openlaws

Deliverable 2.2.d3

Final specification and vision of enrichment tools

Co-financed by the European Commission (DG Justice)
under action grant JUST/2013/JCIV/AG

(April 2014 – March 2016)

JUST/2013/ACTION GRANTS

Grant Agreement Number 4562

Project Start date:	01.04.2014
Project End date:	31.03.2016

Deliverable 2.2.d3 – Final Specification and Vision of Enrichment Tools

Version:	1.0

Document prepared by:	UNIVERSITEIT VAN AMSTERDAM Alexander Boer, Maarten Trompper, Radboud Winkels
Contributors:	Eduard Hirsch, Thomas Heistracher

Deliverable due date:	31.3.2016
Deliverable actual date:	30.4.2016

co-funded by the Civil Justice Programme

of the European Union

Document History

Date	Revision	Comments
9.2.2016	0.1	creation
13.4.2016	0.2	Table of Contents
19.4.2016	0.5	Google docs conversion
30.4.2016	1.0	Added lots of sections, finalisation

Document Authors

Alexander Boer, Maarten Trommper, Radboud Winkels, Eduard Hirsch, Thomas Heistracher.

Participant List

	Short Name	**Organisation Name**	**Country**
1	UVA	Universiteit van Amsterdam	NL
2	SUSS	University of Sussex	GB
3	LSE	London School of Economics and Political Science	GB
4	ALP	Alpenite srl	IT
5	SUAS	Fachhochschule Salzburg GmbH	AT
6	BYW	BY WASS GmbH	AT

Executive Summary

This report describes the final specification of the OpenLaws.eu enrichment tools. It is a far more elaborate description than D2.2.d2 – Initial Specification. Not everything described in this report is part of the final OpenLaws.eu portal. Where it is, this is explicitly mentioned.

The report describes BOLD objects and networks, the peculiarities of legal documents and data and then ways to analyse and visualize these, providing additional metadata (‘enrichment’), either by humans (‘crowd sourcing’) or automatic means.

1 Table of Contents

Table of Contents

1 Introduction

Openlaws.eu aims to initiate a platform and develop a vision for Big Open Legal Data (BOLD): an open framework for legislation, case law, and legal literature from across Europe. Based on open data, open source software and open innovation principles we are adding a *social layer* and a *meta*-layer to existing *legal information* systems. This document follows up on report 2.2.d1 (requirements of enrichment tools) and 2.2.d2 (initial specification of enrichment tools), and specifies the final version of BOLD enrichment tools and vision.

This document primarily follows the familiar model-view-controller paradigm for interactive aspects of the enrichment tools, separating:

1. *Models* of BOLD objects
2. *Views* on BOLD object models
3. *Controllers* of BOLD object models, and
4. *Pipelines* related to BOLD object models.

Section 2 starts with a specification of models of the objects that make up a Big Open Legal Data (BOLD) framework, in the present document called the *BOLD objects*, addressing both big open legal data and the envisioned social networks that will keep the process of enrichment going.
Having introduced these, several sections follow that specify model enrichment processing pipelines, and views and controllers for the BOLD models identified.

2 Models of BOLD objects and BOLD networks

This section starts with a specification of models of the objects that make up a Big Open Legal Data (BOLD) framework, in the present document called the *BOLD objects*, addressing both big open legal data and the envisioned social networks that will keep the process of enrichment going. Having introduced these, several sections follow that specify model enrichment processing pipelines for enrichment of the BOLD models identified.

BOLD objects are identified by one or more URIs (incl. URL, URN, etc), potentially originating from different URI identification schemes. Legal data, narrowly conceived, consists of four major types of BOLD objects:

1. Documents are structured texts, hierarchically decomposable into linked lists of document fragments;

2. Metadada about documents and document fragments in the form of (*subject predicate object*) triples, where the subject is typically a document or document fragment:
3. references are labeled links between document fragments, and
4. labeled links can be used to create arbitrarily complex features of documents and document fragments;
5. Folders and clipboards containing documents, document fragments *and* metadata *as well*;
6. Groups of users.

Enrichment tools functionality is closely linked to BOLD object type, specifically, classified *by role*:

1. Content object: the URI is a URL, and the object may be dereferenced
2. Set-decomposable object: contains a set of other objects.
3. Linked-list-decomposable object: decomposed into a totally ordered set of URI; important for rendering content, and for the sorting function in recommendation.
4. Document: a linked list of document fragments
5. Document fragment: part of a document whose content may be rendered
6. Labeled link object: for metadata
7. Graph: a set of labeled link objects
8. Reference: a labeled link object that links two document fragments
9. Folder: a linked list of BOLD objects
10. Clipboard: a linked list of BOLD objects
11. Group: a set of groups or persons that own folders
12. Person: owns folders, and be part of a group

An abstract overview of the main object types and relationships, and their relationships, is given in Figure 1. The relationships are explained in the following subsections.

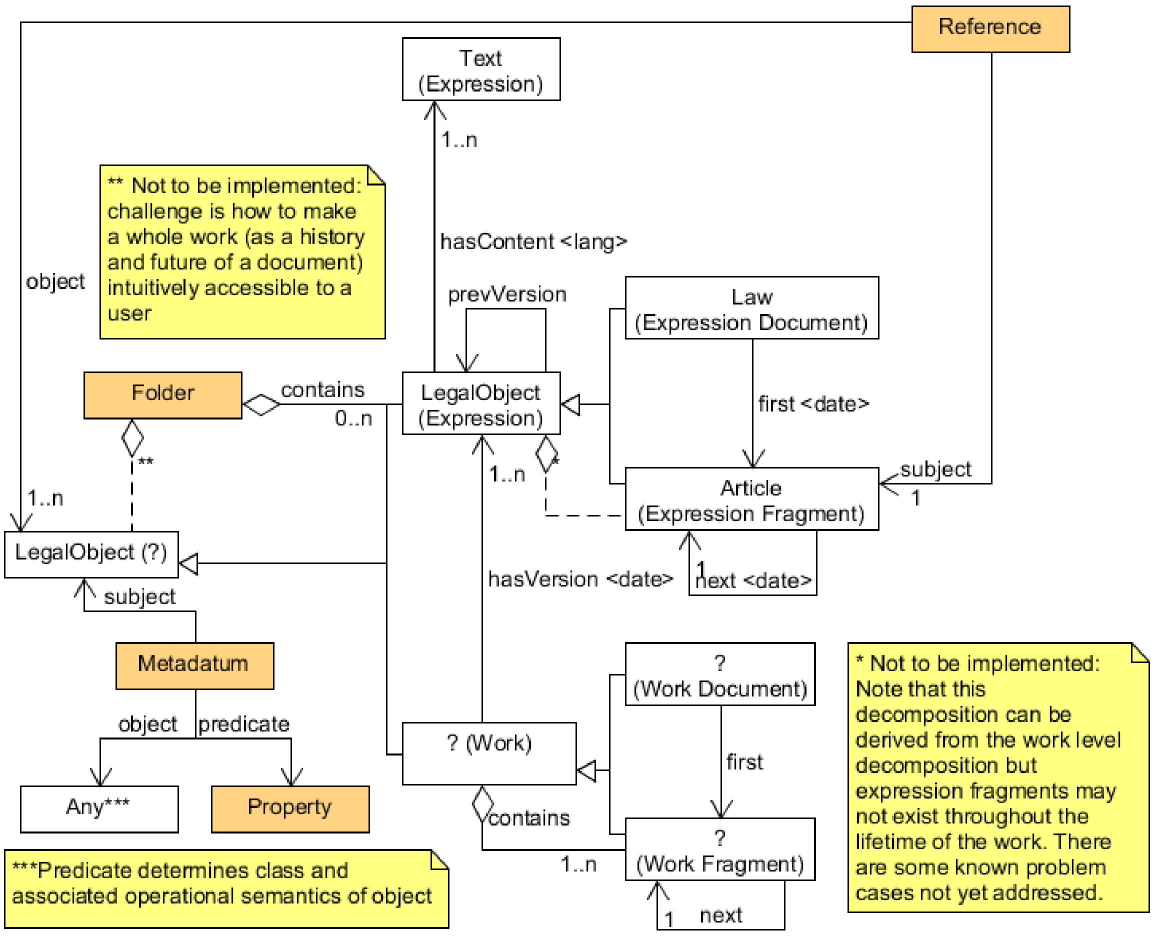

Reference
Text (Expression)
1..n
hasContent <lang>
** Not to be implemented: challenge is how to make a whole work (as a history and future of a document) intuitively accessible to a user
object
prevVersion
Law (Expression Document)
Folder
contains
0..n
LegalObject (Expression)
first <date>
**
1..n
1..n
*
Article (Expression Fragment)
subject
1
LegalObject (?)
hasVersion <date>
1
next <date>
subject
Metadatum
? (Work Document)
* Not to be implemented: Note that this decomposition can be derived from the work level decomposition but expression fragments may not exist throughout the lifetime of the work. There are some known problem cases not yet addressed.
object
predicate
? (Work)
first
Any***
Property
contains
? (Work Fragment)
1..n
***Predicate determines class and associated operational semantics of object
1
next

Figure 1: Abstract view of the main OpenLaws object types

In the following subsections a distinction is made between the work-level document and the expression-level document. The figure captures the implementation of documents in the OpenLaws repository, but **the prototype repository does not distinguish the work level**, basing versioning functionality instead on the option to set a date on the first and next relationships.

2.1 Documents and Content

***Content* objects** contain data, and some, but not necessarily all, content URL permit dereferencing to retrieve this data. In Figure 1 these are called LegalObject (from the OpenLaws repository data model). Content objects are distinguishable by type depending on the expected structure of the data, the operational semantics (e.g. for rendering) associated with that structure, and method for dereferencing.

Generally, BOLD objects are subdivided into :

1. *simple objects*,
2. objects that are decomposable into a *set of BOLD objects,* and
3. objects that – besides being decomposable into sets of objects – are decomposable into linked lists, or totally ordered sets, of *BOLD objects* of objects.

BOLD objects may participate in multiple decompositions. Wherever sets of objects are serialized:

1. **Linked lists of *content* objects** may be lexically structured in order-preserving XML/HTML/PDF data structures. HTML is strongly preferred.
2. ***Unordered sets of BOLD objects*** are, when not embedded in input XML/HTML/PDF data structures, lexically structured in RDF or JSON.

A **BOLD document** is a content object that can be decomposed into a linked list of **BOLD document fragments**. A BOLD document fragment is a content object that *may be* decomposable into a linked list of document fragments. When conceptualizing the document as a tree, all leafs of that tree are content objects. The leafs are the smallest level of structural subdivision that is (in that type of document) commonly ***referred*** **to with an unambiguous reference** in natural language. Below that level the leaf document fragment *content* may be further marked up, but this does not count as BOLD object decomposability.

Note that structured texts may require alternative structural decompositions, **but that OpenLaws prototypes of the enrichment tools do not support this**:

1. Text structured into chapters and articles may have an alternative decomposition into pages, with footnotes; and
2. in the annotation of individual sentences with markup in HTML with SPAN elements, annotators may come to alternative structural decompositions of a sentence, depending on purpose of the annotation.

