Uniwersytet Warszawski
Wydział Matematyki, Informatyki i Mechaniki

Maciej Ogrodniczuk
Nr albumu: 159168

Wykorzystanie SGML i TEI do zapisu polskich danych lingwistycznych

Praca magisterska
na kierunku INFORMATYKA

Praca wykonana pod kierunkiem
prof. Janusza S. Bienia
Instytut Orientalistyki UW

Wrzesień 2000

Pracę przedkładam do oceny
Data Podpis autora pracy

Praca jest gotowa do oceny przez recenzenta
Data Podpis kierującego pracą

Streszczenie

Praca przedstawia koncepcję zapisu polskich danych lingwistycznych zgodną z normami SGML (Standard Generalized Markup Language) i TEI (Text Encoding Initiative). Po wprowadzeniu do tematyki obu formalizmów oraz opisie oferowanych przez nie ogólnych schematów reprezentacji danych autor przedstawia dostępne w ich ramach mechanizmy zapisu wielopoziomowej interpretacji lingwistycznej. Zasadniczą część pracy stanowi projekt zapisu informacji lingwistycznych dla tekstów polskich wraz z dyskusją wyboru sposobu kodowania. Opis kończy aplikacja proponowanego schematu zapisu do danych *Słownika frekwencyjnego polszczyzny współczesnej* — obszernego zbioru tekstów o dużej różnorodności.

Słowa kluczowe

SGML, TEI, teksty, korpusy, słowniki, język polski

Klasyfikacja ACM

Document and Text Processing/Electronic Publishing (I.7.4)

Spis treści

Użyte oznaczenia i skróty

ASCII American Standard Code for Information Interchange — Amerykański Standardowy Kod do Wymiany Informacji; 7-bitowy kod o ponad 30-letnim rodowodzie, będący podstawą innych kodów 7- i 8-bitowych, jego bezpośrednie wykorzystanie obecnie zanika.

DTD Document Type Definition — Definicja Typu Dokumentu, sformalizowany opis struktury tekstu, tworzący wraz z nim dokument SGMLowy.

HTML Hypertext Markup Language — Hipertekstowy Język Adiustacyjny; niezwykle popularna aplikacja SGMLa do opisu (przede wszystkim wizualnego) zawartości stron WWW.

ISO International Organization for Standardization — Międzynarodowa Organizacja Normalizacyjna ISO; federacja krajowych organizacji normalizacyjnych, ustanawiająca normy międzynarodowe większością 3/4 oddanych głosów państw członkowskich.

ISO 10646 Norma międzynarodowa ustanawiająca 4-bajtowy, ogólnoświatowy, uniwersalny zestaw kodowy znaków.

ISO 646 IRV Norma ISO definiująca 7-bitowy kodowy zestaw znaków będący podstawą kodów używanych obecnie w większości komputerów. Od 1991 roku ISO 646 IRV (International Reference Version — Międzynarodowa Wersja Wzorcowa) jest w pełni zgodny z ASCII.

ISO 8859-2 Arkusz normy ISO 8859, znany też pod nazwą Latin-2; 8-bitowy zestaw znaków rezerwujący osobne kody m. in. dla polskich liter.

SGML Standard Generalized Markup Language — Standardowy Uogólniony Język Adiustacyjny; zdefiniowany w międzynarodowej normie ISO 8879 z 1986 roku formalizm zapisu tekstu łączący treść tekstu z informacją o jego strukturze. Od czasu powstania jest przedmiotem żywego zainteresowania tak środowisk naukowych jak biznesowych (wydawców, projektantów systemów zarządzania informacją itp.)
Bieżąca wersja normy obejmuje poprawki techniczne o oznaczeniu kodowym TC3 i pochodzi z końca 1998 roku.

TEI Text Encoding Initiative — Inicjatywa Kodowania Tekstu — projekt wprowadzający spójny standard zapisu informacji o strukturze tekstu z wykorzystaniem predefiniowanych zestawów znaczników.

Wstęp

Praca miała na celu opracowanie i weryfikację koncepcji zapisu polskich danych lingwistycznych zgodnej ze standardami SGML i TEI.

W rozdziale 1 podaję definicje i prezentuję formalizmy wykorzystywane w kodowaniu tekstów zgodnych z SGML i TEI. Jest to krótki przegląd cech obu standardów, mogący odegrać rolę zwięzłego wprowadzenia w ich specyfikę i przedstawienia rodzajów ich zastosowań.

Rozdział 2 opisuje konkretne mechanizmy TEI mogące zostać wykorzystane do zapisu wielopoziomowej interpretacji tekstu. Ich przedstawienie ma na celu analizę ich przydatności do reprezentacji informacji lingwistycznych.

Rozdział 3 przestawia projekt zapisu informacji lingwistycznych dla tekstów polskich wraz z dyskusją wyboru mechanizmu kodowania. Końcowa część rozdziału, mogąca stać się uzupełnieniem przeglądu własności SGML i TEI, poświęcona jest prezentacji metody zapisu polskich znaków w opisywanych standardach.

Rozdział 4 jest opisem zastosowania proponowanego schematu reprezentacji informacji lingwistycznej do zapisu przykładowego zbioru danych — plików *Słownika frekwencyjnego polszczyzny współczesnej* [16]. Początkowa część rozdziału przedstawia budowę słownika i historię przeprowadzonych wcześniej prac.

Oprócz plików słownika frekwencyjnego w pracy wykorzystałem także wyniki grantu KBN nr 8 T11C 002 13 *Zestaw testów do weryfikacji i oceny analizatorów języka polskiego* zrealizowanego pod kierunkiem dr hab. Janusza S. Bienia. Zbiory danych przygotowane przeze mnie potrzeby tego projektu stały się dobrym materiałem do rozpoznania możliwości i ograniczeń struktury dokumentów TEI.

Rozdział 1

Wprowadzenie do SGML i TEI

1.1 SGML jako standard publikacji tekstów

W niniejszym krótkim opisie postaram się przedstawić podstawowe pojęcia wprowadzane przez standard definiujący SGML i przybliżyć reguły rządzące konstrukcją dokumentów. Kwestie techniczne i składniowe zostały ograniczone do minimum, co w większości przypadków oznacza zasygnalizowanie składni na przykładach.

Zainteresowanych pogłębieniem swoich umiejętności SGMLowych odsyłam w pierwszym rzędzie do szeroko dostępnych samouczków (np. sieciowego dokumentu *Getting Started with SGML* [2] czy funkcjonującego już niemal samodzielnie drugiego rozdziału Wytycznych TEI *A Gentle Introduction to SGML* [3], na którym w większości opiera się niniejsze wprowadzenie).

Warto też zdawać sobie sprawę, że dostępnych jest coraz więcej narzędzi umożliwiających weryfikację poprawności, przekształcanie czy wydruk struktur dokumentów. Wiele z nich opisuje np. książka *SGML CD* [8] z dołączoną płytą CD-ROM zawierającą wykonywalne wersje omawianych programów.

1.1.1 Idea SGML

SGML jest formatem zapisu tekstów elektronicznych wprowadzonym normą ISO 8879 z grudnia 1986 roku [4][1] jako odpowiedź na potrzebę standaryzacji metod reprezentacji i wymiany dokumentów tekstowych.

[1] Pełny, obszernie skomentowany tekst normy podaje Charles Goldfarb w książce [13]. W postaci elektronicznej dostępne są także dokumenty TC2 i TC3 [12] (z ang. technical corrections) wprowadzające poprawki do tekstu normy i stanowiące wraz z nią aktualną wersję standardu.

Idea SGML opiera się na pojęciu tzw. *generycznej adiustacji tekstu* (ang. generic markup), czyli takiego jego opisu, który przyporządkowuje fragmentom tekstu określoną strukturę logiczną. W przeciwieństwie do *adiustacji konkretnej* (ang. specific markup) w opisie tym nie mówi się nic o wyglądzie tekstu, rodzaju i stylu czcionek użytych do jego formowania, stosowanych odstępach itp. Rozdzielenie tych dwóch informacji sprawia, że uzyskując w prosty sposób pełną kontrolę nad znaczeniem poszczególnych części dokumentu, nie tracimy nic z możliwości dowolnego przypisania im dodatkowych własności — także typograficznych.

Innym ważnym założeniem SGMLa jest zapewnienie dokumentom maksymalnej przenośności między różnymi środowiskami komputerowymi. Jako że głównym powodem problemów powstających w procesie wymiany dokumentów jest niezgodność zestawów kodowych znaków, SGML został wyposażony w mechanizm zastępowania napisów, który może zostać wykorzystany do utworzenia bezpiecznych, opisowych odwołań — *całostek* (ang. entities) dla nieprzenośnych symboli.

Ujmując rzecz dokładniej, SGML jest metajęzykiem — definiuje metody opisu języków adiustacyjnych, określających w ścisły sposób strukturę dokumentów. Najpopularniejsze z takich języków pochodnych to odkrywany wciąż na nowo HTML (ang. Hypertext Markup Language — Hipertekstowy Język Adiustacyjny), będący zastosowaniem SGMLa do projektu opisu wyglądu stron internetowych i jego dynamiczny krewny XML (ang. eXtensible Markup Language — Rozszerzalny Język Adiustacyjny), pozwalający ponadto w znacznie bardziej elastyczny sposób określać znaczenie prezentowanych danych.

Warto podkreślić, że norma definiująca SGML stała się początkiem serii standardów dotyczących przetwarzania danych tekstowych (Information Processing — Text and Office Systems). Trzy najważniejsze z nich to *Specyfikacja Stylu Dokumentu i Język Semantyczny* (DSSSL, ang. Document Style Specification and Semantics Language), *Standardowy Format Wymiany Dokumentów* (SDIF, ang. Standard Document Interchange Format) i *Standardowy Język Opisu Strony* (SPDL, ang. Standard Page Description Language).

1.1.2 Typ dokumentu i jego definicja

Standard SGML wprowadza pojęcie *typu dokumentu*[2] (ang. document type), związane z wewnętrzną hierarchią jego części składowych oraz *definicji typu*

[2]Polskie odpowiedniki terminów angielskich pochodzą z tłumaczenia reguł składni SGML autorstwa J. S. Bienia [6].

dokumentu (DTD, ang. Document Type Definition) — formalnego opisu jego budowy zawierającego informacje o nazwach przyporządkowanych częściom składowym tekstu (*elementach*), ich dodatkowych własnościach (*atrybutach*) i zależnościach pomiędzy elementami, łączących je w strukturę drzewiastą.

Zapis DTD jest listą deklarowanych jednostek, poprzedzoną słowem kluczowym `DOCTYPE` i nazwą elementu głównego. Oto składniowy szkielet przykładu definicji:

```
<!DOCTYPE antologia [ ... definicje składników ... ]>
```

Składnikami DTD są deklaracje elementów i całostek wykorzystywanych w dokumencie. Zamiast bezpośredniej specyfikacji ich definicji możemy także wykorzystać możliwość włączenia definicji z zewnętrznego pliku, używając *identyfikatora systemowego* (ang. system identifier) o postaci

```
<!DOCTYPE antologia SYSTEM "antolog.dtd" [ ... ]>
```

lub *publicznego identyfikatora formalnego* (ang. formal public identifier) — napisu o ustalonej strukturze reprezentującego upublicznioną definicję typu dokumentu (która może być też formalnie zarejestrowana w agencji autoryzowanej przez ISO[3]). Oto przykładowa deklaracja wykorzystująca identyfikator publiczny:

```
<!DOCTYPE tei.2 PUBLIC "-//TEI P3//DTD Main Document Type//EN"
  [ ... ]>
```

1.1.3 Elementy

Do zapisu konstrukcji dokumentów SGML używa *elementów* — podstawowych jednostek strukturalnych. Ich *identyfikatory generyczne* (ang. generic identifiers) nazywają wydzielone fragmenty tekstu, co doprowadza do powstania usystematyzowanego opisu jego budowy[4].

[3] O dokonaniu rejestracji świadczy znak plusa rozpoczynający identyfikator — DTD TEI (którego identyfikator publiczny przedstawiono w przykładzie) nie zostało zatem zarejestrowane.

[4] Mimo możliwości nadawania elementom znaczących nazw, nie są im przypisywane żadne dodatkowe interpretacje — znaczące jest wyłącznie ich położenie w stosunku do innych części składowych dokumentu.

Zapis elementów wymaga użycia specjalnych znaczników — początkowego i końcowego[5], składających się z nazwy ujętej w nawiasy kątowe (i znaku ukośnika dla znacznika końcowego) oraz ewentualnego zbioru dodatkowych własności — *atrybutów* — umieszczanych bezpośrednio po nazwie określonego elementu[6]:

```
<księga nr=1>
   <tytuł>Gospodarstwo
   <treść>
      <wers nr=1>Litwo! Ojczyzno moja! ty jesteś jak zdrowie;
      <wers nr=2>Ile cię trzeba cenić, ten tylko się dowie,
      <wers nr=3>Kto cię stracił.
   ...
   </treść>
 </księga>
```

Oprócz wypełnienia tekstem element może posiadać zawartość pustą lub składającą się z innych elementów. Formalnym sposobem określenia tej zawartości jest włączenie do DTD opisu o ściśle określonej składni, definiującego tzw. *model zawartości* elementu (ang. content model). Zawiera on ponadto informacje określające, czy wymagany jest w tekście zapis początkowego i końcowego znacznika elementów (wyraźnie zaznaczona i jednoznacznie interpretowalna struktura dokumentu może umożliwiać opuszczanie niektórych znaczników).

Do opisu zawartości stosowane są znaczniki o długiej informatycznej tradycji:

`*` — 0 lub więcej
`+` — 1 lub więcej
`?` — 0 lub 1

} wystąpień elementu

i

`,` — sekwencja
`|` — alternatywa
`&` — dowolna kolejność

} wystąpień elementów

[5] Znaczniki te mogą być opuszczane, jeśli zezwala na to struktura dokumentu i zapis w DTD (opisany niżej). W podanym na następnej stronie przykładzie zjawisko to ilustruje znacznik otwierający `<tytuł>` oraz znaczniki wersów — nie posiadają one odpowiedników zamykających, gdyż koniec tytułu i wersu może być rozpoznany poprzez wystąpienie następnego znacznika (odpowiednio `<treść>` i oznaczenia kolejnego wersu lub końca treści księgi).

[6] Znaczniki elementów są wymiennie nazywane *tagami*, zaś tekst z wyróżnioną przy ich pomocy strukturą — tekstem *otagowanym*. W dalszej części pracy, zgodnie z obowiązującą konwencją, określeń *tag* i *znacznik* będę używał wymiennie.

oraz operator grupowania w postaci nawiasów okrągłych. Stosowane są też specjalne oznaczenia opisujące zawartość tekstową (`#PCDATA`, ciąg znaków interpretowany jako liść drzewa struktury dokumentu) i pustą (`EMPTY`, brak dalszego rozbicia struktury, brak zawartości znakowej). Oto fragment DTD zapisanego z wykorzystaniem podanych reguł:

```
<!ELEMENT wiersz    - - (tytuł?, zwrotka+)>
<!ELEMENT tytuł     - O (#PCDATA)>
<!ELEMENT zwrotka   - O (wers+)>
<!ELEMENT wers      O O (#PCDATA)>
```

Powyższa deklaracja wprowadza element `<wiersz>` posiadający opcjonalny `<tytuł>` i składający się z co najmniej jednej zwrotki. Każda `<zwrotka>` to co najmniej jeden `<wers>`, zaś zarówno `<tytuł>`, jak i `<wers>` zawierają już wyłącznie „czysty" (tj. nie zawierający znaczników) tekst. Między nazwą elementu a modelem zawartości znajduje się oznaczenie wymagań zapisu znacznika początkowego i końcowego w postaci pary znaków `-` (znacznik wymagany) i `O` (znacznik opcjonalny).

Atrybuty elementów — związane z nimi wartości, pozostające jednak bez wpływu na zawartość elementów — muszą również zostać zadeklarowane. Opis atrybutu, oprócz jego nazwy i nazwy odpowiadającego mu elementu zawiera listę dozwolonych wartości, wartość domyślną lub oznaczenie kategorii elementu:

- `ID` — unikatowa wartość identyfikatora,
- `NUMBER` — oznaczenie liczby,
- `IDREF` — wskaźnik do innego elementu (identyfikowanego z wykorzystaniem atrybutu typu `ID`),
- `CDATA` — napis o rozszerzonej zawartości (w jego skład mogą wchodzić wszystkie znaki z dostępnego zestawu; w przypadku wystąpienia znaczników elementów parser SGML zinterpretuje je jako znaki pozbawione specjalnego znaczenia).

