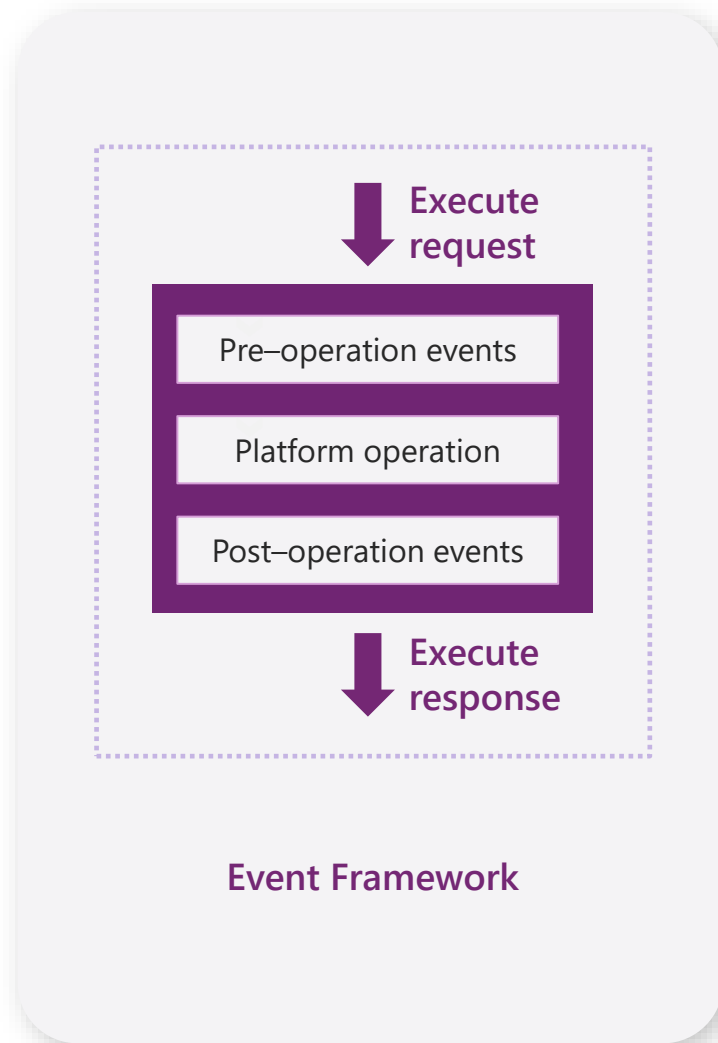
Change tracking - a set of delta changes is returned by Dataverse for a table.



Dataverse plugins for outbound integration

Dataverse plugins integration with external web services

Call external web services when responding to Dataverse events

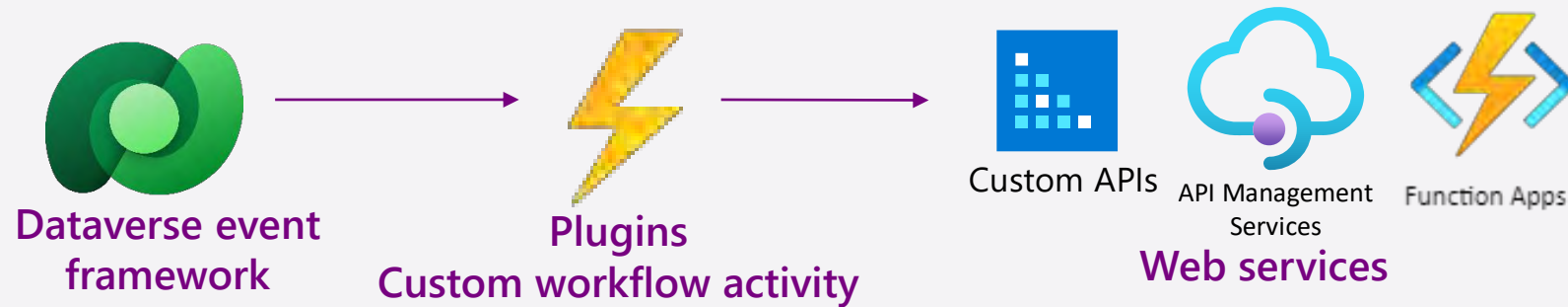


Event framework can trigger:

- Plug-ins
- Classic workflows and custom workflow activities
- Power Automate cloud flows
- Messages to Azure Service Bus
- Messages to Azure Event Hub
- Webhooks.

Dataverse plugins integration with external web services

Call external web services when responding to Dataverse events



Integrate data with External Web APIs

Dataverse plug-ins and custom workflow activities can access the network through the HTTP and HTTPS protocols. This capability provides support for accessing popular web services like social sites, news feeds, web services, and more.

Dataverse plugins integration with external web services

Considerations on accessing external web services from plugins



Prerequisites

- Must use a named web address that requires DNS name resolution.
- There is no provision for prompting the logged-on user for credentials or saving those credentials.
- Allow connections from Power Platform and Dynamics 365 services IP address values specified under the AzureCloud service tag.
- Only the HTTP and HTTPS protocols are allowed.



Pro-Code/Low-Code

- Pro-code using WebClient or HttpClient. Set KeepAlive to false, set Timeout when making external calls in a plug-in. Sample code [here](#).
- (Preview) Low-code plugins can use connectors e.g. SQL connector



Service Protection API limits

- No of requests sent by plug-ins or custom workflow activities not counted towards the limits.
- The additional computation time will be added to the request that triggered them, and it is enforced. E.g., data migration triggers plugins execution



Latency & Performance

- Recommended to use the Post-Operation stage and the asynchronous running mode.
- If you opt-in for sync plugins, consider the tight coupling and the 2-minute execution limit.



Security

- Do not store secrets as free text in the plug-in code.
- Use plugin secure config or custom parameters table hidden behind security roles.
- RetrieveEnvironmentVariableSecretValue custom API can only be called within Power Automate flow today.

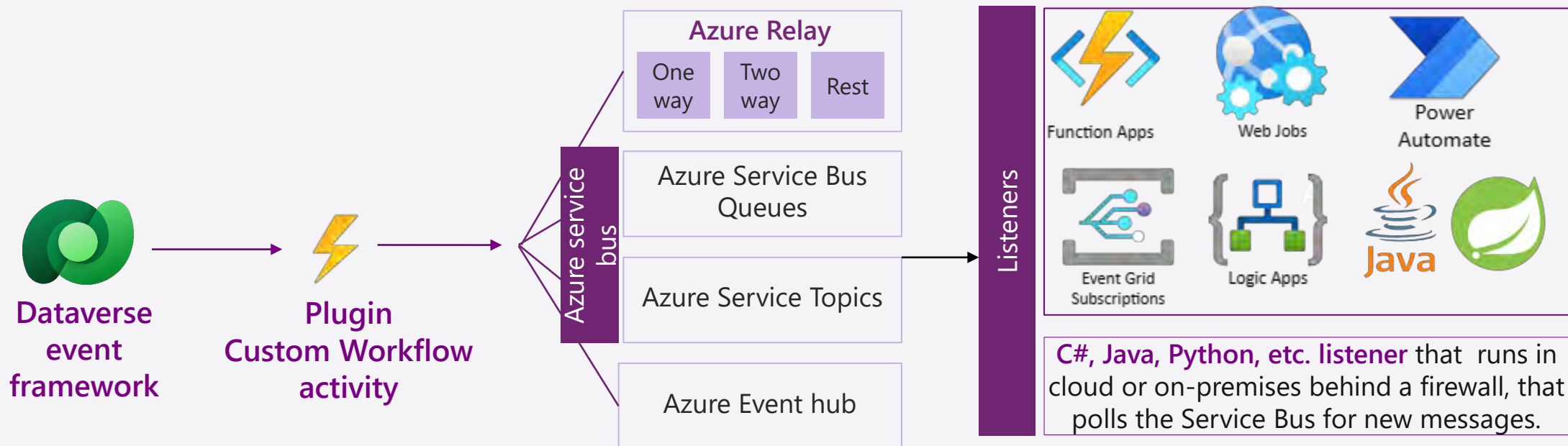


Recommendations

- The great power of plug-ins needs to be applied with some restraint and consideration for the impact it has on the system as a whole.
- For integrations also explore plugins with Azure Service Bus and Webhooks that can be used to push data to external systems where logic can be applied using many different resources.
- Power Automate provides many capabilities that previously were performed using plug-ins.

