pentaho
A Hitachi Group Company

ETL Metadata Injection

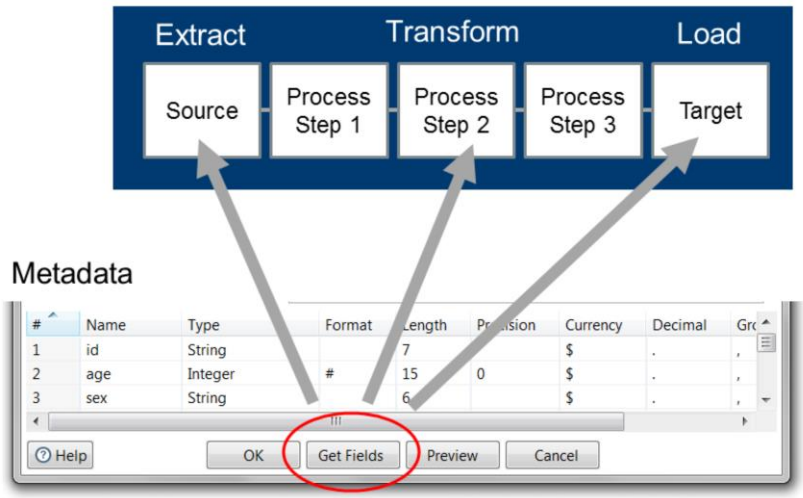
A template-based approach to dynamic data integration

Traditional ETL – Hardcoded Metadata

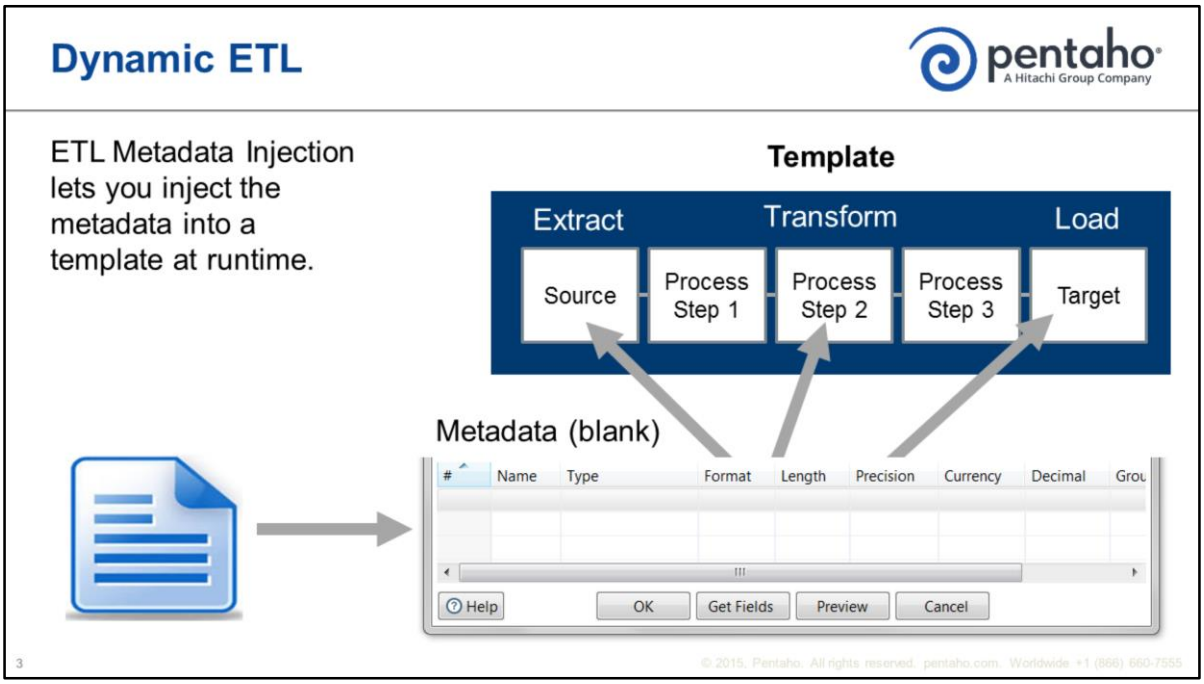
pentaho
A Hitachi Group Company

Metadata details (fields, datatypes, etc.) are required for various steps within a transformation: sources, targets, and/or transformation steps.

