Tirème Sarl

MathML, Mathematical Markup Language

Rédaction : Ulric Genièvre, Yannick Litaiz, Wojciech Machocki,

Ludovic Maurillon, Benoît Roger, Sébastien Vallée, **Pierre Attar**

MathML est un langage de représentation des mathématiques. Plus précisément, MathML permet de coder une forme sémantique et/ou typographique d'un objet mathématique. En ce sens, MathML est un compromis entre les défenseurs du "tout sémantique" (le groupe *OpenMath*, par exemple) et le "tout typographique" (la communauté *TeX* et *LaTeX*).

La partie typographique de MathML se justifie par la spécificité de la notation typographique mathématique où prime l'ambiguïté, par souci pédagogique aussi bien que par souci de densification de l'information. Ainsi, un article de mathématique parlant de matrices n'aurait aucun intérêt à toujours coder la sémantique sous-jacente ! En effet, quel effort de saisie faudrait-il alors mobiliser pour seulement parler de "*la matrice **M** précédemment définie*" ! Pourtant, *M* est bien un objet mathématique ou, plutôt, un objet typographique mathématique.

L'enjeu de MathML est sa reconnaissance ! Dans sa version 2, la spécification semble stable et pouvoir coder bon nombre de besoins... il reste alors à comprendre comment sera réalisée son implémentation, particulièrement sur le Web. En effet, si, depuis Knuth et son moteur de composition TeX, "écrire sur papier" des mathématiques ne pose aucun problème, il n'en va pas de même sur Internet. Avec MathML, un standard existe maintenant. Pour qu'il puisse être largement utilisé, il faut qu'il soit implémenté dans tous les logiciels de consultation, de manière presque native, du fait de l'étroite inter-relation entre la typographie mathématique et le texte.

Les logiciels *Open Source* peuvent aussi aider au développement de MathML. À ce titre, toutes les explications données ci-après sont extraites d'un mémoire d'étudiants en 5^{e} année de Génie mathématique (*INSA* de Rouen) *[sur internet : ./images/rapportMathML.pdf]*, qui ont pris l'initiative de proposer à **Pierre Attar** la libre disposition de leur mémoire, dans le but de participer à une plus large connaissance de MathML et de ses outils.

Objectifs

MathML est un format de définition de mathématique codé en XML, dont le but est d'identifier à la fois la typographie et/ou le sens mathématique. MathML doit permettre aux mathématiques d'être échangées, reçues et traitées sur le Web, comme HTML l'a permis pour le texte. Parallèlement, il doit permettre à des logiciels de calculs formels comme

Mapple ou *Matlab*, ainsi qu'à des applications classiques développées dans des langages standard (C, Java, LISP, etc.) d'échanger et d'interpréter des objets mathématiques complexes.

Pour ce faire, les objectifs de MathML sont :

- encoder de la substance mathématique adaptée à la communication scientifique et mathématique de tout niveau ;
- encoder la notation typographique et/ou la sémantique sous-jacente ;
- faciliter la conversion de (et vers) d'autres formats de présentation et/ou de sémantique mathématique.

Les formats et applications utilisant MathML doivent aussi permettre de *tirer* de ce langage :

- des représentations graphiques ;
- des synthèses vocales ;
- des données traitables par les systèmes algébriques ;
- d'autres représentations dans d'autres langages mathématiques (e.g. TeX) ;
- des représentations "plain text" ;
- des affichages divers, y compris le braille.

Les formats et applications utilisant MathML doivent encore permettre :

- de supporter les longues expressions à la navigation ;
- le passage d'information voulue pour des "rendeurs" spécifiques et des applications ;
- d'être étendu.

Enfin, les formats et applications utilisant MathML doivent être :

- adaptés aux *templates* et autres techniques d'édition ;
- "humainement" lisibles ;
- faciles à gérer et à traiter par les logiciels.

Le W3C Math Working Group *[sur internet : www.w3.org/Math/]* a identifié des objectifs supplémentaires d'implémentation, dont le but est de décrire rapidement les fonctionnalités élémentaires que les moteurs de "rendu" et les logiciels de traitement MathML devraient fournir :

- les équations dans les pages HTML doivent avoir un rendu correct dans les navigateurs, en accord avec les préférences d'affichage des auteurs et des lecteurs, et avec une qualité optimale au vu des capacités de chaque plate-forme ;
- les pages HTML contenant des équations MathML doivent être imprimer correctement, à la résolution optimale de l'imprimante (*idem* pour le braille, les synthétiseurs vocaux, etc.) ;
- les équations MathML dans une page Web doivent être capables de réagir au comportement de l'utilisateur (clic d'un souris par exemple) et de gérer la communication avec d'autres applications, à travers le navigateur ;
- des éditeurs d'équation et des "convertisseurs" doivent être développés pour faciliter la création de pages Web contenant des équations MathML.

