

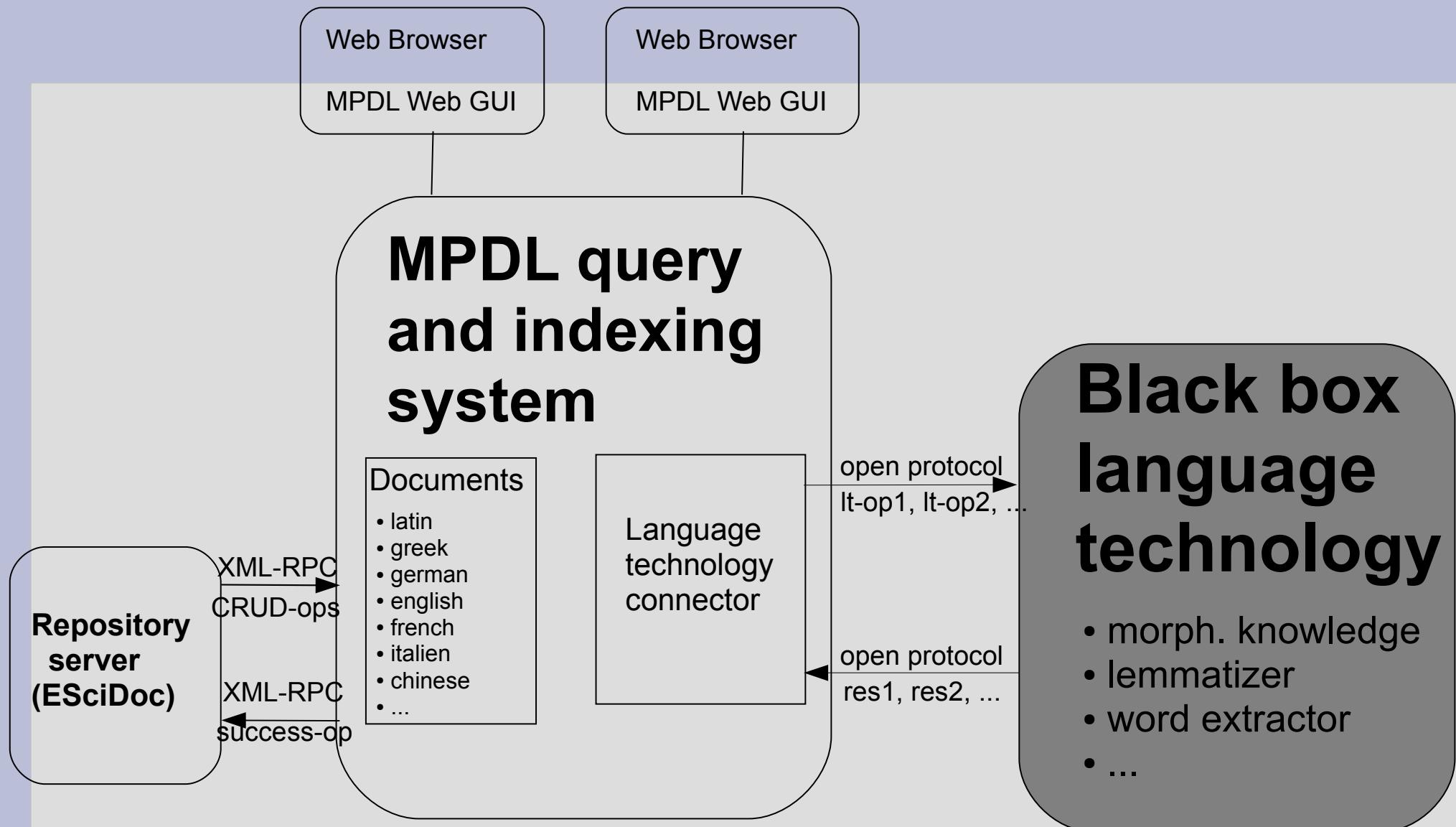
# eXist/Lucene language technology latest developments

