

# 1

## CHAPTER 1

# Choose an extraction route

Intermediate

Edition 1.0

*A report can be perfectly formatted and still be wrong. The first design decision is which business facts it measures, which interface supplies them, and what evidence will show that the extraction is complete.*

### IN THIS CHAPTER

- Define the business question
- Separate operational and posted data
- Choose a retrieval route
- Agree on completeness

## Start with a reporting contract

Finance asks for a margin report by Friday. That request hides several different products. A job profitability report may need estimated or unposted charges. A sales ledger report needs posted transactions, credit notes and payments. A cash report needs allocations and settlement dates. Treating all three as a sum of shipment charge lines creates convincing numbers with incompatible meanings.

Write a reporting contract before the first query. Name the company, business population, date basis, currency basis, refresh target and reconciliation owner. For example: “Forwarding shipments in company GB1, known local costs and revenues as of the most recent successful snapshot, in GBP, with incomplete jobs reported separately.” That is a workable operational report. It is not automatically posted revenue for the month or cash received.

The contract should also name what is excluded. A report limited to forwarding shipments must not silently add consol costs as if they were independent shipment charges. A report using local amounts must not mix companies whose local currencies differ. A report based on creation date must not imply that later changes to older jobs have been captured.

| Contract field | Example decision                                         |
|----------------|----------------------------------------------------------|
| Population     | Known forwarding shipment keys in GB1                    |
| Measure        | Local job-costing revenue less local job-costing cost    |
| Time basis     | Latest validated snapshot, not transaction posting month |
| Completeness   | All expected keys accounted for; missing costing flagged |
| Owner          | Named operations and finance reviewers                   |
| Freshness      | Last completed run displayed with the report             |

## The three useful boundaries

The vendor's **Universal XML Guide**, dated 31 October 2025, separates unposted job-related costs and revenues from posted accounting transactions. Section 5.3 describes JobCosting in Universal Shipment XML and ConsolCosts for forwarding consols. Section 10 describes Universal Transaction XML for posted accounting information, including invoices, credit notes, payments and journals. That distinction is the foundation of this volume.

Native XML serves a different purpose. The **eAdapter Developer's Guide** dated 9 October 2025 documents Native Query as a way to retrieve records from modules that support Native XML. The dataset names and fields come from the instance's available schemas. Do not assume that a Universal Shipment field name is also a Native query field, or that one dataset's criteria entity name is valid for another.

A third boundary is your own reporting store. It is where you preserve rows, reconcile controls and build stable views for Power BI or another reporting tool. In this book, the SQL examples run against a small local reporting database. They are not statements against undocumented CargoWise tables.

### TIP

Keep transport, message schema and reporting model separate. eAdapter Next has its own transport and authentication guidance. A message shape in this book is not an instruction to reuse another interface's endpoint or credentials.

## Match the question to the interface

For one known operational job, start with a Universal Shipment Request, often referred to as XSR. Section 8.2 of the Universal XML Guide documents the DataTarget Type and Key. The response contains the requested Universal Shipment or information explaining why retrieval failed. This is a useful building block for a controlled list of known job keys.

For reference data such as organisations, Native Query provides documented Key and Partial retrieval patterns. A key request targets a valid candidate key. A partial request uses criteria and, where appropriate, wildcard values. Chapter 2 shows source-backed organisation templates and explains how to verify any additional dataset before building a report around it.

For posted accounting, the classic Developer's Guide documents transaction batch requests. Its EXP action exports an existing batch. CRE creates a batch and CEX creates and exports one. A request type that sounds like a query can therefore have actions that change accounting workflow state. This guide's template uses EXP only and assumes finance has identified an existing batch.

For document retrieval, the classic guide documents Universal Document Request. Documents and their metadata are evidence associated with jobs; a PDF attachment is not a replacement for structured transaction data. Chapter 5 explains how to keep those sources connected without inferring missing accounting facts from a receipt template.

## Design for incomplete answers

The absence of JobCosting is not evidence that a shipment cost nothing. Section 5.3.1 says costing information is sensitive and is not included in exported Universal Shipment XML by default. It describes the ORP recipient role for in-house systems. Section 8.2.1.1 also explains that company context can affect the company-specific billing content returned for forwarding jobs.

That gives a useful investigation order: verify the intended company, the access and recipient configuration, the returned job identity, and the actual payload. Do not solve missing costing by inserting zeros. Record the coverage state as unknown and keep the job in the exception report until someone establishes why the data is absent.

The original research notes for this book contained a universal no-pagination assertion and an approximate 50 MB truncation threshold. The vendor references reviewed for this edition did not establish those claims. Instead of relying on them, bound each extraction using capabilities verified for your interface. Measure elapsed time, payload bytes, returned identity count and reconciliation controls. A successful HTTP response does not prove that a business population is complete.

## A small pipeline that can explain itself

**Approved population → Request and raw evidence → Validated rows  
→ Scoped reporting snapshot → Reconciled report**

Each arrow represents an explicit acceptance decision. The population has an owner. The request records its company and entity. The parser rejects unknown shapes rather than inventing amounts. The snapshot records coverage. The report displays both the numbers and the unresolved exceptions. Start with one job whose billing details a reviewer can inspect. Capture its response, redact identifiers for development, and compare every charge row and currency. Then add a second charge with the

same code, a credit, a missing cost and a different original currency. These are ordinary business cases, not exotic edge cases. Chapter 3's synthetic fixture includes them so you can observe the consequences before involving a live system.

## Exercise: define the first report

---

Write six lines: population, measure, time basis, currency basis, completeness control and owner.

Then identify one question your report deliberately does not answer. If the desired measure is posted revenue but your source is JobCosting, change the route or change the label before writing a parser.

A good result is precise enough that another developer can tell whether a row belongs in the report. A weak result says only "all shipments this month". Ask which company, which date, whether cancelled jobs count, and whether a job updated next week must change this month's figures.

## Chapter references

WiseTech Global, **eAdaptor Developer's Guide**, 9 October 2025: sections 2.5.3, 2.5.4, 2.5.6 and 2.5.8, printed pages 48 and 53-62. **Universal XML Guide**, 31 October 2025: sections 5.3.1, 8.2.1.1 and 10, printed pages 58, 101 and 112. The reporting contract and pipeline are CargoMode's proposed engineering approach.

Get the complete guide and runnable companion examples at [cargomo.de](https://cargomo.de).