Data Validation made easy

Jakob Voß (VZG)

ELAG 2022 Conference, Riga

2022-06-09

Outline

- **What** is data validation?
- **Why** is data validation difficult?
- **How** can data validation be made easy?
- **Where** can I use it?

What is data validation?

Detect bad data

- Eventually all data is sequences of bits

Detect bad data

- Eventually all data is sequences of bits
- Data must conform to expected shapes

Detect bad data

- Eventually all data is sequences of bits
- Data must conform to expected shapes
- **Data validation** = check expectations

Expectations

- **completeness**
  e.g. all records have year

Expectations

- **completeness**
  e.g. all records have year
- **constraints**
  e.g. year < 2022

Expectations

- **completeness**
  e.g. all records have year
- **constraints**
  e.g. year $<$ 2022
- **consistency**
  e.g. birth $<$ death (except time-travellers)

Data ≠ Information

- **completeness**
  - *internal*: e.g. all authors have names
  - *external*: e.g. all authors are listed

Data quality

- code: **unit tests**
  against software rot
- data: **data validation**
  against propagation of errors

Why is data validation difficult?

Challenges

- **Big data** & **data integration**
  e.g. bibliographic data + knowledge graphs

Challenges

- **Big data** & **data integration**
  e.g. bibliographic data + knowledge graphs
- **Many formats** & different expectations

Challenges

- **Big data** & **data integration**
  e.g. bibliographic data + knowledge graphs
- **Many formats** & different expectations
- Diverse **validation technologies**

Technologies

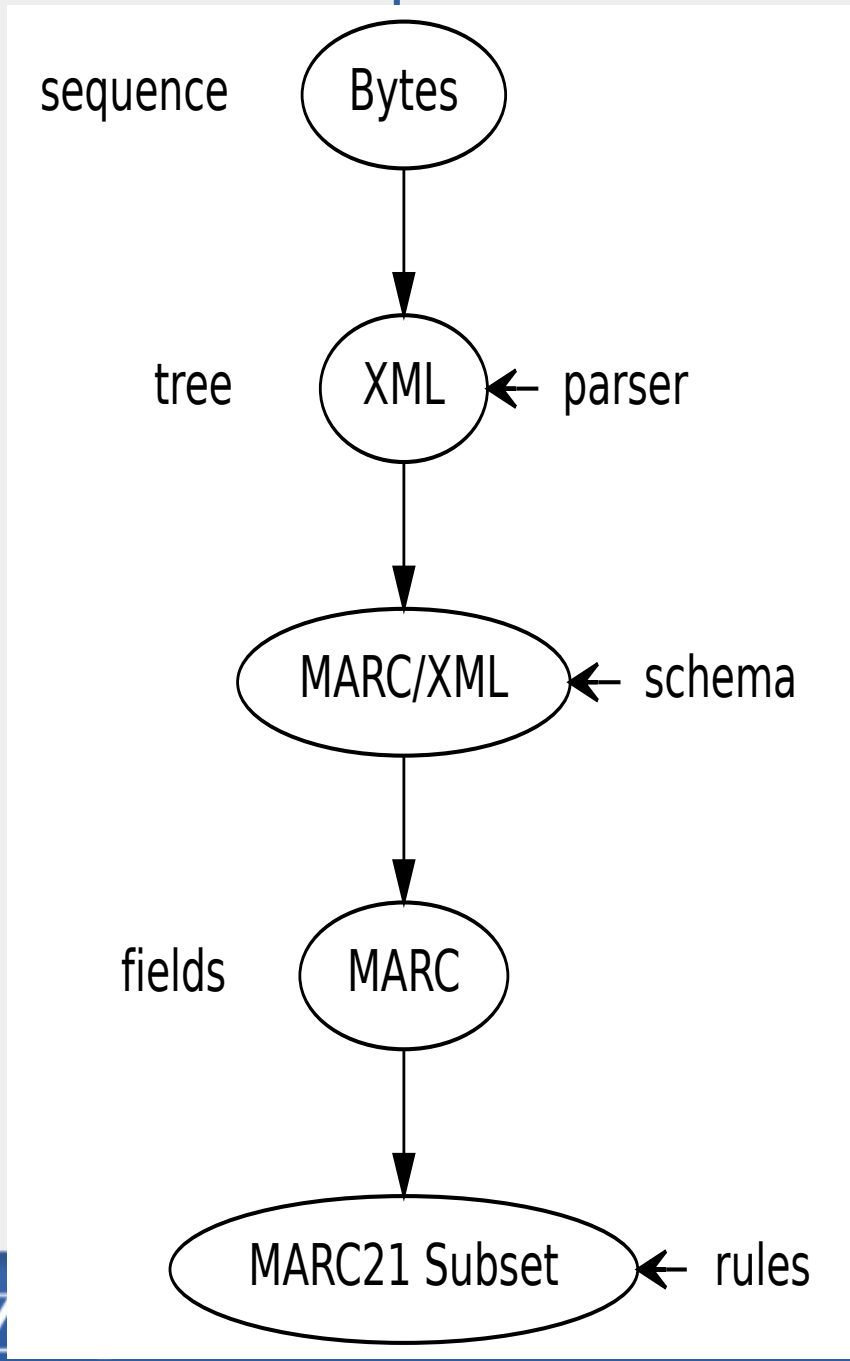
- Custom **parser/rules** (*if ... then ...*)

