Treex

Treex: Modular NLP Framework

Martin Popel
ÚFAL (Institute of Formal and Applied Linguistics)
Charles University in Prague

September 2015, Prague, MT Marathon

Outline

- Motivation, Treex vs. TectoMT
- Treex architecture
- Treex internals
- Future plans
- Conclusion and examples

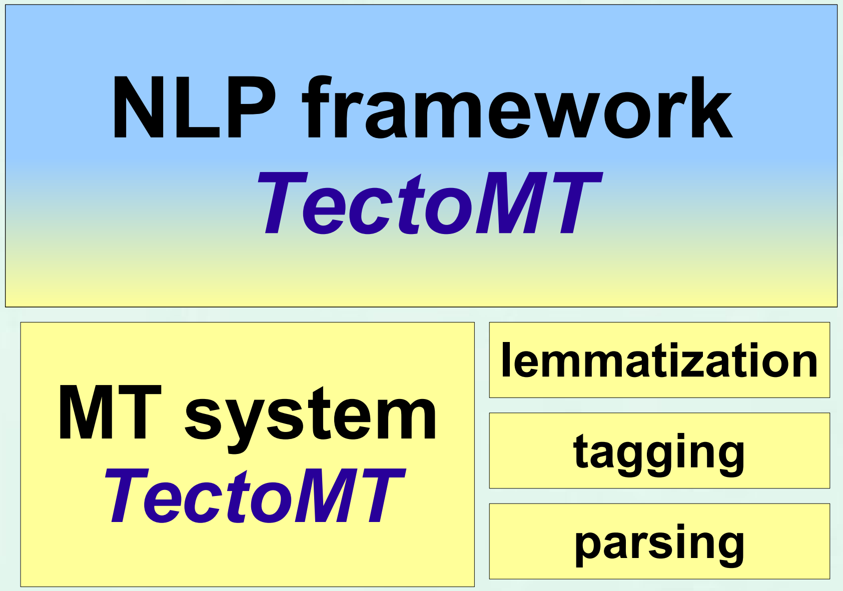
Motivation

Goals of Treex

- elegant integration of in-house and third-party NLP tools
- modularity, reusability, cooperation
- ability to easily modify and add code in a full-fledged programming language (Perl)

Treex vs. TectoMT

2005 (Zdeněk Žabokrtský)



Treex vs. TectoMT

Treex

2005 … 2011

2005

NLP framework
TectoMT

MT system *TectoMT* | **lemmatization** | **tagging** | **parsing**

2011

multi-purpose NLP framework
Treex

MT system *TectoMT* | **lemmatization** | **tagging** | **parsing**

coreference | **PEDT preprocessing**
CzEng analysis | **treebank conversions**
named entity r. | **alignment (word,tree)**
SMT preproc. | **etc.**

Treex vs. TectoMT

Treex

2005 … 2011

NLP framework
TectoMT

MT system
TectoMT

lemmatizati

ta g

ing

Now not only
tectogrammatics
and not only MT
⇨ renamed

multi-purpose
NLP framework
Treex

MT system
TectoMT

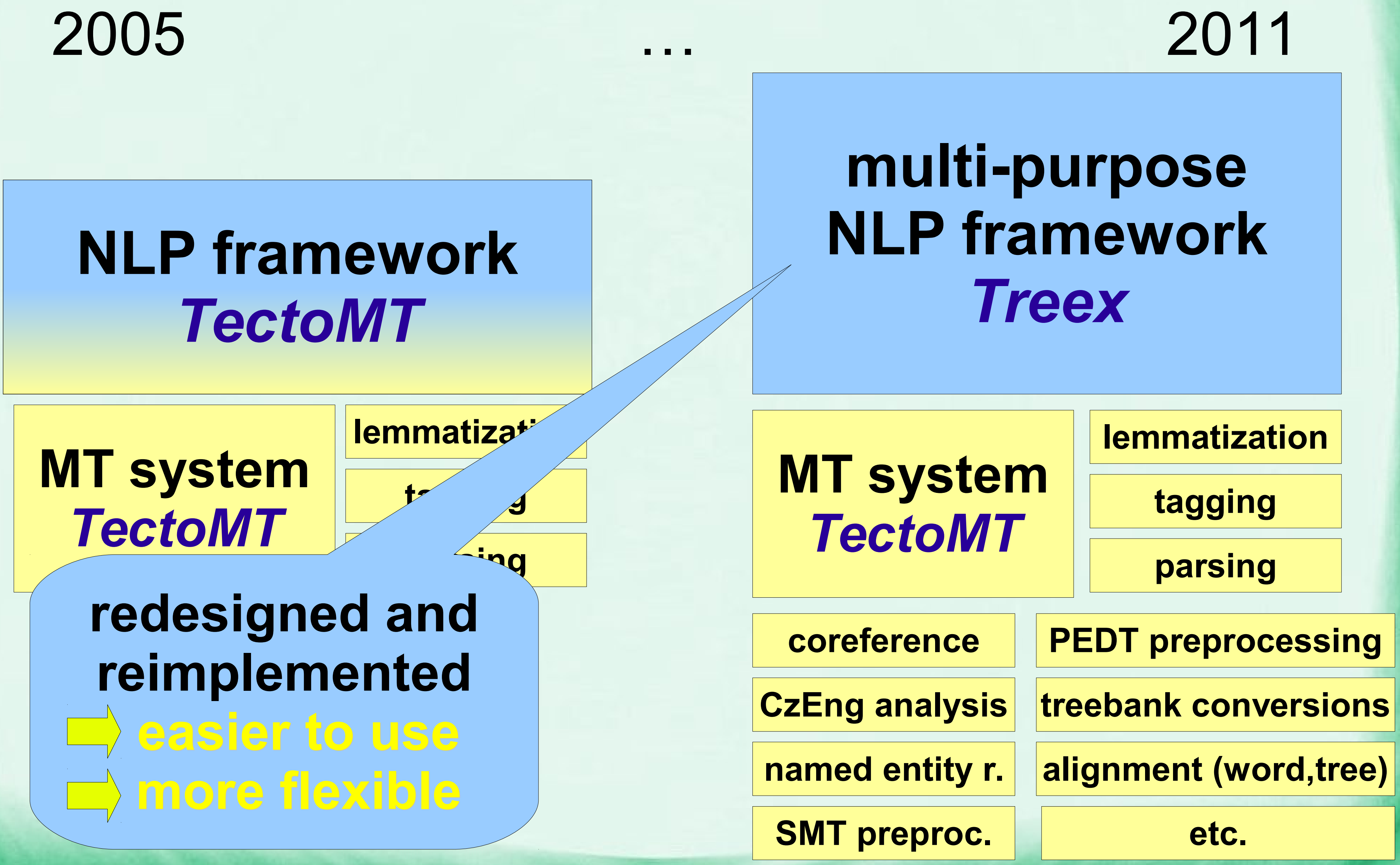
lemmatization

tagging

parsing

coreference	PEDT preprocessing
CzEng analysis	treebank conversions
named entity r.	alignment (word,tree)
SMT preproc.	etc.

Treex vs. TectoMT



Treex vs. TectoMT

Treex

2005

English

Czech

NLP framework
TectoMT

MT system
TectoMT

lemmatizati

ta g

sing

redesigned and reimplemented

- easier to use
- more flexible
- more langs

…

English

Czech

Russian

Tamil

NLP framework

Polish

Esperanto

Spanish

French

tagging

TectoMT

German

Arabic

coreference

PEDT preprocessing

C Vietnamese tr

Hindi

ons

named entity r

alignment (word tree)

Urdu

Finish

*) Most of the listed languages are only drafts of anlysis made by students, not converted to Treex yet. The entire risk as to the quality and performance of the program is with you.

