

**Od nieustrukturyzowanych danych
do przeszukiwalnego korpusu
bogatego w metadane:**

Skyscraper, P4, Smyrna

Daniel Janus

Rebased

Zdrapywanie danych (*webscraping*)

<http://wybory2010.pkw.gov.pl/geo/pl/000000.html>

Zbiorcze informacje o				
Województwo	Liczba			
	mieszkańców	wyborców	obwodów	okręgów wyborczych do sejmików
<u>dolnośląskie</u>	2 843 421	2 337 575	1 818	5
<u>kujawsko-pomorskie</u>	2 052 876	1 649 619	1 398	6
<u>lubelskie</u>	2 180 567	1 754 908	1 803	5
<u>lubuskie</u>	1 004 112	810 168	664	5
<u>łódzkie</u>	2 514 380	2 067 733	1 713	6
<u>małopolskie</u>	3 256 359	2 617 323	2 279	6

Zdrapywanie danych (*webscraping*)

```
▶<div style="clear:both;">...</div>
▼<div id="content">
  ▶<div id="tabs1" class="stContainer ui-tabs ui-widget ui-widget-content ui-corner-all">...</div>
  ▼<div id="tabs2" class="stContainer ui-tabs ui-widget ui-widget-content ui-corner-all">
    ▶<ul class="ui-tabs-nav ui-helper-reset ui-helper-clearfix ui-widget-header ui-corner-all">...</ul>
    ▼<div id="tabs-3" class="ui-tabs-panel ui-widget-content ui-corner-bottom">
      ▼<table class="geo">
        ▼<tbody>
          ▶<tr class="labels">...</tr>
          ▶<tr class="labels">...</tr>
          ▶<tr class="labels">...</tr>
          ▶<tr class="labels">...</tr>
          ▶<tr class="labels">...</tr>
          ▼<tr class="odd">
            ▼<td class="details">
              <a style="text-decoration:underline;" class="tablist" href="020000/020000.html">dolnośląskie</a>
            </td>
            <td>2&nbsp;843&nbsp;421</td>
            <td>2&nbsp;337&nbsp;575</td>
            <td>1&nbsp;818</td>
            <td>5</td>
            <td>26</td>
            <td>169</td>
            <td>3</td>
            <td>30</td>
            <td>139</td>
            <td>78 </td>
            <td>83 </td>
            <td>8 </td>
          </tr>
        </tbody>
      </table>
    </div>
  </div>
</div>
```

html body #container #content #tabs2 #tabs-3 table.geo tbody tr.odd td

Jak zdrapywać?

```
$ wget -r http://www.strona-z-danymi.pl  
$ find . -name '*.html' -exec perl extract.pl {} \;
```

- To nie jest dobry pomysł
 - Nie wszystkie strony wchodzące w skład witryny mają taką samą strukturę
 - `wget` może pobrać za dużo albo za mało; strony mogą się powtarzać
 - Informacja o tym, w którym miejscu witryny jesteśmy, jest semantycznie cenna i nie należy jej gubić

Jak zdrapywać? (2)

```
def scrape_main_page():  
    r = requests.get("http://strona-z-danymi.pl")  
    data = r.text  
    soup = BeautifulSoup(data)  
    for link in soup.find_all('a'):  
        scrape_subpage(link.get('href'))  
  
def scrape_subpage(url):  
    # itd., itp.
```

- Lepiej, ale wciąż nieidealnie
 - Dużo powtarzającego się, technicznego kodu
 - Obsługa błędów, ponawianie prób pobrania, lokalne kopie stron, format wyjściowy, ...
 - Każda witryna jest inna i wymaga innych sztuczek

Skyscraper

<https://github.com/nathell/skyscraper>

- Biblioteka w języku Clojure ułatwiająca tworzenie “zdrapywaczy”
- Bazuje na bibliotece Enlive
- Wyrosła organicznie z “szytych na miarę” programów do pobierania danych z różnych witryn
- Obsługuje szczegóły techniczne pobierania i parsowania danych
- Udostępnia abstrakcje pozwalające skupić się na wyłuskiwaniu danych ze stron WWW

Clojure

- Lisp na platformę JVM
- Typy danych:
 - liczby 42, napisy "IPI PAN", symbole foo, klucze :foo, wartości logiczne :true
 - listy (1 2 3), wektory [1 2 3], mapy {:k1 v1, :k2 v2}, zbiory #{1 2 3}
- Kod składa się ze struktur języka
- (f arg1 arg2) oznacza "wywołaj funkcję f z argumentami arg1 i arg2"

