Information Extraction using PDT Tools and Inductive Logic Programming

Jan Dědek

Department of Software Engineering, Faculty of Mathematics and Physics,
Charles University in Prague, Czech Republic
dedek@ksi.mff.cuni.cz

Aplikace NLP, 31. 3. 2011, MFF UK, Praha

Outline

Information Extraction Task

- The Task of Information Extraction
  - Automatically find the information you're looking for.
  - Pick out the most useful bits.
  - Present it in preferred manner, at the right level of detail.

  - Closely related:
    labeling of mentions in text $\approx$ text annotation

"Axis" of Information Extraction

- Information depth
  - "Document labeling"

    ```
    The event started at  half pas six.
                          ↖time_expression
    ```

  - Uniform representation ("Semantic interpretation")

    ```
    The event started at  half pas six.
                          ↖time_expression=18:30
    ```

- Task complexity
  - Entity recognition

    ```
    J. Dědek  is a PhD student at the  Charles Univ.
    ↖Person                            ↖Organization
    ```

  - Relation extraction

    ```
    J. Dědek  is a PhD student at the  Charles Univ.
    ↖Person     ⟶has_affiliation⟶      ↖Organization
    ```

Example of the web-page with a report of a fire department

HZS Jihomoravského kraje

Zubatého 1, 614 00 Brno, telefon 950 630 111,
http://www.firebrno.cz
Zpravodajství v roce 2006

15.05.2007

V trabantu zemřeli dva lidé

K tragické nehodě dnes odpoledne hasiči vyjížděli na silnici z obce Česká do Kuřimi na Brněnsku.

Nehoda byla operačnímu středisku HZS ohlášena ve 13.13 hodin a na místě zasahovala jednotka profesionálních hasičů ze stanice v Tišnově. Jednalo se o čelní srážku autobusu Karosa s vozidlem Trabant 601. Podle dostupných informací trabant jedoucí ve z Brna do Kuřimi zřejmě vyjel do protisměru, kde narazil do linkového autobusu dopravní společnosti ze Žďáru nad Sázavou. Ve zdemolovaném trabantu na místě zemřeli dva muži – 82letý senior a další muž, jehož totožnost zjišťují policisté.

Hasiči udělali na vozidle protipožární opatření a po vyšetření a zadokumentování nehody dopravní policií vrak trabantu zaklesnutý pod autobusem pomocí lana odtrhli. Po odstranění střechy trabantu pak z kabiny vyprostili těla obou mužů. Obě vozidla – trabant i autobus, pak postupně odstranili na kraj vozovky a uvolnili tak jeden jízdní pruh. Únik provozních kapalin nebyl zjištěn. Po 16. hodině pomohli vrak trabantu naložit k odtahu a asistovali při odtažení autobusu. Po úklidu vozovky krátce před 16.30 hod. místo nehody předali policistům a ukončili zásah.

Odkazy

skryj menu

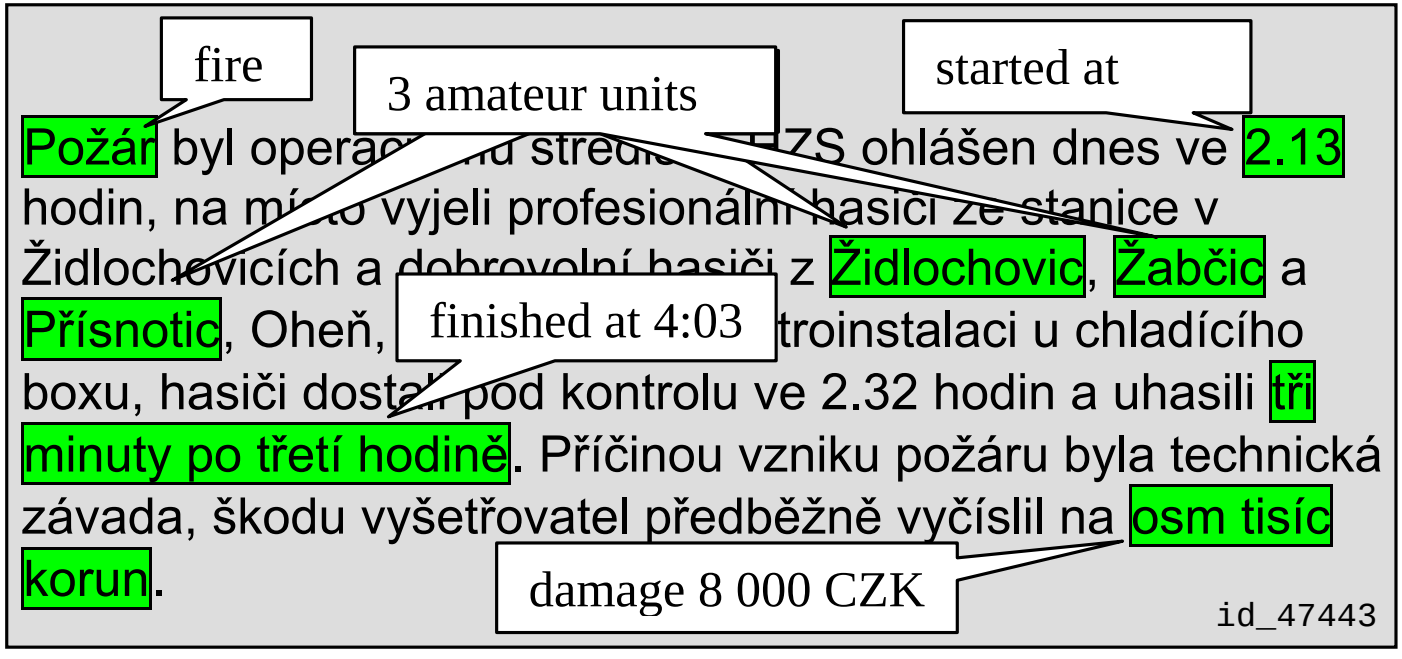
Hasiči
- Generální ředitelství
- hl. m. Praha
- Jihočeský kraj
- Jihomoravský kraj
- Karlovarský kraj
- Královéhradecký kraj
- Liberecký kraj
- Moravskoslezský kraj
- Olomoucký kraj
- Pardubický kraj
- Plzeňský kraj
- Středočeský kraj
- Ústecký kraj
- kraj Vysočina
- Zlínský kraj

Euroskop.cz

V této rubrice Zpravodajství
- Aktualizace stránek
- Archiv zpravodajství
- Bleskové zpravodajství RSS
- Boj proti korupci
- Digitální televize
- Hasiči
- Hlavní zprávy
- Ministerstvo
- Od dopisovatelů (neoficiální)
- Policie
- Regiony
- Servis nejen pro novináře
- Schengenská spolupráce
- WebEditorial

Na našem serveru v jiných rubrikách
- Aktuality Národního archivu

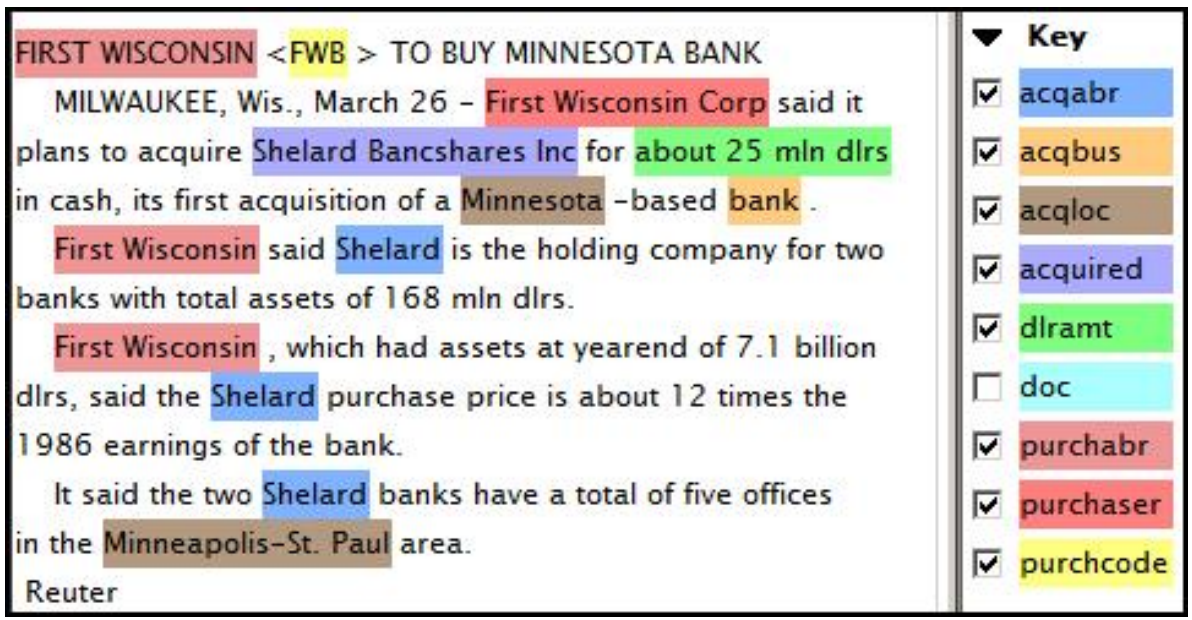
Text of an Accident Report and Contained Information



- Information to be extracted is decorated.

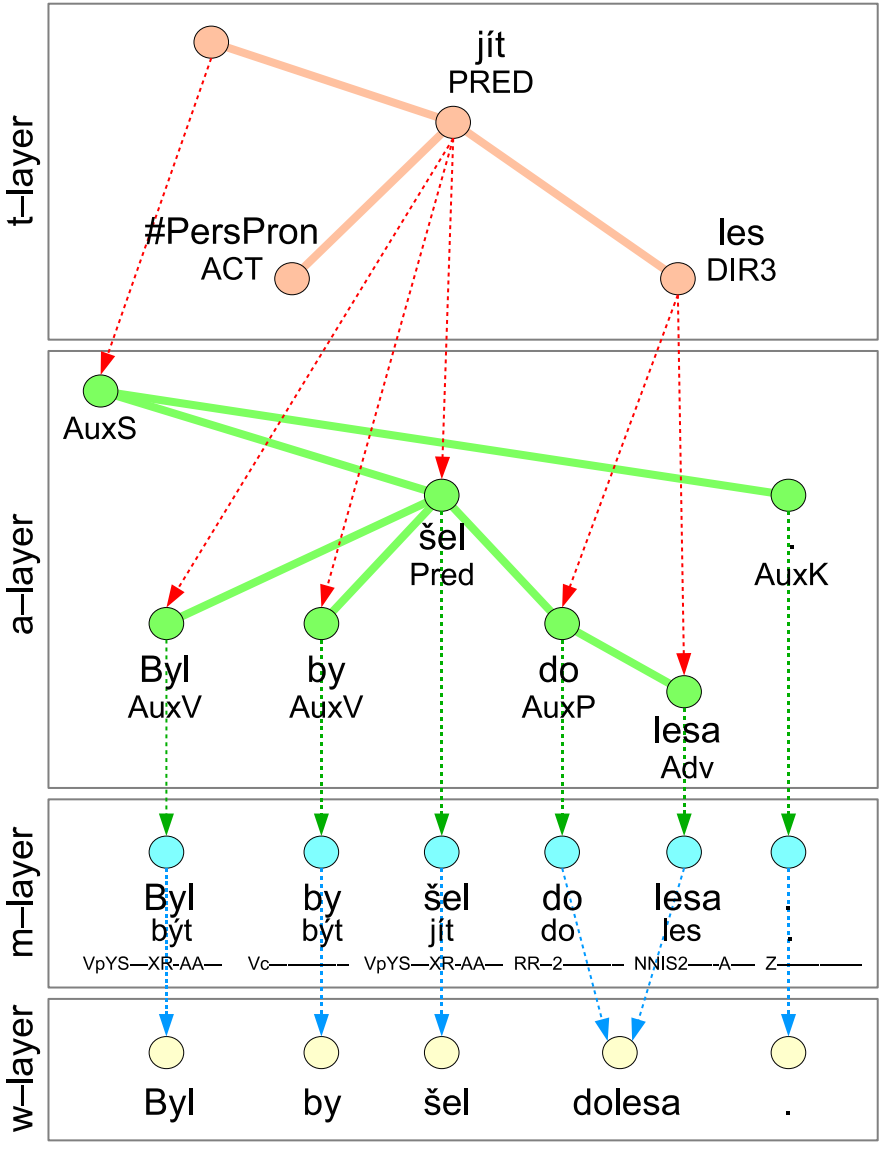
Acquisitions Corpus

- Corporate Acquisition Events
- Acquisitions v1.1 version[1]



[1] from the Dot.kom project's resources:
`http://nlp.shef.ac.uk/dot.kom/resources.html`

Layers of linguistic annotation in PDT



- Tectogrammatical layer
- Analytical layer
- Morphological layer
- PDT 2.0 on-line:

`http://ufal.mff.cuni.cz/pdt2.0/`

Sentence:

Byl by šel dolesa.
He-was would went toforest.

