Standardy

SGML w praktyce

Piotr Bolek

Co to jest SGML

SGML (*Standard Generalized Markup Language*) jest nazwą języka zdefiniowanego w standardzie ISO 8879 opublikowanym w październiku 1986 r. Twórcą standardu jest dr Charles Goldfarb. SGML jest językiem stosunkowo złożonym i bywa określany jako:

- język oznaczania (etykietowania) tekstu (ang. *tagging language*),
- narzędzie do opisywania logicznej struktury tekstów,
- schemat łączenia i odwoływania się do plików,
- system tekstowej bazy danych,
- podstawa systemów hipertekstowych i multimedialnych,
- narzędzie umożliwiające tworzenie szablonów dla dokumentów tekstowych,
- narzędzie pozwalające wykorzystać zakodowany nim tekst w sposób nieprzewidziany przez autora,
- przenośny, niezależny od architektury sprzętu i oprogramowania sposób kodowania tekstu,
- narzędzie umożliwiające zapisywanie dowolnych struktur hierarchicznych,
- rozszerzalny język opisu dokumentów,
- standard komunikacji między różnymi typami sprzętu i programami użytkowymi,
- metajęzyk umożliwiający tworzenie języków prezentacji struktury dokumentów.

Jak pisze C. Goldfarb wszystko to jest prawdą, ale nie całą. SGML jest po części tym wszystkim, ale jako całość jest czymś jeszcze więcej.

Potrzeba istnienia takiego standardu jak SGML wynikała z problemów, które pojawiają się w trakcie elektronicznego przetwarzania tekstów. Częste są w takich przypadkach trudności z konwersją tekstów podczas przenoszenia między różnymi programami (edytorami, systemami składu itp.) Przejście z jednego programu do innego często prowadzi do utraty informacji o strukturze, formatowaniu tekstu, albo nawet jego zawartości merytorycznej. Problemy istnieją także w przypadku wykorzystania w dokumencie niestandardowego zestawu znaków. SGML pozwala rozwiązywać takie problemy. Definiuje on zasady tworzenia języków opisujących logiczną strukturę dokumentów. Dzięki zestawowi deklaracji pozwala także jednoznacznie określić sposób kodowania informacji zastosowany przy tworzeniu dokumentu. Ułatwia więc przenoszenie dokumentów między różnymi systemami.

Standard służy jedynie do opisywania logicznej struktury dokumentów, nie determinuje ostatecznej formy prezentacji informacji, która może być docelowo przekształcana w najróżniejszy sposób. Dokument zakodowany z wykorzystaniem SGML-a może służyć jako postać wyjściowa do formatowania tej samej informacji w różny sposób i prezentacji z użyciem różnych mediów np. w formie drukowanej na papierze, w postacji hipertekstu albo tekstowej bazy danych. Bardzo prosty przykład tego typu zastosowania będzie zaprezentowany na zakończenie artykułu.

Istnieje wiele narzędzi służących do przetwarzania dokumentów zakodowanych w SGML-u. Wiele z nich jest dostępnych na zasadach public domain, coraz częściej pojawiają się także w produktach komercyjnych filtry służące do przekształcania informacji do postaci SGML.

Sposób kodowania dokumentów w SGML-u

Dokument w SGML-u zaczynać się powinien od zestawu deklaracji (*SGML declaration*) określających pewne parametry (np. długość identyfikatorów używanych do oznaczania elementów struktury), definiujących zestaw znaków używany w dokumencie oraz znaki specjalne służące do oznaczania różnych typów identyfikatorów. Deklaracje te nie zawsze występują w postaci jawnej, często ukryte są w systemie służącym do przygotowywania lub przetwarzania tekstu.

Drugim ważnym elementem jest Definicja Typu Dokumentu (ang. *Document Type Definition* – *DTD*). DTD określa w sposób formalny logiczną strukturę dokumentu – podobnie jak style dokumentu w LaTeX-u (struktura ta musi być zachowana jednak bardziej rygorystycznie niż w LaTeX-u – nie byłoby możliwe opuszczenie jednego poziomu nagłówków np. bezpośrednio

po `\chapter` użycie `\subsection`, co w LaTeX-u daje efekt nieco dziwaczny, ale nie powoduje błędu).

Definicja DTD może znajdować się wewnątrz dokumentu, w jego nagłówku, jeżeli jednak ma być użyta przy tworzeniu wielu tekstów, zwykle zapisywana jest w oddzielnym pliku.