Oto przykładowa deklaracja atrybutu:

```
<!ATTLIST wiersz
          lp     NUMBER                                #REQUIRED
          stan   (odrzucony | doPoprawki | doDruku)    odrzucony>
```

Definicje atrybutów opatrywane są kwalifikatorami wartości:

- `#REQUIRED` — wartość wymagana, musi być podana,
- `#CURRENT` — w razie niepodania wartości zostanie przyjęta wartość używana ostatnio,
- `#IMPLIED` — wartość nie musi zostać podana.

1.1.4 Całostki ogólne

Wspomniana już idea całostki wyposaża SGML w mechanizm zastępowania napisów. W najprostszy sposób realizują go *całostki ogólne* (ang. general entities), których zadaniem jest ułatwienie tworzenia zarówno tekstu dokumentu, jak i jego DTD poprzez włączanie do zasadniczej części dokumentu napisów stałych lub stanowiących zawartość zewnętrznych plików.

Definicja całostki ogólnej przyjmuje jedną z dwóch postaci:

```
<!ENTITY MIMUW "Wydział Matematyki, Informatyki i Mechaniki UW">
```

lub

```
<!ENTITY ZasadyRekrutacji SYSTEM "rekmimuw.txt">
```

Pierwsza z nich określa tzw. *całostkę wewnętrzną* (ang. internal entity) — napis o podanej wartości. Druga definiuje *całostkę zewnętrzną* (ang. external entity), której wartością jest, jak w przypadku DTD, element zewnętrzny (tu: zawartość pliku o podanej nazwie).

Po zadeklarowaniu całostki jej wartość może być wielokrotnie używana w tworzonym dokumencie. Poprzedzając nazwę całostki znakiem ampersanda i kończąc ją średnikiem wydajemy polecenie wstawienia określonego nią tekstu w miejscu odwołania.

```
&MIMUW; - rekrutacja studentów
&ZasadyRekrutacji;
```

1.1.5 Sekcje wyróżnione i całostki parametryczne

Specjalną funkcję spełniają w dokumencie *całostki parametryczne* (ang. parameter entities). Są to całostki o zawężonym zbiorze wartości i ściśle określonej funkcji, używane do sterowania przetwarzaniem dokumentu przez parser SGMLowy. W odróżnieniu od całostek ogólnych identyfikowane są znakiem procenta.

Jedną z możliwości wykorzystania całostek parametrycznych jest tworzenie tzw. *sekcji wyróżnionych* (ang. marked sections) — wydzielonych części tekstu włączanych do dokumentu w zależności od wartości całostki[7], określanej słowami kluczowymi `INCLUDE` i `IGNORE` (odpowiednio polecenia dołączenia i wykluczenia opisanego fragmentu).

Poniższy przykład przedstawia definicję całostki i jej użycie w sekcji wyróżnionej:

```
<!ENTITY % ostrzeżenie 'INCLUDE'>

Przypominamy o terminowym opłacaniu abonamentu.
<![ %ostrzeżenie; [
W przypadku nieuregulowania należności do końca bieżącego
miesiąca zmuszeni będziemy do naliczenia karnych odsetek.
]]>
```

Pamiętając, że użycie całostki ogranicza się wyłącznie do zastępowania tekstu, możemy tworzyć bardzo skomplikowane dokumenty, wpływając nie tylko na ich zawartość, ale i na postać samego DTD (z możliwości tej będziemy intensywnie korzystać podczas tworzenia dokumentów zgodnych z formatem TEI).

Całostki parametryczne mogą być również wykorzystane do bezpośredniego zastępowania tekstu, pełnią jednak wówczas rolę szczególną — umożliwiają wpływanie na postać DTD dokumentu, na przykład poprzez rozszerzenie listy atrybutów elementu o wartości domyślne („dziedziczone"):

```
<!ENTITY % atrybutyGlobalne '
           id      ID      #IMPLIED
           n       CDATA   #IMPLIED
           lang    IDREF   #CURRENT' >
```

[7] Szczegółowy opis kontroli sterowania w SGMLu zawiera np. rozdział *Marked sections* w Wytycznych TEI [3] lub *Using marked sections* w książce *Practical SGML* [14].

```
<!ELEMENT segment  - - (#PCDATA)>
<!ATTLIST segment
          %atrybutyGlobalne;
          typ   CDATA      #IMPLIED>
```

1.1.6 Deklaracja SGMLowa i dokument SGMLowy

Poprawny *dokument SGMLowy* (ang. SGML document) składa się z *prologu* (ang. SGML prolog) zawierającego *deklarację SGMLową* (ang. SGML declaration) i definicję typu dokumentu oraz *egzemplarza dokumentowego* (ang. document instance) — właściwego tekstu, oznaczonego (otagowanego) zgodnie ze specyfikacją podaną w DTD.

Celem deklaracji jest stworzenie środowiska do zapisu dokumentu, co dokonuje się poprzez określenie m. in. dozwolonego zestawu znaków (szerszy opis tej części deklaracji zawiera rozdział 3.3.1), symboli używanych do ograniczania znaczników, dopuszczalności ich skracania i pomijania, rozróżniania kasztowości liter i definicji innych własności leksykalnych. Pominięcie deklaracji oznacza niejawne włączenie jej specjalnej postaci — tzw. *wzorcowej składni konkretnej* (ang. reference concrete syntax), predefiniującej wymagane własności z wykorzystaniem najczęściej używanych wartości[8]. W poniższym dokumencie przykładowym, używającym wzorcowej składni konkretnej, wyraźnie pokazano jego dwuczęściową budowę (DTD i egzemplarz dokumentu):

```
<!DOCTYPE dokument [
  <!ELEMENT dokument   - - (element+)>
  <!ELEMENT element    - - (#PCDATA)>
  <!ELEMENT calostka  "dokument przykładowy">
]>

<dokument>
   <element>To jest &calostka;</element>.
</dokument>
```

[8]Mimo wygodnego sposobu na uniknięcie każdorazowego zapisu pełnej deklaracji, jaki zapewnia wzorcowa składnia konkretna, do niektórych zastosowań takie podejście może okazać się niewystarczające. Przykładem jest sam format TEI, wymagający np. możliwości użycia dłuższych niż standardowe 8 znaków nazw identyfikatorów generycznych. Także poprawny zapis w dokumencie polskich znaków wymaga zmiany deklaracji (patrz sekcja 3.3). Szczegółowy wykaz pozycji deklaracji SGMLowej zawiera np. rozdział *The SGML declaration* książki *Practical SGML* [14].

1.2 TEI — SGMLowy schemat kodowania tekstów

TEI (ang. Text Encoding Initiative — Inicjatywa Kodowania Tekstów) to nazwa wspólnego projektu ACH (ang. Association for Computers and the Humanities — Stowarzyszenie Informatyczno-Humanistyczne), ACL (ang. Association for Computational Linguistics — Stowarzyszenie Linwistyki Komputerowej) i ALLC (ang. Association for Literary and Linguistic Computing — Stowarzyszenie Obliczeń Literackich i Lingwistycznych), mającego określić spójny standard zapisu informacji tekstowej dla celów lingwistycznych i humanistycznych. Zalążkiem TEI była konferencja ACH w Vassar College w listopadzie 1987 roku, zwołana w celu przedyskutowania możliwości stworzenia takiego standardu oraz określenia jego ewentualnego zakresu i zawartości. Ustalono, że istnieje potrzeba określenia jednolitego, dostatecznie ogólnego formatu wymiany danych humanistycznych, stworzenia reguł jego składni i określenia metajęzyka stosowanego do jego zapisu.

1.2.1 Podstawowe założenia TEI

Główny cel projektu określono jako wypracowanie zbioru reguł zapisu i wymiany danych tekstowych o możliwie szerokim zakresie tematycznym[9], określenie zalecanej formalnej składni takiego zapisu i metod konwersji istniejących zbiorów do nowego formatu. Dodatkową miarą zaangażowania podmiotów tworzących nowy standard stała się deklaracja licznych właścicieli dużych archiwów tekstów dołożenia wszelkich starań do szybkiego wdrożenia uzyskanych wyników oraz zakończone sukcesem poszukiwania funduszy.

Zaawansowanie prac opisywano w wielu dokumentach publikowanych na bieżąco; obecnie najpełniejszym zbiorem wyników są wydane w kwietniu 1994 roku *Wytyczne dotyczące Elektronicznego Zapisu i Wymiany Tekstów* (ang. Guidelines for Electronic Text Encoding and Interchange [3]), określane też nazwą *TEI P3* (trzecia propozycja TEI)[10]. Dokument ten definiuje SGML

[9] Humanistyczny i lingwistyczny rodowód TEI nie ograniczył jednak rodzaju reprezentowalnych danych do postaci tekstowej — przyjęte założenia wspierają także mechanizmy łączenia tekstów m. in. z danymi dźwiękowymi i wizualnymi.

[10] W połowie 1999 roku pojawiła się poprawiona wersja *Wytycznych*, będąca odpowiedzią na ponad 200 komentarzy i propozycji zgłoszonych od czasu ukazania się dokumentu *TEI P3*. Dokonano w niej korekty większości wykrytych błędów składniowych i logicznych, poważniejsze zmiany nie zostały jednak wprowadzone z powodu konieczności zachowania zgodności formatu z powstałymi wcześniej zbiorami danych (dodano zaledwie jeden nowy element, `<ab>`). W chwili obecnej trwają prace nad czwartą wersją propozycji TEI (*P4*).

jako bazę nowego formatu zapisu tekstów i wprowadza mechanizmy kodowania licznych własności dokumentów różnych typów. Dla większości z nich zasady zapisu oparto na istniejących standardach, dla niektórych owe standardy wypracowano od podstaw.

Wytyczne TEI opisują w dość szczegółowy sposób możliwość nadawania tekstom wielu typów odpowiedniej konstrukcji logicznej z wykorzystaniem takich cech SGML jak możliwość definiowania struktur równoległych czy tworzenia łączy hipertekstowych w obrębie jednego lub między wieloma wzajemnie zewnętrznymi tekstami. Pozostałe zalety opisanego formalizmu to ścisłość definicji, jej łatwość w użyciu i rozszerzalność.

1.2.2 Składniki tekstu zgodnego z TEI

Schemat kodowania tekstu opisany w *Wytycznych TEI* wprowadza pojęcie *zestawów tagów* — zbiorów znaczników, z których, według przyjętych reguł SGMLowych, tworzone jest DTD dokumentu zgodnego z założeniami TEI.

Składowe zestawy tagów należą do jednej z trzech kategorii:

- *tagów rdzennych i nagłówkowych* (ang. core and header tag sets), czyli elementów składowych tekstów wszystkich rodzajów, dostępnych dla wszystkich dokumentów;
- *tagów bazowych* (ang. base tag sets) — składników tekstów poszczególnych rodzajów; dla danego dokumentu może być określony tylko jeden z nich;
- *tagów dodatkowych* (ang. additional tag sets) — elementów używanych w procesie szczegółowej analizy lub przetwarzania tekstu dowolnego typu.

Zestaw tagów rdzennych zawiera elementy niezależne zarówno od typu tekstu, jak i od rodzaju jego analizy, często występujące w dokumentach i zasługujące na specjalną uwagę. Są to na przykład oznaczenia podziału tekstu na akapity (znacznik `<p>`) — szeroko rozumiane jednostki segmentacji na poziomie wyższym od podziału zdaniowego, ogólne znaczniki obsługi cytatów (`<q>`), fragmentów obcojęzycznych (`<foreign>`), nazw własnych (`<name>`), oznaczeń czasu (`<date>`, `<time>`) oraz interpunkcji. Nagłówek TEI (opisany szczegółowo w *Dodatku B*) umożliwia dołączenie do zapisywanego tekstu elektronicznego pewnych informacji bibliograficznych, dokumentujących proces jego tworzenia i kodowania.

Określenie typu opisywanego tekstu odbywa się poprzez wybór jednego z ośmiu zestawów tagów bazowych, co oznacza jego zakwalifikowanie do jednej z poniższych kategorii:

- proza,
- poezja,
- dramat,
- zapis wypowiedzi ustnych,
- słownik drukowany,
- terminologia,
- antologia tekstów,
- nieuporządkowany zbiór tekstów różnych typów.

Pierwsze sześć klas zawiera dokumenty jednorodne w swej budowie, których klasyfikacja nie sprawia w większości przypadków znaczących problemów. Dwie pozostałe służą do zapisu tekstów łączących cechy wielu typów.

Zestawy tagów dodatkowych stosowane są do zapisu informacji określonego typu, rozbudowujących analizę lub zapewniających dodatkowe przetwarzanie tekstu. Nie są one jednak stosowane na tyle często, by wejść w skład zestawu tagów rdzennych; nie mogą też stać się częścią zestawów bazowych, gdyż występują w wielu kategoriach dokumentów. Elementy poszczególnych zestawów umożliwiają zapis:

- hiperłączy, informacji o segmentacji oraz związkach pomiędzy częściami tekstu,
- prostych analiz i interpretacji tekstu,
- struktur cech (opisanych szczegółowo w rozdziale 2.3),
- stopnia jednoznaczności lub niejednoznaczności tekstu,
- manuskryptów,
- krytycznej analizy tekstu,
- określeń czasu i nazw własnych,

- grafów, sieci i drzew,
- tabel, formuł i powiązań z elementami graficznymi,
- informacji klasyfikacyjnych szczególnie pożytecznych w kodowaniu korpusów tekstów.

Dodatkowe zestawy tagów mogą być wybierane i łączone w dowolny sposób, co pozwala na opis wielu rodzajów tekstów o dużym stopniu różnorodności i złożoności.

1.2.3 Wybór zestawów tagów

DTD jest tworzone z wykorzystaniem udostępnianego przez SGML mechanizmu sterowania analizą dokumentu za pomocą całostek parametrycznych i sekcji oznaczanych, opisanego w części 1.1.5. Użytkownik włącza definicję wybranego zestawu tagów do dokumentu przypisując określonej całostce wartość `INCLUDE`.

Oto szkielet przykładowego dokumentu zgodnego z TEI (ang. TEI-conformant), do którego DTD włączono bazowy zestaw tagów dla prozy oraz dodatkowy zestaw tagów do opisu dat i nazw własnych:

```
<!DOCTYPE tei.2
[ <!ENTITY % TEI.prose       'INCLUDE'>
  <!ENTITY % TEI.names.dates 'INCLUDE'> ]>

<tei.2>
  <!-- Treść dokumentu -->
</tei.2>
```

W wersji dokumentu nie opatrzonej dodatkowymi instrukcjami wykorzystywane są całostki parametryczne z wartością `IGNORE`, definiowane automatycznie odpowiednim zapisem w pliku głównym DTD (*tei2.dtd*)[11], co powoduje niejawne wykluczanie niepożądanych zestawów tagów, często stosowane w plikach opisujących konstrukcje DTD TEI. Oto fragment tego pliku z domyślną definicją wartości jednej z całostek:

[11] Zgodnie z wymogami SGML, w razie wystąpienia wielu definicji elementu lub całostki wiążąca jest tylko pierwsza z nich, zatem deklaracja całostki z wartością `INCLUDE` na początku przetwarzania dokumentu ma pierwszeństwo przed definicją standardową.

```
<!ENTITY % TEI.prose 'IGNORE'>

<![ %TEI.prose
    [<!ENTITY % TEI.prose.dtd SYSTEM "teipros2.dtd">
     %TEI.prose.dtd;
    ]
]>
```

1.2.4 TEI Lite i CES — schematy kodowania oparte na TEI

Często stosowanym uzupełnieniem TEI stały się oparte na zaleceniach TEI P3 schematy kodowania TEI Lite i CES, powstałe jako odpowiedź na potrzeby użytkowników oryginalnego zestawu założeń, domagających się jego pewnej specjalizacji za cenę rezygnacji z części rzadziej wykorzystywanych funkcji. W dalszej części pracy opieram się na pełnym zestawie tagów TEI z racji zbyt dużych ograniczeń wprowadzanych przez oba schematy, w wielu przypadkach mogą być one jednak z powodzeniem stosowane bez utraty funkcjonalności TEI.