Dataverse plugins integration with Azure services

Pass runtime context to Azure Services



Integrate data with Azure service bus

Pass plugin execution context to one or more Azure solutions in the cloud. Use Azure Service Bus when the likelihood exists that the other system is not available or is limited in its ability to process high volumes of messages because the messages can be queued, allowing the receiving system to process the messages as fast as they can.

Dataverse integration with Azure Service Bus

Considerations for integrations with Azure Service Bus



Pro-Code/Low-Code

- Low-code option to create a step for the event and publish the plug-in context to the Azure Service Bus endpoint
- Plug-in or custom workflow activity code call the Azure Service Bus endpoint.



Security

- Shared Access Signatures (SAS) on Service Bus namespace, queue or topic. Must be updated after solution import.



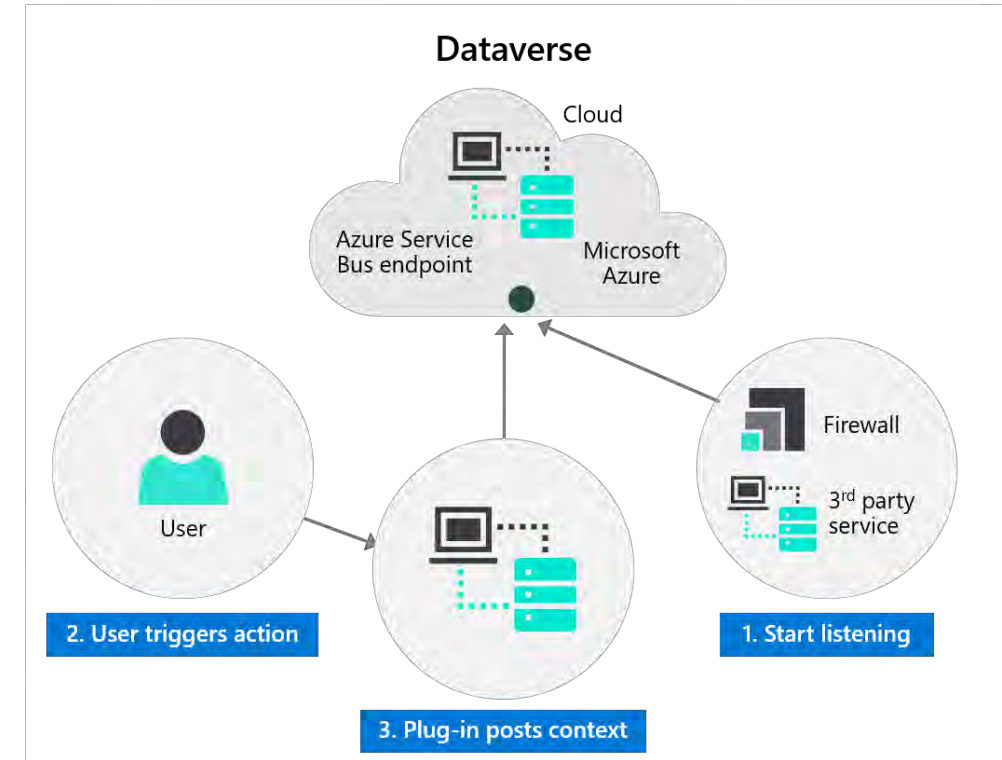
Message format

- Format XML, JSON, .NET binary. Payload max 192Kb.



Latency

- Asynchronous/near real time
- For async, each post to Azure Service Bus is performed by a system job of the Dataverse asynchronous service (System Jobs for status). If it fails, it executes plugin again in an exponential pattern. (more to less frequent, see [Retry Guidance Service Bus](#))



Service Bus Endpoint contract types:

- **Queue** endpoint, supports one listener that doesn't need to be actively listening. Inherent loose coupling between components.
- **Topic** endpoint, similar queue except it supports multiple listeners.

Low-code publish Dataverse plugin execution context to Azure Service Bus queue

Demo



Low-code publish Dataverse plugin execution context to Azure Service Bus queue and C# listener

Example scenario: on-premise system, limited in how many records it can process at a time, needs to be informed when opportunities are created in Dataverse. High number of opportunities are created in Dataverse daily.

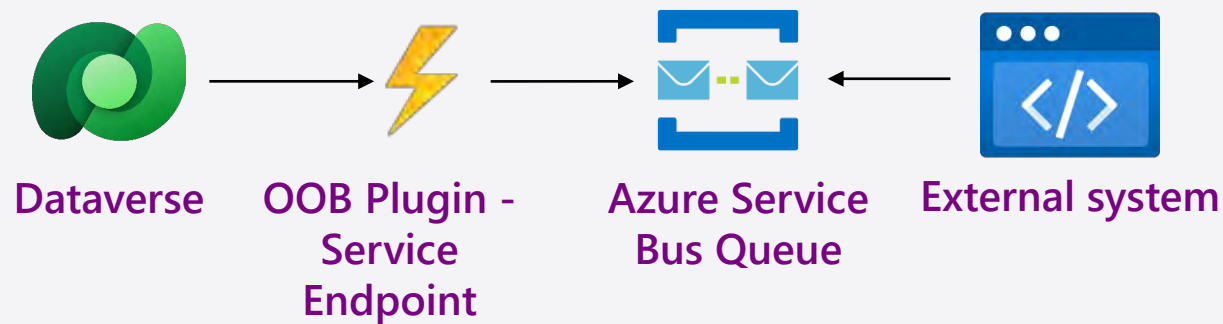
Step 1: Create Azure Service Bus Namespace.

Step 2: Create queue and configure SAS.

Step 3: Use SAS connection string to register queue endpoint. Register step for table event.

Step 4: Configure/write receiver. For this scenario, we will use a console application as receiver.

Step 5: Trigger table event and observe receiver.



Use Queues for one or multiple competing receivers

Step 1 – Create Service Bus namespace

The screenshot shows the Microsoft Azure portal interface. At the top is a blue header bar with the Microsoft Azure logo, a search bar, and several utility icons. Below the header, the 'Azure services' section displays a row of service tiles: 'Create a resource', 'Microsoft Entra ID', 'Users', 'Key vaults', 'Subscriptions', 'Resource Graph Explorer' (highlighted by a mouse cursor), 'Resource groups', 'Azure Synapse Analytics', 'Microsoft Sentinel', and 'More services'. Below this, the 'Resources' section features two tabs, 'Recent' and 'Favorite', with 'Recent' selected. It contains a table of recently viewed resources.

Name	Type	Last Viewed
[Resource Group]	Resource group	10 hours ago
[Logic App]	Logic app	a week ago
[Service Bus Namespace]	Service Bus Namespace	2 weeks ago
[Subscription]	Subscription	2 weeks ago
[Storage Account]	Storage account	2 months ago
[Data Factory (V2)]	Data factory (V2)	3 months ago
[Key Vault]	Key vault	4 months ago
[Storage Account]	Storage account	5 months ago
[Synapse Workspace]	Synapse workspace	8 months ago
[Log Analytics Workspace]	Log Analytics workspace	9 months ago
[Application Insights]	Application Insights	9 months ago
[Application Insights]	Application Insights	9 months ago

[See all](#)

Step 2 - Create Queue and configure SAS

The screenshot displays the Microsoft Azure portal interface. At the top, the navigation bar includes the Microsoft Azure logo, a search bar, and various utility icons. The main content area is titled "IntegrationDemoNamespace | Overview" and shows a deployment status of "Deployment". A sidebar on the left lists navigation options: Overview, Inputs, Outputs, and Template. The central panel features a green checkmark icon and the message "Your deployment is complete". Below this, deployment details are listed: Deployment name (IntegrationDemoNamespace), Subscription, and Resource group. The start time is noted as 10/28/2023, 10:29:24 AM. Action buttons for Delete, Cancel, Redeploy, Download, and Refresh are visible. A "Go to resource" button is prominently displayed. At the bottom left, there is a "Give feedback" link. The right sidebar contains three promotional sections: "Cost management" with a link to "Set up cost alerts", "Microsoft Defender for Cloud" with a link to "Go to Microsoft Defender for Cloud", and "Free Microsoft tutorials" with a link to "Start learning today". A final section, "Work with an expert", includes a link to "Find an Azure expert".

Microsoft Azure

Search resources, services, and docs (G+)

Home >

IntegrationDemoNamespace | Overview

Deployment

Search

Delete Cancel Redeploy Download Refresh

Overview

Inputs

Outputs

Template

✓ Your deployment is complete

Deployment name : IntegrationDemoNamespace Start time : 10/28/2023, 10:29:24 AM

Subscription :

Resource group :

> Deployment details

✓ Next steps

Go to resource

Give feedback

Tell us about your experience with deployment

Cost management

Get notified to stay within your budget and prevent unexpected charges on your bill.

