Data management for heterogeneous research environments with CaosDB: Experiences from an MPDL Open Source development project

Daniel Hornung[1] Florian Spreckelsen[1] Freja Nordsiek[2]

[1]IndiScale GmbH, Göttingen

[2]Max Planck Institute for Dynamics and Self-Organization, Göttingen

2022-05-13

Caosdb

an open scientific database

Intro: structured high-volume data from the clouds

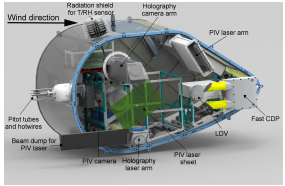
Intro: structured high-volume data from the clouds

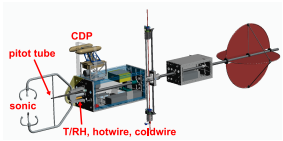
MAX PLANCK INSTITUTE FOR DYNAMICS AND SELF-ORGANIZATION

Fluid Physics Laboratory for Pattern Formation Biocomplexity LFPB

Max-Planck CloudKites (MPCK)

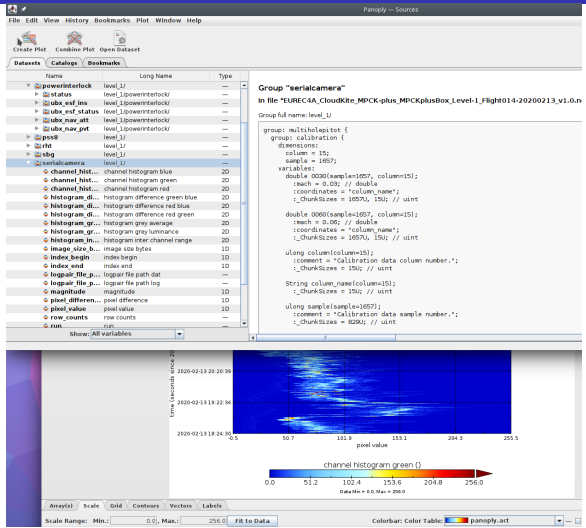
- Balloon-borne atmospheric instruments
- Measure atmospheric turbulence and cloud droplets *in situ*
- Data consists of videos and timeseries from separate instruments
- 100 MiB – 5 TiB of raw data per flight
- Time consuming to manually go through all the data and process all of it
- Important to find flights and parts of flights of interest to focus attention resources





Intro: structured high-volume data from the clouds

- Data structured in HDF5 files
- Multiple processing stages
- Analyzed by different researchers



Intro: structured high-volume data from the clouds

- Data structured in HDF5 files
- Multiple processing stages
- Analyzed by different researchers

Intro: structured high-volume data from the clouds

- Data structured in HDF5 files
- Multiple processing stages
- Analyzed by different researchers

Intro: structured high-volume data from the clouds

- Data structured in HDF5 files
- Multiple processing stages
- Analyzed by different researchers

Solution:

MPDL (Max Planck Digital Library) Open-Source development.

→ Enhance the CaosDB toolkit to handle these and many other use cases.

History

- CaosDB started at MPI-DS around 2011
- Running stable since ca. 2016, released as open-source (AGPLv3)[1] in 2018
- Increasing adoption since 2020
- Commercial support by IndiScale GmbH
  - distribution branded as LinkAhead
  - DH, FS work at IndiScale

Caosdb

an open scientific database

History

- CaosDB started at MPI-DS around 2011
- Running stable since ca. 2016, released as open-source (AGPLv3)[1] in 2018
- Increasing adoption since 2020
- Commercial support by IndiScale GmbH
  - distribution branded as LinkAhead
  - DH, FS work at IndiScale

Caosdb
an open scientific database

GitLab

[1] https://gitlab.com/caosdb

History

- CaosDB started at MPI-DS around 2011
- Running stable since ca. 2016, released as open-source (AGPLv3)[1] in 2018
- Increasing adoption since 2020
- Commercial support by IndiScale GmbH
  - distribution branded as LinkAhead
  - DH, FS work at IndiScale

[1] https://gitlab.com/caosdb

What is CaosDB?

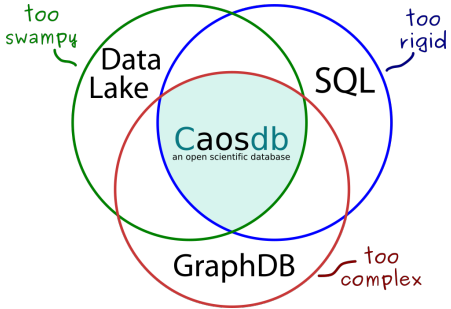
CaosDB: Software framework for agile, semantic data management

- Typed references between entities → modeling of semantic context, reproducible science
- CaosDB stores raw data as reference → ideal for huge data files
- Powerful query language → analysis right in the database
- Flexible data model → no migration necessary when modifying data structures

What is CaosDB?