Schemat TEI Lite znany pod kodową nazwą TEI U5 utworzono jako wybór elementów TEI P3 znajdujący zastosowanie wszędzie tam, gdzie pełny zestaw znaczników i wyrafinowane konstrukcje zapisu okazują się mało przydatne — przy kodowaniu tekstów o prostej strukturze lub zadaniach nie wymagających wysokiej dokładności opisu. Założenia projektu obejmują możliwość wykorzystania niemal wszystkich tagów rdzennych TEI. Określono, że wprowadzone zmiany nie mogą znacząco ograniczać różnorodności reprezentowalnych tekstów, a schemat kodowania powinien dać się wykorzystać przy użyciu istniejących narzędzi SGMLowych i umożliwiać stosowanie opisanych w Wytycznych TEI rozszerzeń. Tak określone cele stosują się jednak także do pełnego TEI, główna modyfikacja sprowadza się więc do ograniczenia zestawu dostępnych znaczników (około 130 rodzajów[12], czyli 1/4 zawartości TEI P3) poprzez uproszczenie zapisu informacji bibliograficznej, eliminację zaawansowanych konstrukcji w rodzaju struktur cech oraz przede wszystkim zawężenie możliwości opisu tekstu do podstawowych struktur (segmenty, akapity, cytaty, wyróżniki często używanych pól specjalnych — dat, oznaczeń skrótów itp.).

Schemat CES (ang. Corpus Encoding Standard — Standard Kodowania Korpusów) to z kolei jeden z wyników projektu EAGLES (Expert Advisory

[12] Jak podaje wprowadzenie do TEI Lite [7] autorstwa twórców TEI.

Group on Language Engineering Standards) — aplikacji TEI określającej minimalny zbiór założeń, jakie musi spełniać strukturalna i typograficzna reprezentacja korpusu tekstów, wyposażonej w dodatkowe mechanizmy zapisu informacji lingwistycznej. W przeciwieństwie do TEI Lite, CES stanowi znaczne rozszerzenie możliwości TEI poprzez redefinicję znaczników i poszerzenie listy ich atrybutów w celu przesunięcia akcentu na dokładność zapisu informacji bibliograficznej i lingwistycznej. CES wprowadza trzy poziomy reprezentacji tekstu:

- poziom 1, wprowadzający ogólną strukturę dokumentu, z podziałem tekstu na segmenty do poziomu akapitu,
- poziom 2, wymagający dokładnego zapisu informacji wewnątrz akapitów, na poziomie międzyzdaniowym,
- poziom 3, z pełnym kodowaniem zdań[13].

[13]Dokładny opis założeń schematu CES można znaleźć w dokumentach projektu EAGLES [1].

Rozdział 2

Mechanizmy zapisu danych lingwistycznych zgodne ze specyfikacją TEI

Mimo swego ściśle lingwistyczno-humanistycznego rodowodu zestaw tagów TEI jako ogólna propozycja spójnego schematu adiustacyjnego znajduje zastosowanie także w wielu innych dziedzinach kodowania tekstów. Udostępniane środki wyrazu analizy tekstu (rozumianej jako pewna syntaktyczna bądź semantyczna interpretacja) są więc odpowiednio ogólne — przykładem może być opisany niżej mechanizm zapisu segmentacji na dowolnym poziomie znaczeniowym. Kwestia zapisu informacji lingwistycznej potraktowana została jednak przez autorów DTD TEI z wyjątkową uwagą. Świadczy o tym obecność specjalnego zestawu tagów `TEI.analysis`, wprowadzającego znaczniki umożliwiające zapis prostych analiz.

Znacznie bardziej wyrafinowanym sposobem przedstawienia charakterystyki tekstu jest użycie mechanizmu *struktur cech*, umożliwiającego powiązanie samodzielnie definiowanych identyfikatorów z wartościami lub zbiorami wartości. Posiada on ogromną siłę wyrazu, umożliwia bowiem tworzenie struktur danych ogólnego zastosowania o dowolnej budowie. Z racji elastyczności większej niż w przypadku pozostałych mechanizmów, został on opisany dokładniej.

Poniższe przedstawienie metod zapisu dodatkowej informacji lingwistycznej może służyć za przegląd udostępnianych przez format TEI mechanizmów tworzenia własnoręcznie zaprojektowanych struktur tekstu, rozbijających go na określonych, wybranych poziomach (składniowych, morfologicznych, ale także innych — dowolnego typu).

2.1 Ogólny mechanizm segmentacji tekstu

Najprostszym sposobem reprezentacji struktury tekstu jest wykorzystanie ogólnego mechanizmu segmentacji, przez *Wytyczne TEI* opisywanego jako naturalne rozszerzenie sposobu oznaczania tych cech tekstu, dla których nie zdefiniowano odrębnych schematów adiustacji.

Zbiór znaczników wprowadzających podział na segmenty umieszczono w dodatkowym zestawie tagów `TEI.linking`, zawierającym ponadto proste mechanizmy tworzenia odwołań do części składowych dokumentów i określania odpowiedniości poszczególnych fragmentów tekstu.

Jednostkę tekstu wyróżnia znacznik `<seg>`, dla którego dostępne są atrybuty:

- `type` — szeroko rozumiany opis rodzaju segmentu,
- `function` — charakteryzacja jego funkcji,
- `subtype` — dodatkowe pole kategoryzacji,
- `part` — informacja o kompletności segmentu, dla której zdefiniowano dozwolony zestaw wartości:
  - `Y` — segment niekompletny,
  - `N` — segment kompletny lub nie określono informacji o jego kompletności,
  - `I` — początkowa część niekompletnego segmentu,
  - `M` — środkowa część niekompletnego segmentu,
  - `F` — końcowa część niekompletnego segmentu.

Znacznik segmentacji może być zagnieżdżany, stosownie do liczby poziomów interpretacji segmentu, co ilustruje poniższy przykład. Pokazano też sposób wykorzystania dodatkowego atrybutu `subtype` do przedstawienia dwóch poziomów analizy:

```
<seg type=zdanie subtype=wspZłoż>
   <seg type=zdanie subtype=pojedyncze>
      <seg type=słowo subtype=czasownik>Wysłałem
      <seg type=słowo subtype=rzeczownik>list
   <seg type=znak subtype=znakInterp>,
   <seg type=słowo subtype=spójnik>ale
```

```
<seg type=zdanie subtype=pojedyncze>
   <seg type=słowo subtype=partykuła>nie
   <seg type=słowo subtype=czasownik>doszedł
<seg type=znakInterpunkcyjny>.
```

Zaletą tego mechanizmu jest możliwość zakodowania informacji o jednostce tekstu umieszczonej na dowolnym poziomie jego reprezentacji — przedmiotem analizy nie muszą być tylko poszczególne zdania, słowa czy znaki, ale też np. części zdań czy akapitów. Jest to więc naturalny sposób zapisu danych pochodzących z analizy składniowej wypowiedzeń czy będących wynikiem wyodrębnienia z tekstu fragmentów przekraczających granice zdań.

Jednolitość zapisu informacji o segmentacji oraz możliwość dowolnej definicji typów i podtypów segmentów to sposób nadzwyczaj prosty, pozbawiony możliwości tworzenia sformalizowanych opisów używanych struktur oraz zbiorów odwołań. Jest on jednak na tyle czytelny, że do prostych zastosowań może przydać się bardziej niż znajomość pozostałych opisywanych metod.

2.2 Podział na jednostki składniowe

Drugi sposób zapisu tekstu wykorzystuje dla celów analitycznych specjalny zestaw tagów, w skład którego wchodzą m. in. oznaczenia jednostek składniowych. Za jego pomocą można wprowadzić podział tekstu na zdania, frazy, morfemy, słowa, nawet znaki, a dla każdego podziału określić jego rodzaj i funkcję. Oto wybór dostępnych znaczników:

- `<s>` — zdanie,
- `<cl>` — zdanie składowe,
- `<phr>` — fraza gramatyczna,
- `<w>` — wyraz; dostępny atrybut `lemma`, przeznaczony do zapisu formy podstawowej,
- `<m>` — morfem; dostępny atrybut `baseform` do zapisu formy podstawowej morfemu,
- `<c>` — pojedynczy znak.

Ponadto, jak w przypadku segmentacji ogólnej, dla znaczników wszystkich typów dostępne są atrybuty `type` i `function`, o opisanym wyżej znaczeniu.

Użycie tego sposobu zapisu sprawia, że zyskuje on na przejrzystości, o czym można się przekonać kodując przykład z poprzedniego podrozdziału (zauważmy, jak zmieniła się zawartość atrybutu `type` — zapisana w nim informacja została przeniesiona na poziom znaczników, co pozwoliło na uproszczenie zapisu kategoryzacji):

```
<s type=wspZłoż>
   <s type=pojedyncze>
      <w type=czasownik>Wysłałem
      <w type=rzeczownik>list
   <c type=znakInterp>,
   <w type=spójnik>ale
   <s type=pojedyncze>
      <w type=partykuła>nie
      <w type=czasownik>doszedł
   <c type=znakInterp>.
```

Warto pamiętać, że chociaż drugi mechanizm jest rozszerzeniem pierwszego, mogą być one stosowane łącznie do zapisu dwóch rodzajów analizy — obejmującej swym zakresem poszczególne jednostki składniowe (znaczniki `<s>`, `<phr>`, `<w>` i `<c>`) oraz szerzej rozumiane segmenty tekstu (`<seg>`):

```
<seg type="opowieść Szwejka">
  <s>Od maleńkości mam takiego pecha.</s> <s>Zawsze chcę
  coś naprawić, zrobić dobrze i zawsze stanie się z tego
  jakaś nieprzyjemność dla mnie i dla otoczenia.</s>
  ...
</seg>
```

2.3 Struktury cech

Formalizm *struktur cech* (ang. feature structures) jest najogólniejszym mechanizmem zapisu informacji udostępnianym przez TEI. Mówiąc najprościej, polega on na wprowadzeniu metody zapisu nazwanych wartości (*cech*) i ich zbiorów (*struktur cech*). Oba terminy wywodzą się w prostej linii z lingwistyki i zostały wprowadzone już w 1949 roku przez Romana Jakobsona jako sposób reprezentacji własności języka mówionego (inwentaryzacja fonemów

wg cech dystynktywnych). Siła wyrazu tak prostego zapisu szybko znalazła zastosowanie w wielu innych gałęziach nauki (w dziedzinie teorii informacji funkcjonuje on jako dobrze znany sposób organizacji danych oparty o pojęcia pola i rekordu).

TEI udostępnia struktury cech w celu umożliwienia opatrzenia tekstu analizą o dowolnie ustalonym rodzaju i stopniu dokładności. Tekst rozdziału opisuje główne elementy teorii struktur cech i przedstawia najważniejsze techniki stosowane do ich zapisu[1].

2.3.1 Oznaczenia cech i struktur cech

Pojęcie cechy jest na tyle ogólne, że do jego zapisu wprowadza się tylko dwa elementy SGMLowe[2]:

- `<f>`, oznaczający cechę (od ang. feature),
- `<fs>`, grupujący cechy w struktury (od ang. feature structures).

Struktura jest reprezentacją pojedynczej analizy jednostki tekstu, na którą składają się wartości jej poszczególnych poziomów. Jej zapis, pomijając na wstępie szczegóły implementacyjne, składa się z elementu `<fs>`, wypełnionego dowolną liczbą elementów `<f>`. Ze względu na wielość możliwych rodzajów analizy wprowadza się atrybut `type` elementu `<fs>`, identyfikujący typ analizy i odróżniający struktury od siebie[3]. Oto szkielet struktury zapisany z użyciem zdefiniowanych elementów:

```
<fs type="struktura fonologiczna">
  <f ...> ... reprezentacja cechy ... </f>
  ...
  <f ...> ... reprezentacja innej cechy ... </f>
</fs>
```

[1] Pełny opis formalizmu struktur cech zawierają *Wytyczne TEI* [3] oraz artykuł *TEI Recommendations for FS Markup* [17], omawiający ponadto wcześniejsze propozycje zapisu struktur i dyskutujący przyczyny wyboru jego obecnej formy.

[2] Przedstawione w dalszej części opisu pojęcia *biblioteki* i *deklaracji struktur cech* są w pewnym sensie dodatkowe; ułatwiają one zarządzanie strukturami i weryfikację ich poprawności, lecz pozostają poza obrębem głównej części definicji.

[3] Jak się okaże, ścisłe rozróżnienie struktur wymaga pewnych dodatkowych zabiegów — SGML nie posiada właściwości umożliwiających wykluczenie błędów logicznych wynikających z użycia cechy z jednej struktury w innej strukturze. Może to zapewnić dodatkowe narzędzie wykorzystujące formalny opis zawartości struktur — omawiane niżej *deklaracje struktur cech*.

2.3.2 Zapis wartości cech

Reprezentacja poszczególnych cech jest równie prosta. Nazwy cech zapisywane są w atrybutach oznaczających je elementów, natomiast wartości wypełniają zawartość elementu oznaczającego cechę. Taki sposób zapisu podyktowany jest koniecznością zapewnienia możliwości elastycznej definicji wartości cechy (nie musi być nią wartość atomowa) oraz wykluczenia wystąpień tzw. *zawartości mieszanej* (ang. mixed content) — umożliwienia wypełnienia wartości cechy strukturą lub czystym tekstem na tym samym poziomie[4].

Zawartość elementu `<f>` może stanowić:

- element `<plus>` lub `<minus>`, określony globalnie jako pusty (ang. empty tag) i oznaczający wystąpienie lub brak cechy binarnej,
- wartość atomowa określonego typu:
  - symbol (`<sym>`) definiujący cechę w postaci wartości ze skończonego zbioru,
  - napis dowolnej postaci oznaczony elementem `<str>`,
  - wartość liczbowa, oznaczona jako `<nbr>`,
- jeden z tagów pustych oznaczających wartość niedookreśloną lub nieznaną[5]:
  - `<any>` — wartość dowolna,
  - `<dft>`[6] — wartość standardowa dla danej cechy[7],

[4] Występowanie zawartości mieszanej, łączącej znaczniki i tekst, jakkolwiek dozwolone przez standard definiujący SGML nie jest jednak zalecane, gdyż może sprawiać trudności w przetwarzaniu zawierających ją dokumentów. Powstaje wówczas również problem interpretacji danych, np. znaków odstępu (spacje, znaki tabulacji, znak nowej linii), pomijanych przy intepretacji tagów i uwzględnianych podczas analizy nieotagowanego tekstu.

[5] Warto zauważyć, że w podanej propozycji nie występuje żaden znacznik odpowiadający nie określonej wartości kategorii (pojęcie wartości nieznanej nie obejmuje tego przypadku). Być może jest to niedopatrzenie, lecz nawet wówczas identyfikacja wartości nie określonej jest możliwa poprzez odpowiedni zapis w pliku deklaracji systemu cech (FSD). Szerzej o sposobie tego zapisu informuje rozdział 2.4.3.

[6] Uwaga: znacznik `<dft>` jest często mylony ze znacznikiem `<default>`, używanym do oznaczenia domyślnej wartości atrybutu w deklaracji dodatkowych zestawów tagów. Podobieństwo nazw elementów jest zresztą nieuniknione w tak rozbudowanym zbiorze tagów, jakim jest standard TEI — zawiera on definicje przeszło 500 znaczników i 1800 całostek.

[7] Wartości standardowe definiowane są w *Deklaracji systemu cech*, opisanej szczegółowo w rozdziale 2.4.3.

  - `<unknown>` — wartość nie znana twórcy dokumentu,

- zagnieżdżona struktura cech (element `<fs>`),
- lista lub zbiór wartości (elementy `<list>` i `<set>`).

Oto przykład zapisu struktury cech o dość różnorodnej zawartości:

```
<fs type="opis morfologiczny">
   <f name=forma>
      <str>dzieckiem</str></f>
   <f name=formaPodstawowa>
      <str>dziecko</str></f>
   <f name=kategoria>
      <sym value=rzeczownik></f>
   <f name=przypadek>
      <sym value=narzędnik></f>
   <f name=rodzaj>
      <sym value=nijaki></f>
   <f name=liczba>
      <sym value=pojedyncza></f>
   <f name=nazwaWłasna>
      <minus></f>
   <f name=liczbaSynkretyzmów>
      <nbr>1</nbr></f>
</fs>
```

2.3.3 Reprezentacja niejednoznaczności

Aby zapisać kilka interpretacji analizy (przypadek dość częsty również w opisie morfologicznym form w języku polskim) opisane powyżej metody są niestety niewystarczające. Użycie elementu `<any>` świadczyłoby o dopuszczeniu wystąpienia wszystkich możliwych wartości, zaś zapis w postaci listy lub zbioru oznacza, że wartością analizy jest cała lista lub zbiór wartości, a nie ich alternatywa.