2.2 Automatic Enrichment of Mark-up

It often happens that a corpus supplies documents that are not marked up or marked up only sparsely. Id est, the document content is largely plain text, without any described hierarchy or annotations. Because there are many regularities in legal writing, natural language processing (NLP) can be effective in the task of identifying various text elements, such as section titles and references.

The tasks require corpus-specific descriptions implementations, but we describe a general approach to inferring document structure below. The general process we describe below is a fairly standard NLP pipeline of tokenization, text segmentation and parsing a syntax tree, except our basic linguistic unit is not words but text blocks. Consequently, the syntax tree we generate corresponds not to a parsed sentence, but to a document hierarchy.

2.2.1 Parsing text elements

We assume that input documents reach us as either plain text or sparsely marked up XML documents, and this task should split the input document in such a way that we end up with a sequence of tokens. The semantic value that we apply to tokens is corpus-specific and task-specific. Because we wish to mark-up documents with annotations, it follows that we tokenize elements at least as granular as the blocks that would be annotated.

It is very likely that the author has provided some sort of visual guidance (e.g., employing whitespace between paragraphs and titles); we can use this to our advantage in tokenization.

The reference implementation of Dutch case law uses a custom grammar that tokenizes document portions into objects that keep a reference to their position in the original XML document, as well as applies some pre-processing for the following task of tagging the tokens with their role within the document. We use both a deterministic and a probabilistic approach to tagging the text elements (the latter using Conditional Random Fields, [7]) as one of four labels: numberings, section titles, new lines and text blocks.

Depending on the task, at this point we can start enriching mark-up. For instance, if we are looking for legal references, we can find a URL for the given reference, and add a link to the text.

2.2.2 Parsing a document hierarchy

We might want to combine the document elements into a document hierarchy. We can describe this problem as parsing a stochastic context-free grammar (SCFG), where the probability of production rules have a conditional component, based on their components. This allows us to deal with sequential numberings without severely complicating our grammar. The total number of possible tree derivations is exponential, but we can employ dynamic programming to construct the best possible parse tree in quadratic time and space.

For our reference implementation, we implement such a SCFG, together with the CYK algorithm extended with unary production rules.

2.3 Metadata and User-created Folders

Labeled links between BOLD objects are *(subject predicate object)* triples. Triples are a type of simple BOLD objects. RDF and JSON data has a standard interpretation as *(subject predicate object)* triples, forming a graph. These triples may be characterized where appropriate by:

1. a subset *R* of the possible subjects, predicates, and objects (product S X P X O),
2. a set of *edges* with predicate *p* E(p) = { (s,o): (s, p, o) in *R*}, or
3. a set of *features* of a subject *s*, F(s) = { (p,o): (s, P, o) in *R*}.

BOLD graphs are BOLD objects decomposable into a set of labeled links between BOLD objects. Graphs are the raw material for application of network analysis techniques.

BOLD references are labeled links between document fragments.

Folders and clipboards are user-created editable linked lists of BOLD objects. These are mainly initiated by user search and copy-to-clipboard actions.

2.4 Recommendation model metadata

The main function of folders and metadata is, from the enrichment tools point of view, the support of the recommendation function. Recommendation in OpenLaws is based on model-based collaborative filtering using topic models generated by **Latent Dirichlet allocation** [2]. **Latent Dirichlet allocation** (**LDA**) is a generative statistical model that allows sets of observations to be explained by unobserved classes that explain why some parts of the data are similar. Figure 2 illustrates the abstract concept,

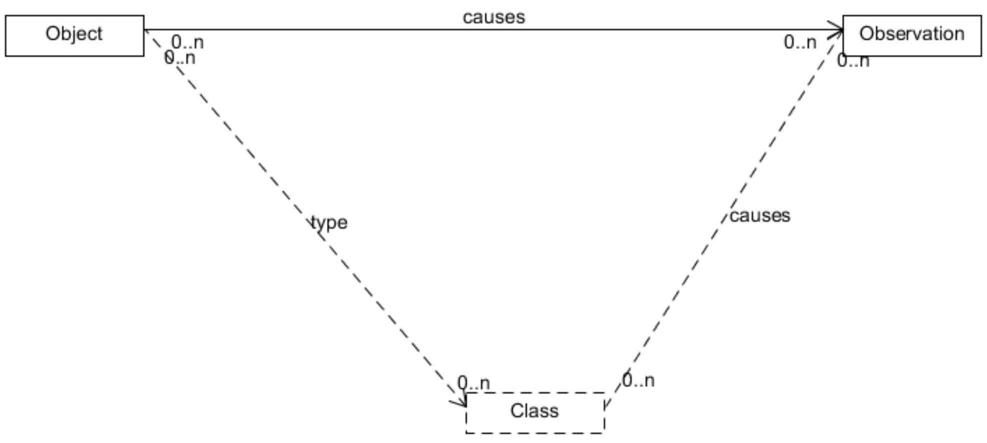


Figure 2:Abstract concept of LDA

and Figure 3 the interpretations in terms of documents and user-created folders.

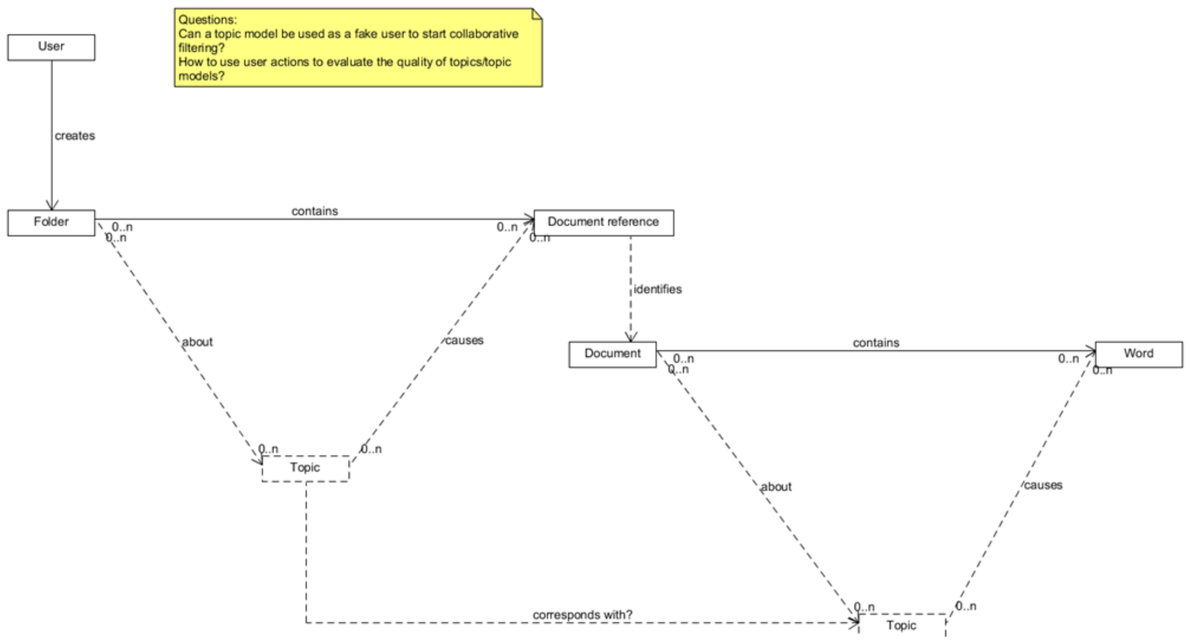
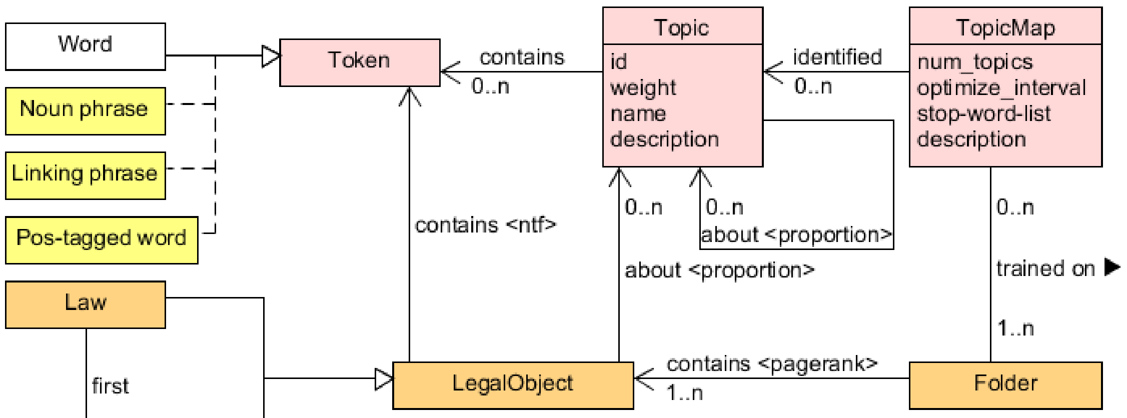

Questions:
Can a topic model be used as a fake user to start collaborative filtering?
How to use user actions to evaluate the quality of topics/topic models?
User
creates
Folder
contains
Document reference
identifies
about
causes
Topic
Document
contains
Word
about
causes
corresponds with?
Topic
0..n

Figure 3: LDA interpretation for documents and folders

If observations are words collected into documents, it posits that each document is a mixture of a small number of topics and that each word's creation is attributable to one of the document's topics. If observations are additions of URI references to a user-created folder, it posits that each folder is a mixture of a small number of topics and that each addition of a URI reference is attributable to one of the folder's topics.

Since the OpenLaws repository's recommendation function cannot initially be based on collaborative filtering of user-created folders, the recommendation enrichment is instead based on word-based topic modeling, where each topic model generated by the LDA algorithm is treated *as if* it were a user, and documents classified as belonging to a topic are added to a folder *in the order of proportion of the document classified into the topic*. The recommendation can therefore be based on classification of the similarity of the user and user-created folder to initially existing automatically generated user profiles. This addresses the *cold start* problem of encouraging a new user base to generate collaborative filtering data starting with recommendations generated automatically.

The metadata generated by the recommendation function enrichments is shown in Figure 4. The LDA algorithm produces for each allocation of a LegalObject (i.e. document or fragment of a document) to a topic a proportion score that can be used for ranking topics relative to a document and vice versa.



Input is from the Neo4j graph database, output also goes into Neo4j

The topic maps are computed with LDA (using Mallet library, java code)

To determine relatedness between topics we feed the token list of the topic as a text to LDA as a proxy for overlap in contents. Manual editing would also make much sense.

Pagerank is is used as a proxy for network centrality, and computed specifically for folders based on internal links, and based on shared topics, but multiplied by proportion (probability of presence, vs. the probability of attention captured by pagerank). This weighs direct links more heavily.

