STUDIA INFORMATICA 2000
Volume 21 Number 3 (41)

Zdzisław SROCZYŃSKI
Politechnika Śląska, Instytut Matematyki

MATHML — JĘZYK OPISU WYRAŻEŃ MATEMATYCZNYCH W DOKUMENTACH INTERNETOWYCH

Streszczenie. W pracy przedstawiono język opisu wyrażeń matematycznych na potrzeby dokumentów internetowych — MathML (Mathematical Markup Language). Pokazano najważniejsze zadania związane z implementacją i wdrożeniem MathML, jak również metody zapewnienia współistnienia i wzajemnej konwersji w ramach dotychczas stosowanych rozwiązań. Istotnym elementem pracy są przykłady oraz porównanie wydajności różnych metod wizualizacji wzorów w przeglądarkach internetowych.

MATHML — MARKUP LANGUAGE FOR MATHEMATICAL DOCUMENTS ON THE WORLD WIDE WEB

Summary. MathML — markup language for including mathematical notation in web pages has been described in the paper. Main tasks related to the implementation of MathML standard, ways of conversion and cooperation with other systems have been shown and analysed. A comparison of different visualization methods and examples are an important part of the paper.

1. Wprowadzenie – przegląd dotychczasowych rozwiązań

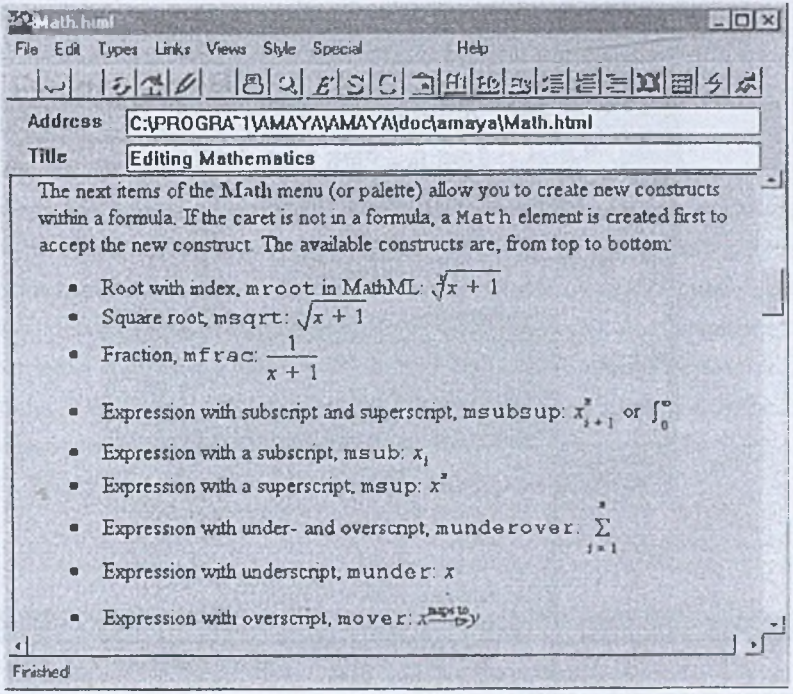
Zapis wzorów matematycznych w dokumentach przeznaczonych do publikacji w Internecie sprawiał zawsze sporo kłopotów i żadne z wypracowanych dotychczas rozwiązań nie zyskało sobie ani powszechnej aprobaty użytkowników, ani też nie stało się formalnym standardem. Najprostsze metody publikacji tekstów zawierających dużą liczbę wzorów matematycznych, polegające na konwersji z systemów składu typu LaTeX czy z edytorów tekstu (np. MS Word) do plików graficznych w formacie GIF lub JPG rażą nieproporcjonalnie dużą

objętością dokumentu wynikowego w stosunku do niesionej informacji oraz niską jakością wydruku.

Ponadto rozwiązania takie uzależniają zarówno autora, jak i odbiorcę od zastosowanego narzędzia, nie da się bowiem wykorzystać stworzonych w taki sposób stron WWW bez źródłowego dokumentu w standardzie LaTeX czy w formacie odpowiedniego edytora tekstu. Można powiedzieć, że pliki graficzne zawierają wyłącznie informacje o wyglądzie wzoru, nie można z nich jednak odtworzyć struktury wzoru. Przeczy to w oczywisty sposób idei języka HTML, będącego uniwersalnym i samowystarczalnym językiem opisu strony. W pewnym zakresie niedoskonałość tę można ograniczyć modyfikując formaty plików graficznych tak, aby zawierały również opis struktury wzoru w pewnej notacji (np. LaTeX). Wizualizacja tak skonstruowanych wzorów w przeglądarce internetowej niczym się nie różni od przedstawiania zwykłych rysunków, ale istnieje możliwość edycji wzoru bezpośrednio ze zmodyfikowanego pliku graficznego za pomocą specjalistycznego oprogramowania [5]. Wciąż pozostają jednak problemy objętości i jakości wydruku.

Dążenie do ograniczenia objętości dokumentów internetowych zawierających wzory matematyczne, a tym samym przyspieszenia ich wczytywania, zaowocowało pojawieniem się aplikacji zapisujących wzory w postaci odpowiednio sformatowanych symboli i standardowych znaczników HTML. Dokumenty takie, choć zajmują niewiele miejsca, odbiegają znacznie od standardów wyznaczonych przez współczesne systemy DTP, jeśli chodzi o jakość.

Rozwiązaniem opisanych powyżej problemów byłoby zapisywanie struktury wzoru bezpośrednio w pliku HTML w postaci rozkazów tekstowych podobnych do standardowych znaczników HTML. Taki dokument wymagałby tylko jednej transakcji z serwera WWW do komputera z przeglądarką, a sama wizualizacja następowałaby już bezpośrednio w przeglądarce. Pierwszym rozwiązaniem wykorzystującym podobną ideę były próby stworzenia apletu w języku Java, który dokonywałby wizualizacji wzoru po stronie przeglądarki na podstawie opisu struktury w pewnym języku. Zasadniczą wadą takiego rozwiązania jest powolność procesu wizualizacji (strona jest transmitowana szybciej, ale dłużej trwa jej wyświetlanie) wynikająca z natury języka Java.