Set up cost alerts >

Microsoft Defender for Cloud

Secure your apps and infrastructure

Go to Microsoft Defender for Cloud >

Free Microsoft tutorials

Start learning today >

Work with an expert

Azure experts are service provider partners who can help manage your assets on Azure and be your first line of support.

Find an Azure expert >

Step 3 - Create Dataverse service endpoint and register message processing step

The screenshot displays the Microsoft Azure portal interface. The left-hand navigation pane shows the 'Settings' section with 'Shared access policies' selected. The main content area shows the 'dataversedemoqueue' (IntegrationDemoNamespace/dataversedemoqueue) | Shared access policies view. A search bar and a list of policies are visible, with 'DataverseQueuePolicy' selected. The right-hand pane shows the configuration for 'SAS Policy: DataverseQueuePolicy'. The 'Send' checkbox is checked, and the 'Primary Connection String' field is highlighted with a red circle. The connection string is 'Endpoint=sb://integrationdemonamespace.servicebus.windows.net;/SharedAccessKeyN...'. Other fields include 'Primary Key', 'Secondary Key', 'Secondary Connection String', and 'SAS Policy ARM ID'.

Microsoft Azure

Search resources, services, and docs (G+/I)

Home > IntegrationDemoNamespace | Overview > IntegrationDemoNamespace | Queues > dataversedemoqueue (IntegrationDemoNamespace) | Shared access policies

dataversedemoqueue (IntegrationDemoNamespace/dataversedemoqueue) | Shared access policies

Service Bus Queue

Search

+ Add

Search to filter items...

Policy

DataverseQueuePolicy

Claim

Send

Settings

Shared access policies

Properties

Locks

Automation

CLI / PS

Tasks (preview)

Export template

Help

Support + Troubleshooting

SAS Policy: DataverseQueuePolicy

Save Discard Delete Regenerate Primary Key ...

☐ Manage

☒ Send

☐ Listen

Primary Key

uSrUoV...

Secondary Key

SkqLGC//fS...

Primary Connection String

Endpoint=sb://integrationdemonamespace.servicebus.windows.net;/SharedAccessKeyN...

Secondary Connection String

Endpoint=sb://integrationdem...

SAS Policy ARM ID

/subscriptions/131...

Step 4 – Configure receiver

```
1 using System.Runtime.Serialization.Json;
2 using System.Text;
3 using System.Threading.Tasks;
4 using Azure.Identity;
5 using Azure.Messaging.ServiceBus;
6 using Microsoft.Xrm.Sdk;
7 using Newtonsoft.Json;
8 using System.ServiceModel;
9
10
11 // the client that owns the connection and can be used to create senders and receivers
12 ServiceBusClient client;
13
14 // the processor that reads and processes messages from the queue
15 ServiceBusProcessor processor;
16
17 // The Service Bus client types are safe to cache and use as a singleton for the lifetime
18 // of the application, which is best practice when messages are being published or read
19 // regularly.
20 //
21 // Set the transport type to AmqpWebSockets so that the ServiceBusClient uses port 443,
22 // If you use the default AmqpTcp, make sure that ports 5671 and 5672 are open.
23
24 // TODO: Replace the <NAMESPACE-NAME> and <QUEUE-NAME> placeholders
25 var clientOptions = new ServiceBusClientOptions()
26 {
27     TransportType = ServiceBusTransportType.AmqpWebSockets
28 };
29
30 client = new ServiceBusClient("IntegrationDemoNamespace.servicebus.windows.net",
31     new DefaultAzureCredential(), clientOptions);
32
33 ServiceBusProcessorOptions processorOptions = new ServiceBusProcessorOptions();
34 processorOptions.ReceiveMode = ServiceBusReceiveMode.PeekLock;
35 //processorOptions.ReceiveMode = ServiceBusReceiveMode.ReceiveAndDelete;
36
37 // create a processor that we can use to process the messages
38 // TODO: Replace the <QUEUE-NAME> placeholder
39 processor = client.CreateProcessor("dataversedemoqueue", new ServiceBusProcessorOptions());
40
41 try
42 {
43     // add handler to process messages
44     processor.ProcessMessageAsync += MessageHandler;
45
46     // add handler to process any errors
47     processor.ProcessErrorAsync += ErrorHandler;
48
49     Console.WriteLine("Press any key to start the processing");
50     Console.ReadKey();
51 }
52
53 }
```

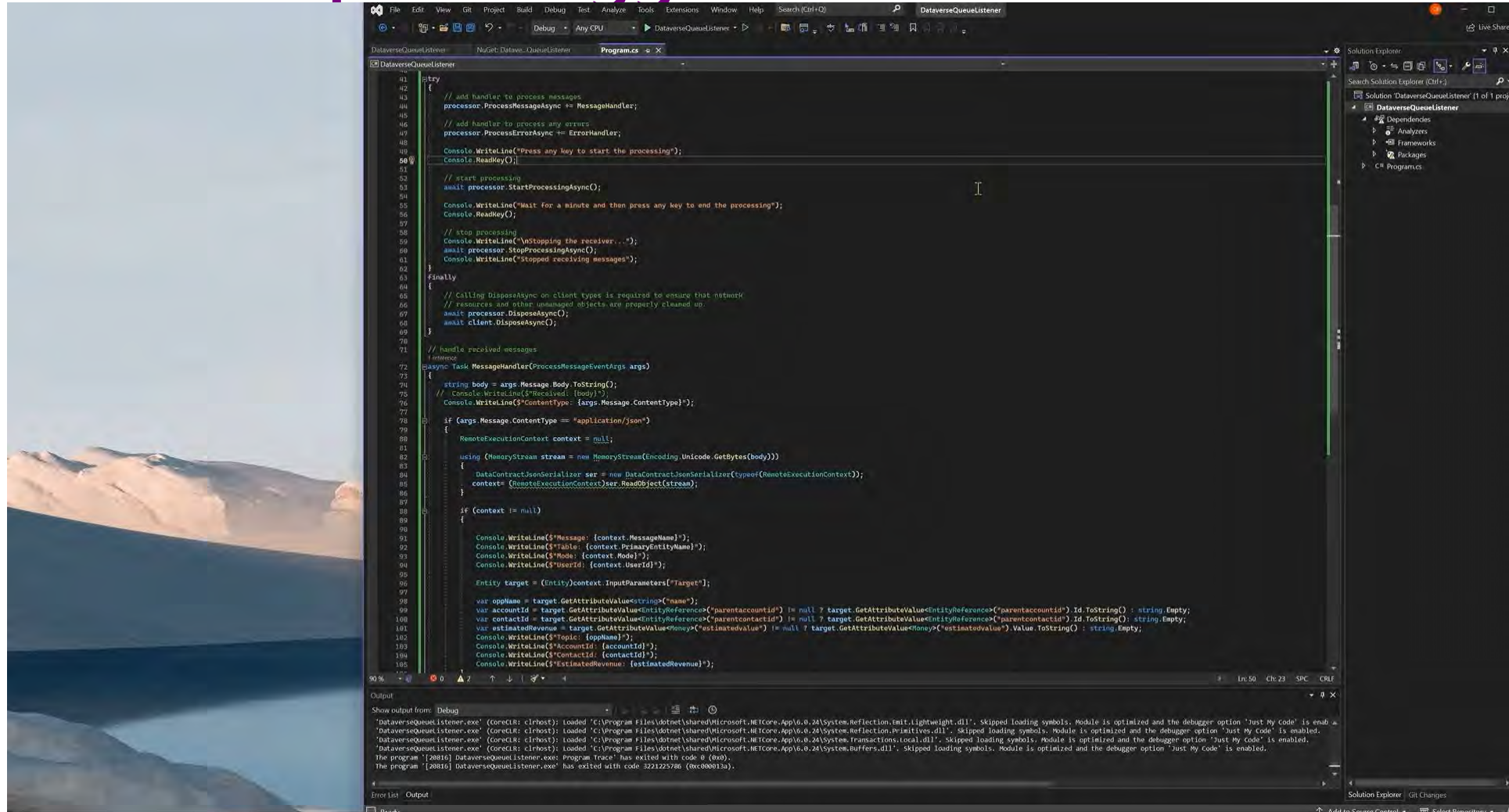
```
using System.Runtime.Serialization.Json;
using System.Text;
using System.Threading.Tasks;
using Azure.Identity;
using Azure.Messaging.ServiceBus;
using Microsoft.Xrm.Sdk;
```

```
// handle received messages
1 reference
2 async Task MessageHandler(ProcessMessageEventArgs args)
3 {
4     string body = args.Message.Body.ToString();
5     // Console.WriteLine($"Received: {body}");
6     Console.WriteLine($"ContentType: {args.Message.ContentType}");
7
8     if (args.Message.ContentType == "application/json")
9     {
10         RemoteExecutionContext context = null;
11
12         using (MemoryStream stream = new MemoryStream(Encoding.Unicode.GetBytes(body)))
13         {
14             DataContractJsonSerializer ser = new DataContractJsonSerializer(typeof(RemoteExecutionContext));
15             context = (RemoteExecutionContext)ser.ReadObject(stream);
16         }
17     }
18 }
```