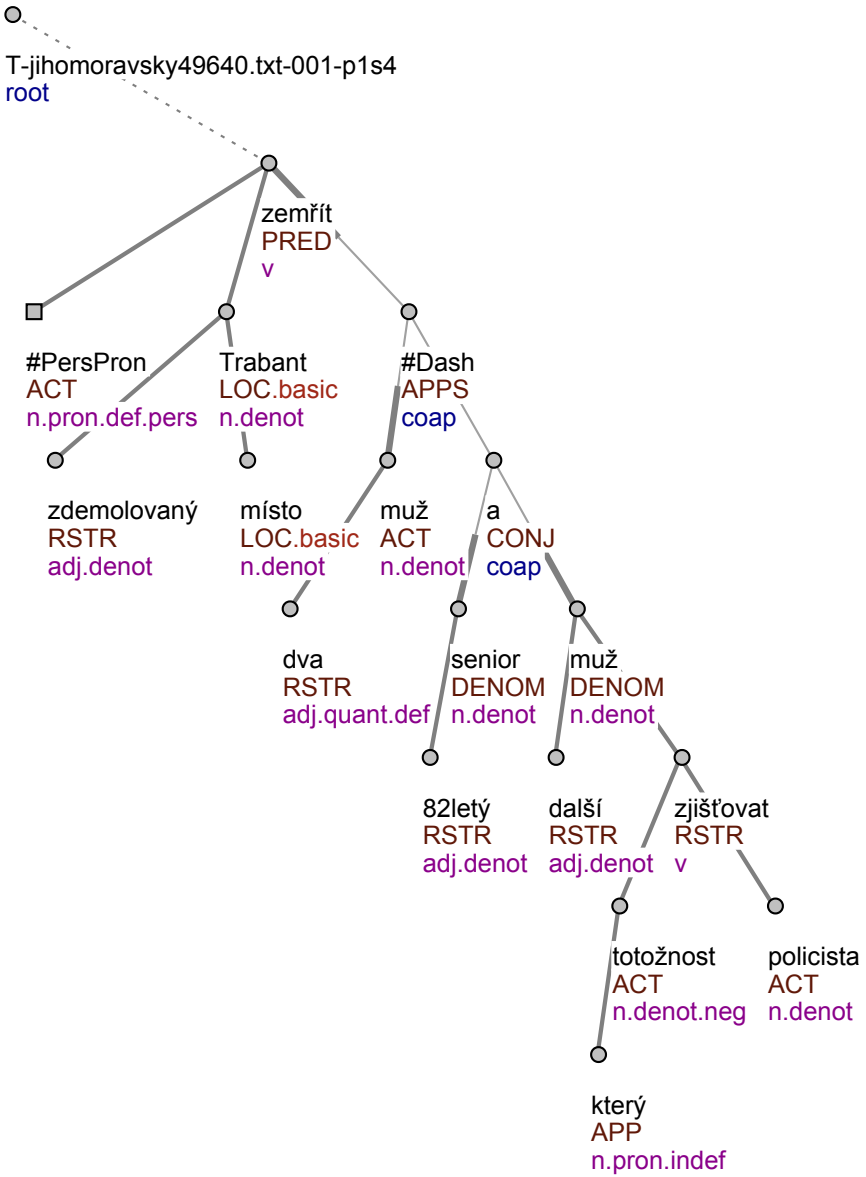
Tools for machine linguistic annotation

1. Segmentation and tokenization
2. Morphological analysis
3. Morphological tagging
4. McDonnald's Maximum Spanning Tree parser
   – Czech adaptation
5. Analytical function assignment
6. Tectogrammatical analysis
   – Developed by Václav Klimeš

- Available within the TectoMT[2] project

[2] http://ufal.mff.cuni.cz/tectomt/

Example of an output tectogrammatical tree



- Lemmas
- Functors
- Semantic parts of speech

Sentence:

Ve zdemolovaném trabantu na místě zemřeli dva muži – 82letý senior a další muž, jehož totožnost zjišťují policisté.

Two men died on the spot in demolished trabant – . . .

Netgraph

- `http://quest.ms.mff.cuni.cz/netgraph/`

- PML Tree Query
- Query Engine and Query Language for TreeBanks
- `http://ufal.mff.cuni.cz/~pajas/pmltq/`

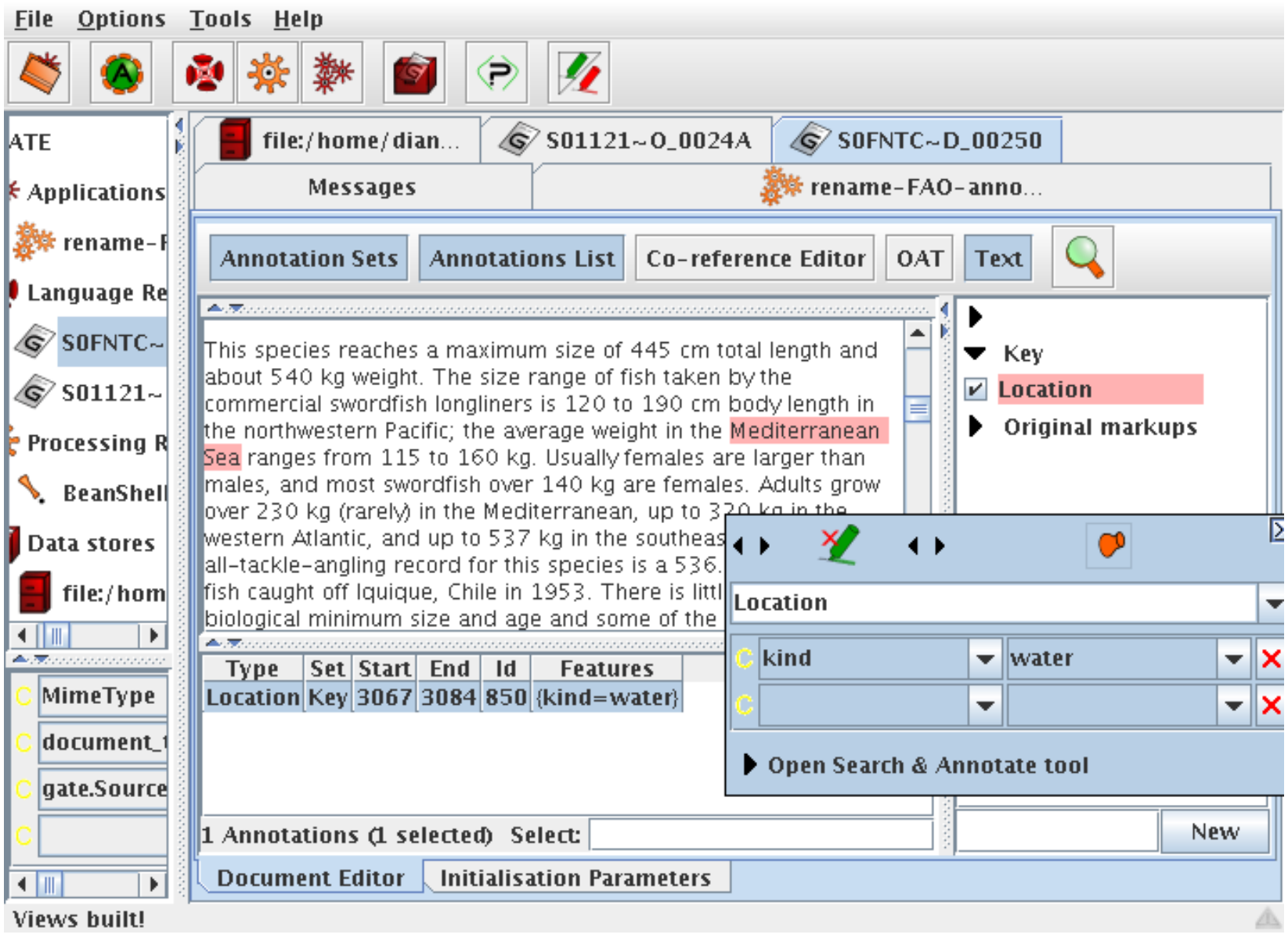
GATE info

- General Architecture for Text Engineering
- University of Sheffield, UK
- Natural Language Processing (NLP)
- Information Extraction (IE)
- Text Annotation
- Developed in Java
- `http://gate.ac.uk/`

GATE features

- Document and annotation management
- Language and processing utility resources
  - Taggers, Parsers, Coreference-processors, Named entity recognizers, Alignment tools, WordNet, Yahoo search, etc
- JAPE grammar rules
- Performance evaluation tools
- Machine learning facilities
  - `http://gate.ac.uk/sale/talks/gate-course-aug10/track-3/module-11-machine-learning/`
  - Slides: `module-11.pdf`
- Ontology support

