

Adobe Commerce - SAP Business One Integration

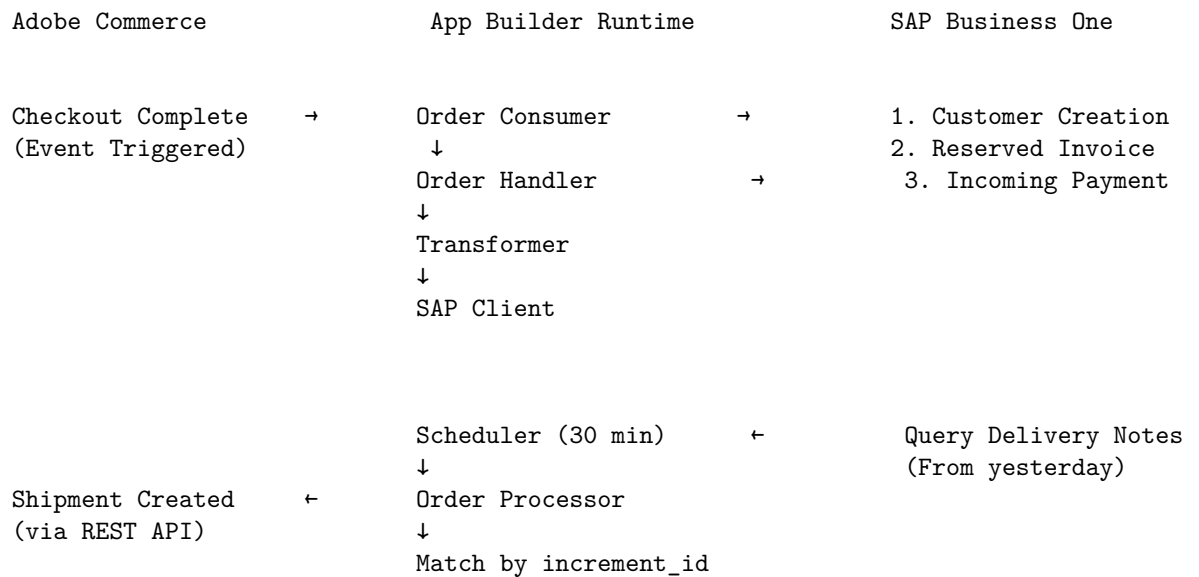
Overview

This integration synchronizes orders and shipments between Adobe Commerce and SAP Business One using Adobe Developer App Builder.

What It Does

Feature	Description
Order Sync	When a customer completes checkout in Adobe Commerce, the order is automatically created in SAP B1
Shipment Sync	When a delivery note with tracking number is created in SAP B1, a shipment is created in Adobe Commerce
Sync Method	Real-time event for orders + 30-minute scheduled polling for shipments
Event Trigger	checkout_submit_all_after

Architecture



Order Creation Flow (Commerce → SAP B1)

When a customer completes checkout, the integration performs three steps in SAP B1:

Step 1: Customer Creation/Validation

Customer Code Format: - Guest Orders: C{increment_id}{country_code}
- Example: C000123456UK - Registered Customers: C{customer_id}{country_code}
- Example: C789IE

VAT Status: - UK customers → vExempted - All other countries → vLiable

Step 2: Reserved Invoice Creation

Creates a draft invoice in SAP B1 with: - **NumAtCard** = Adobe Commerce increment_id (critical for order matching) - Customer code - Order line items
- **Shipping cost as separate line item** - Taxes and totals

Important: Shipping/delivery charges are added as a separate line item in the SAP reserved invoice, not included in product line items.

Step 3: Incoming Payment

Creates payment record linked to the reserved invoice.

Shipment Sync Flow (SAP B1 → Commerce)

A scheduler runs every 30 minutes to check for new delivery notes:

1. **Query SAP B1:** Get delivery notes from yesterday onwards
2. **Filter:** Only notes with tracking numbers
3. **Match Orders:** Use NumAtCard field to match with Commerce increment_id
4. **Create Shipment:** Add tracking number to Commerce order

Installation

Prerequisites

- Adobe Developer Console access
- Adobe Commerce 2.4.4+
- SAP Business One with Service Layer API
- Node.js 18+