Elementy. W DTD zdefiniowane są wszystkie logiczne części danego typu dokumentu – elementy (ang. *elements*), które mają podobne zastosowanie jak LaTeX-owe środowiska. Każdy element składa się z etykiety otwierającej (ang. *start-tag*) i zamykającej (ang.*end-tag*). Etykieta otwierająca ma zwykle postać `<elem>`, a zamykająca `</elem>`. Inaczej niż w LaTeX-u, nie wszystkie elementy występujące w tekście muszą mieć zaznaczone jawnie etykiety otwierające i zamykające. W deklaracji SGML możliwe jest ustawienie parametrów dopuszczających różne formy minimalizacji wpisywanych etykiet. Minimalizacja taka może polegać na możliwości opuszczenia etykiety (`OMITTAG`), użyciu skróconej wersji etykiety (`SHORTTAG`), grupowaniu etykiet (`RANK`), albo automatycznym rozpoznawaniu etykiet (`DATATAG`). Dodatkowo możliwe jest używanie skrótów (ang. *short references*), pozwalających w pewnych kontekstach używać zamiast etykiet zdefiniowanych uprzednio symboli, lub ich sekwencji.

Po dopuszczeniu możliwości opuszczania etykiet można pominąć etykietę otwierającą lub zamykającą, jeżeli da się z kontekstu wydedukować konieczność jej istnienia. Jeżeli np. zdefiniujemy elementy `chap`, `title` i `p` w następujący sposób:

```
<!ELEMENT chap - - (title,p+)>
<!ELEMENT title O O (#PCDATA)>
<!ELEMENT p - O (#PCDATA)>
```

to możemy napisać w dokumencie:

```
<chap>
<title>Tytuł rozdziału
<p>Pierwszy akapit tekstu
</chap>
```

a nawet:

```
<chap>
Tytuł rozdziału
<p>Pierwszy akapit tekstu
</chap>
```

i będzie to równoważne z:

```
<chap>
<title>Tytuł rozdziału
</title>
<p>Pierwszy akapit tekstu
</p>
</chap>
```

W powyższym przykładzie możemy także zobaczyć jak wygląda definiowanie i używanie elementów. Element `chap` zdefiniowaliśmy jako składający się z tytułu (element `title`), z następującym po nim (przecinek „`,`”określa następstwo elementów) co najmniej jednym akapitem (znak plus „`+`” oznacza wystąpienie co najmniej jednokrotne). Element `title` zdefiniowany został jako zawierający dane tekstowe (`#PCDATA` ang. *parsed characted data*), bez żadnych elementów niższego poziomu. Znaki „`- -`”, „`O O`” i „`- O`” występujące w definicjach elementów `chap`, `title` i `p` określają czy możliwe jest pominięcie odpowiednich etykiet otwierających lub zamykających. Znak minus „`-`” na pozycji pierwszej oznacza, że etykieta otwierająca musi wystąpić, na pozycji drugiej, że wymagana jest etykieta zamykająca. Litera „`O`” zezwala na pominięcie odpowiedniej etykiety.

Atrybuty. Każdy z elementów może zawierać atrybuty określające pewne cechy szczególne np. fakt numerowania listy wyliczeń:

```
<list type="numbered">
<item> Pierwszy
<item> Drugi
</list>
```

Element typu `list` ma w tym przykładzie atrybut `type`, któremu przypisano wartość `numbered`. W DTD zawarte są definicje atrybutów wszystkich elementów wraz z ich typami i dopuszczalnymi wartościami.

Definicje. W dokumencie SGML istnieje także możliwość definiowania pewnych powtarzalnych części jako pewnego rodzaju definicji zwanych w terminologii SGML-a *entities*. Definicja może zastąpić pewien często używany tekst i jest w takim przypadku podobna do bezparametrowej definicji TeX-owej np.:

```
<!ENTITY sgml "Standard Generalized
  Markup Language">
```

W tekście dokumentu każdy napis `&sgml;` zostanie zastąpiony przez `Standard Generalized Markup Language`.

Inny rodzaj definicji może być używany w pliku DTD, dając możliwość definiowania pewnych wspólnych cech kilku elementów naraz. Przykład tego typu definicji pojawi się w przykładowym DTD na końcu artykułu.

Inne składowe dokumentu SGML. W dokumencie SGML może wystąpić jeszcze wiele innych składowych, np. wyróżnione sekcje (ang. *marked sections*), instrukcje formatujące (ang. *procesing instructions*), poddokumenty (ang. *subdocuments*), odwołania do znaków niedostępnych z klawiatury (ang. *character references*). Nie ma tutaj miejsca na szczegółowe omówienie ich definiowania i sposobu wykorzystania.

Praca z SGML-em

W praktyce najważniejszą sprawą jest wstępne zidentyfikowanie logicznej struktury dokumentu i zbudowanie hierarchii elementów tej struktury. Następnie należy stworzyć lub wykorzystać istniejący DTD, określający strukturę dokumentu formalnie.