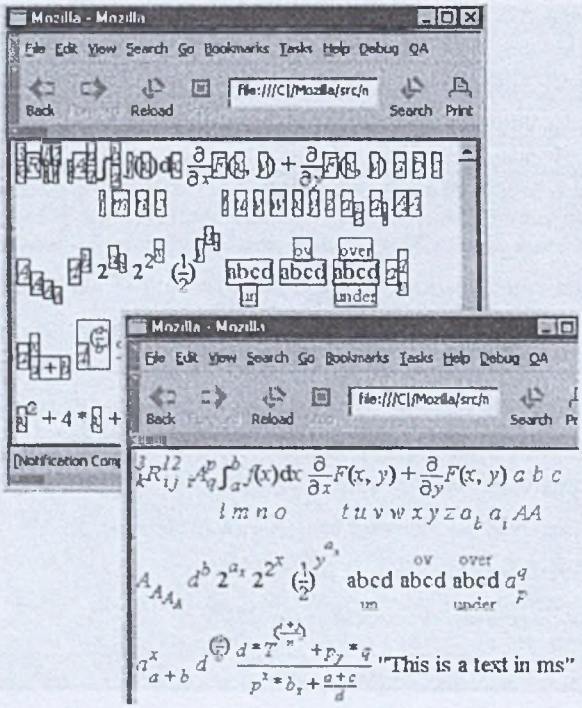


Rys. 1. Wzory matematyczne w eksperymentalnej przeglądarce Amaya (opracowanej przez konsorcjum W3C)

Fig. 1. Equations rendered by experimental browser *Amaya* (from W3C Consortium)

Nadmienić należy, że choć to ostatnie rozwiązanie nie wyszło jak dotąd poza stadium prób, z pewnością wyznaczyło właściwą drogę prowadzącą do rozwiązania omawianego zagadnienia. Jednym z możliwych udoskonaleń jest napisanie specjalnej „wtyczki" (ang. *plug-in*) dla przeglądarki WWW, realizującej wizualizację wzorów zapisanych w odpowiednim formacie. Wtyczki tworzy się jako dynamicznie ładowane biblioteki skompilowane dla konkretnego typu procesora, stąd mogą one działać wielokrotnie szybciej niż aplety, a ponadto nie wymagają stosowania nowych narzędzi programistycznych i kosztownego czasem przenoszenia kodu. Idealnym rozwiązaniem byłoby oczywiście zatwierdzenie ogólnego standardu opisu struktury wzorów i wbudowanie jego interpretera bezpośrednio w przeglądarki WWW.

Prace tego typu zostały podjęte, ale zarówno w przypadku przeglądarki Internet Explorer, jak i Netscape Navigator, ze względu na złożoność zagadnienia nie należy się spodziewać ostatecznych i wyczerpujących problem wyników w najbliższej wersji.



Rys. 2. Wzory matematyczne zapisane w MathML w przeglądarce Mozilla (Netscape) (u góry z zaznaczoną strukturą, na dole postać finalna). Źródło: [9]

Fig. 2. MathML equations rendered by Mozilla browser (structure view above, final rendering below). Source: [9]

2. Standard MathML

Jedynym standardem branym w tej chwili pod uwagę w celu rozwiązania opisanych powyżej problemów jest MathML (ang. *Mathematical Markup Language*) [1][3], język zaproponowany przez konsorcjum WWW do opisu wzorów matematycznych na stronach WWW. Język ten posługuje się zapisem czysto tekstowym, stanowiąc naturalne rozszerzenie standardu HTML o brakujące dotychczas znaczniki pozwalające uzyskać wzory. Ponadto MathML pozwala na pełne wykorzystanie dokumentów HTML przy wymianie danych między różnymi aplikacjami matematycznymi. Wprowadzono w nim bowiem ścisły podział na warstwę prezentacji, która służy wyłącznie wizualizacji, oraz warstwę struktury, dzięki której z kolei można interpretować sens zapisu matematycznego w dowolnym programie. Znika więc podstawowy problem dotychczasowych użytkowników, polegający na konieczności utrzymywania kopii

dokumentu w lokalnym formacie wykorzystywanego programu (w celu dokonania zmian i poprawek) oraz aktualizacji odpowiednio przekonwertowanego dokumentu na serwerze WWW (gdzie wzory musiały występować w postaci plików graficznych, co w praktyce uniemożliwiało ich późniejszą obróbkę).

Niestety, jak dotąd żadna z dwóch najpopularniejszych przeglądarek internetowych nie obsługuje tego standardu, co niewątpliwie wynika ze złożoności zagadnienia; rozwiązania firm trzecich są natomiast niedopracowane, mało dostępne i przez to bardzo mało rozpowszechnione. Do ich zasadniczych wad można zaliczyć niedostępność niektórych symboli, zależność od systemu operacyjnego, ograniczenie rozdzielczości wydruku do parametrów ekranu (a więc nie lepiej niż przy zastosowaniu plików GIF), brak interakcji z resztą dokumentu i przeglądarką WWW (m.in. brak wyrównywania do linii bazowej tekstu) i wreszcie konieczność otaczania wzorów dodatkowym kodem HTML wywołującym odpowiedni moduł plug-in [6]. Przykład wzoru zapisanego w języku MathML zawiera tabela 1.

Tabela 1

Przykładowy wzór matematyczny

Kodowanie w języku MathML	Wizualizacja w programie *Edytor wzorów*
<mrow> <mi>y</mi> <mo>=</mo> <mfrac> <mrow> <mn>2</mn> <mfrac> <mn>1</mn> <mn>2</mn> </mfrac> <mo>&Invisibletimes;</mo> <mi>b</mi> <mo>+</mo> <mfenced> <mn>2</mn> <mo>&Invisibletimes;</mo> <mi>a</mi> <mo>-</mo> <mn>1</mn> </mfenced> </mrow> <mrow> <mn>3</mn> <mo>&Invisibletimes;</mo> <msqrt> <mi>a</mi> </msqrt> </mrow> </mfrac> </mrow>	$y=\frac{\frac{1}{2}-b+\left(2a-1\right)}{3\sqrt{a}}$

Zapis za pomocą języka MathML jest obszerniejszy niż za pomocą samego HTML, lecz jest to i tak mało w porównaniu z formatami graficznymi [5].

Tabela 2 zawiera porównanie niektórych formatów używanych do zapisu wzorów matematycznych dla publikacji internetowych. Wynika z niej wyraźna przewaga starego standardu LaTeX pod względem zwięzłości zapisu oraz potencjalna przewaga MathML, jeśli chodzi o jakość wizualizacji w przeglądarkach internetowych.

Tabela 2

Formaty zapisów wzorów matematycznych używane w publikacjach internetowych (dla porównania umieszczono również format LaTeX). Źródło: [5]

Format	Rozmiar przykładowego wzoru (w bajtach)	Możliwość odtworzenia struktury wzoru	Jakość wizualizacji/wydruku w przeglądarce internetowej (1-5)
HTML (PRE)	122	–	2/1
MSWord eksport do GIF	1210	–	4/3
Edytor wzorów eksport do GIF	835	+	4/3
MathML	387	+	5/5[1]
LaTeX	53	+	n/d[2]

Ciągle pozostaje jednak problem opracowania odpowiedniego oprogramowania rozszerzającego możliwości przeglądarek internetowych o pełną i bezproblemową interpretację wzorów zapisanych w MathML. Jest to zadanie na najbliższą przyszłość i jego rozwiązanie z pewnością sprawi, że publikacja tekstów technicznych i matematycznych będzie równie prosta i efektywna jak innych dokumentów.