Treex vs. TectoMT

Treex

2005

English

Engli

Czech

amil

amework

Esperanto

Special offer
Call now and get
one extra Treex
for free

M

Tect

ectoMT

French

agging

erman

Arabic

reimpl

te

coreference

PEDT preprocessing

easie use

more flexible

more langs

C

Vietnamese

tr

Hindi

ons

named entity r

alignment (word tree)

Urdu

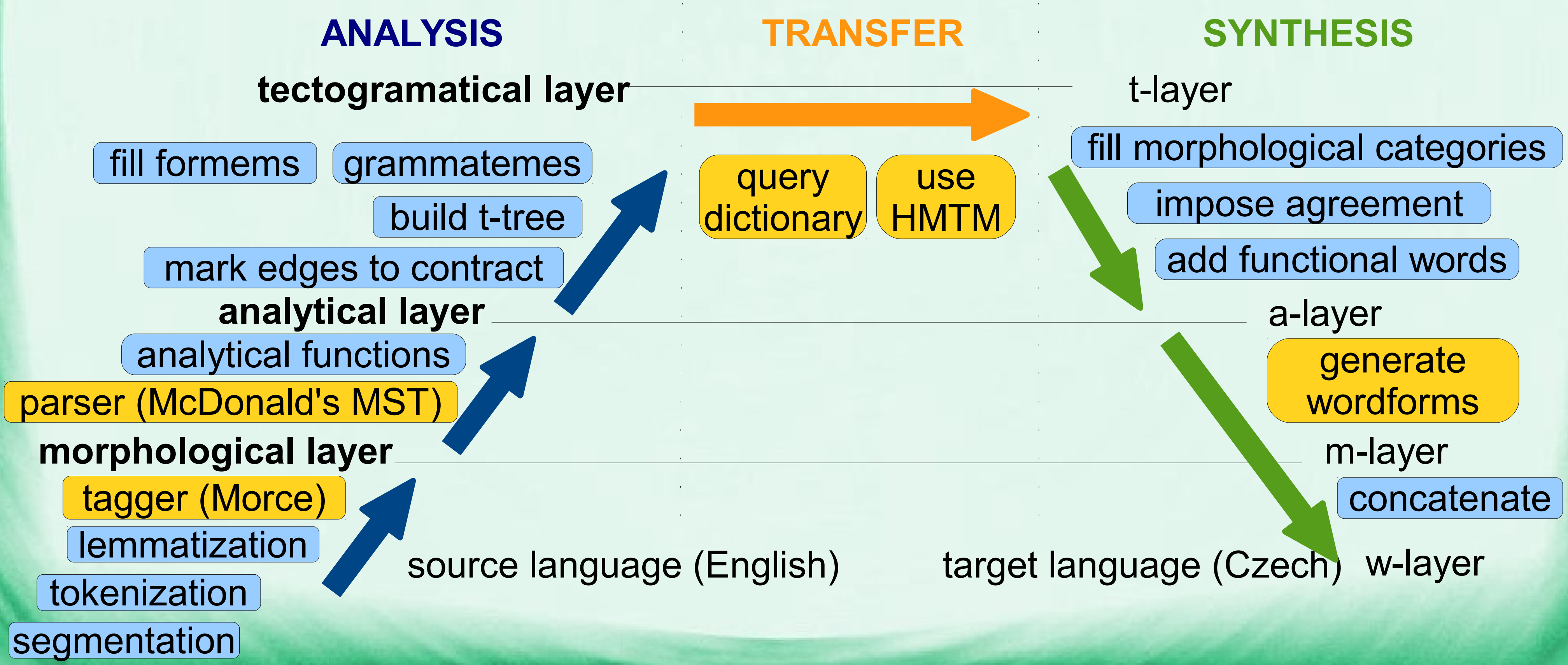
Finish

*) Most of the listed languages are only drafts of anlysis made by students, not converted to Treex yet. The entire risk as to the quality and performance of the program is with you.

TectoMT

linguistically motivated MT system (English to Czech pilot)

- deep syntactic (tectogrammatical) transfer
- translation process divided to more than 90 “blocks“
- combining statistical and rule based blocks

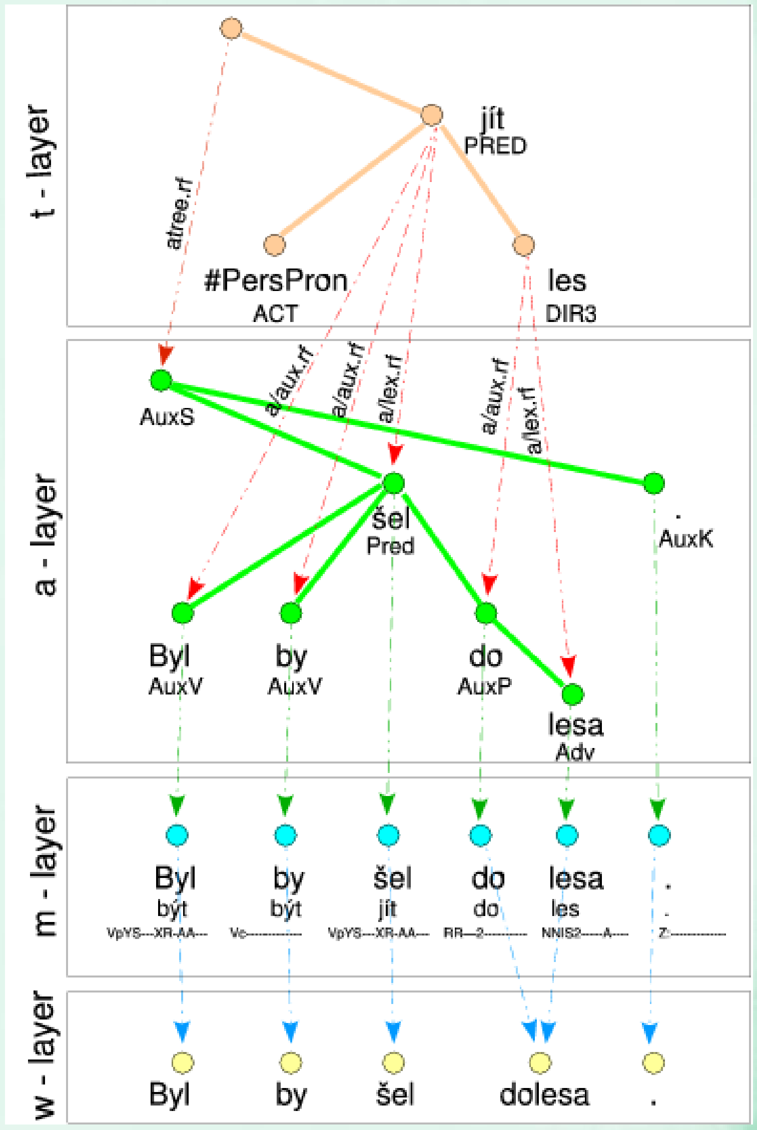


4 layers of language description

implemented in Prague Dependency Treebank (PDT)

Treex

- **tectogrammatical layer**
  deep-syntactic dependency trees
- **analytical layer**
  surface-syntactic dependency trees, labeled edges
- **morphological layer**
  lemma & POS tag for each word
- word layer
  raw (tokenized) text



4 layers of language description
implemented in Prague Dependency Treebank (PDT)