Although the prototype will be limited to LDA on words, an upgrade path is foreseen that will incrementally use the output of concept mapping (consisting of <noun phrase, linking phrase, noun phrase triples>) for determining topics

ntf = normalized term frequency (may be omitted)

LDA will be supervised by an editor, who validates the output and provides names and descriptions

Figure 4: Metadata generated by LDA

Since each topic contains a set of distinctive tokens, a topic can be rendered a document and therefore a proportion score for allocations of topics to topics can be determined as well.

2.5 Networks

2.5.1 Social Networks

Social network graphs are composed of persons and groups, both types of agent.

Groups are decomposable into sets of agents, shared memberships of a group can be interpreted as links between agents and vice versa, and shared members can be interpreted as links between groups and vice versa, following a social theory attributed mainly to Breiger [4]. Persons are agent that are not decomposable (i.e. leaf agents). Persons may play distinct roles[1] that can be interpreted as group memberships (and vice versa). Groups may be directly expressed by persons as a creative act, or discovered through network analysis techniques by looking at their activities as producers and users of BOLD. Persons may decide to which group of which they are member productions and uses are associated, and they may constrain membership of groups or visibility of documents to specific agents.

Because no function of the enrichment tools depends on social networks, because of the *cold start* problem mentioned before, these are not implemented.

2.6 Importing and Exporting BOLD objects

The openlaws importers bring together various datasets to a central, homogenized database. This means that for every legal corpus that openlaws supports, there needs to be a pipeline which extracts documents from a given corpus, converts them to openlaws format, and writes the converted documents to the openlaws database. During importing, we can also take the opportunity to enrich markup or to infer extra metadata.

Note that some assumptions need to be made to ensure that the documents that we receive as input can be converted to BOLD object. The most important is that the document must have some substantial content which could be represented as HTML.

[1]E.g. a legal scholar specialized in insolvency may at the same time be a part time judge in a cantonal court mainly dealing with small claims, and has distinct information needs within these distinct capacities, manifested in his production and use of information.

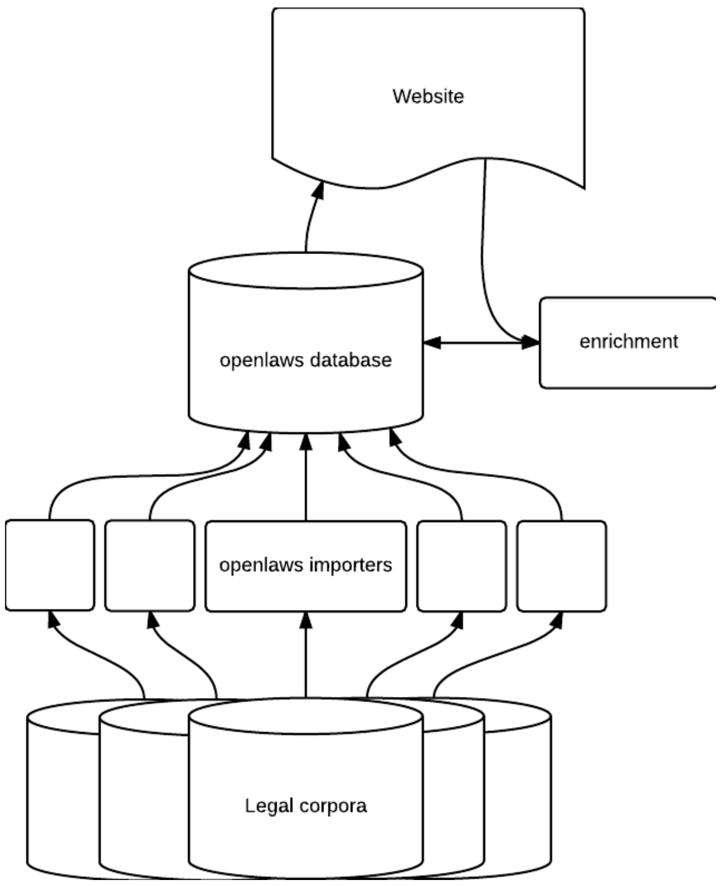


Figure 5:Dataflow from external data sources to the OpenLaws database and enrichment

3 Bibliographic identity

Bibliographic convention [5] is to distinguish legal texts and text fragments on at least four levels, as distinguished by MetaLex [3]:

1. On the *item level* legal texts and text fragments can be dereferenced by identifier and copied, resulting in a new item;
2. On the *manifestation level* any change to the data produces a new manifestation, including a change of data format, annotation of structure, or the embedding of metadata;
3. On the *expression level* only a change of the text by its author produces a new expression;
4. On the *work level* a text is identified by the details of its publication: as long as the title, author, and publication date remain the same, expressions are versions of the same work.

The **BOLD document** is essentially a manifestation. Most of the metadata refers to the corresponding work or expression, however.

A **BOLD work may be decomposed into a set of BOLD document expressions** and a BOLD document may be decomposed into a set of manifestations. Enrichment consists of 1) creating alternative manifestations of an expression, or 2) adding metadata about an expression or work. One work may be expressed multiple times. An expression may have many manifestations. Items may be freely copied, resulting in new items. **In the OpenLaws repository, only one manifestation of each expression may be assumed to exist, and the work is not explicitly distinguished, but implicitly accessibly (see** Figure 6)**.**

3.1 Versioning

For regulatory text, the distinction between works and expressions is of critical importance, because the text is typically changed over time. Most works are decomposable into a single linked list of expressions over time. In some cases (retroactive annulment of changes and unforeseen changes to scheduled changes in the future) the versioning chain may change over time retroactively, resulting in alternative versions of a text, in which case the versions cannot be expressed as a linked list (but rather as a partially ordered set). **In OpenLaws prototypes of the encoding tools a single linked list of expressions may be assumed to exist, accessible by setting a data on the first or next relationships (see** Figure 6)**.**

3.2 Languages

Many works are moreover available in alternative language variants. Support of this is obviously an important requirement in the EU. Each language variant is a variant of another expression. An alternative language creates an alternative decomposition into a linked list of expressions. In addition (non-authoritive) translations may exist: these are not considered expressions of the original work. **In OpenLaws prototypes of the encoding tools a single linked list of expressions may be assumed to exist: this does not allow for alternative language support. Topic modeling, as used in recommendation is not transparent to alternative languages as well.**

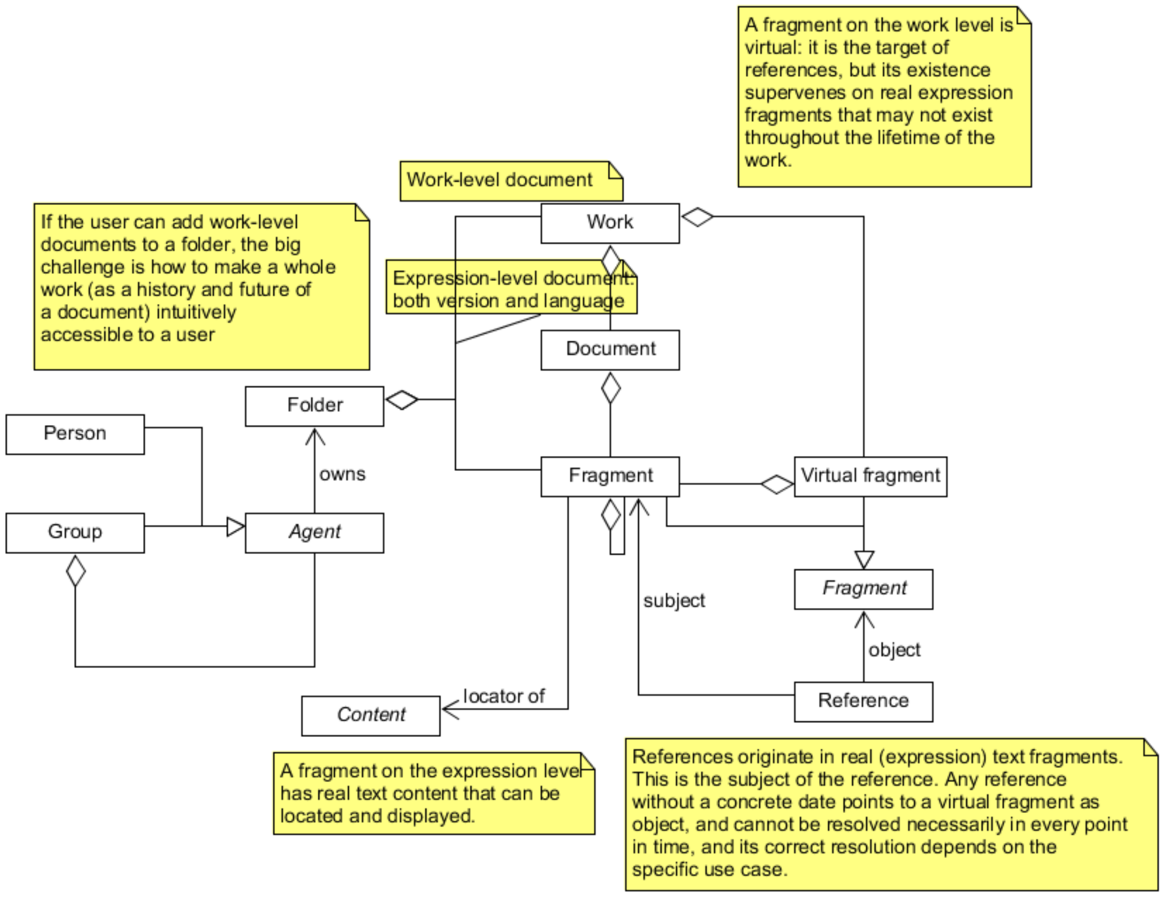

A fragment on the work level is virtual: it is the target of references, but its existence supervenes on real expression fragments that may not exist throughout the lifetime of the work.
Work-level document
If the user can add work-level documents to a folder, the big challenge is how to make a whole work (as a history and future of a document) intuitively accessible to a user
Work
Expression-level document: both version and language
Document
Folder
Person
owns
Fragment
Virtual fragment
Group
Agent
subject
Fragment
object
Content
locator of
Reference
A fragment on the expression level has real text content that can be located and displayed.
References originate in real (expression) text fragments. This is the subject of the reference. Any reference without a concrete date points to a virtual fragment as object, and cannot be resolved necessarily in every point in time, and its correct resolution depends on the specific use case.

Figure 6:Bibliographic identity of objects in the OpenLaws repository

3.3 References

Conventionally, regulatory text refers to other regulatory text on the work level: which version should be used is left to the reader. Some call this a *dynamic* reference. A court decision refers to a specific version of a regulatory text on the expression level. Some call this a *static* reference. Any other text that refers to legislation by default refers to a specific version, unless the text is under editorial control and guaranteed to be up to date with the text it refers to. Obviously, texts may discuss an old version, compare an old version to a new version, compare two language variants of a version, or (very frequently) discuss an anticipated version of a regulatory text[2].