## Content

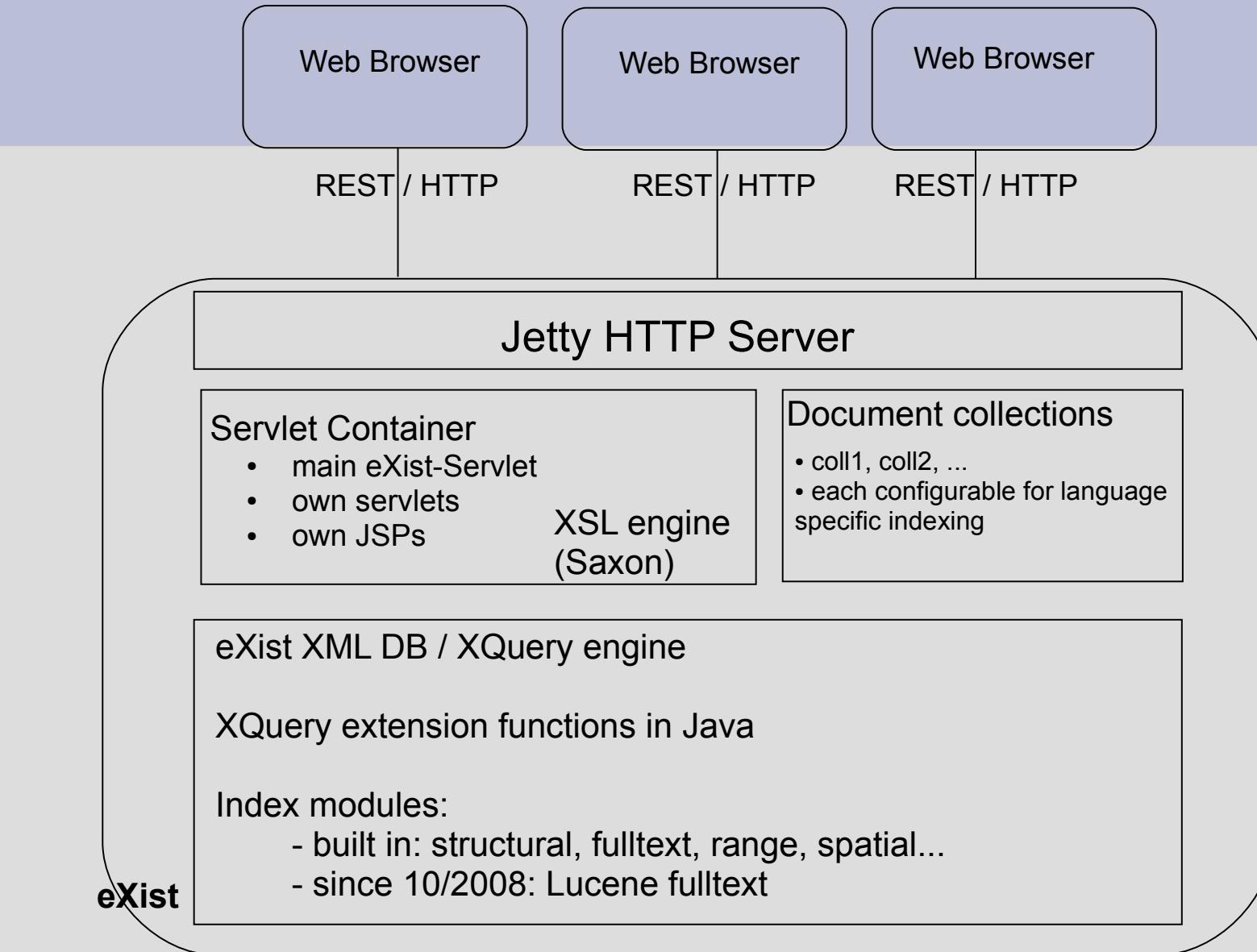
1. Indexing in eXist by Lucene language technology
2. Querying in eXist by Lucene language technology
3. eXist/Lucene function: index terms
4. MPDL with Donatus language technology
5. Cooperation ideas