Il est reconnu que la conversion de (ou vers) d'autres systèmes de notation ou médias peut entraîner une perte d'information.

En outre, comme nous l'avons dit, l'un des objectifs de MathML est la diffusion d'information mathématique sur Internet. Par conséquent, MathML se doit d'être en relation avec les autres technologies Internet connues. Il faut plus particulièrement que l'on puisse être en mesure de :

- convertir les langages de balises mathématiques existants vers MathML ;
- ajouter du MathML dans les pages HTML, afin que les applications Web qui manipulent, dès aujourd'hui, du HTML puissent, dans le futur, manipuler facilement du MathML ;

- "rendre" du MathML dans les navigateurs existants de quelque façon que ce soit (e.g. par images), même si cela est loin d'être idéal.

Note. Rappel : cette partie consacrée aux objectifs de MathML est issue du travail rédigé par des étudiants de 5e année de Génie mathématique, à l'INSA de Rouen : "Conversion TEX et rendu MathML" [sur internet : ./images/rapportMathML.pdf]. Merci à Ulric Genièvre, Yannick Litaize, Wojciech Machocki, Ludovic Maurillon, Benoît Roger, Sébastien Vallée d'avoir décidé de façon volontaire et active de participer au développement de l'information sur le glossaire de Tirème.

Principes

Syntaxe et grammaire

MathML est une application XML. Par conséquent, sa syntaxe est en partie dictée par la syntaxe XML et sa grammaire est spécifiée par une DTD. Ainsi, nous retrouvons les éléments classiques de la syntaxe XML, à savoir :

- les caractères Unicode XML (incluant les caractères ASCII ordinaires) ;
- les attributs ;
- les entités ;
- la structure de ses éléments ;
- la casse des chaînes de caractères.

Cependant, le *W3C* a trouvé, du fait de sa volonté d'encourager la modularisation des applications construites avec XML, que la forme actuelle d'une DTD ne convenait pas exactement : un groupe fut créé afin de développer des spécifications pour les Schema. MathML étant conçu pour que les mathématiques puissent tirer avantage des dernières évolutions technologiques en matière de Web, il doit donc y avoir un Schema pour MathML, ce qui est le cas.

Par ailleurs, MathML définit aussi certaines règles de syntaxe et de grammaire qui lui sont propres, lui permettant d'encoder davantage d'information, sans pour autant introduire beaucoup plus d'éléments et sans utiliser une DTD ou un XML plus complexes. Evidemment, l'un des inconvénients de ces règles spécifiques à MathML est qu'elles sont totalement transparentes pour un processeur XML classique. Ces règles sont de deux types :

- des critères sur la valeur des attributs ;
- des restrictions sur les *éléments fils* (des règles d'ordre, par exemple).

L'ordonnancement des *éléments fils* est très pratique (voire indispensable) dans bon nombre de cas, car il évite des problèmes de compréhension sans surcharger la syntaxe XML.

Exemple : Une fraction est constituée de deux éléments principaux : le numérateur et le dénominateur. MathML fournit donc une balise `mfrac` possédant seulement deux fils pouvant être de tous types, le premier étant obligatoirement le *numérateur*, le deuxième étant obligatoirement le *dénominateur*.

$$\frac{a}{b}$$

s'écrit donc :

```
<mfrac>
    <mi>
```

```
            a
        </mi>
        <mi>
            b
        </mi>
    </mfrac>
```

Les informations supplémentaires sur la valeur des attributs résident essentiellement sur le typage de ces valeurs. Ce problème peut être géré en préférant les Schema aux DTD. Il peut aussi être traité lors du parsing, en demandant à l'analyseur de vérifier la validité des données passées en valeur d'attribut. Le non-respect de ces règles entraîne une erreur MathML, bien qu'il ne gêne par le parsing XML.