- ✓ Create a receiver – in this example I am creating a console application. Follow steps documented here [Quickstart - Use Azure Service Bus queues from .NET app](#)

- ✓ Add code to the receiver to parse Dataverse message - see more sample code documented here [Write a listener application for a Microsoft Azure solution \(Microsoft Dataverse\)](#)

Step 5 – Trigger Dataverse event



```
11 try
12 {
13     // add handler to process messages
14     processor.ProcessMessageAsync += MessageHandler;
15
16     // add handler to process any errors
17     processor.ProcessErrorAsync += ErrorHandler;
18
19     Console.WriteLine("Press any key to start the processing");
20     Console.ReadKey();
21
22     // start processing
23     await processor.StartProcessingAsync();
24
25     Console.WriteLine("Wait for a minute and then press any key to end the processing");
26     Console.ReadKey();
27
28     // stop processing
29     Console.WriteLine("Stopping the receiver...");
30     await processor.StopProcessingAsync();
31     Console.WriteLine("Stopped receiving messages");
32 }
33 finally
34 {
35     // Calling DisposeAsync on client types is required to ensure that network
36     // resources and other unmanaged objects are properly cleaned up.
37     await processor.DisposeAsync();
38     await client.DisposeAsync();
39 }
40
41 // handle received messages
42 async Task MessageHandler(ProcessMessageEventArgs args)
43 {
44     string body = args.Message.Body.ToString();
45     Console.WriteLine($"Received: {body}");
46     Console.WriteLine($"ContentType: {args.Message.ContentType}");
47
48     if (args.Message.ContentType == "application/json")
49     {
50         RemoteExecutionContext context = null;
51
52         using (MemoryStream stream = new MemoryStream(Encoding.Unicode.GetBytes(body)))
53         {
54             DataContractJsonSerializer ser = new DataContractJsonSerializer(typeof(RemoteExecutionContext));
55             context = (RemoteExecutionContext)ser.ReadObject(stream);
56         }
57
58         if (context != null)
59         {
60             Console.WriteLine($"Message: {context.MessageName}");
61             Console.WriteLine($"Table: {context.PrimaryEntityName}");
62             Console.WriteLine($"Mode: {context.Mode}");
63             Console.WriteLine($"UserId: {context.UserId}");
64
65             Entity target = (Entity)context.InputParameters["Target"];
66
67             var oppName = target.GetAttributeValue<string>("name");
68             var accountId = target.GetAttributeValue<EntityReference>("parentaccountid") != null ? target.GetAttributeValue<EntityReference>("parentaccountid").Id.ToString() : string.Empty;
69             var contactId = target.GetAttributeValue<EntityReference>("parentcontactid") != null ? target.GetAttributeValue<EntityReference>("parentcontactid").Id.ToString() : string.Empty;
70             var estimatedRevenue = target.GetAttributeValue<Money>("estimatedvalue") != null ? target.GetAttributeValue<Money>("estimatedvalue").Value.ToString() : string.Empty;
71
72             Console.WriteLine($"Topic: {oppName}");
73             Console.WriteLine($"Accountid: {accountId}");
74             Console.WriteLine($"Contactid: {contactId}");
75             Console.WriteLine($"EstimatedRevenue: {estimatedRevenue}");
76         }
77     }
78 }
```

Output

Show output from: Debug

"DataverseQueueListener.exe" (CoreCLR: clrhost): Loaded 'C:\Program Files\dotnet\shared\Microsoft.NETCore.App\6.0.24\System.Reflection.Emit.Lightweight.dll'. Skipped loading symbols. Module is optimized and the debugger option 'Just My Code' is enabled.

"DataverseQueueListener.exe" (CoreCLR: clrhost): Loaded 'C:\Program Files\dotnet\shared\Microsoft.NETCore.App\6.0.24\System.Reflection.Primitives.dll'. Skipped loading symbols. Module is optimized and the debugger option 'Just My Code' is enabled.

"DataverseQueueListener.exe" (CoreCLR: clrhost): Loaded 'C:\Program Files\dotnet\shared\Microsoft.NETCore.App\6.0.24\System.Transactions.Local.dll'. Skipped loading symbols. Module is optimized and the debugger option 'Just My Code' is enabled.

"DataverseQueueListener.exe" (CoreCLR: clrhost): Loaded 'C:\Program Files\dotnet\shared\Microsoft.NETCore.App\6.0.24\System.Buffers.dll'. Skipped loading symbols. Module is optimized and the debugger option 'Just My Code' is enabled.

The program '[20816] DataverseQueueListener.exe' has exited with code 0 (0x).

The program '[20816] DataverseQueueListener.exe' has exited with code 3221225786 (0xc000013a).



**Dataverse
webhooks**

Dataverse integration – Webhooks to external web services

Volumes

- Webhooks can only scale to the point at which your hosted web service can handle the messages.

Latency

- Webhooks enables synchronous and asynchronous steps

Message format:

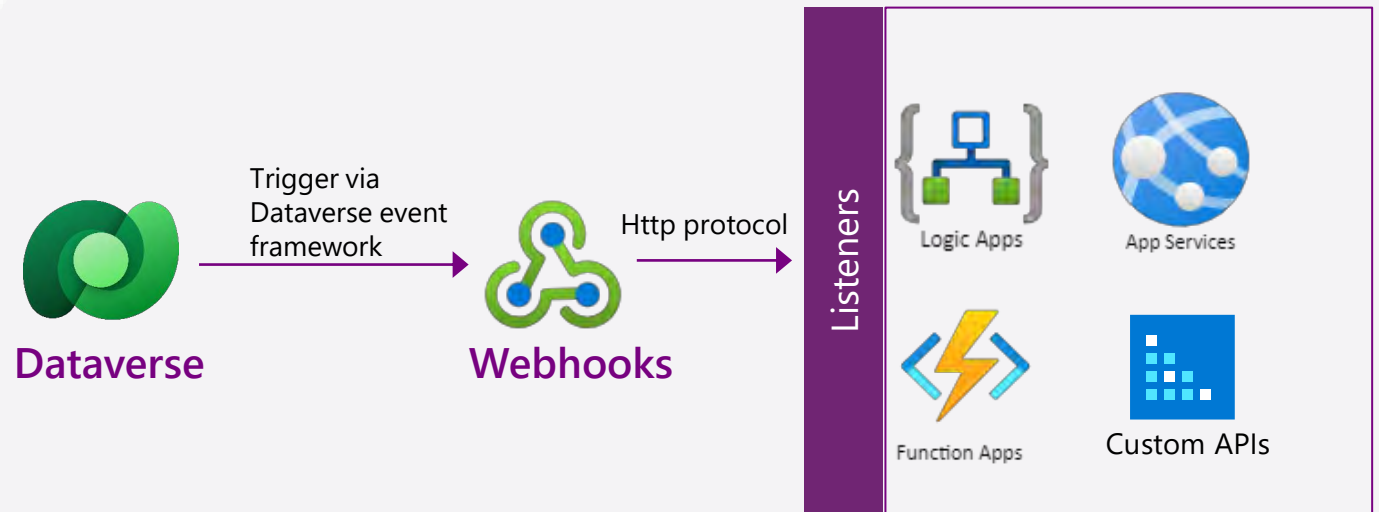
- Webhooks send POST requests with JSON payload and can be consumed by any programming language or web application hosted anywhere.

Security

- The endpoint can be secured by using an authentication header or query string parameter keys.

Low-Code/Pro-Code

- (low-code) can be registered to run on table events (low-code) or (pro-code) can be invoked from a plug-in or custom workflow activity. 60 seconds timeout.



Integrate data with webhooks

Pass plugin execution context to web APIs and services using a lightweight HTTP pattern. Consume the JSON payload with any programming language or web application hosted anywhere.

