January 2022

Data Pipelines as a Web Service

Jan-Simon Pendry

Data Pipelines as a Web Service

ABSTRACT

This disclosure describes techniques that leverage the mechanism of pipeline-designer tools to create web services. Whereas web services are typically written as software code in conventional programming languages, the described techniques deploy dataflow design tooling and workflows to provide web services, in the form of a REST API or as a RPC endpoint, without requiring the user to write code. The techniques herein described are referred to as pipeline-as-a-service.

KEYWORDS

- Data pipeline
- Web service
- Pipeline designer

BACKGROUND

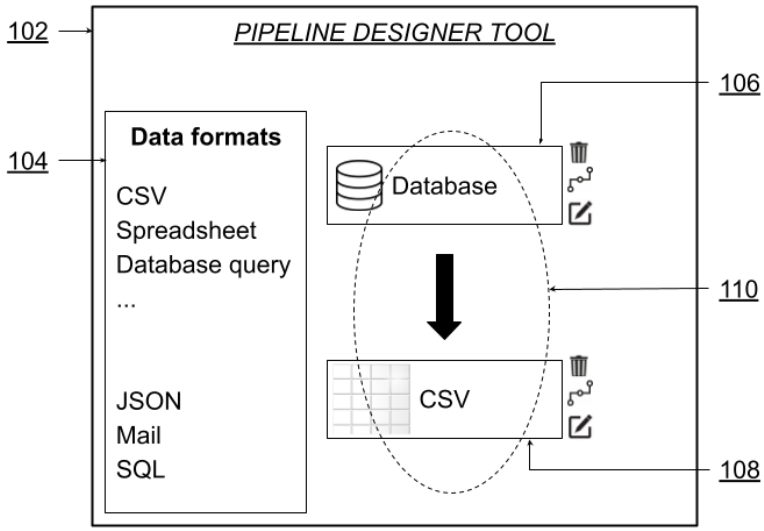

Fig. 1: User interface of a pipeline designer tool

Fig. 1 illustrates a pipeline designer tool (102). The pipeline designer tool enables a user such as an engineer to create a batch process that can take in data, possibly from a variety of sources (106), process it by transforming or grouping it, and emit (110) it to one or more targets (108). The sources and targets can be select-and-dragged from a variety of data formats (104) and linked with each other to form a graph. Such graphs are known as ETL pipelines, standing for Extract, Transform, Load, and are widely used across the industry.

The pipeline designer tool can be operated with little or no code, enabling even non-programmers to develop and execute pipelines using a graphical user interface. Workflows and pipelines can be implemented using a combination of proprietary workflow components and dataflow engines, built, e.g., on top of Apache Beam, a dataflow framework suitable for creating batch and streaming pipelines. The pipelines are typically executed in batch mode. For example, using a scheduler, a regular batch flow is launched according to an external trigger such as time-of-day. The flow is configured with parameters, some pre-configured at the time the flow is deployed and some derived from the trigger, e.g., the start time, the time the flow was last run, etc. The trigger for a given pipeline to start execution can be the completion (or the reaching of some milestone) of another pipeline.

A web service can serve data from a server to a client in response to a request from the client. The web service is typically hand-created using a programming language (e.g., Java, Go, Python, JavaScript, etc.). The creation of web services is time and labor-intensive and requires engineering expertise.

DESCRIPTION

This disclosure describes techniques that leverage the mechanism of pipeline-designer tools to create web services. Whereas web services are typically written by hand in conventional

programming languages, the described techniques deploy tooling originally intended to design dataflows and workflows to provide web services, e.g., a representational state transfer (REST) API, a remote procedure call (RPC) endpoint, etc. The techniques described herein are referred to as pipeline-as-a-service.

In the case of a web service, additional parameters are derived from the incoming web request, including the uniform resource locator (URL), the HTTP action, authentication data, along with any payload, e.g., from a POST request. The HTTP request can trigger execution of the pipeline, with the parameters in the HTTP request being used to configure the pipeline.