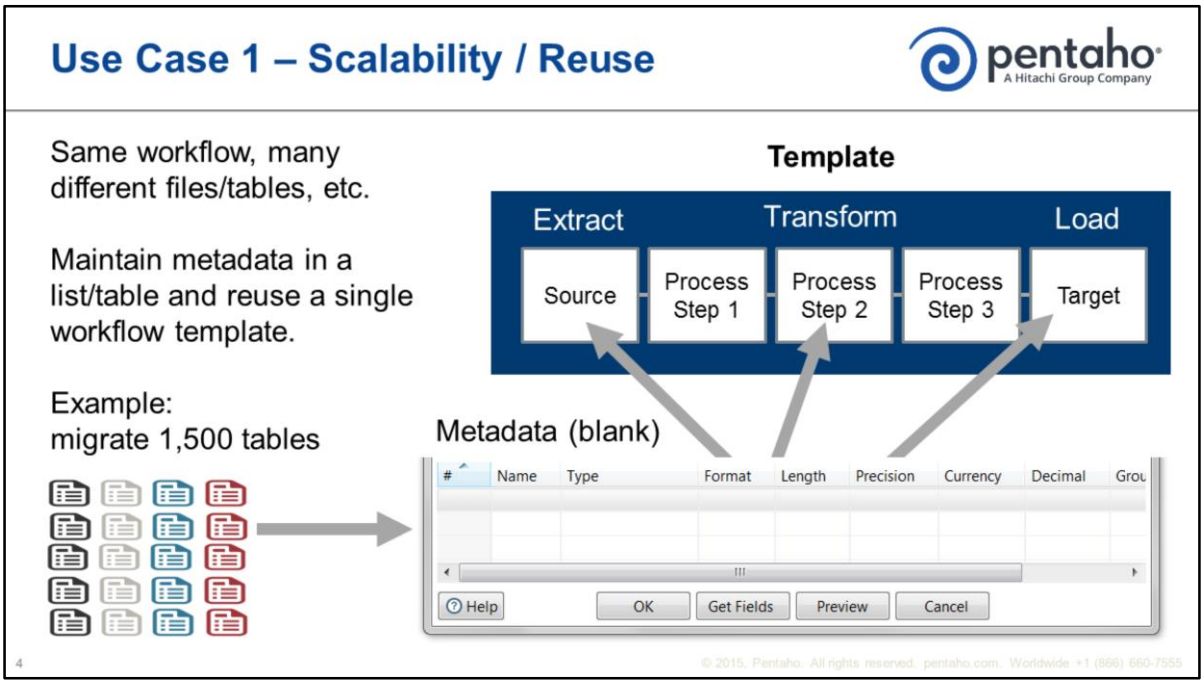
Legacy ETL tools require you to hardcode the metadata at development time.



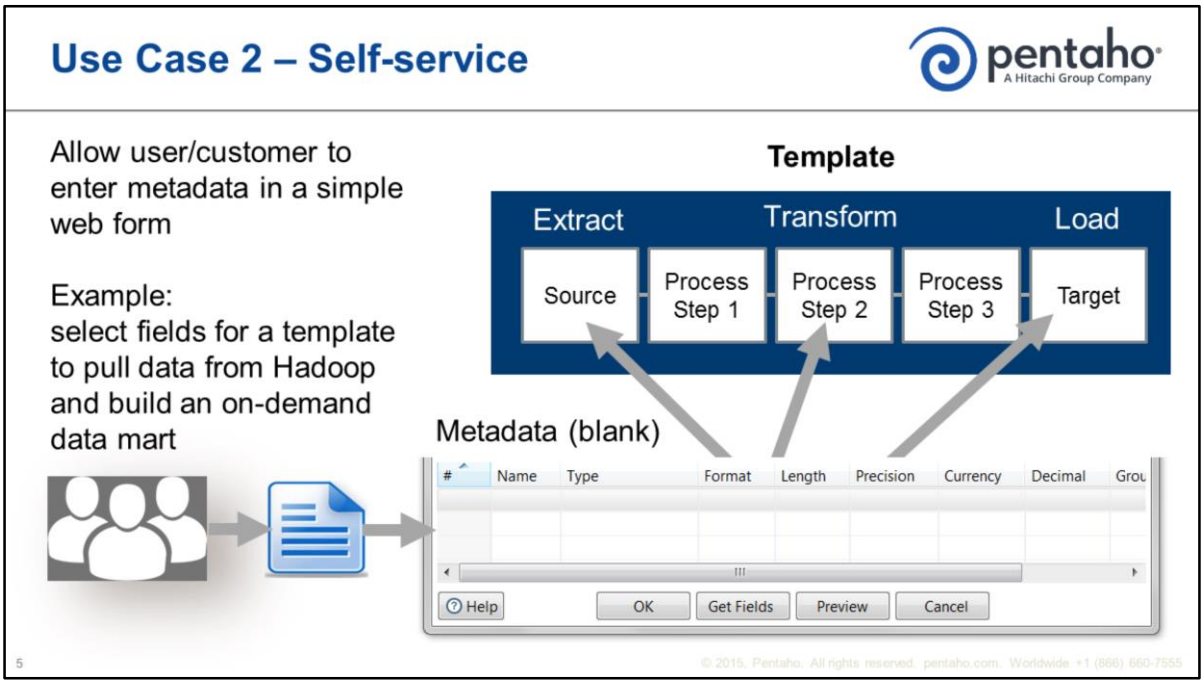
- Traditional ETL requires you to hard code metadata into the data workflow.
- This metadata includes things like field names, data types, string lengths, date formats, and so on.
- This type of metadata can appear anywhere in the workflow:
  - Source metadata are used to parse the source
  - Transformation metadata can, for example, determine which fields to group on and which to aggregate and how to aggregate
  - Target metadata can provide formatting details for the output
- This hardcoded approach often results in thousands of data workflows that essentially do the same thing. The only difference is in the hardcoded metadata.



