User Scenarios — Origo Cloud Events DocEx

Publisher: Origo **Version:** 28.0.0.0 **Submission Date:** 2026-07-07 **Test Environment:** Requires credentials for at least one document exchange provider (Advania, Unimaze, or InExchange). Test accounts are available from each provider. BIS 3.0 reference data and UBL rendering work without credentials.

Test Credentials

Provider	Credential Type	How to Obtain
Advania	API key + company ID	Contact Advania support (test sandbox available)
Unimaze	OAuth client credentials	Register at Unimaze developer portal
InExchange	API key	Contact InExchange partner support
BIS 3.0	None required	Reference data is bundled
UBL Rendering	None required	Uses local BC data only

Scenario 1: Install Extension

Area: Installation

Setup

1. Open a Business Central environment (online sandbox or production).
2. Ensure "Origo Cloud Events Core" is already installed.

Steps

1. Navigate to Extension Management.
2. Search for "Origo Cloud Events DocEx".
3. Install the extension.
4. Wait for installation to complete.

Expected Results

- Extension installs without errors.
- No data loss or service interruption.
- Cloud Events Setup page shows new document exchange message types.

Scenario 2: BIS 3.0 Country Codes Lookup

Area: BIS 3.0 Reference Data

Setup

1. Extension is installed.
2. No provider credentials required.

Steps

1. Submit a Cloud Event message with Type = `DocumentExchange.BIS30.CountryCodes`.

2. Leave Subject and Send Content empty (full list) or provide a filter in the JSON payload.
3. Process the task via the Task API.
4. Retrieve the response from the Data API.

Expected Results

- Response contains a JSON array of ISO country codes with names as defined by Peppol BIS 3.0.
- Response Content-Type is `application/json`.

Scenario 3: BIS 3.0 Document Type Codes

Area: BIS 3.0 Reference Data

Setup

1. Extension is installed.

Steps

1. Submit a Cloud Event message with Type = `DocumentExchange.BIS30.DocumentTypeCodes`.
2. Process the task.
3. Retrieve the response.

Expected Results

- Response contains Peppol document type codes (e.g., 380 = Commercial Invoice, 381 = Credit Note).
- Data matches the current Peppol BIS 3.0 specification.

Scenario 4: Advania — Receive Unread Documents

Area: Advania / Receive

Setup

1. Configure Advania credentials in Cloud Events Setup (API key, company ID).
2. Ensure the Advania test sandbox has at least one unread document.

Steps

1. Submit a Cloud Event message with Type = `DocumentExchange.Advania.GetUnread`.
2. Process the task.
3. Retrieve the response.

Expected Results

- Response contains a JSON array of unread document references (document ID, sender, date, type).
- Each entry includes enough information to fetch the full document.

Scenario 5: Advania — Get Full Document

Area: Advania / Receive

Setup

1. Advania credentials configured.
2. A document ID is known (from Scenario 4 or test data).

Steps

1. Submit a Cloud Event message with Type = `DocumentExchange.Advania.GetDocument`.
2. Set Subject to the document ID (or provide in JSON payload).
3. Process the task.
4. Retrieve the response.

Expected Results

- Response contains the full document data (structured JSON with header, lines, amounts).
- Document fields include sender info, dates, currency, line items.

Scenario 6: Advania — Send Invoice (CreateInvoice)

Area: Advania / Send

Setup

1. Advania credentials configured.
2. A posted sales invoice exists in BC with valid customer electronic address.

Steps

1. Submit a Cloud Event message with Type = `DocumentExchange.Advania.CreateInvoice`.
2. Provide the invoice number as Subject or in JSON payload.
3. Process the task.
4. Retrieve the response.

Expected Results

- Response confirms the invoice was submitted to Advania.
- Response includes a tracking ID or reference for status follow-up.
- No error message in the response.

Scenario 7: Advania — Status Sync

Area: Advania / Status

Setup

1. Advania credentials configured.
2. At least one previously submitted document exists.

Steps

1. Submit a Cloud Event message with Type = `DocumentExchange.Advania.StatusSync`.
2. Process the task.
3. Retrieve the response.

Expected Results

- Response contains updated status information for submitted documents.
- Statuses reflect the current state in the Advania system (e.g., delivered, rejected, pending).

Scenario 8: Unimaze — Receive Unread Documents

Area: Unimaze / Receive

Setup

1. Configure Unimaze credentials in Cloud Events Setup.
2. Ensure the Unimaze test environment has unread documents.

Steps

1. Submit a Cloud Event message with Type = `DocumentExchange.Unimaze.GetUnread`.
2. Process the task.
3. Retrieve the response.

Expected Results

- Response contains a JSON array of unread document references from Unimaze.
- Each entry includes document ID, sender, document type, and received date.

