C3Grid

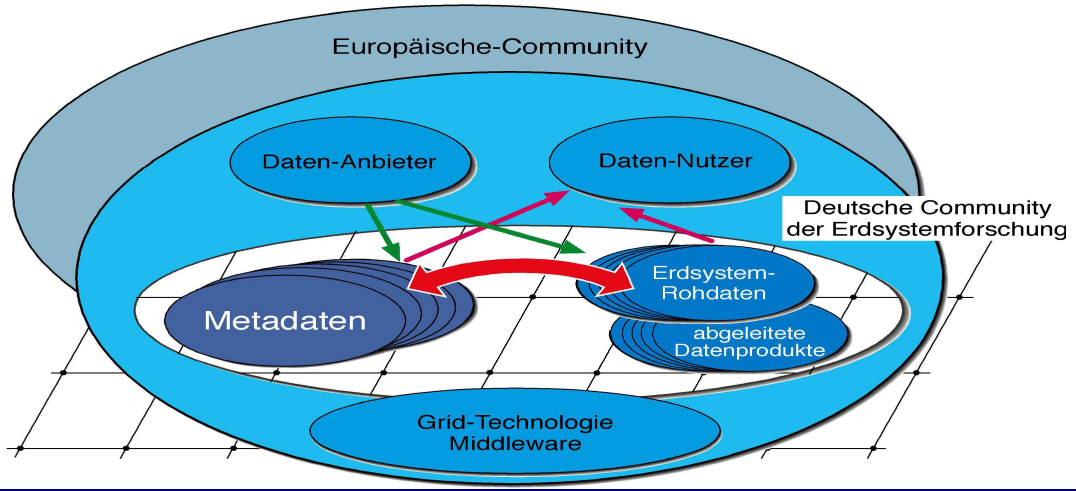
Data and Information Management Based on Apache Lucene

Tobias Langhammer, Zuse Institut Berlin (ZIB)

C3-Grid

Collaborative Climate Community Grid:

- Grid environment for German climate and earth system research
- access on distributed data resources of different providers
- remote data processing and scientific analysis
- exchange of data between institutes
- transfer over high bandwidth networks


Europäische-Community
Daten-Anbieter
Daten-Nutzer
Deutsche Community der Erdsystemforschung
Metadaten
Erdsystem-Rohdaten
abgeleitete Datenprodukte
Grid-Technologie Middleware

Requirements

Special Requirements

- high volume model and observation data handling
- distributed post-processing
- community-specific metadata

Consequences

- avoid transfers by replica management and scheduling
- local processing and analysis of data close to where it is stored
- prediction of future storage and compute resources to guide scheduling
- community-specific metadata schema and management

Data in C3-Grid

What?

- historical data in archives
- results of simulations on supercomputers
- using differing data formats

Where?

- stored in 53 World Data Centers (WDCs) with different focus
- local institutes

How?