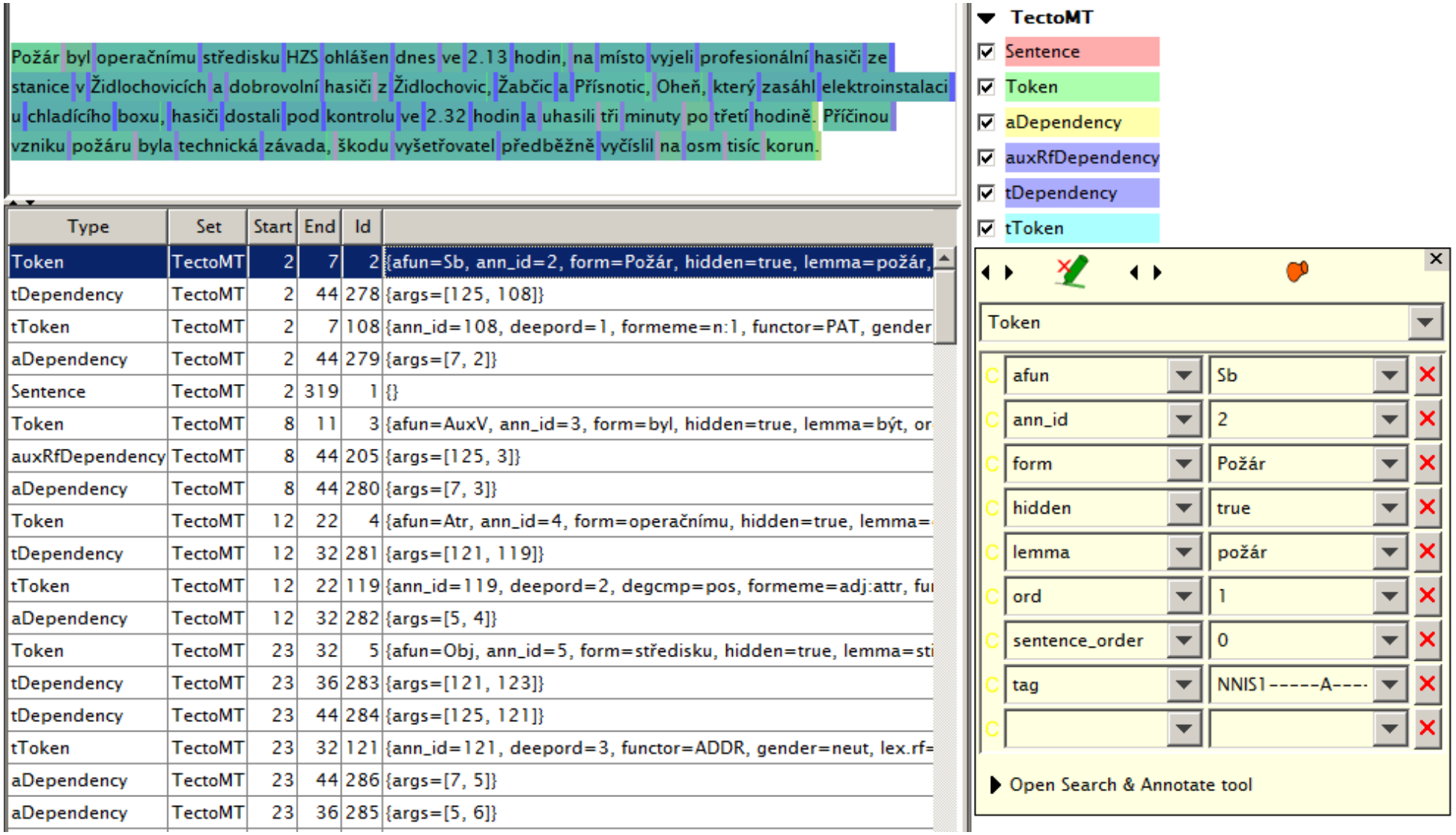
GATE screen shot



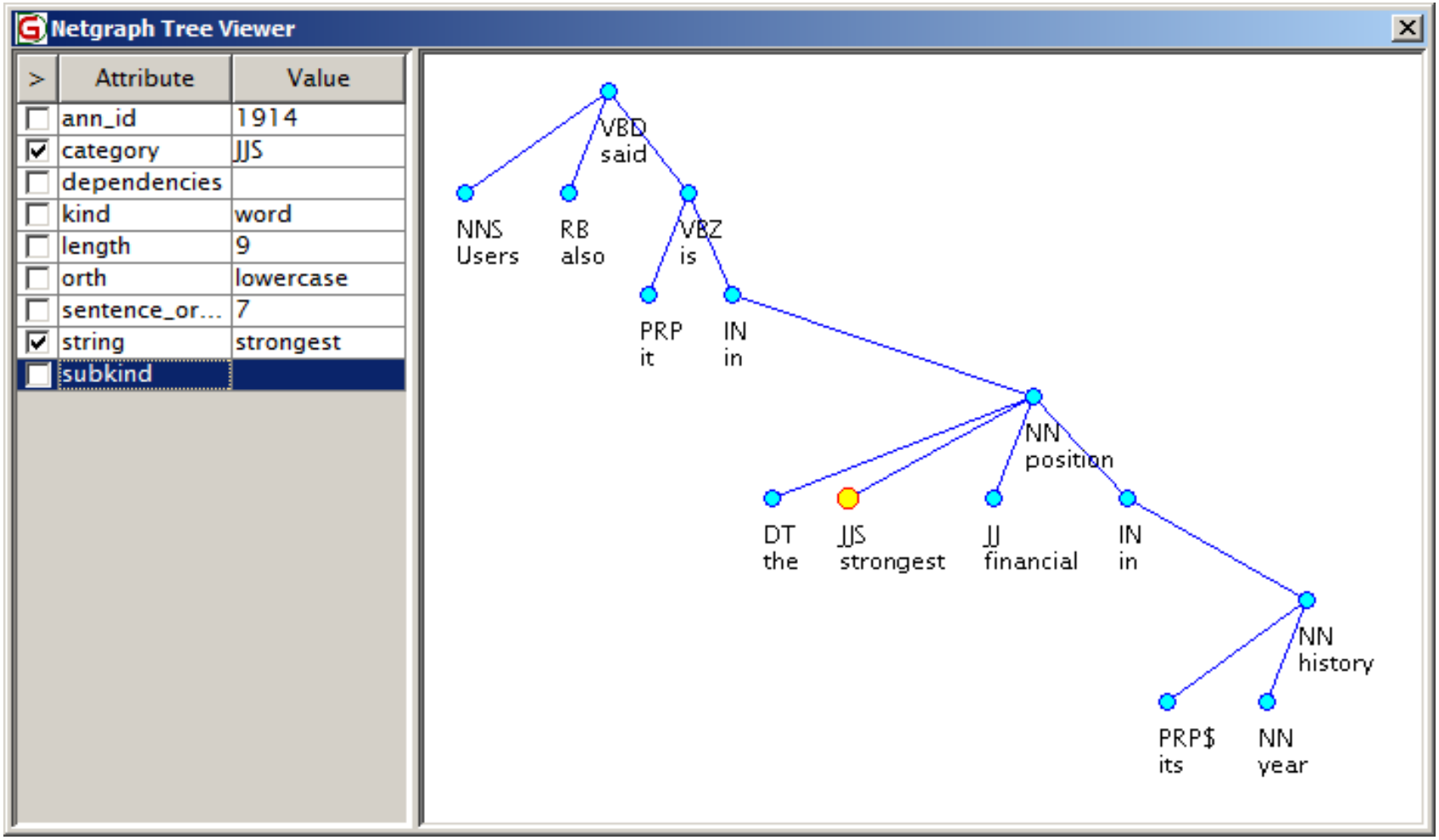
Integration of PDT in GATE

- Implemented Batch TectoMT Language Analyzer
  - Transformation of PDT annotations to GATE
- Netgraph used as a tree viewer
  - Works also for Standford Depndencies
- `http://czsem.berlios.de/`

PDT in GATE

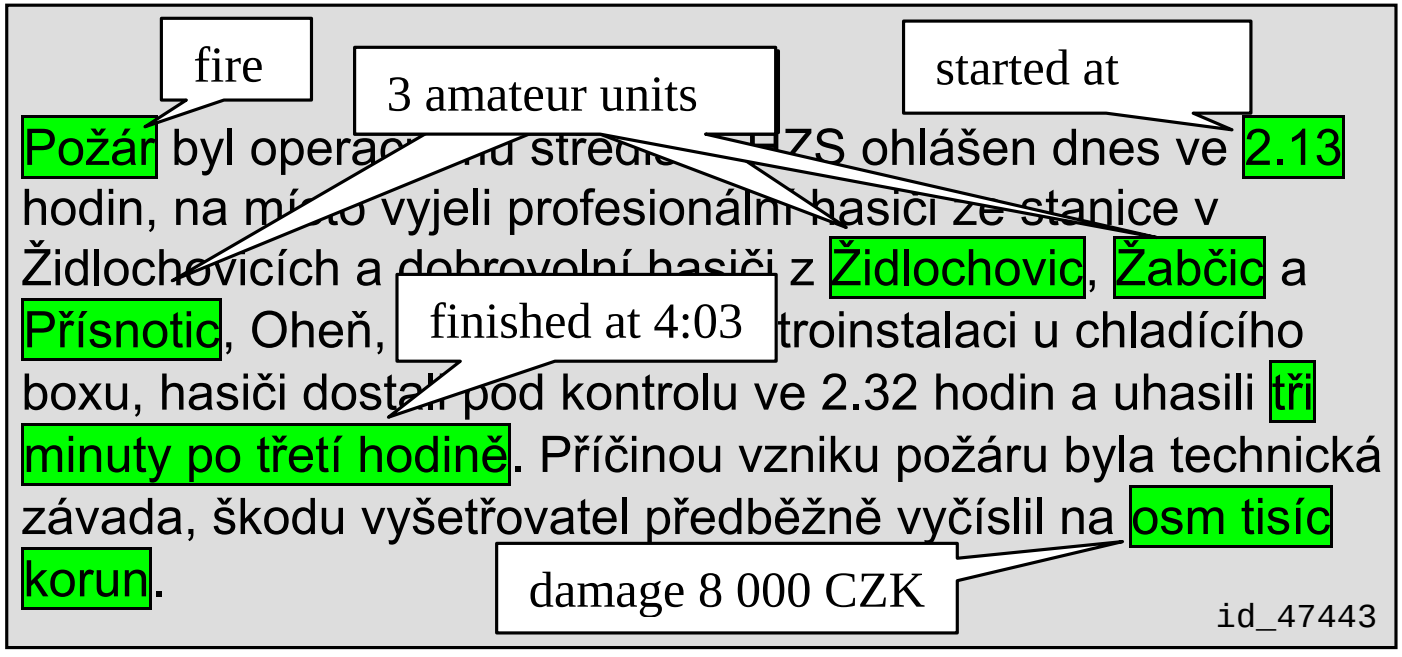

Požár byl operačnímu středisku HZS ohlášen dnes ve 2.13 hodin, na místo vyjeli profesionální hasiči ze stanice v Židlochovicích a dobrovolní hasiči z Židlochovic, Žabčic a Přísnotic, Oheň, který zasáhl elektroinstalaci u chladícího boxu, hasiči dostali pod kontrolu ve 2.32 hodin a uhasili tři minuty po třetí hodině. Příčinou vzniku požáru byla technická závada, škodu vyšetřovatel předběžně vyčíslil na osm tisíc korun.
Type Set Start End Id
Token TectoMT 2 7 2 {afun=Sb, ann_id=2, form=Požár, hidden=true, lemma=požár,
tDependency TectoMT 2 44 278 {args=[125, 108]}
tToken TectoMT 2 7 108 {ann_id=108, deepord=1, formeme=n:1, functor=PAT, gender
aDependency TectoMT 2 44 279 {args=[7, 2]}
Sentence TectoMT 2 319 1 {}
Token TectoMT 8 11 3 {afun=AuxV, ann_id=3, form=byl, hidden=true, lemma=být, or
auxRfDependency TectoMT 8 44 205 {args=[125, 3]}
aDependency TectoMT 8 44 280 {args=[7, 3]}
Token TectoMT 12 22 4 {afun=Atr, ann_id=4, form=operačnímu, hidden=true, lemma=
tDependency TectoMT 12 32 281 {args=[121, 119]}
tToken TectoMT 12 22 119 {ann_id=119, deepord=2, degcmp=pos, formeme=adj:attr, fu
aDependency TectoMT 12 32 282 {args=[5, 4]}
Token TectoMT 23 32 5 {afun=Obj, ann_id=5, form=středisku, hidden=true, lemma=sti
tDependency TectoMT 23 36 283 {args=[121, 123]}
tDependency TectoMT 23 44 284 {args=[125, 121]}
tToken TectoMT 23 32 121 {ann_id=121, deepord=3, functor=ADDR, gender=neut, lex.rf=
aDependency TectoMT 23 44 286 {args=[7, 5]}
aDependency TectoMT 23 36 285 {args=[5, 6]}
TectoMT
Sentence
Token
aDependency
auxRfDependency
tDependency
tToken
Token
afun Sb
ann_id 2
form Požár
hidden true
lemma požár
ord 1
sentence_order 0
tag NNIS1-----A----
Open Search & Annotate tool

Netgraph Tree Viewer in GATE (for Stanford Dependencies)



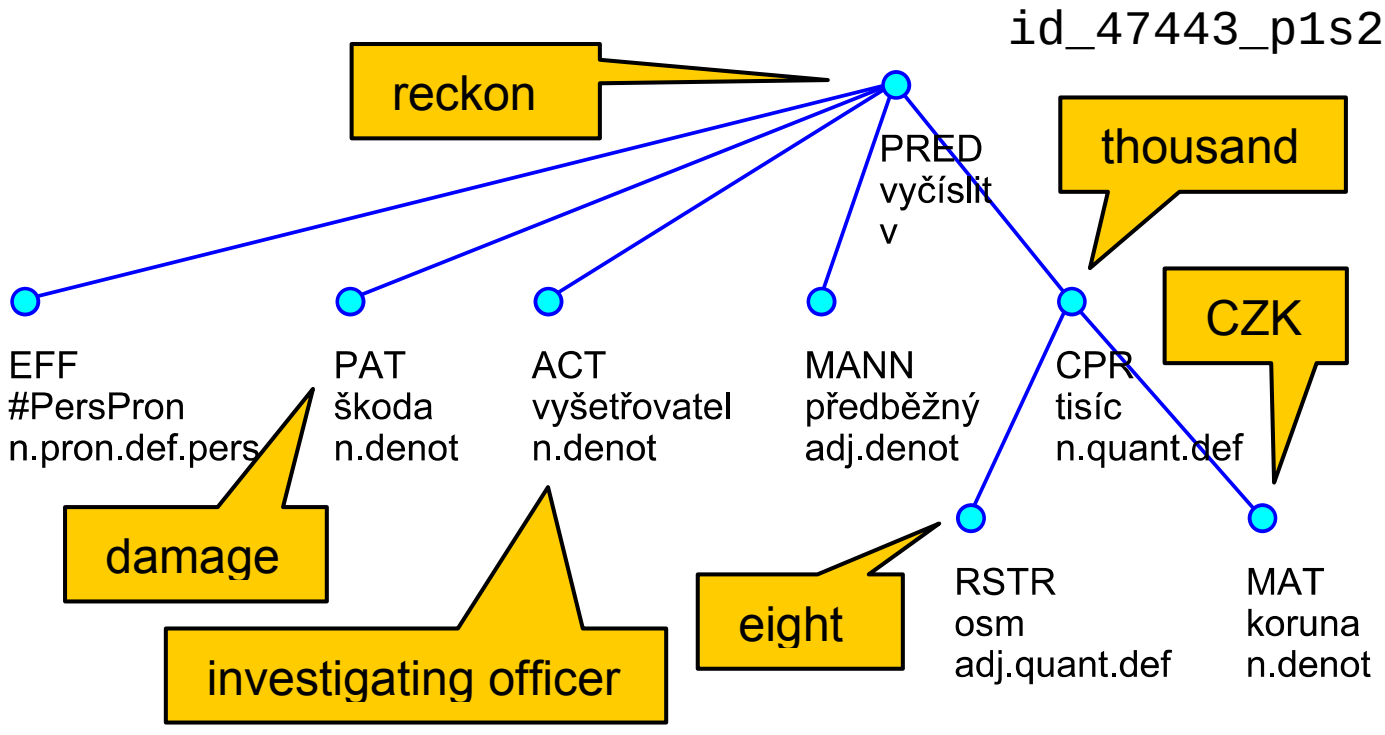
Sentence: Users also said it is in the strongest financial position in its 24-year history.

How to extract the information about the damage of the accident?



- How to extract the information about the damage of the accident?
- See the last sentence on the next slide.

Corresponding linguistic tree



…, škodu vyšetřovatel předběžně vyčíslil na osm tisíc korun.

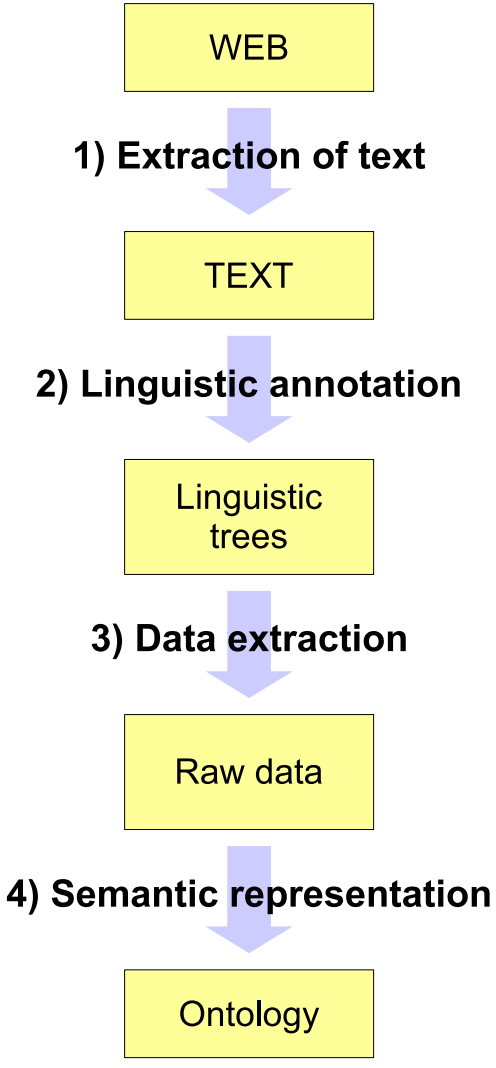
…, investigating officer preliminarily reckoned the damage to be 8 000 CZK.