Consume webhook with Azure function

Tutorial



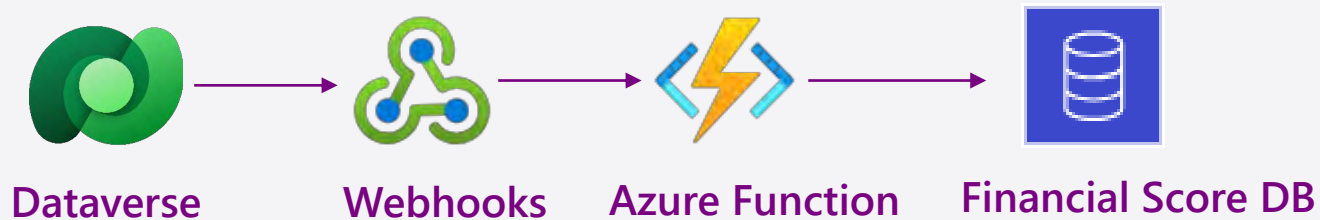
Consume webhook with Azure function

Example scenario: When contact customer is created, we need to calculate a financial score and store it in an external database. The calculation is a complex algorithm with input parameters Dataverse contact information and data from external systems. The score and contact details must be saved in a separate db.

Step 1: Create Azure function. Add code to parse Dataverse execution context and execute custom logic. Publish function.

Step 2: Register Dataverse webhook with Azure function url. Register relevant message step e.g. in our scenario create contact.

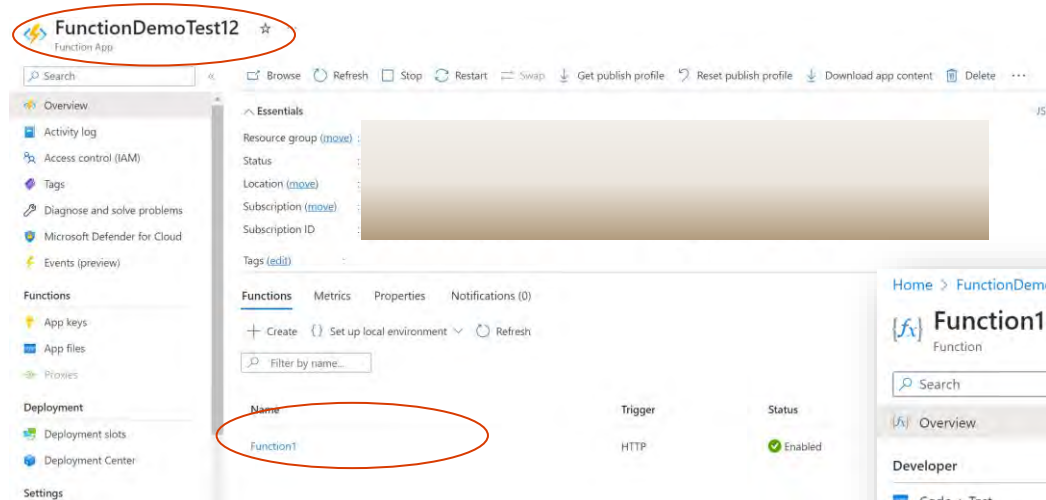
Step 3: Trigger event and create contact. Observe that Azure function is triggered.



We will use an Azure function for the financial score risk rating. It receives the Dataverse runtime context via webhook, executes the calculation logic, then saves the score in an external database.

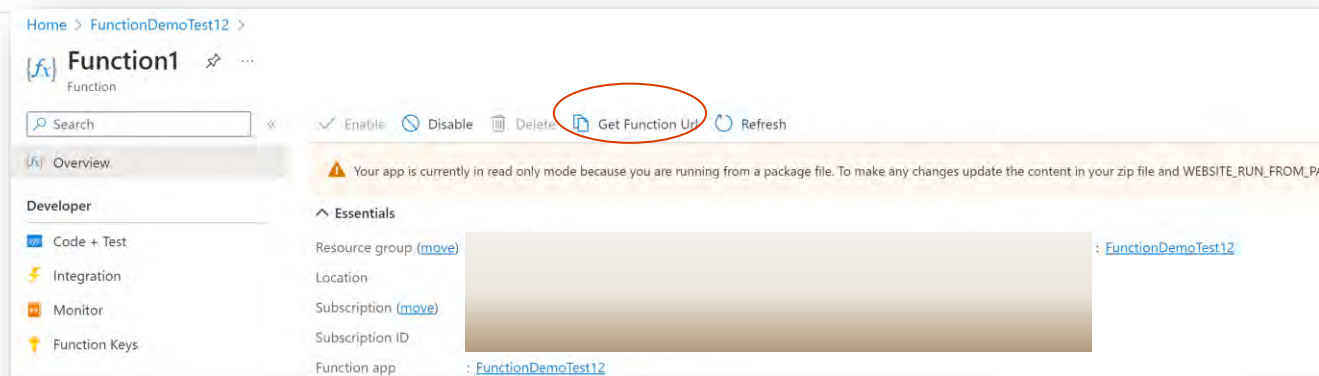
Step 1 - Create Azure Function

- ✓ Create Azure Function, with http trigger. See step by step and sample code [Write an Azure Function that processes Dataverse events - Training | Microsoft Learn](#)
- ✓ Write code to parse execution context, score calculation logic and score saving method.
- ✓ Publish Azure function and save function url.

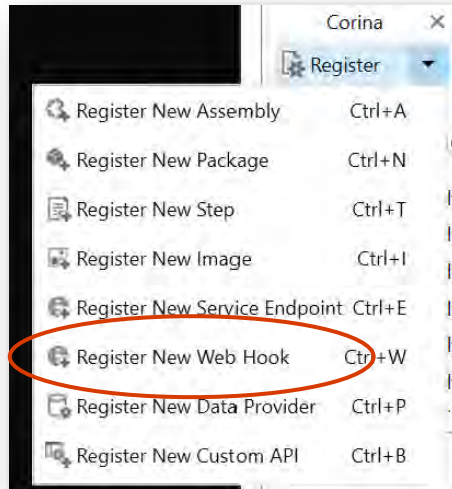


```
21 public static async Task<ActionResult> Run(  
22     [HttpTrigger(AuthorizationLevel.Function, "get", "post", Route = null)] HttpRequest req,  
23     ILogger log)  
24     {  
25         log.LogInformation("C# HTTP trigger function processed a request.");  
26  
27         string queryParams = "";  
28         foreach (var q in req.Query)  
29         {  
30             queryParams += $"Key: {q.Key} Value: {q.Value}\n";  
31         }  
32  
33         string requestBody = await new StreamReader(req.Body).ReadToEndAsync();  
34         dynamic data = JsonConvert.DeserializeObject(requestBody);  
35         string requestHeader = "";  
36         foreach (var h in req.Headers)  
37         {  
38             requestHeader += $"Key: {h.Key} Value: {h.Value}\n";  
39         }  
40  
41         log.LogInformation("Query Parameters:\n" + queryParams);  
42         log.LogInformation("Request Header: \n" + requestHeader);  
43         log.LogInformation("Request Body:\n" + requestBody);  
44         string requestBodyFormatted = JValue.Parse(requestBody).ToString(Formatting.Indented);  
45         log.LogInformation("Request Body Formatted:\n" + requestBodyFormatted);  
46  
47         try  
48         {  
49             dynamic target = ((JArray)data.InputParameters).FirstOrDefault(f => f["key"].Value<string>() == "Target")["value"];  
50             log.LogInformation("Target: \n" + JsonConvert.SerializeObject((object)target).ToString());  
51  
52             foreach (dynamic field in target.Attributes)  
53             {  
54                 log.LogInformation($"Name: {field.key} Value: {field.value.Value}");  
55             }  
56         }  
57         catch (Exception ex)  
58         {  
59             log.LogInformation(ex.ToString());  
60         }  
61  
62         return (ActionResult)new OkObjectResult(data.InitiatingUserId);  
63     }
```

Example Azure function parses execution context sent via webhook



Step 2 – Register Webhook using Azure function url



WebHook Registration

Name *	Function Demo Test 1
Endpoint URL *	https://functionden
Authentication	WebhookKey
Value	

Update Existing Step

General Configuration Information

Message *	Create
Primary Entity	contact
Secondary Entity	none
Filtering Attributes	Message does not support Filtered Attributes
Event Handler	(WebHook) Function Demo Test 1
Step Name *	Function Demo Test 1: Create of contact
Run in User's Context	Calling User
Execution Order *	1
Description	Function Demo Test 1: Create of contact

Event Pipeline Stage of Execution	Execution Mode	Deployment *
PostOperation	<input checked="" type="radio"/> Asynchronous <input type="radio"/> Synchronous	<input checked="" type="checkbox"/> Server <input type="checkbox"/> Offline

- ✓ Register webhook using function url (remove the code= portion) and paste value of code as webhookkey.
- ✓ For the webhook just created, add a message step for when it should be triggered. e.g., contact creation.