Treex

- **tectogrammatical layer**
  deep-syntactic dependency trees

- abstraction from many language-specific phenomena

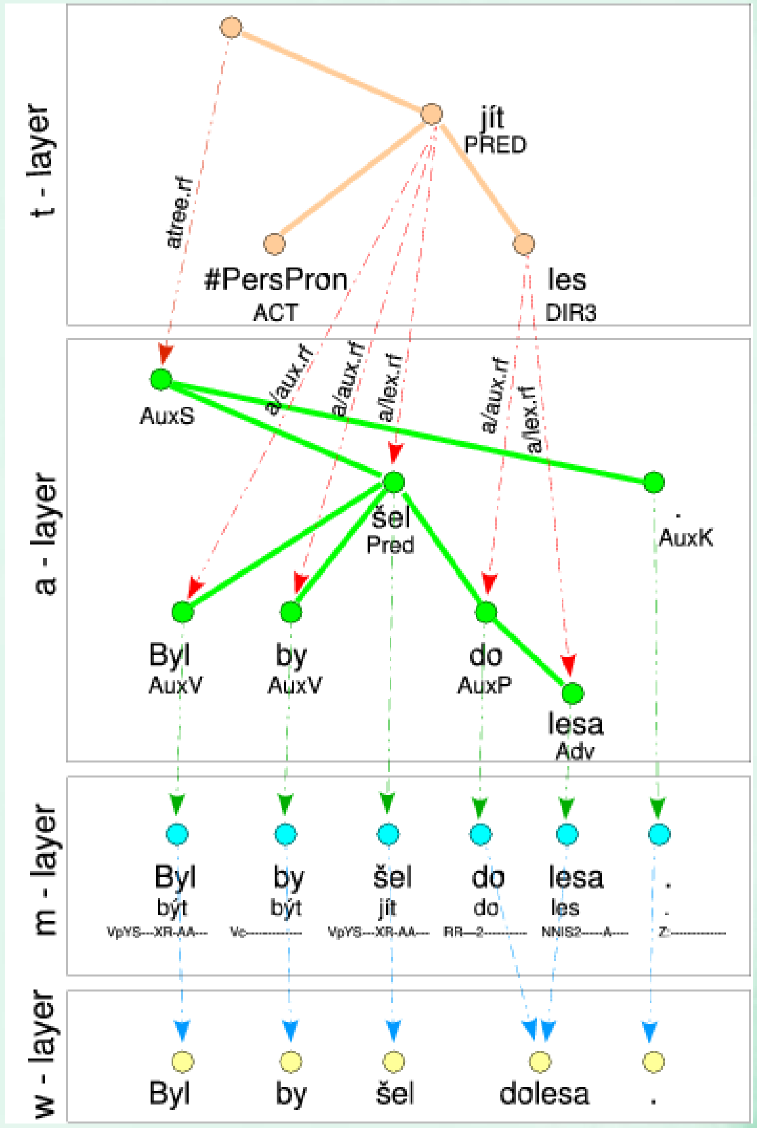
- autosemantic (meaningful) words
  ~ **nodes**

- functional words (prepositions, auxiliaries)
  ~ **attributes**

- syntactic-semantic relations (dependecies)
  ~ **edges**

- added nodes (e.g. becasue of pro-drop)

- ...



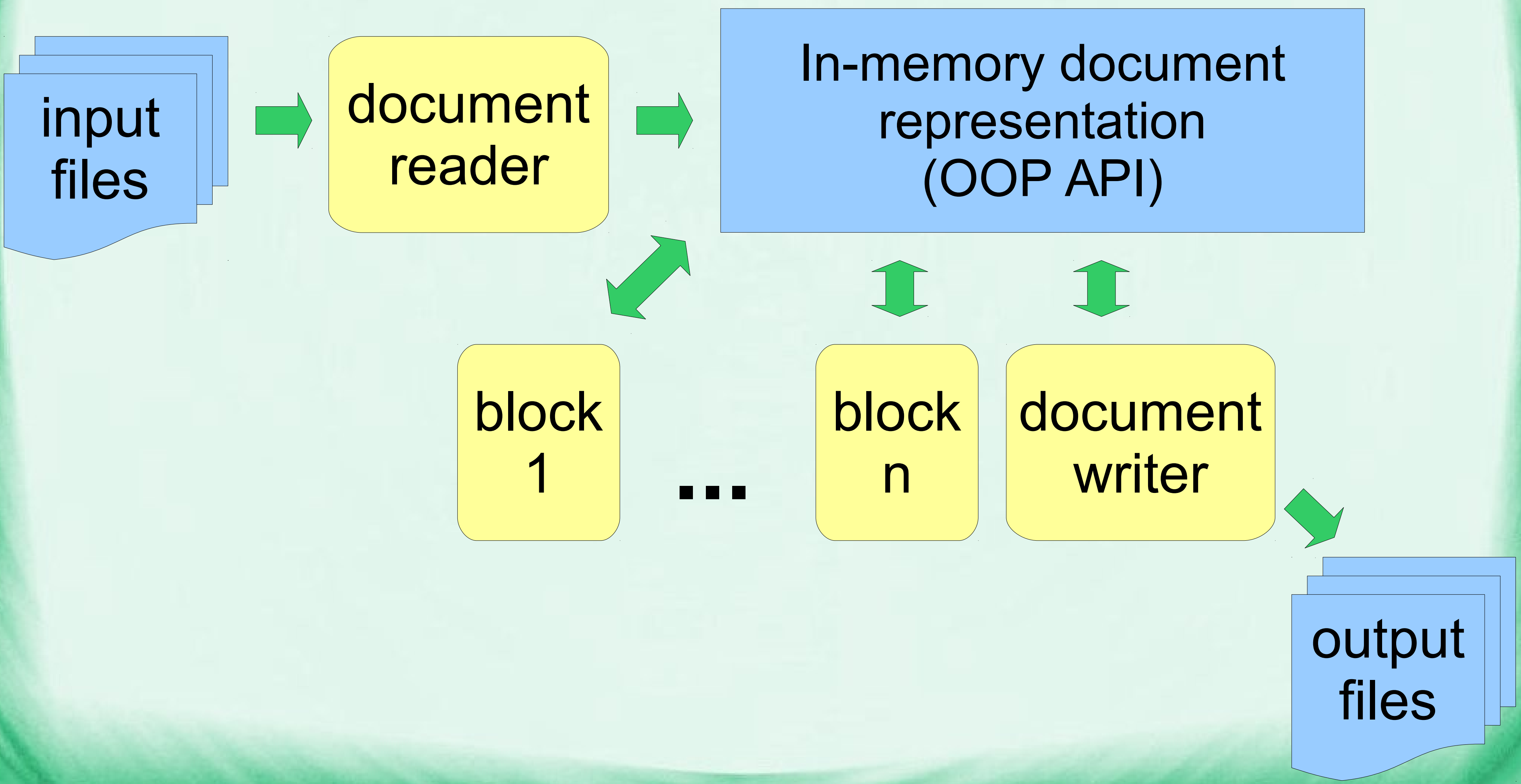
layers of language description
implemented in Treex

