VIVO data ingest automatization

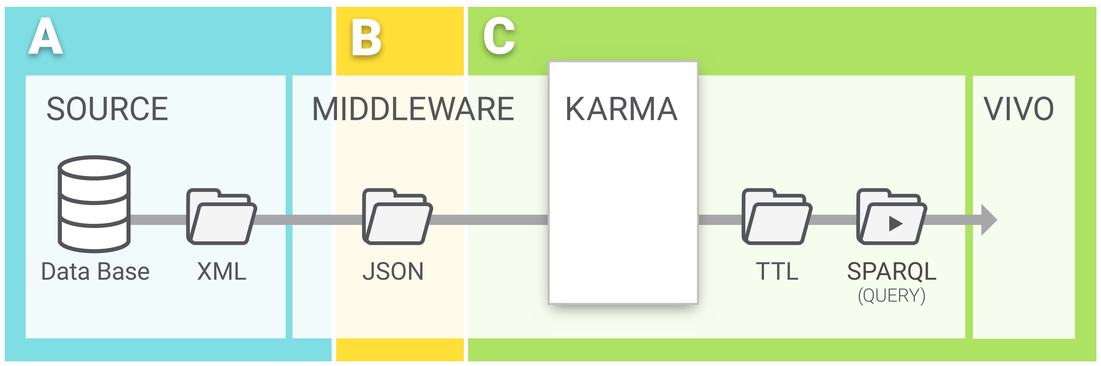
using Karma and a standard middleware with a microservice scheme

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

Since 2014, the Pontifical Catholic University of Chile started a digital transformation process to improve the academic content visibility and digital indexation. VIVO was the chosen platform for visibilizing scholar profiles and their research projects, papers, courses and curriculum. An internal interdisciplinary team (IT Department, Digital Media and Library Services, Research and Data Analysis Departments, and UX Division) have worked on the VIVO Project assessing local needs and analyzing our internal information management systems regarding scholars profiles and academic production.

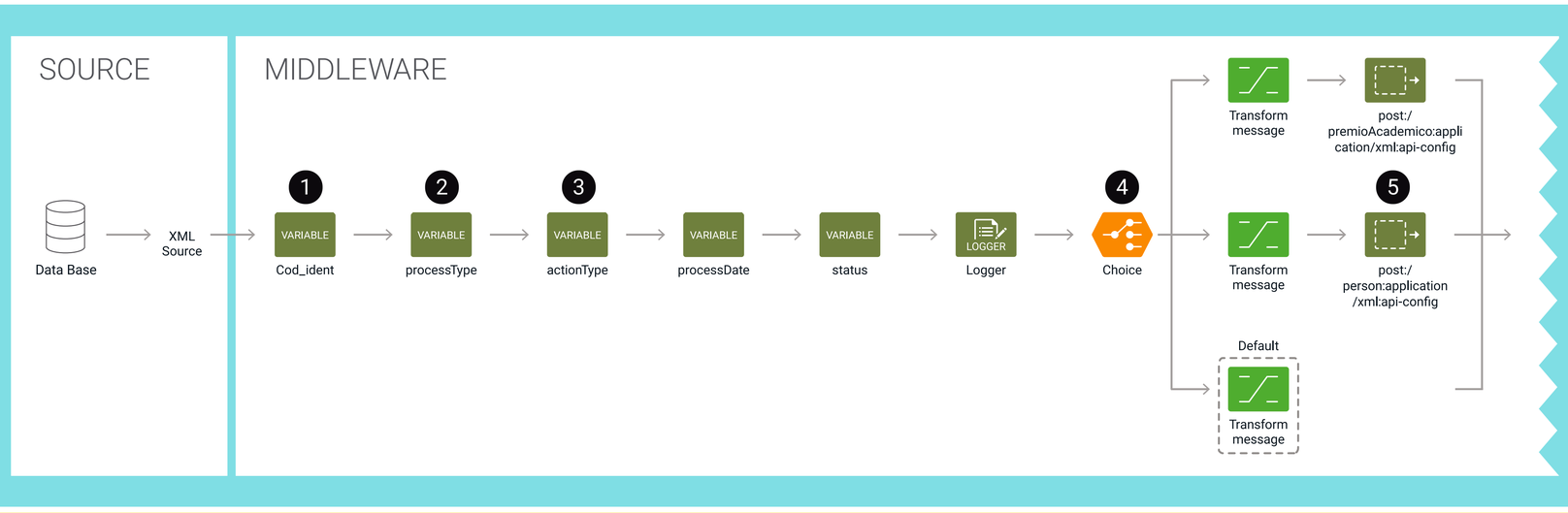
An Enterprise Service BUS (ESB) standard middleware, **Mulesoft**, used by the internal IT Department was the chosen technology to build integrations with VIVO so these integrations could be as **standard as possible** and preferably “transparent” to **ensure scalability** of the VIVO project.