2.1. Zadania związane z wdrożeniem standardu MathML

MathML jest pierwszą implementacją standardu XML zatwierdzoną przez Konsorcjum WWW i z tego względu powodzenie tego projektu jest uznawane za swego rodzaju test samej idei XML. W chwili obecnej trwają intensywne prace nad udostępnieniem MathML szerokiemu gronu użytkowników Internetu. Projekty te można podzielić na kilka obszarów:

1. wdrożenie języka MathML w przeglądarkach internetowych za pomocą:
   - modułów plug-in,
   - rozszerzenia możliwości interpretacyjnych samych przeglądarek;
2. stworzenie oprogramowania umożliwiającego łatwą edycję wzorów zapisanych w standardzie MathML
   - adaptacja istniejących edytorów w zakresie zapisu i odczytu wyrażeń matematycznych zapisanych w MathML,
   - edytory zbudowane na podstawie standardu MathML;
3. opracowanie narzędzi do konwersji z powszechnie akceptowanego w środowisku naukowym standardu TeX/LaTeX na język MathML i odwrotnie;
4. opracowanie oprogramowania umożliwiającego sprawdzanie poprawności zapisu wzorów w formacie MathML, w celu uniknięcia forsowania rozwiązań firmowych niezgodnych ze

[1] w chwili obecnej niedostępne

[2] nie ma jak dotąd możliwości bezpośredniej wizualizacji dokumentów zapisanych w standardzie LaTeX w przeglądarkach internetowych

standardem, jak to miało miejsce w przypadku HTML (oferowany przez firmę Geometry Technologies pakiet WebEQ już niestety takie rozwiązania zawiera, lecz są one stosunkowo mało istotne – dotyczą bowiem tylko kilku specyficznych parametrów), np. http://www.w3c.org/math/validator.

Jednym z najbardziej oczywistych pytań, które może się w tym miejscu pojawić, jest pytanie o sens tworzenia nowego standardu, w sytuacji gdy od lat istnieje uznany standard opisu wyrażeń matematycznych — LaTeX. Podstawową wadą systemu LaTeX w zastosowaniach dla WWW jest niezgodność ze standardem XML (a więc obrazowo mówiąc „małe podobieństwo do HTML"). Natomiast MathML jest w pełni oparty na wytycznych XML, co daje klarowny, łatwy do interpretacji i weryfikacji kod. Jednocześnie jednak MathML zachowuje pewne cechy (w postaci nazw symboli, liter greckich itp.) standardu LaTeX. Z założenia rozbudowa drzewa interpretacji kodu HTML w przeglądarce WWW o znaczniki MathML powinna być stosunkowo mało skomplikowana, czego nie można powiedzieć o kodzie w języku LaTeX. Niestety, również MathML w praktyce wymaga specjalnego podejścia w końcowym etapie wizualizacji wzoru [9]. Standardowe podejście wykorzystujące do przedstawiania dokumentów XML za pomocą języka opisu stylu prezentacji stron CSS (Cascading Style Sheets) pozwala na poprawne wyświetlanie tylko niektórych znaczników.

Podobieństwo do uznanego języka LaTeX przy formalnym oparciu na standardzie XML jest też istotne z innego punktu widzenia. Wielu bowiem użytkowników darzy LaTeX ogromnym zaufaniem opartym na latach doświadczeń z tym stabilnym systemem generującym wysokiej jakości dokumenty. Proste i jasne zasady translacji z LaTeX do MathML i odwrotnie, czemu niewątpliwie sprzyja podobieństwo wielu oznaczeń, dają nadzieję na stworzenie w krótkim czasie sprawnych narzędzi – konwerterów, które zwiększą akceptację dla standardu MathML w obszarach zdominowanych dotychczas przez LaTeX oraz rozwiązania firmowe (np. Mathematica, Scientific Workplace).

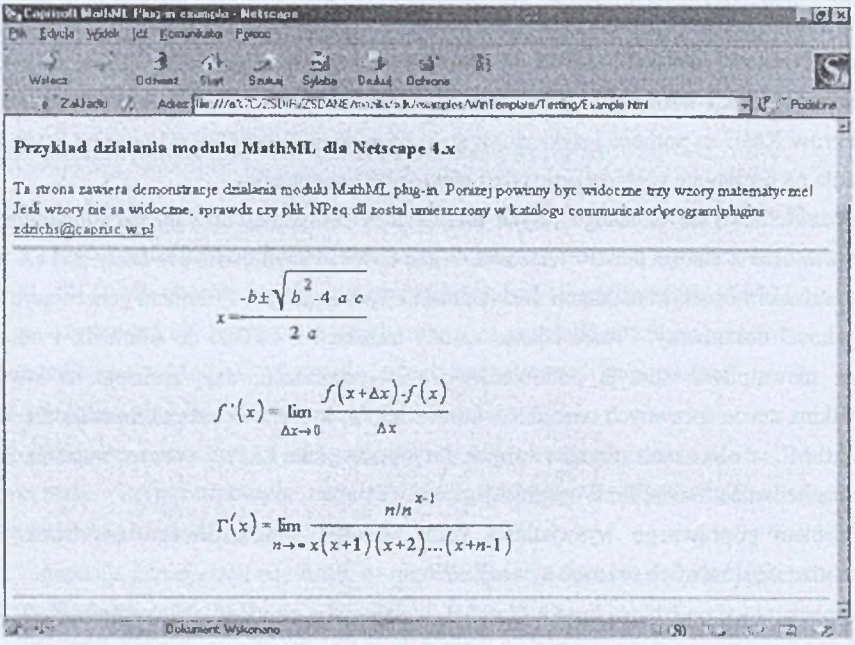
Problem poprawnego wyświetlania kodu MathML umieszczonego na stronie WWW sprowadza się do dwóch różnych sytuacji:

- implementacji podzbioru języka MathML w najnowszych wersjach przeglądarek WWW,
- oglądania wzorów w starszych wersjach najpopularniejszych przeglądarek, jak również w mniej popularnych przeglądarkach (np. Opera, przeglądarki dostępne dla mniej popularnych systemów, takich jak BeOS, AmigaOS).

Drugie zagadnienie jest niezwykle istotne ze względu na ciągle dużą popularność przeglądarek MS Internet Explorer w wersjach 4 i 5 oraz Netscape Navigator w wersjach 4.x, które nie zapewniają wsparcia dla MathML. Trudno byłoby wyobrazić sobie powodzenie standardu, który byłby dostępny jedynie dla wąskiego grona użytkowników najnowszych wersji najpopularniejszych przeglądarek. Trudność w upowszechnieniu standardu MathML może sprawić

również postawa firmy Microsoft, która ze względu na pośpiech przedkłada jak dotąd zgodność swojego pakietu Office z przeglądarką Internet Explorer nad wdrażanie standardu MathML. Stąd moduł Edytor równań w Office 2000 przy zapisie w formacie HTML generuje opis wzoru w postaci pewnego podzbioru wektorowego języka SVG (Scalable Vector Graphic) a nie MathML [6].