Skyscraper: spostrzeżenia i abstrakcje

- Pobieranie danych ma strukturę drzewiastą
- Różne węzły drzewa są przetwarzane w różny sposób
- W każdym węźle pobieramy nowe informacje
- W każdym węźle znamy ścieżkę, którą do niego dotarliśmy, oraz informacje zebrane z węzłów nadrzędnych
- **Kontekst:** mapa zawierająca “zdrapane dotychczas dane”
- **Processor:** fragment kodu odpowiedzialny za przetwarzanie konkretnego typu węzłów
 - Funkcja (kontekst, sparsowany HTML) → lista nowych kontekstów

Skyscraper: przykład

<http://miejski.pl>

```
(defn seed []
  (for [x (concat (map str (char-range \a \z)) ["0-9" "inne"])]
    {:letter x,
     :url (format "http://www.miejski.pl/%s-0.html" x),
     :processor :letter}))

(defprocessor letter
  :cache-template "miejski/:letter/1"
  :process-fn (fn [res ctx]
    (for [a (select res [:div#strony :a])]
      {:page (text a),
       :url (href a),
       :processor :page})))
```

Skyscraper: przykład

<http://miejski.pl>

```
> (seed)
({:letter "a", :url "http://www.miejski.pl/a-0.html", :processor :letter}
 {:letter "b", :url "http://www.miejski.pl/b-0.html", :processor :letter}
 ; ... etc.
 {:letter "inne", :url "http://www.miejski.pl/inne-0.html", :processor :letter})
```

```
> (first (scrape :seed))
{:letter "a", :page "1", :word "A A A sie naebaem",
 :type :summary, :value "Zwrot używany przez ludzi po spożyciu alkoholu."} ;
```

- `scrape` zwraca leniwą sekwencję liści-kontekstów

Skyscraper: pamięć podręczna

- Dwa rodzaje pamięci podręcznej
 - Pamięć podręczna surowych dokumentów (*HTML cache*)
 - Pamięć podręczna przetworzonych dokumentów (*processed cache*)
- Obie domyślnie włączone, ale można je wyłączyć
- Hierarchiczne klucze
- Implementacje:
 - System plików (domyślna)
 - Jednoplikowa baza danych typu *key-value store* (MapDB; `skyscraper-cache-mapdb`)
 - Pamięć RAM

Skyscraper: pozostałe cechy

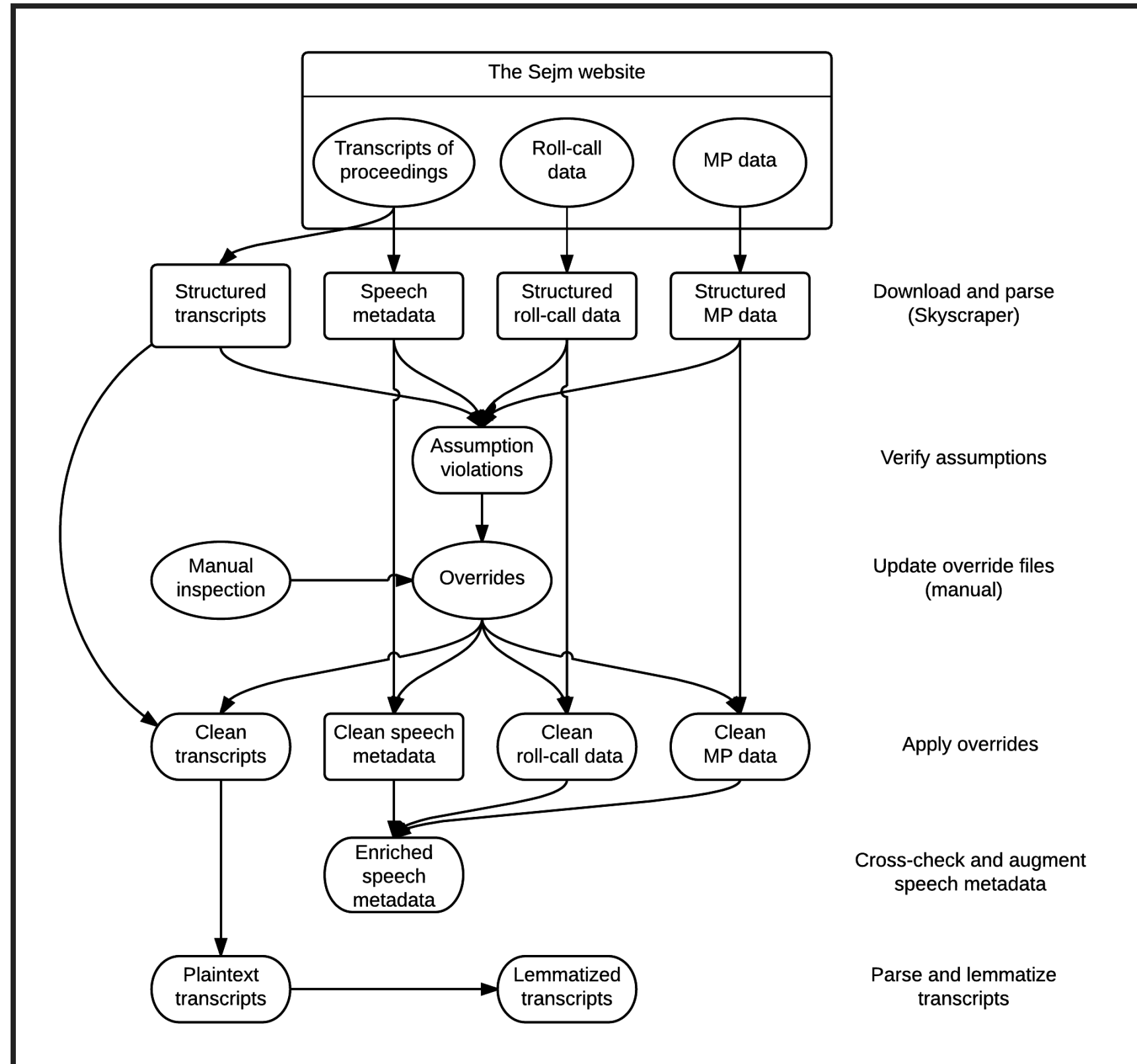
- `scrape - csv`
- Uaktualnienia: dowolny procesor można oznaczyć jako `:updatable true`; wtedy uruchomienie Skyscrapera w trybie uaktualnień pobierze odpowiednie strony jeszcze raz, nawet jeśli były już w pamięci podręcznej
- Zawężanie danych: w parametrze `only` można podać okrojony fragment kontekstu

```
> (first (scrape :seed :only {:letter "e"}))  
{:letter "e", :page "1", :word "E-faj", :type :summary,  
 :value "Skrót od e-papierosa. Używane głównie przez palaczy. Rozwinięcie  
 skrótu (potocznie): elektroniczny fajek."}
```

