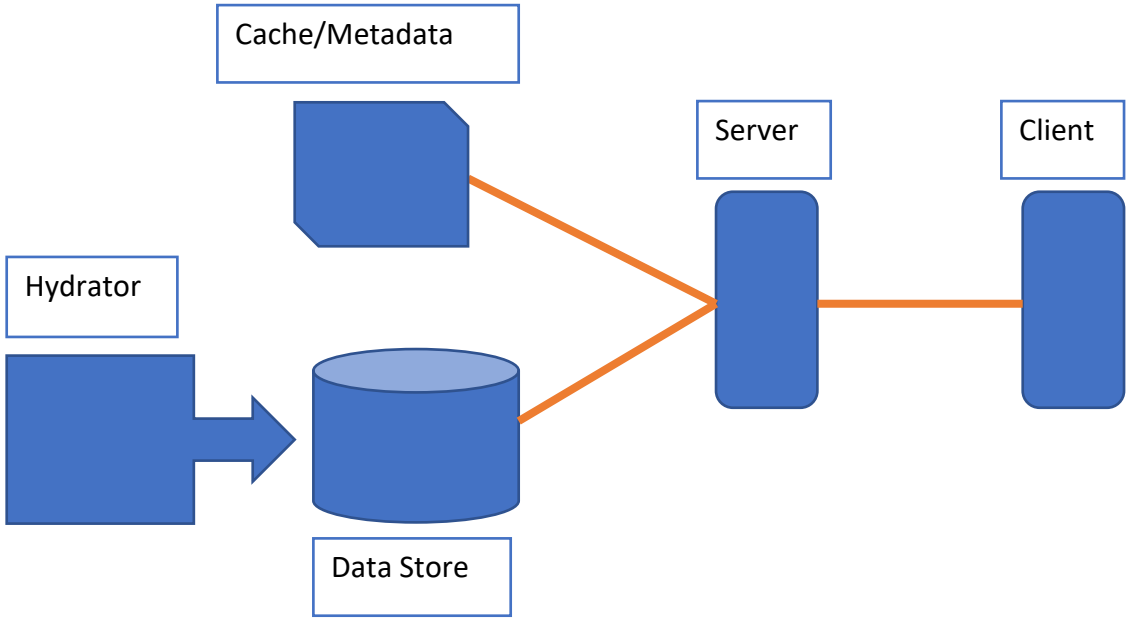
The Postal Service

Danni Beaulieu, David Souther, Peter Weisdepp, Xiaomei Xie
Seattle University, Software Architecture, 5200



Summary:

The Postal Service will be created with a basic client-server style augmented with a hydrated data store and metadata cache. These components will orchestrate the ability of a client to execute searches for existing postal addresses either with or without a country format specified.

Client:

The client is a web component which makes requests to the server on behalf of the user. A user can search for an address across countries or limited to a country-specific format. The client communicates via RPC style HTTP requests and displays the normalized responses. One option for this component is to use JavaScript to create a responsive user experience in which limited values are constrained.

Server:

The server is a back-end component which processes requests from the client, normalizes and validates the information, and returns a response to the client within a reasonable time frame to the user. The server receives communication from the client via RPC style HTTP requests, which maps to an internal model of postal addresses that can be translated into a query. The query is executed against the data store, and the server communicates back to the client via an RPC style HTTP response. Validation of a postal address may be done as either a standalone

client request or as part of a search request. In order to validate the data, the server matches it against the known country-specific formats loaded from the metadata cache. One option for this component is to use Java in tandem with an HTTP framework in order to provide reliable and robust messaging capabilities.

Data Store:

The data store is a relational database which functions as persistent storage for postal addresses across all supported countries. It is acted on by the hydrator which will insert postal addresses in bulk. It interacts with the server by providing these addresses as the results of a query; the relational aspect eases this interaction.

Cache/Metadata:

The server has access to a metadata cache populated with country-specific address formats; this could be expanded in the future to cache other address-related data as well or popularly referenced addresses. One option for this component is a flat file which can be read in as a resource either locally or via cloud storage.

Hydrator:

The data store is hydrated via a script. This script can be run once or on a schedule to create addresses in the data store. One option for this component is to use python to process CSV or JSON files en masse to populate or append to the “known world” of postal addresses. Data cleaning or shaping could be introduced here if desired.