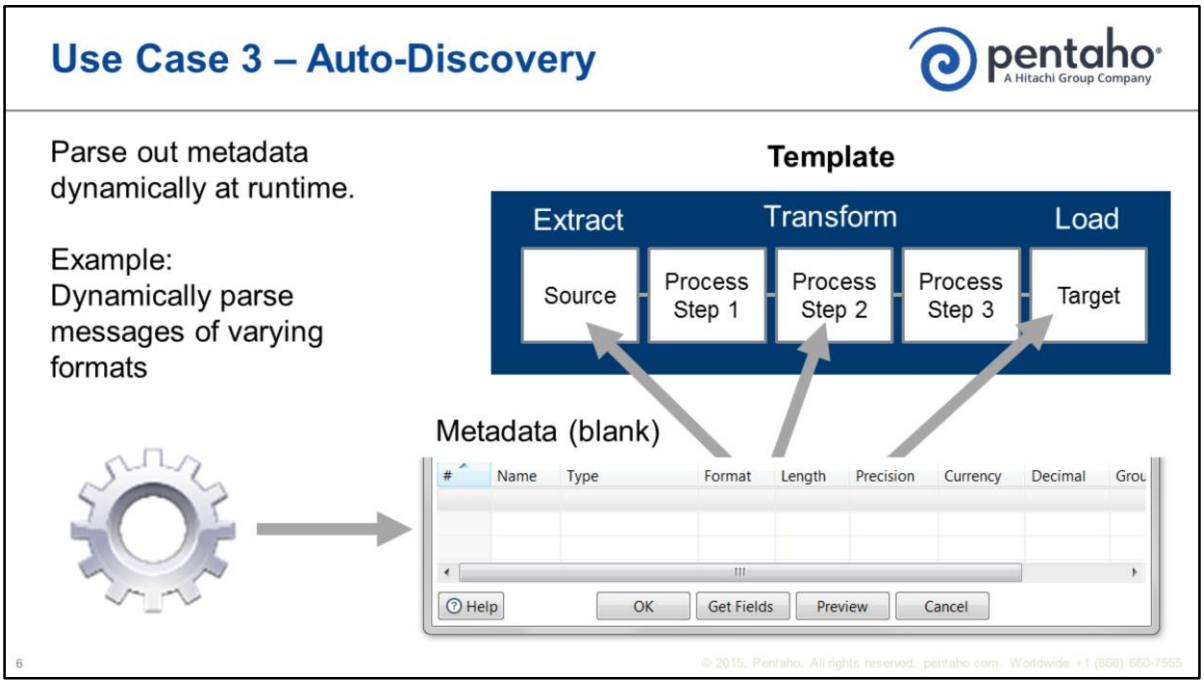
- Pentaho provides a template-based approach to ETL.
- An ETL template allows you to define the overall workflow without specifying any metadata.
- At runtime you can pull metadata from virtually any source and inject the metadata into the template, a process called “ETL metadata injection”.
- You have three options for using the rendered template:
  - You can immediately run the rendered template
  - You can save a copy of the rendered template
  - Or both, save and run the rendered template
- Note that these workflows can work with a wide array of data types: relational, flat files, NoSQL, JSON, log files, semi-structured, unstructured, and many more.