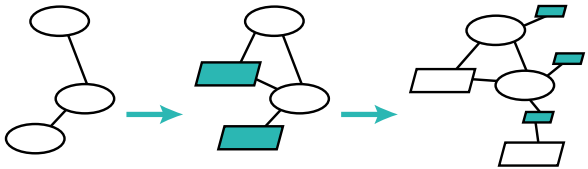
CaosDB: Software framework for agile, semantic data management

- Typed references between entities → modeling of semantic context, reproducible science
- CaosDB stores raw data as reference → ideal for huge data files
- Powerful query language → analysis right in the database
- Flexible data model → no migration necessary when modifying data structures



Flexibility matters

- Research questions evolve
- Experiment setup, data acquisition, collaborators change
- SQL-like databases cost time to adapt



Agile data management with CaosDB

Flexibility matters

- Research questions evolve
- Experiment setup, data acquisition, collaborators change
- SQL-like databases cost time to adapt

CaosDB: change the data structure, while using old and new data side-by-side.

Data model
ca. 2018

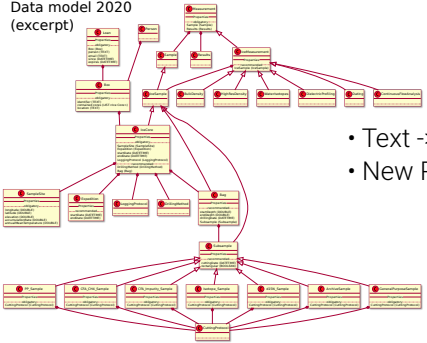
Flexibility matters

- Research questions evolve
- Experiment setup, data acquisition, collaborators change
- SQL-like databases cost time to adapt

CaosDB: change the data structure, while using old and new data side-by-side.

Data model ca. 2018

Data model 2020 (excerpt)

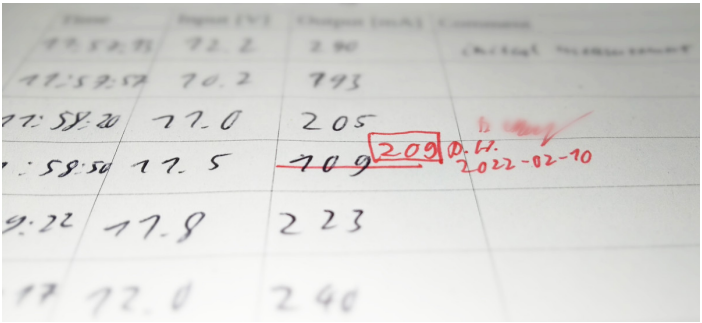


- Text -> object entities
- New RecordTypes

Versioning of data

Question

- How to correct mistakes?
  - Compliance with good scientific practice rules



Versioning of data

Question

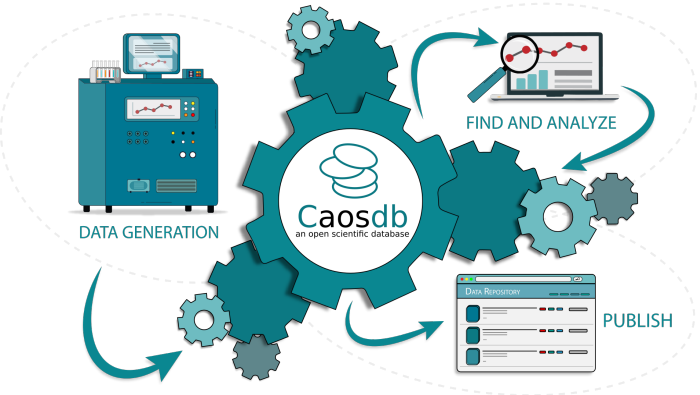
- How to correct mistakes?
  - Compliance with good scientific practice rules
- Reconstruct earlier states

- CaosDB keeps track of the version history:
  - Who?
  - When?
  - What was changed?
- For example:
  Who borrowed a sample? How much of the sample was used up in the process?

CaosDB: Integration of structured data

Question

How do I get data into my system?
How can everyone follow FAIR data principles?
How do I get my data out of the system for publication?