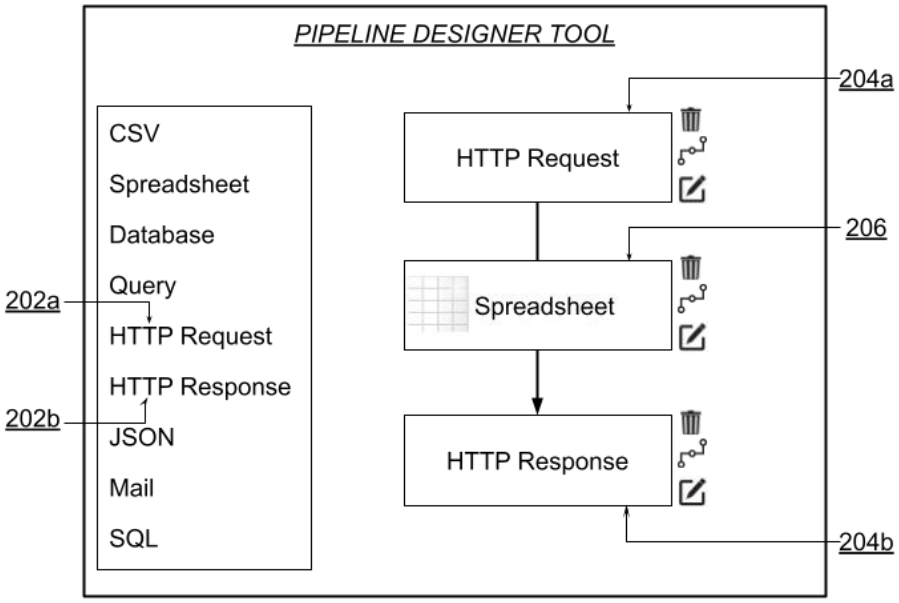

Fig. 2: A pipeline designer tool augmented to create web services

In addition to conventional source and target types, the pipeline designer UI is augmented to support HTTP requests (202a) and HTTP responses (202b). The output of the pipeline is the response to the web service request. The HTTP request and response connectors (204a-b) in the GUI include source and target URLs and other parameters that define the behavior of the

pipeline. A user of the tool can create an interconnected ETL pipeline-graph that includes HTTP requests and responses.

In the example of Fig. 2, the HTTP request includes parameters that govern the retrieval of data from a spreadsheet (206). Data from the spreadsheet is transformed and posted to the HTTP response. The pipeline itself may be read-only or may write data directly or indirectly via some other service. Execution of the pipeline is based on the credentials of the requesting user, possibly in addition to the ambient credentials of the web service itself. In an example use case, the described techniques can be used to build a user interface for a banking application with a button that fetches transaction data from the web service of the bank to render parts of the user's screen in real-time.

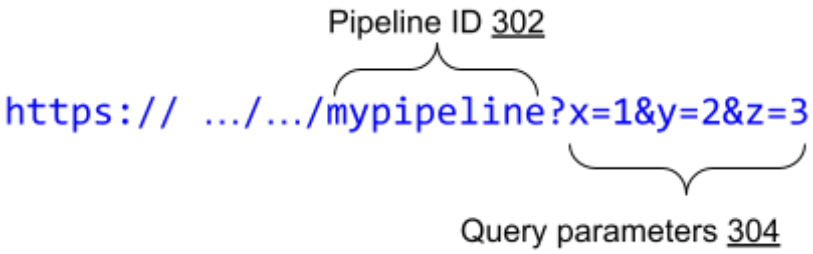

Fig. 3: Example format of URLs included in an HTTP connector

Fig. 3 illustrates an example format for URLs included within an HTTP connector. The pipeline ID (302) defines the pipeline, e.g., tells the server which pipeline to run. The query parameters (304) are parameters for the pipeline, e.g., which can be used to configure and operate the pipeline. The URL of Fig. 3 can be used in an HTTP GET request. In the case of an HTTP POST, the body of the POST, which is the data that comes into the pipeline, goes into the HTTP request connector.