To support existing referencing practices, the object of a reference triple can be the URI identifying a document fragment (for the expression level), or a URI referring to a **virtual work-level fragment**, which identifies a fragment present in all, or most, expressions of the work. **The virtual fragment is decomposable into a linked list of document fragments.** Figure 6 **illustrates the open problems. These are rare, but at present unresolved in the OpenLaws repository. Generally, references are not supported by (all) OpenLaws importers, and reference networks are therefore not yet supported in the implemented enrichment pipeline.**

3.4 Mixed Content and Quoting

Legal text is often quoted, in modifying legislation, in court decisions, papers and books, etc. Quoting is an alternative way of referencing information, and the quoted text fragment is both a part of the quoted and of the quoting document. It is moreover a potential source of interesting and innovative manifestations of text fragments. **Quotes are treated as a type of references: reference by inclusion. The OpenLaws repository importers do not mark quoted content, and this type of reference is therefore not supported.**

4 Functions

When we have a (large) collection of BOLD objects with relations, we can try to enrich them in basically two ways:

1. Have users add metadata. This is what is typically called ‘the wisdom of the crowd’ or ‘crowd sourcing’. Users can highlight text, annotate it with key-words or comments, make collections of documents in folders (and name these) or otherwise specify relations between documents.
2. Run algorithms on the data and add metadata based on their outcomes. This is what we call automatic enrichment. Basically, we have taken two approaches: (1) Use network analysis to generate metadata from the network of BOLD objects; (2) use text comparison methods (like LDA) to generate metadata.

4.1 Basic Functions implemented in OpenLaws Prototype

For reference (in the bottom table) a filtered functionality list implemented for the prototype can be

[2]The news value of a discussion of a legal rule is highest well before the new rule goes into actual effect. The importance of anticipation of changes in the law should not be underestimated!

found. It is possible to categories that functionality as T1 (explained in Table 3: Technology categorisation). Further from that implementation we will develop our new functionalities which need to be enhanced for enrichment and further improvements.

Table 1: Function List

OBJECT	DESCRIPTION	FUNCTIONS
SEARCH	search content	search
FOLDER	manage content	add/remove legal object
		load content
GROUPS	manage user groups	add users
		add new group
		remove group
		rename group
		get list
ROLES	manage roles	add new, remove, rename
		get list
USERS	manage users	add, delete, rename/update
		login (get session)
		request pwd reset
		confirm pwd reset
		search (by name)
USERS/GROUPS	manage user/group relation	add to role
		add to group
		remove from role
		remove from group
		get groups of user
NOTIFICATIONS	retrieve notifications	get single
		get list
TAGS	manage/retrieve tags	get list
		get single
		get simplified (autocomplete)
		add, remove, rename

The available model

Based on research and further improvements the functions are built on the model in Figure 7: Graph Model.

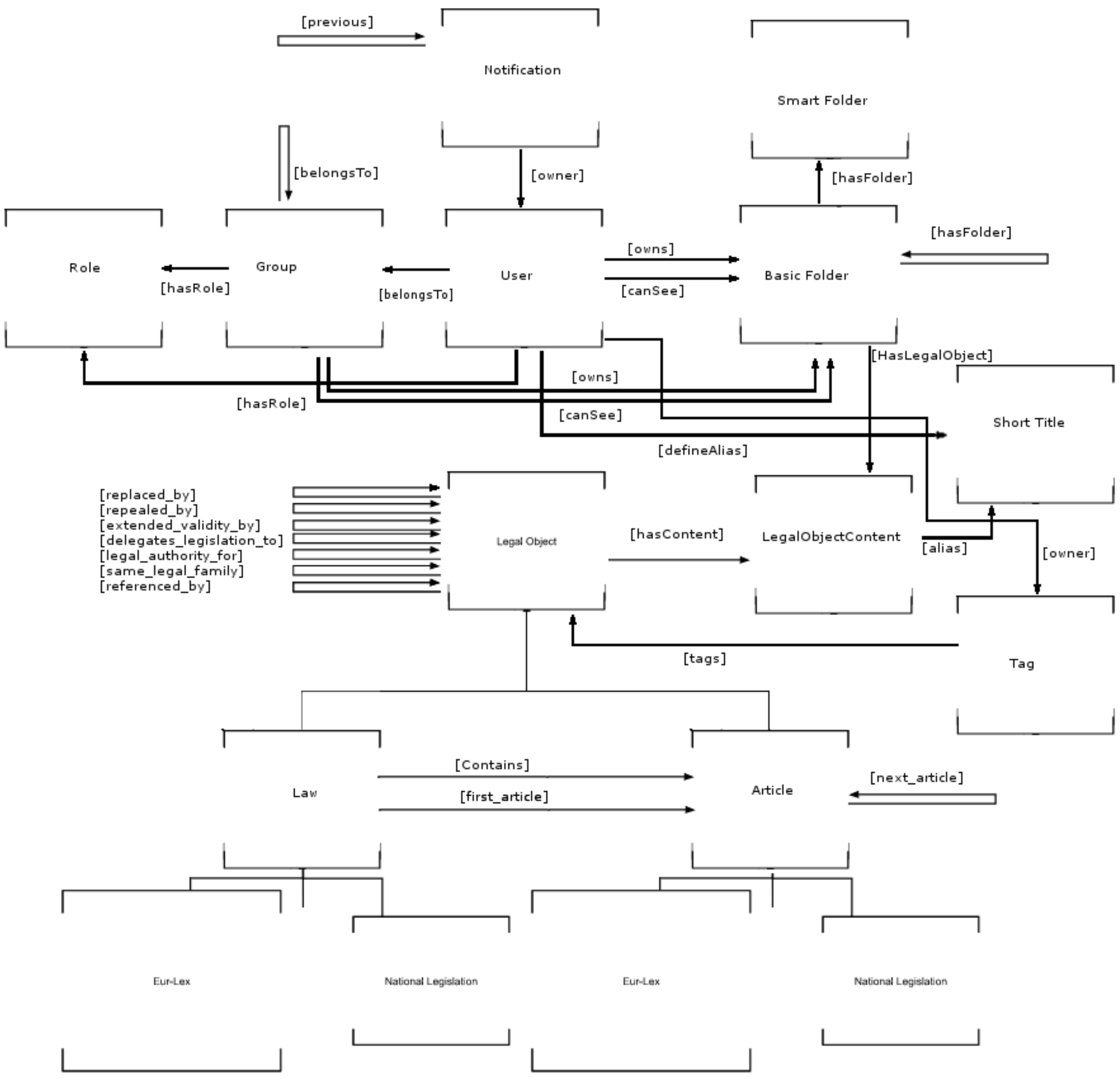


Figure 7: Graph Model

Although this architecture may change in future it is the foundation of the current platform. As this model is implemented using a graph database (Neo4j) it is highly changeable and consequently not binding to initial design decisions.

User Type Relations

As anticipated in the "handbook for stakeholders" there are many different user groups. They might potentially profit of the current legal information platform prototype in different ways.

Table 2: User Types

legal expertise (lower to higher)	judges lawyers notaries general counsels legal scholars (T3)	judiciary legislative authorities large law firms (T4)	Vision: BIG OPEN LEGAL DATA (T5+)
	small enterprises law students semi-professionals (T2)	medium enterprises gov. administrations legal publishers (T3)	large enterprises (T4)
	(individual) citizen (T1)	citizens (groups/associations/etc.) (T2)	society (as a whole) Member States (T4)
	individual/institutional size (smaller to larger)		

In Table 2: User Types users/groups are shown which most likely benefit of the current system. Extended enrichment functionalities will increase the affinity for legal professionals and communities of bigger size. The darker the background is coloured, the higher is the chance of that group to use openlaws. Or in other words: enrichment implementation process needs to be more advanced for the light coloured user/groups in order to profit of the platform.

Subsequent sections will deal with those new functionalities and concepts. This document will list and explain them in further detail without a claim of completeness, because enrichment should be understood as continuous process which improves openlaws over the time.

Some of mentioned ideas might be already realized by openlaws company spin-off at time of review but most are not yet implemented. Some might be even beyond current project scope and subject to realisation of further projects or external partners.

Prognosis

Analysing the table of the User Type Relation section further it is possible to categorise technologies relevant for the groups from T1 to T5+ where the number acts as year reference but also as an indicator which groups might profiting most (colour coding). Although not all technologies introduced for example in T2 are necessarily (only) related to "groups, associations or small enterprises". Thus it is rather a kind of classification which tries to combine the evolvement year with the user size/expertise.

Table 3: Technology categorisation

TECH TYPE	USER GROUP	LEGAL EXPERTISE	INST. SIZE	TIME (YEARS)
T1	citizen	1	1	**0-1**
T2	groups associations	1	2	**1-2**
T2	small enterprises	2	1	**1-2**
T3	judges lawyers notaries …	3	1	**2-3**
T3	medium enterprises gov. Administrations legal publishers	2	2	**2-3**
T4	society member states	1	3	**3-4**
T4	judiciary legislative authorities large law firms	2	3	**3-4**
T4	large enterprises	3	2	**3-4**
T5+	Bold VISION	3	3	**5+**

For further clarification take a look at the technology list above. T5+ in the technology type column stands for ideas/concepts which are extremely difficult and only with high effort reachable or just not realisable at the current knowledge/technology base of computer science.

In graph of **Error! Reference source not found.** prognosis for Table 3: Technology categorisation is shown and when combined with Table 2: User Types grasping the concept might be more obvious.

When combining Table 3: Technology categorisation and Table 2: User Types a graph can be generated which is shown in Figure 8. This graph should help making the concept more obvious. The larger the institutional size is, the higher is the procedural overhead usually.

Similar logic applies for legal expertise: handling legal processes is getting more and more complex, the higher the legal expertise of the user group is. The consequence here is that with increasing institutional size and legal expertise, also the technology which might help the respective user group is getting more complex, be it either to implement or to integrate the technology in an existing infrastructure.

Also taking that time component also into account the three-dimensional plot of Figure 8: Evolvement prognosis can be depicted. In other words: the closer - visually - a green point with a user group is the more complex a task for that group will be and the later in time it will be realised.

Looking deeper into the prognosis, the plot is having the BOLD Vision located outside of the cuboids bounds because it's the beyond the year 2020 and may be categorized as T5+ according Table 2: User Types/Table 3: Technology categorisation.

4.2 Annotating and highlighting

Annotating and highlighting is an important and crucial part of a legal platform, which is in demand to explain and clarify different parts of laws or legal aspects for people who like to quickly understand the legal content. This applies for layperson as well as legal experts.

As mentioned in the beginning of the section there are basically two ways to add enrichment functions: **Manual** or **Automated** which of course also applies for annotations.

Manual annotations in openlaws can be generated by:

- Layperson
- Expert
- Voting's
- Social Media Interaction
- ...