Treex

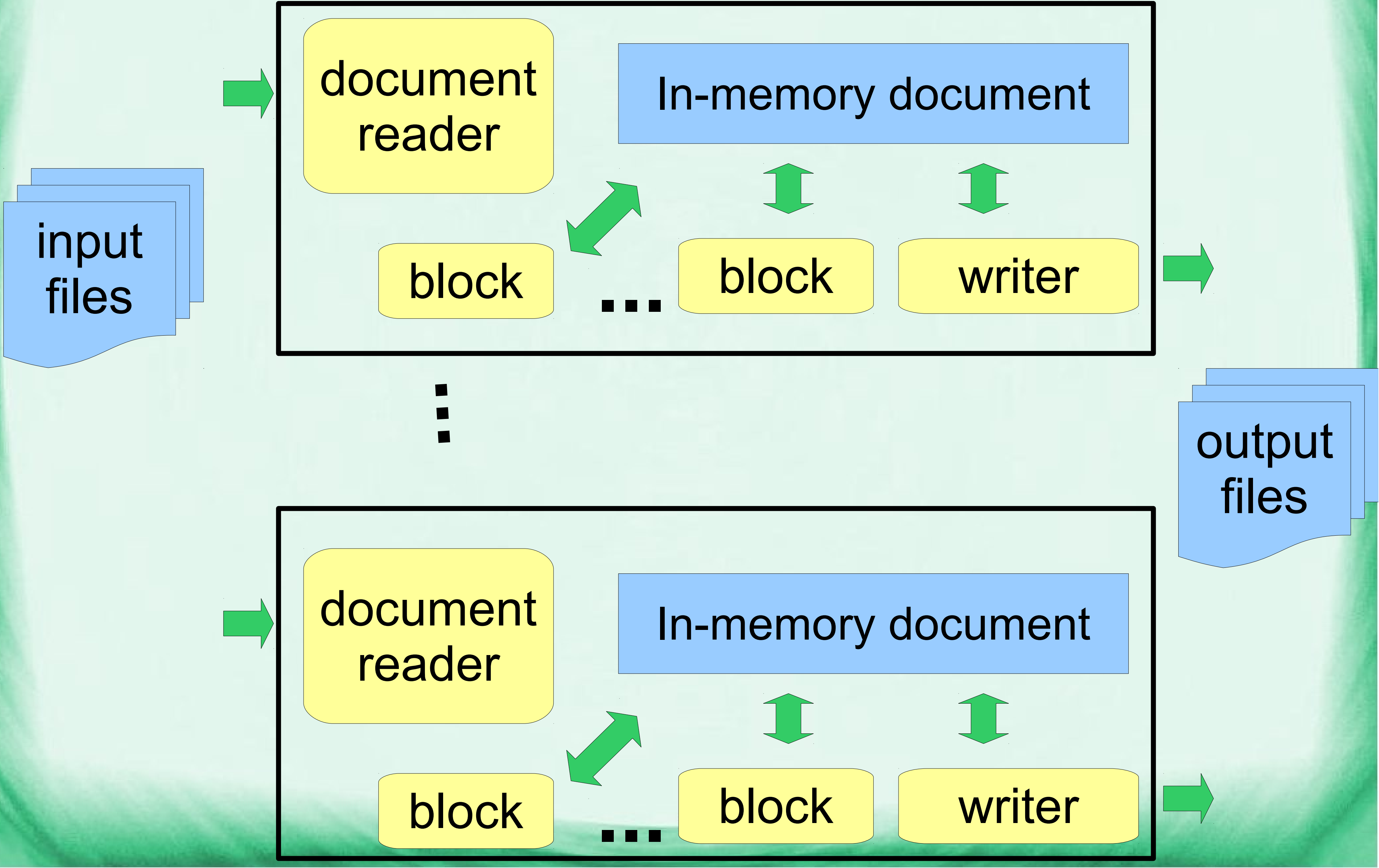
- Mostly backward compatible adaptations (adding attributes)
  - **formeme** (n:2, n:k+3, v:že+vfin, v:rc, adj:attr)
  - attributes for clauses, is_passive (→ diathesis),...
- is_member (for conjuncts on a-layer) is stored with prepositions

- All layers stored in **one file**
- A-layer and m-layer merged into one
- Two more layers:
  - P-layer phrase-structure trees
  - N-layer named entities

Treex architecture



Treex architecture
parallelization (using SGE cluster)



Treex architecture processing units

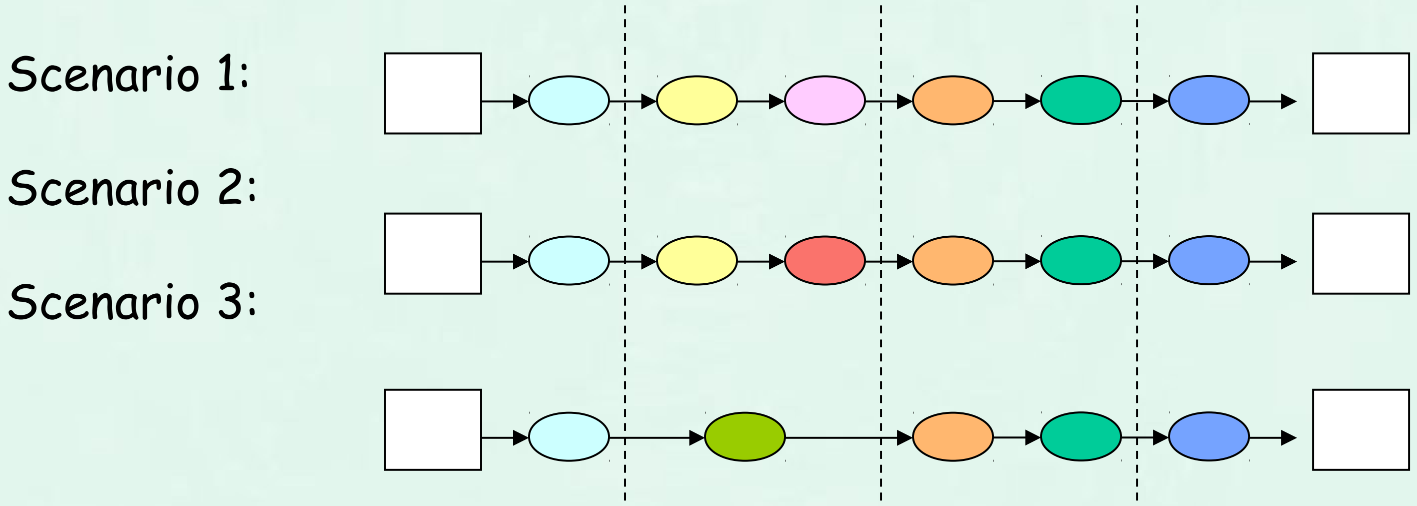
Treex

- **block** – elementary processing unit in Treex
  - corresponding to a given NLP subtask
  - one Perl class (Treex::Block::*), saved in one file
- **scenario** – a sequence of blocks
  - saved in plain text files or a Treex::Scen::* Perl class
  - just a list of the blocks' names and their parameters
- **application** – represents an end-to-end NLP task
  - described by a scenario that
    - starts with a **reader** (input conversion)
    - ends with a **writer** (output conversion)
  - Readers can split the input file into more in-memory docs.
  - There are readers&writers for a number of popular formats: plain text, CoNLL, PDT PML, Penn MRG, Tiger...

`*.treex.gz`

Treex architecture processing units

Blocks can be easily substituted with an alternative solution.



Treex architecture processing units

Blocks can be easily substituted with an alternative solution.