Technologies

- Custom **parser/rules** (*if ... then ...*)
- **Schema languages**

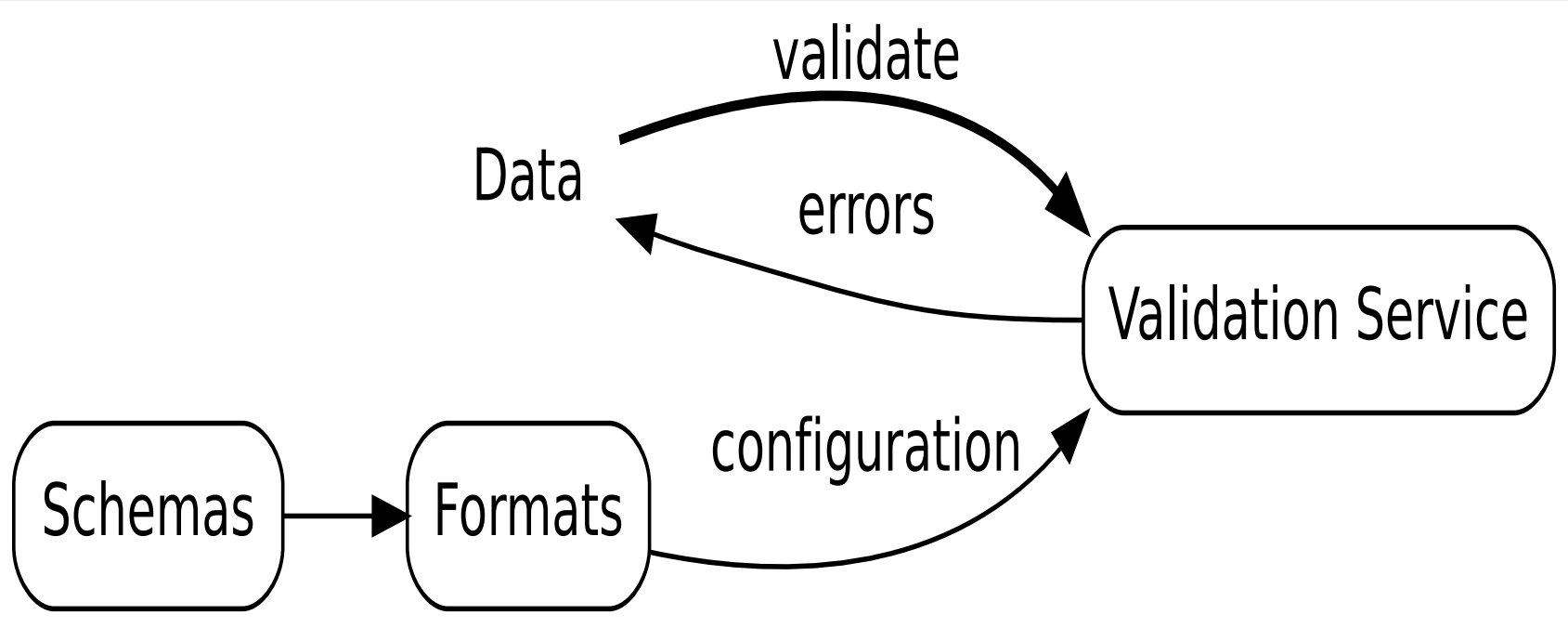
JSON	JSON Schema
XML	XSD, DTD, Schematron
RDF	SHACL/ShEx
String	RegEx, EBNF
MARC	Avram

Levels of description



How to make data validation easy

Abstraction



API

Request	`data` (file, URL, file or stream) and `format` identifier (+ optional version)
Response	list of errors
Error	`message` (+ optional positions)

Where can I use it?