W każdym razie stworzenie odpowiednich modułów *plug-in* dla obu najpopularniejszych przeglądarek w wersjach od 4 wzwyż rozwiązałoby niewątpliwie wiele podobnych dylematów. Ważne wydaje się przy tym, aby były to dedykowane szybkie moduły przygotowane dla konkretnych systemów operacyjnych. Nasuwające się w takich zastosowaniach niezależne od platformy aplety Javy są bowiem zbyt mało wydajne do sprawnej wizualizacji złożonych dokumentów (szczególnie za pomocą starszych komputerów, których użytkownicy mogą przecież bez problemów efektywnie wykorzystywać system LaTeX).

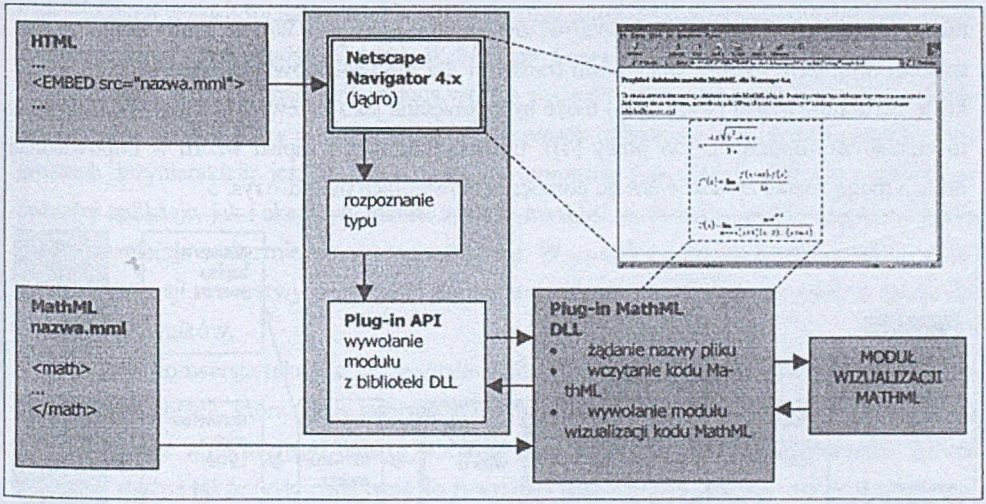


Rys. 3. Przykład działania modułu plug-in dla Netscape (Sylaba) Navigator 4.x wyświetlającego wzory zapisane w języku MathML. Wywołanie modułu plug-in uzyskano za pomocą znacznika EMBED

Fig. 3. Example of MathML plug-in for Netscape Navigator 4.x. Plug-in called with EMBED tag

Niestety, w celu wywołania odpowiedniego modułu plug-in strona WWW musi zawierać specjalne wstawki do kodu HTML (znaczniki OBJECT lub EMBED). Strony WWW, które wyglądałyby poprawnie zarówno po wizualizacji w najnowszych przeglądarkach, jak i w ich

starszych wersjach z wykorzystaniem *plug-in*, są w związku z tym trudniejsze do zaprojektowania. W efekcie może dojść do sytuacji podobnej do obecnie spotykanego dualizmu przy projektowaniu stron DHTML (Dynamic HTML trzeba w wielu przypadkach kodować osobno dla Nawigatora i MSIE), w której dobry projektant stron WWW będzie musiał utrzymywać dwa zestawy dokumentów. Trudno więc będzie przekonać do języka MathML autorów wykorzystujących sprawdzone narzędzia zapisujące wzory w postaci zbiorów graficznych GIF [5].

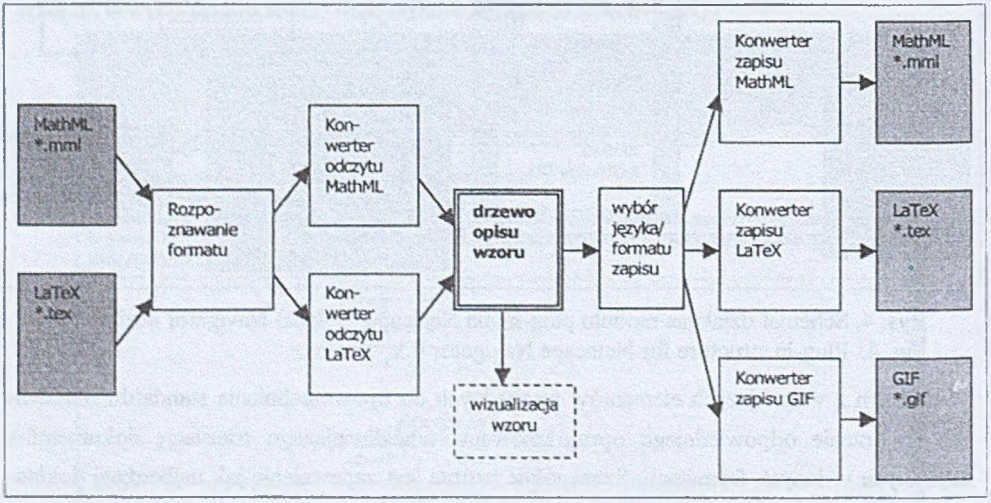


Rys. 4. Schemat działania modułu plug-in dla Netscape (Sylaba) Navigator 4.x
Fig. 4. Plug-in structure for Netscape Navigator 4.x

Jednym z ważniejszych elementów niezbędnych do upowszechnienia standardu MathML jest stworzenie odpowiedniego oprogramowania umożliwiającego translację dokumentów zapisanych w innych formatach. Szczególnie istotne jest zapewnienie jak najbardziej dokładnego tłumaczenia dokumentów zapisanych w LaTeX ze względu na dużą popularność tego systemu w środowisku naukowym i akademickim. Pewne kłopoty może przy tym sprawić duża elastyczność i tolerancja systemu LaTeX na błędy. Dokumenty LaTeX mogą zawierać makrodefinicje, których jak dotąd brak w definicji MathML. Rozwiązaniem mogłoby być rozszerzenie standardu MathML lub każdorazowe zastępowanie wywołania makrodefinicji jej treścią (co i tak może okazać się niewykonalne, np. dla pętli). System LaTeX potrafi ponadto niejako automatycznie poprawiać niektóre błędy użytkownika, generując poprawnie wyglądające dokumenty. Z pewnością wiele jest takich nie do końca poprawnych dokumentów w licznych archiwach dostępnych poprzez Internet. W przypadku młodego standardu MathML, dla którego nie zostały dotychczas nawet opracowane zasady wzorcowej wizualizacji, trudno o równie wyrafinowaną korekcję błędów. Z drugiej strony prostota i konsekwencja zapisu w MathML, wynikające ze zgodności z XML (a więc z odpowiedniej DTD – Document Type

Definition), umożliwiają stosunkowo prostą weryfikację podstawowych błędów. Nie zmienia to jednak faktu, iż nieprawidłowy dokument w LaTeX może po mechanicznym przetłumaczeniu na MathML zawierać bardzo trudne do zlokalizowania i poprawienia błędy.