Po zidentyfikowaniu wszystkich logicznych elementów tekstu i wybraniu odpowiedniego typu dokumentu należy określić strukturę dokumentu wstawiając do tekstu etykiety (ang. *tags*), które zaznaczają typy poszczególnych elementów dokumentu. Wprowadzenie etykiet do tekstu może mieć miejsce w trakcie tworzenia dokumentu i czasem może być zupełnie niewidoczne dla użytkownika. Etap ten może stanowić koniec pracy z tekstem. Dalsza obróbka może być dokonywana przez innych. Zwykle należy jednak jeszcze dokonać weryfikacji dokumentu, czyli sprawdzić czy struktura stworzonego dokumentu jest zgodna z definicjami zawartymi w DTD. Do weryfikacji struktury służą programy zwane parserami, czytające DTD oraz dokument i stwierdzające czy dany dokument ma strukturę określoną w definicji czy nie. Czasem parser może także dokonać przekształcenia struktury dokumentu na postać kanoniczną (rozwiniętą), w której jawnie występują wszystkie etykiety otwierające i zamykające wraz ze wszystkimi atrybutami. Wyjście z parsera może być wykorzystane w etapie następnym – przygotowaniu dokumentu do prezentacji w sformatowanej postaci.

Formatowanie tekstu to etap, którym standard się już nie zajmuje, ale etap ten jest zwykle niezbędny, aby zakodowana informacja mogła być w sensowny sposób wykorzystana. Dokument może być sformatowany w sposób prosty bezpośrednio z dokumentu źródłowego, przez prostą edycję przy pomocy standardowych narzędzi, służących do przetwarzania tekstu (sed, awk, Perl), albo po uprzednim przetworzeniu tekstu przez parser na postać „rozwiniętą”.

Przykład

Na koniec chciałbym przedstawić bardzo prosty przykład prezentujący opisane powyżej cechy SGML oraz możliwości, które daje ten sposób kodowania informacji. Przykładem tym jest przygotowywanie tabel do publikacji w WWW. Nie jest to przykład bardzo skomplikowany, stanowi on próbę pokazania możliwości praktycznego wykorzystania SGML-a.

Problem i propozycja rozwiązania. Prezentacja tabel w WWW sprawia problemy, ponieważ brak jest jednoznacznego sposobu ich kodowania w języku HTML. W definicji standardu HTML3 istnieje co prawda dosyć ciekawy sposób tworzenia tabel, ale większość używanych przeglądarek nie obsługuje jezyka HTML3. Sposób zapisywania tabel rozumiany przez przeglądarkę Netscape też nie jest całkiem wygodny, chociaż daje duże możliwości i jest częściowo zgodny ze standardem HTML3. Większość przeglądarek zupełnie jednak nie radzi sobie z wyświetlaniem tabel.

W zaproponowanym rozwiązaniu tabele tworzone są w kilku wariantach – oddzielnie dla Netscape’a w postaci przez niego zrozumiałej, a oddzielnie dla pozostałych przeglądarek – w postaci preformatowanego tekstu ASCII (tabelki są w tym przypadku składane z wykorzystaniem programu `groff` z preprocesorem `tbl`). Obie te wersje (oraz dodatkowa trzecia LaTeX-owa) są tworzone z tej samej wersji źródłowej zapisanej w SGML-u.

DTD. Aby umożliwić przetwarzanie tabel należy przede wszystkim przygotować odpowiednią defi-

nicję typu dokumentu – DTD. Do tego celu została wykorzystana zmodyfikowana i nieco uproszczona wersja definicji tabel z DTD dla języka HTML 3.0.

```
<!-- tab.dtd - DTD dla tabelek -->

<!ELEMENT TABLE - - (TR+)>
<!ATTLIST TABLE
     border (border) #IMPLIED
     colspec CDATA  #IMPLIED
     units (en|pixels|relative) en
     width NUMBER   #IMPLIED
     padding NUMBER #IMPLIED
     spacing NUMBER #IMPLIED
     >

<!ENTITY % cell "TH | TD">
<!ENTITY % horiz.align
        "left|center|right|justify">
<!ENTITY % vert.align
        "top|middle|bottom|baseline">

<!ELEMENT TR O O (%cell)* >
<!ATTLIST TR
     align (%horiz.align) #IMPLIED
     >

<!ELEMENT (%cell) - O (#PCDATA)>
<!ATTLIST (%cell)
     colspan NUMBER  1
     align (%horiz.align) #IMPLIED
     >

<!ELEMENT TAB O O ANY>
```

Narzędzia. Do przetwarzania tabel użyty został parser `sgmls` Jamesa Clarka, oraz program `sgmlspl` napisany w Perlu, korzystający z biblioteki `SGMLS.pm` (autorem programu i biblioteki jest David Megginson z Kanady).