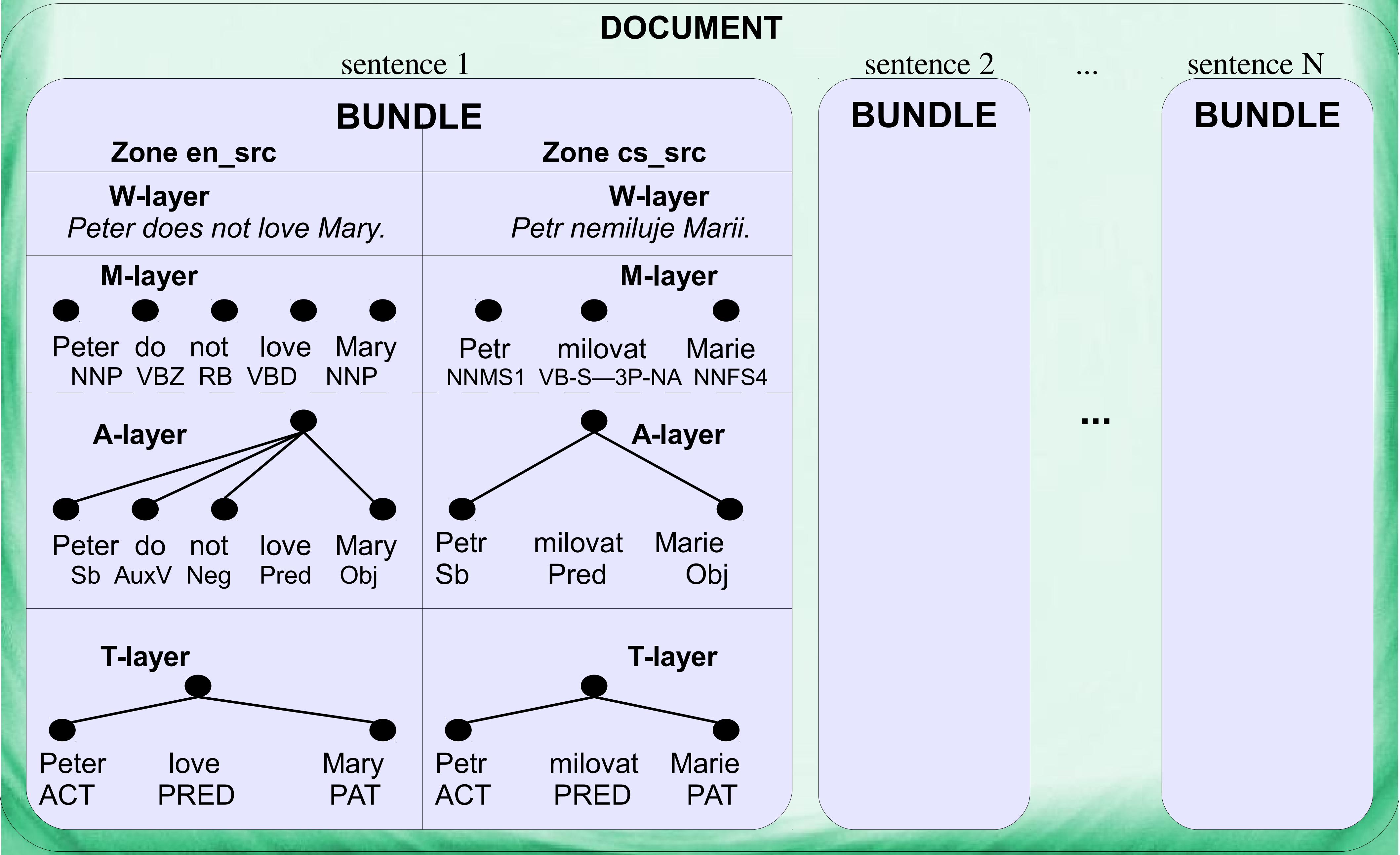
Scenario A	Scenario B
`W2A::EN::Segment`	`W2A::SegmentOnNewlines`
`W2A::EN::Tokenize`	`W2A::EN::TagLinguaEn`
`W2A::EN::TagMorce`	
`W2A::EN::Lemmatize`	`W2A::EN::Lemmatize`
`W2A::EN::ParseMST`	`W2A::EN::ParseMalt`

Treex architecture
data units

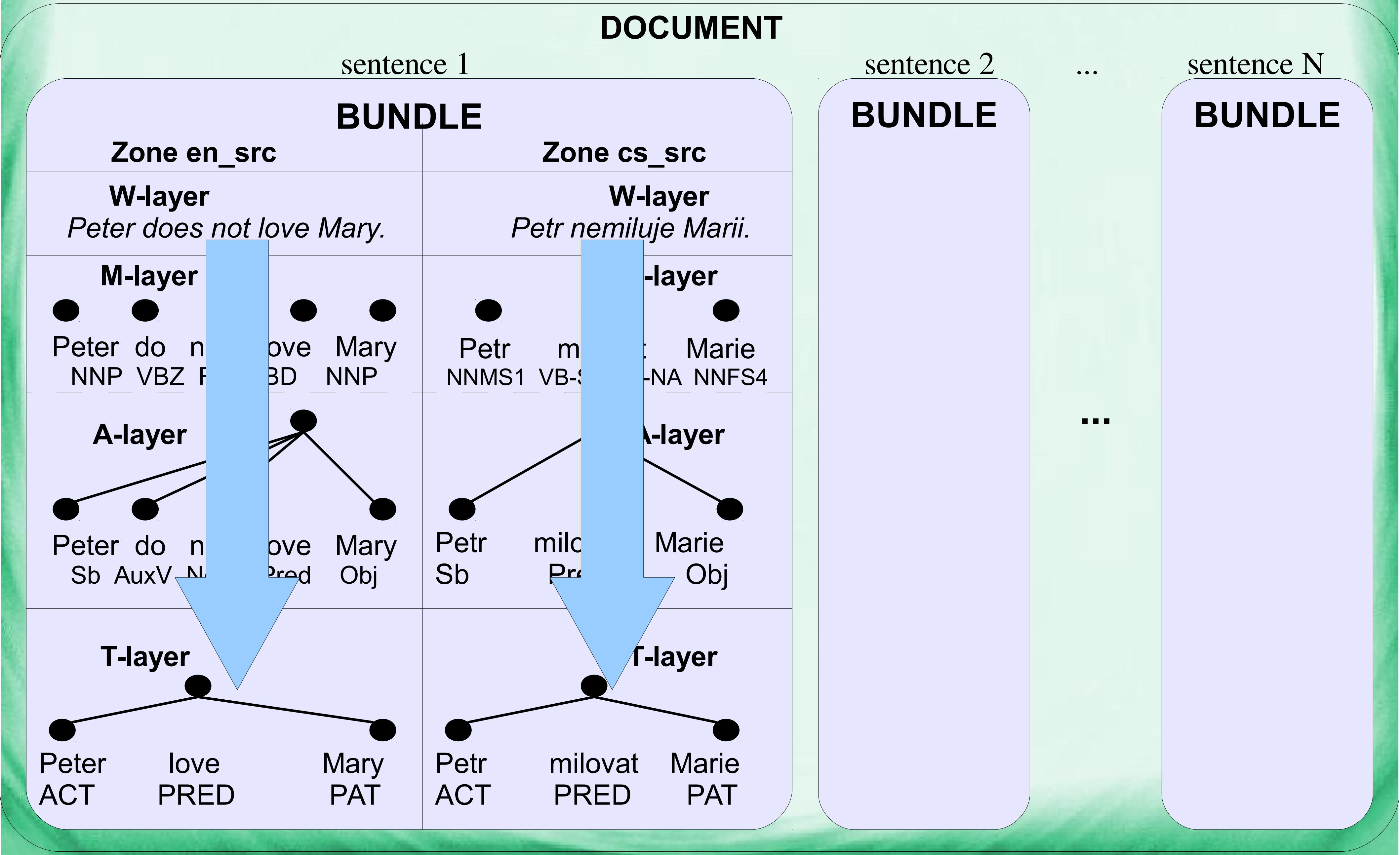
Treex

- **Document**
  - stored in one file
  - sequence of sentences
- **Bundle** (“bundle of trees”)
  - corresponds to one sentence
- **Zone**
  - one for each language (Arabic, Czech, English,...)
  - and optionally a variant (“selectors” src, trg, ref,...)
- **Tree**
  - layer of language description: A, T (plus P, N)
  - m-layer is stored with the a-layer in one tree