Prostsze wydaje się opracowanie translatora dokumentów zapisanych w MathML na inne języki, w tym oczywiście LaTeX. Wynika to z prostoty i konsekwencji samego MathML oraz możliwości zastosowania makrodefinicji LaTeX do konwersji bardziej rozbudowanych, nie mających bezpośrednich odpowiedników, znaczników MathML. W celu zapewnienia uniwersalności oraz łatwej konserwacji kodu translator dla MathML powinien mieć budowę modułową. Struktura wzoru (wyrażenia) może być pamiętana jako drzewo obiektów (podobnie jak to zostało zaproponowane w pracy [4]). Operacje odczytu i zapisu wzoru w odpowiedniej notacji mogą zostać zrealizowane za pomocą odpowiednich metod – rys. 5.



Rys. 5. Schemat blokowy uniwersalnego konwertera dla języka MathML
Fig. 5. Block scheme of an universal converter for MathML

Istnieje możliwość stosunkowo łatwego rozpoznania formatu wczytywanego dokumentu za pomocą charakterystycznych ciągów znaków występujących w zapisie LaTeX czy MathML[1]. Pozwala to na zrealizowanie konwertera automatycznie rozpoznającego typ wczytywanego pliku (i ewentualnie wyświetlającego przetworzony wzór), a więc bardzo wygodnego dla użytkownika. Właściwą konwersję z MathML do LaTeX da się natomiast sprowadzić do ciągów operacji tekstowych typu „znajdź–zamień". Wizualizacja wzoru przed właściwą konwersją wymaga jednak przetworzenia wzoru do postaci drzewa obiektów, wykorzystywanego przez moduł wyświetlający. Natomiast konwersja z LaTeX do MathML to nie-

[1] każdy dokument zapisany w MathML powinien ze względów formalnych zawierać podciąg znaków „<math>", a praktycznie musi zawierać podciąg „<mrow>"

co bardziej złożone zadanie, wymagające zapamiętania przetworzonych w danej chwili znaczników tak, aby można było wygenerować we właściwych miejscach powstającego kodu MathML odpowiadające im zamknięcia bądź otwarcia.

Dodatkowym wyzwaniem przy projektowaniu konwertera dla systemu MathML może okazać się potrzeba zapewnienia tłumaczenia między dwoma warstwami składowymi samego MathML. Warstwa prezentacji, w założeniu mająca wyłącznie ułatwić wizualizację wzorów, z oczywistych względów najszybciej doczeka się implementacji (pewien jej podzbiór już oferują takie programy, jak *Amaya*, *EzMath*, *WebEQ* czy *Formula Wizard*[1]).

Natomiast warstwa struktury, służąca według twórców języka MathML do zapisu znaczenia wzoru w sposób łatwy do interpretacji w systemach obliczeniowych, symulacyjnych i programach inżynierskich, jest na razie wyraźnie pomijana (jak dotąd zarówno wymienione powyżej aplikacje, jak i eksperymentalna obsługa MathML w Mozilli – publicznej wersji Navigatora, wspierają wyłącznie warstwę prezentacji). W związku z tym narzędzie potrafiące dokonać konwersji z warstwy prezentacji do warstwy struktury byłoby z pewnością cenne dla wielu użytkowników.

Niestety, konwersja taka jest szczególnie trudna do zrealizowania ze względu na charakter warstwy prezentacji, która niesie informację jedynie o wyglądzie wzoru. W wielu przypadkach zapisy podobne z graficznego punktu widzenia mogą mieć zupełnie różne znaczenie; pewne znaczniki można także wykorzystywać do tworzenia zupełnie odmiennych znaczeniowo wyrażeń (np. indeksy mogą być wykorzystywane do zapisywania całek). Natomiast konwersja z warstwy struktury do warstwy prezentacji może być bez wątpienia wykonana jednoznacznie, co zresztą przewiduje specyfikacja W3C Consortium [1].

Równie istotna może okazać się konwersja do formatów graficznych (np. GIF, JPG, PNG). Wynika to z faktu istnienia w specyfikacji MathML parametru ALTIMG, który w założeniu jest przeznaczony dla przeglądarek, których twórcy byli świadomi istnienia MathML, ale z pewnych względów nie mogli zapewnić jego interpretacji. Takie przeglądarki mogą wyświetlać wzory korzystając z pliku graficznego wskazywanego przez wspomniany znacznik. Oczywiście sytuacja ta wymaga kolejnych, specjalnych narzędzi i dodatkowego wysiłku ze strony projektanta stron WWW. Jest to kolejny argument wskazujący na niezwykle istotne znaczenie programów dokonujących translacji pomiędzy różnymi formatami.

2.2. Warstwa prezentacji MathML

Warstwa prezentacji języka MathML jest najważniejszym jego podzbiorem z racji praktycznej przydatności dla ogółu użytkowników. Warstwa ta zostanie w pierwszej kolejności

[1] http://indy.cs.concordia.ca/mathml/authoring/index.html

zaimplementowana w przeglądarkach i różnego rodzaju aplikacjach narzędziowych. Dla wielu zresztą zastosowań używanie drugiej warstwy – struktury – nie jest w ogóle konieczne.

Większość znaczników MathML ma dobrze znany z HTML format:

```
<znacznik>wnętrze</znacznik>
```

przy czym nie przewiduje się znaczników z opcjonalnym domknięciem. Pewna ilość znaczników jest natomiast pusta i ma postać:

```
<znacznik/>
```

znaczniki te jednak występują głównie w warstwie struktury.

Znacznik może posiadać parametry i wówczas jego postać jest następująca:

```
<znacznik par1='treść' par2="treść">wnętrze</znacznik>
```

W większości przypadków (poza nawiasami) parametry mają znaczenie drugorzędne (najczęściej związane z precyzyjnym ustalaniem wzajemnych proporcji i położenia fragmentów wzoru) i w praktyce mogą zostać pominięte.

Każdy znacznik może zawierać kolejne znaczniki, o ile wynika to z jego zastosowania. Standard precyzyjnie określa, ile elementów potomnych powinno znajdować się wewnątrz każdego znacznika. Przykładowo, znacznik <mfrac> (ułamek) powinien zawierać dokładnie dwa znaczniki, opisujące oczywiście licznik i mianownik, natomiast znacznik <mrow> może zawierać dowolną liczbę znaczników, które zostaną ułożone obok siebie na tej samej wysokości względem linii bazowej. Znacznikiem charakterystycznym MathML jest znacznik <math>, który powinien otaczać każdy poprawny zapis w tym języku (również np. umieszczony w schowku). Znacznik ten może wystąpić wyłącznie na zewnątrz wyrażenia, stąd oprogramowanie edytujące wzory zapisane w MathML powinno dbać o dodawanie go przy zapisie wzoru i usuwanie przy odczycie (a w szczególności przy wklejaniu ze schowka).