Step 3 – Trigger Dataverse event

The screenshot displays the Microsoft Dynamics 365 interface for a user named Corina Sales Hub. The left-hand navigation pane includes sections for Customers (Dashboards, Page2Containers, Accounts, Contacts), Sales (Leads, Opportunities, Competitors), Sales Analytics (Sales usage reports, Sales Acceleration...), and Collateral (Quotes, Orders, Invoices, Products, Sales Literature). The 'Contacts' section is currently selected. The main area shows a table titled 'My Active Contacts' with columns for Full Name, Email, Company Name, and Business Phone. The table contains 17 rows of contact data. The top row is for Alex Baker, and the bottom row is for cc17 bb17. The interface also includes a search bar at the top, a 'Quick find' box, and a 'Rows: 49' indicator at the bottom left.

Full Name ↑	Email	Company Name *	Business Phone
Alex Baker	corina.balan2007@gmail.com	Cogiris Ltd	619-555-0127
Altin Tuin			
Andreea Balan			
Anne Weiler	annew@CRM874988.OnMicroso...	Fast Machines	879-789-7777
Avery Howard	avery@treyresearch.net	Trey Research	567-555-0137
Carla Yates	carla@treyresearch.net	Trey Research	456-555-0156
cc bb		A. Datum Corporation test2	
cc1 bb1		A. Datum Corporation test2	
CC10 bb10		A. Datum Corporation test2	
cc11 bb12		CC10 bb10	
cc13 bb13		Account	
cc14 bb14		Fast Machines	
cc15 bb15		A. Datum Corporation test2	
cc16 bb16		A. Datum Corporation test2	
cc17 bb17		A. Datum Corporation test2	

Step 3 – Monitor function execution

[Home](#) > [FunctionDemoTest12](#) > [Function1](#)

**Function1 | Monitor** ...
Function

Overview

Developer

- Code + Test
- Integration
- Monitor**
- Function Keys

Invocations Logs

Success Count
✓ 18
Last 30 Days

Error Count
✗ 0
Last 30 Days

Invocation Traces
The twenty most recent function invocation traces. For more advanced analysis, run the query in Application Insights.
[Run query in Application Insights](#) [Refresh](#)

Date (UTC)	Success	Result Code	Duration (ms)	Operation Id
2023-10-29 12:14:34.295	✓ Success	200	297	
2023-10-29 12:04:16.892	✓ Success	200	331	
2023-10-29 11:53:45.358	✓ Success	200	329	
2023-10-29 11:47:58.480	✓ Success	200	358	
2023-10-29 11:37:18.551	✓ Success	200	260	
2023-10-29 11:29:28.915	✓ Success	200	340	
2023-10-29 08:38:05.361	✓ Success	200	114	
2023-10-29 08:01:19.292	✓ Success	200	469	
2023-10-29 07:36:05.871	✓ Success	200	102	
2023-10-29 00:21:25.453	✓ Success	200	399	
2023-10-29 00:13:32.456	✓ Success	200	104	



Dataverse Business Events

Dataverse Process Integration with Business Events

How to design and expose business events in Dataverse

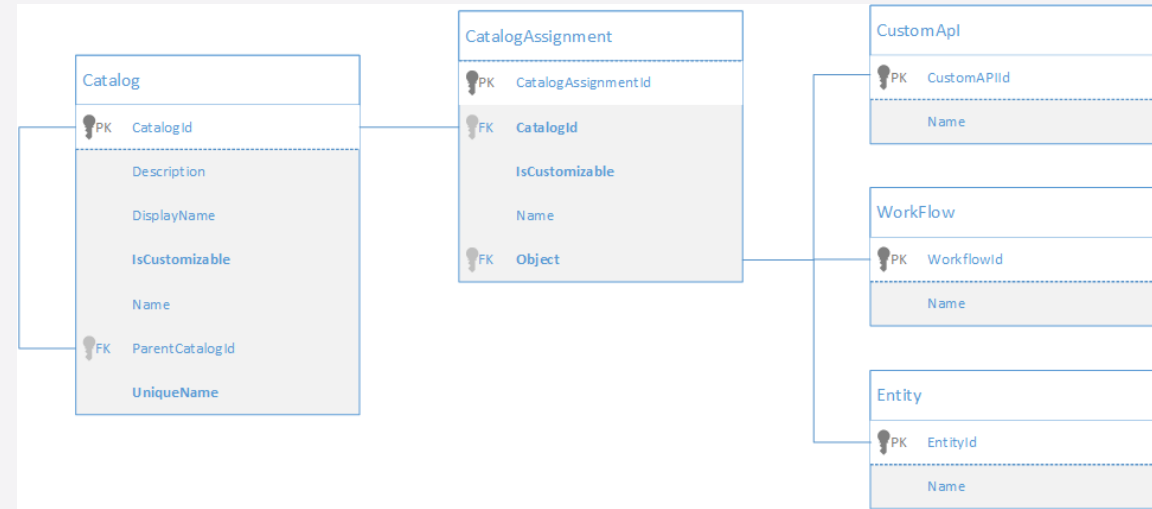
Design principles

Clear intent: The intent behind creating a business event must be clearly understood.

Specific: The event should be specific so that a subscriber doesn't need to filter whether or not they should respond to it.

Lightweight: The event should contain only that data necessary to describe the event. If the subscriber requires additional data, the information in the event should provide the context to allow them to retrieve it if necessary.

Not for transferring data: If your intent is to transfer data to a recipient and, in effect, realize a data export scenario, you don't have a good use case for business events.



Catalog business events

Process integration with Business Events

Dataverse Business events provides new ways to expose events and compose your business logic to respond to them asynchronously. e.g. using Power Automate cloud flow with When an action is performed trigger.

Dataverse Process Integration with Business Events

Considerations and scenarios examples

Triggers

- Table events (including virtual tables) and custom APIs to emit events.
- They can be triggered by operations via Dataverse Event Framework and also by external systems by calling custom API endpoint registered as business event.

Latency

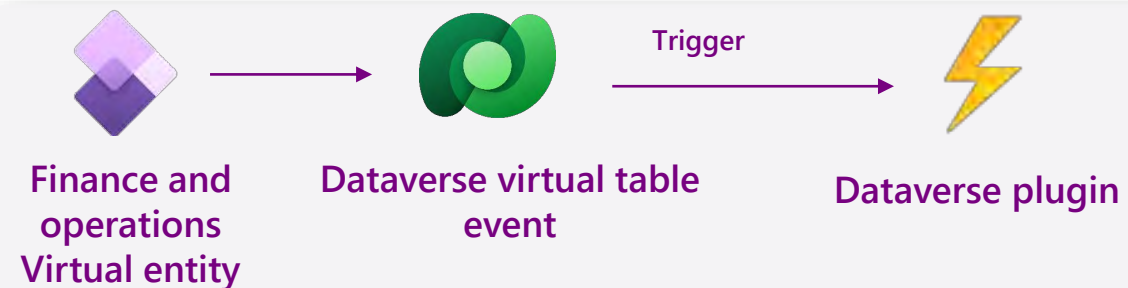
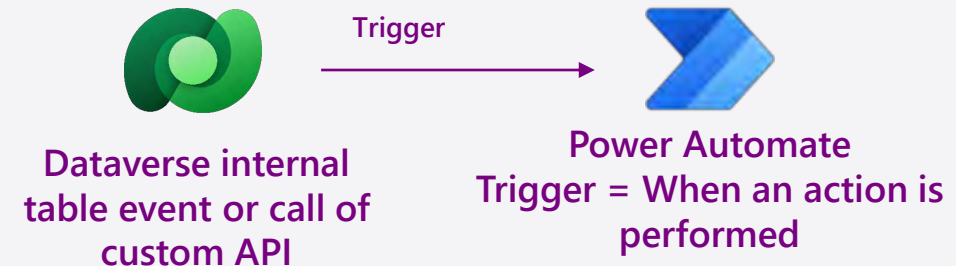
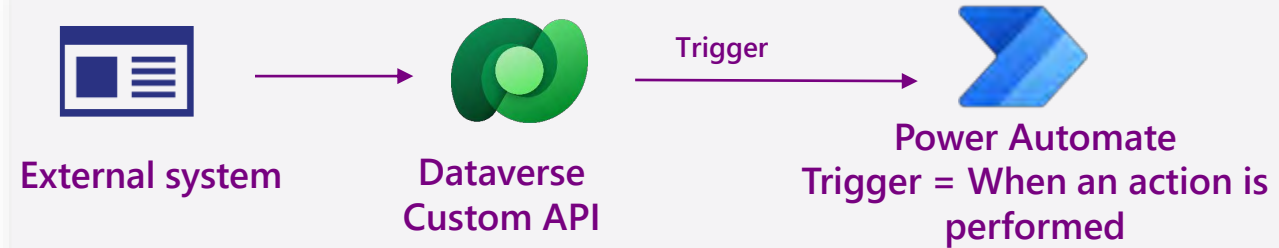
- Asynchronous, event is emitted after the operation completes successfully.