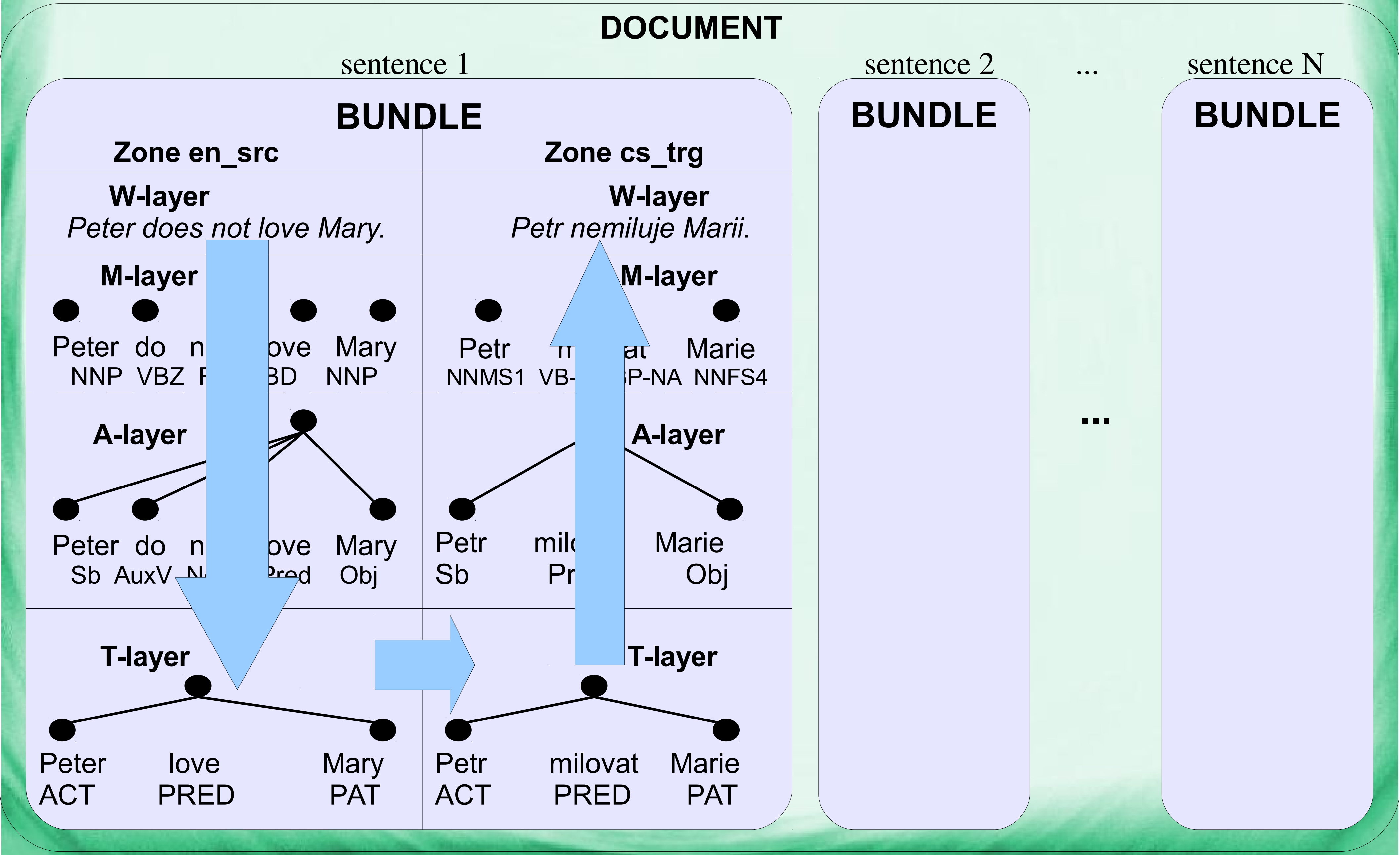
Treex architecture data units



Treex architecture
data units



Treex architecture data units



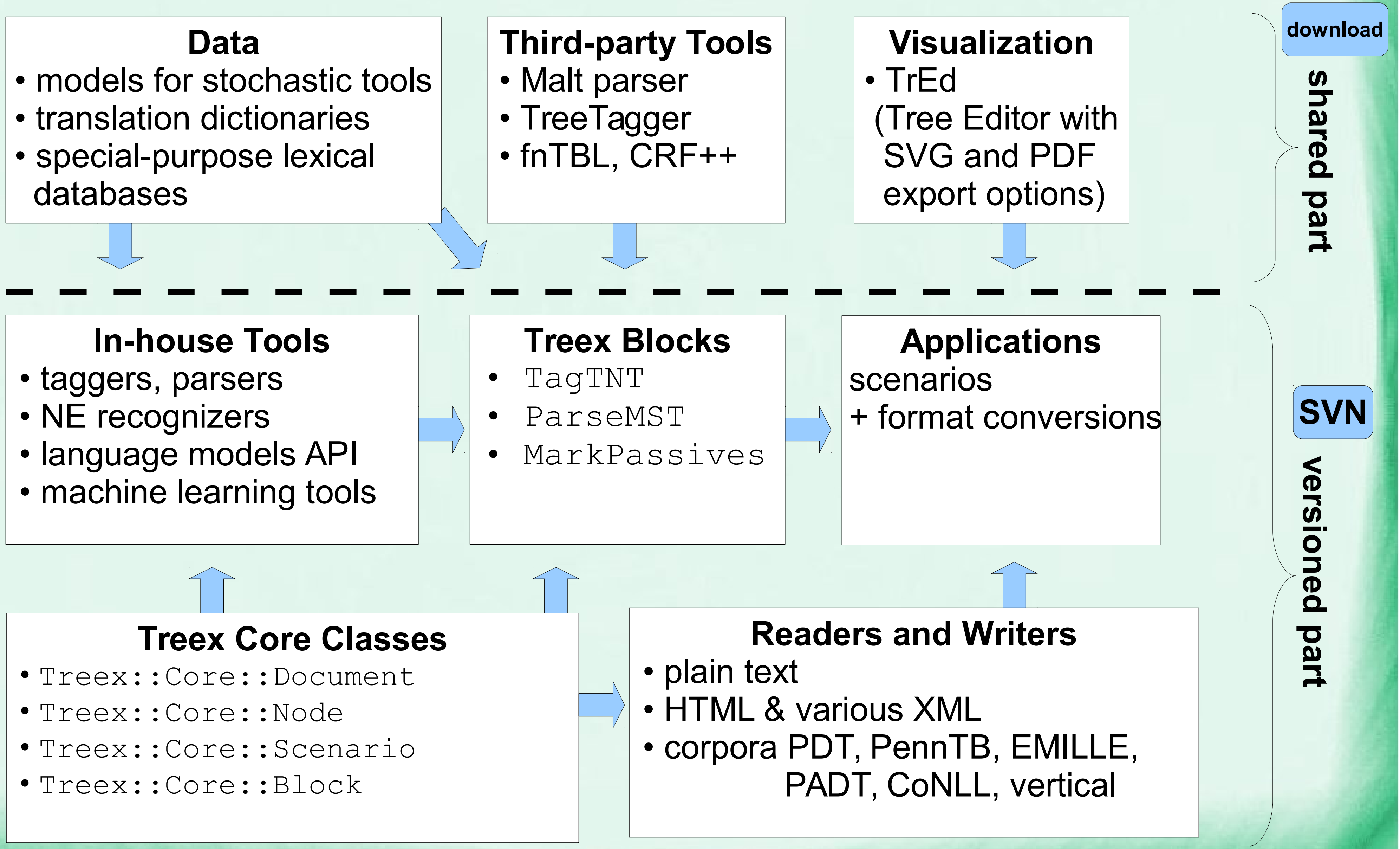
Internals – Design decisions

Treex

- Perl (wrappers for binaries, Java,...)
- Linux (some applications platform-independent)
- OOP (Moose)
- Open source (GNU GPL for the versioned part)
- Neutral w.r.t. methodology (statistical, rule-based)
- Multilingual
- Open standards (Unicode, XML)

Internals – Components

Treex



Data
- models for stochastic tools
- translation dictionaries
- special-purpose lexical databases

Third-party Tools
- Malt parser
- TreeTagger
- fnTBL, CRF++

Visualization
- TrEd (Tree Editor with SVG and PDF export options)