Standard TEI wprowadza zatem nowy znacznik `<vAlt>` (od ang. value alternation), zezwalający na zapis niejednoznaczności wartości. Oto przykład jego wykorzystania do zapisu synkretyzmu form dwóch przypadków:

```
<fs type="opis morfologiczny">
   <f name=forma>
      <str>królu</str></f>
   ...
   <f name=przypadek>
      <vAlt>
         <sym value=miejscownik>
         <sym value=wołacz>
      </vAlt>
   </f>
   ...
</fs>
```

W ten prosty sposób możemy zapisać niejednoznaczność w obrębie jednej kategorii analizy. Do pełnego zapisu danych lingwistycznych jest to jednak wciąż mechanizm niewystarczający — bardzo często występują bowiem w języku synkretyzmy form przekraczające granice kategorii. Do ich zapisu, z racji potencjalnie różnej zawartości, musi zostać wprowadzony jeszcze jeden dodatkowy znacznik. W standardzie TEI jest nim `<fAlt>` (od ang. feature alternation). Oddzielenie zapisów poszczególnych interpretacji następuje z wykorzystaniem znacznika `<fs>` pozbawionego typu:

```
  <fs type="opis morfologiczny">
     <f name=forma>
        <str>myto</str></f>
     <fAlt>
        <fs>
           <f name=hasło>
              <str>myć</str></f>
           <f name=kategoria>
              <sym value=czasownik></f>
           ...
        </fs>
        <fs>
           <f name=hasło>
              <str>myto</str></f>
           <f name=kategoria>
              <sym value=rzeczownik></f>
           <f name=przypadek>
              <vAlt>
              ...
```

```
            </vAlt></f>
          ...
        </fs>
    </fAlt>
    ...
</fs>
```

2.3.4 Biblioteki cech i struktur cech

Aby ułatwić zarządzanie dużymi strukturami cech, zwalniając dokument z konieczności przechowywania pełnych opisów nazw kategorii i dokładnych definicji rodzajów wartości, standard TEI wprowadza pojęcia *biblioteki cech* (ang. feature library), *biblioteki struktur cech* (ang. feature-structure library) oraz *biblioteki wartości cech* (ang. feature-value library). Każda z wymienionych struktur umożliwia skrócenie zapisu cech do ich identyfikatorów. Biblioteki oznaczane są odpowiednio tagami `<fLib>`, `<fsLib>` oraz `<fvLib>` i posiadają atrybut `type` określający rodzaj przechowywanej w nich informacji.

Poniżej przedstawiam dwa przykłady porównujące zapis bezpośredni (w postaci ciągu struktur cech) z zapisem wykorzystującym biblioteki. Ich zastosowanie jest oczywiście przerysowane, warto jednak zwrócić uwagę na sposób rozdzielania informacji i jej wiązanie za pomocą identyfikatorów[8].

Pierwszy wariant zapisu opiera się o bezpośrednie wyliczenie wartości cech w postaci nazwanego rekordu:

```
<fs id=rzeczMianMęskiPoj
    type="opis morfologiczny rzecz. rodz. męs. w mian. l. p.">
   <f name=kategoria>
      <sym value=rzeczownik></f>
   <f name=przypadek>
      <sym value=mianownik></f>
   <f name=rodzaj>
      <sym value=męski></f>
   <f name=liczba>
      <sym value=pojedyncza></f>
</fs>
```

[8] W tak prostym przykładzie pożytek z użycia bibliotek jest raczej znikomy, jednak w przypadku mnogości cech przy niewielkim zestawie używających ich struktur można w ten sposób osiągnąć znaczne korzyści.

```
...
<fs>
   ...
</fs>
```

Wariant drugi wprowadza zbiory wartości — biblioteki, będące podstawą tworzonego niżej opisu struktury. Oprócz cech wymienionych wyżej pełnią one funkcję dokumentacyjną, ułatwiając rozpoznanie rodzaju zastosowanego schematu zapisu.

W poniższym przykładzie rekordy biblioteki wartości (`<fvLib>`) umożliwiają zewnętrzną obsługę budowy struktur cech, zaś użycie bibliotek cech (`<fLib>`) ułatwia tworzenie prostych odwołań do często złożonych struktur (`<fsLib>`):

```
<fvLib type=przypadki>
   <sym value=mianownik  id=mian>
   <sym value=dopełniacz id=dop>
   ...
</fvLib>
...

<fLib type="opisy kategorii morfologicznych">
   <f id=katRzecz name=kategoria fVal=rzecz></f>
   ...
   <f id=przypMian name=przypadek fVal=mian></f>
   <f id=przypDop  name=przypadek fVal=dop></f>
   ...
</fLib>

<fsLib id=morfDef type="używane opisy morfologiczne">
   <fs id=rzeczMianMęskiPoj
       feats='katRzecz przypMian rodzMęski liczPoj'></fs>
   ...
</fsLib>
```

2.3.5 Bezpośrednie łączenie struktur cech z tekstem

Sens istnienia struktur cech z opisem morfologicznym w oderwaniu od tekstu ogranicza się do przechowania informacji o odmianie wyrazów w

wielkiej, mało uporządkowanej bazie danych. Z drugiej jednak strony, połączenie struktury o zaproponowanej wyżej budowie poprzez wstawienie jej w sąsiedztwie tekstowego wystąpienia odpowiadającej jej formy stwarza poważny problem, chociażby dla czytelności tekstu. Jego rozwiązaniem jest przypisanie strukturom unikatowych identyfikatorów, z którymi powiązane zostaną poszczególne egzemplarze form. Wszystkie struktury zostaną natomiast zgromadzone w bibliotece struktur, w naszym przypadku — repozytorium opisów morfologicznych.

Powyższy zapis posiada jeszcze jedną ważną zaletę: umożliwia powiązanie wielu tekstowych wystąpień form z pojedynczą strukturą. Służy do tego:

- atrybut `inst` elementu `<fs>`, którego zawartość stanowi listę instancji formy o kształcie powiązanym z daną strukturą,
- atrybut `ana` elementu reprezentującego jednostkę tekstu, pełniącego funkcję wskaźnika do struktury i zapewniającego połączenie obu informacji.

Oto przykład powiązania struktury cech z dwoma wystąpieniami formy (realizowanego poprzez zapis jednoznacznie określonych identyfikatorów):

```
<fs type="opis morfologiczny" id="kowalaSSGM---P"
    inst="pr120731 pu004101">
...
</fs>
```

i odpowiadający mu zapis wiążący pojedyncze wystąpienie formy z więcej niż jednym opisem struktury:

```
<w id=pr120731 ana="kowalaSSGM---P kowalaSSTM---P">kowala</w>
```

2.3.6 Hipertekstowe łączenie struktur cech z tekstem

Jeszcze innym sposobem tworzenia powiązań między elementami tekstowymi a odpowiadającymi im strukturami cech, dającym największe korzyści, jest możliwość zewnętrznego definiowania łączy. Zebrane w specjalnej strukturze, przechowują one ogół relacji ustanowionych między składnikami tekstu, ograniczając przechowywaną w nim informację do identyfikatorów jednostek.

Do zapisu zbioru relacji służą dodatkowe elementy `<link>`, reprezentujące łącza, `<linkGrp>`, grupujące łącza oraz mechanizm określania domen łączy (ang. domain).

Wszystkie łącza przechowują informację o powiązaniach typu jeden-do-jednego, zaś sieci połączeń mogą powstawać poprzez określanie wartości już wcześniej użytych. Identyfikatory składowych części łączy, oddzielone spacją, podawane są w atrybucie `targets`:

```
<link targets="pr120731 kowalaSSGM---P">
```

2.4 Deklaracja systemu cech

Wielką zaletą jawnego wyspecyfikowania definicji typu dokumentu (DTD) jest spełniana przez nią funkcja dokumentacji struktury adiustacyjnej stosowanej w tekście. DTD może także posłużyć za kryterium formalnej poprawności dokumentu, mogące zostać wykorzystane w procesie jego automatycznej walidacji.

Wprowadzenie struktur cech narusza postulat ścisłej definicji — słuszne ograniczenie wpływu użytkownika na strukturę dokumentu m. in. poprzez nakaz umieszczania definiowanych wartości w atrybutach elementów sprawia, że jesteśmy w stanie zapewnić jedynie syntaktyczną — nie semantyczną — poprawność konstrukcji zbudowanych przy użyciu elementów `<fs>` i `<f>`. Powodem takiej sytuacji jest niemożność wymuszenia technikami SGMLowymi określonych wartości atrybutów znaczników.

Aby jednak udostępnić formalny zapis warunków nakładanych na struktury cech, autorzy koncepcji postanowili wprowadzić dodatkowy dokument nazwany *deklaracją systemu cech* (Feature System Declaration, FSD).

2.4.1 Założenia podstawowe

Deklaracja systemu cech umożliwia wyeliminowanie niezgodności zapisu wartości cech, a zatem np. sytuacji typu:

```
<f name=przypadek>
   <sym value=nijaki></f>
```

Aby to zapewnić, w jej skład wchodzą następujące elementy:

- lista nazw cech wraz z krótkim opisem ich funkcji,
- deklaracja wartości dozwolonych dla poszczególnych cech,
- lista typów struktur cech wraz z ich opisem,
- deklaracja przynależności cech do określonych struktur,
- wykaz ograniczeń nakładanych na struktury cech udostępniających lub zabraniających współwystępowania wielu wartości cech,
- deklaracja wartości domyślnych.

Celem stworzenia FSD jest umożliwienie oprogramowaniu przetwarzającemu dokument użycia zawartej w niej informacji do weryfikacji poprawności zapisu struktur cech. Możliwość określenia wartości standardowych umożliwia ponadto znaczne skrócenie zapisu struktur, z pominięciem cech o wartościach określonych globalnie[9].

Deklaracje struktur cech tworzące FSD umieszczane są w osobnym dokumencie, dołączanym do tekstu TEI.

2.4.2 Postać dokumentu z deklaracją

Deklaracja systemu cech zapisywana jest jako dokument typu `<teiFsd2>`. Składa się on z nagłówka TEI — elementu `<teiHeader>`, identycznego z nagłówkiem dokumentu `<tei.2>` (jego struktura opisana jest dokładnie w Dodatku B) oraz zbioru deklaracji struktur cech wraz z nakładanymi na nie dodatkowymi warunkami. Oto struktura pliku zawierającego deklarację:

```
<!DOCTYPE teiFsd2 SYSTEM 'teifsd2.dtd'>

<teiFsd2>
   <teiHeader>
      ...
   </teiHeader>
```

[9]Dodatkową, rzadko zauważaną cechą deklaracji jest możliwość wykorzystania jawnej specyfikacji wartości cech do rozwinięcia znacznika `<any>`, używanego do oznaczenia wartości dowolnej. Gdy dostępna staje się informacja o zbiorze wartości dopuszczalnych, pojęcie to przestaje być abstrakcyjne i może zostać wyrażone jako alternatywa podanych oznaczeń.

```
    <fsDecl type="opis morfologiczny">
       <fsDescr>Reprezentacja opisów morfologicznych</fsDescr>

       <fDecl name=kategoria>
          ...
       </fDecl>

       <fDecl name=przypadek>
          ...
       </fDecl>

       ...
    </fsDecl>

    ...
</teiFsd2>
```

Każda deklaracja struktury cech (może być ich wiele) składa się z[10]:

- opcjonalnego opisu znaczenia struktury (element `<fsDescr>`), podanego ciągłym tekstem,
- zbioru deklaracji cech (elementy `<fDecl>`), opisujących zakres i wartości domyślne poszczególnych cech.

2.4.3 Deklaracje poszczególnych cech

Cechy deklarowane są za pomocą elementu `<fDecl>`, w którego atrybucie `name` podawana jest nazwa identyfikująca cechę, zaś w atrybucie `org` — rodzaj wartości (atomowa, zbiór, lista).

Zawartością deklaracji są następujące elementy[11]:

[10] Trzecim elementem deklaracji jest nie używany w pracy opcjonalny zapis ograniczeń nakładanych na wartości cech (`<fsConstraints>`). Jest to wykaz warunków uzależniających występowanie poszczególnych cech w obrębie jednej struktury od jej innych cech. Szczegółowy opis mechanizmu definiowania ograniczeń zawiera rozdział *Feature System Declaration Wytycznych TEI*.

[11] Należałoby jeszcze wspomnieć o możliwości warunkowego definiowania wartości domyślnych za pomocą elementów `<if>` i `<then>` — są one nadawane na podstawie analizy wartości innych cech. Więcej informacji o tym mechaniźmie zawiera rozdział *Feature System Declaration Wytycznych TEI* [3].

- <fDescr>, jak w przypadku deklaracji struktur zawierający opis znaczenia deklarowanej cechy i jej wartości (ciągłym tekstem),
- <vRange>, definiujący zakres dozwolonych wartości,
- <vDefault>, określający wartość domyślną, nadawaną w razie braku cechy o podanej nazwie.

Dozwolone i domyślne wartości cech definiowane są w postaci identycznej z ich wystąpieniami w tekście (patrz rozdział 2.3.2). Dodatkowo możliwe jest użycie specjalnego znacznika <none> do oznaczenia wartości nie określonej. Oto przykład zapisu kategorii przypadka, wykorzystujący wartość <none> dla oznaczenia nieokreśloności przypadka dla części mowy nie definiujących tej kategorii:

```
<fDecl name=przypadek>
   <fDescr>Wartość przypadka gramatycznego</fDescr>

   <vRange>
      <vAlt>
         <sym value=mianownik>
         <sym value=dopełniacz>
         <sym value=celownik>
         <sym value=biernik>
         <sym value=narzędnik>
         <sym value=miejscownik>
         <sym value=wołacz>
      </vAlt>
   </vRange>

   <vDefault>
      <none>
   </vDefault>
</fDecl>
```

2.4.4 Połączenie tekstu TEI z deklaracją systemu cech

Deklaracja systemu cech, będąca dokumentem odrębnym od wykorzystującego ją tekstu otagowanego znacznikami TEI, musi z nim jednak zostać formalnie połączona — choćby po to, by użytkownik tekstu mógł w prosty sposób odnaleźć potrzebne definicje struktur cech.

Dla każdej deklaracji tworzona jest całostka zewnętrzna, której nazwa zostaje powiązana z nazwą pliku zawierającego treść deklaracji. Pliki te, będące poprawnie zbudowanymi dokumentami SGMLowymi, należą do kategorii dokumentów podrzędnych (ang. subdocument), których użycie jest domyślnie wyłączone, zostaje jednak udostępnione przez deklarację SGMLową TEI. Drugim miejscem zapisu informacji o powiązaniu jest nagłówek TEI, a dokładnie jego składnik `<encodingDesc>` (patrz Dodatek B). Dla każdej zdefiniowanej struktury cech jest w nim umieszczany element `<fsdDecl>`, posiadający atrybuty:

- `type`, określający typ struktury cech zdefiniowanej w deklaracji (wartość atrybutu `type` odpowiedniej struktury),
- `fsd`, wykorzystujący wartość całostki zewnętrznej zawierającej treść deklaracji.

Dokument TEI może zawierać wiele deklaracji systemu cech. Każda jest wówczas identyfikowana osobną całostką przechowującą jej treść oraz odrębnym zapisem w nagłówku.

Oto szkielet pełnego dokumentu TEI z wyróżnionymi składnikami łączącymi deklaracje systemu cech z dokumentem:

```
<!DOCTYPE tei.2 SYSTEM 'tei2.dtd' [
   <!ENTITY fsd.morf SYSTEM 'morf.fsd' SUBDOC>
]>

<tei.2>
   <teiHeader>
      ...
      <encodingDesc>
         ...
         <fsdDecl type="opis morfologiczny" fsd=fsd.morf>
      </encodingDesc>
      ...
   </teiHeader>
   ...
</tei.2>
```

Rozdział 3

Projekt struktury dokumentu

W tym rozdziale postaram się przedstawić projekt zapisu dokumentów zawierających informację lingwistyczną zgodny z omówionymi standardami. Dość obszerna dyskusja reprezentacji połączeń opisów struktur z tekstem pod względem ich czytelności ma na celu ukazanie możliwości istnienia wariantów zapisu, którego optymalizacja winna uwzględniać dalsze wykorzystanie tekstu.