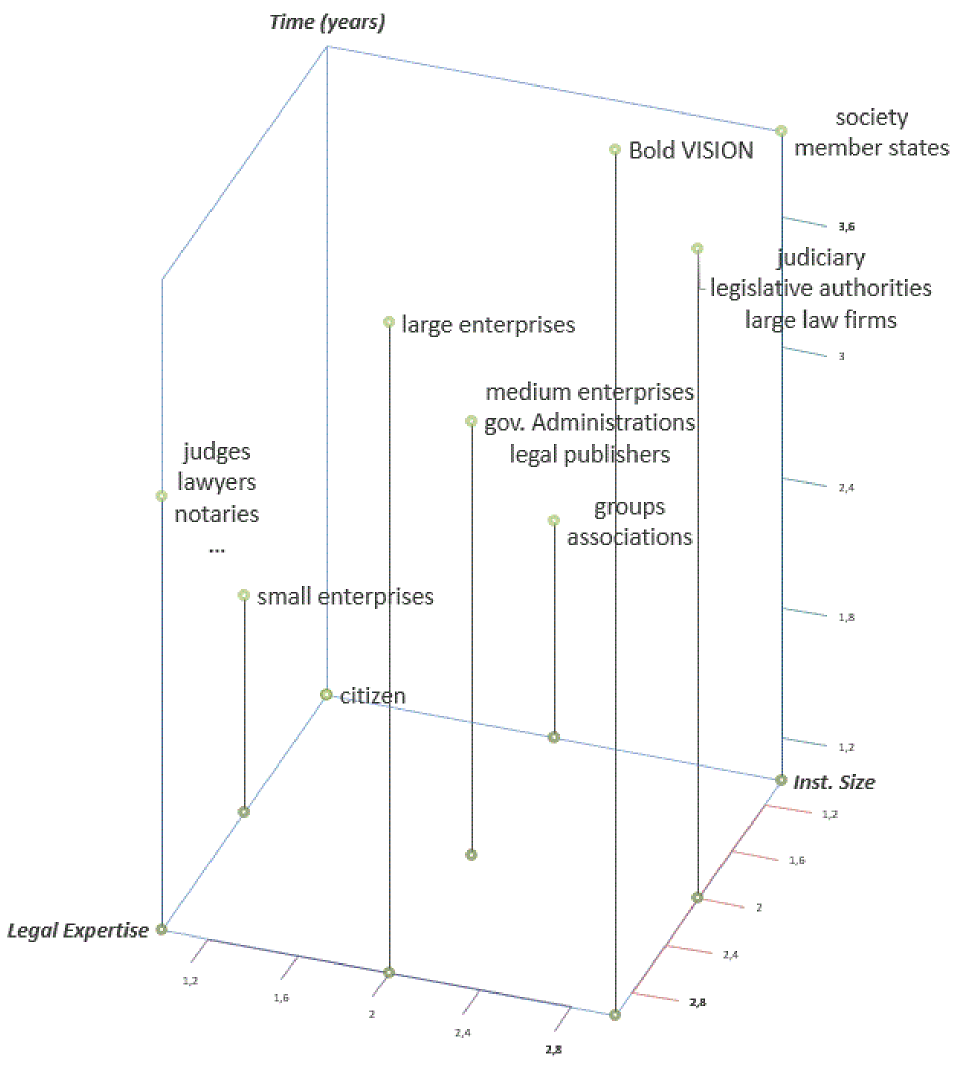


Figure 8: Evolvement prognosis

Automated annotations candidates:

- Natural Language Processing
- Reference Processing
- Dictionary Approaches
- Predefined knowledge trees
- Machine based learning
- Extended Indexing and Grouping/Classification
- Complexity Index Estimation
- …

When looking closer at annotations a break down into different types can be achieved:

- **Bookmark:** Marker at (a point in) a resource
- **Change:** Request for modification
- **Classification:** Assignment of a group or a class
- **Description:** Description of a target
- **Highlight:** Highlighted section of a resource
- **Link:** Relationship of unspecified semantics
- **Moderation:** Assignment of quality or a review
- **Question:** Question about a target
- **Reference:** Citation or reference pointer for legal objects or outer resources
- **Replay:** Response to a previous statement
- **Tag:** A tag on target resource – textual

These types may be all support in future or even generated automatically where possible and useful. To give some examples the further sections explain ideas which are going beyond simple manual annotating and highlighting.

Community Highlights

These can be seen as an extension to simple highlighting. All user-generated highlights per legal items are collected, normalized and displayed within the particular legal item. Text areas that were marked often are featuring a higher colour intensity, while areas that were less highlighted have a lighter colour intensity.

Automated Citation Creation

Although not a direct annotation technique this enrichment idea provides an automated creation of a suitable citation formats for each particular legal item. It will support the Citation Style Language (CSL) to be flexible enough for future modification and the addition of individual, legal field-specific language styles for different member states.

4.3 Notification

Notifications are highly connected with OpenLaws System-Events and the Messaging functionalities. They may be manually pushed by a user or sent because of an event like a particular content change.

Subscribe to Event -Enrichments

OpenLaws tracks legal changes in detail when new laws are inserted or existing ones are changed. In

OpenLaws people should be able to attach to specific those event by some kind of subscription.

This should be implemented in way that different subscription types are possible. In case of a specific occasion the user might be informed by an internal massaging system or/and via e-mail.

As mentioned these events might be a simple legal object change incident or also based on a more complex **time-based analysis**.

Time-based Analysis

Time-based analysis could offer information on user behaviours or folder/annotation creation. Including sophisticated **monitoring** possibilities legal companies may track and get a better overview of their legal situation and legal evolvement in their field. Especially because of the complex legal structures many companies find in.

Those complex structures sometimes are not manageable anymore which a lot companies puts into a semi-legal state as picky details of laws could be easily overlooked.

Traffic/Frequency information is also a potential time-based candidate to give users better notifications by assigning an importance indicator to different events.

Social Media

When a bigger user base is established social media will play a central role for notification functionality because notifications can be also extended to be used as normal messaging system where users communicated to each other, share information and also could rate content. As those changes are recorded they also will be part of the notification system and generate different subscription types.

"Folder Changed" - notification

A more detailed example of an event subscription could be a change of a folder. Users will collect information in Folders depending on der necessities or requirements and might even share them to others using the social features of the platform.

Sometime those folders may be work-in-progress and change quite often. Consequently it would be interesting to get – after a subscription – a notification that some new legal objects where added to the folder I have got in my favourites.

But not only when an object is added but also when a law changes - or anything else happened in terms of that folder - people could be informed.

4.4 Recommendation

This section lists enrichment functions for:

1. *sorting* documents and text fragments into folders automatically by topic affinity (using topic models);
2. *ranking* documents and text fragments in folders automatically by relevance to the topic (using topic models); and
3. *ranking* documents and text fragments globally by overall relevance in the whole corpus (using PageRank/network centrality).

These functions depend on the availability of links between documents, document content, on the content of user-created folders. During the Openlaws projects a number of prototype implementations were made for testing recommendation, but most of these were not integrated into the OpenLaws repository. See Table 4 for a list of experiments, and the data sets used, evaluations, and resulting publications. The third function, ranking globally by relevance, is not supported in the implemented OpenLaws repository.

Note that enrichment functions are *offline* functions, in the sense that a command to execute these functions may be queued for processing at a convenient offtime.

Table 4: List of enrichment experiments

Functionality	Data set	Prototype implementation	User evaluation	Repository Integration	Publication	Results
Network centrality	Dutch case law	yes	yes	No	Winkels , Boer, Vredebregt, Van Someren (Jurix 2014, best paper award)	acceptable to users
Pagerank		no	no	No		presumably similar to network centrality
LDA words	Dutch case law	yes	yes	Yes	Winkels & Boer, 2016 (CELSE2016)	acceptable to users
LDA words + network centrality	Dutch case law	yes	yes	No	Winkels & Boer, 2016 (CELSE2016)	network centrality has no added value
LDA words + Pagerank		no	no	No		presumably similar to LDA words + network centrality
LDA noun phrases	Dutch case law	ongoing	ongoing	No		
LDA + Cmap	Dutch case law	ongoing	ongoing	No		
LDA noun phrases + Cmap	Dutch case law	ongoing	ongoing	No		
LDA collaborative filtering on folders		ongoing	no	no		
Cmap	UK case law	yes	yes	no	Boer & Sijtsma, 2014 (Nail 2014)	Better noun (phrase) selection required

4.4.1 Network centrality

Global relevance of documents and document fragments can be determined by computing the network centrality in the whole corpus. Experiments depend on having a sufficiently complete set of documents, and have been performed on the corpus of Dutch law and case law, but not using the OpenLaws repository (cf. [7]). The experiments computed network centrality offline, but the PageRank algorithm may be assumed to approximate the results of network centrality, for theoretical reasons. Pageranks may be user to sort documents and document fragments in a set *that cannot be assumed to be about a single, or a few topics*.

Given the state of development of importers for the OpenLaws repository, and lacking reference resolution, **this function could not be integrated into the enrichment tools.** Evaluation with users as a recommendation were fairly positive, although the function helps more for laymen than expert users.

4.3.2 Topic Models

Recommendation in OpenLaws is based on model-based collaborative filtering using topic models generated by **Latent Dirichlet allocation**. **Latent Dirichlet allocation** (**LDA**) is a generative statistical model that allows sets of observations to be explained by unobserved classes that explain why some parts of the data are similar. If observations are words collected into documents, it posits that each document is a mixture of a small number of topics and that each word's creation is attributable to one of the document's topics. If observations are additions of URI references to a user-created folder, it posits that each folder is a mixture of a small number of topics and that each addition of a URI reference is attributable to one of the folder's topics.

A number of objects can be classified by topic based on the topic models, as shown in the figure above: documents, whole folders of documents, and search phrases. Within a topic, it is possible moreover to sort the documents by relevance to the topic (i.e. the proportion score). **The only fully implemented pipeline generates a topic model based on words in documents.** Other experiments (completed and still ongoing) generate topic models based on:

1. noun phrases and words
2. folders as lists of references
3. lists of references as they occur in a document.

In theory it is moreover possible to generate topic models from relevant whole propositions (generated as a side effect of a visualization experiment that is still ongoing).

The implemented pipeline works as indicated in Figure 10 **(the path shown in green).** A list of references (or: a folder) is fed to the pipeline. The texts the references refer to are dereferenced and fed to the topic modeler, which proposes a topic model and a text-to-topic allocation. For each topic a folder is created in the OpenLaws repository, and texts are entered into the folder based on the text-to-topic allocation.