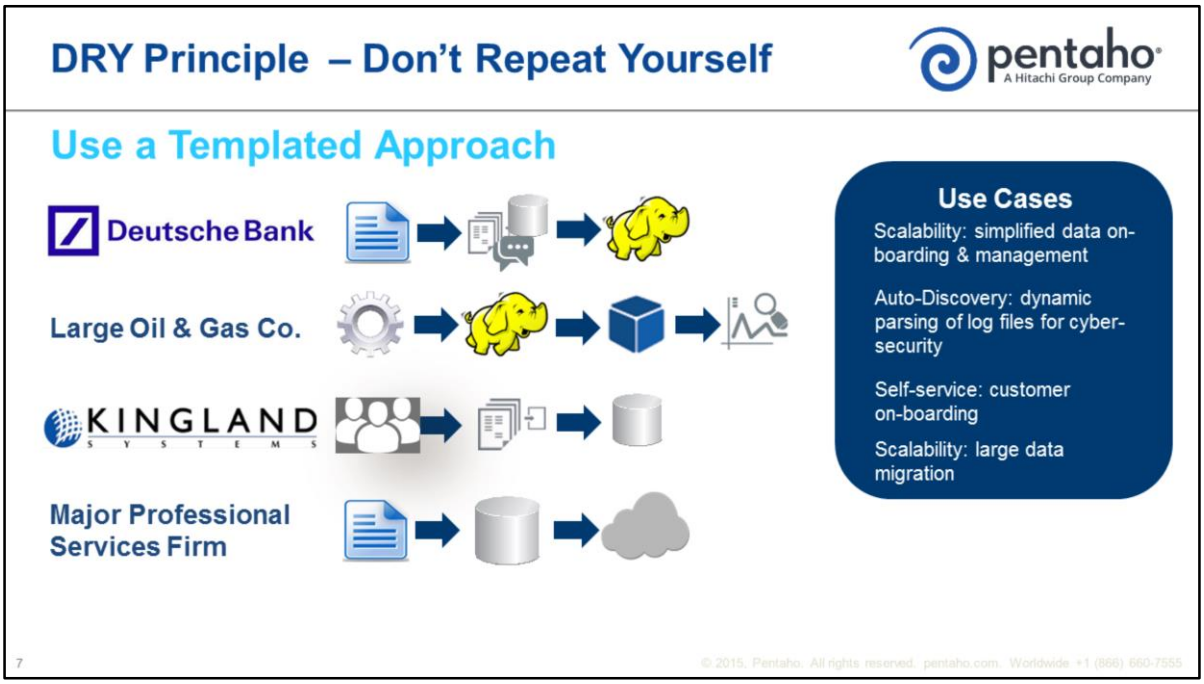
- We see three broad design patterns in use by our customers.
- The first and most common is around scalability.
- Customers can greatly simplify the development and management of workflows for many different data sources and/or targets by maintaining a catalog of metadata. This could be a mature metadata management system or even a just a few database tables.
- A single workflow template can be used. At runtime the workflow can fetch the metadata, inject it into the template and run it.
- Pentaho also provides the tools for capturing and managing the metadata as well.



- Another common use case is around self-service onboarding of data.
- You can enable your internal or external customers to upload their data in their native format and specify the metadata required to parse the data.
- Pentaho can orchestrate ingesting the data in its raw format, parse it using the metadata, and transform it into the desired target schema.
- I'll be demonstrating an example of this shortly.



- A third design pattern is what I call “auto-discovery”
- This is the ability for Pentaho to scan the source and detect the metadata at run-time
- This detection can involve precise matching, for example detecting field names for which we can look up the metadata in a catalog
- This could also involve “fuzzy” matching where natural language processing, regex pattern matching, and other techniques can be used to make intelligent decisions around the metadata



- All of these design patterns apply the DRY principle – Don't Repeat Yourself
- Here are some example of these design patterns in action:
- At Deutsche Bank, they are using a small set of Pentaho workflows and a catalog of metadata to onboard a wide array of data sources into Hadoop
- A large oil and gas company is leveraging dynamic auto-discovery of metadata for processing a variety of semi-structured logs for threat detection and cyber-security
- A major professional services firm leveraged Pentaho's ETL metadata injection to migrate 1,500 tables from DB2 to the cloud leveraging a half-dozen workflows.
- These are just a few examples.
- Next I'll show an example of self-service data onboarding.