Polish Parliamentary Proceedings Processor (P4)

- Zestaw zbiorów danych pozyskanych z witryny Sejmu RP
 - Dane o posłach (kadencje I-VIII)
 - Wyniki głosowań (*roll-call*; kadencje I-VIII)
 - Stenogramy sejmowe (kadencje I-VIII) z metadanymi
- Także: oparty o bibliotekę Skyscraper program do pobierania i przebudowywania tych danych
- Półautomatyczna weryfikacja i korekcja błędów
- Uaktualniany na bieżąco

P4: schemat działania



P4: formaty danych

- Dane o posłach: CSV
- Wyniki głosowań: CSV
 - Dla każdej kadencji: macierz *głosowania* \times *posłowie*
 - Dodatkowo dla każdego głosowania: przynależność partyjna głosującego
 - Metadane w osobnym pliku
- Stenogramy:
 - HTML (wyczyszczony i wewnątrznie poprawnie skroslinkowany)
 - Czysty tekst
 - Czysty tekst, zlematyzowany
 - Metadane w osobnym pliku CSV


P4: metadane stenogramów

```
<meta content="56" name="POS">
<meta content="3" name="DZIEN">
<meta content="17-12-2009" name="DATA">
<meta content="149" name="WYP">
<meta content="Sekretarz Stanu w Ministerstwie Spraw Wewnętrznych
i Administracji Tomasz Siemoniak" name="KTO">
<meta content="21" name="PUNKT">
<meta content="Sprawozdanie Komisji Administracji i Spraw Wewnętrznych o stanowisku Sen
w sprawie ustawy o uregulowaniu stanu prawnego niektórych nieruchomości
pozostających we władaniu Polskiego Autokefalicznego Kościoła Prawosławn
(druki nr 2572 i 2580)." name="TYTUL">
```

- Wzbogacone o przynależność partyjną mówcy
 - W kadencjach 1-2: z danych o posłach (na koniec kadencji)
 - W kadencjach 3-8: z danych o głosowaniach (w momencie wypowiedzi)

P4: śmieci w danych

1 kadencja, 6 posiedzenie, 1 dzień, wypowiedź 17, poseł Witold Gadomski:

```
<P>    "Przyjąć ustawę o prowizorium budżetowym na okres  
od dnia 1 stycznia do 31 -iFÖ-ĐYÍòkDñ▪ũTJ&#9632;30-  
marca 1992 r. w brzmieniu przedłożenia rządowego z następującymi poprawkami:</P>
```