Setup Steps

1. **Create App Builder Project**
 - Go to Adobe Developer Console

- Create new App Builder project
- Add required APIs: I/O Events, Adobe I/O Events for Commerce, I/O Management API
- For SaaS deployments, also add: **Adobe Commerce as a Cloud Service API** (required for calling REST APIs). See API Integration documentation for details
- Download workspace configuration JSON

2. Install & Configure

```
npm install
cp env.dist .env
# Edit .env with your credentials
```

3. Configure Environment Variables

Adobe Commerce (PaaS - OAuth1):

```
COMMERCE_BASE_URL=https://your-domain.com/rest/
COMMERCE_CONSUMER_KEY=
COMMERCE_CONSUMER_SECRET=
COMMERCE_ACCESS_TOKEN=
COMMERCE_ACCESS_TOKEN_SECRET=
COMMERCE_GRAPHQL_ENDPOINT=
```

Adobe Commerce (SaaS - IMS OAuth):

```
OAUTH_CLIENT_ID=
OAUTH_CLIENT_SECRET=
OAUTH_TECHNICAL_ACCOUNT_ID=
OAUTH_TECHNICAL_ACCOUNT_EMAIL=
OAUTH_ORG_ID=
OAUTH_SCOPES=AdobeID, openid, read_organizations, additional_info.projectedProductContent
OAUTH_BASE_URL=https://ims-na1.adobelogin.com/ims/token/
# OAUTH_HOST= (optional, defaults to https://ims-na1.adobelogin.com)
```

Commerce Event Module Configuration:

```
COMMERCE_ADOBE_IO_EVENTS_MERCHANT_ID=your_company_name
```

Note: Set this to your merchant's company name. Value can only contain alphanumeric characters and underscores.

SAP B1:

```
SAP_BASE_URL=https://your-sap-instance.com
SAP_COMPANY_DB=
SAP_USERNAME=
SAP_PASSWORD=
```

Note: For information on how to obtain SAP Business One credentials and configure the Service Layer API, refer to the

SAP Business One Service Layer documentation.

SAP B1 Store and Payment Mappings:

```
STORE_ID_MAPPING={1:{"U_TRC_DIMENSION1":"C001","COGSCostingCode":"C001","CostingCode2":  
PAYMENT_METHOD_MAPPING={"chcybersource":1,"paypal_express":13,"hummm_gateway":6}}
```

Note: Configure these mappings based on your SAP B1 setup. `STORE_ID_MAPPING` maps Commerce store IDs to SAP dimensions and costing codes. `PAYMENT_METHOD_MAPPING` maps Commerce payment method codes to SAP payment method codes.

Adobe Workspace:

```
IO_MANAGEMENT_BASE_URL=https://api.adobe.io/events/  
IO_CONSUMER_ID=  
IO_PROJECT_ID=  
IO_WORKSPACE_ID=  
AIO_runtime_namespace=
```

Note: `AIO_runtime_namespace` format is `{orgId}-{projectName}-{workspaceName}`. This can be found in the workspace JSON or set automatically during deployment.

4. Connect to App Builder

```
aio login  
aio console org select  
aio console project select  
aio console workspace select  
aio app use
```

5. Deploy

```
aio app deploy
```

6. Run Onboarding

The configuration files are already pre-configured for this integration:

- `./scripts/onboarding/config/starter-kit-registrations.json`
- configured for order synchronization
- `./scripts/onboarding/config/events.json` - configured for `checkout_submit_all_after` event

Run:

```
npm run onboard
```

7. Configure Adobe Commerce

- Place the downloaded `workspace.json` file in the `./scripts/onboarding/config/` directory

- Navigate to: Stores > Configuration > Adobe Services > Adobe I/O Events
- Enable Adobe I/O Events
- Copy the contents of the `workspace.json` file into the **Adobe I/O Workspace Configuration** field
- Set provider ID and instance ID (from onboarding output)

8. Subscribe to Event

```
npm run commerce-event-subscribe
```

Configuration

Event Subscription

This integration uses only one event:

Event	Description	Required Fields
<code>checkout_submitTriggered</code>	Triggered after checkout completion	<code>increment_id</code> , <code>customer_id</code> , <code>items</code> , <code>billing_address</code> , <code>shipping_address</code> , <code>payment</code> , <code>shipping_method</code>