- Basic Idea: use tree queries (tree patterns) to extract the information.

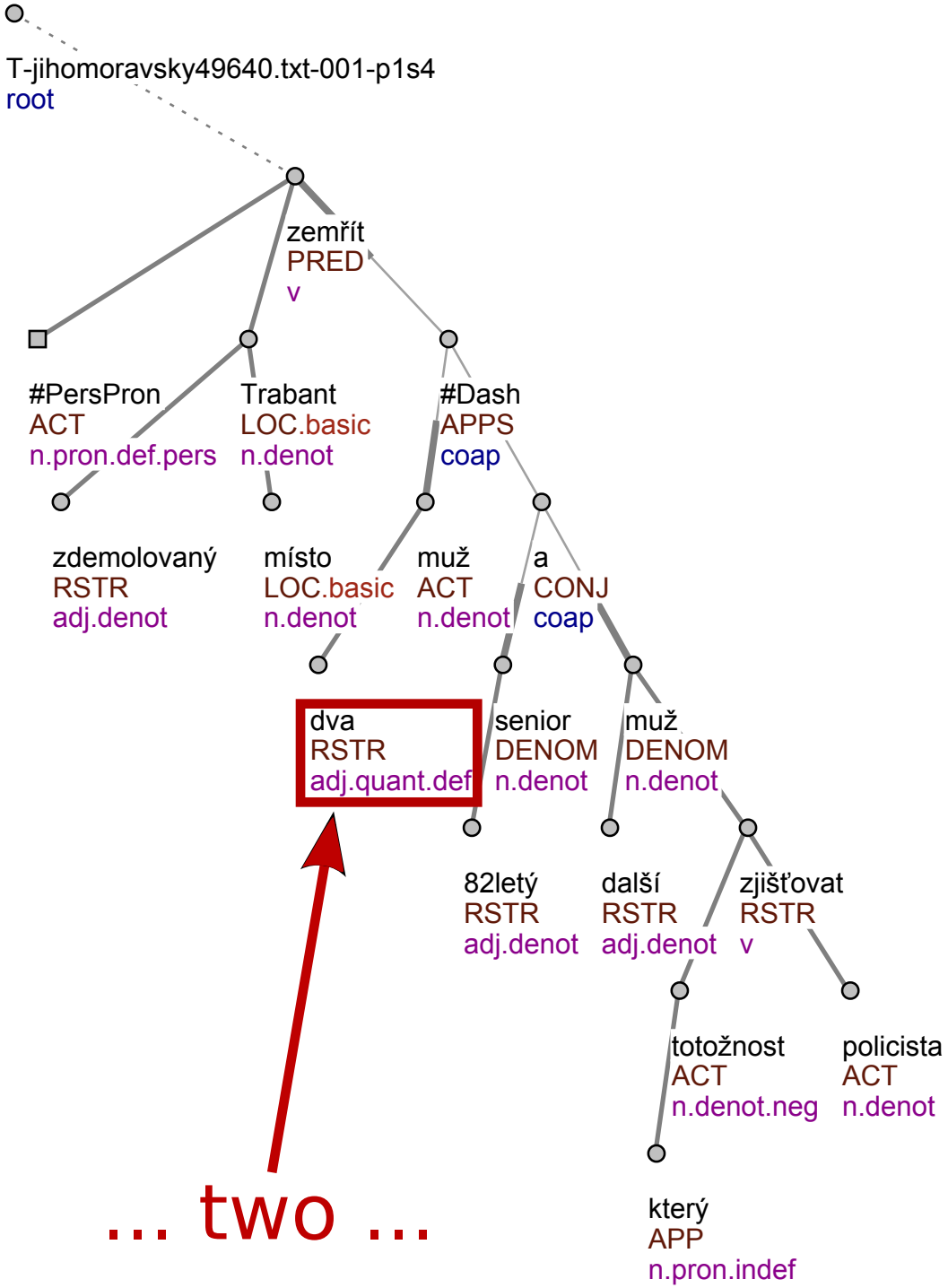
Introduction of Our Solution

- Extraction of semantic information from texts.
- Exploiting of linguistic tools.
  - Mainly “from” the Prague Dependency Treebank project.
    - Related tools – language analyzers (TectoMT), Netgraph, etc.
  - Experiments with the Czech WordNet.
- Rule based extraction method.
  - Extraction rules $\approx$ tree queries
  - ILP learning of extraction rules

Schema of the extraction process

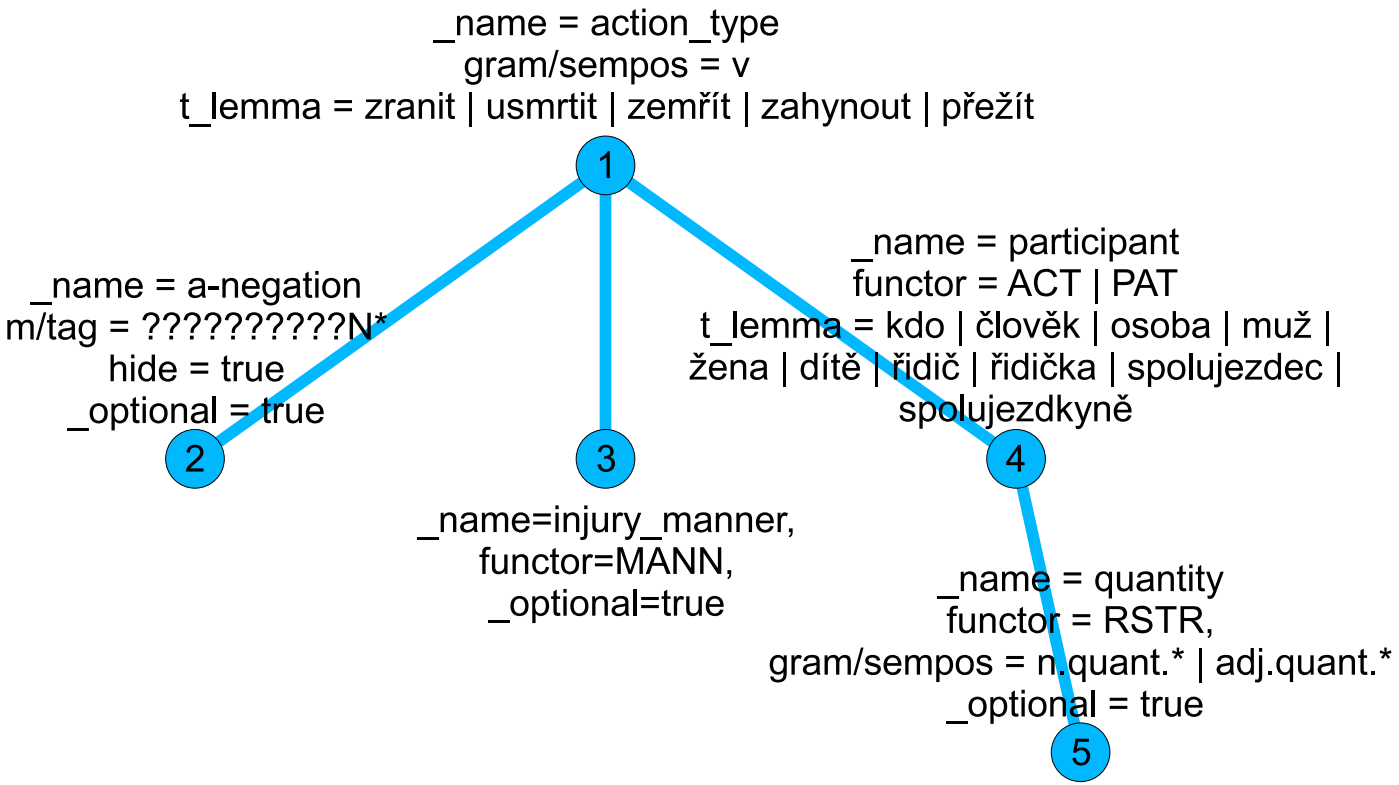


1. Extraction of text
   - Using RSS feed to download pages.
   - Regular expression to extract text.
2. Linguistic annotation
   - Using chain of 6 linguistic tools (see on next slides).
3. Data extraction
   - Exploitation of linguistic trees.
   - Using extraction rules.
4. Semantic representation of data
   - Ontology needed.
   - Semantic interpretation of rules.
   - Far from finished in current state.



- How to extract the information about two dead people?

Extraction rules – Netgraph queries



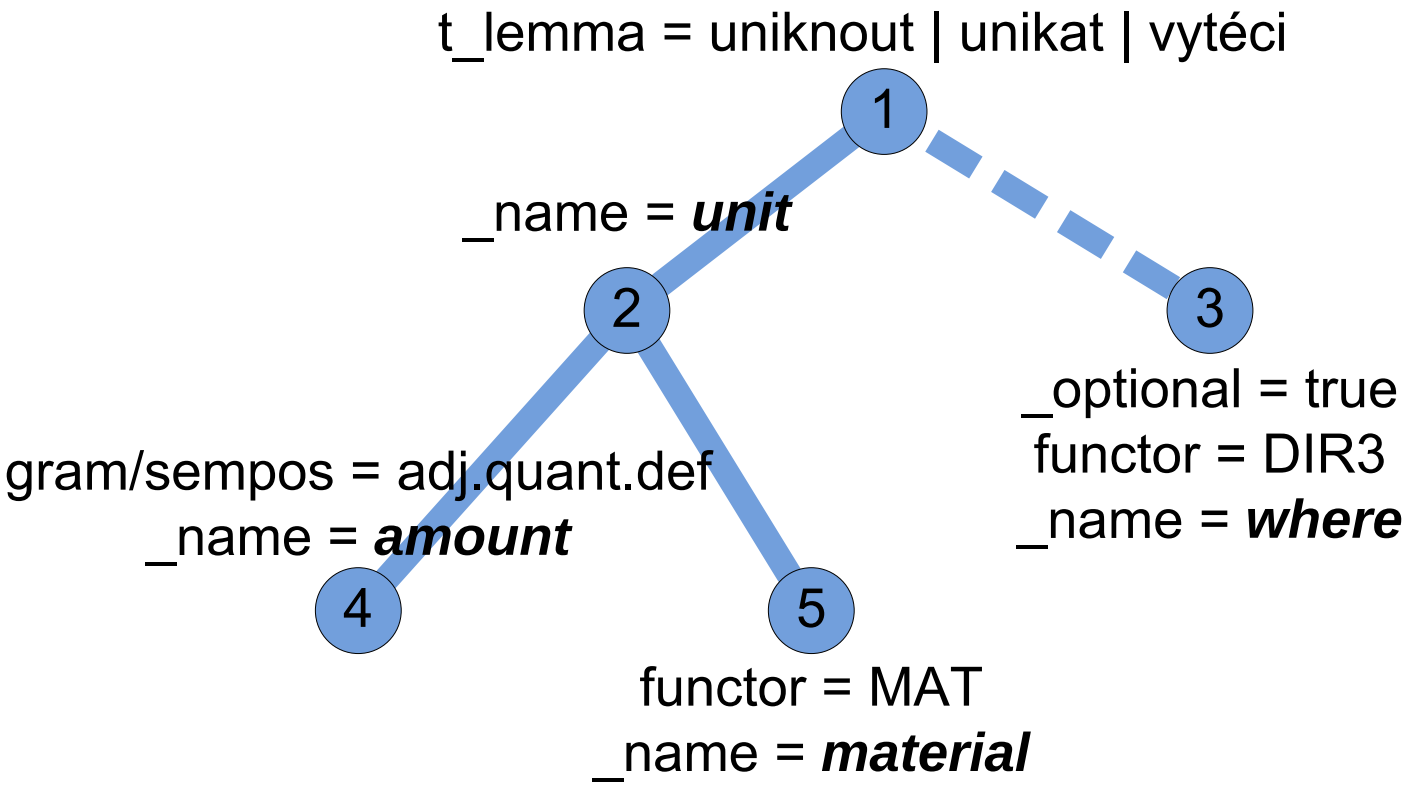
- Tree patterns on shape and nodes (on node attributes).
- Evaluation gives actual matches of particular nodes.
- Names of nodes allow use of references.