Wybór rozwiązania wyprzedza nieco treść następnego rozdziału, opisującego aplikację schematu kodowania informacji lingwistycznej do rzeczywistego zbioru tekstów. Przedstawiam jednak również alternatywy jej zapisu.

3.1 Wybór mechanizmu zapisu

Metoda zapisu informacji każdego rodzaju powininna spełniać co najmniej poniższe postulaty:

- **prostoty**, by z wynikowego tekstu mógł skorzystać także człowiek, nie tylko oprogramowanie analizujące,
- **przezroczystości**, by otagowanie tekstu rozszerzało, lecz nie zmieniało jego postaci sprzed zapisu analizy,
- **rozszerzalności**, by nowe, nie reprezentowane w chwili obecnej cechy tekstu mogły być w prosty sposób dodane, gdy zajdzie taka potrzeba.

W przypadku informacji lingwistycznej warto dodatkowo zwrócić uwagę na ścisłość i kompletność definicji jej zapisu, zapobiegającej lub przynajmniej ograniczającej niewłaściwe użycie opisów kategorii lingwistycznych oraz

reprezentowalność wariantów i niejednoznaczności w zapisywanych danych — dość często zdarza się, że analizowanym jednostkom możemy przypisać więcej niż jedną interpretację.

Według zamieszczonego w Wytycznych TEI [3] wprowadzenia do kodowania informacji lingwistycznej, powszechna metoda lingwistycznej adiustacji tekstu polega na wprowadzeniu kodów dla poszczególnych własności leksykalnych i opatrzeniu nimi kolejnych jednostek tekstu. Analizator (człowiek lub maszyna), posiadając tabelę zdefiniowanych kodów, może je w prosty sposób odczytać. W przykładowym zdaniu

```
Ogary#SNP poszły#V3PP w#P las#SAS.
```

dość łatwo jesteśmy w stanie identyfikować kody kategorii gramatycznych (uproszczone dla celów ilustracji mechanizmu — w rzeczywistym zastosowaniu, opisanym w następnych rozdziałach, ich zapis będzie nieco bardziej złożony).

Odmianą tego sposobu jest opisany w rozdziale 2.1 ogólny mechanizm segmentacji tekstu, umożliwiający zapis kodu gramatycznego dla poszczególnych jednostek składniowych tekstu:

```
<seg type=zdanie>
   <seg type=słowo subtype=SNP>Ogary</seg></seg>
   <seg type=słowo subtype=V3PP>poszły</seg>
   <seg type=słowo subtype=P>w</seg>
   <seg type=słowo subtype=SAS>las</seg>
   <seg type=znakInterpunkcyjny subtype=kropka>.</seg></seg>
```

Jak można szybko zauważyć, ogromną wadą takiego zapisu jest jego ograniczona rozszerzalność. Wprowadzenie nowej kategorii gramatycznej (albo, co wiązałoby się z jeszcze poważniejszymi modyfikacjami, analizy innego poziomu) wymaga zmian w całej metodzie kodowania. Zwięzły zapis informacji w atrybutach elementów wymusza wstępne scalanie atomowych danych i uniemożliwia tworzenie rozbudowanych struktur[1].

Metoda wykorzystująca naturalny podział tekstu na części składowe (rozdział 2.2) również udostępnia jedynie najprostszy mechanizm zapisu analiz:

[1] Oczywiście, dostępna pozostaje metoda zagnieżdżania segmentów, lecz sztuczne rozbudowywanie struktury nie służy czytelności ani dokładności reprezentacji danych, współistniejących na tym samym poziomie analizy

```
<s>
  <w type=SNP  lemma="ogar">Ogary</w>
  <w type=V3PP lemma="iść">poszły</w>
  <w type=P    lemma="w">w</w>
  <w type=SAS  lemma="las">las</w>
  <c type=PER>.</c>
</s>
```

Na poziomie słowa jest co prawda dostępny atrybut przechowujący formę podstawową hasła, lecz rozszerzalność kodowania jest znów trudna; co więcej, nie da się jej zapewnić nawet w sposób opisany przy analizie segmentów — tworzenie zagnieżdżeń struktury nie jest już możliwe.

Niedogodności te mogą zostać usunięte z użyciem mechanizmu struktur cech. Konieczność łączenia wartości, czynność dalece niepożądana, eliminowana jest poprzez rozdzielenie informacji na podstawowe składniki — cechy, grupowane w większe jednostki — struktury cech. Rozszerzalność zapisu zagwarantowana jest samą budową struktur; dodanie nowej cechy nie przedstawia najmniejszego problemu, nie jest też trudna rozbudowa analizy o kolejne poziomy — wystarczy utworzyć nową strukturę cech. W znacznie elastyczniejszy sposób może być także dobierany zakres analizy, gdyż bez dodatkowych zabiegów reprezentowana jest informacja o charakterze zapisanych wartości (symboliczna, liczbowa, napisowa itd.) oraz grupowalności cech.

Bardzo dobrze reprezentowane są także własności ściśle lingwistyczne, jak opis niejednoznaczności analizy (wynikających tak z synkretyzmu w obrębie jednej części mowy, jak i z homonimii form), opisywanych specjalnym mechanizmem wartości alternatywnych. Utworzenie deklaracji systemu cech wprowadza natomiast ścisłość i kompletność definicji, zapobiegając użyciu niewłaściwej postaci struktur cech i określając wartości domyślne.

Proponowany zapis analiz jest ponadto bardziej zgodny z duchem SGMLa — mimo że dostępnymi środkami nie sposób zapewnić zgodności zapisu struktur z ich deklaracją, potencjalne problemy ograniczane są do minimum poprzez wymuszenie zapisu cech złożonych w treści elementów, a wartości prostych (symbole z ograniczonych zbiorów, wartości liczbowe lub logiczne) w treści ich atrybutów.

3.2 Zapis informacji morfologicznej i składniowej

Wykorzystując prosty przykład z poprzedniej sekcji, przeanalizuję teraz sposób połączenia struktur cech z tekstem TEI. Kolejne przybliżenia rozwiązania mają pokazać, że wybór optymalnego zapisu wymaga analizy wielu czynników — stopnia dokładności interpretacji, zmienności tekstu, rodzaju jego zastosowań (przetwarzanie maszynowe czy dostęp bezpośredni) itd. Przedstawiony projekt w swej końcowej wersji został dostosowany do wymogów zbioru *Słownika frekwencyjnego*, opis przetwarzania którego znajduje się w następnym rozdziale. Warto jednak zdawać sobie sprawę, że zaproponowana postać zapisu może okazać się niewłaściwa dla innych, odmiennych zastosowań — wprowadzenie statycznych, znaczących identyfikatorów np. dla tekstów modyfikowanych względnie często nie jest dobrym pomysłem.

3.2.1 Identyfikatory anonimowe

Pierwszy projekt zapisu polega na utworzeniu dużego słownika analiz w postaci biblioteki struktur cech i powiązaniu jego haseł z wystąpieniami form w tekście:

```
<fsLib>
   ...
   <fs type="opis morfologiczny" id="an311909" inst="pr120731">
      <f name=formaPodstawowa>
         <str>ogar</str></f>
      <f name=kategoria>
         <sym value=rzeczownik></f>
      ...
   </fs>
   ...
</fsLib>

<s>
  <w id=pr120701 ana="an311909">Ogary</w>
  <w id=pr120702 ana="an334503">poszły</w>
  <w id=pr120703 ana="an416425">w</w>
  <w id=pr120704 ana="an120366">las</w>
  <c type=PER>.</c>
</s>
```

Wielką zaletą tego zapisu jest anonimowość identyfikatorów analiz (rozumiana jako brak ich znaczenia w stosunku do opisów zawartych w określanych nimi strukturach), umożliwiająca niemal dowolną zmianę postaci analizy bez wpływania na postać tekstu. Jest to jednak opis zupełnie nieczytelny, co w przypadku tekstów nie podlegających częstym zmianom świadczy na niekorzyść takiego rozwiązania.

3.2.2 Identyfikatory znaczące

Rozwiązaniem mogącym poprawić czytelność tekstu jest wprowadzenie identyfikatorów znaczących, używanych także w przykładach podawanych w *Wytycznych TEI* [3] i tekście o strukturach cech ze zbioru *TEI Background and Context* [17]. Są one tworzone na podstawie informacji przechowywanych w przyporządkowanych im strukturach i oprócz swej funkcji identyfikującej (wykorzystywanej przez analizator elektroniczny) mogą być użyte przez osobę analizującą tekst do szybkiego odczytania danych ze struktur.

Oto przykład wykorzystujący znaczące identyfikatory z zapisem formy podstawowej leksemu oraz skrótu kodów morfologicznych:

```
<fsLib>
   <fs type="opis morfologiczny" id="ogarSPNA---P"
       inst="pr120731">
      <f name=formaPodstawowa>
         <str>ogar</str></f>
      <f name=kategoria>
         <sym value=rzeczownik></f>
      ...
   </fs>
   ...
</fsLib>

<s>
  <w id=pr120701 ana="ogarSPNA---P">Ogary</w>
  <w id=pr120702 ana="iśćVP-R3POP">poszły</w>
  <w id=pr120703 ana="wD------P">w</w>
  <w id=pr120704 ana="lasSSNI---P">las</w>
  <c type=PER>.</c>
</s>
```

Zapis ten zaczyna nieco przypominać bezpośrednie kody typów jednostek tekstu opisywane w rozdziale 2.2, jego znaczenie jest jednak inne — dokładny opis zapewniają jedynie wartości struktur cech, identyfikatory mogą natomiast stanowić skrót ich zapisu.

3.2.3 Ulepszenia zapisu

Oba proponowane opisy polegały na utworzeniu bazy struktur cech, przechowującej informację o wszystkich leksemach użytych w tekście. Jej przydatność w niektórych przypadkach może być jednak niewielka. Lepszym rozwiązaniem może być wówczas rezygnacja z przechowywania w bibliotece struktur informacji o formach leksemów i odpowiadających im formach podstawowych i przeniesienie jej na poziom tekstu. W pewnych zastosowaniach może to przynieść spore korzyści także osobom korzystającym z tekstu, gdyż informacja o formie podstawowej będzie dostępna bezpośrednio. Także odciążenie biblioteki struktur zmieni jej funkcję z bazy danych haseł na zbiór ich interpretacji.

Przeniesienie formy podstawowej na poziom tekstu może zostać zrealizowane z wykorzystaniem atrybutu `lemma` elementu `<w>`. Po usunięciu z opisów struktur pól określających formę leksemu możemy już wprowadzić prosty opis analizy morfologicznej, będący np. odpowiednim kodem pozycyjnym utworzonym wg zasad *Zestawu Kodowego do Reprezentacji Polskiej Informacji Lingwistycznej* K. Głowińskiej i M. Wolińskiego, opisanego w Dodatku A:

```
<fsLib>
   <fs type="opis morfologiczny" id="SSGM---P"
       inst="pr120731 pu004101">
      <f name=kategoria>
         <sym value=rzeczownik></f>
      ...
   ...
   </fs>
</fsLib>

<w id=pr120731 lemma=kowal ana="SSGM---P SSTM---P">kowala</w>
```

Warto zwrócić uwagę, jak po zmniejszeniu objętości biblioteki może zmienić się budowa zapisów struktur cech — atrybut przechowujący instancje może się nadmiernie rozrastać. Listy instancji, z racji symetrii między zawartością atrybutu `ana` elementu opisującego jednostkę tekstu a listą wystąpień

mogą być jednak w razie potrzeby tworzone automatycznie. Jeśli natomiast taki sposób zapisu powiązań okaże się niepożądany, połączenie struktur cech z tekstem może się dokonać z pomocą mechanizmu łączy hipertekstowych, dyskutowanego w rozdziale 2.3.6.

Użycie łączy jest również warte rozważenia, gdyż autorzy koncepcji opisują ją jako bardzo wygodną do przetwarzania automatycznego w środowisku hipertekstowym wskazując na wygodę w edytowaniu powiązań. Uważam jednak, że jej użycie w stosowanym przeze mnie rozwiązaniu nie przyniosłoby większych korzyści, z perspektywy użytkownika-człowieka spowodowałoby natomiast utrudnienia w dostępie do niektórych informacji.

Niewątpliwą zaletą wprowadzenia obsługi łączy jest zmniejszenie ilości danych przechowywanych w samym tekście — jedynym atrybutem form pozostaje wówczas unikatowy identyfikator. Rezygnacja z atrybutu przechowującego analizy oznaczałaby jednak (przy zachowaniu czytelnego zapisu interpretacji w tekście) konieczność ich zapamiętania w identyfikatorze, co wiąże się znów z łączeniem atomowych danych. Proponowany schemat identyfikacji form i odpowiadających im struktur sprawia natomiast, że pozostawienie wskaźników w bezpośrednim sąsiedztwie form i opisów, bez tworzenia dodatkowego zbioru łączy, nie tylko nie zmniejsza, ale wręcz poprawia czytelność tekstu. Użyteczność zbioru łączy zapisanego jako zbiór par samych identyfikatorów jest chyba niewątpliwie mniejsza.

Na korzyść przyjętego rozwiązania przemawia też względna stabilność kodowanego tekstu — w razie konieczności dokonywania zmian w zapisanym tekście lub łączenia istniejących struktur z jego nowymi fragmentami, rozwiązanie wykorzystujące zbiory łączy może okazać się lepsze, gdyż uaktualnienia będzie wymagał jedynie zbiór łączy.

3.2.4 Zapis informacji składniowej

Do zapisu informacji składniowej warto wykorzystać znaczniki jednostek opisane w rozdziale 2.2: `<s>`, `<cl>`, `<phr>`, `<w>`, `<m>` i `<c>`, reprezentujące tradycyjny podział na zdania, klauzule, frazy, słowa, morfemy i znaki. Ich atrybuty umożliwiają przechowanie prostych oznaczeń klasyfikacyjnych, wystarczających do większości zastosowań i nie wymagających szerszych wyjaśnień. Opisane w poprzednim rozdziale połączenie jednostek z analizą morfologiczną form umożliwia stworzenie spójnego zapisu tekstu do poziomu interpretacji zdań; fragmenty szersze muszą być opisywane w inny sposób — np. z wykorzystaniem mechanizmu ogólnej segmentacji z rozdziału 2.1.

3.3 SGML i TEI a sprawa polska

Problem zapisu tekstów w języku polskim w systemach komputerowych posiada długą historię[2]. Ustanawiane normy i wprowadzane nimi kodowe zestawy znaków nie zawsze cieszyły się popularnością wśród użytkowników komputerów i często zdarzało się, że większe znaczenie uzyskiwały kody lokalne (Mazovia, DHN).

W chwili obecnej do zapisu tekstów polskich najpowszechniej stosowane są strony kodowe 852 i 1250, kod ISO 8859-2 i zdobywający coraz większą popularność Unicode (ISO 10646). Ich użycie w tekstach SGMLowych jest jak najbardziej możliwe, ale wiąże się z dokonaniem pewnych dodatkowych zabiegów.

3.3.1 Polskie litery w tekście dokumentu SGMLowego

Jak już wspominałem w części wprowadzającej do tematyki SGMLowej (rozdział 1.1.6), miejscem określenia zestawu znaków używanego w dokumencie jest deklaracja SGMLowa. Wzorcowa składnia konkretna, wykorzystywana w większości przypadków, jako zbiór dozwolonych znaków SGMLowych określa jednak 7-bitowy zestaw ISO 646 IRV — zbyt wąski, by poprawnie kodować polskie znaki w każdym z używanych obecnie systemów.

Symboliczne kodowanie polskich liter (bez wprowadzania znaków o opisach 8-bitowych) jest oczywiście możliwe, jego używanie jest jednak skrajnie niewygodne[3]. Co więcej, jedną z zasad SGMLa jest stosowanie tak naturalnego zapisu, jak to tylko możliwe, co również wyklucza użycie symboli do zastosowań lokalnych.