Les notations utilisées pour décrire les types de valeurs sont les suivantes :

`number`	entier ou rationnel, pouvant commencer par l'opérateur binaire "-"
`unsigned-number`	entier ou rationnel non signé
`integer`	entier, pouvant commencer par l'opérateur binaire "-"
`positive-integer`	chaîne de caractères quelconques (toujours la totalité de valeur de l'attribut)
`string`	chaîne de caractères quelconques (toujours la totalité de valeur de l'attribut)
`character`	caractère seul, quelconque (excepté l'espace), ou une entité.
`#rrggbb`	code RVB d'une couleur
`h-unit`	unité de longueur (la liste des unités est donnée ci-après)
`v-unit`	unité de hauteur (la liste des unités est donnée ci-après)
`css-fontfamily`	unité CSS
`css-colorname`	unité CSS
`form+`	une ou plusieurs instances de « `form` »
`form*`	zéro ou plusieurs instances de « `form` »
`f1 f2 ... fn`	une instance de chaque `fi`, séparée ou non par des espaces
`f1|f2 ...| fn`	n'importe lequel des `fi` spécifiés
`[form]`	zéro ou une instance de « `form` »
`(form)`	1 instance de « `form` »
mot « plain text »	ce mot, littéralement présent dans la valeur de l'attribut
quoted symbol (e.g. '+' ou "+")	ce symbole, littéralement présent dans la valeur de l'attribut

L'ordre de priorité des opérateurs de cette syntaxe est la suivante (du plus fort au plus faible) :

`form+` ou `form*`,

- `f1 f2` ... `fn` (séquence),
- `f1|f2` ...| `fn` (alternative).

Dans la valeur des attributs, les mots-clés et les nombres doivent être séparés par des espaces, exceptées les "unités" (`h-unit`, `v-unit`) qui suivent les nombres. Pour les valeurs des attributs, l'espace n'est pas particulièrement requis, mais il est autorisé dans toutes les expressions exposées précédemment, excepté devant une unité (pour une comptabilité avec CSS1), entre le signe "-" et la valeur du nombre négatif, et entre # et `rrggbb`.

Le signe "+" n'est pas autorisé dans la syntaxe d'un nombre entier ou rationnel positif, sauf s'il est explicitement demandé.

Comme nous l'avons dit dans le tableau précédent ci-dessus, quelques attributs acceptent comme valeur des hauteurs et des largeurs (`h-unit` et `v-unit`), qui sont généralement formés d'un nombre suivi d'une unité. `h-unit` et `v-unit` se réfèrent donc à une unité pour exprimer respectivement la hauteur et la largeur lors du rendu des balises. Les unités possibles sont :

`em`	relatif à la largeur de la police courante
`ex`	relatif à la hauteur de la police courante
`px`	pixels (1 pixel = 2.54 cm)
`in`	inches
`cm`	centimètres
`mm`	millimètres
`pt`	points (1 point = 1/72 inch)
`pc`	picas (1 pica = 12 pts)
`%`	pourcentage de la valeur par défaut

Ces unités sont tirées des spécifications de CSS. Cependant, la syntaxe MathML reste différente sur ce point de la syntaxe CSS, puisque les nombres dans les feuilles de styles CSS ne peuvent être rationnels et peuvent être précédés du signe "+".

Lorsqu'il n'y a pas d'unité, le nombre donné sert de multiplicateur à la valeur par défaut.

Exemple :

```
<mo maxsize = "2"> ... </mo>
```

équivaut à

```
<mo maxsize = "200%"> ... </mo>
```

Cependant, pour rester compatibles avec CSS, il est rare que les unités de longueur soit optionnelles. Toujours par compatibilité avec CSS, 0 n'a pas besoin d'être suivie d'une unité, même si la syntaxe en spécifie une.

Pour faciliter la comptabilité avec CSS1, tous les éléments MathML acceptent les attributs `class`, `style` et `id`, en plus des attributs définis pour chaque élément. Les moteurs de rendu ne supportant pas CSS se doivent d'ignorer ces attributs. La valeur de ces attributs est définie comme étant une chaîne de caractères.

Tous les éléments MathML acceptent aussi l'attribut `other`, qui permet de passer des attributs non standard sans violer la DTD. Cet attribut peut s'avérer pratique, mais il est fortement déconseillé de l'utiliser, car, dans la plupart des cas, il est possible de trouver dans MathML d'autres façons de transmettre ces informations.

Les éléments MathML

Tous les éléments MathML appartiennent à l'une des catégories suivantes :

- les éléments de présentation,
- les éléments de contenu,
- les éléments d'interface.