- Wykrywalne automatycznie
 - Czy w dokumencie występują niestandardowe tagi HTML?
 - Czy występują bardzo długie słowa?
 - Czy występują słowa zawierające znaki nielacińskie?
- Ręcznie utrzymywany plik poprawek (offsety znaków do wycięcia)

P4: niespójności

1 kadencja, 36 posiedzenie, 2 dzień, wypowiedź 34:

```
<meta content="Poseł Wojciech Arkuszewski" name="KTO">
```

```
...
```

```
<P><B><FONT SIZE="+1">Poseł Maciej Manicki</FONT></B></P>
```

```
<P> Panie Marszałku! Wysoka Izbo! Sojusz Lewicy Demokratycznej generalnie popiera
```

- Ręcznie rozstrzygnięte na podstawie analizy wystąpienia przewodniego marszałka

P4: niespójności

1 kadencja, 27 posiedzenie, 1 dzień, wypowiedź 14:

```
<meta content="Poseł Tadeusz Kowalczyk" name="KTO">
```

- Który to z dwóch posłów o tym imieniu i nazwisku (w większości odróżnialnych po imieniu ojca)?

1 kadencja, 14 posiedzenie, 4 dzień, wypowiedź 56:

```
<meta content="Poseł Andrzej Kamiński" name="KTO">
```

- Nie było takiego posła
- Często wariantywna pisownia nazwisk: *Edward Müller/Muller, Halina Nowina-Konopka/Nowina Konopka, Piotr Van/van der Coghén*

