Big Data, organisation and analysis

Hands on data access via REST API

Steffen M. Noe, Spring 2024

REST API

REpresentational **S**tate **T**ransfer

- API = Application Programming Interface
- Allows access to data by using machine-2-machine (m2m) techniques
- Used to automatise the data access and reduce manual work like download data first, then upload to somewhere etc.
- Using the HTTP/HTTPS protocols

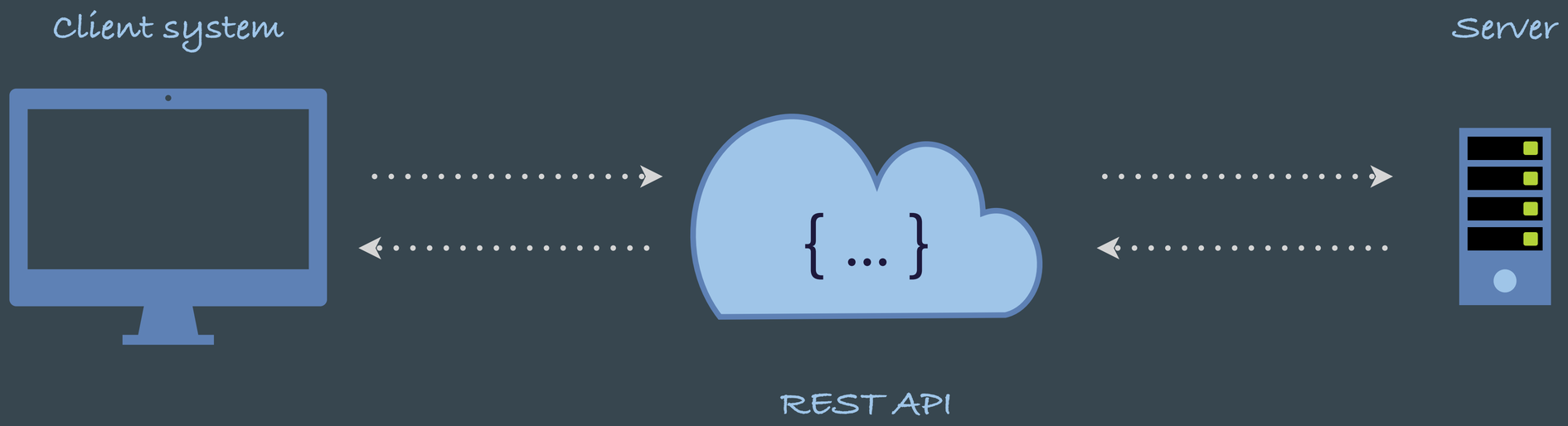
REST API

Architectural constraints

- Uniform interface — Same rules for all components to speak to each other.
- Client-server — Server stores/manipulate information. Client take information and displays it
- Stateless — Communication contains all information to process the request. Client need to authenticate itself.
- Cacheable — Server, client and intermediate components can cache resources.
- Layered system — Each component just see the other components it directly links with, no beyond sight
- Code on demand — Optionally, a component can download code on demand to extent its capacities.

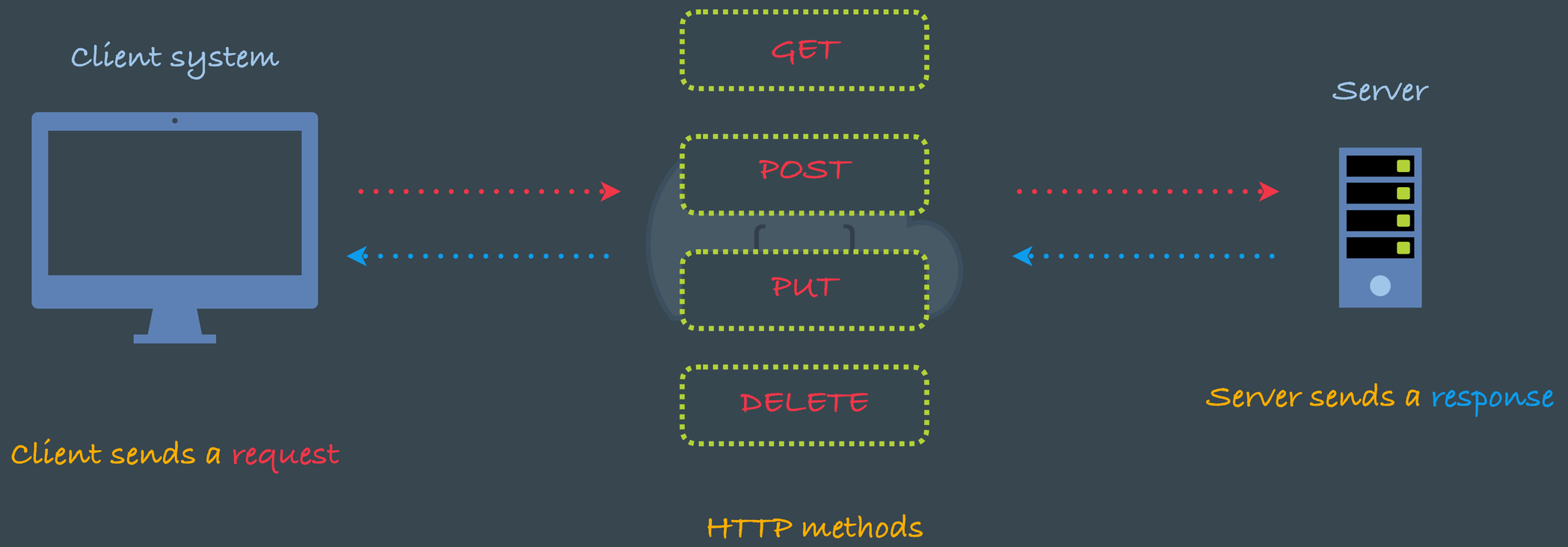
REST API

Principle



REST API

Principle



Python request package

- The request package handles the HTTPS requests in an easy way
- We can use the package to send the REST API's methods to a server
- We can use it to get information on the response
- We can get information on the type of data in the response
- The package can be used to "scrape" any kind of textual data
- In combination with the io package we can decode textual data to the right format

Pandas can also operate on REST APIs

- Pandas allows us to directly read a csv from a REST API
- We need to “learn” about the file structure before we can use pandas
- This can be done by having a good metadata description for large files or by downloading once and analysing