Korzystanie z polskich znaków zapisanych w wymienionych we wstępie systemach kodowych wymaga zmiany deklaracji SGMLowej w celu włączenia obsługi szerszego zestawu znaków. Modyfikacja obejmuje zmianę parametru `CHARSET` deklaracji, zawierającego wykaz dozwolonych kodów. Najłatwiej zrobić to deklarując dodatkowy zestaw znaków, np. w poniższy sposób (pierwsza część opisu jest kopią wzorcowej składni konkretnej):

[2] Ogólną problematykę zapisu tekstów oraz użycia systemów kodowych porusza artykuł *Kodowanie tekstów polskich w systemach komputerowych* [5]. Warto też przeczytać tekst *Character Representation* [10] ze zbioru *TEI — Background and Context.*

[3] Wyraz *żółć* zapisany w postaci czterech nawet najbardziej znaczących symboli jest tego najlepszym przykładem.

```
<!SGML "ISO 8879-1986"
  CHARSET
  BASESET "ISO 646-1983//
           CHARSET International Reference Version (IRV)//
           ESC 2/5 4/0"
  DESCSET   0      9     UNUSED
            9      2     9
           11      2     UNUSED
           13      1     13
           14     18     UNUSED
           32     95     32
          127      1     UNUSED
  BASESET "ISO Registration Number 101//
           CHARSET ECMA-94 Right Part of Latin Alphabet Nr. 2//
           ESC 2/13 4/2"
  DESCSET 128     32     UNUSED
          160     95     32
          255      1     UNUSED
  ...
>
```

Użyty identyfikator publiczny stanowi odwołanie do definicji zgodnego z ISO Latin-2 zestawu znaków będącego częścią standardu ECMA-94[4].

Rozszerzenie zestawu dostępnych znaków o dodatkowe pozycje (numery kodowe ASCII większe niż 160) włącza możliwość wykorzystania w dokumencie SGMLowym polskich liter zapisywanych w standardzie ISO Latin-2[5]. Sposób ten eliminuje konieczność wprowadzenia kodowania symbolicznego znaków o kodach wyższych niż 128, stanowiąc naturalny pomost między środowiskiem SGMLowym a stosowanymi w praktyce systemami zapisu polskich znaków.

[4]Kompletny wykaz standardów i raportów technicznych ECMA znajduje się na stronie Stowarzyszenia pod adresem `http://www.ecma.ch/publicat.htm`.

[5]W podobny sposób — udostępniając znaki o ściśle określonych kodach — można dokonać rozszerzenia zestawu znaków dokumentu o polskie litery zapisywane w dowolnym systemie kodowym. Dla niektórych z nich (np. Mazovia czy DHN, zapisujące polskie litery jako znaki o kodach niższych niż 160) powyższa definicja okaże się bowiem niewystarczająca.

3.3.2 Polskie litery w opisie struktury dokumentu

Opisana powyżej redefinicja zestawu dostępnych w dokumencie SGMLowym znaków jest warunkiem koniecznym do zapewnienia poprawnej obsługi wystąpień kodów spoza zbioru domyślnego (numery ASCII z przedziału 0-127) w tekście dokumentu, nie wystarcza jednak, gdy zamierzamy użyć polskich znaków także w elementach adiustacyjnych. Jest to problem dość ważny w obliczu jednego z głównych założeń SGMLa — zalecenia stosowania w maksymalnym stopniu dla dokumentów o zasięgu lokalnym schematów adiustacyjnych opartych na języku macierzystym (ang. native-language tagging).

Idea konstrukcji dokumentów SGMLowych, oparta na rozdzieleniu struktury tekstu i tworzących go danych, określa wyraźnie zasady tworzenia identyfikatorów o określonych funkcjach adiustacyjnych. W połączeniu z zapisami składni konkretnej daje to w wyniku m. in. definicje podstawowego zestawu znaków możliwych do wykorzystania w nazwach elementów struktury. Wzorcowa składnia konkretna mocno ogranicza ów zestaw[6], możność użycia polskich liter musi być zatem włączona w rozszerzenie definicji składni konkretnej, dotyczące np. identyfikatorów elementów.

Za strukturę znakową identyfikatorów odpowiada sekcja `NAMING` definicji składni konkretnej (`SYNTAX`). Rozpoczyna się ona deklaracją czterech stałych:

- `LCNMSTRT` (skrót ang. lower case name start characters),
  definiującą dozwolone znaki małych liter rozpoczynające nazwy,
- `UCNMSTRT` (ang. upper case name start characters),
  definiującą znaki dużych liter rozpoczynające nazwy,
- `LCNMCHAR` (ang. lower case name characters),
  definiującą znaki małych liter wchodzące w skład nazw,
- `LCNMCHAR` (ang. upper case name characters),
  definiującą znaki dużych liter tworzące nazwy.

Po nazwie stałej w cudzysłowach następuje zapis znaków definicji rozszerzającej zestaw standardowy (litery alfabetu łacińskiego i cyfry). Oto sekcja `NAMING` składni wzorcowej:

[6]Identyfikatory elementów muszą rozpoczynać się literą a-z (A-Z) alfabetu łacińskiego i składać co najwyżej z liter tego alfabetu, cyfr oraz znaku minusa i kropki.

```
NAMING  LCNMSTRT  ""
        UCNMSTRT  ""
        LCNMCHAR  "-."
        UCNMCHAR  "-."
```

W celu włączenia w skład dozwolonego zestawu kodów odpowiadających polskim znakom należy włączyć je do łańcucha odpowiedniej stałej używając przywołań znaków numerycznych (ang. numerical character references) — całostek specjalnego rodzaju, zapisywanych w postaci kodu liczbowego poprzedzonego sekwencją `&#` i zakończonego znakiem średnika. Oto przykładowa postać sekcji `NAMING` rozszerzonej o definicje znaków z zestawu ISO Latin-2 (użycie innej strony kodowej wymaga zmiany przywołań):

```
NAMING  LCNMSTRT  "&#230;&#179;&#243;&#182;&#188;&#191;"
        UCNMSTRT  "&#198;&#163;&#211;&#166;&#172;&#175;"
        LCNMCHAR  "-.&#177;&#230;&#234;&#179;&#241;&#243;
                     &#182;&#188;&#191;"
        UCNMCHAR  "-.&#161;&#198;&#202;&#163;&#209;&#211;
                     &#166;&#172;&#175;"
```

3.3.3 TEI: Maksymalna przenośność czy zgodność ze standardami lokalnymi?

Jak podają *Wytyczne TEI* [3], elektroniczny zapis znaków tekstu wymaga:

- wybrania zestawu znaków, którego użyjemy do zapisu tekstu,
- przystosowania dokumentu do bezbłędnego transportu danych między różnymi środowiskami komputerowymi,
- wprowadzenia specjalnego kodowania znaków niemożliwych do uzyskania w bezpośredni sposób;
- określenia sposobu oznaczania zmian w kodowaniu znaków i dodatkowych własności zapisu tekstu (użycie różnych alfabetów, zmiana kierunku tekstu).

Pierwsze trzy problemy pojawiają się w przypadku każdego tekstu używającego znaków charakterystycznych dla języka polskiego.

Projekt TEI stwierdza, że żaden zestaw znaków nie jest preferowany w zapisie dokumentów. Możliwe jest zatem użycie dowolnego standardu o dostępności i zrozumiałości w odpowiednim środowisku lokalnym. W przypadku polskim będzie to zapewne jedna ze stron kodowych o ugruntowanej pozycji (Latin-2, CP-1250). Możliwość zapisu polskich znaków w SGMLu uzyskamy już po rozszerzeniu zestawu znaków dokumentu poprzez wprowadzenie opisanych w poprzednim punkcie zmian w deklaracji SGMLowej[7].

Użycie lokalnego zestawu kodowego wydaje się prawie zawsze najlepszym rozwiązaniem — wyjątkiem jest sytuacja częstego przenoszenia tekstu między różnymi środowiskami komputerowymi, dla których nie jesteśmy w stanie zagwarantować poprawności reprezentacji wszystkich znaków używanych w dokumencie. Teksty o szerokim zasięgu wykorzystania, co do których istnieje obawa, że niestandardowe znaki mogłyby zostać niepoprawnie przesłane lub zinterpretowane, należy zakodować z wykorzystaniem całostek SGMLowych[8].

Dla większości symboli i znaków używanych w językach europejskich istnieją standardy nazewnictwa, na podstawie których stworzono zestawy całostek, łatwe do użycia w dokumentach TEI. „Polskie znaki" reprezentowane są w nich przez całostki umieszczone w zestawie o nazwie ISOLat2. Jego użycie ogranicza się do włączenia do dokumentu odpowiedniej deklaracji publicznej:

```
<!ENTITY % ISOLat2
  PUBLIC "ISO 8879-1986//ENTITIES Added Latin 2//EN">
%ISOLat2;
```

Daje nam to dostęp do standardowych oznaczeń polskich liter[9]:

[7] Standard TEI nakłada ponadto na opisany schemat dodatkowy warunek: dokumentacji stosowanego systemu zapisu znaków w postaci tzw. *deklaracji systemu zapisu* (ang. writing system declaration, WSD). Jest to pomocniczy dokument SGMLowy opisujący w sposób formalny znaczenie dodatkowych znaków. Sposób tworzenia dokumentów WSD opisują dokładnie *Wytyczne TEI* [3]; dla języka polskiego tworzenie deklaracji od podstaw okazuje się jednak niepotrzebne, o ile używanym systemem zapisu znaków jest ISO Latin-2, dla którego istnieje odpowiedni dokument WSD o identyfikatorze publicznym `-//TEI P2:1993//NOTATION WSD for ISO 8859-2//EN`.

[8] Użycie całostek (patrz rozdział 1.1.4 objaśniający budowę i użycie całostek) oznacza ograniczenie zestawu używanych w dokumencie znaków do podzbioru ISO 646, zawierającego znaki najmniej podatne na błędną interpretację podczas transmisji (znaki alfanumeryczne alfabetu łacińskiego z rozróżnieniem kasztowości, symbole cyfr oraz podstawowe znaki interpunkcyjne).

[9] Warto zwrócić uwagę na znajomą nazwę oznaczenia dla *ą* i *ę* — przyczyną jest polskie pochodzenie diakrytu.

Nazwa całostki	Znaczenie
aogon	ą
cacute	ć
eogon	ę
lstrok	ł
nacute	ń
sacute	ś
zacute	ź
zdot	ż

Nazwa całostki	Znaczenie
Aogon	Ą
Cacute	Ć
Eogon	Ę
Lstrok	Ł
Nacute	Ń
Sacute	Ś
Zacute	Ź
Zdot	Ż

Rozdział 4

Słownik frekwencyjny polszczyzny współczesnej — zastosowanie przyjętego rozwiązania

W rozdziale tym przedstawię opis wykorzystania zaproponowanego mechanizmu zapisu danych lingwistycznych do zakodowania korpusu *Słownika frekwencyjnego polszczyzny współczesnej* [16], opartego o obszerny elektroniczny zbiór tekstów o dużej różnorodności.

Pierwsza część rozdziału przybliża zawartość korpusu słownika i przedstawia jego krótką historię (w oparciu o artykuł Z. Saloniego [19]). Wynikiem opisanego następnie przetwarzania danych są zbiory słownika w nowym formacie, umieszczone na dołączonej do pracy płycie CD-ROM (opis jej zawartości przedstawia Dodatek C).

4.1 Zbiory danych słownika

Słownik frekwencyjny polszczyzny współczesnej, będący zbiorem informacji o częstościach wyrazów języka polskiego, oparty jest o zbiór 500 000 słów zawartych w pięciu równych transzach, odpowiadających pięciu stylom współczesnej polszczyzny pisanej. Są to: styl popularnonaukowy, wiadomości prasowych, publicystyczny, prozy artystycznej i dramatu artystycznego. Teksty stylów pochodzą z lat 1963-1967 i zostały dobrane metodą losowania.

Aby w maksymalnym stopniu uniezależnić zawartość zbioru od tematyki poszczególnych prac i stylu pisarskiego autorów, w skład każdej transzy weszło 2000 próbek ciągłego tekstu o długości ok. 50 słów (niedokładność

wynika z założenia, by koniec próbki pokrywał się z końcem zdania). Źródła próbek zostały dobrane tak, by uzyskać reprezentację możliwie wielu tekstów.

Prace nad słownikiem (od początku oparte na założeniu, że będą w całości prowadzone metodami komputerowymi) rozpoczęto już w 1967 roku. Po zebraniu materiału zapisanego na 10 000 papierowych fiszek i wprowadzeniu go pod kierunkiem prof. Jerzego Woronczaka z Uniwersytetu Wrocławskiego do komputera Eliott 803 a potem ODRA 1204, dokonano obliczeń częstości i grupowania haseł. Nośnikiem zbiorów stały się najpierw perforowane taśmy papierowe, a następnie taśmy magnetyczne (lata 80-te). Tomy poświęcone wszystkim stylom [15] wydano w postaci powielonych wydruków w latach 1974-77. Od roku 1985 inicjatywę w pracach nad słownikiem przejął Uniwersytet Warszawski. Uaktualnienia nośników danych oraz sporządzenia list rangowych dokonał Krzysztof Szafran. On też, przy współpracy Janusza S. Bienia i Hanny Kołodziejskiej przygotował wydanie słownika w formie drukowanej [16], które ukazało się nakładem Instytutu Języka Polskiego PAN w 1991 roku (w skład słownika weszły hasła o frekwencji łącznej – we wszystkich transzach – nie niższej niż 4[1]). W chwili obecnej zbiory słownika dostępne są w postaci płyty CD-ROM z zapisem pięciu plików poszczególnych stylów wraz z danymi bibliograficznymi źródeł[2].

Po połączeniu informacji bibliograficznej z treścią próbek pojedynczy zapis w pliku ma następującą postać:

```
27 E. Bryll Studium 1963 LSW str. 83
Porucznik był wyraźnie zadowolony[211] ze[62] swego[221]
konceptu. Spojrzał triumfalnie na[64] mnie[44]
i prokuratora[141] Domaniewskiego[/][141], wybuchnął
śmiechem i jeszcze raz[8] zaczął: Jeden[211] zabity[211].
Ja go[42] nie chciałem zabić... W[66] pierwszej[261]
chwili[161] żaden[211] z[62] nas[42] nie zrozumiał nawet
sensu tych[222] słów. Oficer uśmiechnięty[211] jeszcze,
szukający[211] czegoś[42] po[66] kieszeniach munduru,
znieruchomiał, przybrawszy taki[241] wyraz[141] twarzy[121],
jakby zamiast[62] poszukiwanego[221] przedmiotu natknął[501]
się w[66] głębinach kieszeni[121] na[64] szpilkę.
```

Pierwsza linia segmentu określa numer próbki w ramach transzy (przykład pochodzi ze stylu prozy artystycznej), autora i tytuł tekstu źródłowego, rok

[1]Liczba wszystkich haseł listy zbiorczej słownika (po usunięciu powtórzeń) wyniosła prawie 38 500; w wydaniu książkowym znalazło się ich przeszło 10 000.

[2]Dokładny opis zawartości słownika i niedawnych prac przy dopisywaniu informacji bibliograficznej do tekstów próbek znajduje się w pracy Marty Nazarczuk [18].

jego wydania, nazwę (skrót nazwy) wydawnictwa oraz identyfikator położenia fragmentu w pozycji źródłowej. W nawiasach kwadratowych w tekście znajdują się dopisane ręcznie na etapie zbierania próbek cyfrowe kody fleksyjne niektórych form, ujednoznaczniające ich kategorie odmiany w przypadku możliwości wystąpienia synkretyzmów oraz pewne dodatkowe oznaczenia (nazw własnych, skrótowców, form złożonych itp.[3]).

4.2 Przetwarzanie danych i ich postać wynikowa

Jak wynika z przytoczonego przykładu, pełny opis informacji morfologicznej, pożądany w przypadku udostępnienia tekstu do przetwarzania maszynowego, nie był dla słownika dostępny. Jego uzyskanie, wymagające kilku dodatkowych zabiegów, stało się możliwe dzięki wykorzystaniu doskonałego analizatora morfologicznego SAM-99 Krzysztofa Szafrana[4]. Jest to narzędzie oparte na schematycznym indeksie a tergo Jana Tokarskiego [21] udostępniające na podstawie pojedynczych słów tekstowych charakterystykę gramatyczną form wraz z ich postacią podstawową.

Dokonane przeze mnie przetwarzanie treści próbek polegało na uzyskaniu jak najbardziej szczegółowego opisu tworzących je form z uwzględnieniem istniejących kodów, rozszerzeniu zakresu przechowywanej informacji morfologicznej i opatrzeniu nią haseł wcześniej nie oznaczonych. Większą część tej pracy udało się wykonać automatycznie, przy użyciu wspomnianego narzędzia. W przypadku niektórych form (nazwy własne, skrótowce, formy złożone w rodzaju *nie sposób*, *przede wszystkim* i nieliczne inne) analiza morfologiczna okazała się jednak nieskuteczna — formy tego typu zostały opisane ręcznie.