Raw data extraction output

```
<QueryMatches>
  <Match root_id="T-vysocina63466.txt-001-p1s4" match_string="2:0,7:3,8:4,11:2">
    <Sentence>
      Při požáru byla jedna osoba lehce zraněna - jednalo se
      o majitele domu, který si vykloubil rameno.
    </Sentence>
    <Data>
      <Value variable_name="action_type" attribute_name="t_lemma">zranit</Value>
      <Value variable_name="injury_manner" attribute_name="t_lemma">lehký</Value>
      <Value variable_name="participant" attribute_name="t_lemma">osoba</Value>
      <Value variable_name="quantity" attribute_name="t_lemma">jeden</Value>
    </Data>
  </Match>
  <Match root_id="T-jihomoravsky49640.txt-001-p1s4" match_string="1:0,13:3,14:4">
    <Sentence>
      Ve zdemolovaném trabantu na místě zemřeli dva muži - 82letý senior
      a další muž, jehož totožnost zjišťují policisté.
    </Sentence>
    <Data>
      <Value variable_name="action_type" attribute_name="t_lemma">zemřít</Value>
      <Value variable_name="participant" attribute_name="t_lemma">muž</Value>
      <Value variable_name="quantity" attribute_name="t_lemma">dva</Value>
    </Data>
  </Match>
  <Match root_id="T-jihomoravsky49736.txt-001-p4s3" match_string="1:0,3:3,7:1">
    <Sentence>Čtyřiatřicetiletý řidič nebyl zraněn.</Sentence>
    <Data>
      <Value variable_name="action_type" attribute_name="t_lemma">zranit</Value>
      <Value variable_name="a-negation" attribute_name="m/tag">VpYS---XR-NA---
      </Value>
      <Value variable_name="participant" attribute_name="t_lemma">řidič</Value>
    </Data>
  </Match>
</QueryMatches>
```

SELECT action_type.t_lemma, a-negation.mtag, injury_manner.t_lemma, participant.t_lemma, quantity.t_lemma **FROM** ****extraction rule****

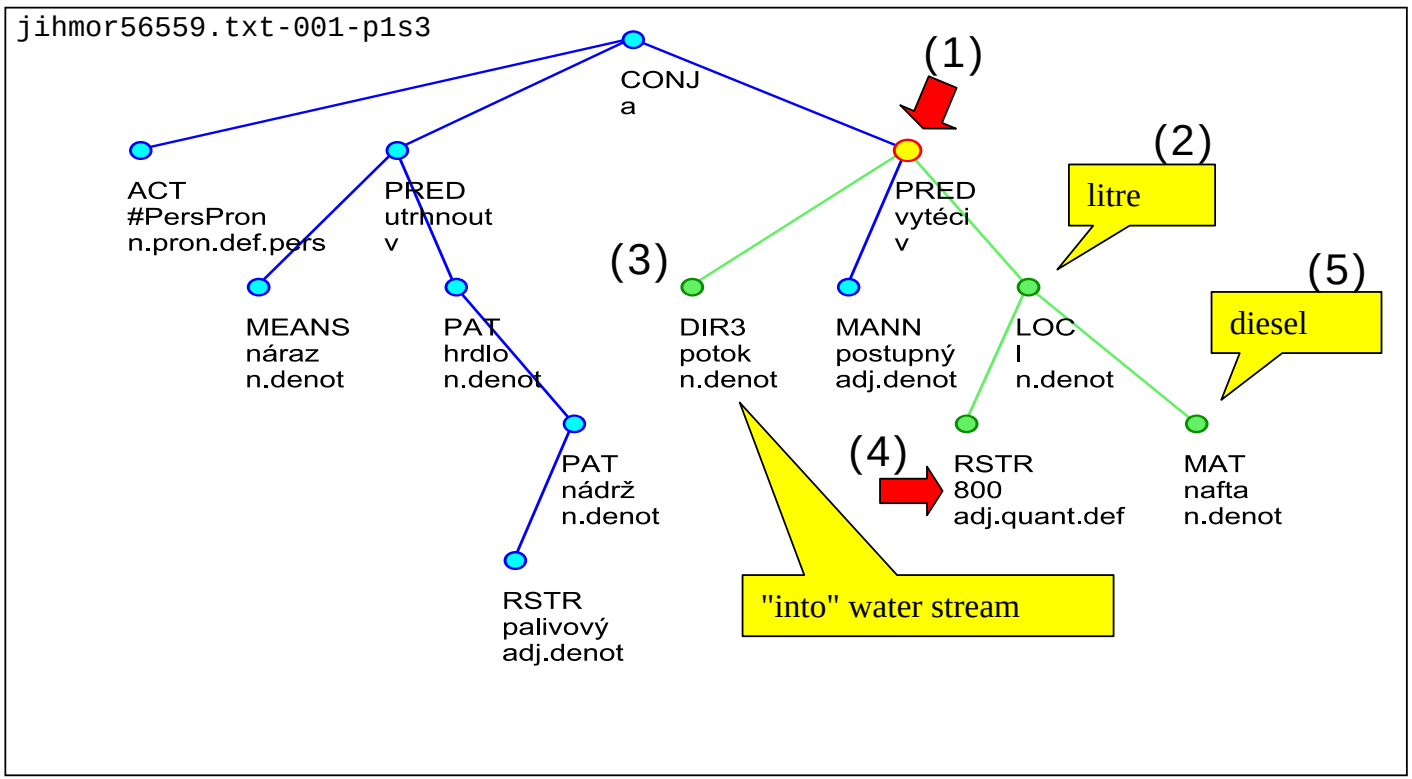
Extraction rules – Environment Protection Use Case



Matching Tree

"Due to the clash the throat of fuel tank tore off and 800 litres of oil (diesel) has run out to a stream."

"Nárazem se utrhl hrdlo palivové nádrže a do potoka postupně vyteklo na 800 litrů nafty."



Raw data extraction output

```
<QueryMatches>
  <Match root_id="jihmor56559.txt-001-p1s3" match_string="15:0,16:4,22:1,23:2,27:3">
    <Sentence>Nárazem se utrhl hrdlo palivové nádrže a do potoka postupně vyteklo na
800 litrů nafty.</Sentence>
    <Data>
      <Value variable_name="amount" attribute_name="t_lemma">800</Value>
      <Value variable_name="unit" attribute_name="t_lemma">1</Value>
      <Value variable_name="material" attribute_name="t_lemma">nafta</Value>
      <Value variable_name="where" attribute_name="t_lemma">potok</Value>
    </Data>
  </Match>
  <Match root_id="jihmor68220.txt-001-p1s3" match_string="3:0,12:4,21:1,22:2,27:3">
    <Sentence>Z palivové nádrže vozidla uniklo do půdy v příkopu vedle silnice zhruba
350 litrů nafty, a proto byli o události informováni také pracovníci odboru životního
prostředí Městského úřadu ve Vyškově a České inspekce životního prostředí.</Sentence>
    <Data>
      <Value variable_name="amount" attribute_name="t_lemma">350</Value>
      <Value variable_name="unit" attribute_name="t_lemma">1</Value>
      <Value variable_name="material" attribute_name="t_lemma">nafta</Value>
      <Value variable_name="where" attribute_name="t_lemma">půda</Value>
    </Data>
  </Match>
...
```

litre

water stream

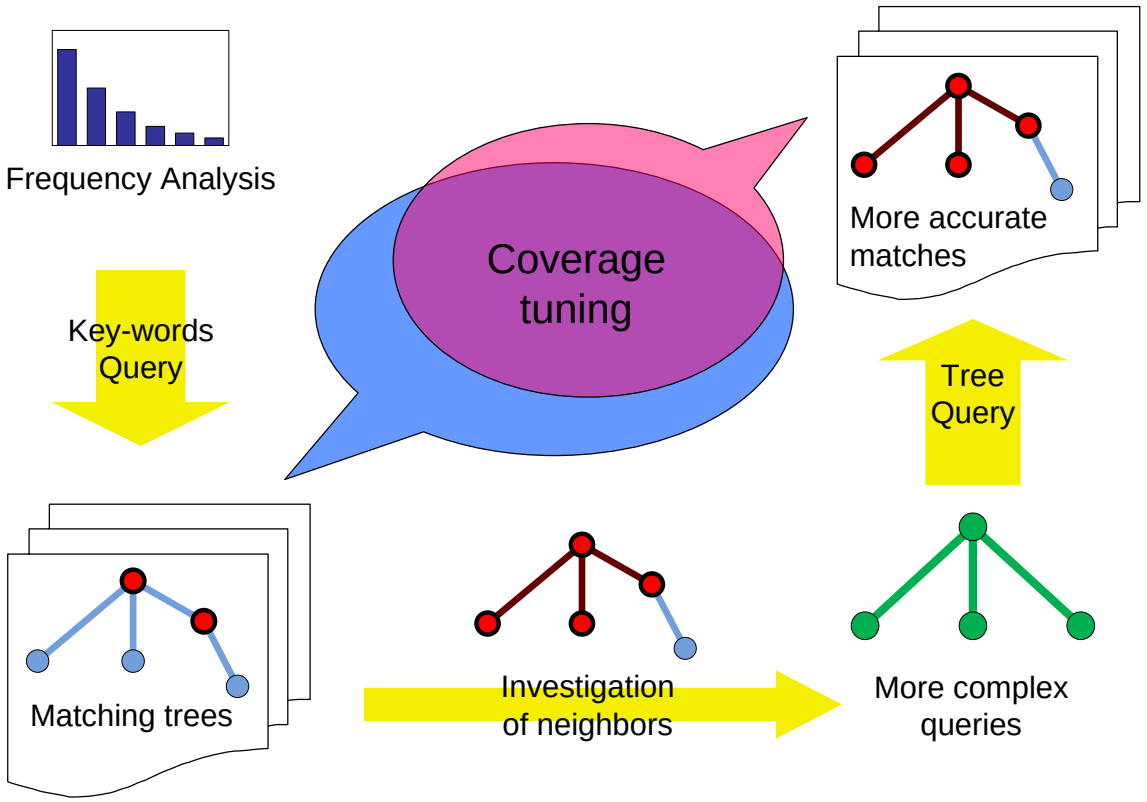
diesel

soil

SELECT amount.t_lemma, unit.t_lemma, material.t_lemma, where.t_lemma

FROM ****extraction rule****

Design of extraction rules – iterative process



1. Frequency analysis → representative key-words.
2. Investigating of matching trees → tuning of tree query.
3. Complexity of the query ≅ complexity of extracted data.

Corpus of Fire-department articles

- Fire-department articles
- Published by The Ministry of Interior of the Czech Republic[3]
- Processed more than 800 articles from different regions of Czech Republic
- 1.2 MB of textual data
- Linguistic tools produced 10 MB of annotations, run time 3.5 hours
- Extracting information about injured and killed people
- 470 matches of the extraction rule, 200 numeric values of quantity (described later)

[3] http://www.mvcr.cz/rss/regionhzs.html

Inductive Logic Programming

- Inductive Logic Programming (ILP)
  - is a Machine Learning procedure for multirelational learning
  - Heuristic and iterative method, learning is usually slow
  - It is capable to deal with graph or tree structures naturally
  - Learns form positive and negative examples
    - Positive and negative tree nodes
    - It is necessary to label tree nodes from corresponding labeled text (not trivial problem)
  - Learned rules are strict (no weights, probabilities, etc.)
    - Easier human understanding, modification
    - Possibility of sharing of rules amongst different tools
    - Lower performance (precision, recall)

ILP principles

- Learning examples $E = P \cup N$ (Positive and Negative)
- Background knowledge B
- ILP task – to find hypothesis H such that:

$$(\forall e \in P)(B \cup H \models e) \ \ \& \ \ (\forall n \in N)(B \cup H \not\models n).$$

ILP Example

Types of ground variables

```
animal(dog). animal(dolphin) ... animal(penguin).
class(mammal). class(fish). class(reptile). class(bird).
covering(hair). covering(none). covering(scales).
habitat(land). habitat(water). habitat(air).
```

Background knowledge

```
has_covering(dog, hair). has_covering(crocodile, scales).
has_legs(dog,4). ... has_legs(penguin, 2). etc.
has_milk(dog). ... has_milk(platypus). etc.
homeothermic(dog). ... homeothermic(penguin). etc.
habitat(dog, land). ... habitat(penguin, water). etc.
has_eggs(platypus). ... has_eggs(eagle). etc.
has_gills(trout). ... has_gills(eel). etc.
```

ILP Example

Positive examples

```
class(lizard, reptile).
class(trout, fish).
class(bat, mammal).
```

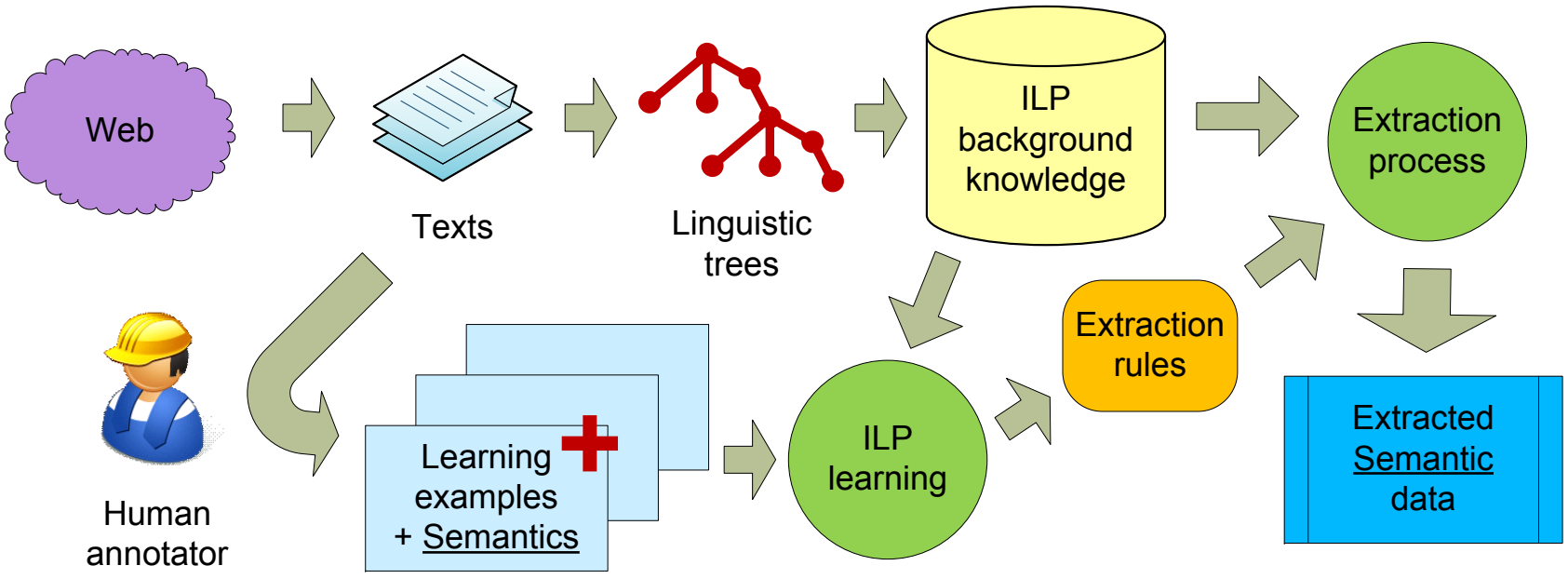
Negative examples

```
class(trout, mammal).
class(herring, mammal).
class(platypus, reptile).
```