Smyrna

PROSTY KONKORDANCER

OBSŁUGUJĄCY JĘZYK POLSKI

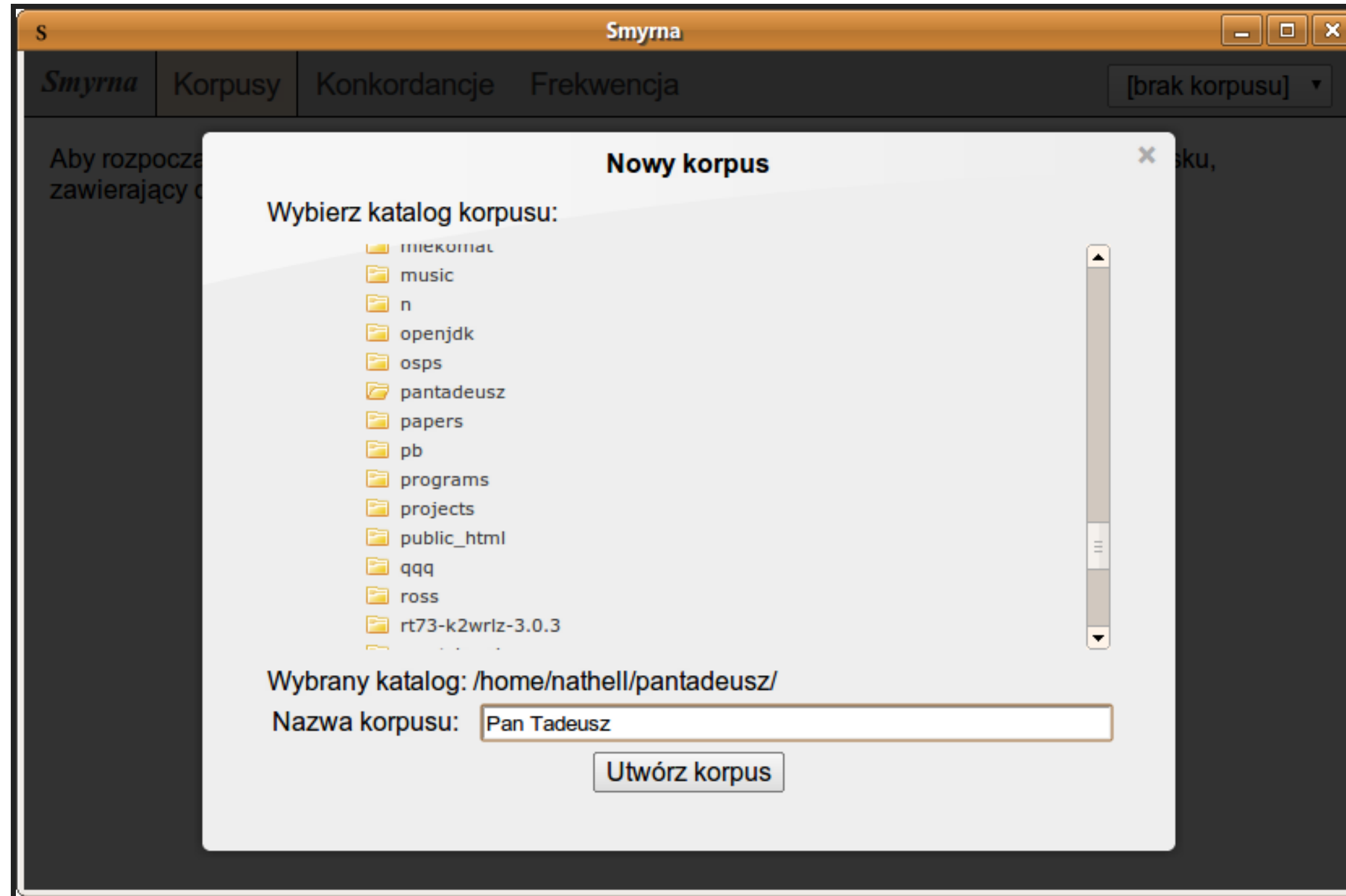


Polikarp ze Smyrny (ok. 70–156)

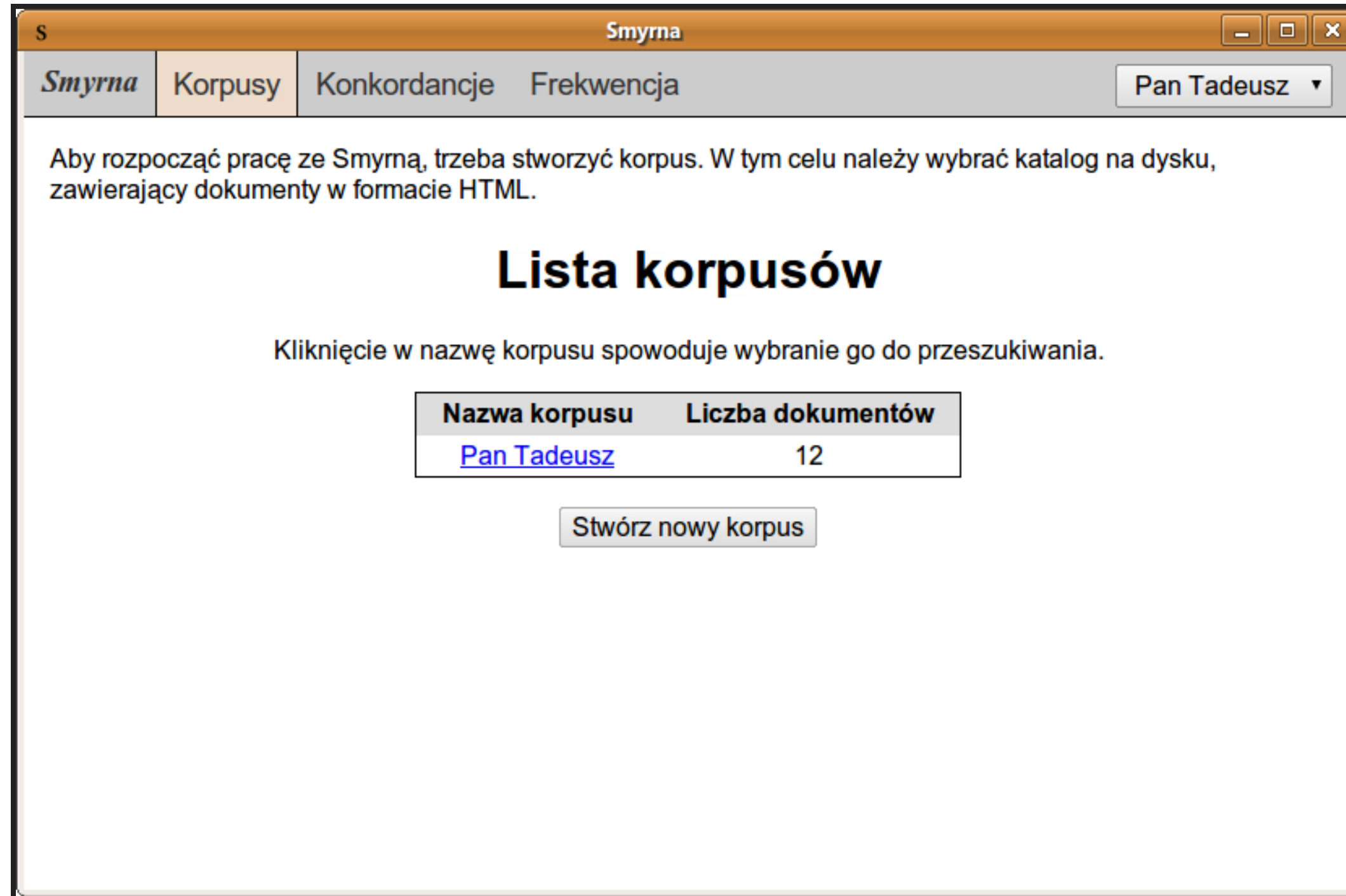
Smyrna 0.1 (2011)