Les éléments de présentation

Cette classe, composée d'environ 30 éléments acceptant un peu plus de 50 attributs, regroupe l'ensemble des éléments censés représenter des notations typographiques mathématiques. De manière générale, chaque élément de présentation correspond à plusieurs types de structure de notation, comme des intégrales, des exposants, des indices, des matrices, etc. Toutes les formules mathématiques seront mises en forme grâce à ces balises emboîtées les unes dans les autres, avec, aux extrémités de l'arbre résultant, les éléments simples tels que des symboles, des chiffres, des caractères…

Les balises de présentation sont de deux types :

- **Les token elements** : ce sont les symboles, les noms, les nombres, etc. En général, ils n'ont que des caractères et des éléments `mchar` comme fils. Tous les identificateurs, les nombres, les opérateurs, doivent être présentés à l'aide de ces éléments. Ils permettent aussi de représenter du texte ou des espaces, qui ont davantage un intérêt esthétique que réellement significatif, et pour représenter des chaînes de caractères afin de les rendre compatibles avec des systèmes algébriques.
- **Le Layout schemata** : permet de construire des expressions. Par conséquent, on comprendra aisément que l'on nomme ces éléments des "constructeurs d'expression". Ces expressions spécifient dans quel ordre les sous-expressions sont construites et intégrées (d'où l'importance du nombre et de la disposition de ses *enfants*). On appelle aussi un *enfant* de ces constructeurs un *argument*.

Les constructeurs peuvent eux-mêmes être regroupés en plusieurs classes :

- les éléments généraux : `mrow`, `mstyle`…,
- les scripts : `msup`, `munder`…,
- les tableaux : `mtable`, `mtr`…,
- `maction`.

Un exemple simple permet de comprendre la syntaxe de ces éléments. Prenons les racines d'un polynôme de second degré. On obtient :

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

, ce qui en MathML s'écrit :

```
<mrow>
    <mi>x</mi>
    <mo>=</mo>
    <mfrac>
        <mrow>
            <mrow>
```

```
                <mo>-</mo>
                <mi>b</mi>
            </mrow>
            <mo><mchar name="PlusMinus"/></mo>
            <msqrt>
                <mrow>
                    <msup>
                        <mi>b</mi>
                        <mn>2</mn>
                    </msup>
                    <mo>-</mo>
                    <mrow>
                        <mn>4</mn
                        <mo><mchar name="InvisibleTims"/></mo>
                        <mi>a</mi>
                        <mo><mchar name="InvisibleTimes"/></mo>
                        <mi>c</mi>
                    </mrow>
                </mrow>
            </msqrt>
        </mrow>
        <mrow>
            <mn>2</mn>
            <mo><mchar name="InvisibleTimes"/></mo>
            <mi>a</mi>
        </mrow>
    </mfrac>
</mrow>
```

L'élément mrow

Cet élément est utilisé pour regrouper un nombre quelconque de sous-expressions, généralement conçues d'un ou plusieurs opérateurs (de type `mo`) agissant eux-mêmes sur différentes expressions. La plupart des éléments agissent par défaut comme si leurs arguments étaient contenus dans un élément `mrow`. Cet élément accepte seulement les attributs communs à l'ensemble des balises MathML, à savoir `id`, `xref`, `class` et `style`.

Les éléments `mrow` sont rendus, comme leur nom l'indique, comme étant une rangée horizontale dans laquelle sont insérées des expressions (de gauche à droite) et par les synthétiseurs vocaux, comme une séquence d'expression. Mais cet élément n'est pas responsable du rendu des espaces entre les opérateurs et les sous-expressions. En effet, les opérateurs `mo` se chargent de fournir cette information. De même, cet élément n'indique pas directement où le système de formatage peut "couper" les expressions et revenir à la ligne (par exemple, lorsque l'expression est trop longue pour être visualisée sur une seule ligne), mais la présence des éléments `mrow` donne des points de repères sémantiques à l'interpréteur pour qu'il puisse déterminer ou séparer l'expression.

Si les balises `mrow` ne contiennent qu'un seul argument, tout se passe comme si l'élément était seul, sans être encapsulé dans des balises `mrow`. Cette équivalence est présente pour simplifier le travail des développeurs d'outils MathML. De même, si la balise `mrow` redéfinit des attributs (l'attribut *style* par exemple), il n'est pas demandé, dans la spécification MathML, d'exercer des règles de rendu particulières, mais les nouvelles directives d'affichage induites devront se répercuter sur l'argument.

Il n'existe aucune règle grammaticale pour l'insertion et l'utilisation des balises `mrow`. En fait, tout est laissé à l'appréciation de l'auteur. Cependant, il existe quelques recommandations pour l'utilisation de cet élément. L'intérêt de ces règles est triple :

- améliorer l'affichage en associant des espacements aux `mrow` lors du rendu ;
- favoriser les sauts de lignes lorsque cela est nécessaire au sein d'une même équation ;
- améliorer la compréhension sémantique lorsque l'on transmet le document MathML a un interpréteur algébrique ou à des synthétiseurs vocaux.

Généralement, on regroupera un opérateur avec l'ensemble de ses arguments, de manière à rendre les choses plus lisibles et donc plus compréhensibles. Mais voici les règles précises que l'on doit suivre.