In-house Tools
- taggers, parsers
- NE recognizers
- language models API
- machine learning tools

Treex Blocks
- `TagTNT`
- `ParseMST`
- `MarkPassives`

Applications
scenarios
+ format conversions

Treex Core Classes
- `Treex::Core::Document`
- `Treex::Core::Node`
- `Treex::Core::Scenario`
- `Treex::Core::Block`

Readers and Writers
- plain text
- HTML & various XML
- corpora PDT, PennTB, EMILLE, PADT, CoNLL, vertical

Internals – Statistics

Treex

- Developed since 2005, over ten developers
- Over 400 blocks (140 English, 120 Czech, 60 English-to-Czech, 30 other languages, 50 language independent)
- Taggers (5 English, 3 Czech, 1 German and Russian, Tamil)

  Parsers (Dep. 2 English, 3 Czech, 2 German; Const. 2 English)

  Named Entity Recognizers (2 Czech,1 English)
- Speed example: Best version of English-to-Czech MT

  1.2 seconds per sentence plus 90 seconds loading,

  with 20 computers in cluster: 2000 sentences in 4 min

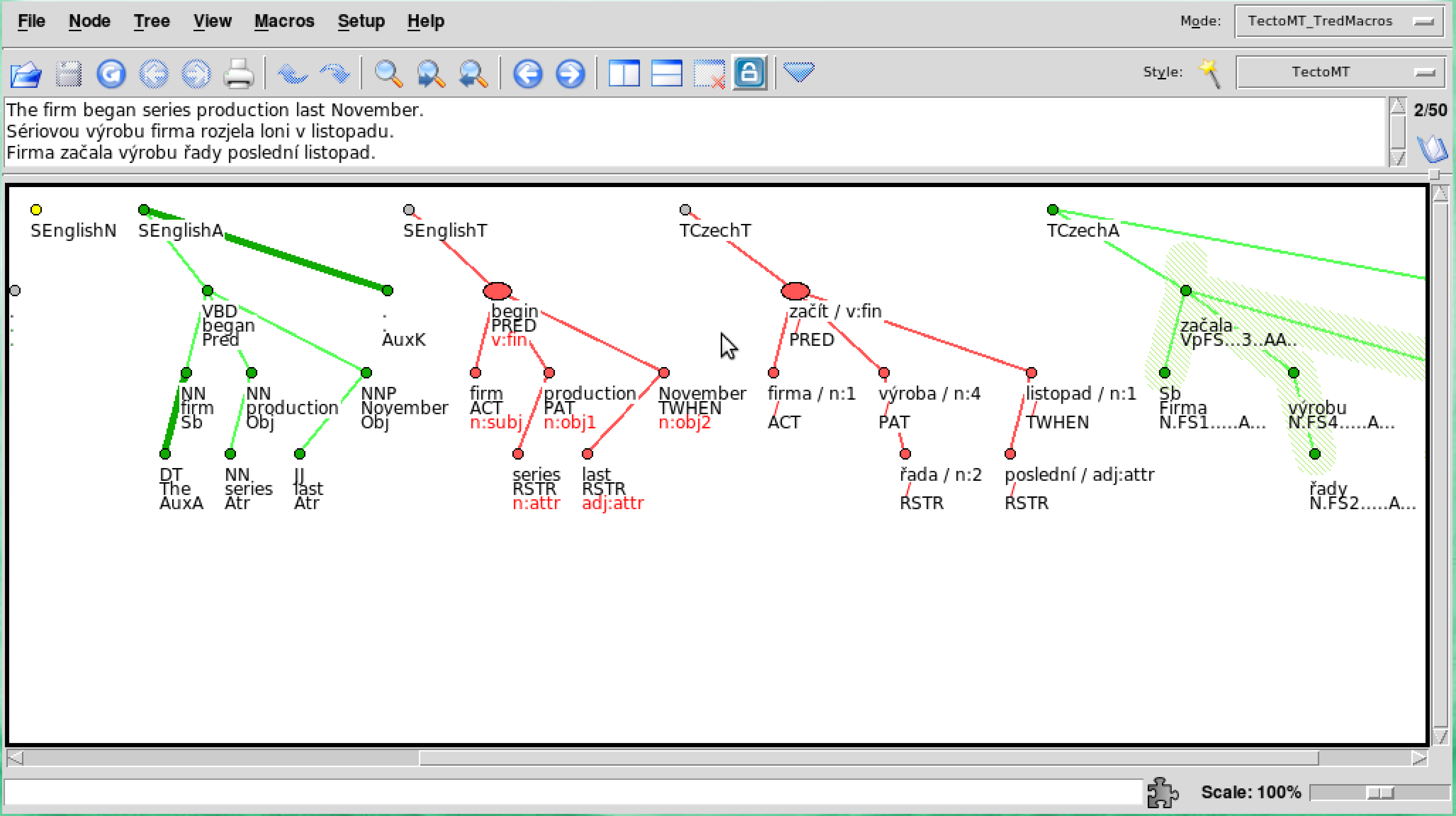
Conclusion
Treex main properties

Treex

- emphasized efficient development, modular design and reusability
- stratificational approach to the language
- unified object-oriented interface for accessing data structures
- comfortable development

TrEd visualization

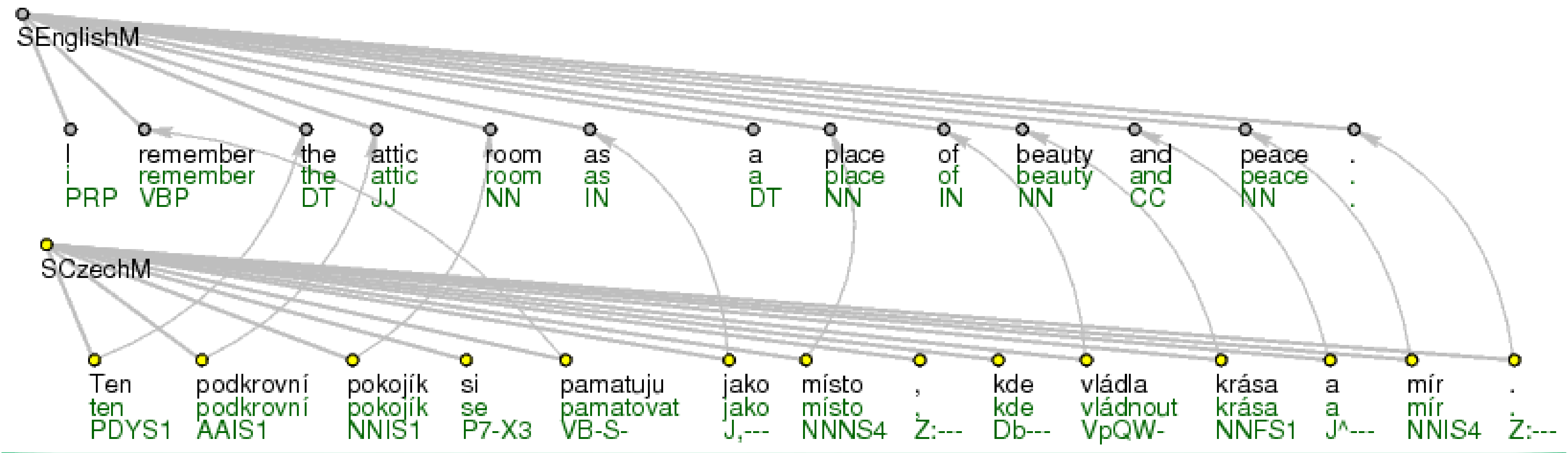
translation



The firm began series production last November.
Sériovou výrobu firma rozjela loni v listopadu.
Firma začala výrobu řady poslední listopad.