- Konkordancer dla języka polskiego pozwalający na łatwe tworzenie własnych korpusów
- Obsługa dokumentów w formacie HTML
- Maksymalna prostota użytkowania
 - *Nieekspresywny* język zapytań
 - Odbiorcy: użytkownicy nietechniczni, nielingwiści
- Prezentacja dokumentów w ich oryginalnej postaci
- Analiza morfologiczna na bieżąco
- Generowanie list frekwencyjnych

Smyrna: tworzenie korpusu



Smyrna: lista korpusów



The screenshot shows a web browser window titled 'Smyrna'. The navigation menu includes 'Smyrna', 'Korpusy', 'Konkordancje', and 'Frekwencja'. A dropdown menu is open for 'Pan Tadeusz'. The main content area contains a message about creating a corpus, a heading 'Lista korpusów', an instruction to click on a corpus name, a table with one entry, and a 'Stwórz nowy korpus' button.

Aby rozpocząć pracę ze Smyrną, trzeba stworzyć korpus. W tym celu należy wybrać katalog na dysku, zawierający dokumenty w formacie HTML.

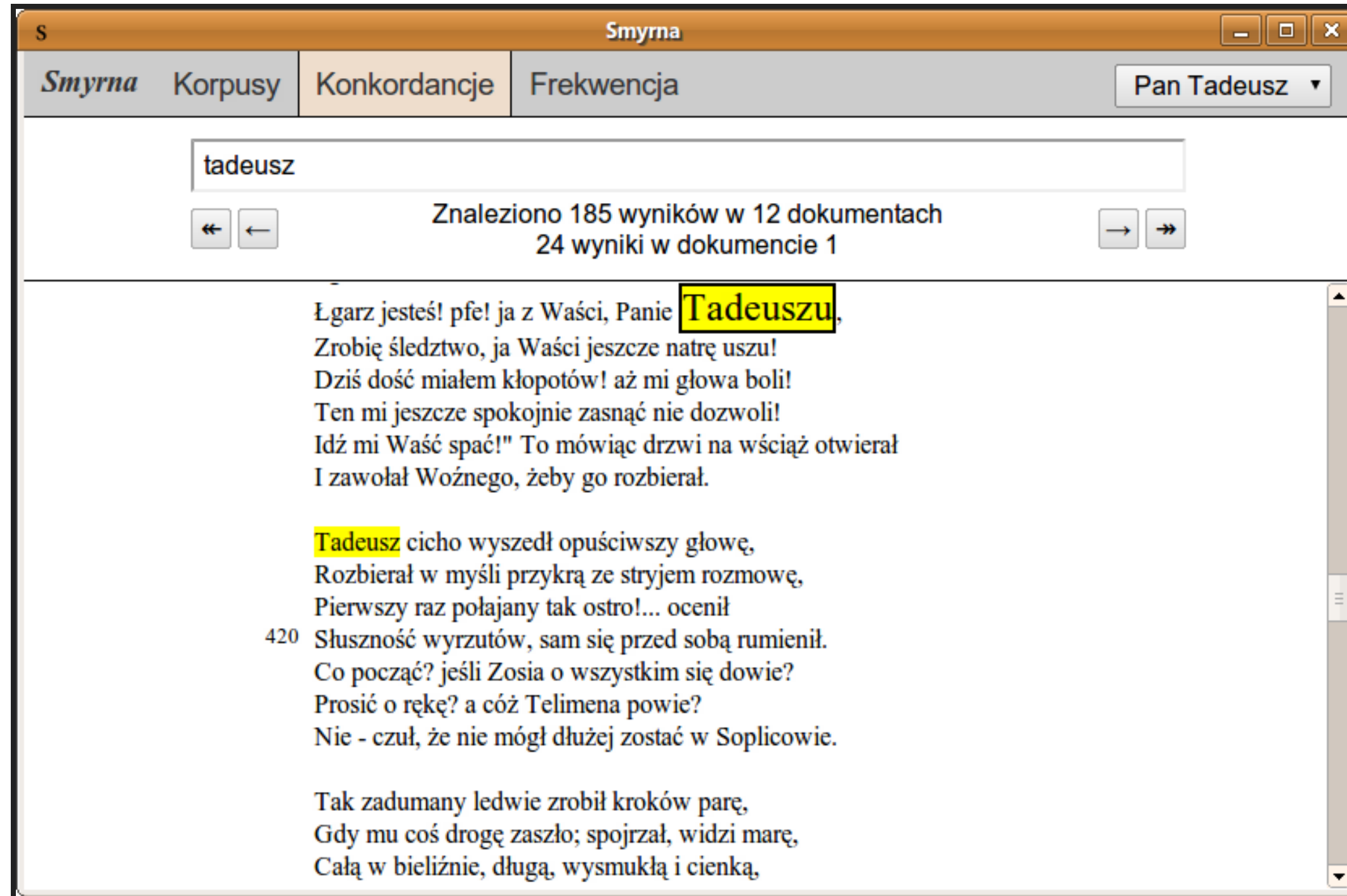
Lista korpusów

Kliknięcie w nazwę korpusu spowoduje wybranie go do przeszukiwania.

Nazwa korpusu	Liczba dokumentów
Pan Tadeusz	12

Stwórz nowy korpus

Smyrna: wyszukiwanie



The screenshot shows a web application window titled "Smyrna". The interface includes a navigation bar with tabs for "Smyrna", "Korpusy", "Konkordancje", and "Frekwencja". A dropdown menu on the right is set to "Pan Tadeusz". A search input field contains the text "tadeusz". Below the input, a status bar indicates "Znaleziono 185 wyników w 12 dokumentach" and "24 wyniki w dokumencie 1". The main content area displays text from a literary work, with the word "Tadeuszu" highlighted in yellow in the first line. The text continues with several lines of dialogue and narrative.

Smyrna

Korpusy Konkordancje Frekwencja Pan Tadeusz

tadeusz

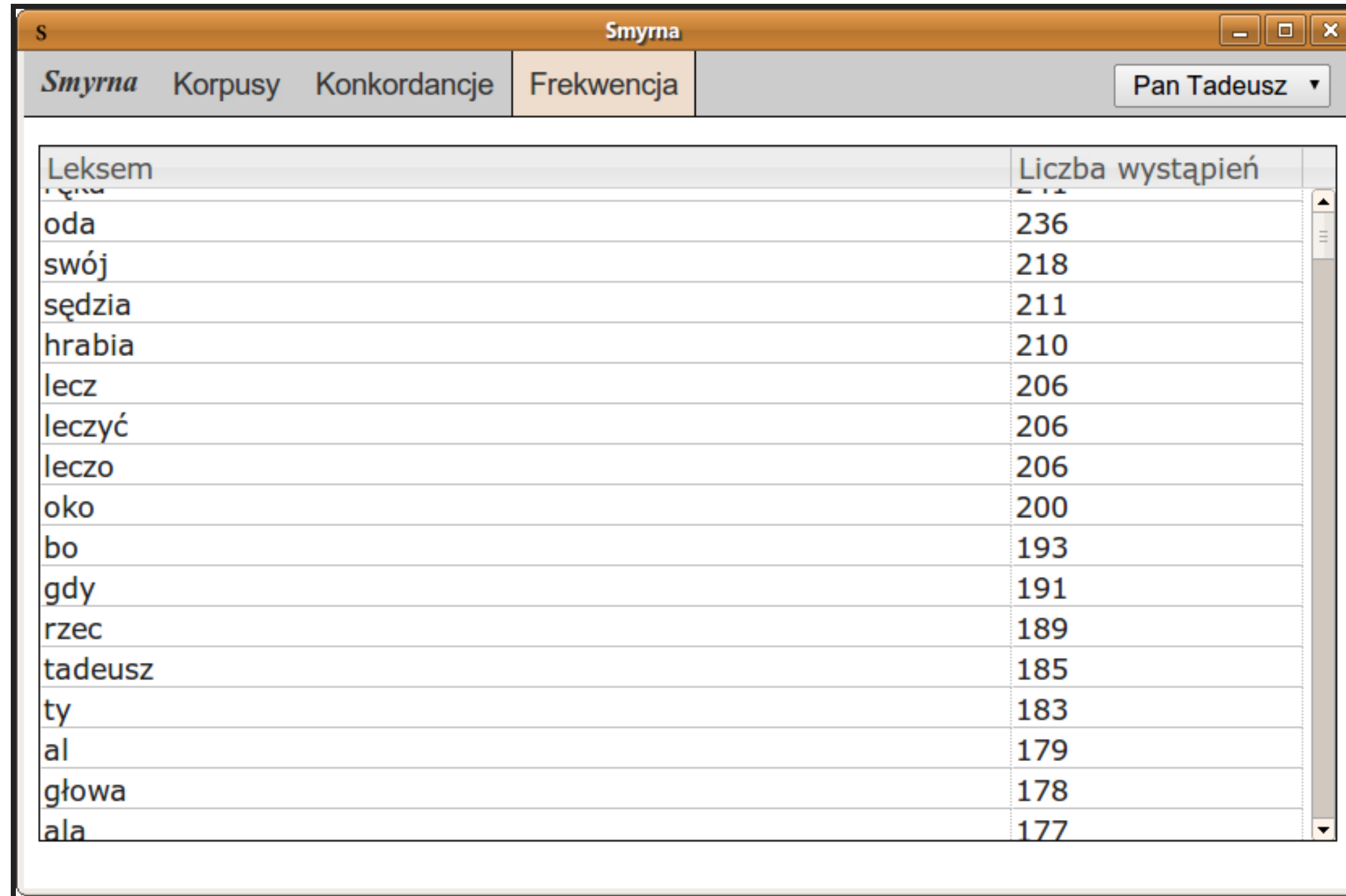
Znaleziono 185 wyników w 12 dokumentach
24 wyniki w dokumencie 1