Pro-Code/Low-Code

- Empowers citizen developers to define automation logic in Power Automate to respond to specific events.
- For pro-dev, it can help simplify plugin logic e.g., call a custom API with input params from plugin vs using shared variables in plugins; or use a flow rather than a plugin for a table event such as GrantAccess message.

Security

- Asynchronous, event is emitted after the operation completes successfully.



These are some examples of business events scenarios.

Azure Synapse Link for Dataverse



Azure Synapse Link for Dataverse

Setup

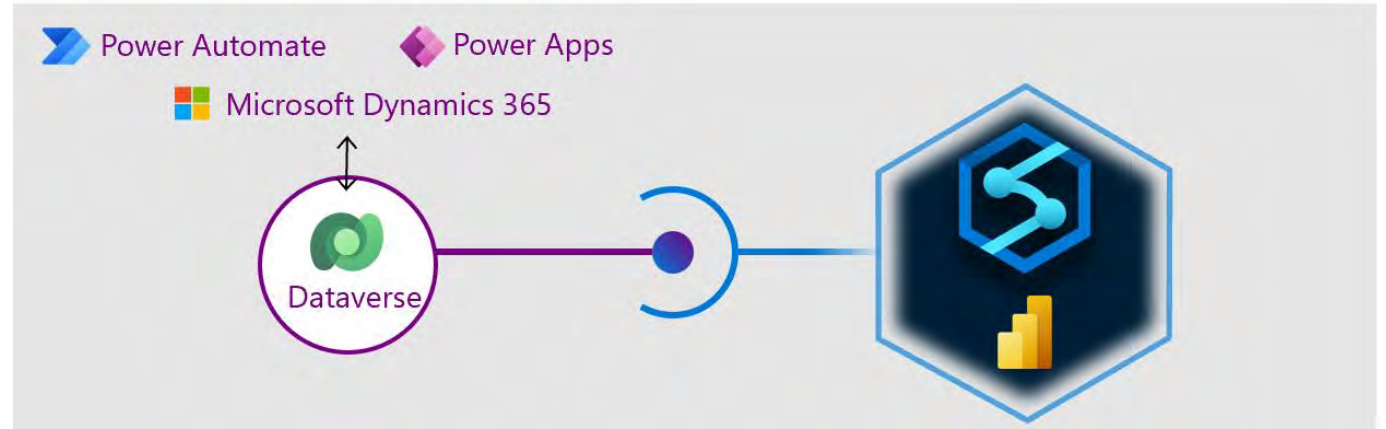
- Integrated with Power Apps maker portal. Ready to run Synapse Serverless SQL query experience, Synapse Apache Spark experience, Synapse Pipelines experience.
- Solution aware which enables seamless ALM, cost effective

Latency

- Near real-time data replication, PaaS with BYOL soft SLA of 15 mins.
- Built for eventual data consistencies.
- Append mode is recommended for better performance.

Security

- You can limit access to your storage account to requests from specified IP addresses, IP ranges, subnets in an Azure Virtual Network (VNet), or resource instances of some Azure services.



Using Azure Synapse Link, continuously export data from Dataverse to Azure Data Lake Storage Gen2 and Azure Synapse Analytics.

Options to read Dataverse data from Azure Synapse :

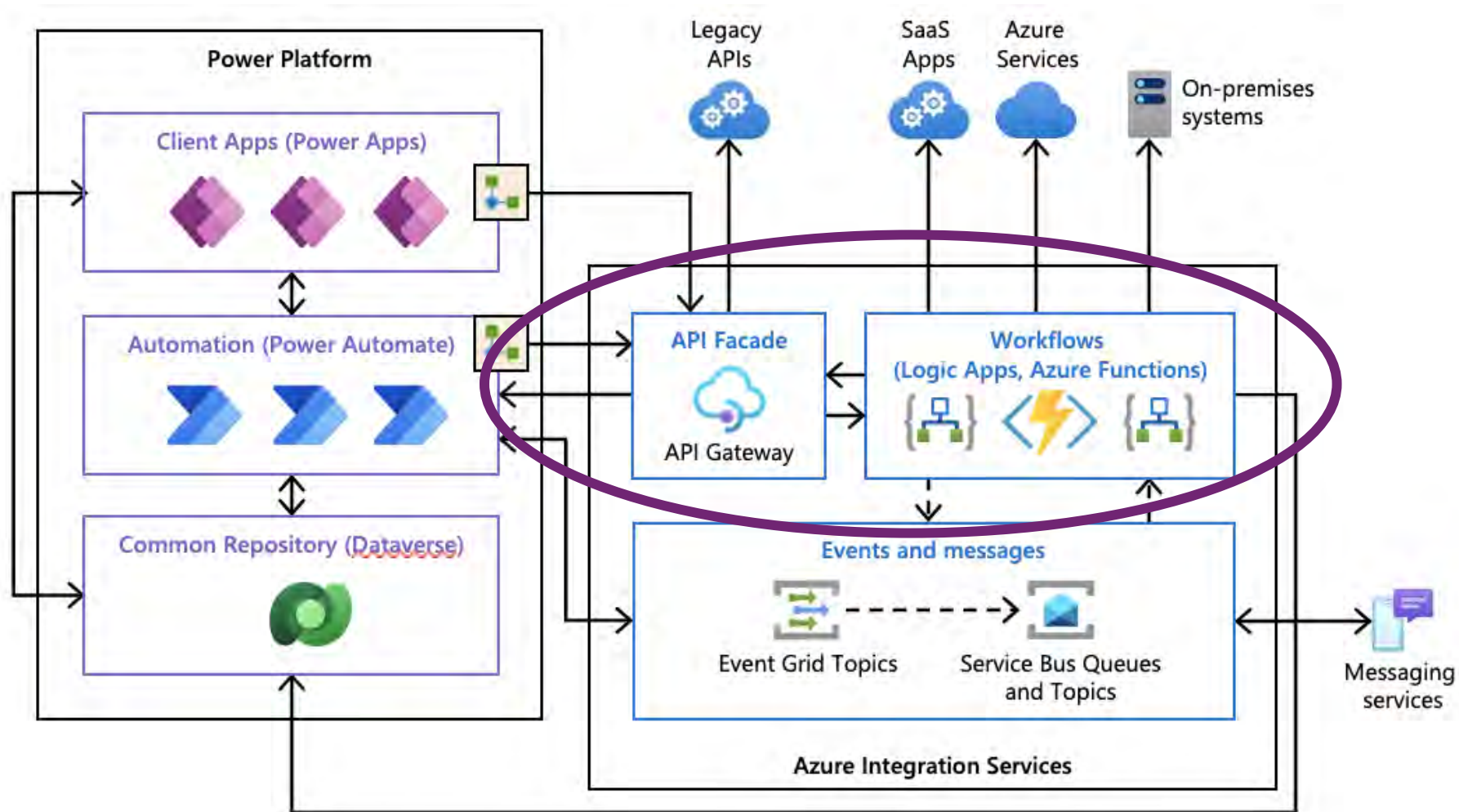
- [Query Azure Synapse Link for Dataverse data with serverless SQL](#)
- [Transform Azure Synapse Link for Dataverse data with Apache Spark](#)
- [Visualize Azure Synapse Link for Dataverse data with Power BI](#)
- [Copy exported Dataverse data to dedicated SQL pool](#)
- [For ETL, Azure Synapse Pipelines](#)

!Not suitable for real time or very near-time latency integrations!

Conclusion



Enterprise solutions can be a little of everything...