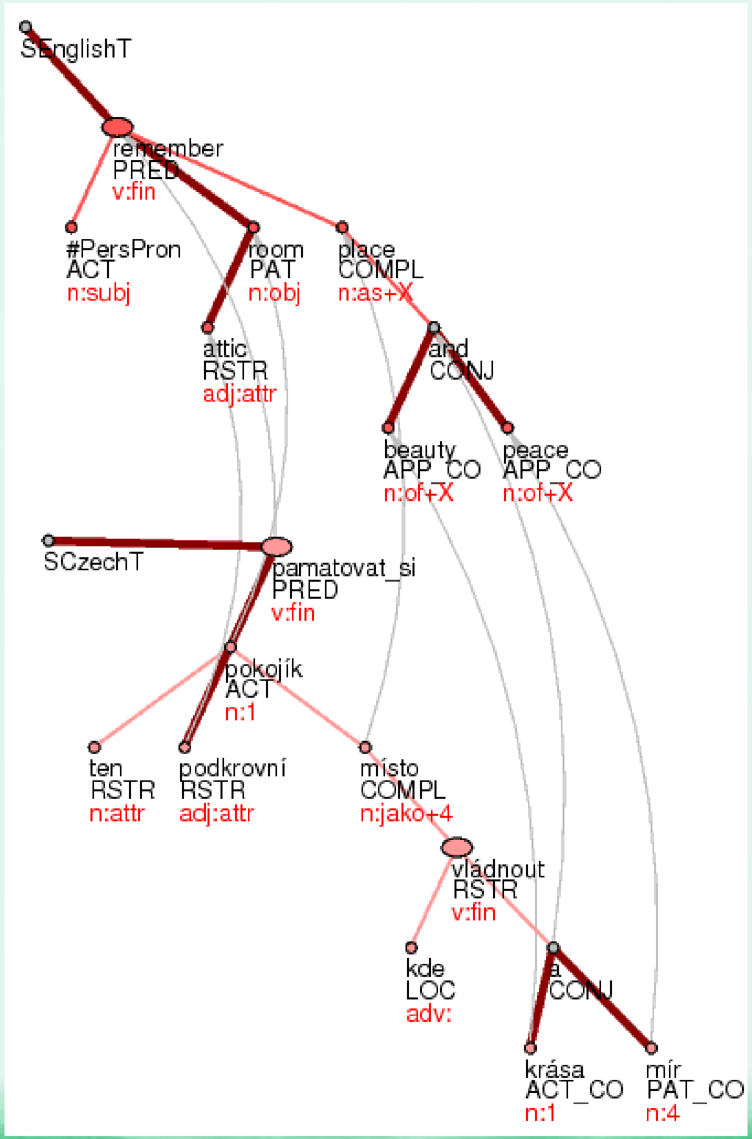
TrEd visualization

word alignment on the morphological layer



TrEd visualization

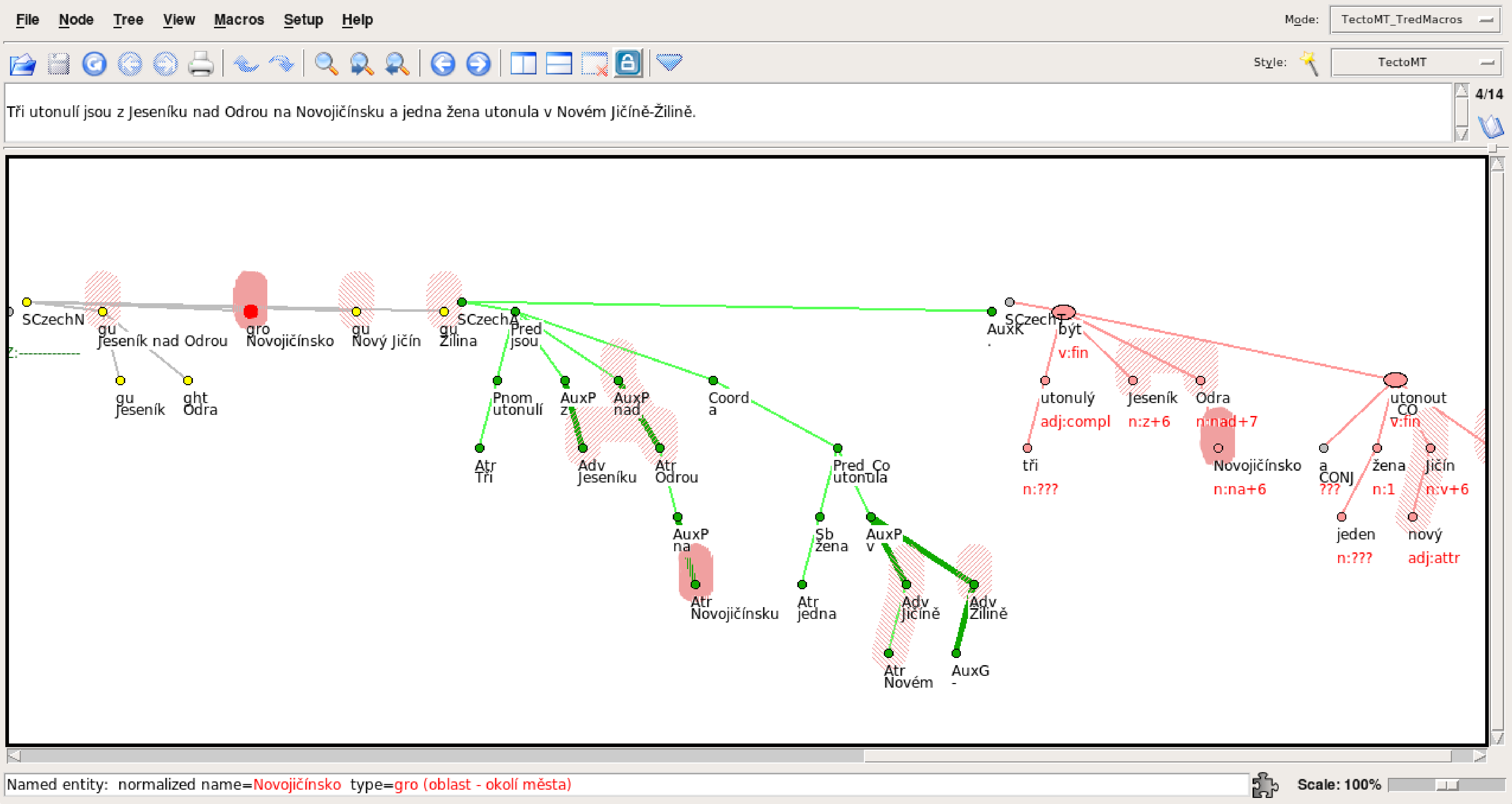
word alignment on the tectogrammatical layer



TrEd visualization

Treex

named entities



Tři utonulí jsou z Jeseníku nad Odrou na Novojičínsku a jedna žena utonula v Novém Jičíně-Žilině.

Named entity: normalized name=Novojičínsko type=gro (oblast - okolí města)

Block example – SVO to SOV code

Treex

```
package Tutorial::Svo2SovSolution;
use Moose;
use Treex::Core::Common;
extends 'Treex::Core::Block';

sub process_anode {
  my ( $self, $a_node ) = @_;
  if ( $a_node->tag =~ /^V/ ) {       # verb found
    foreach my $child ( $a_node->get_echildren() ) {
      if ( $child->afun eq 'Obj' ) {    # object found
        # Move the object and its subtree so it precedes the verb
        $child->shift_before_node($a_node);
      }
    }
  }
  return;
}
1;
```

Treex core
Treex convention
Perl keyword/convention

Thank you

Treex

Cooperation is welcomed.

http://ufal.mff.cuni.cz/treex

Thank you

Treex is growing!

http://ufal.mff.cuni.cz/treex