Validation Server

- **Web service** to validate data
- Configured with formats and schemas
- github.com/gbv/validation-server (Node)
- format.gbv.de/validate

Request API

- HTTP GET & POST

Request API

- HTTP GET & POST
  - raw data or web form file upload

Request API

- HTTP GET & POST
  - raw data or web form file upload
  - Use in any web application (CORS)

Request API

- HTTP GET & POST
  - raw data or web form file upload
  - Use in any web application (CORS)
- Web interface

Request API

- HTTP GET & POST
  - raw data or web form file upload
  - Use in any web application (CORS)
- Web interface
- Command line (requires configuration)

Demo

`https://format.gbv.de/validate`

Example

```
curl https://format.gbv.de/validate/vzg-article \
  --data-binary @article.json
```

```
[
  {
    "message": "must be array",
    "position": {
      "jsonpointer": "/authors"
    }
  }
]
```

Benefits

- Registry of known formats and schemas
- No local installation required
- Unified API

Summary & Outlook

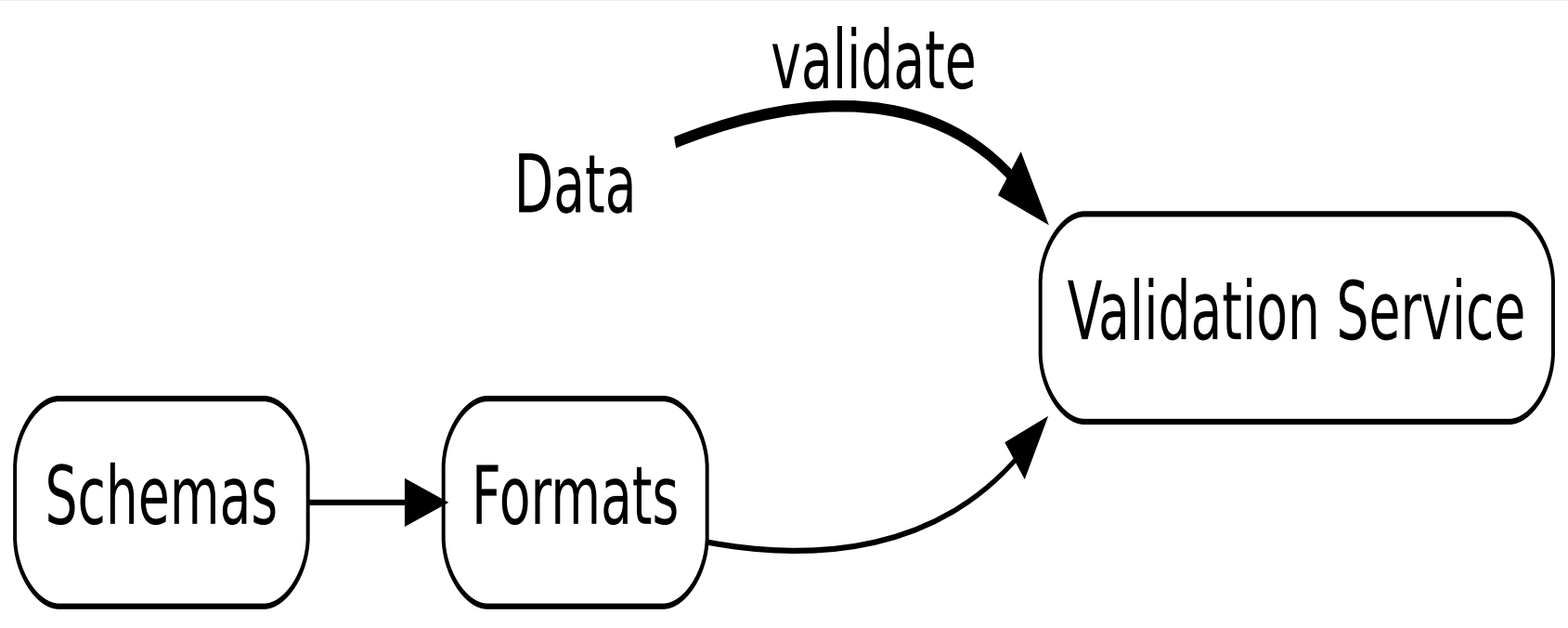
Data quality paradigms

- **Authority based**
  done by experts is good by definition

Data quality paradigms

- **Authority based**
  done by experts is good by definition
- **Evidence based**
  continuous measuring & improving

Abstraction



Implementation

- Validation Server
- Configure formats with schemas
- Public instance format.gbv.de/validate

Planned features

- Support more schema languages
  (Avram, EBNF, Schematron SHACL/ShEx...)
- Support validating MARC21
- Show error context

Alternatives

- Build-in rules of black-box library system
- Validator engines for each schema language (e.g. `xmllint`)
- Metadata Quality Assurance Framework