Induced rules

```
class(A,reptile) :- has_covering(A,scales),
                    has_legs(A,4).
class(A,mammal) :- homeothermic(A), has_milk(A).
class(A,fish) :- has_legs(A,0), has_eggs(A).
class(A,reptile) :- has_covering(A,scales),
                    habitat(A,land).
class(A,bird) :- has_covering(A,feathers).
```

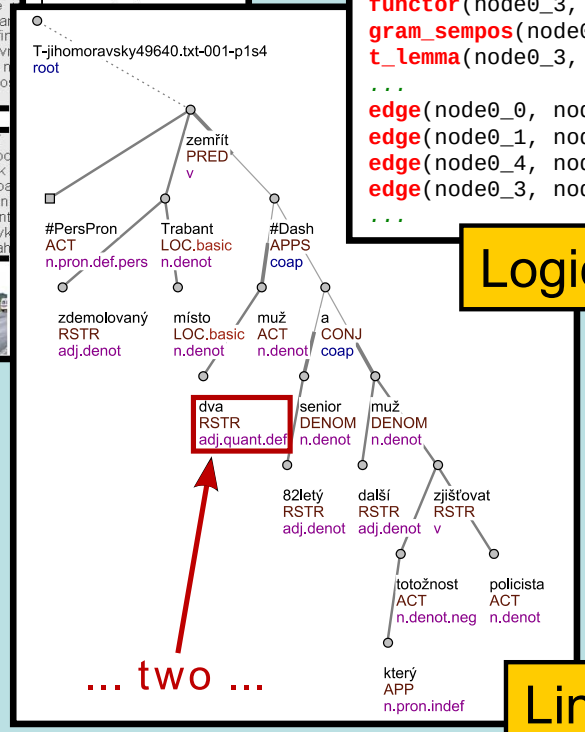
Integration of ILP in our extraction process



- Main point: transformation of trees to logic representation.
- Human annotator does not need to be a linguistic expert.

Logic representation of linguistic trees

Source web page



Linguistic trees

```
tree_root(node0_0). node(node0_0).
id(node0_0, t_jihomoravsky49640_txt_001_p1s4).
%%%%%%%% node0_1 %%%%%%%%%%%%%%%%%%
node(node0_1).
functor(node0_1, pred).
gram_sempos(node0_1, v).
t_lemma(node0_1, zemrit).
%%%%%%%% node0_2 %%%%%%%%%%%%%%%%%%
node(node0_2).
functor(node0_2, act).
gram_sempos(node0_2, n_pron_def_pers).
t_lemma(node0_2, x_perspron).
%%%%%%%% node0_3 %%%%%%%%%%%%%%%%%%
node(node0_3). id(node0_3,
functor(node0_3, loc).
gram_sempos(node0_3, n_denot).
t_lemma(node0_3, trabant).
...
edge(node0_0, node0_1). edge(node0_1, node0_2).
edge(node0_1, node0_3). edge(node0_3, node0_4).
edge(node0_4, node0_5). edge(node0_3, node0_6).
edge(node0_3, node0_7). edge(node0_3, node0_8).
...
```

Logic representation

Linguistic trees in ILP

Types of ground variables

```
token(id_559). token(id_341). token(id_243).
tToken(id_622). tToken(id_630). tToken(id_94).
t_lemmaT(advisor). t_lemmaT(tender). t_lemmaT(earn).
semposT('v'). semposT('n.quant.def'). semposT('n.denot').
functorT('PAT'). functorT('ACT'). functorT('DIR3').
negationT(neg1). negationT(neg0). ...
```

Background knowledge

```
t_lemma(id_622, earn).      t_lemma(id_630, dlr).
functor(id_622, 'PRED').    functor(id_630, 'ACT').
sempos(id_622, 'v').        sempos(id_630, 'n.denot').
negation(id_622, neg0).     number(id_630, pl).
tense(id_622, ant).         ...
tDependency(id_622, id_630). tDependency(id_622, id_623).
...
lex_rf(id_622, id_559).
```

Linguistic trees in ILP

Positive examples

```
mention(acquired,'id_54').
mention(acquired,'id_60').
mention(acquired,'id_13').
```

Negative examples

```
mention(acquired,'id_12').
mention(acquired,'id_13').
mention(acquired,'id_14').
```

Configuration

```
:- mode(1,t_lemma(+tToken,#t_lemmaT)).
:- mode(1,functor(+tToken,#functorT)).
:- mode(1,lex_rf(+tToken,-'Token')).
:- mode(1,lex_rf(-tToken,+'Token')).
:- mode(*,tDependency(+tToken,-tToken)).
:- mode(1,tDependency(-tToken,+tToken)).
:- mode(1,mention(#class_attribute_value,+'Token')).
:- determination(mention/2,t_lemma/2).
:- determination(mention/2,functor/2).
:- determination(mention/2,lex_rf/2).
:- determination(mention/2,tDependency/2).
```

Examples of learned rules – Acqusitions

Example

[Rule 1] [Pos cover = 23 Neg cover = 6]

```
mention_root(acquired,A) :-
   'lex.rf'(B,A), t_lemma(B,'Inc'), tDependency(C,B),
    tDependency(C,D), formeme(D,'n:in+X'), tDependency(E,C).
```

[Rule 11] [Pos cover = 25 Neg cover = 6]

```
mention_root(acquired,A) :-
   'lex.rf'(B,A), t_lemma(B,'Inc'), tDependency(C,B),
    formeme(C,'n:obj'), tDependency(C,D), functor(D,'APP').
```

[Rule 75] [Pos cover = 14 Neg cover = 1]

```
mention_root(acquired,A) :-
   'lex.rf'(B,A), t_lemma(B,'Inc'), functor(B,'APP'),
    tDependency(C,B), number(C,pl).
```

Example Czech fireman data, Czech words are translated.

Example

[Rule 1] [Pos cover = 14 Neg cover = 0]

```
damage_root(A) :- lex_rf(B,A), has_sempos(B,'n.quant.def'),
   tDependency(C,B), tDependency(C,D),
   has_t_lemma(D,'investigator').
```

[Rule 2] [Pos cover = 13 Neg cover = 0]

```
damage_root(A) :- lex_rf(B,A), has_functor(B,'TOWH'),
   tDependency(C,B), tDependency(C,D), has_t_lemma(D,'damage').
```

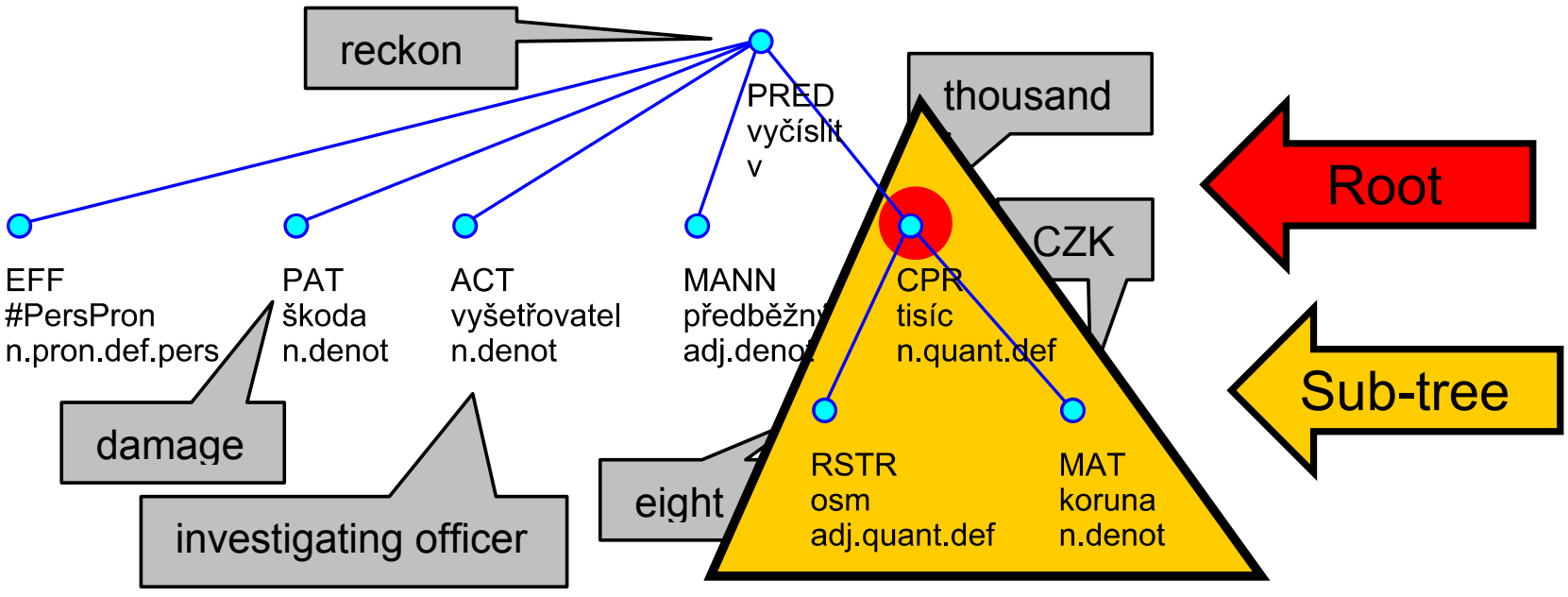
[Rule 1] [Pos cover = 7 Neg cover = 0]

```
injuries(A) :- lex_rf(B,A), has_functor(B,'PAT'),
   has_gender(B,anim), tDependency(B,C), has_t_lemma(C,'injured').
```

[Rule 8] [Pos cover = 6 Neg cover = 0]

```
injuries(A) :- lex_rf(B,A), has_gender(B,anim), tDependency(C,B),
   has_t_lemma(C,'injure'), has_negation(C,neg0).
```

Root/Subtree Preprocessing/Postprocessing (Chunk learning)



…, škodu vyšetřovatel předběžně vyčíslil na osm tisíc korun.

…, investigating officer preliminarily reckoned the damage to be eight thousand Crowns (CZK).

Evaluation results

task/method	matching	missing	excess	overlap	prec.%	recall%	F1.0%
damage/ILP	14	0	7	6	51.85	70.00	59.57
damage/ILP – lenient measures					74.07	100.00	85.11
dam./ILP-roots	16	4	2	0	88.89	80.00	84.21
damage/Paum	20	0	6	0	76.92	100.00	86.96
injuries/ILP	15	18	11	0	57.69	45.45	50.85
injuries/Paum	25	8	54	0	31.65	75.76	44.64
inj./Paum-afun	24	9	38	0	38.71	72.73	50.53

- 10-fold cross validation
- Two tasks: ‘damage’ and ‘injuries’
- Root/subtree preprocessing/postprocessing used for ‘damage’ task

Czsem Mining Suite – the implementation

- Czsem Mining Suite – the implementation
- Contains almost all presented features.
- Web: `http://czsem.berlios.de/`
- Installation instructions:
  `http://czsem.berlios.de/czsem_install.html`

- Caution: TectoMT system is very complex and proper use and installation not trivial, although feasible :-)
- For TectoMT Unix/Linux is platform is strongly recommended.

Summary

- Implemented a system for extraction of semantic information
- Based on third party linguistic tools (TectoMT[4])
- Extraction rules adopted from Netgraph[5] application.
- ILP used for learning rules.
- All methods integrated inside GATE[6].