Alternatively, **an existing topic model may be used to classify texts (including search phrases), without affecting the topic model. Classification is efficient enough to be used for online search in the repository.**

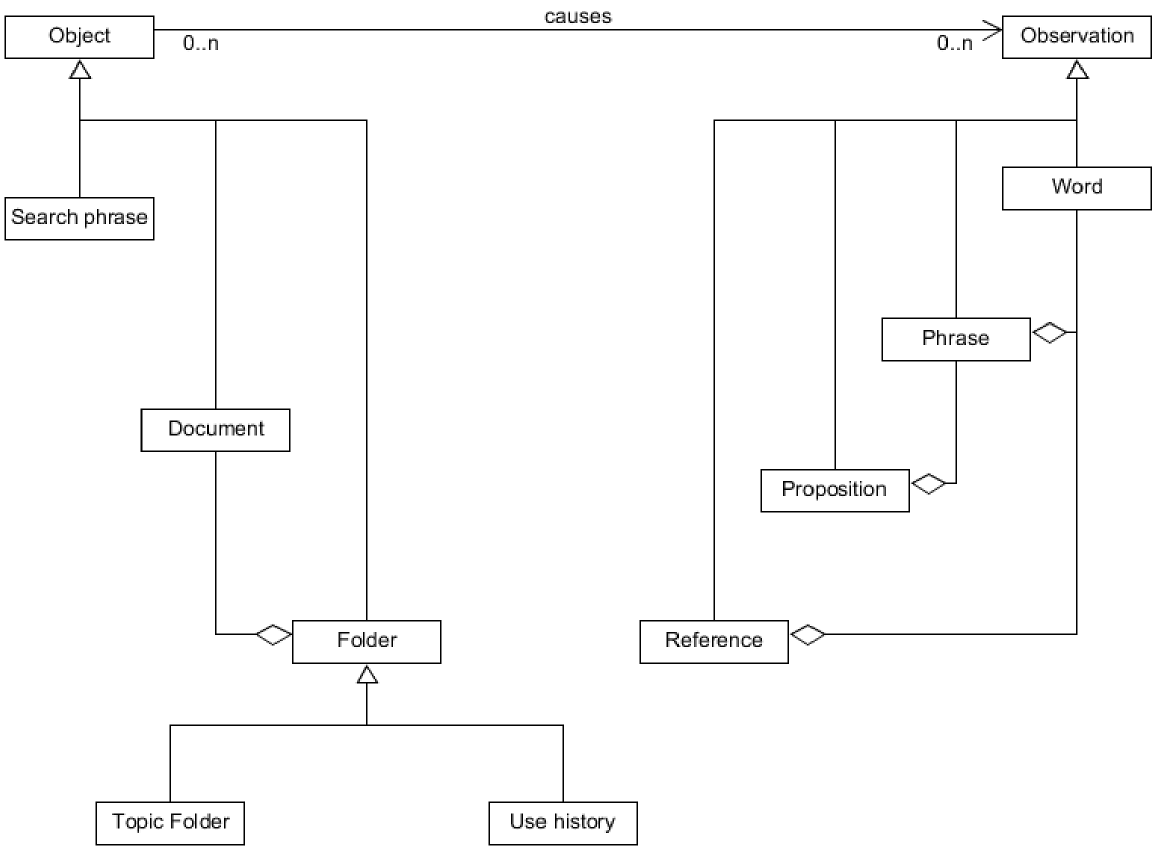


Figure 9: Taxonomy of OpenLaws objects and observations. White arrow-points indicate subtype relations; white diamonds contain relations.

4.3.3 Evaluation and ongoing developments

It is important to use future OpenLaws users to evaluate the quality of topic models. One obvious way to do that is by allowing the users to add whole topic folders to their own folders as subfolders. In order to assist users in understanding at a glance what a topic is about, we have experimented with a way of visualizing topics using concept maps. Tokenizing noun phrases is a step required for constructing a topic map, but is moreover expected to result in better topic maps. This development is shown in yellow **(not implemented in the repository)**.

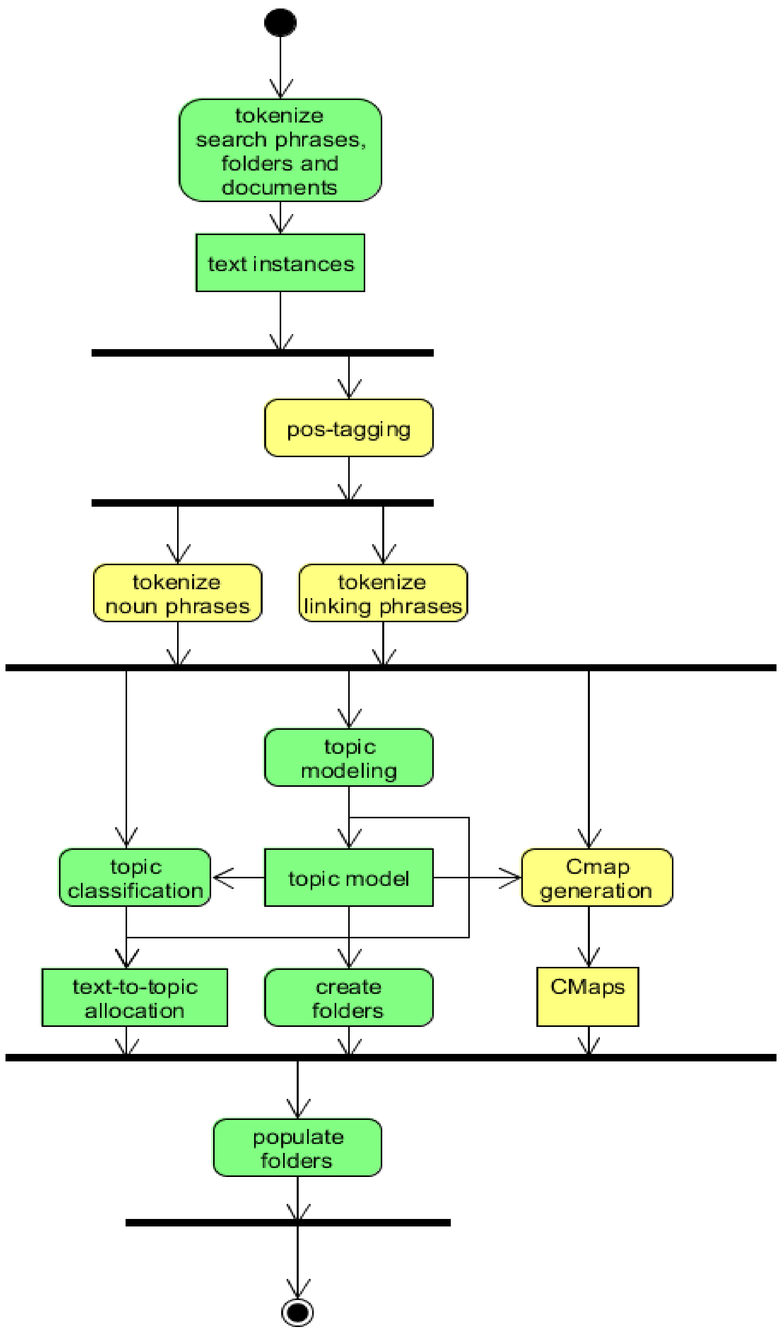


Figure 10: Implemented enrichment pipeline (in green)

4.5 Visualisation

Is not directly creating new meta data in terms of enrichment but takes legal parts as well as generated (automated or manually) meta data content and visualises those outcomes in a more human understandable and more easily graspable way.

Showing time related aspects or connections to open data are just some of the possibilities which can be used to create visualisations.

4.5.1 Concept maps

Concept maps can be used to visualize both the contents of a document and the contents of a topic folder at a glance. The concept map, as formalised by Novak, is a structured diagram containing concepts connected by linking phrases. It's a hierarchical tree-like structure, which is often concentrated around a focus question. In the centre the superordinate concept is displayed, and the meaning of a concept is in part determined by the concepts directly related to it. Concepts are indicated by a label inside a box, often consisting of a noun phrase. Linking phrases are the arcs connecting associated concepts, in most cases consisting of a verb phrase. Concepts connected by a linking phrase are referred to as propositions (see Figure 11). The concept map proposition is syntactically equivalent to the triple in RDF and JSON, and can be inserted as metadata in the OpenLaws repository.

In 2014 we performed an experiment on automated concept mapping for documents, without the use of a topic model. In that experiment, the selection of noun phrases was determined to be a weakness of the approach. An ongoing experiment tests whether good topic maps can be generated for both documents and topics (if trained on noun phrases), based on the process. **This functionality is not integrated in the OpenLaws repository.**

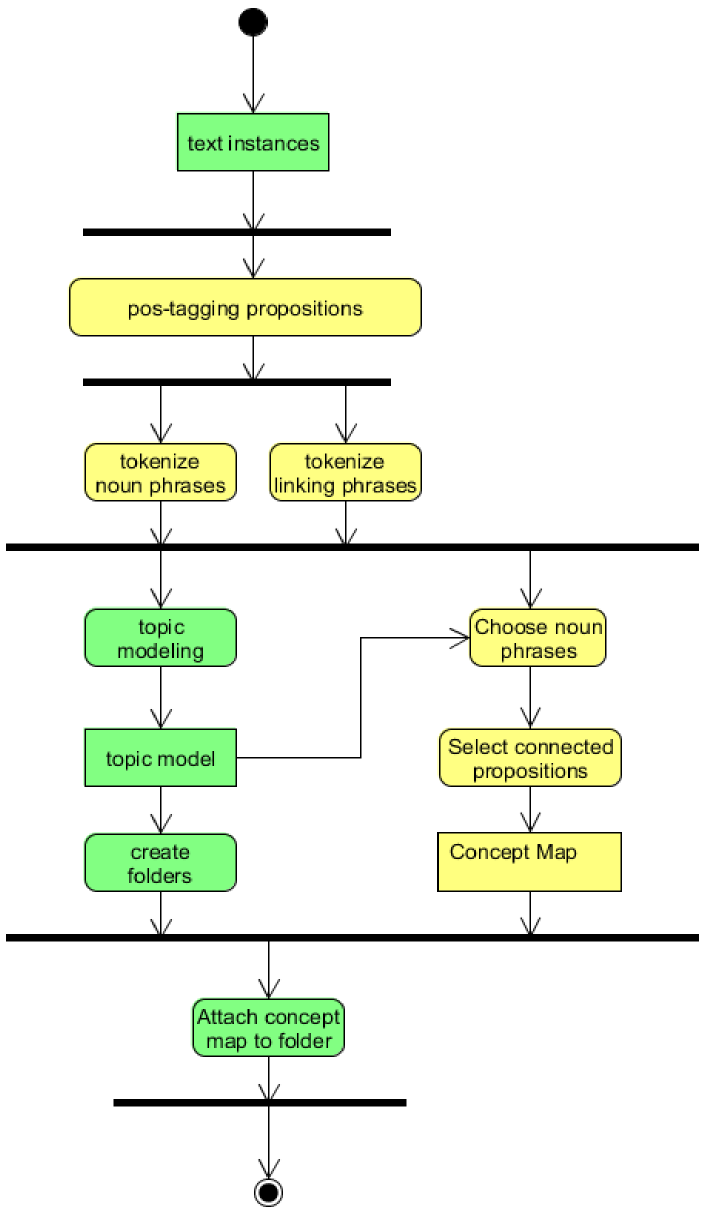


Figure 11: Ongoing experiments on topic modelling to steer concept map generation

4.5.2 Concept Maps "alike" Example

Concept maps are a very rich in rendering of aspects which cannot easily been just by basically listing topics of specific legal search. In this example the results of simple search on OpenLaws is analyse by Lingo3G clustering engine [5]. The results are collections of text documents which are clearly-labelled into hierarchical thematic folders called clusters.