This poster exemplifies the **development of this internal solution to automatically insert, modify and erase VIVO data from private internal data sources using Mulesoft in a microservice scheme, and Karma for triplets delivery**. The example given here refers to an integration flow that was developed to ingest the Person (Scholar) name extracted from an internal data source that handles the scholar`s names for all university academics.



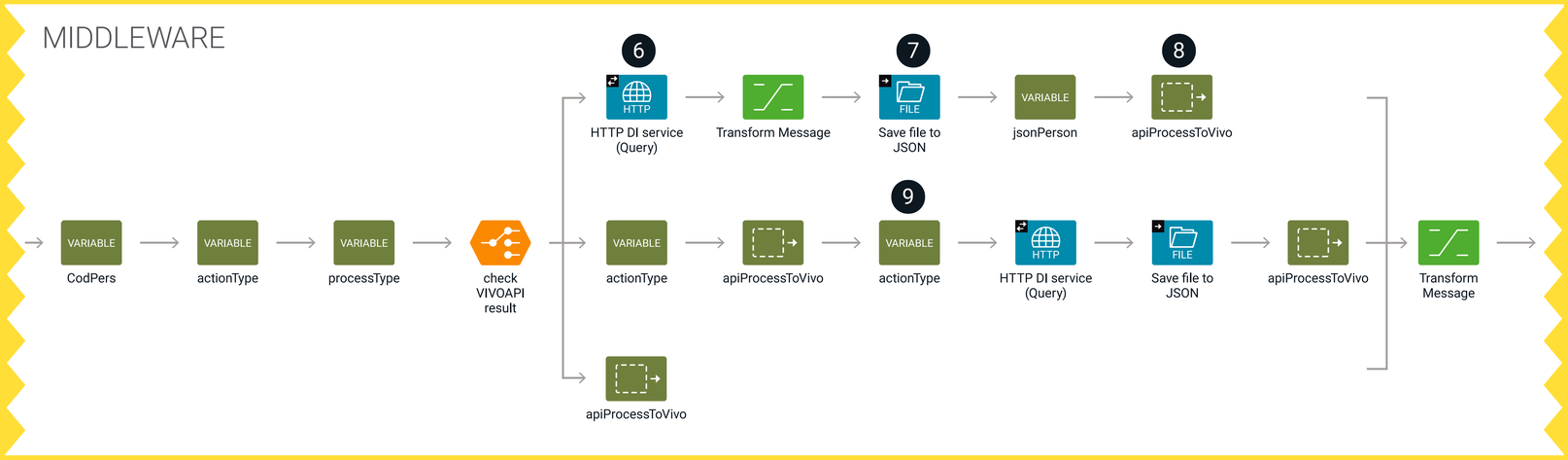
A Main Flow

This flow identifies the 1 "Person code”, the 2 "process type” in this case “Person”, the 3 “Action Type", if it is an **insert, update** or **delete** of data; the "process date” is the date from the database; and the status from the database, whether if it is ok or not ok. The 4 Choice Component identifies the type of process and sends the data to the corresponding 5 “Person” flow.