# MPDL

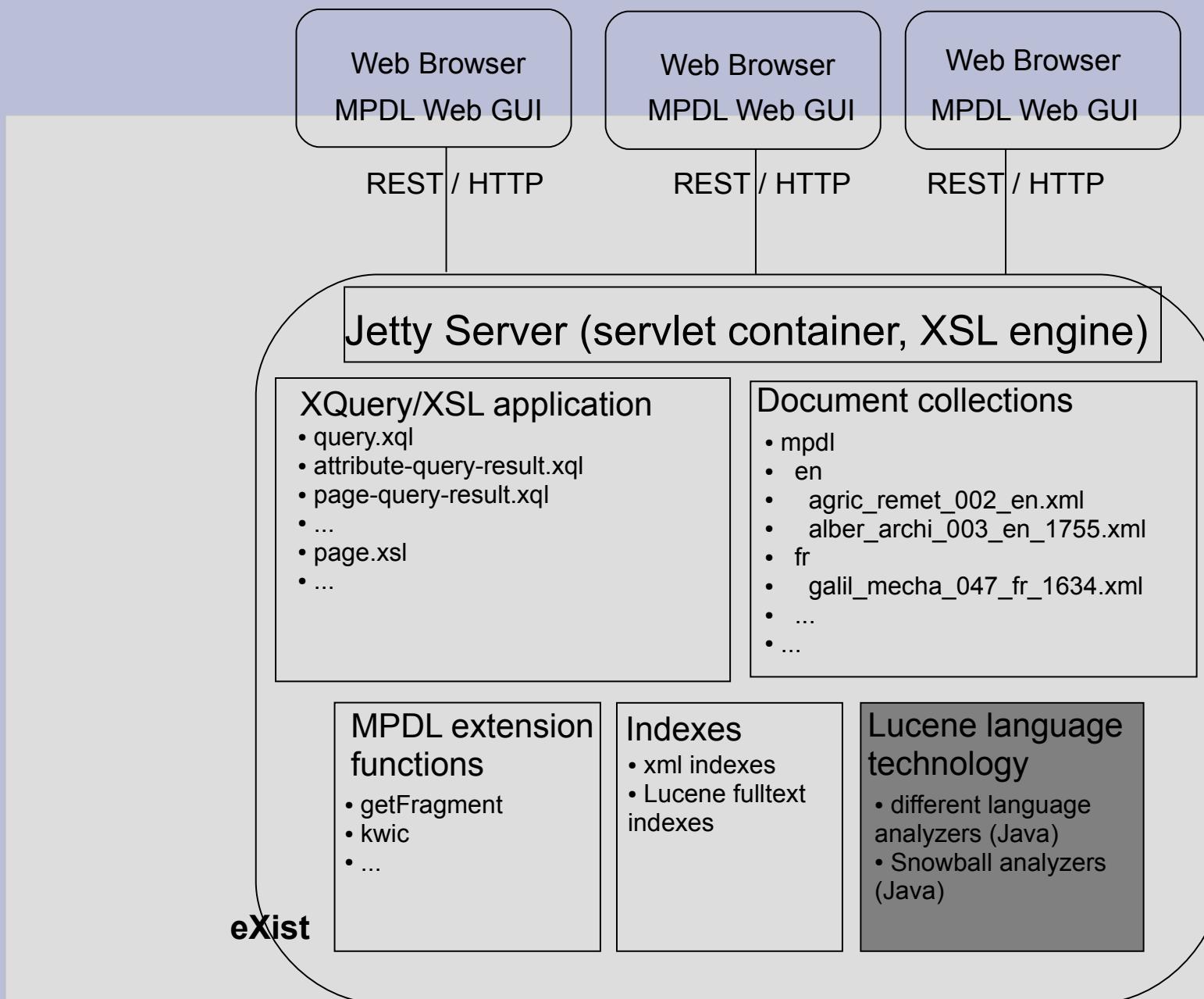
## Black box: language technology



# The XML based query and indexing system eXist



# MPDL with Lucene language technology



# Lucene lemmatizer

- Lucene is part of eXist since 10/2008 (in its working release)
- next plans in eXist: stable release of eXist with Lucene, better integration
- language specific lemmatizer: free (Java), add-on to Lucene, only simple
- first use in archimedes test collection
  - language specific subcollections of documents (with its own configuration)
  - used: en, de, fr, el, zh, cjk, nl
  - not yet used: Thai, Brazilian, cz, ru, all Snowball analyzers

**Example: german lemmatizer (Java method, based on: „A fast and simple stemming algorithm for german words“ by joerg.caumanns@isst.fhg.de)**

1. use of small letters
2. some words are not handled (stopwords, etc.)
3. cut regular suffixes at the end of the word
  - nd, em, er
  - e, s, n, t
4. cut female form at the end of the word
  - erin (e.g. Schneiderin --> Schneider)
5. resolve irregular suffixes
  - z --> x (e.g. Matrizen --> Matrix)
6. remove ge-particle
  - gege --> ge (e.g. gegeben --> geben)
7. resolve umlauts and other special characters
  - ä, ö, ü, ß
  - sch, ch, ei, ig, st --> \$, §, %, &, #, !