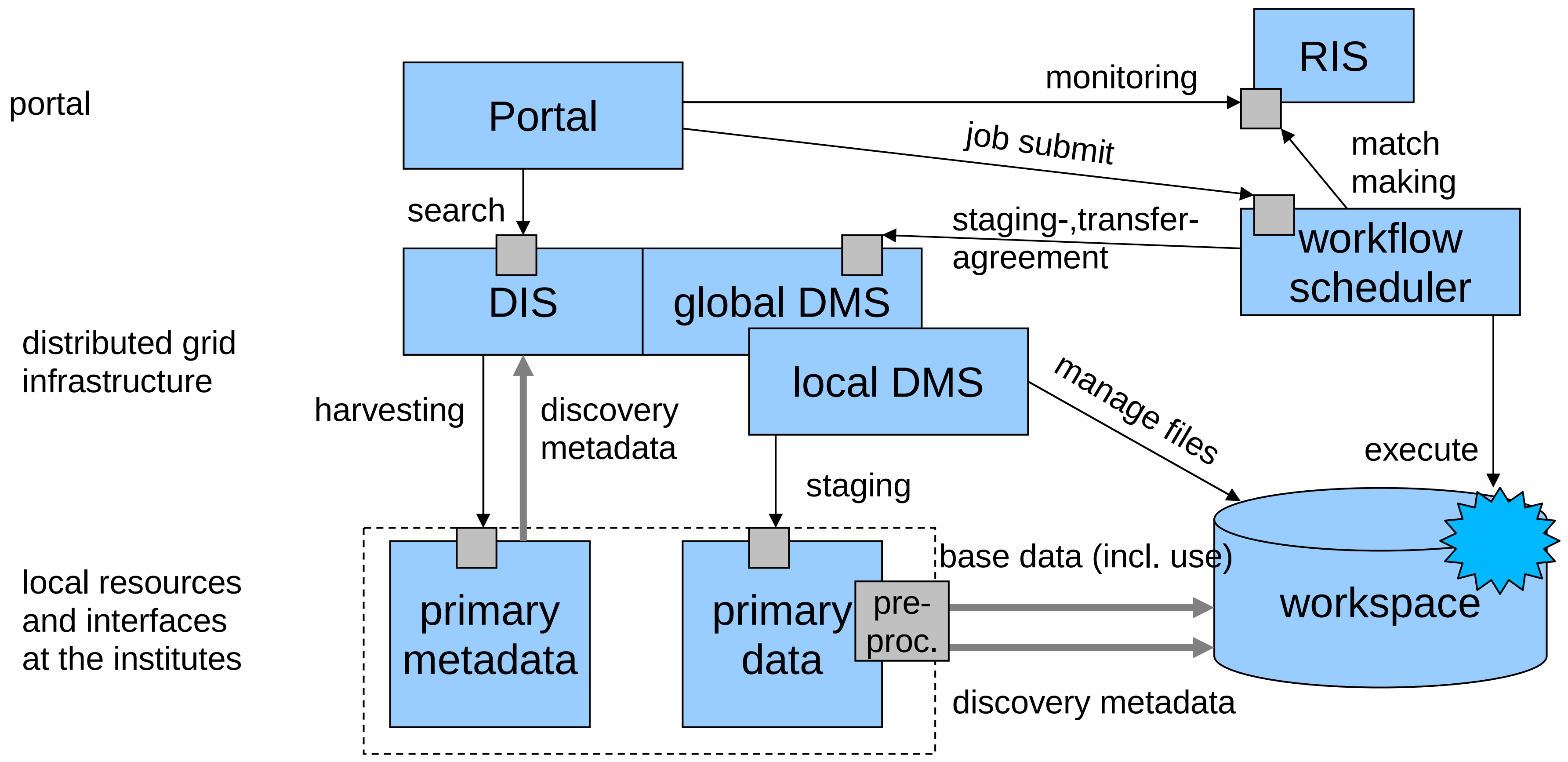
- Example in Germany: WDC-Climate (MPI for Meteorology and DKRZ in Hamburg)
  - September 2005: 220 Terabyte in a database running with Linux
  - 6 DBMS instances: 1 DB metadata, 5 DBs base data, tapes with disk cache

Metadata in C3-Grid

type of metadata	describes what?	is needed for what?