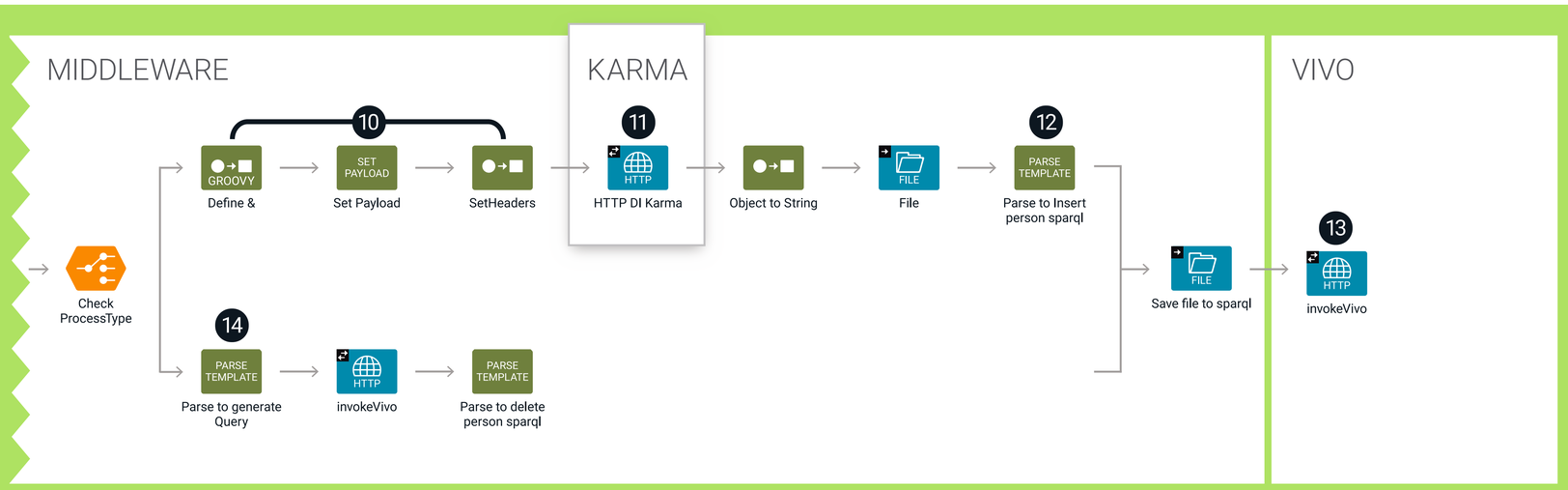
B Person Flow

Depending on the action (insert, update or delete), the flow varies: For **insert** action, queries a service at the data source 6 “HTTP DI service" (a select to a DB) is to retrieve information about the person through the person´s ID, responding with a JSON, which is transformed into a string to be saved in the 7 "filesystem as a JSON". This file is sent via the variable "jsonPerson" to the 8 “Process to VIVO” flow. For **delete** actions, the deleted data passes immediately through the “Process to VIVO” flow. For **update** actions, the data is deleted via “Process to VIVO” flow and then the 9 “action type” status is changed to **insert**.



C Process To VIVO Flow

It checks whether the data needs an insert or delete, and what is the processtype. For **insert**, the 10 headers are set to send the JSON file to the 11 KARMA service to be processed. KARMA responds with an object of type TTL that is stored in the filesystem and transformed into SPARQL using the component 12 "Parse template". Once it is stored in the filesystem, it is sent to 13 "invokeVivo", a class created in Java to send the SPARQL file into the VIVO API. For **delete**, there is a 14 generated SPARQL query sent to VIVO API to get all the Person data. Once the data is obtained, the component “Parse to delete person SPARQL” generates the query that is sent to VIVO.



Authors: Claudia Gutiérrez, Ricardo Vera, Ana Triviños, Pablo Valdivia. **Poster Design:** Alejandro Figueroa.
Dirección Digital UC, Pontificia Universidad Católica de Chile

www.vivoadvocacy.uc.cl