Szczególną cechą MathML jest zależność od kontekstu i zawartości znaczników [9]. Takie podejście daje możliwość poprawnego interpretowania oznaczeń zmiennych i tekstów (np. zawartość znacznika <mi> jest wyświetlana w kursywie tylko dla pojedynczych znaków). Innym przykładem jest wyświetlanie zawartości znacznika <mo> za pomocą słownika operatorów. Dla każdego operatora słownik przechowuje informacje o sposobie jego wizualizacji łącznie z odstępami i wielkością.

Niektóre znaczniki, takie jak np. <mfenced> (otoczenie nawiasami) są w standardzie określone jako równoważne bardziej rozbudowanym konstrukcjom złożonym z kilku innych znaczników (tabela 3).

Tabela 3

Zapis z nawiasami z wykorzystaniem znacznika <mfenced> i bez. Źródło: [1]

Zapis „(x)" za pomocą <mfenced>	Zapis „(x)" za pomocą <mo>

`<mfenced>`
`  <mi>x</mi>`
`</mfenced>`
	`<mrow>`
	`  <mo> ( </mo>`
	`  <mi>x</mi>`
	`  <mo> ) </mo>`
	`</mrow>`

Niestety, możliwość taka znacznie utrudnia zarówno interpretację, jak i weryfikację tak zakodowanych wyrażeń, dodatkowo zmniejszając czytelność zapisu. Z tego względu pożądane byłoby usunięcie tej możliwości w kolejnych wersjach MathML.

Kolejnym elementem zapisu w języku MathML są stałe. Format stałej:

```
&wartość;
```

wykorzystuje jako wartości literały znane ze standardu LaTeX.

Stąd na przykład zapis \alpha znany z LaTeX w języku MathML będzie wyglądał następująco: α przy czym powinien być otoczony znacznikiem <mi> — „identyfikator":

```
<mi>&alpha;</mi>
```

Jak widać, zapis w MathML jest znacznie mniej oszczędny niż w LaTeX, ale zgodnie z założeniami jego twórców jest on przeznaczony raczej do przetwarzania przez specjalistyczne oprogramowanie, a nie przez człowieka [1]. Stąd też wada ta nie ma zbyt wielkiego znaczenia.

Najważniejsze znaczniki warstwy prezentacji MathML zawiera tabela 4. Szczególnie ciekawy jest znacznik <maction>, który pozwala na konstruowanie dokumentów matematycznych reagujących interaktywnie na pewne działania odbiorcy. Dynamika wzorów zapisanych w MathML może polegać na: zmianie koloru pisaka bądź tła przy najechaniu myszką, wyświetleniu napisu w pasku statusu, zadziałaniu jak typowy odsyłacz HTML lub zamianie jednego wyrażenia na drugie.

Większe możliwości daje dopiero perspektywa języka makrodefinicji dla standardu MathML. Najważniejsze potencjalne zastosowania takiego języka (przewidywanego zresztą w specyfikacji) mogą obejmować [1]:

- skrócenie zapisu, szczególnie przy długich powtarzających się fragmentach wzorów,
- rozszerzenie warstwy struktury,
- kontrolę nad wyglądem wyświetlanego wzoru,
- rozszerzenia dla osób niepełnosprawnych, pozwalające na dodatkową kontekstową interpretację zapisu w MathML w celu wygenerowania słownego opisu zamiast rysunku.

Tabela 4

Podstawowe znaczniki MathML (warstwa prezentacji). Źródło: [1]

Znacznik MathML		zastosowanie
grupa	nazwa	
podstawowe	<mi>	identyfikator
	<mn>	liczba
	<mo>	operator
	<mtext>	tekst
	<mspace/>	odstęp
	<ms>	literał
proste wyrażenia	<mrow>	grupa podwyrażeń ułożona poziomo
	<mfrac>	ułamek
	<msqrt> <mroot>	pierwiastek
	<mfenced>	otoczenie nawiasami
	<mstyle>	zmiana stylu
indeksy i granice	<msub>	indeks dolny
	<msup>	indeks górny
	<msubsup>	indeks górny i dolny
	<munder>	indeks pod
	<mover>	indeks nad
	<munderover>	indeksy nad i pod wyrażeniem, używane do zapisywania sum, całek itp.
macierze	<mtable>	macierz
	<mtr>	wiersz macierzy
	<mtd>	komórka macierzy
interakcje	<maction>	interakcyjne działania związane z wyrażeniem np. odsyłacze, zmiana koloru.

2.3. Warstwa prezentacji, warstwa struktury i inne formaty zapisu w jednym dokumencie

W wielu sytuacjach wykorzystanie odpowiedniej dla zadania warstwy MathML z pewnością okaże się zupełnie wystarczające. Mogą jednak zdarzyć się zapisy niejednoznaczne, bądź trudne do zinterpretowania w warstwie prezentacji, wymagające osobnego zakodowania w warstwie struktury tak, aby było możliwe ich późniejsze

przetwarzanie. Z drugiej strony czasem wzór zapisany wyłącznie w warstwie struktury może wymagać również opisania za pomocą warstwy prezentacji, by jego wizualizacja była w pełni zgodna z intencjami autora. Autor dokumentu MathML może również przewidywać konieczność wykorzystania dodatkowego oprogramowania i stąd pojawić się może konieczność dodania do zapisu w MathML kodowania w zupełnie innym standardzie (np. Mathematica, Maple, TeX). Aby wyeliminować konieczność dokonywania konwersji w takich przypadkach, co niejednokrotnie byłoby kłopotliwe lub nieprecyzyjne, standard MathML przewiduje specjalny znacznik <semantics> [1].

Znacznik <semantics> oczekuje elementu pochodnego, zawierającego podstawowe kodowanie wzoru (w warstwie prezentacji lub struktury) oraz pewną ilość znaczników <annotation> lub <annotation–xml>. Znacznik <anotation> może zawierać opis w dowolnej notacji, natomiast <annotation–xml> przeznaczony jest do wstawiania zapisu zgodnego z wytycznymi XML. Obydwa te znaczniki powinny być uzupełnione parametrem encoding, określającym typ zawartości. Przykładowe zapisy wykorzystujące znacznik <semantics> zawiera tabela 5.

Za pomocą znacznika <annotation> można uzupełnić zapis wzoru o opis w dowolnej notacji, na przykład wykorzystywanej przez program tworzący wykresy. Przykład takiego zastosowania zawiera tabela 6.