resource metadata	compute and storage resources	match making (job and data placement) monitoring (ensure the functionality)
discovery metadata	data objects	scientific description (with ISO 19115 / 19139) search
use metadata	data objekts and files	access on data
replica information	log./phys. files	replica management, agreements with scheduler

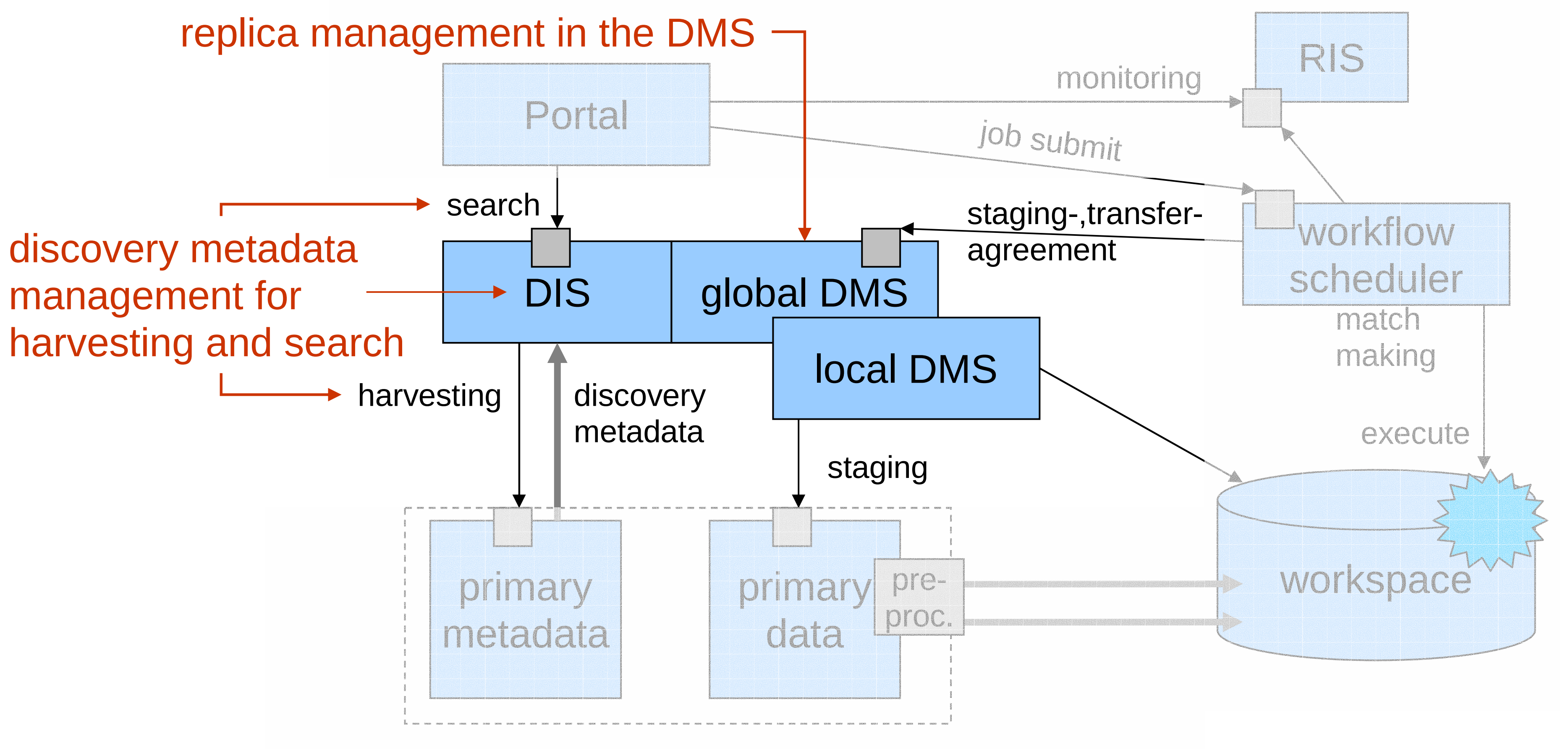
rem.: discovery metadata $\cap$ use metadata $\neq \emptyset$