CaosDB: Integration of structured data

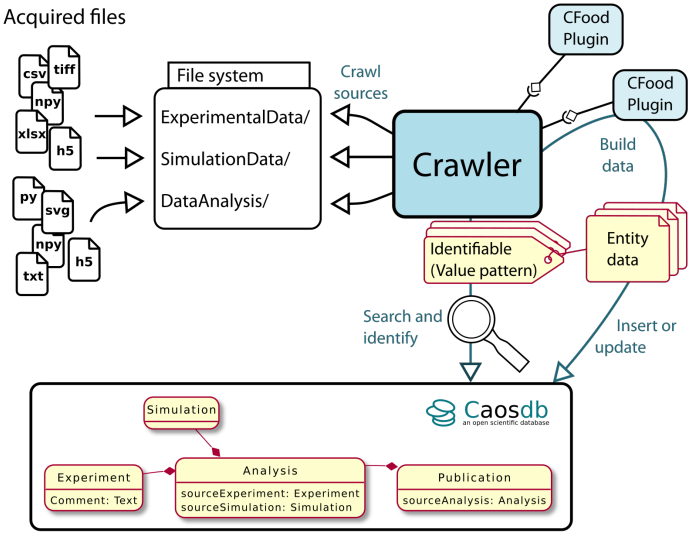
Question

How do I get data into my system?
How can everyone follow FAIR data principles?
How do I get my data out of the system for publication?

CaosDB Crawler:
automatic parsing and integration

NEW → Generic HDF5 framework

Diagram: Based upon work by Alexander Schlemmer



Integrating CaosDB into the research: REST and gRPC APIs, language bindings

Integrating CaosDB into the research: REST and gRPC APIs, language bindings

- Two APIs:
  - **REST**/XML-over-HTTP, docs: https://docs.indiscale.com/caosdb-server/specification
  - **gRPC**/protobuf, docs: https://docs.indiscale.com/caosdb-proto **NEW**
- Powerful query language [1] → analysis right in the database
- REST clients: Javascript / Python [2]
- Client libraries based upon the gRPC interface (Code at GitLab):
  - C/C++ [3] **NEW**
  - Julia [4] **NEW**
  - Octave / Matlab [5] **NEW**
  - JavaScript (under development)
  - R (under development)

[1] https://docs.indiscale.com/caosdb-server/CaosDB-Query-Language
[2] https://gitlab.com/caosdb/caosdb-pylib
[3] https://gitlab.com/caosdb/caosdb-cpplib
[4] https://gitlab.com/caosdb/caosdb-julialib
[5] https://gitlab.com/caosdb/caosdb-octavelib

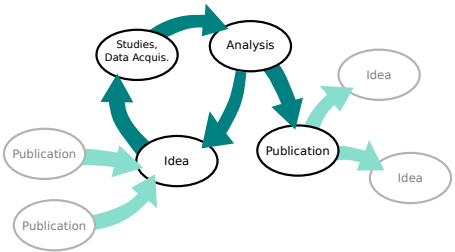
Soft factors matter: Make software usable

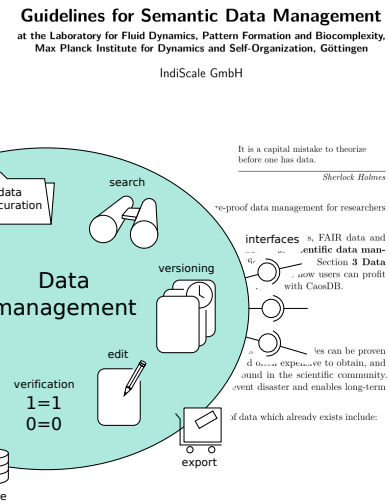
- Overhauled CaosDB documentation: https://docs.indiscale.com/
- Tutorials: https://docs.indiscale.com/caosdb-pylib/tutorials/
- Workshops on different technical levels
- Data management guidelines for researchers

Soft factors matter: Make software usable

- Overhauled CaosDB documentation: https://docs.indiscale.com/
- Tutorials: https://docs.indiscale.com/caosdb-pylib/tutorials/
- Workshops on different technical levels
- Data management guidelines for researchers

  How do I *live* the data management plan?
  → to be published as a best practices paper





Try it out!