Do zapisu polskiej informacji morfologicznej K. Głowińska i M. Woliński stworzyli ogólny pozycyjny kod uwzględniający jedną z możliwych interpretacji analizy form. Wpływ słownika frekwencyjnego na jego postać ograniczył się do ostatniej pozycji, przechowującej informację niefleksyjną, pochodzącą bezpośrednio z plików słownikowych. Jest to osobne pole do oznaczania nazw własnych i skrótowców, traktowanych zwykle w specjalny sposób[5]. Kod

[3] Pełny opis struktury próbek, postaci i znaczenia kodów pierwotnych oraz procesu ich dodawania zawiera praca Marty Nazarczuk [18].

[4] Opis użytkowy wcześniejszej wersji analizatora jest treścią pracy [20].

[5] Szczegółowy opis zestawu przedstawia tabela w Dodatku A.

ten może zostać łatwo dostosowywany do przechowania większej ilości informacji poprzez rozszerzanie go o nowe oznaczenia (np. kod interpunkcyjny umieszczany na pierwszej pozycji).

Zachowanie podziału tekstu na transze (style) i próbki – jednostki wyższego rzędu – jest najłatwiej osiągane przez zastosowanie mechanizmów grupowania oferowanych przez TEI m. in. dla celów kodowania korpusów.

Wyodrębnienie transz następuje w wyniku użycia zarezerwowanego dla korpusów elementu głównego `<teiCorpus.2>`, którego zawartość stanowią teksty stylów reprezentowane w postaci składowych elementów `<tei.2>`. Korpus opisuje ponadto nagłówek `<teiHeader>`, stanowiący wspólny dla wszystkich transz opis bibliograficzny zawartości zbioru. Oto zarys struktury dokumentu:

```
<teiCorpus.2>
   <teiHeader type=corpus>
      ... informacje bibliograficzne ...
   </teiHeader>

   <tei.2 id=stylA>
      ... styl A ...
   </tei.2>

   <tei.2 id=stylB>
      ... styl B ...
   </tei.2>

   ... pozostałe style ...

</teiCorpus.2>
```

Reprezentacja informacji w ramach stylów realizowana jest z wykorzystaniem elementu `<group>`, łączącego w prosty sposób teksty poszczególnych próbek. Informacje nagłówkowe zapisywane są na dwóch poziomach: nadrzędnym (element `<teiHeader>`), odpowiadającym opisowi bibliograficznemu całego stylu oraz bezpośrednio na poziomie próbek (element `<head>`), z wykorzystaniem dostępnej informacji bibliograficznej tekstów składowych. Oto schemat opisu:

```
<tei.2 id=stylA>
   <teiHeader type=text>
```

```
      ... bibliografia stylu ...
   </teiHeader>

   <text lang=PL>
      <group>
         <text id=pu0001>
            <front>
               <head>
                  <bibl>
                     ... bibliografia próbki ...
                  </bibl>
               </head>
            </front>
            <body>
               ... tekst próbki ...
            </body>
         </text>
      </group>
   </text>
</tei.2>
```

Występujące w powyższym opisie elementy dodatkowe (`<front>`, `<body>`, `<bibl>`) są wynikiem wymagań składniowych schematu TEI — ich wprowadzenie jest z formalnego punktu widzenia konieczne, aczkolwiek nie zmienia znaczenia użytego zapisu, a jedynie wyraźniej oddziela jego poszczególne składniki.

Numery próbek, pochodzące ze zbiorów źródłowych korpusu i będące raczej ich identyfikatorami niż częścią opisu bibliograficznego, zostały przeniesione do identyfikatorów tekstów składowych — atrybutów `id` elementów `<text>`. Jednocześnie w celu zachowania unikatowości identyfikatorów zostały one rozszerzone o dwuliterowy symbol transzy (*PU* – publicystyka, *WP* – wiadomości prasowe, *PO* – styl popularnonaukowy, *PR* – proza, *DR* – dramat) i cztery cyfry numeru próbki (każda transza liczy ich 2 000).

Opis bibliograficzny próbek słownikowych został zapisany z wykorzystaniem specjalnych znaczników z zestawu tagów rdzennych (patrz rozdział 1.2.2), mogących wystąpić na dowolnym poziomie analizy — także w ramach segmentów[6]:

[6] Większość z wymienionych elementów to odpowiedniki znaczników nagłówkowych. Ich dokładny opis zawiera Dodatek B.

- `<title>` — tytuł tekstu,
- `<author>` — autor tekstu,
- `<publisher>` — wydawca (nazwa lub skrót nazwy wydawnictwa),
- `<date>` — data wydania,
- `<biblScope>` — odnośnik bibliograficzny (tu: opis strony).

Wspomniany element `<bibl>` występujący w sekcji nagłówka oddziela informację bibliograficzną od treści próbki. Przykładowy opis przybiera zatem postać:

```
<head>
   <bibl>
      <author>E. Bryll
      <title>Studium
      <publisher>LSW
      <date>1963
      <biblScope>str. 83
   </bibl>
</head>
```

Do podziału treści próbek na jednostki składniowe wykorzystany został zestaw znaczników opisany w rozdziale 2.2, a właściwie jego część wprowadzającą prostą identyfikację zdań, słów i znaków interpunkcyjnych. Oto przykład zapisu:

```
<body>
   <text>
     <p><s><w>Porucznik</w> <w>był</w> <w>wyraźnie</w>
           <w>zadowolony</w> <w>ze</w> <w>swego</w>
           <w>konceptu</w><c>.</c></s>
     ...
   </text>
</body>
```

Dodanie uzyskanej informacji morfologicznej nastąpiło natomiast na drodze definicji struktur cech w zmodyfikowanej postaci, opisanej w rozdziale 3.2.3. Opis ten polega na wykorzystaniu atrybutu `lemma` pola definicji słowa (`<w>`)

w celu zmniejszenia objętości zbioru struktur. Zgodnie z przeprowadzoną dyskusją można było także usunąć z opisu struktur informacje o instancjach związanych z nimi form (dane te są powielone w części tekstowej i mogą być łatwo odtworzone w razie potrzeby). Opisu struktur identyfikowane są kodami morfologicznymi systemu pozycyjnego, zaś składniki tekstu — zapisem złożonym z pierwszych dwóch liter nazwy stylu, czterech cyfr określających numer próbki i dwóch cyfr zapisujących numer porządkowy ortograficznego słowa w próbce. Oto fragment omawianego przykładu z pełnym opisem struktury jednego z haseł:

Zapis w bibliotece struktur

```
<fsLib>
   ...
   <fs type="opis morfologiczny" id="SSGM---------P">
      <f name=kategoria>
         <sym value=rzeczownik></f>
      <f name=liczba>
         <sym value=pojedyncza></f>
      <f name=przypadek>
         <sym value=mianownik></f>
      <f name=rodzaj>
         <sym value=męskoosobowy></f>
      <f name=zakres>
         <sym value=nazwaPospolita></f>
   </fs>
   ...
</fsLib>
```

Zapis w pliku słownika

```
<s>
   <w id=pr120731 lemma="porucznik"
                  ana="SSNM---------P">Porucznik</w>
   ...
</s>
```

Format ten wyklucza zapis niejednoznaczności kodowania polegającego na występowaniu homonimii form (możliwy jest zapis synkretyzmów jednej kategorii, lecz nie można zmienić postaci formy hasłowej), lecz w przypadku tekstów słownika frekwencyjnego nie zachodzi sytuacja, by formie przyporządkowana była więcej niż jedna analiza — wszystkie wystąpienia zostały ujednoznacznione jeśli chodzi o ich przynależność do zakresów znaczeniowych form hasłowych.

Formalnym uzupełnieniem powyższego zapisu jest umieszczona w osobnym dokumencie SGMLowym deklaracja systemu cech zbudowana na zasadach podanych w rozdziale 2.4. Zawiera ona definicje składników opisu form w postaci ich odpowiedników z fleksyjnego systemu kodowego w ramach poszczególnych kategorii odmiany. Wartości domyślnych nie określono — dla każdej kategorii dostępna jest jawna specyfikacja składników w bibliotece struktur. Oto przykład sekcji deklaracji dotyczącej wartości przypadka — pełna deklaracja dla plików słownika znajduje się na dołączonej do pracy płycie CD-ROM (patrz Dodatek C):

```
<fsDecl type="polskie kategorie morfologiczne">
   <fsDescr>Reprezentacja opisów morfologicznych
            dla języka polskiego</fsDescr>

   <fDecl name="kategoria">
      <fDescr>Typ jednostki zdania (część mowy)</fDescr>

      <vRange>
         <vAlt>
            <sym value="czasownik">
            <sym value="rzeczownik">
            <sym value="przymiotnik">
            <sym value="liczebnik">
            <sym value="zaimek">
            <sym value="przysłówek">
            <sym value="przyimek">
            <sym value="spójnik">
            <sym value="wykrzyknik">
            <sym value="partykuła">
         </vAlt>
      </vRange>

      <vDefault>
```

```
            <none>
          </vDefault>
      </fDecl>

      ... pozostałe kategorie ...

  </fsDecl>
```

Na zakończenie warto przypomnieć, że konwencja zapisu za pomocą struktur cech jest w dużej mierze umowna — na pewno warto jak najbardziej uściślać definicje poszczególnych cech, lecz sam zapis dopuszcza czasem pewną dowolność[7].

[7]Jest nią np. możliwość wykorzystania cech binarnych (oznaczanych jako `<plus>` i `<minus>`) do zapisu liczby zamiast używanych w moim projekcie wartości symbolicznych. Powodem ich wyboru był jednak postulat zachowania spójnego zapisu wewnątrz wszystkich kategorii.

Zakończenie

Zestaw tagów TEI wraz z SGMLowymi konstrukcjami umożliwiającymi bezpośrednie użycie polskich znaków w wybranym standardzie kodowym okazuje się dobrym formalizmem do zapisu polskich danych lingwistycznych. Po dokonaniu niewielkich zabiegów, związanych z utworzeniem reprezentacji analiz morfologicznych i składniowych właściwych dla języka polskiego zastosowanie dostępnych mechanizmów wydaje się mieć ogromną przewagę nad systemami nie objętymi standardami lub tworzonymi doraźnie.

Przedstawiony zapis, sprawdzony na przykładzie danych *Słownika frekwencyjnego* [16], może być z łatwością użyty do kodowania elektronicznego innych zbiorów tekstów zawierających informacje lingwistyczne, jak również rozszerzany o nowe pozycje czy modyfikowany, z uwzględnieniem indywidualnych potrzeb użytkowników.

Rozwinięciem pracy mogłoby być stworzenie dokładnej struktury zapisu informacji składniowych, zawierające klasyfikację fraz i zdań. Mogłaby ona stać się dobrym uzupełnieniem opisanego ogólnego mechanizmu analizy składniowej udostępnianego przez TEI i zaproponowanego opisu morfologicznego form polskich.

Innym ciekawym pomysłem na poprawę jakości kodowania polskich tekstów w ramach standardu TEI poprzez podniesienie zrozumiałości zapisu SGMLowego mogłoby być zastosowanie mechanizmu definicji polskich odpowiedników znaczników TEI. Możliwość taką przewidziano przy tworzeniu standardu, a jej skrótowy opis znalazł się w rozdziale *Modifying the TEI DTD: Renaming Elements Wytycznych TEI* [3]. W przypadku zastosowań lokalnych, nie włączających zapisywanych danych w skład projektu o większej, np. międzynarodowej skali, w zapisie tekstu najważniejsza staje się jego własność określana jako *descriptive markup* — jasność i zrozumiałość adiustacji. Postulat ten nie zawsze jest spełniony wobec konieczności używania predefiniowanych znaczników, co osłabia zainteresowanie wielu potencjalnych użytkowników standardem TEI, prowadząc do tworzenia własnych definicji typów dokumentów, prawie zawsze gorszych od zawartości DTD

TEI. Rozwiązaniem tego problemu jest prosta redefinicja nazw znaczników wprowadzonych przez TEI poprzez zastąpienie ich polskimi odpowiednikami (`<fileDesc>` – `<opisPliku>`). Polskie nazwy elementów przyczyniłyby się do znacznej poprawy zrozumiałości adiustacji, bez utraty jej zgodności z międzynarodowym standardem.

Dodatek A
Oznaczenia fleksyjne

Poniżej przedstawiam propozycję schematu oznaczeń fleksyjnych do dokładnego opisu tekstów polskich opartą o projekt pod nazwą *Zestaw Kodowy do Reprezentacji Polskiej Informacji Lingwistycznej* autorstwa Katarzyny Głowińskiej i Marcina Wolińskiego.

Pozycyjny charakter kodów, opisany w poniższej tabeli, zapewnia łatwość ich przetwarzania. Dla uproszczenia pominięto oznaczenia nieadekwatności kategorii, określane znakiem minusa na odpowiedniej pozycji kodowej.

Pozycja	**Znaczenie**	**Kod**	**Objaśnienie**
1	typ jednostki zdania (część mowy)	V	czasownik
		S	rzeczownik
		A	przymiotnik
		N	liczebnik
		Z	zaimek
		D	przysłówek
		P	przyimek
		C	spójnik
		I	wykrzyknik
		T	partykuła
		X	kod nieznany
2	liczba	S	pojedyncza
		P	mnoga
3	przypadek[1]	N	mianownik
		G	dopełniacz
		D	celownik
		A	biernik
		I	narzędnik
		L	miejscownik
		V	wołacz

[1]Pozycja może zostać wykorzystana do przechowania informacji o wartości przypadkowej formy dla liczebników zbiorowych oraz kategorii przypadka form, z którymi łączą się przyimki.

Pozycja	Znaczenie	Kod	Objaśnienie
4	rodzaj	M	męski
		P	męskoosobowy (l. poj.)
		A	męskozwierzęcy
		I	męskorzeczowy
		F	żeński
		N	nijaki
		O	męskoosobowy (l. mn.)
		R	niemęskoosobowy
		T	plurale tantum
5	stopień	P	równy
		C	wyższy
		S	najwyższy
6	osoba	1	pierwsza osoba
	lub oznaczenie	2	druga osoba
	formy	3	trzecia osoba
	bezosobowej	I	bezokolicznik
	czasownika[2]	B	forma bezosobowa (-no, -to)
		U	imiesłów przysłówkowy uprzedni
		W	imiesłów przysłówkowy współczesny
7	czas	T	teraźniejszy
		P	przeszły
		F	przyszły złożony
8	tryb	O	oznajmujący
		P	przypuszczający
		R	rozkazujący
9	aspekt	D	dokonany
		N	niedokonany
10	strona	C	czynna
		B	bierna
		Z	zwrotna
11	akcentowość	T	forma akcentowana
		N	forma nieakcentowana
12	poprzyimkowość	T	forma poprzyimkowa
		N	forma niepoprzyimkowa

[2]Połączenie kategorii osoby z identyfikatorami form bezosobowych stało się możliwe wobec rozłączności obu grup oznaczeń.

Pozycja	Znaczenie	Kod	Objaśnienie
13	oznaczenie dodatkowe form czasownikowych[3]	I	bezokolicznik jako forma składowa czasu przyszłego (będzie *pisać*)
		S	forma na -ł jako składowa form czasu przyszłego (będzie *pisał*)
		P	forma na -ł jako czas przeszły z ruchomą końcówką (skoroś *zjadł*)
		W	forma trybu przypuszczającego z ruchomą partykułą (bym *napisał*)
		R	opisowa forma trybu rozkazującego trzeciej osoby (niech *pisze*)
		B	czasownik *być* jako składowa form czasów złożonych (*będzie* pisał)
		O	czasownik *być*, *bywać*, *zostać* jako składowe form strony biernej (*jest* czytany, *bywał* sporządzany, *zostanie* zapisany)
14	oznaczenie nazw własnych	P	nazwa pospolita
		W	nazwa własna
		S	skrótowiec

[3]Pole z oznaczeniami dodatkowych własności form czasownikowych zostało wprowadzone w celu reprezentacji obecnych na listach rangowych Słownika Frekwencyjnego opisów funkcji czasowników złożonych. W przypadku ogólnym uzyskanie tego rodzaju informacji drogą analizy automatycznej może nie być zadaniem łatwym.

Dodatek B
Struktura nagłówka TEI

Zadaniem nagłówka TEI jest zgromadzenie informacji dokumentujących proces tworzenia i przetwarzania następującego po nim tekstu, w tym opisu bibliograficznego źródła wersji elektronicznej, danych autorów tekstu, osób i instytucji zaangażowanych w jego powstanie itd.