Integration services are a gateway to Azure!

Resources



Dataverse Integration Resources

Learning paths

- ✓ [*Integrate with Microsoft Power Platform and Microsoft Dataverse - Training | Microsoft Learn*](#)
- ✓ [*Integrate Microsoft Dataverse Azure solutions - Training | Microsoft Learn*](#)
- ✓ [*Solution Architect series: Implement integrations with Power Platform - Training | Microsoft Learn*](#)

Technical Talks

- ✓ [*Dynamics 365 Integration General Guidance*](#)
- ✓ [*Business Events Overview*](#)
- ✓ [*Export to Azure Data Lake Overview*](#)
- ✓ [*Export to Azure Data Lake Scenarios*](#)

Azure integration services

Dataverse

- ✓ [*OAuth authentication*](#)
- ✓ [*Dataverse Web API*](#)
- ✓ [*Azure integration \(Microsoft Dataverse\)*](#)
- ✓ [*Write a custom Azure-aware plug-in*](#)
- ✓ [*Webhooks*](#)
- ✓ [*Write an Azure Function that processes Dataverse events*](#)
- ✓ [*Subscribe to events in Dataverse - Finance & Operations | Dynamics 365*](#)

- ✓ [*Integration and automation platform options in Azure | Microsoft Learn*](#)
- ✓ [*Compare Azure messaging services - Azure Service Bus | Microsoft Learn*](#)
- ✓ [*Azure Integration Services | Microsoft Azure*](#)
- ✓ [*Quickstart - Use Azure Service Bus queues from .NET app - Azure Service Bus | Microsoft Learn*](#)

Webhooks vs Azure Service Bus

Consider using **Webhooks** when:

- A third-party Web API endpoint already exists that you want to use for integration purposes.
- The external operation that you're performing needs to occur immediately.
- You want the entire transaction to fail unless the external service successfully processes the webhook payload.
- Authentication through authentication headers and query string parameter keys is preferred.
- Message format is only JSON.

Consider using **Azure Service Bus** when:

- High-scale asynchronous processing/queueing is a requirement.
- Multiple subscribers might need to consume a given Dataverse event.
- You want to govern your integration architecture in a centralized location.
- Decouple applications and protect them from temporary peaks.
- Shared Access Signature (SAS) authentication is preferred and feasible.
- Message format can be .NET Binary, JSON or XML.

Both Webhooks and Azure Service Bus can be invoked from a plug-in or custom workflow activity.

Azure Function vs Plugins

Consider using **Azure Functions** :

- for complex or compute intensive that run in Azure, to reduce load on the Dataverse's application host.
- For tight integration with Azure services.
- In combination with webhooks or wrapped as custom connector.
- Execution might take longer than 2 minutes.
- Combine with Webhooks, Power Automate or Logic Apps.

Consider using **Plugins** :

- If you need to update data in the most high-performing manner.
- For tight integration with Dataverse and full rollback support.
- Execution will complete within 2 minutes.
- Combine with Power Automate, Logic Apps, external web services, functions.

You can invoke Azure functions from plug-ins or combine plug-ins with Azure Service Bus then function.

Power Automate vs Logic Apps

Power Automate includes the following features :

- It's packaged as part of a solution
- Performs RPA with desktop flows
- Uses the Approvals connector
- Includes a Send Notifications connector
- Has a limit on the number of flow runs each month
- Trigger types: automated, instant, scheduled UI flow, business process

Logic Apps includes the following features:

- Performs Enterprise Integration including EDI
- Has higher performance
- Can be more easily monitored by using Azure tools
- Has better error handling
- Can't be packaged in solutions
- Has a consumption-based or fixed pricing model through an Azure subscription
- Trigger types: Http, WebHook, Scheduled, Http manual call



QUESTIONS

Q&A

Are the .Net SDK & OData calls considered equally performant?



The Dataverse Web API provides a RESTful programming experience but ultimately all data operations go through the underlying organization service. See more:

[Use the Microsoft Dataverse Web API \(Dataverse\) - Power Apps | Microsoft Learn](#)

Are there any integration patterns available for API-first strategies which will limit throttling of Dataverse APIs?



There are several strategies you can use such as: using multiple app users (as the throttling limits are per user per web server), using multiple threads if the operations run quickly, try the latest xmultiple, remove affinity cookie, etc. See more here:

[How to maximize throughput - Power Apps | Microsoft Learn](#)

Will Power Automate run in user context or service account? How to make a flow run using a service account? What license considerations should we consider?



It is possible to use service principal for flows, please see here:

[Change the owner of a cloud flow in Power Automate - Power Automate | Microsoft Learn](#)

and here:

[Authenticate connections with managed identities - Azure Logic Apps | Microsoft Learn](#)

Q&A

Instead of using Logic Apps to update record in Dataverse from Azure Data Factory pipeline activity, can we call the Dataverse action directly from ADF?



Dataverse is supported as a sink for copy activity in Azure Data Factory:

[Copy activity - Azure Data Factory & Azure Synapse | Microsoft Learn](#)

You should evaluate pros and cons between the different capabilities offered by copy activity in ADF and the Logic Apps connector.

Can you recommend when to use or not to use either of these (as they kind of have similar capabilities): Event Grid, Event Hub, Service Bus?



Use Event Grid for reactive programming, for example event notifications of a condition or state change, Event Hub for big data pipelines and Service Bus for high-value enterprise messaging.

See more for events vs messages and in depth comparison of these services:

[Compare Azure messaging services - Azure Service Bus | Microsoft Learn](#)

When are you planning to include Azure VNET integration with Plugins?



Stay tuned, this is planned for Jan 2024. See more:

[Connect Dataverse plug-ins to endpoint-enabled resources | Microsoft Learn](#)

Q&A

Nowadays there are connectors for a lot of platforms. Why would I choose a code-first approach when I could do the same thing with Power Automate? What should I do when the existing connector is not offering same functionality available in the platform?



If a connector is available for your scenario, and it fulfills your requirements, it's worth considering that before planning to implement custom code. However, there might be other factors to choose a code-first approach, like for example existing pre-requisites in your project architecture, or the need for more flexibility in exception handling or retry policies, etc. Please see previous slides for more information.

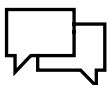
If you use an F&O virtual table to trigger an event in Dataverse, will that event trigger the data event framework in F&O or is the Dataverse trigger completely outside of F&O meaning we will not spend any resources in F&O?



With F&O virtual tables, data resides in F&O and it may trigger business events. See more:

[Enable Virtual Tables to support Dataverse events \(Microsoft Dataverse\) - Power Apps | Microsoft Learn](#)

Is there any deprecation plan for workflow activities?



Please monitor following links for more information on planned deprecations:

[Important changes \(deprecations\) coming in Power Apps and Power Automate - Power Platform | Microsoft Learn](#)

Q&A

Regarding importing data into Dataverse, the Initial Sync has a hard limitation regarding the number of records and the number of lookups supported for a single table. Are there any plans to update/expand the built-in Initial Sync functionality?



There are no plans to extend the limits associated with using initial sync (for dual-write customers). When over the limits, you can use data migration tools to import data in Dataverse and/or F&O and then start the table maps and skip initial sync.

How useful are Dataflows in Dataverse for integrations?



Dataverse dataflows are an option you can evaluate, basing on your requirements.

They're probably not suggested for complex integrations or orchestrations but may be useful for simple data migration or data import tasks.

You can learn more on Dataverse dataflows here:

[Create and use dataflows in Power Apps - Power Apps | Microsoft Learn](#)

Is CRM SDK now superseded by Dataverse ServiceClient?



The SDK is not going away, we are transitioning to a new SDK which has improved capabilities, for example support for .NET Core framework. The new SDK is the suggested option for online applications connecting to Dynamics 365;; the change is not affecting plugin assemblies:

[Transition client applications to Dataverse ServiceClient - Power Apps | Microsoft Learn](#)

Dankie Faleminderit **Shukran** Chnorakaloutioun Hvala Blagodaria
 Děkuji **Tak** Dank u Tänan Kiitos **Merci** Danke Ευχαριστώ A dank
 Mahalo ἰτι. **Dhanyavād** Köszönöm Takk Terima kasih **Grazie** Grazzi

Thank you!

감사합니다 Paldies Choukrane Ačiū Благодарам ありがとうございます
 谢谢 Баярлалаа **Dziękuję** Obrigado Mulțumesc **Спасибо** Ngiyabonga
 Ďakujem Tack Nandri Kop khun Teşekkür ederim Дякую Хвала Diolch