Tabela 5

Łączenie kodowania za pomocą znaczników warstwy struktury i prezentacji MathML

Łączenie warstwy struktury z warstwą prezentacji $f'(x)$	
<semantics> <mrow> <msup> <mi>f</mi> <mi>'</mi> </msup> <mfenced open='(' close=')'> <mi>x</mi> </mfenced> </mrow> <annotation-xml encoding="MathML-Content"> <apply> <diff/> <ci>f</ci> <bvar><ci>x</ci></bvar> </apply> </annotation-xml> </semantics>	<semantics> <apply> <diff/> <ci>f</ci> <bvar><ci>x</ci></bvar> </apply> <annotation-xml encoding="MathML-Presentation"> <mrow> <msup> <mi>f</mi> <mi>'</mi> </msup> <mfenced open='(' close=')'> <mi>x</mi> </mfenced> </mrow> </annotation-xml> </semantics>

Tabela 6

Łączenie kodowania za pomocą warstwy struktury MathML i innej notacji, niezgodnej z XML [3]

Łączenie warstwy struktury z opisem w dowolnej notacji $\{x,y,z \mid x^2+y^2+z^2=1\}$

```
<semantics>

<set>
  <bvar>  <ci>x</ci>  </bvar>
  <bvar>  <ci>y</ci>  </bvar>
  <bvar>  <ci>z</ci>  </bvar>
  <condition>
   <reln> <eq/>
    <apply> <plus/>
      <apply> <power/> <ci>x</ci> <ci>2</ci> </apply>
      <apply> <power/> <ci>y</ci> <ci>2</ci> </apply>
      <apply> <power/> <ci>z</ci> <ci>2</ci> </apply>
    </apply>
    <cn>1</cn>
   </reln>
</condition>
</set>

<annotation encoding='oogl'>
SPHERE
1
0 0 0
</annotation>

</semantics>
```

Najbardziej znanym formatem, który może pojawiać się w konstrukcjach wykorzystujących znacznik <semantics>, jest LaTeX – dotąd powszechnie wykorzystywany standard zapisu dokumentów matematycznych i inżynierskich. Nie byłoby to jednak zjawisko pożądane. LaTeX jest bowiem bardzo odległy od idei HTML czy XML i jego wizualizacja wymagałaby każdorazowo użycia specjalnego modułu rozszerzającego możliwości przeglądarki.

2.4. Przykłady wzorów

Mimo braku kompletnych i sprawdzonych aplikacji wykorzystujących standard MathML do zapisu wzorów matematycznych można pokusić się o przedstawienie przykładowych konstrukcji języka MathML. Może się jednak okazać, że wizualizacja konkretnych znaczników będzie się zmieniać wraz z rozwojem edytorów i przeglądarek akceptujących MathML nie tylko jako format do eksportu danych, ale jako podstawowy format zapisu dokumentu.

Tabela 7

Przykładowy wzór matematyczny

Kodowanie w języku MathML	Wizualizacja w programie *Edytor wzorów*

```
<mrow>
 <munderover><mo>int</mo>
  <mi>a</mi>
  <mo>&infty;</mo>
 </munderover>
 <mrow>
  <mi>f</mi>
  <mfenced open='(' close=')'>
   <mi>x</mi>
  </mfenced>
  <mi>dx</mi>
```

$$\int_a^{\infty} f(x)\,dx = \lim_{B\to\infty} \int_a^{B} f(x)\,dx$$

```
  <mo>=</mo>
  <munderover><mo>lim</mo>
  <mrow>
   <mi>B</mi>
   <mo>&rightarrow;</mo>
   <mo>&infty;</mo>
  </mrow>
  </munderover>
  <mrow>
   <munderover><mo>int</mo>
    <mi>a</mi>
    <mi>B</mi>
   </munderover>
   <mrow>
    <mi>f</mi>
    <mfenced open='(' close=')'>
     <mi>x</mi>
    </mfenced>
    <mi>dx</mi>
   </mrow>
  </mrow>
 </mrow>
</mrow>
```

Powyższy przykład ilustruje elastyczność warstwy prezentacji języka MathML. Jak widać, stosunkowo łatwo można w nim zapisywać nawet złożone dokumenty. Jest to jednak przykład statyczny, podobnie jak dokumenty w innych systemach składu. Największą potencjalną zaletą dokumentów wykorzystujących MathML do przedstawienia różnego rodzaju wzorów jest natomiast interakcja z odbiorcą, podobna do tej, która występuje podczas oglądania dokumentów WWW. Może ona polegać na różnorodnych reakcjach na najechanie kursorem myszki na określony obszar wzoru.

Tabela 8

Przykładowy wzór matematyczny

Kodowanie w języku MathML	Wizualizacja w programie *Edytor wzorów*

`<mrow>`
`<msup>`
`<mi>f</mi>`
`<mo>'</mo>`
`</msup>`
`<mfenced open='(' close=')'>`
`<mrow>`
`<mi>x</mi>`
`</mrow>`
`</mfenced>`
`<mo>=</mo>`
`<munderover><mo>lim</mo>`
`<mrow>`
`<mo>&Delta;</mo>`
`<mi>x</mi>`
`<mo>&rightarrow;</mo>`
`<mn>0</mn>`
`</mrow>`
`</munderover>`
`<mrow>`
`<mfrac>`
`<mrow>`
`<mi>f</mi>`
`<mfenced open='(' close=')'>`
`<mrow>`
`<mi>x+</mi>`
`<mo>&Delta;</mo>`
`<mi>x</mi>`
	$f'(x) = \lim_{\Delta x \to 0} \frac{f(x+\Delta x) - f(x)}{\Delta x}$

```
    </mrow>
    </mfenced>
    <mo>-</mo>
    <mi>f</mi>
    <mfenced open='(' close=')'>
    <mrow>
     <mi>x</mi>
    </mrow>
    </mfenced>
   </mrow>
   <mrow>
    <mo>&Delta;</mo>
    <mi>x</mi>
   </mrow>
   </mfrac>
  </mrow>
</mrow>
```