Deux opérateurs adjacents, séparés ou non par des expressions (i.e. tout ce qui n'est pas un opérateur), appartiennent au même élément `mrow` si et seulement si les deux opérateurs ont la même priorité et si ces opérateurs appartiennent au dictionnaire des opérateurs fourni dans la recommandation MathML (appendice F). Dans tous les autres cas, les opérateurs seront dans des mrow distincts.

On entend par "opérateur" l'ensemble des opérateurs algébriques, mais aussi des parenthèses et des séparateurs.

Exemple : En respectant cette règle, l'exemple 2x +y – z s'écrit :

```
<mrow>
    <mrow>
        <mn>2</mn>
        <mo>&InvisibleTimes</mo>
        <mi>x</mi>
    </mrow>
    <mo>+</mo>
    <mi>y</mi>
    <mo>-</mo>
    <mi>z</mi>
</mrow>
```

de même, (x, y) s'écrit :

```
<mrow>
    <mo>(</mo>
    <mrow>
        <mn>x</mn>
        <mo>,</mo>
        <mi>y</mi>
    </mrow>
    <mo>)</mo>
</mrow>
```

L'élément maction

L'élément `maction` permet d'associer une action particulière à une expression. Cela permet d'ajouter une certaine interactivité au document. L'élément `maction` peut avoir un nombre quelconque d'arguments et il possède les deux attributs suivants :

- **`actiontype`** : cet attribut est indispensable. Son absence génère une erreur lors de l'analyse du document. Il n'y a pas de valeur par défaut, nous allons lister ci-après les valeurs possibles ;
- **`selection`** : de type entier, cet attribut définit le nombre de sous-expressions concernées par l'action liée. Par défaut, cette valeur est 1.

La valeur de l'attribut `actiontype` n'est pas formalisée, elle est fortement dépendante du type d'application utilisée. Pour indication, des valeurs possibles sont :

- **`toggle`** : permet d'afficher à tour de rôle les différentes sous-expressions, en changeant l'affichage à chaque clic de l'utilisateur ;
- **`hightlight`** : l'expression est surlignée lorsqu'on la sélectionne avec la souris ;
- **`menu`** : fournit un pop-up menu des sous-expressions.

Les balises de contenu

Cette classe regroupe une centaine d'éléments acceptant à peu près une douzaine d'attributs. Les éléments de contenu sont soumis aux mêmes règles d'ordre que les éléments de présentation.

Ces éléments, qui correspondent à une large gamme d'opérateurs, de relations et de fonctions, véhiculent essentiellement de la sémantique, des entités mathématiques au sens propre, censées être dégagés de toute notion de notation. Le but de ces éléments est de structurer une information afin de la retransmettre à des interpréteurs algébriques. Ainsi, l'ensemble des entités mathématiques étant conséquent, il existe un grand nombre de balises de contenu. De plus, il existe peu d'attributs contre un grand nombre d'éléments, le W3C Math Working Group *[sur internet : www.w3.org/Math/]* préférant lever le plus d'ambiguïté possible en introduisant de nouveaux éléments plutôt que de nouveaux attributs.

Un bon nombre de fonctions et d'opérations nécessitent des "quantificateurs" pour être bien définies. Par exemple, une intégrale doit spécifier les bornes d'intégration, ainsi que la variable sur laquelle nous intégrons. Il y a donc un certains nombre de **"qualifier schemata**" comme `bvar` ou `lowlimit`, utilisés avec des opérateurs comme `diff` (différentiel) ou `int` (intégrale).

Présentons à présent deux des éléments les plus importants des balises de contenu : les balises `apply` et `declare`.

L'élément declare

L'élément `declare` est particulièrement important pour les éléments de contenu qui peuvent être évalués par des systèmes algébriques. Cet élément fournit un simple mécanisme d'assignation, où une variable peut être déclarée comme étant d'un certain type, avec une certaine valeur.

L'élément apply

L'élément `apply` est utilisée pour appliquer des opérations à des expressions, ainsi que pour créer de nouveaux objets mathématiques à partir d'objets existants. Il permet par exemple d'appliquer une fonction à un ensemble d'arguments. Une fois de plus, l'ordre des arguments à une importance capitale : le premier fils correspond à la fonction à appliquer, les fils restants correspondant aux arguments de la fonction. En ce qui concerne les opérations algébriques, il faut respecter, comme en LISP, la notation polonaise inversée ; l'opération a – b s'écrit :

```
<apply>
    <minus/>
    <ci>a</<i>
    <ci>b</ci>
</apply>
```