- Main advantages:
  - Automated selection of learning features
  - “Language independent”
  - Rule based

[4] http://ufal.mff.cuni.cz/tectomt/
[5] http://quest.ms.mff.cuni.cz/netgraph/
[6] http://gate.ac.uk/

Future work

- Use some Knowledge Base (e.g. WordNet).
- Adaptation of this method on other languages.
- Evaluation of the method on other datasets.
- Be able to provide more semantics.
  - e.g. sophisticated semantic interpretation of extracted data

Semantic Web Introduction

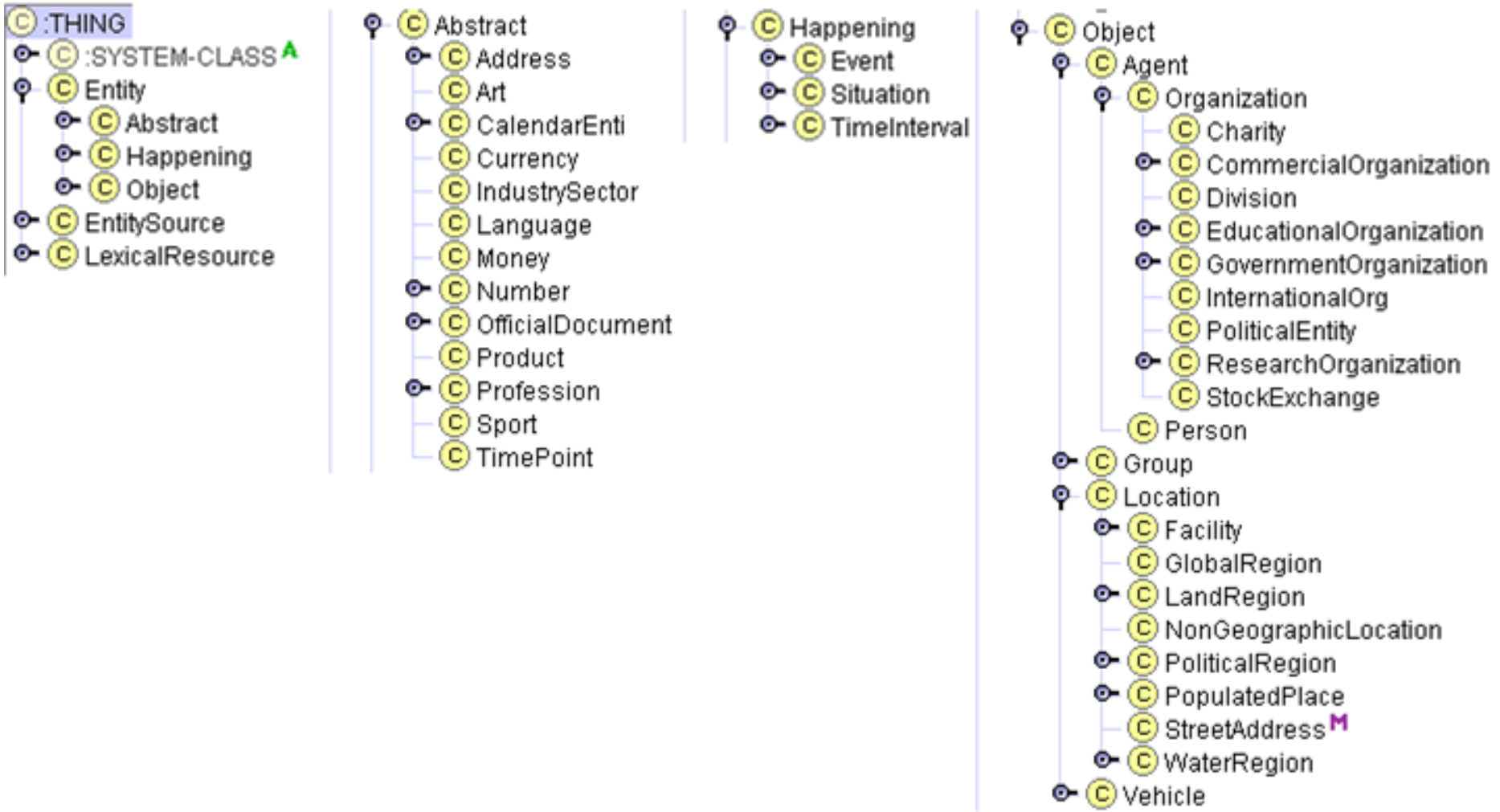
We use semantic web ontologies to express the semantics.

- RDF, OWL languages
- Motivated by description logics

- Concepts or Classes
- Predicates or Relations
- Individuals or Instances

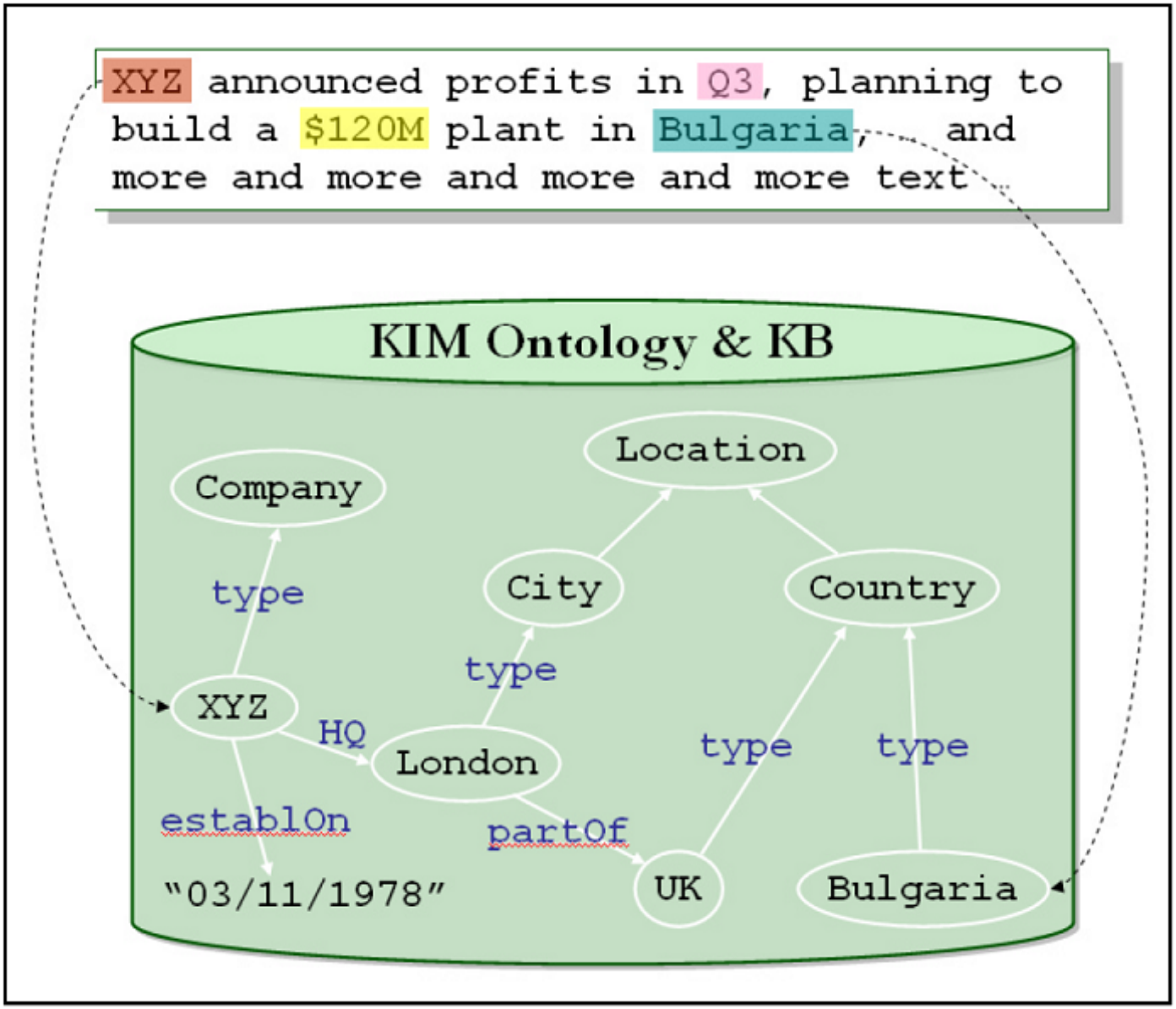
- RDF triples: <Subject> <Predicate> <Object>
- RDF triples form a named oriented graph
  - Basic data structure of the Semantic Web

Ontology (example)

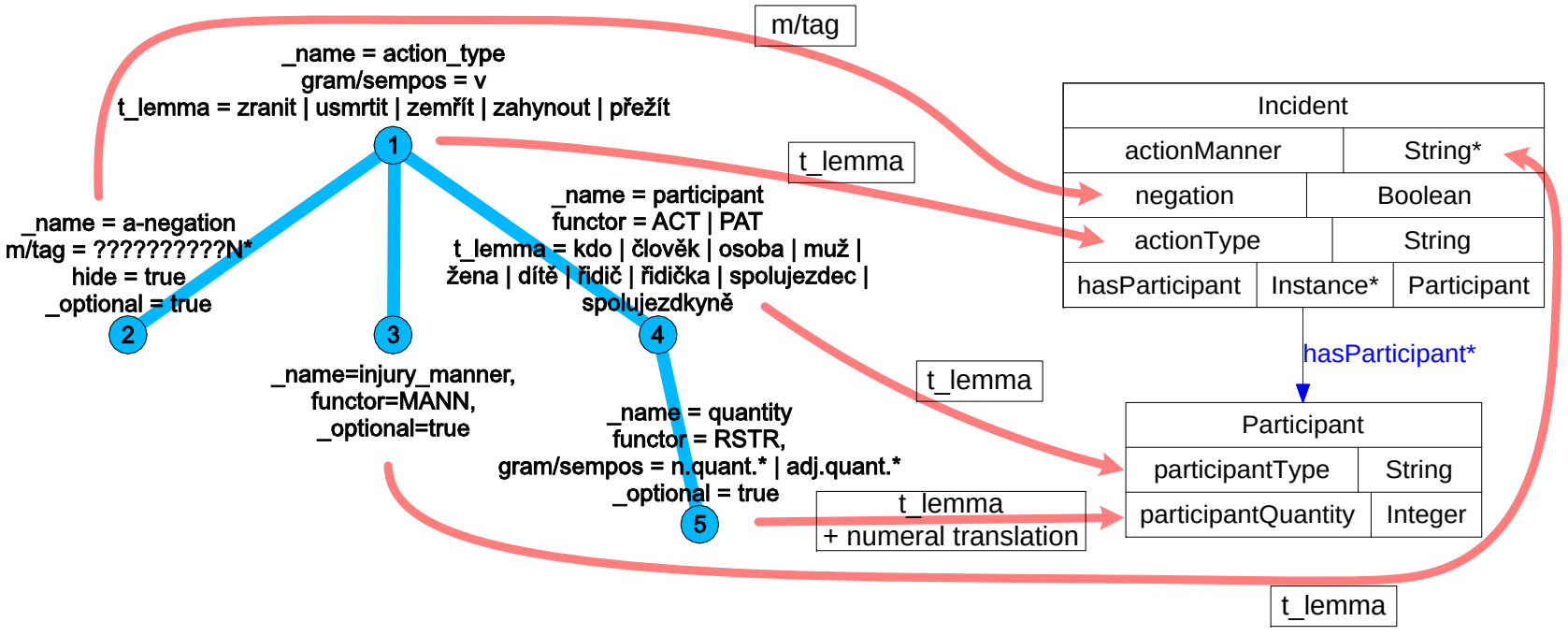


- PROTON (PROTo ONtology)
  `http://proton.semanticweb.org/`

Semantic Annotation (http://www.ontotext.com/kim/)