Scenario 9: Unimaze — Submit Transaction

Area: Unimaze / Send

Setup

1. Unimaze credentials configured.
2. A valid UBL document or BC invoice ready for submission.

Steps

1. Submit a Cloud Event message with Type = `DocumentExchange.Unimaze.SubmitTransaction`.
2. Provide document data in Send Content (UBL XML or reference to BC document).
3. Process the task.
4. Retrieve the response.

Expected Results

- Response confirms successful submission to Unimaze network.
- Response includes a transaction ID for tracking.

Scenario 10: InExchange — Get Incoming Documents

Area: InExchange / Receive

Setup

1. Configure InExchange credentials in Cloud Events Setup.
2. Ensure InExchange test account has incoming documents.

Steps

1. Submit a Cloud Event message with Type = `DocumentExchange.InExchange.GetIncoming`.
2. Process the task.
3. Retrieve the response.

Expected Results

- Response contains a list of incoming documents from InExchange.

- Each document includes ID, sender, type, and status.

Scenario 11: InExchange — Send Document

Area: InExchange / Send

Setup

1. InExchange credentials configured.
2. A document ready for electronic delivery.

Steps

1. Submit a Cloud Event message with Type = `DocumentExchange.InExchange.SendDocument`.
2. Provide document data in Send Content.
3. Process the task.
4. Retrieve the response.

Expected Results

- Response confirms document was submitted to InExchange.
- Outbound tracking ID is returned.

Scenario 12: UBL Rendering — Invoice

Area: UBL Rendering

Setup

1. Extension installed (no external credentials needed).
2. A posted sales invoice exists in BC.

Steps

1. Submit a Cloud Event message with Type = `DocumentExchange.UBL.RenderBilling`.
2. Set Subject to the posted sales invoice number.
3. Process the task.
4. Retrieve the response.

Expected Results

- Response contains valid UBL 2.1 Invoice XML.
- XML validates against Peppol BIS 3.0 billing schema.
- All mandatory fields are populated (supplier, customer, lines, totals, tax).

Scenario 13: UBL Rendering — Order

Area: UBL Rendering

Setup

1. Extension installed.
2. A sales order exists in BC.

Steps

1. Submit a Cloud Event message with Type = `DocumentExchange.UBL.RenderOrder`.
2. Set Subject to the sales order number.
3. Process the task.
4. Retrieve the response.

Expected Results

- Response contains valid UBL 2.1 Order XML.
- Order lines, quantities, prices, and delivery information are correctly rendered.

Scenario 14: Partner Lookup (Advania)

Area: Advania / Partner Discovery

Setup

1. Advania credentials configured.

Steps

1. Submit a Cloud Event message with Type = `DocumentExchange.Advania.GetTradingPartners`.
2. Optionally provide a search filter in JSON payload (e.g., by name or electronic address).
3. Process the task.
4. Retrieve the response.

Expected Results

- Response contains a list of trading partners registered in the Advania network.
- Each partner entry includes name, electronic address, and supported document types.

Scenario 15: Document PDF Retrieval

Area: Advania / Document Retrieval

Setup

1. Advania credentials configured.
2. A document ID is known for a document that has a PDF representation.

Steps

1. Submit a Cloud Event message with Type = `DocumentExchange.Advania.GetDocumentPdf`.
2. Set Subject to the document ID.
3. Process the task.
4. Retrieve the response.

Expected Results

- Response contains the PDF binary data (base64 encoded or as blob).
- Content-Type indicates PDF.
- PDF is viewable and matches the source document.

Scenario 16: Error Handling — Invalid Credentials

Area: Error Handling

Setup

1. Configure Cloud Events Setup with deliberately invalid provider credentials (wrong API key).

Steps

1. Submit a Cloud Event message with Type = `DocumentExchange.Advania.GetUnread`.
2. Process the task.
3. Retrieve the response.

Expected Results

- Response contains a structured error JSON (not a raw exception).
- Error message clearly indicates authentication failure.
- No unhandled exceptions or BC error dialogs.
- System remains functional for other message types.

Scenario 17: Permissions

Area: Security

Setup

1. Create a test user with no special permissions.
2. Assign the Cloud Events DocEx permission set to the user.

Steps

1. Log in as the test user.
2. Access Cloud Events Setup and verify document exchange settings are visible.
3. Submit and process a BIS 3.0 reference data request.
4. Remove the permission set from the user.
5. Attempt the same operation.

Expected Results

- With permission set: User can access setup and process document exchange messages.
- Without permission set: User receives a permission error and cannot execute document exchange operations.
- No elevation of privileges beyond what the permission set grants.

Scenario 18: Uninstall Extension

Area: Installation

Setup

1. Extension is installed with data from previous scenarios.

Steps

1. Navigate to Extension Management.
2. Uninstall "Origo Cloud Events DocEx".
3. Confirm uninstallation.

Expected Results

- Extension uninstalls cleanly without errors.
- No orphaned data or broken references in the Cloud Events Core tables.
- Other Cloud Events extensions continue to function normally.
- Re-installation is possible without issues.