Reprenons l'exemple

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

. Selon sa sémantique, il s'écrira :

```
<apply>
    <eq/>
    <ci>x</ci>
    <apply>
        <divide/>
        <apply>
            <fn><mo><mchar name='PlusMinus'/></mo></fn>
            <apply>
                <minus/>
                <ci>b</ci>
            </apply>
            <apply>
```

```
                <root/>
                <apply>
                    <minus/>
                    <apply>
                        <power/>
                        <ci>b</ci>
                        <cn>2</cn>
                    </apply>
                    <apply>
                        <times/>
                        <cn>4</cn>
                        <ci>a</ci>
                        <ci>c</ci>
                    </apply>
                </apply>
                <cn>2</cn>
            </apply>
        </apply>
        <apply>
            <times/>
            <cn>2</cn>
            <ci>a</ci>
        </apply>
    </apply>
</apply>
```

Ce code est beaucoup plus long que celui utilisé pour la notation et l'élément `apply` y joue un rôle primordial.

Notons ici que les éléments de contenu et de présentation peuvent être mélangés. La seule règle à respecter dans ce cas est que le contenu "panaché" n'est autorisé à apparaître que s'il veut vraiment dire quelque chose. Par exemple, pour l'équation

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

, le caractère "±" n'existe pas en balise de contenu. On utilise donc `fn` pour déclarer que cet opérateur agit en tant qu'élément de contenu.

Une autre option est l'utilisation de la balise `semantics`. Cet élément est utilisé pour lier des expressions MathML avec d'autres types de notations. Une utilisation courante de la balise `semantics` est de joindre des éléments de contenu à des éléments de présentation, en tant qu'annotation sémantique. Ainsi, l'auteur peut spécifier qu'une notation non standard est utilisée lors du rendu d'une expression utilisant seulement des éléments de contenu. Prenons l'exemple de l'intégrale de 0 à t de 1/x, codée en balises de contenu. Le rendu par défaut serait sans doute

$$\int_0^t (1/x)\,dx$$

. Nous allons donc fournir une expression de présentation qui servira d'annotation sémantique afin d'obtenir l'affichage

$$\int_0^t \frac{dx}{x}$$

.

Il faut cependant savoir que l'utilisation présentée ci-dessus des annotations ne fait pas partie des spécifications MathML. L'utilisation qui en sera faite sera donc liée au moteur de rendu choisie.

Note. Rappel : cette partie consacrée à la syntaxe et à la grammaire de MathML est issue du travail rédigé par des étudiants de 5e année de Génie mathématique, à l'INSA de Rouen "Conversion TEX et rendu MathML" [sur internet : ./images/rapportMathML.pdf]. Merci à Ulric Genièvre, Yannick Litaize, Wojciech Machocki, Ludovic Maurillon, Benoît Roger, Sébastien Vallée d'avoir décidé de façon volontaire et active de participer au développement de l'information sur le Glossaire XML.

Informations connexes

Mais finalement, c'est quoi la notation typographique mathématique ? Pour mieux comprendre la problématique, voici un extrait d'une étude réalisée par **Pierre Attar**.

La notation typographique mathématique (extrait d'un article publié dans "Documents Numériques", Vol. 1, N° 2, Editions Hermes)

Le champ de cet article s'inscrit dans la nécessité d'adapter les primitives graphiques du discours écrit pour lui permettre de s'intéresser à l'écrit scientifique : il s'agit de comprendre la nature, l'utilité et les évolutions de la notation typographique mathématique, comme outil supportant la spécificité de la communication scientifique.

En effet, s'il n'utilisait que les possibilités de l'écrit, le discours mathématique serait bavard : comprendre une phrase–texte décrivant un objet mathématique est beaucoup plus difficile que d'en comprendre une vision plus synthétique représentée par son "dessin". Par exemple, le symbolisme

$\sqrt[3]{4}$

est plus rapidement appréhendé – faisant partie d'une culture mathématique partagée – que le texte « *la racine cubique de la valeur entière quatre* ».

Ainsi, pour atteindre ses objectifs de communication, la communauté scientifique a été obligée d'enrichir l'expression écrite traditionnelle d'un ensemble de fonctionnalités. De nouveaux symboles et arrangements graphiques apparaissent, de nouvelles polices de caractères sont utilisées ; toutes choses qui donnent une réalité à part entière à la typographie mathématique.

Cette typographie est une extension de la typographie traditionnelle. Elle ne vient pas s'y substituer, elle l'enrichit, pour créer un nouveau mode permettant d'expliciter des concepts mathématiques : cette typographie est complètement intégrée au mode textuel. Cette intégration – interpénétration – est de nature fondamentalement différente de celle qui est utilisée pour faire coopérer les modes graphique ou image au texte, où la coopération tient plutôt de la juxtaposition.