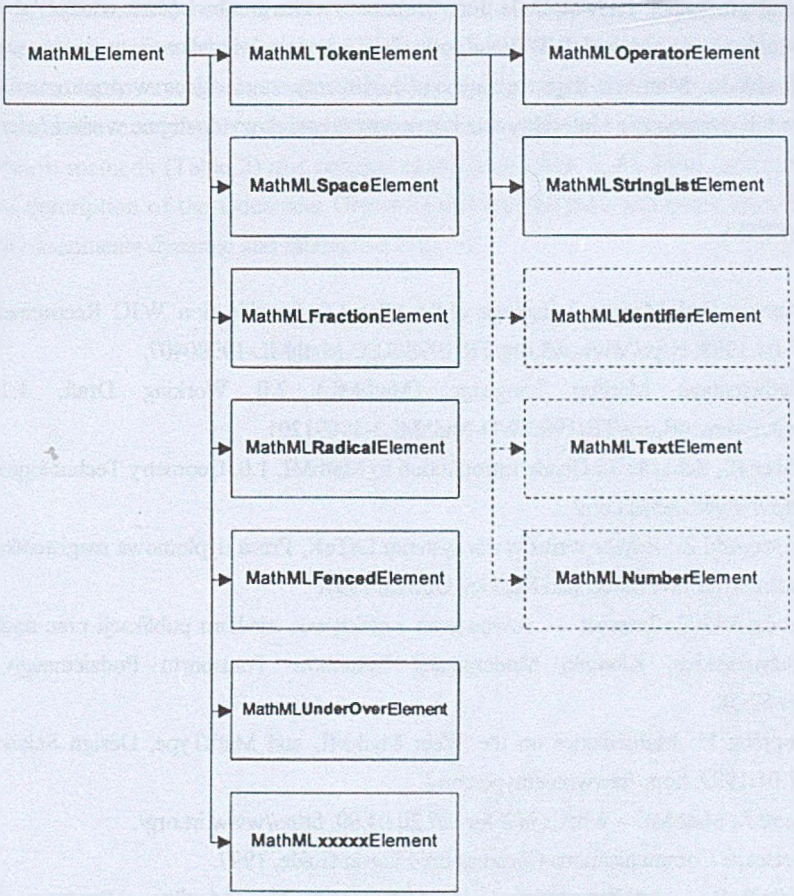
Obszar taki może działać jak odsyłacz (*link*) w dokumencie HTML, może zmieniać swój kolor, może wreszcie zmienić wizualizowany wzór. Wszystkie te opcje dają wiele możliwości uatrakcyjnienia dokumentu. Można w prosty sposób zapewnić przejście do opisu zastosowanych w konkretnym momencie praw czy twierdzeń, można wyróżniać najważniejsze fragmenty dokumentu, można wreszcie przedstawić kolejne kroki przekształceń czy też równorzędne drogi rozumowania.

Jeszcze dalej posuniętym rozwinięciem tej idei może być zintegrowanie z przeglądarką edytora wzorów tak, aby odbiorca dokumentu mógł w każdej chwili dokonać żądanych poprawek lub przeprowadzić odpowiednie przekształcenia. Jedynym rozwiązaniem tego rodzaju w chwili obecnej jest przeglądarka Amaya (patrz rys. 1) opracowana przez konsorcjum WWW.

3. Document Object Model dla MathML

Niezmiernie ważnym uzupełnieniem standardu MathML jest załączony w proponowanej drugiej wersji tego języka model obiektowy (DOM). Specyfikacja ta może pełnić istotną rolę w zastosowaniach wykorzystujących języki skryptowe oraz podczas tworzenia modułów rozszerzeń (plug-in) dla przeglądarek. Dokumenty MathML wzbogacone o skrypty mogą zapewniać o wiele bardziej rozbudowaną interakcję, niż to wynika z możliwości znaczników MathML. Ponadto skrypty pozwoliłyby połączyć wyrażenia matematyczne z resztą dokumentu internetowego, a więc formularzami, ramkami oraz wszelkiego rodzaju grafiką. Moduły plug-in mające dostęp do drzewa obiektów MathML mogłyby zapewniać współpracę przeglądarki internetowej z oprogramowaniem matematycznym i inżynierskim w zakresie obliczania, przekształcania i modelowania wzorów zapisanych na stronie internetowej.

Proponowana przez W3C Consortium druga wersja standardu MathML [2] zawiera wstępną (ang. non-normative) propozycję DOM. Zarys tej specyfikacji przedstawia rys. 6.



Rys. 6. DOM dla języka MathML. Przerywaną linią oznaczono obiekty świadomie jak dotąd pomijane przez projektantów. XXXXX – Style, Padded, Table, tableRow, TableCell, Action, Script, Multiscripts

Fig. 6. DOM for MathML. The objects not included in W3C Consortium project marked with slashed line. XXXXX – Style, Padded, Table, tableRow, TableCell, Action, Script, Multiscripts

4. Podsumowanie

Przedstawiona w niniejszym opracowaniu idea języka MathML daje prawdziwą szansę wprowadzenia dokumentów matematycznych do powszechnie dostępnej sieci WWW. Ze względu na specyfikę HTML opis wyrażeń matematycznych na potrzeby Internetu musi się

różnić od znanego ze standardu LaTeX, nie może również na dłuższą metę wykorzystywać reprezentacji graficznej. Niewątpliwie dużo jeszcze wysiłku trzeba będzie włożyć, aby dokumenty matematyczne w sieci WWW odpowiadały jakością wizualizacji w zaawansowanych systemach składu. MathML daje nadzieję na dokumenty zawierające wzory matematyczne, a przy tym tak dynamiczne i interaktywne jak pozostałe przekazy dostępne w sieci Internet.

LITERATURA

1. Mathematical Markup Language (MathML) 1.0 Specification W3C Recommendation, 07.04.1998, http://www.w3.org/TR/1998/REC-MathML-19980407.
2. Mathematical Markup Language (MathML) 2.0 Working Draft, 1.12.1999, http://www.w3.org/TR/1999/WD-MathML2-19991201.
3. Miner R., Schaefer J.: Gentle Introduction to MathML 1.0, Geometry Technologies, 1999, http://www.webeq.com/.
4. Sroczyński Z.: Edytor wzorów dla systemu LaTeX, Praca dyplomowa magisterska. Instytut Informatyki Politechniki Śląskiej, Gliwice 1997.
5. Sroczyński Z.: Internet — nowoczesne i efektywne medium publikacji prac naukowych i inżynierskich, Kierunki Modernizacji Systemów Transportu Podziemnego, 1999, str. 52-58.
6. Topping P.: Mathematics on the Web: MathML and MathType, Design Science, Inc., 21.01.1999, http://www.mathtype.com/.
7. Boye J.: MathML – What's in it for us? 20.04.99, http://www.irt.org/.
8. Netscape Communications Corporation, Plug-in Guide, 1997.
9. Sidje R. B.: Implementation of MathML in Mozilla: Progress Report http://www.mozilla.org/projects/mathml/update.html, 18.09.1999.

Recenzent: Prof. dr hab. inż. Tadeusz Czachórski

Wpłynęło do Redakcji 5 kwietnia 2000 r.

Abstract

Serving, receiving and processing documents containing mathematical notation was supported by many tools for a long time but there was the lack of applications and standards for

the World Wide Web. MathML (Mathematical Markup Language) is an XML application for describing mathematical meaning and structure of the document (Fig. 1). A general description of the standard is presented in the paper (Table 4). Main tasks related to the implementation of MathML standard, ways of conversion (Fig. 5) and cooperation with other systems (Tables 5, 6) have been shown and analysed. Important parts of the paper are the comparison of different visualization methods (Table 2) and original examples (Tables 7, 8). Final part of the article contains description of the Document Object Model for MathML and possibilities of making MathML documents dynamic and interactive (Fig. 6).