SAP B1 Endpoints

Endpoint	Purpose	Key Fields
<code>/BusinessPartners</code>	Customer creation	<code>CardCode</code> , <code>CardName</code> , <code>VatStatus</code>
<code>/Orders</code>	Reserved invoice	<code>NumAtCard</code> (<code>increment_id</code>), <code>DocumentLines</code> (including shipping line)
<code>/IncomingPayments</code>	Payment records	<code>CardCode</code> , <code>PaymentInvoices</code>
<code>/DeliveryNotes</code>	Delivery tracking	<code>NumAtCard</code> , <code>TrackingNumber</code> , <code>DocDate</code>

Important Notes

- **Shipping Cost:** Added as a separate line item in the reserved invoice
- **Order Matching:** Uses `NumAtCard` field in SAP to store Commerce `increment_id`
- **Scheduler Frequency:** Runs every 30 minutes to check for new deliveries
- **Query Window:** Checks delivery notes from yesterday onwards (`DocDate >= yesterday`)

Project Structure

```
actions/
  order/
    commerce/created/      # Order creation handler (Commerce → SAP)
    commerce-order-api-client.js
    commerce-shipment-api-client.js
  ingestion/
    order-sync/            # 30-minute scheduler for delivery sync (SAP → Commerce)
  auth.js
  sap-client.js            # SAP B1 API client
  responses.js
scripts/
  onboarding/              # Setup automation
  commerce-event-subscribe/ # Event subscription
app.config.yaml            # Runtime configuration
package.json
.env                       # Credentials (not in git)
```

Troubleshooting

Orders Not Creating in SAP B1

Check: - Event subscription in Commerce Admin - SAP credentials in .env -
Action logs: `aio runtime activation list --name order-commerce/consumer`

Test SAP Connection:

```
curl -X POST "${SAP_BASE_URL}/Login" \
  -H "Content-Type: application/json" \
  -d '{"CompanyDB":"DB","UserName":"user","Password":"pass"}'
```

Shipments Not Creating

Check: - NumAtCard field is populated in SAP orders - Scheduler is running: `aio runtime activation list --name ingestion/order-sync` - Tracking numbers exist in SAP DeliveryNotes - Date filter (from yesterday) is working

Manual Trigger:

```
aio runtime action invoke ingestion/order-sync --result
```

Customer Code Issues

Ensure consistent format: - Guest: `C{increment_id}{country_code}` - Registered: `C{customer_id}{country_code}` - Country code must match between systems

VAT Status Issues

- UK/GB → vExempted
- Other countries → vLiable

Monitoring

View Logs

List recent activations

```
aio runtime activation list
```

View specific logs

```
aio runtime activation logs <activation-id>
```

Real-time polling

```
aio runtime activation poll
```

Debug Tracing

1. Adobe Developer Console → Events → Event Registrations
2. Select your registration → Debug Tracing tab
3. Check event delivery status and responses

Deployment

Development

```
aio app deploy
```

Production

```
aio console workspace select # Choose production
```

```
aio app deploy --force-build
```

Verify

```
aio runtime action list
```

```
aio runtime action get order-commerce/created
```

Authentication

Adobe Commerce

- **PaaS:** OAuth1 (configure COMMERCE_* variables)
 - Requires: COMMERCE_CONSUMER_KEY, COMMERCE_CONSUMER_SECRET, COMMERCE_ACCESS_TOKEN, COMMERCE_ACCESS_TOKEN_SECRET
- **SaaS:** IMS OAuth (configure OAUTH_* variables, leave COMMERCE_* blank)

- Requires: OAUTH_CLIENT_ID, OAUTH_CLIENT_SECRET, OAUTH_TECHNICAL_ACCOUNT_ID, OAUTH_TECHNICAL_ACCOUNT_EMAIL, OAUTH_ORG_ID, OAUTH_SCOPES
- Optional: OAUTH_HOST (defaults to <https://ims-na1.adobelogin.com>)

SAP B1

- Session-based authentication via Service Layer API
- Login endpoint: POST {SAP_BASE_URL}/Login
- Returns session cookie for subsequent API requests
- Requires: SAP_BASE_URL, SAP_COMPANY_DB, SAP_USERNAME, SAP_PASSWORD

API References

- Adobe Commerce REST API
- SAP Business One Service Layer API
- Adobe I/O Events
- App Builder

Version

1.0.2 - Order creation in SAP B1 triggered by `checkout_submit_all_after`
- Shipment synchronization via 30-minute scheduler - Customer management with VAT compliance - Reserved invoice with shipping as separate line item - Incoming payment creation - Order matching via `NumAtCard` field

Last Updated: October 2025