Useful links

Live Demo demo.indiscale.com

Code gitlab.com/caosdb

Docs docs.indiscale.com

Caosdb

an open scientific database

Max Planck Institute for
Dynamics and Self-Organization

IndiScale
Data Services

Appendix

- Comparison to SPARQL (RDF query language for e.g. WikiData)
- Data model evolution (at the AWI)
- CaosDB Crawler

Comparison to SPARQL

SPARQL

```
SELECT DISTINCT ?item ?itemLabel ?givenName ?familyName WHERE {
    ?item wdt:P31 wd:Q5;  # Any instance of a human.
          wdt:P27 wd:Q145; # United Kingdom
          wdt:P21 wd:Q6581072; # female
          wdt:P106 wd:Q36180; # writer
          wdt:P569 ?birthday;
          wdt:P570 ?diedon;
          wdt:P734 [rdfs:label ?familyName];
          wdt:P735 [rdfs:label ?givenName].
  FILTER(?birthday > "1870-01-01"^^xsd:dateTime
       && ?diedon < "1950-01-01"^^xsd:dateTime)
  FILTER(regex(?givenName, "M.*") || regex(?familyName, "M.*"))
    SERVICE wikibase:label { bd:serviceParam wikibase:language "en" }
  }
```

Comparison to SPARQL

SPARQL

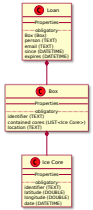
```
SELECT DISTINCT ?item ?itemLabel ?givenName ?familyName WHERE {
    ?item wdt:P31 wd:Q5;  # Any instance of a human.
          wdt:P27 wd:Q145; # United Kingdom
          wdt:P21 wd:Q6581072; # female
          wdt:P106 wd:Q36180; # writer
          wdt:P569 ?birthday;
          wdt:P570 ?diedon;
          wdt:P734 [rdfs:label ?familyName];
          wdt:P735 [rdfs:label ?givenName].
  FILTER(?birthday > "1870-01-01"^^xsd:dateTime
       && ?diedon < "1950-01-01"^^xsd:dateTime)
  FILTER(regex(?givenName, "M.*") || regex(?familyName, "M.*"))
    SERVICE wikibase:label { bd:serviceParam wikibase:language "en" }
  }
```

CaosDB query language

```
SELECT given_name, family_name FROM Writer
  WITH gender=f AND country=UK AND birthday > 1870 AND death < 1950
       AND (given_name LIKE "M*" OR family_name LIKE "M*")
```

Data model evolution

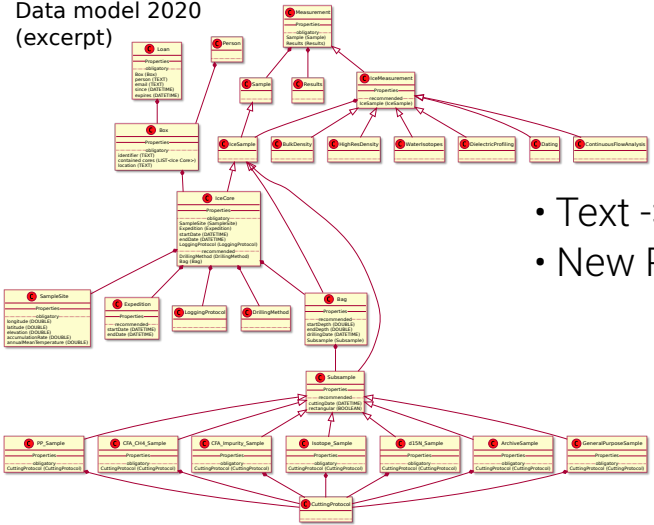
Data model
ca. 2018



Data model evolution

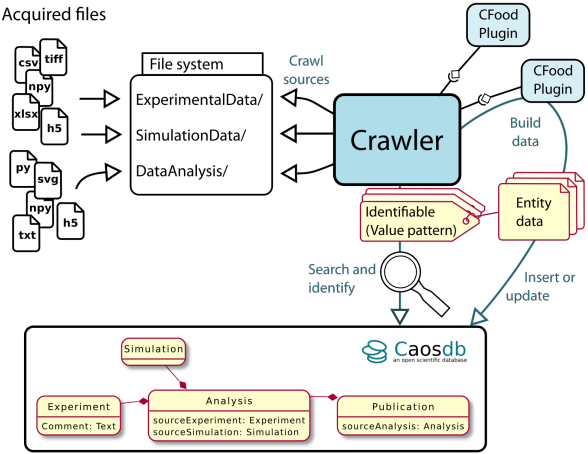
Data model ca. 2018

Data model 2020 (excerpt)



- Text -> object entities
- New RecordTypes

Automated crawler



1. Look for potential data to insert.
2. Identify existing, matching data.
3. Insert or update data as necessary.
4. (Notify administrators if necessary.)

Diagram: Based upon work by Alexander Schlemmer