# Indexing documents with eXist and the Lucene lemmatizer

1. add document to an eXist document collection
2. get Lucene analyzer class for that document
  - defined in an .xconf-file in a document collection
  - language specific: different Java analyzer classes:  
Example:  

```
<lucene>
<analyzer class = "org.apache.lucene.analysis.de.GermanAnalyzer"/>
<text match="//text///*"/>
</lucene>
```
3. analyzer class lemmatizes each word in each text node or text attribute
  - e.g. class GermanAnalyzer
  - method „stem(String word)“ --> lemmatized word
4. add index entries: for each lemmatized word: at key add value
  - key: lemmatized word: e.g. professor
  - value: xml documentId + xml nodeId: e.g. 137 3.6.7.4/1

# Querying with eXist and the Lucene lemmatizer

1. get involved document nodes of that query in that query context
2. loop over document nodes and collect results (performance problem with many nodes ?):
  - get defined analyzer for that node (language specific)
  - parse Lucene query string: lemmatization of query terms, etc.
  - perform Lucene query and collect its result

"real" querying with Lucene language analyzers is now possible (own extension of eXist was needed for that). This means that a query is reduced to the base form and then finds all hits over the base form of the index (what we really wanted).

MPDL-Test system: German examples german (not complete of forms):

folg: folgt, folgen, folg

männ: männer, mann, männ

vollständig: vollständig, vollständige, vollständiges, vollständigen, vollständig

vollständigst: vollständigste, vollständigstes, vollständigsten, vollständigst

# Function: index terms with eXist and the Lucene language technology

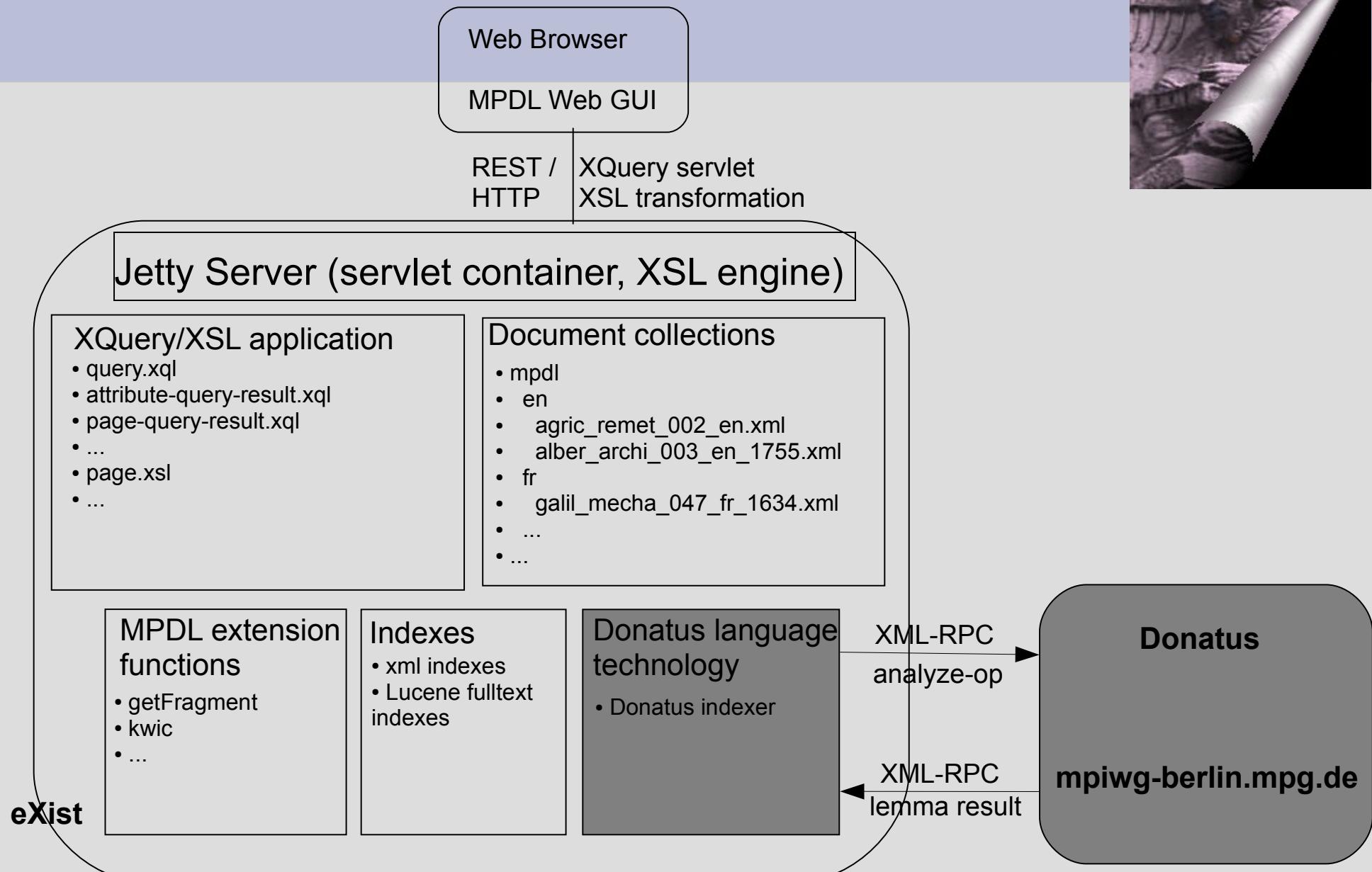
The eXist function index-terms (which we used in our web interface for browsing alphabetically through the index terms) does not work for Lucene indexes (a bug). I found the candidate method in a Java-Class, but the method is much too slow. With slight modifications I got huge improvements but there could be a limit in getting more performance because eXist with Lucene consists of millions of Lucene-documents (for each XML-node there is one Lucene document with hundreds of fields) and the Lucene methods of getting terms are then not so performant. We have to see what can be done in this function ...