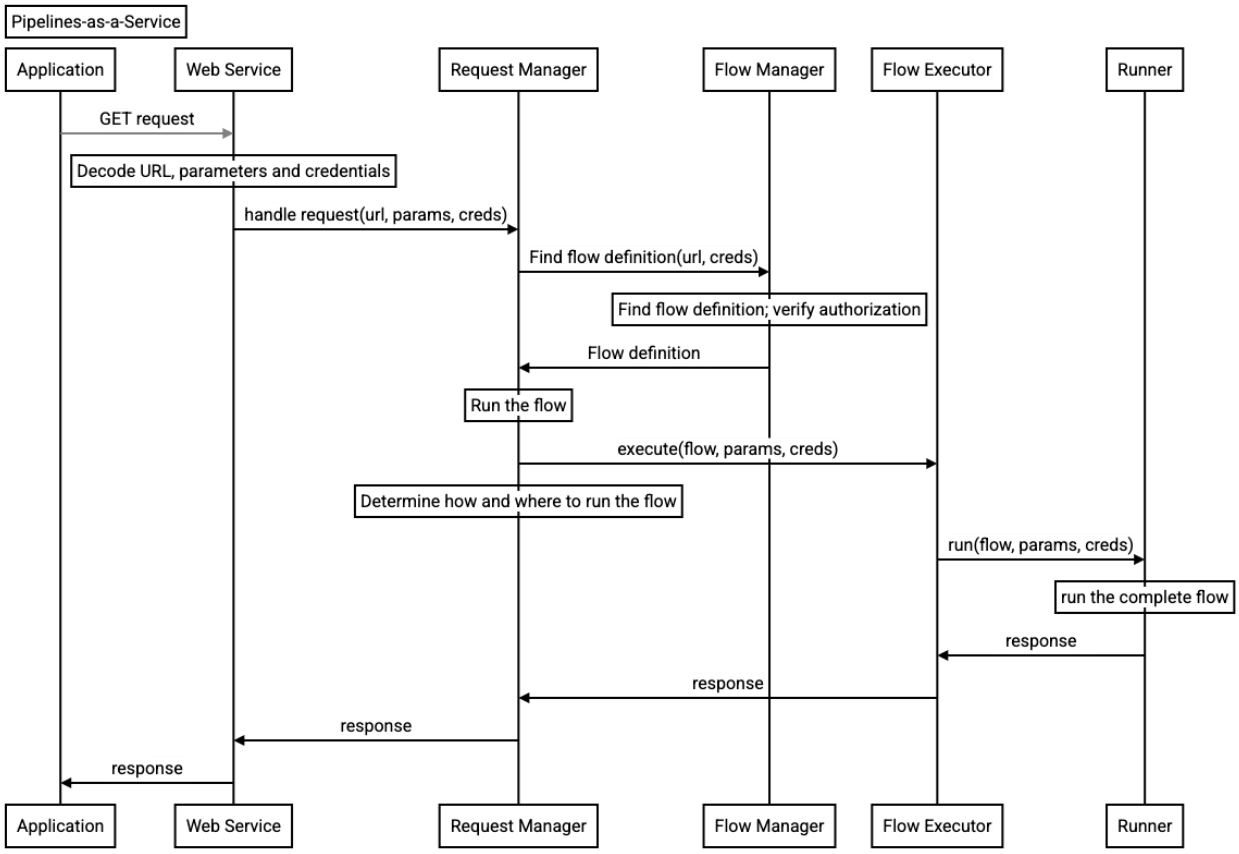

Fig. 4: Pipelines-as-a-service

Fig. 4 illustrates, in a pipeline-as-a-service, the sequence of communications between an application, a web service, and associated components (request manager, flow manager, flow executor, runner).

Bringing the capability of web services via pipeline-designer tools directly to the analyst community can provide efficiency gains for creation of services as is currently available for creation of flows. Further, such gains can be achieved without requiring analysts to learn an entirely new technology stack.

In this manner, the described techniques empower users such as business analysts or others that are not in programming or engineering roles to build and deploy web services using familiar low-code/no-code design tools and associated configuration management and

deployment functionalities. In addition, the techniques enable engineers to build and deploy web services quickly without needing to write code.

Example use case

A business analyst wants to extend the user interface (UI) of an existing customer relationship management (CRM) such that some of the displayed data is sourced from sources internal to the analyst's organization. The conventional approach is to create an ETL pipeline that fetches the necessary data from internal services, enriches the data as necessary, and uploads the enriched dataset to the CRM. This has some disadvantages, e.g., the data may be out of date, since it is updated via a scheduled batch process. More importantly, it may entail uploading internal (possibly proprietary) data to an external CRM cloud service. With the described techniques of pipelines-as-a-service, the same analysts that design and build ETL pipelines for other use cases can use ETL-pipeline technology and knowledge to build and deploy the web service.

Alternatively, a plug-in can be built for the CRM UI that can access an internal web service to fetch the necessary data on demand and render it as part of the UI. This keeps the data within organizational control and enables provision of up-to-date information to the user. However, to support the plug-in, it is required that the web service be built, e.g., hand-created by writing software code.

CONCLUSION

This disclosure describes techniques that leverage the mechanism of pipeline-designer tools to create web services. Whereas web services are typically written as software code in conventional programming languages, the described techniques deploy dataflow design tooling and workflows to provide web services, in the form of a REST API or as a RPC endpoint,

without requiring the user to write code. The techniques herein described are referred to as pipeline-as-a-service.

REFERENCES

[1] Apache Beam https://beam.apache.org/