Program `sgmlspl` działa na danych wyjściowych z parsera `sgmls`. Argumentem tego programu jest plik ze specyfikacją, w którym w postaci kodu w Perlu zdefiniowane są akcje, które należy wykonać po napotkaniu kolejnych elementów struktury dokumentu. Dla każdego z formatów wyjściowych konieczny jest oczywiście inny plik ze specyfikacją. Kod specyfikacji nie będzie tutaj przedstawiany ponieważ jest dosyć specyficzny i stosunkowo skomplikowany, byłby więc mało zrozumiały nawet dla osób znających Perla. Zainteresowanych odsyłam do dokumentacji biblioteki `SGMLS.pm` i programu `sgmlspl`.

Przykładowa tabelka. Oto przykładowa tabelka zapisana zgodnie ze zdefiniowaną w `tab.dtd` specyfikacją.

```
<!doctype tab system>

<TABLE width=440 border padding=3
          COLSPEC="lrrrr">
<TR ALIGN=CENTER>
 <TD>
 <TD COLSPAN=4>Zasoby
 <TR>
 <TD>
 <TD align=center colspan=2>Udokumentowane
 <TD align=center colspan=2>Wydobywalne
<TR ALIGN=left>
 <TD>Surowce
 <TD>mld t <TD> %
 <TD> mld t <TD> %
<TR>
 <TD> Węgiel kam.
 <TD> 36,18 <TD> 63,1 <TD> 14,47 <TD> 69,8
<TR>
 <TD>Węgiel brun.
 <TD>  2,74 <TD>  4,8 <TD>  2,46 <TD> 11,9
<TR>
 <TD>Gaz ziemny
 <TD>  0,17 <TD>  0,3 <TD>  0,14 <TD>  0,7
<TR>
 <TD>Ropa naftowa
 <TD> 0,005 <TD> 0,01 <TD> 0,004 <TD> 0,01
<TR>
 <TD>Energia wód
 <TD> 18,29 <TD> 31,9 <TD> 3,65 <TD> 17,6
</TABLE>
```

Tabelka ASCII. Oto przykładowa tabelka sformatowania programem `groff` i preprocesorem `tbl`:

```
+-------------+--------------------------------+
|             |            Zasoby              |
+-------------+----------------+---------------+
|             | Udokumentowane | Wydobywalne   |
+-------------+--------+-------+-------+-------+
|Surowce      | mld t  | %     | mld t | %     |
+-------------+--------+-------+-------+-------+
|Węgiel kam.  |  36,18 |  63,1 | 14,47 | 69,8  |
+-------------+--------+-------+-------+-------+
|Węgiel brun. |   2,74 |   4,8 |  2,46 | 11,9  |
+-------------+--------+-------+-------+-------+
|Gaz ziemny   |   0,17 |   0,3 |  0,14 |  0,7  |
+-------------+--------+-------+-------+-------+
|Ropa naftowa |  0,005 |  0,01 | 0,004 | 0,01  |
+-------------+--------+-------+-------+-------+
|Energia wód  |  18,29 |  31,9 |  3,65 | 17,6  |
+-------------+--------+-------+-------+-------+
```

Taką tabelkę można umieścić na stronie HTML przeznaczonej dla „ubogich” przeglądarek WWW.

Tabelka HTML. A tak wygląda ta sama tabelka po przetworzeniu na HTML oglądana pod Netscapem:

	Zasoby			
	Udokumentowane		Wydobywalne	
Surowce	mld t	%	mld t	%
Węgiel kam.	36,18	63,1	14,47	69,8
Węgiel brun.	2,74	4,8	2,46	11,9
Gaz ziemny	0,17	0,3	0,14	0,7
Ropa naftowa	0,005	0,01	0,004	0,01
Energia wód	18,29	31,9	3,65	17,6

Tabelka LaTeX. Na koniec ta sama tabelka w LaTeX:

	Zasoby			
	Udokumentowane		Wydobywalne	
Surowce	mld t	%	mld t	%
Węgiel kam.	36,18	63,1	14,47	69,8
Węgiel brun.	2,74	4,8	2,46	11,9
Gaz ziemny	0,17	0,3	0,14	0,7
Ropa naftowa	0,005	0,01	0,004	0,01
Energia wód	18,29	31,9	3,65	17,6

Literatura

1. Charles F. Goldfarb: *The SGML Handbook*, Oxford University Press, 1995,
2. Martin Bryan: *SGML, an author's guide to the Standard Generalized Markup Language*, Addison-Wesley, 1988.
3. David Megginson: Dokumentacja do biblioteki SGMLS.pm i programu sgmlspl, `http://www.uottawa.ca/~dmeggins/SGMLSpm/sgmlspm.html`, `http://www.uottawa.ca/~dmeggins/sgmlspl/sgmlspl.html`
4. Daniel Gilly: *Unix in the Nutshell*, O'Reilly & Associates, 1994.

◇ Piotr Bolek
`P.Bolek@ia.pw.edu.pl`