Un mode de communication à part entière

Avant de s'intéresser plus précisément à la notation typographique mathématique, il est nécessaire d'en comprendre sa spécificité et son utilisation. Cela permettra d'éviter bon nombre des faux débats qui se traduisent par des utilisations abusives de la notion "d'équations mathématiques". Certes, les mots n'ont que le sens qu'on leur accorde, mais

derrière ces mots, bon nombre de personnes pense à tout autre chose que ce qui fait la spécificité de la notation typographique mathématique : une extension du discours écrit pour permettre l'expression d'un réel discours scientifique.

Sans prétendre cerner l'activité scientifique humaine, il semble qu'elle puisse appartenir à deux registres différents qui sont, d'une part, celui de la recherche et de l'élaboration et, d'autre part, celui de la communication et de la diffusion de résultats. À chacune de ces activités correspond des outils de travail différents, qui utilisent les concepts mathématiques sous–jacents à l'activité, selon un mode différencié.

L'activité de recherche, d'élaboration de résultats, est par définition extrêmement difficile à cerner et à expliciter ; le propos ici est de se centrer seulement sur les modes de communication utilisés lors de cette activité. Les éléments de base utilisés sont, de façon non exhaustive, la parole (ou son substitut qui est le mail sur Internet), les formules mathématiques, ainsi que les graphiques issus d'outils de représentation de ces mêmes formules. Depuis ces dernières années, un autre élément de communication apparaît autour des outils de modélisation et des outils de calcul numérique. Avec cette nouvelle catégorie d'outils, les données sont souvent stockées dans des formats propriétaires ; ce sont de réelles descriptions d'un concept mathématique, souvent vu d'un point de vue comportementaliste, voire opératoire.

La communication liée à l'activité de recherche se fait surtout à destination des "pairs", qui sont souvent eux–mêmes partie prenante du travail en cours : ce qui est échangé présuppose une connaissance commune importante des travaux en cours. Du coup, il est possible d'échanger des fragments d'information qui ne soient pas en eux–mêmes "auto–suffisants", et la communication peut être sommaire et partielle. Ainsi, deux personnes peuvent échanger un modèle mathématique électronique, issu d'un logiciel de calcul numérique . Un coup de téléphone, voire un mail d'accompagnement suffira à décrire l'objet envoyé, le récepteur ayant une excellente connaissance du contexte dans lequel l'information qu'il doit recevoir a été élaborée.

En revanche, ce mode n'est pas suffisant, dès lors que l'échange à un but de diffusion, de communication de résultats. En effet, la communauté de "pairs" devient alors plus étendue et les connaissances partagées ne sont pas aussi bien définies que précédemment. Il s'agit alors d'expliquer, d'expliciter, de (dé)montrer, tout en accompagnant le lecteur vers le résultat. Pour ce faire, le moyen depuis longtemps privilégié est l'expression écrite, dans lequel la notation typographique mathématique prend toute sa valeur, dès lors que l'expression devient scientifique. Même si aujourd'hui, avec l'avènement du Web et des CD–ROM, le support de consultation électronique pourrait devenir prépondérant, les modèles de communication reposeront toujours sur la culture commune de l'écrit, et il semble aujourd'hui encore difficile de s'en séparer.

Cela ne veut pas pour autant dire que les objets électroniques issus des outils de modélisation n'ont pas de place dans les extensions possibles du discours écrit pour mieux utiliser les nouvelles fonctionnalités offertes par la consultation électronique. Cela veut simplement dire que pendant un temps encore important, le discours écrit conservera toute sa spécificité et son pouvoir de représentation, de communication et de partage d'idées. Dans ce cadre, les objets électroniques issus d'outils de modélisation mathématique ne remplaceront pas la notation typographique mathématique, ils s'intégreront par juxtaposition, comme cela est actuellement réalisé sur papier pour utiliser le pouvoir de représentation de l'image et du graphisme. C'est aussi déjà de cette façon que fonctionnent bon nombre d'éditeurs d'information électronique, pour intégrer la vidéo, le son et les objets 3D à leur produits.