Those clusters can be nicely visualized by concepts maps by Lingo3G add on tools. This example Figure 12: Search result clustering and visualisation shows a search in OpenLaws with additional clustering for "Schengen" which can be seen in Figure 12: Search result clustering and visualisation.

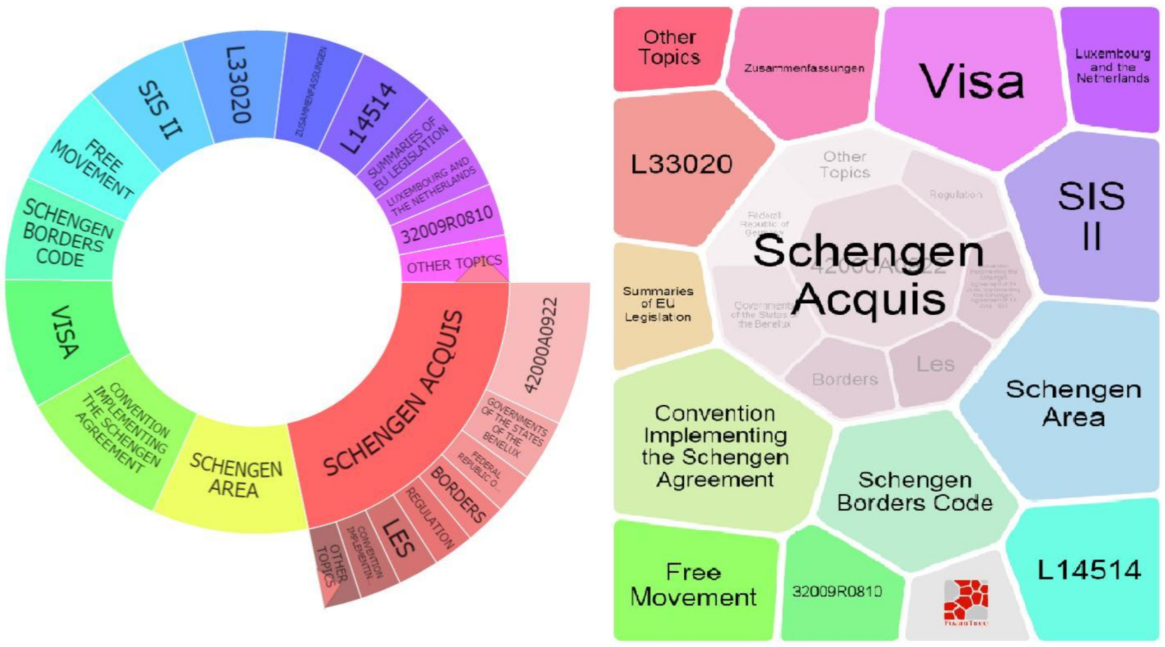


Figure 12: Search result clustering and visualisation

4.5.3 Maps (cartographic)

Cartographic maps might be used for simple aspects like showing where lawyers are located or more sophisticated aspects like showing grouping main topics of local civil court decisions. Here are examples following up.

Map of Lawyers – This map includes a comprehensive list of layers for a specific member country. The list does not only hold contact data for each lawyer, but also keywords regarding the specialisation. These keywords can then be used to filter lawyers. In addition, each entry also has a geographic coordinate of the lawyers' premises. Therefore, geographic searches also become possible.

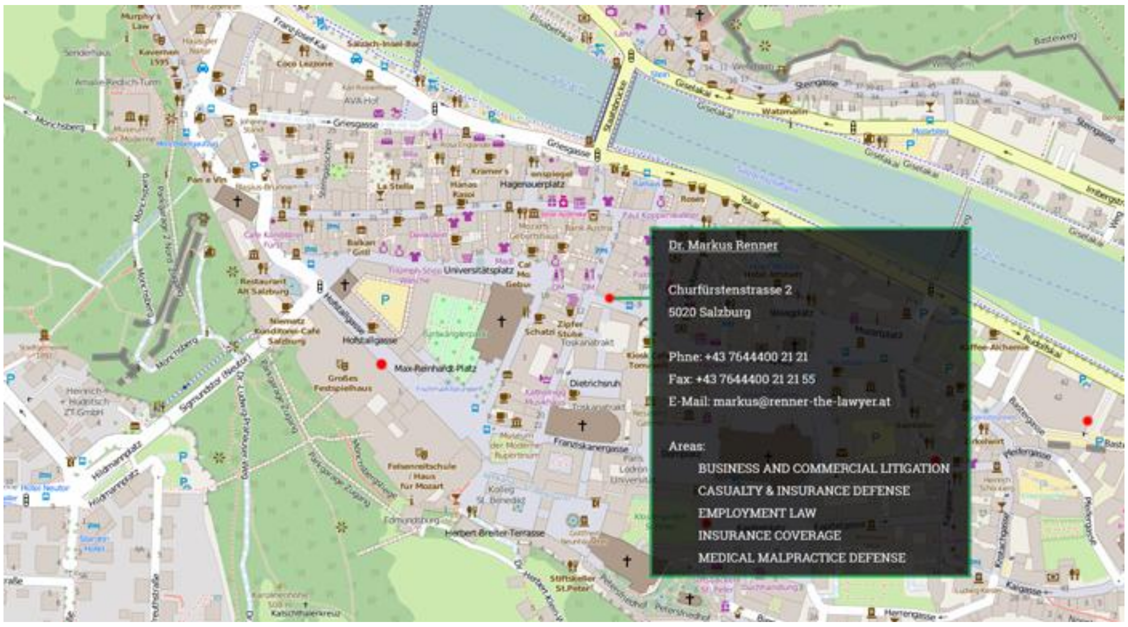

Dr. Markus Renner
Churfürstenstrasse 2
5020 Salzburg
Phne: +43 7644400 21 21
Fax: +43 7644400 21 21 55
E-Mail: markus@renner-the-lawyer.at
Areas:
BUSINESS AND COMMERCIAL LITIGATION
CASUALTY & INSURANCE DEFENSE
EMPLOYMENT LAW
INSURANCE COVERAGE
MEDICAL MALPRACTICE DEFENSE

Figure 13: Lawyers map

Map of local courts – This map shows all official courts in a country. Additionally it provides an order list of the "most-important" or "most-handled" legal fields/areas of that particular court.

Maps connected with Open Data – Open Data like on www.data.gv.at are very useful open sources which can be connected with locality. For example demographic components or the use of an open facility like public transport in relation to their location and whatever type of legal information.

4.5.4 Graph Clusters

Straight forward and more or less shipping with a graph database like Neo4j comes a clustered view of specific query (using cypher language) executed against it. Thus by calling a statement like

match (n:ATLaw)-[r]-(rn) return n,rn limit 500

it is possible to get a well visualised structure of the Austrian law. It is not purpose of that document to describe that query further. This query illustrates that although it is quickly written; it gives a very sophisticated and detailed visualisation. Due to large size of the Austrian law that query limits the resulting nodes and relations to 500. In Figure 14: automatic clustering of Neo4j the graphical output is shown.

It becomes apparent when investigating the graph further that clustering has occurred. On the left side there are gathering couple of laws forming a cluster. Those have some kind of similarity which might be analysed in detail. Similarities that a layperson or even an expert did not anticipate beforehand.

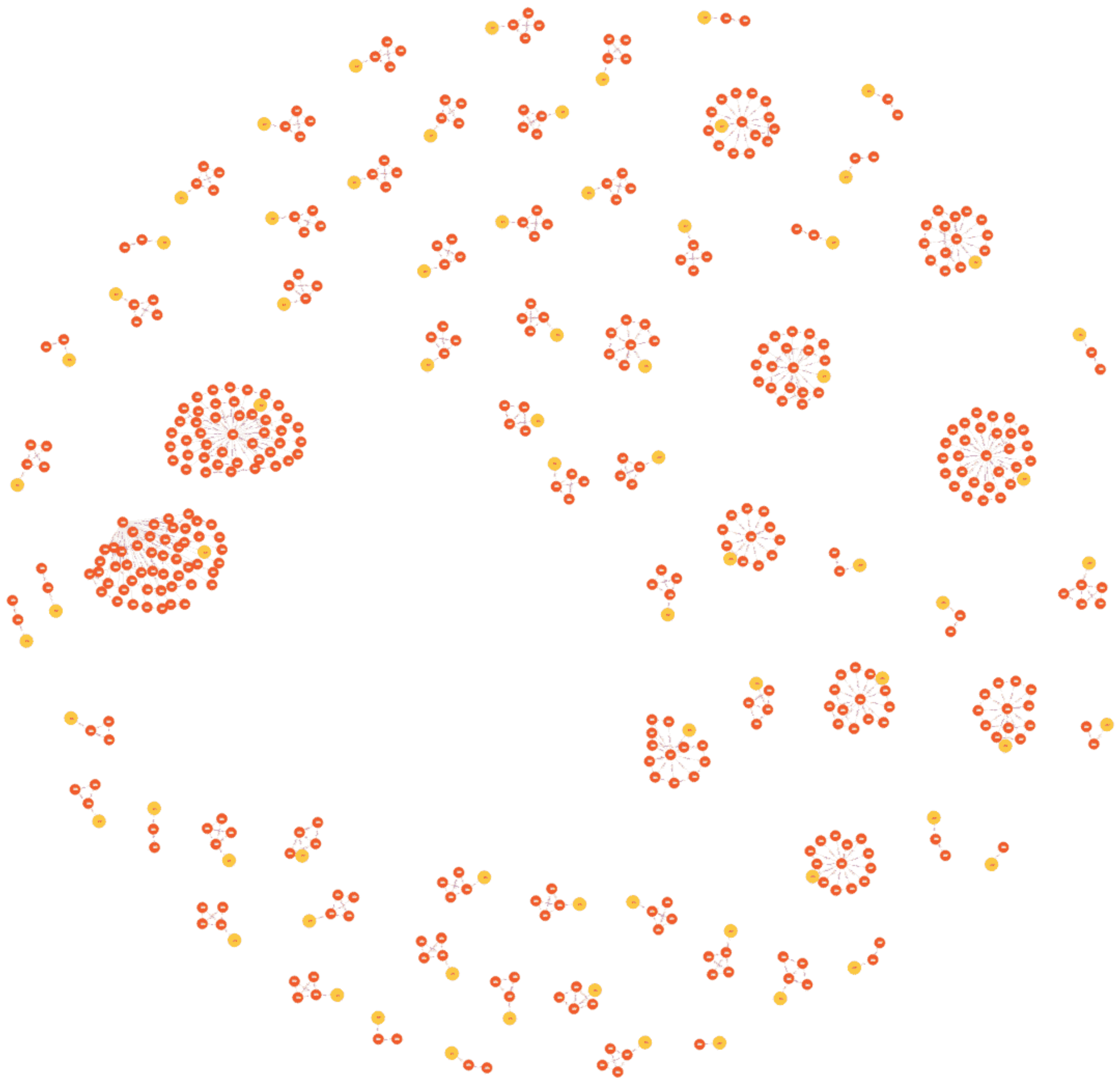


Figure 14: automatic clustering of Neo4j

The original query has been focusing on breadth first search. Consequently numerous laws have been found having a short depth distance can be recognized. On contrast it would have been possible putting emphasis on just a view nodes or even one node and using cypher query language possibilities to advance a search into the graph structure depth.