Poniższy opis ma pomóc w poprawnej interpretacji tagów użytych do zapisu danych nagłówkowych w poddawanych konwersji zbiorach tekstów[2].

B.1 Główne grupy elementów

Nagłówek jest wymaganym składnikiem dokumentu zgodnego z TEI (ang. TEI-conformant). To struktura o ściśle określonej budowie — korzeniem drzewa elementów jest element `<teiHeader>`, mieszczący pozostałe dane nagłówkowe, zorganizowane w cztery grupy:

- *opis pliku* (ang. file description), określony elementem `<fileDesc>`, będący zapisem informacji bibliograficznej o pliku komputerowym i źródłach tekstu elektronicznego,
- *opis kodowania* (ang. encoding description), zapisany w elemencie `<encodingDesc>`, stanowiący dokumentację zależności tekstu źródłowego od jego zapisu elektronicznego (rodzaj i cel wprowadzonych zmian, stopień identyczności obu reprezentacji tekstu, sposób zapisu niejednoznaczności i wykrytych błędów itp.),

[2] Oprócz rozdziału *The TEI Header* w *Wytycznych TEI* warto zapoznać się z tekstami D. Dunlopa [9] i R. Giordano [11] ze zbioru *TEI — Background and Context*.

- *profil tekstu* (ang. text profile), zdefiniowany elementem `<profileDesc>`, opisujący kwestie klasyfikacji, języka, typu i pochodzenia tekstu,
- *historia poprawek* (ang. revision history), określona elementem `<revisionDesc>`, zawierająca zapis zmian wprowadzanych w tekście elektronicznym w miarę jego rozwoju.

Oto ich schematyczny zapis:

```
<teiHeader>
   <fileDesc> ... </fileDesc>
   <encodingDesc> ... </encodingDesc>
   <profileDesc> ... </profileDesc>
   <revisionDesc> ... </revisionDesc>
</teiHeader>
```

B.2 Nagłówek minimalny

Z czterech części składowych nagłówka obowiązkowy jest tylko zapis pierwszej — opisu pliku, w którym także niektóre pola mogą zostać pominięte. Dla większości zastosowań taki rodzaj dokumentacji okazuje się wystarczający, często używa się więc specjalnej, minimalnej postaci nagłówka:

```
<teiHeader>
   <fileDesc>
      <titleStmt> ... </titleStmt>
      <publicationStmt> ... </publicationStmt>
      <sourceDesc> ... </sourceDesc>
   </fileDesc>
</teiHeader>
```

B.3 Opis pliku

Opis pliku jest najważniejszą częścią nagłówka. Jego składnikami są następujące elementy (dalszy podział grupuje jednostki niższego rzędu, zapisywane jako zawartość odpowiednich struktur):

- *deklaracja tytułu* (ang. title statement), określona elementem `<titleStmt>`, grupująca informacje o tytule publikacji oraz osobach i instytucjach odpowiedzialnych za jego powstanie, złożona z elementów:
  - `<title>` — tytuł pracy,
  - `<author>` — informacja o autorze lub autorach pracy (dane osobowe lub zbiorowe),
  - `<editor>` — informacja o osobie lub osobach odpowiedzialnych za edycję, kompilację, tłumaczenie tekstu itp.,
  - `<sponsor>` — nazwa sponsora,
  - `<funder>` — nazwa fundatora,
  - `<principal>` — identyfikator osoby odpowiedzialnej za powstanie tekstu elektronicznego,
  - `<respStmt>` — dodatkowa informacja o osobach odpowiedzialnych za powstanie, zapis, edycję tekstu dostarczana w przypadku nieadekwatności pozostałych elementów nagłówka; w skład modelu zawartości `<respStmt>` wchodzą elementy `<resp>` (rodzaj zadania) i `<name>` nazwa własna określająca jego wykonawcę.
- *deklaracja wydania* (ang. edition statement), zapisana w elemencie `<editionStmt>`, grupująca informacje dotyczące pojedynczego wydania tekstu; dodatkowo uszczegółowiana elementami:
  - `<edition>` — szczegóły opisu wydania,
  - `<respStmt>` — jak w przypadku deklaracji tytułu, z ograniczeniem do informacji o opisywanym wydaniu.
- *rozmiar* (ang. extent), zdefiniowany elementem `<extent>`, przechowujący oszacowanie rozmiaru tekstu wyrażone w zrozumiałych jednostkach pojemności informacyjnej (kilobajty, megabajty), liczbą jednostek logicznych (wyrazów, zdań) lub nośników fizycznych (dyskietek, płyt CD),
- *deklaracja publikacji* (ang. publication statement), określona elementem `<publicationStmt>`, zawierająca informacje dotyczące publikacji i dystrybucji tekstu, opisywane ciągłym tekstem opatrzonym znacznikiem `<p>` lub elementami o ścisłej strukturze:
  - `<publisher>` — dane wydawcy tekstu,

  - `<distributor>` — dane podmiotu odpowiedzialnego za dystrybucję tekstu,
  - `<authority>` — podmiot odpowiedzialny za udostępnienie tekstu (inny niż dwa poprzednie identyfikatory),
  - `<pubPlace>` — miejsce publikacji,
  - `<idno>` — numer identyfikacyjny pozycji (np. ISBN),
  - `<availability>` — opis dostępności tekstu, tj. ograniczeń jego wykorzystania i rozprowadzania, zakres praw autorskich itp.; może zostać podany ciągłym tekstem (w elemencie `<p>`) i opatrzony atrybutem `status` o wartości `free` (tekst ogólnodostępny), `unknown` (stan praw nieznany) lub `restricted` (wykorzystanie tekstu ograniczone),
  - `<date>` — data publikacji w dowolnym formacie,
  - `<address>` — adres osoby lub organizacji odpowiedzialnej za publikację; tekst musi zostać podany wewnątrz elementu `<addrLine>` (patrz przykład poniżej).

- *deklaracja notatek* (ang. notes statement), wyrażona elementem `<notesStmt>`, udostępniająca dodatkowe informacje bibliograficzne; jej zawartość stanowią elementy `<note>`,

- *opis źródła* (ang. source description), identyfikowany elementem `<sourceDesc>`, dostarczający danych bibliograficznych o źródle tekstu elektronicznego; może zostać podany ciągłym tekstem (`<p>`) lub przy użyciu elementów:

  - `<bibl>` — opis bibliograficzny o swobodnej strukturze,
  - `<biblStruct>` — opis zawierający wszystkie elementy opisu bibliograficznego,
  - `<biblFull>` — pełny opis zawierający wszystkie elementy składowe,
  - `<scriptStmt>` — opis tekstu mówionego,
  - `<recordingStmt>` — opis nagrań mowy,

Oto zwięzła ilustracja opisu pliku wraz z przykładowym wypełnieniem elementów (którego źródłem jest rozdział *The TEI Header Wytycznych TEI* [3]). Dla zachowania przejrzystości opuszczono niektóre znaczniki zamykające:

```
<teiHeader>
   <fileDesc>
      <titleStmt>
         <title>Opowiadania "Złoty żuk" i "Wahadło",
                wersja elektroniczna
         <author>Poe, Edgar Allen (1809-1849)
         <editor>Eric Johnson
         <sponsor>Association for Computers and the Humanities
         <funder>The National Endowment for the Humanities
         <principal>Gary Taylor
         <respStmt>
            <resp>wyboru dokonał
            <name>James D. Benson
      </titleStmt>

      <editionStmt>
         <edition>Wydanie pierwsze
         <respStmt>
            <resp>Adaptacja:
            <name>Elizabeth Kirk
      </editionStmt>

      <extent>3200 zdań

      <publicationStmt>
         <publisher>Oxford University Press
         <distributor>Oxford Text Archive
         <authority>James D. Benson
         <pubPlace>Oxford
         <date>2000
         <idno type=ISBN>0-19-254705-4
         <address>
            <addrLine>21 High Street, Oxford M24 3DF
         <availability status=restricted>
            <p>Tekst dostępny jedynie do celów naukowych
               po skontaktowaniu się z wydawcą.
      </publicationStmt>

      <notesStmt>
         <note>Skanowanie OCR wykonano w Instytucie Informatyki
               Uniwerystetu Warszawskiego
```

```
        </notesStmt>

        <sourceDesc>
           <biblStruct lang=EN>
           <monogr>
              <author>Poe, Edgar Allen (1809-1849)
              <title>Złoty żuk
              <imprint>
                 <pubPlace>London
                 <date>1850
        </sourceDesc>
     </fileDesc>
     ...
  </teiHeader>
```

B.4 Dodatkowe wskazówki dla opisujących zbiory danych

W przypadku elektronicznego kodowania zbioru tekstów nagłówek warto umieścić zarówno na poziomie poszczególnych składników zbioru, jak też na poziomie w stosunku do nich nadrzędnym. Dla rozróżnienia obu typów nagłówków element `<teiHeader>` posiada opcjonalny atrybut `type` o dozwolonych wartościach `text` (wartość domyślna, nagłówek umieszczony na poziomie pojedynczego tekstu) i `corpus` (nagłówek całego korpusu).

Przy tworzeniu nagłówka warto też pamiętać, że większość jego elementów składowych zezwala na podanie wartości w postaci ciągłego tekstu, bez wykorzystywania szczegółowych pól struktury. Takie podejście może pomóc w budowaniu złożonych opisów bibliograficznych, których przekształcanie do znormalizowanej postaci równałoby się z niedokładną interpretacją lub nawet utratą pewnych informacji. Problemy takie nie wystąpią, gdy w opisie użyjemy tekstu poprzedzonego znacznikiem akapitu (`<p>`), musimy jednak pamiętać, że wówczas jednoznaczna interpretacja jego zawartości może być utrudniona.

Dodatek C
Opis zawartości dołączonej płyty CD-ROM

Zawartość płyty stanowią katalogi `PRACA` i `KORPUS`. W katalogu `PRACA` znajduje się tekst niniejszej pracy w postaci plików wynikowych DVI i Postscript. Katalog `KORPUS` zawiera pliku korpusu *Słownika Frekwencyjnego Polszczyzny Współczesnej* w opisanym formacie.

Trzonem korpusu jest plik `sfpw.sgm`, tworzący środowisko zapisu jego tekstów poprzez dołączenie definicji typu dokumentu oraz zawierający informacje nagłówkowe wersji elektronicznej Korpusu. Do walidacji dokumentów wykorzystywana jest umieszczona w pliku `latin2.dcl` deklaracja SGMLowa zezwalająca na używanie w tekstach Korpusu polskich znaków w standardzie ISO Latin-2 oraz umożliwiająca poprawną interpretację znaczników TEI.

Właściwe teksty poszczególnych stylów (transz) znajdują się w pięciu plikach:

- `a-publ.sgm` — styl A (publicystyka), 10,3 MB,
- `b-prasa.sgm` — styl B (drobne wiadomości prasowe), 10,2 MB,
- `c-popul.sgm` — styl C (popularnonaukowy), 10,6 MB,
- `d-proza.sgm` — styl D (proza), 11,3 MB,
- `e-dramat.sgm` — styl E (dramat), 12,8 MB.

Każdy z nich zawiera nagłówek transzy oraz teksty próbek opatrzone informacją bibliograficzną w dostępnej postaci.

Formalna deklaracja systemu cech znajduje się w pliku `morf.fsd`, natomiast wspólne dla wszystkich transz deklaracje struktur cech mieści plik `fslib.sgm`.

Ponadto katalog zawiera plik `iso88592.wsd` z deklaracją WSD dla standardu ISO Latin-2, niezbędną do walidacji dokumentów z językiem tekstu określonym jako polski.

Literatura cytowana

[1] *Corpus Encoding Standard — Document CES 1*, wersja 1.5. 2 lutego 2000 <http://www.cs.vassar.edu/CES/>.

[2] *Getting Started with SGML — A Guide to the Standard Generalized Markup Language and Its Role in Information Management*, 1995. ArborText Inc. 7 lutego 2000 <http://www.arbortext.com/Think_Tank/SGML_Resources /Getting_Started_with_SGML/getting_started_with_sgml.html>

[3] *Guidelines for Electronic Text Encoding and Interchange (TEI P3).* Red. C. M. Sperberg-McQueen i Lou Burnard. Chicago, Oxford, 1994. The Association for Computers and the Humanities (ACH), The Association for Computational Linguistics (ACL), The Association for Literary and Lingustic Computing (ALLC). Dostępne: <http://etext.lib.virginia.edu/TEI.html>

[4] *International Standard ISO 8879 Information Processing — Text and Office Systems — Standard Generalized Markup Language (SGML).* Genewa, 1986. ISO (International Organization for Standardization).

[5] Bień, Janusz S. *Kodowanie tekstów polskich w systemach komputerowych.* Referat wygłoszony na konferencji „Multimedia w nauczaniu języka rodzimego jako obcego". Uniwersytet Śląski, Katowice. 7-8 grudnia 1988. Przedruk: Postscriptum 27-29. S. 4-27. Dostępny też: <ftp://ftp.mimuw.edu.pl/pub/users/polszczyzna/ogonki/katow98.*>

[6] Bień, Janusz S. *Polskojęzyczne reguły składni SGML.* Warszawa, 1998.

[7] Burnard, Lou; Sperberg-McQueen, C. M. *TEI Lite: An Introduction to Text Encoding for Interchange.* Czerwiec 1995 <http://www.uic.edu/orgs/tei/intros/teiu5.html>.

[8] DuCharme, Bob. *SGML CD — A Complete SGML Toolkit.* 1998. Prentice Hall PTR.

[9] Dunlop, Dominic. *Practical Considerations in the Use of TEI Headers in a Large Corpus*, W: „Text Encoding Initiative — Background and Context". S. 85-98. Red. Nancy Ide i Jean Veronis. 1995. Kluwer Academic Publishers.

[10] Gaylord, Harry E. *Character representation* W: „Text Encoding Initiative — Background and Context". S. 51-73. Red. Nancy Ide i Jean Veronis. 1995. Kluwer Academic Publishers.

[11] Giordano, Richard. *The TEI Header and the Documentation of Electronic Texts* W: „Text Encoding Initiative — Background and Context". S. 75-84. Red. Nancy Ide i Jean Veronis. 1995. Kluwer Academic Publishers.

[12] Goldfarb, Charles F. *Final text of revised TC2 to ISO 8879:1986 (merged text of TC2 and TC3).* 6 grudnia 1998 <http://www.ornl.gov/sgml/sc34/document/0029.htm>.

[13] Goldfarb, Charles F. *The SGML Handbook.* 1990. Oxford University Press.

[14] Herwijnen van, Eric. *Practical SGML.* Boston-Dordrecht-London, 1994. Kluwer Academic Publishers.

[15] Kurcz, Ida; Lewicki, Andrzej; Sambor, Jadwiga; Woronczak, Jerzy. *Słownictwo współczesnego języka polskiego. Listy frekwencyjne.* Warszawa, 1974. Uniwersytet Warszawski.

[16] Kurcz, Ida; Lewicki, Andrzej; Sambor, Jadwiga; Szafran, Krzysztof. *Słownik frekwencyjny polszczyzny współczesnej*, Kraków, 1990. Instytut Języka Polskiego PAN.

[17] Langendoen, D. Terence; Simons, Gary F. *A Rationale for the TEI Recommendations for Feature-Structure Markup* W: „Text Encoding Initiative — Background and Context". S. 191-209. Red. Nancy Ide i Jean Veronis. 1995. Kluwer Academic Publishers.

[18] Nazarczuk, Marta. *Wstępne przygotowanie korpusu „Słownika frekwencyjnego polszczyzny współczesnej" do dystrybucji na CD-ROM.* Praca magisterska napisana pod kierunkiem dra hab. Janusza S. Bienia. Warszawa, 1997. Instytut Języka Polskiego Uniwersytetu Warszawskiego.

[19] Saloni, Zygmunt. *Słownik frekwencyjny polszczyzny współczesnej.* W: ComputerWorld. S. 16-17. 4 listopada 1991.

[20] Szafran, Krzysztof. *Analizator morfologiczny SAM-95 — opis użytkowy.* Maj 1996. Instytut Informatyki Uniwersytetu Warszawskiego.

[21] Tokarski, Jan. *Schematyczny indeks a tergo polskich form wyrazowych.* Red. Zygmunt Saloni. Warszawa. Wydawnictwo Naukowe PWN.

[22] Wall, Larry; Schwartz, Randal L. *Programming Perl.* 1991. O'Reilly & Associates, Inc.