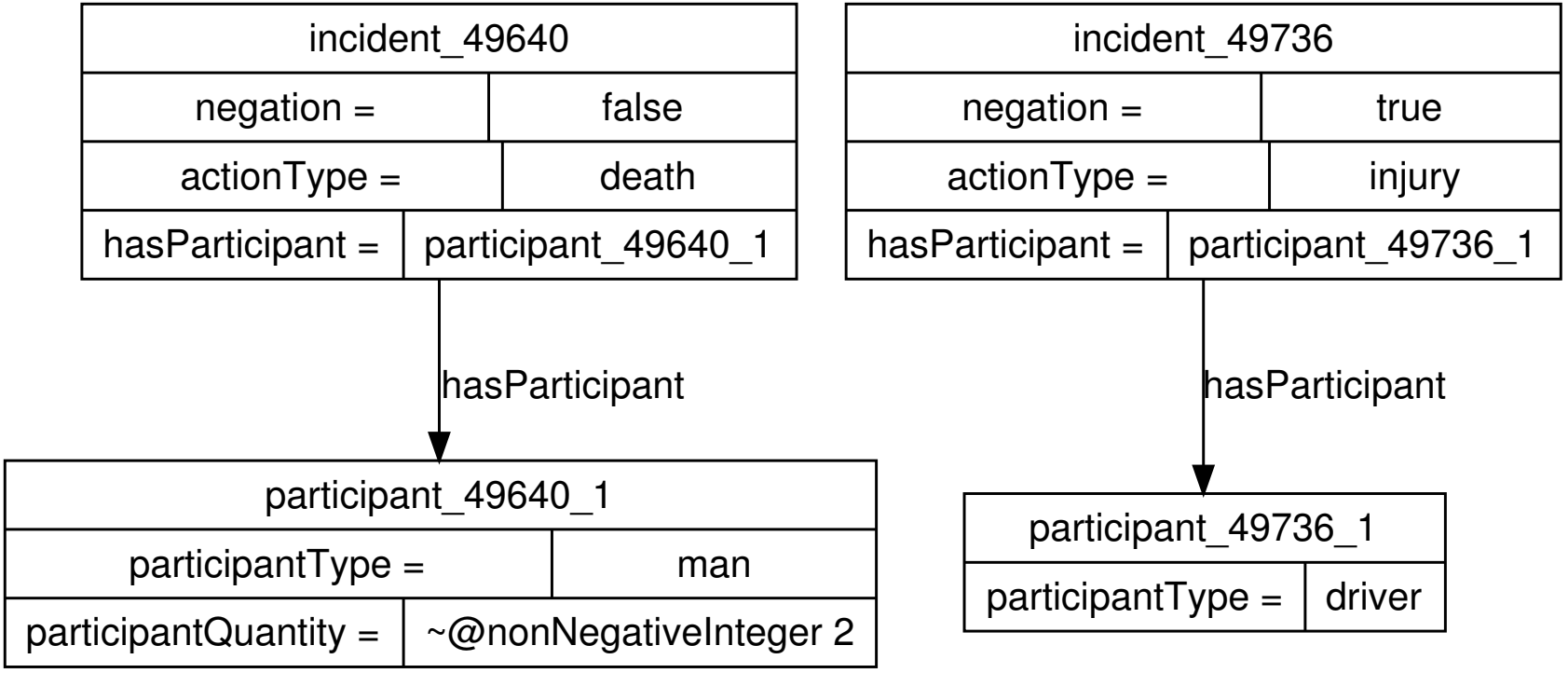


Semantic interpretation of extraction rules



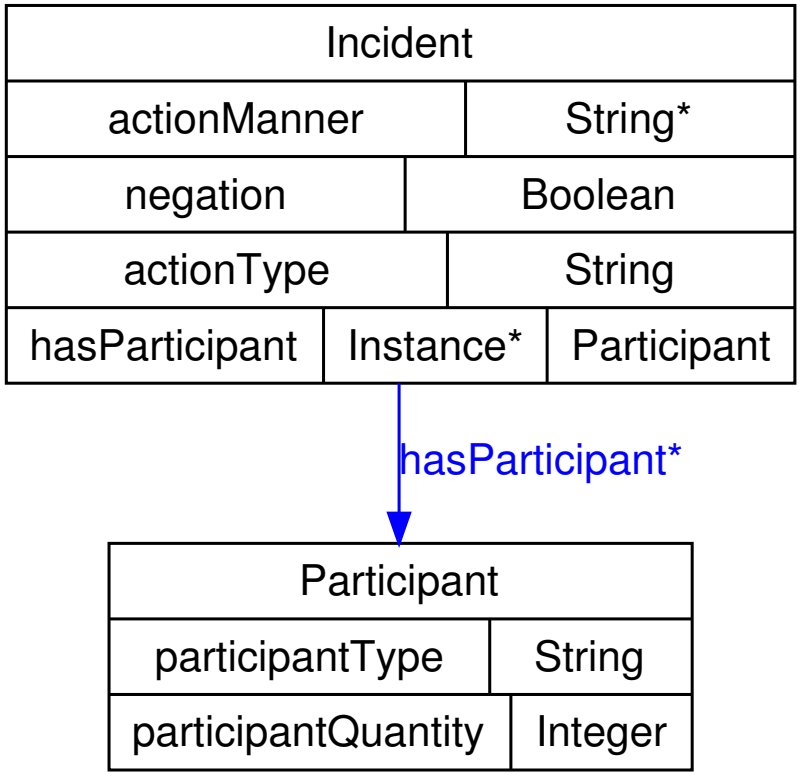
- Determines how particular values of attributes are used.
- Gives semantics to extraction rule.
- Gives semantics to extracted data.

Semantic data output



- Two instances of two ontology classes.

The experimental ontology

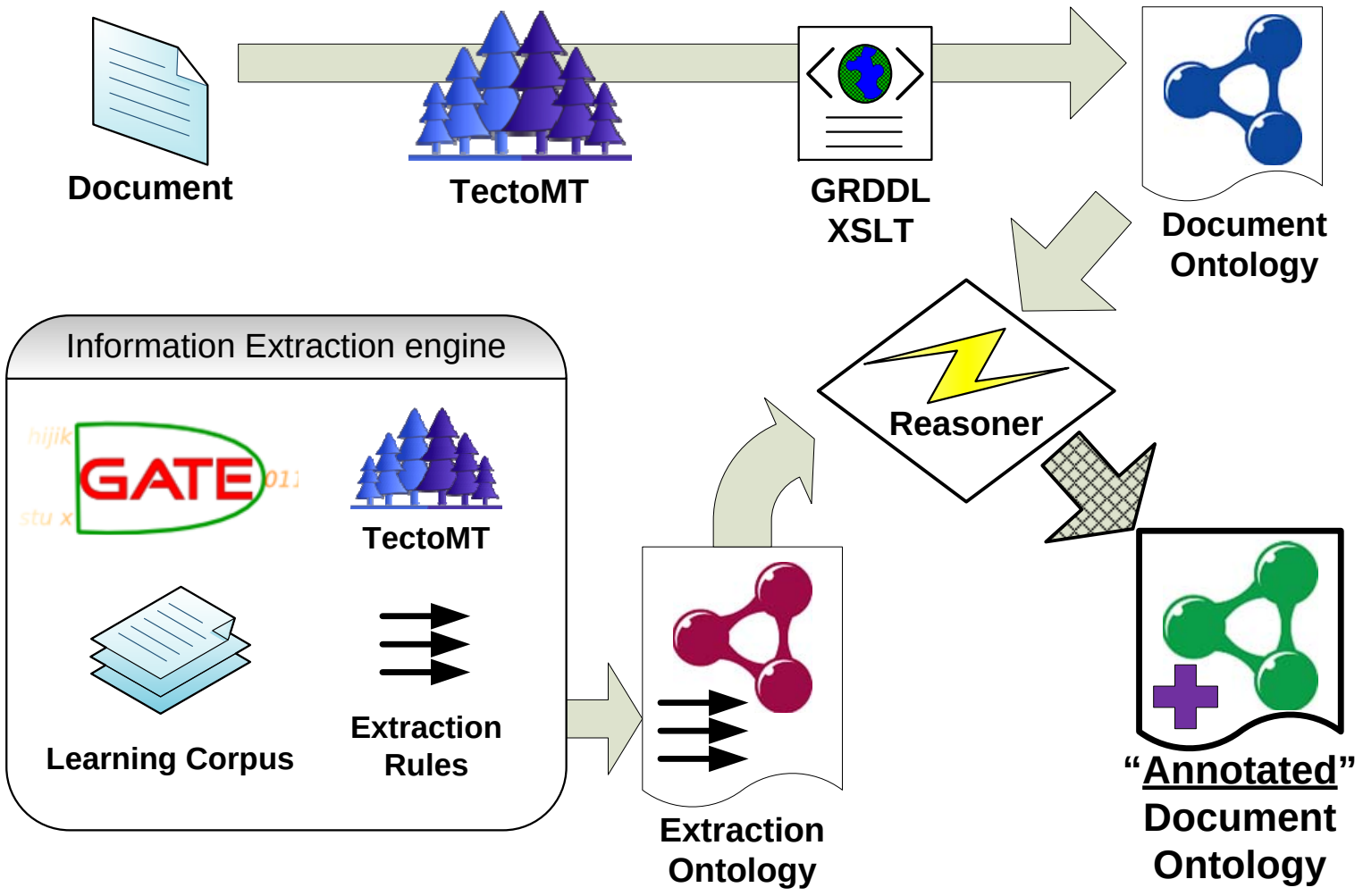


- Two classes
  - Incident and Participant
- One object property relation
  - hasParticipant
- Five datatype property relations
  - actionManner (light or heavy injury)
  - negation
  - actionType (injury or death)
  - participantType (man, woman, driver, etc.)
  - participantQuantity

Transformation of PML to RDF

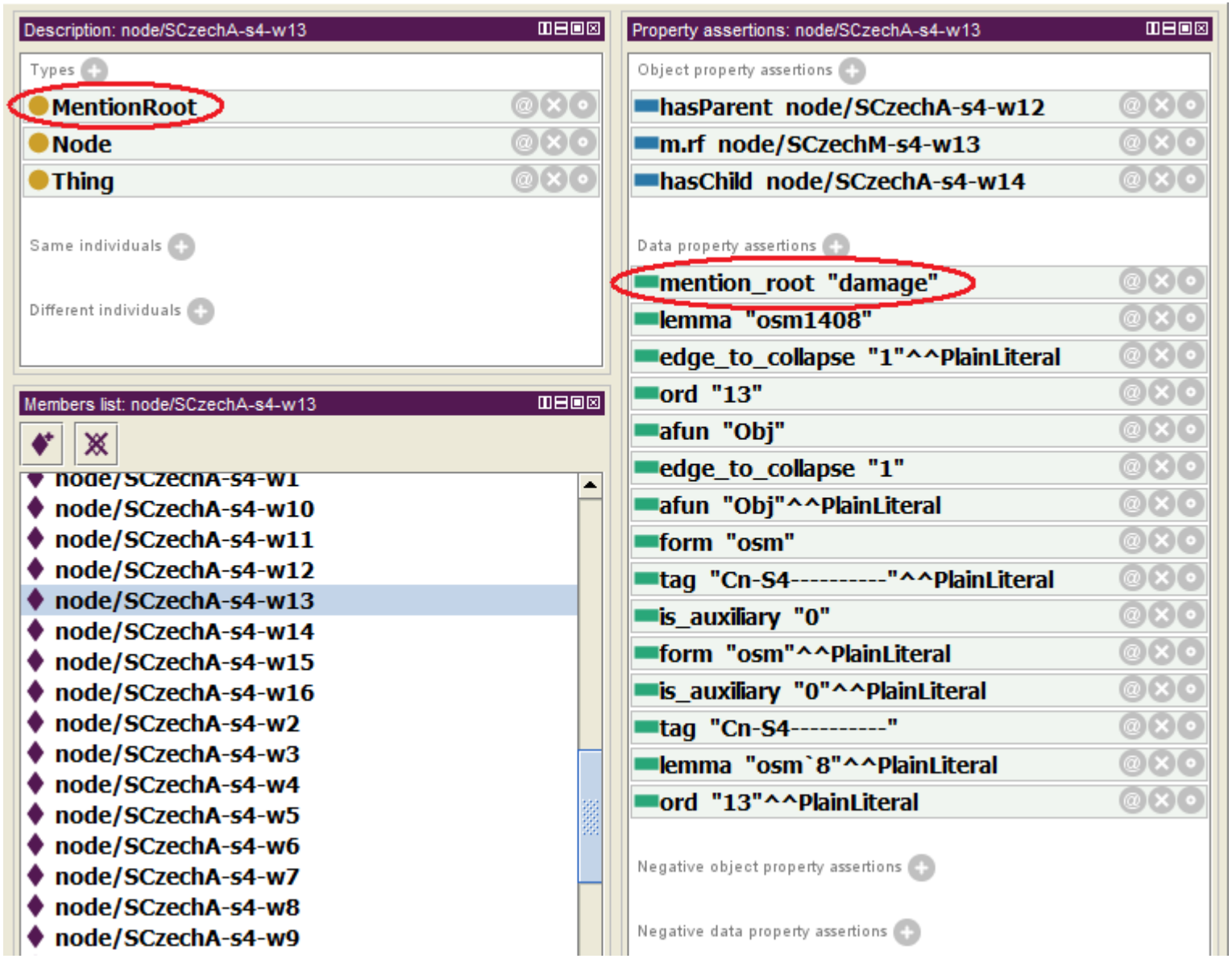
- Quite simple XSLT transformation
- Allows working with PDT annotations inside Semantic Web tools
  - Ontology Editors
  - Reasoners
  - Query tools (graph queries)
  - ?Visualization and navigation tools?
- In our case interpretation of extraction rules by a OWL reasoner

Extraction Rules Interpreted by OWL Reasoner



- Tool independent extraction ontologies

PDT in The Protégé Ontology Editor



Examples of extraction rules in the native Prolog format.

[Rule 1] [Pos cover = 23 Neg cover = 6]

```
mention_root(acquired,A) :-
   'lex.rf'(B,A), t_lemma(B,'Inc'), tDependency(C,B),
    tDependency(C,D), formeme(D,'n:in+X'), tDependency(E,C).
```

[Rule 11] [Pos cover = 25 Neg cover = 6]

```
mention_root(acquired,A) :-
   'lex.rf'(B,A), t_lemma(B,'Inc'), tDependency(C,B),
    formeme(C,'n:obj'), tDependency(C,D), functor(D,'APP').
```

[Rule 75] [Pos cover = 14 Neg cover = 1]

```
mention_root(acquired,A) :-
   'lex.rf'(B,A), t_lemma(B,'Inc'), functor(B,'APP'),
    tDependency(C,B), number(C,pl).
```

Examples of extraction rules in Protégé 4 – Rules View's format

[Rule 1]

```
lex.rf(?b, ?a), t_lemma(?b, "Inc"), tDependency(?c, ?b),
tDependency(?c, ?d), formeme(?d, "n:in+X"),
tDependency(?c, ?e)
      -> mention_root(?a, "acquired")
```

[Rule 11]

```
lex.rf(?b, ?a), t_lemma(?b, "Inc"), tDependency(?c, ?b),
formeme(?c, "n:obj"), tDependency(?c, ?d), functor(?d, "APP")
      -> mention_root(?a, "acquired")
```

[Rule 75]

```
lex.rf(?b, ?a), t_lemma(?b, "Inc"), functor(?b, "APP"),
tDependency(?c, ?b), number(?c, "pl")
      -> mention_root(?a, "acquired")
```