Although not directly created by Neo4j "Figure 15: EUR-Lex cluster" also shows an even better visualisation of a cluster.

4.6 Others

4.6.1 Human Machine Enrichment

OpenLaws interaction space between a human and machine currently as implemented in the prototype is a HTML interface based on AngularJS. Although AngularJS offers a great possibility for dynamic web applications and manifold interaction possibilities it is based on JavaScript and therefore difficult to handle for many simplistic or old browsers. Also HTML5 is used which is nowadays highly understood to modern browser but problematic for users with older personal computers or devices.

Nevertheless OpenLaws has the potential to overcome that issue by providing different interfaces for different purposes, which is possible because of the REST based interface of server implementation. Any software client with access to the web having the documentation of the web-service is able to pull all the information and interact with the system.

Thus different interfaces are possible without parsing a complex structure beforehand to extract the relevant parts. Examples would be:

- Audio Interface
- Braille Interface
- eBook Format Interface

4.6.2 Compliance Levels

When it comes to legal compliance OpenLaws could be lifted to a technology level T3 or T4 as described in the beginning of section 4. Companies often find themselves in a very complex legal structure where most of the people do not catch on – even experts.

By analysing company settings you can build up graph relations. It is possible with the Neo4j graph database to build up those relations without changing the base architecture of OpenLaws. Having that company graph the legal situation can be analysed in detail and monitored for future changes.

Many hidden legal constraints, requirements or problematic issues beyond current means could get visible.

Offering this new facility for uncovering, OpenLaws also can be utilized for being compliant to specific standards and to which extend. All factors are clearly defined and can be accounted for.

4.6.3 Network Analysis and Clustering

Graph database builds the core of OpenLaws. Therefore – having a network of nodes – network analyses are highly useful. Processing the existing node-web as well as building up new linkages can be part of that analyses.

Traditional databases only have references to other entities over (foreign) keys whereas graph databases have "real" relations where the focus on the reference is as high as the focus on the node itself. The huge benefit of graph databases consequently is to determine connections and inter linkages.

This concept allows easily to group for different topics or create and visualize groups or clusters (see Figure 15: EUR-Lex cluster).

Network analyses also can be part of a selected portion of graph references to build up different clusters regarding different topics or meeting other requirements.

4.6.4 Correlation with EU-Laws

EU-Law has a lot of directives and recommendations for national legislation. But actually at the moment there is no clear indicator if the different EU-Countries abide to all EU-laws or not. Only legal experts on EU-law are able to judge if a specific EU-law has been implemented in national legislation or not.

The reason for that is that EU-law/national-law relation is not one to one. Each country with their complex legislation has to include the different topics into their laws and regulations. As the foundations are quite different the resulting legal written codex also will be.

When reading a country based law parts of an EU-law might recognizable but other parts might be implemented in completely different paragraphs with different legal code §.

Consequently natural language processing and/or applying knowledge graphs it might be possible in future to identify which part of an EU-law/regulation has been implemented in which part of national law.

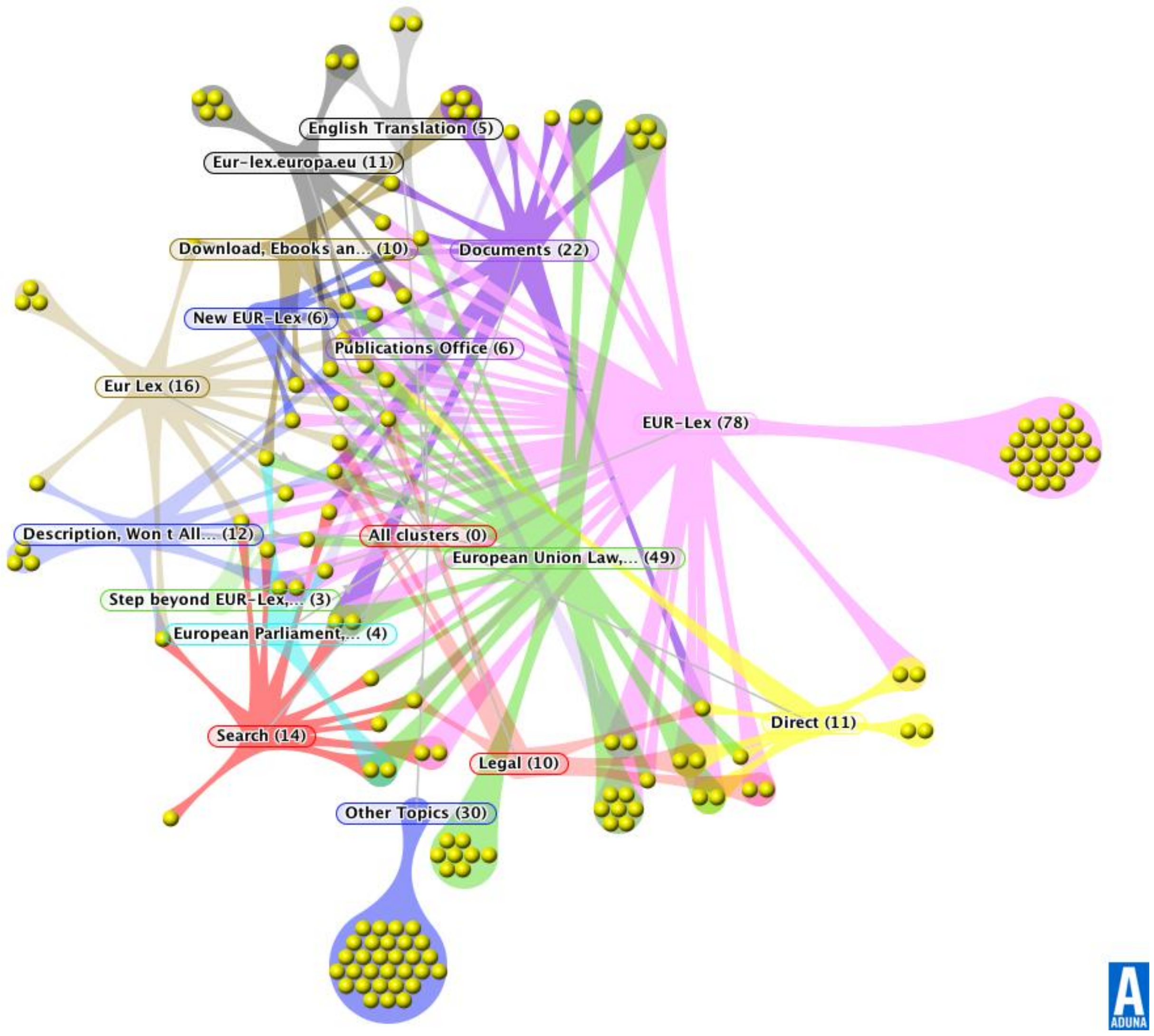


Figure 15: EUR-Lex cluster

This is technically a very challenging task because it is necessary to determine accurately the topics and how the specific law is implementing the law. Further an algorithm needs to be able to compare this results with other documents to tell if

- the topic is the same
- and resulting law implementation characteristic is the same or similar to a specific extend regarding that topic

Today it is possible to quite accurately determine a topic of a document, but reading the fine granular structure of legal content with delicate and difficult interpretable meaning, needs to be categorised as technology in terms of the BOLD Vision (T5+) - see Table 3: Technology categorisation.

4.6.5 Thesauri and Abbreviation Indexes

OpenLaws gives a large base of legal information. EU-law as well as national law with references (maps) in-between them can be included. The large textual base can be part of a thesauri analyses building up a thesauri automatically.

"... we could attempt to generate a thesaurus automatically by analysing a collection of documents. There are two main approaches. One is simply to exploit word co-occurrence. We say that words co-occurring in a document or paragraph are likely to be in some sense similar or related in meaning, and simply count text statistics to find the most similar words. The other approach is to use a shallow grammatical analysis of the text and to exploit grammatical relations or grammatical dependencies. For example, we say that entities that are grown, cooked, eaten, and digested, are more likely to be food items. Simply using word co-occurrence is more robust (it cannot be misled by parser errors), but using grammatical relations is more accurate."[1, 192]

Additionally to [1] in our case we are able to take **references between legal objects** into account which are closely related and improve in this way the quality of the generated thesauri.

Similar approaches can be suitable for achieving automatic generation of an **abbreviation index**. Except that a computational logic needs to determine if the letter sequence represents an abbreviation or a whole word. Thus the abbreviation index might be a by-product of thesauri generation.

4.6.6 Extended Folders – Codex Creation

This enrichment tool aims at supporting users to generate their own book-style codex for their personal use. Users can choose whether they would like to generate the codex with all their highlights and annotations or only with each of them or just blank collections of legal items of their choice.

4.6.7 Open Data

Open data is a rich source of public sector information and will grow in quantity and quality over time. Thus data portals like www.data.gv.at offer a lot of potential in terms of enrichment possibilities which are not yet discovered. As OpenLaws offers a central easy searchable and extendable database of legal information of the whole legislation of many countries, Meta information of open data could add a lot of aspects to several laws we did not think of.

This might be even relevant when new laws are necessary to establish or current legal parts are adopted to modern society. Then it is possible to take many different aspects into account before actually changing something.

6 References

1. Baeza-Yates, R., & Ribeiro-Neto, B. (1999). *Modern information retrieval* (Vol. 463). New York: ACM press.

2. D.M. Blei, A.Y. Ng & M.I. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research 3, 2003*, pp. 993-1022.

3. A. Boer and T. van Engers. A MetaLex and metadata primer: Concepts, use, and implementation. In Legislative XML for the Semantic Web, pages 131–149. Springer, 2011.

4. R.L. Breiger. The duality of persons and groups. In: B. Wellman, S. Berkowitz (Eds.), Social Structures: Network approach, Cambridge Univ. Press, Cambridge (1988), pp. 83–98.

5. Lingo 3G search clusters, https://carrotsearch.com/

6. K. G. Saur. Functional requirements for bibliographic records. *UBCIM Publications - IFLA Section on Cataloguing*, 19, 1998.

7. C. Sutton & A. McCallum (2011). An Introduction to Conditional Random Fields. *Foundations and Trends in Machine Learning, 4*, 4:267-373.

8. R.G.F. Winkels, A. Boer, B. Vredebregt & A. van Someren. Towards a Legal Recommender System. In R. Hoekstra (ed). *JURIX 2014*. *Volume 271 of Frontiers in Artificial Intelligence and Applications*, IOS Press, Amsterdam, pp. 169-178. Best paper award.