Spécificités de la typographie mathématique

La typographie mathématique propose de sortir d'une certaine linéarité du mode textuel pour faire appel au symbolisme qui synthétise des concepts et par là même densifie les contenus écrits. Pour ce faire, la typographie mathématique consiste à :

- *Étendre le jeu de caractères par de nouveaux caractères et des symboles*. Ainsi, aux caractères de la typographie traditionnelle s'ajoutent des jeux de caractères qui n'existent que pour la typographie mathématique comme les notions d'infini, d'orthogonalité, etc. Ces caractères et symboles sont en perpétuelle évolution et renouvellement, dès lors que l'on considère le travail réalisé par les chercheurs en mathématiques qui continuent à faire vivre cette science, et donc à la faire évoluer.
- *Formaliser l'utilisation de constructeurs graphiques*. C'est une nouvelle spécificité de la typographie mathématique. Par exemple, certains constructeurs, comme la racine, l'intégrale, la division ou encore la sommation, ont des spécificités d'étirement en fonction de leur contenu.
- *Définir des règles d'assemblage entre les objets graphiques*. Les caractères et symboles mathématiques suivent des règles d'assemblage particulières, qui rompent avec l'assemblage séquentiel de la typographie traditionnelle. Ainsi, des positionnements relatifs sont utilisés pour, par exemple, délimiter les bornes d'une intégrale.
- *Formaliser l'utilisation de polices de caractères*. Dans la typographie mathématique, une différenciation claire est réalisée entre les enrichissements utilisés pour définir des variables, des fonctions ou des nombres.

À l'instar de la typographie utilisée pour le mode textuel, la typographie mathématique ne fait que décrire les éléments mathématiques d'un discours. Même si sa formalisation semble plus codifiée, elle a les mêmes limites que le mode textuel : elle ne prétend pas porter un sens intrinsèque, complet et représentatif d'un concept mathématique qu'elle présente. C'est l'activité intellectuelle du lecteur qui donnera un sens au graphisme représenté. Le lecteur doit, pour ce faire, avoir connaissance d'un alphabet et d'une syntaxe spécifique : c'est sa culture scientifique.

La typographie mathématique peut alors s'autoriser des ambiguïtés et des imprécisions : elle gagne en densité et concision, toute chose qui lui confère un pouvoir pédagogique lui permettant de mettre en valeur des éléments de discours.

Éléments typographiques de la notation mathématique

Le nombre de constructeurs nécessaire à la notation typographique mathématique est relativement faible : il repose sur un ensemble de fonctionnalités graphiques permettant de spatialiser du texte ou des symboles graphiques, spécifiques à l'expression scientifique.

Tout d'abord, peu de notations mathématiques ont une définition typographique étroitement liée à la symbolique mathématique qu'elles représentent : un système doit, au minimum, connaître les sémantiques graphiques associées aux fractions et aux racines. Ces deux éléments ont la particularité de requérir une modification complexe de leur symbolisation en fonction des contenus.

D'autres éléments typographiques sont ensuite généralisés : ils reposent tous sur la notion de réserves graphiques hiérarchisées les unes par rapport aux autres, avec étirement de certains symboles. Le constructeur de base – nécessairement puissant – est alors celui qui permet le positionnement relatif d'informations les unes par rapport aux autres, pour permettre de définir des exponentielles, des indices et des vecteurs, etc.

À ce système s'ajoute des notions de sur/sous–lignages ; les traits horizontaux doivent s'étendre en fonction de la taille de la boîte référencée. Ces systèmes sont de même nature que le système vertical de parenthésage, qui inclut les notions d'intégration, de sommation

et de produit. En effet, une intégration peut, par exemple, être comprise comme étant une parenthèse gauche spécifique, s'appliquant à une boîte : l'intégrante. La spatialisation de boîtes dans un espace à deux dimensions permet, grâce aux parenthèses, de mettre en place la notion de matrices.

Pour accompagner ces constructeurs, le format de codage doit proposer, outre le jeu de caractères du mode textuel, l'accès à l'alphabet grec et un jeu de symboles mathématiques conséquent. Pour présenter ce dernier, les caractères et symboles seront accompagnés de polices de caractères identifiantes du mode mathématique et de la nature des objets sous–jacents, comme les variables ou les fonctions.

Recommandations(s)

- *Le langage de balisage mathématique (MathML) version 2.0*
  Recommandation, version 2.0, du 21-02-2001
  Document sur http://www.yoyodesign.org/doc/w3c/mathml2/overview.html
- *Mathematical Markup Language*
  Recommandation, version 3.0, du 20-10-2010
  Document sur http://www.w3.org/TR/MathML3/
- *Mathematical Markup Language Specification*
  Recommandation, version 1.01, du 07-07-1999
  Document sur http://www.w3.org/TR/REC-MathML/
- *Mettre des mathématiques sur le Web avec MathML*
  Note, version 2003, du 2003
  Document sur http://www.yoyodesign.org/doc/w3c/math-on-the-web-with-mathml/
- *Putting mathematics on the Web with MathML*
  Note, version 2003, du 2003
  Document sur http://www.w3.org/Math/XSL/