Łgarz jesteś! pfe! ja z Waści, Panie **Tadeuszu**,
Zrobię śledztwo, ja Waści jeszcze natrę uszu!
Dziś dość miałem kłopotów! aż mi głowa boli!
Ten mi jeszcze spokojnie zasnąć nie dozwoli!
Idź mi Waść spać!" To mówiąc drzwi na wścież otwierał
I zawołał Woźnego, żeby go rozbierał.

Tadeusz cicho wyszedł opuściwszy głowę,
Rozbierał w myśli przykrą ze stryjem rozmowę,
Pierwszy raz połajany tak ostro!... ocenił
420 Słuszność wyrzutów, sam się przed sobą rumienił.
Co począć? jeśli Zosia o wszystkim się dowie?
Prosić o rękę? a cóż Telimena powie?
Nie - czuł, że nie mógł dłużej zostać w Soplicowie.

Tak zadumany ledwie zrobił kroków parę,
Gdy mu coś drogę zaszło; spojrział, widzi marę,
Całą w bieliźnie, długą, wysmukłą i cienką,

Smyrna: lista frekwencyjna



Leksem	Liczba wystąpień
oda	236
swój	218
sędzia	211
hrabia	210
lecz	206
leczyć	206
leczo	206
oko	200
bo	193
gdy	191
rzec	189
tadeusz	185
ty	183
al	179
głowa	178
ala	177

Smyrna 0.3 (2016)

- Obsługa korpusów bogatych w metadane
- Obsługa sporych korpusów (P4: ok. 280 tys. dokumentów, ok. 200 mln słów)
- Tworzenie obszarów z filtrów i wyszukiwanie tylko w obszarach
 - `within` z Poliqarpa na sterydach
- Porównania obszarów
- Język zapytań nadal nieekspresywny
- Nowy, zwarty, binarny format korpusu, inspirowany Poliqarpem
 - Jeden plik!

Smyrna 0.3: technicznie

- Aplikacja webowa, działająca lokalnie na komputerze użytkownika
- Backend: Clojure
- Interfejs użytkownika: ClojureScript (Reagent)
 - Poprzednie wersje: CoffeeScript + Backbone.js
- Ok. 1200 linii kodu
- Przepisana w zasadzie od zera

Lematyzacja

- Lematyzator unigramowy, wytrenowany na milionowym podkorpusie NKJP
- Jeśli nie ma w częstościach NKJP: najczęstsza forma podstawowa z Morfologia (PoliMorf)
- Wydzielona biblioteka do lematyzacji (Polelum)
- Może kiedyś: prawdziwe tagowanie (Pantera?)

Format korpusu

- Pojedynczy plik ZIP *bez kompresji*
 - Można mmapować poszczególne kawałki pliku
- Główny element: obraz korpusu (*image*)
 - Zawiera *sparsowane dokumenty HTML*, zserializowane i skompresowane kanonicznym kodem Huffmana
 - Rozmiar alfabetu: ok. 500 000 symboli
 - Rodzaje symboli: tekst, tag otwierający, atrybut HTML, wartość atrybutu, symbol NOSPACE
- Pozostałe elementy: słowniki (w formacie CFSA2 morfologik-stemming), pliki offsetów, dane do dekompresji, indeks odwrotny

DEMO

DZIĘKUJĘ!

Daniel Janus

Rebased

<https://github.com/nathell/skyscraper>

<http://smyrna.danieljanus.pl>