1. Get index terms: All index-terms for example german document Grunert  
Needed time with builtin eXist method: 766 seconds !
2. All index-terms for sentences in example german document Grunert  
Needed time with little corrected builtin eXist method: 156 seconds
3. First 50 index-terms for sentences in example german document Grunert  
Needed time with more corrected builtin eXist method: 9 seconds
4. First 50 index-terms for sentences in example german document Grunert  
Needed time with more more corrected builtin eXist method: 3 seconds

...

# MPDL with Donatus language technology

see: <http://archimedes.fas.harvard.edu/cgi-bin/donatus>



# Indexing with eXist and the Donatus lemmatizer

With the Lucene language technology developments the basis is also done to begin with the development of a Donatus language analyzer (this needs development time and is not performant at the beginning).

# Indexing with eXist and the Donatus lemmatizer

1. add document to an eXist document collection

2. get Donatus analyzer class for that document

- defined in an .xconf-file in a document collection
- language specific: different Java analyzer classes:

Example:

```
<lucene>
<analyzer class = "de.mpg.mpiwg-berlin.mpdl.lt.DonatusGermanAnalyzer"/>
<text match="//text///*"/>
</lucene>
```

3. analyzer class lemmatizes each word in each text node or text attribute of that document

- e.g. class DonatusGermanAnalyzer
- method „stem(String word)“
  - call of DonatusHandler (only one time for performance reasons)
    - prepare the document for Donatus
    - open xml-rpc connection to Donatus Server in Berlin
    - call method „donatus.analyze“ for that document
    - receive the result (all lemmatized words) as an XML document and caches it as <lemma, variant, language> pairs
  - get each lemmatized word via the cached result of the DonatusHandler for that document

4. add index entries: for each lemmatized word: at key add value

- key: lemmatized word: e.g. professor
- value: xml documentId + xml nodeld: e.g. 137 3.6.7.4/1

# Donatus lemmatizer: example result

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE morphology [
<!ELEMENT morphology (lemma*, context-form*)>
<!ELEMENT lemma (definition?, variant*)>
<!ELEMENT context-form (tokens, analysis)>
<!ELEMENT definition (#PCDATA)>
<!ELEMENT variant (analysis)*>
<!ELEMENT analysis EMPTY>
<!ELEMENT tokens (token+)>
<!ELEMENT token EMPTY>
(...)
<morphology xmlns="http://archimedes.fas.harvard.edu/ns/morphology/3"
xmlns:xlink="http://www.w3.org/1999/xlink">
    <lemma form="Aristoteles" lang="la">
        <variant form="Aristotelem">
            <analysis desc="N masc acc sg" xlink:type="simple"/>
        </variant>
        <variant form="Aristoteles">
            <analysis desc="N masc acc pl" xlink:type="simple"/>
            <analysis desc="N masc nom/voc pl" xlink:type="simple"/>
            <analysis desc="N masc nom sg" xlink:type="simple"/>
        </variant>
        <variant form="Aristoteli">
            <analysis desc="N masc dat sg" xlink:type="simple"/>
        </variant>
        <variant form="Aristotelis">
            <analysis desc="N masc gen sg" xlink:type="simple"/>
        </variant>
    </lemma>
    <lemma form="Aristotelius" lang="la">
        <variant form="Aristoteli">
            <analysis desc="N masc/neut gen sg" xlink:type="simple"/>
        </variant>
    </lemma>
(...
</morphology>
```