C3-Grid Services (1)



RIS: Resource Information Service
DIS: Data Information Service
DMS: Data Management Service

C3-Grid Services (2)



RIS: Resource Information Service
DIS: Data Information Service
DMS: Data Management Service

Apache Lucene

- ...is a full-text search engine library

  I.e., it maps each word to the documents it appears in.
- ...is 100% Java (requires no database backend!)
- ...provides *fields*, which are similar to attributes in relational DBs
- ...provides a Google-style query parser
- ...provides the ranking of results
- ...has indices which are about 20-30% of the indexed text
- ...has a token-analyser for many languages EN, DE, FR,...

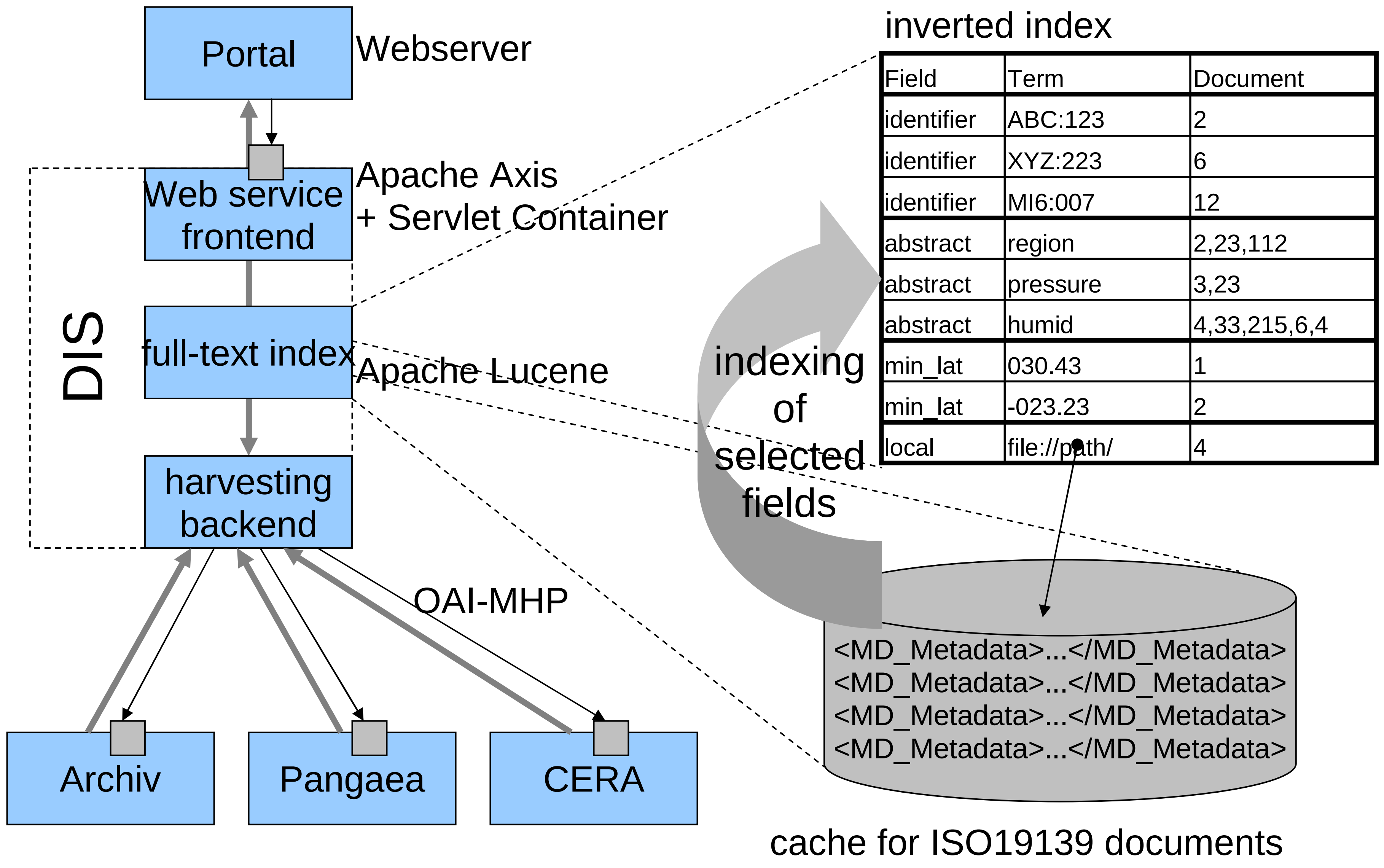
Apache Lucene - Terms

- document: sequence of fields
- field: sequence of terms
- term: a string

  two different strings in different fields are different terms
- inverted index:

  maps a term to a document list containing this term
- optional storing of non-inverted fields
- indexes can be build on a complete field or on individual tokens

Data Information Service with Lucene


Portal
Webserver
Web service frontend
Apache Axis + Servlet Container
DIS
full-text index
Apache Lucene
harvesting backend
OAI-MHP
Archiv
Pangaea
CERA
indexing of selected fields
inverted index
<MD_Metadata>...</MD_Metadata>
<MD_Metadata>...</MD_Metadata>
<MD_Metadata>...</MD_Metadata>
<MD_Metadata>...</MD_Metadata>
cache for ISO19139 documents

Field	Term	Document
identifier	ABC:123	2
identifier	XYZ:223	6
identifier	MI6:007	12
abstract	region	2,23,112
abstract	pressure	3,23
abstract	humid	4,33,215,6,4
min_lat	030.43	1
min_lat	-023.23	2
local	file://path/	4

Search in Lucene

Requirements for the C3-Grid

1. full-text search
   - keywords, authors,...
   - this is what Lucene is for
2. range search
   - space (floating or fixed point number)
     - geographic latitude,
     - geographic longitude,
     - height level,
   - time (date or time stamp)
   - this is what Lucene is **not** good in

     … next slide

Range Search in Lucene (1)

Problem 1: Lucene uses only strings

Solution: find a mapping to strings which preserves the order

	Empire State Building		ZIB
latitude	**73°59'09" W**	$<$	**13°17'52" E**
floating point	-73.986	$<$	13.298
32 bit integer	-73986	$<$	52456
unsigned int (add 2^{31}):	2147409662	$<$	2147536104
hex-dump	**"7FFECEFE"**	$<_{lex}$	**"8000CCE8"**

Range Search in Lucene (2)

Problem 2: The range search is inefficient or limited

- Standard RangeQuery:
  - expands **["7FFECEFE" .. "8000CCE8"]** to disjunction
    (match "7FFECEFE" OR match "7FFECEFFF" OR match "7FFECF000" OR ... OR match "8000CCE8")
  - bad: complexity grows with the size of the query and not with the number of matches!
- ConstantScoreRangeQuery:
  - linear traversal of the index
  - good: complexity grows with the number of matches and not with the size of the query!
  - bad: no ranking possible

Range Search in Lucene (3)

- hybrid solution:
  - each document has range fields for different resolutions

    lat1= "7F", lat2="7FFE", lat3="7FFED2", lat4="7FFED2A0"
  - split range to do several range queries

    match ["7FFECEFE" .. "8000CCE8"] in latitude

    → match ["77FFCEFE" .. "77FFCEFF"] in lat4,

    match ["77FFCF" .. "77FFFF"] in lat3,

    match [] in lat2,

    match ["78","79"] in lat1,

    match [] in lat2,

    match ["800000".."8000CB"] in lat3,

    match ["8000CC00".."8000CCE8"] in lat4
  - 7 ranges with max. 256 Elements: disjunction of max. length 1792
  - compared to plain RangeQuery: disjunction of length 2^{32}=4294967296

Conclusions

- Lucene is suitable for data-specific metadata in the C3-Grid
  - full-text and range search on discovery metadata
  - also for replica information in the data management system
- easy deployment, no DB backend required
- big user community
- first tests on indices up to 11 GB are promising

Thank you!