An HTML-Native Math Proposal

March 5, 1997

Stephen Buswell sb@stilo.demon.co.uk
Stéphane Dalmas stephane.dalmas@sophia.inria.fr
Stan Devitt jsdevitt@maplesoft.com
Angel Diaz diaz@watson.ibm.com
Marc Gaetano marc.gaetano@sophia.inria.fr
Robert Sutor sutor@watson.ibm.com
Terry van Belle vanbelle@maplesoft.com
Stephen Watt smwatt@watson.ibm.com

Contents

1 Introduction and overview

This working document details a form for representing mathematics in HTML. The model, based on discussions over the past few months among members of the WWW Consortium Working Group on HTML Mathematics, attempts to take into account the practical necessities arising in mathematical markup, while harmonizing as closely as possible with the existing HTML constructs and overall philosophy.

This work should be seen in the context of the overall effort to produce a standard for HTML-based mathematics. The reader is encouraged to consult the HTML-Math working group home page, `http://www.ams.org/html-math/`.

From this point several resources connected with HTML Mathematics may be reached. We draw particular attention to the following two documents:

- Robert Miner's analysis of browser interface considerations:

  `http://www.geom.umn.edu/~rminer/w3c/interface/`,

  and

- a proposal from Wolfram Research Inc,

  `http://www.ams.org/html-math/wolfram-proposal-960531.html`.

The HTML-Math working group recognizes that the eventual HTML-math standard will likely be realized first by browser plug-ins, before integrated implementation in mainstream browsers. Therefore the working group shares the growing interest in seeing improved support for communication between plug-ins and the browser environment. Examples of the sort of information required for high-quality math plug-ins are text base-line, and size of the current font. These implementation questions are explored in Robert Miner's analysis, which should be read as a co-quisite to this document.

Secondly, the HTML-math working group has discussed at some length the proposal drafted by Bruce Smith of Wolfram Research Inc. and available on the HTML-math home page. While the precise proposal has evolved somewhat from the posted document, we recognize the considerable experience at WRI brings to bear in the editing of a broad range of mathematical expressions, and see the issues and the set of operators put forward by the WRI proposal as an important contribution to the eventual HTML definition.

The current document builds on this work by specifically addressing the need for a so-called "HTML-native" syntax for representing mathematics, and the mechanism used to capture mathematical semantics in such an environment.

Ideally, the overall syntactic structure should be compatible with the current HTML definition. To achieve this, numerous issues must be taken

into account. The primary goals are summarized in the working group charter as:

"... the overall goal of HTML-Math Working group is to develop an open specification for HTML-Math that

- Is suitable for teaching, and scientific and technical communication;
- Is easy to learn and to edit by hand for basic math notation, such as arithmetic, polynomials and rational functions, trigonometric expressions, univariate calculus, sequences and series, and simple matrices;
- Is well suited to template and other math editing techniques;
- Insofar as possible, allows conversion to and from other math formats, both presentational and semantic, such as TeX and computer algebra systems. Output formats may include graphical displays, speech synthesizers, computer algebra systems input, other math layout languages such as TeX, plain text displays (e.g. VT100 emulators), and print media, including braille. It is recognized that conversion to and from other notational systems or media may lose data in the process;
- Allows the passing of information intended for specific renderers;
- Supports efficient browsing of lengthy expressions;
- Provides for extensibility, for example through contexts, macros, new rendering schema or new symbols.
- Some extensions may necessitate the use of new renderers. "

Furthermore, " ... the Work will be complete when:

1. A formal specification of an HTML-Math markup scheme has been approved by consensus of group members. The scheme will:
   - suffice as a basis for visual rendering of most mathematical notation (e.g. at the level of a research mathematics journal article); and
   - enable renderings to other forms as mentioned above to a level acceptable to group members.
2. A reference implementation of software which visually renders mathematical notation as specified in item 1 exists that demonstrates feasibility of the notation, and facilitates implementation of rendering engines for other media. "

This proposal investigates how best to achieve these goals while

- using an HTML (SGML) syntax,

- using *OpenMath* to provide a representation for the underlying mathematical objects,
- trying to adhere to existing or emerging international standard (SGML, XML, DSSSL, *OpenMath*),
- trying to use existing solutions for our problems even if more specialized solutions could be devised,
- being generally implementable: you need not be an expert in the fields of mathematical typesetting or computer algebra (at least in the sense of the members of the working group) to be able to provide a decent implementation.

We acknowledge that this draft is incomplete in its exposition of necessary details. For example, the tags used have been largely representative. Nor have we consulted the comprehensive set of entities and gliphs kindly compiled by the combined efforts of PatricK Ion, Nico Poppelier and Niel Soiffer.

1.1 Syntax issues

There have been many discussions on the issue of what syntax to use to markup mathematics for publication, be it electronic or otherwise. The choice is typically between some *common* syntax[1] and the regular HTML/SGML syntax that uses tags to represent the structure.

The advantages of using pure SGML syntax are:

- It is the syntax used by the surrounding document.
- HTML markup (such as HREF, COLOR...) can be used inside the MATH element.
- Any tool that can parse HTML could be adapted to parse the content of the MATH element.
- Standard SGML tools and techniques (such as DSSSL) can be used.
- This syntax is very regular (easy to generate and unambiguous).

However, this solution has two main drawbacks: the syntax is not "natural" to most people who are more used to TEX or an infix notation, and it is tedious to write by hand.

Although manual input is possible, it is expected that authoring software applications will become quickly available to produce this HTML syntax. Computer algebra system vendors have already expressed a great interest in supporting HTML Math as an output format. Support for *OpenMath*

[1] "Common" here meaning close to a syntax already used by people writing mathematics, such as a TEX syntax, or an "infix" syntax as used by computer algebra systems

is also a good reason why tools will certainly become available to markup mathematical objects so that they can be reused via cut-and-paste or drag-and-drop into computational software.

In this proposal we therefore advocate and use a native SGML syntax. Note that we do make some concessions through use of some special SGML features to enable simple formulas to be entered simply, as described in 4.2. While these features are not all valid XML (at least for the current XML draft), they do provide a well defined syntax, and vehicle for the automatic translation into XML or a normalized form. In addition, their use can be considered temporary, and probably only for manual entry of math markup before the wide availability of authoring tools.

Note that there is no other complete ready-made solution to our syntax problem. TEX syntax does not readily provide the structure we need to easily add semantic content in addition to visual formatting information. Operator precedence parsing provides the structure and a natural notation but it seems to be necessary stay with a few common infix operators if we want to avoid problems such as difficult parsing by humans and overly complicated parsers that are hard to specify and implement.

1.2 Semantic issues

The HTML Math specification is not only about presentation of math formulas on Web pages. It aims to provide a way to communicate the meaning of the mathematical objects from the author to the readers.

Two important issues are *freedom of expression* and *reusability.* Authors want the freedom to specify the markup for their formulas. But too much freedom has a big drawback: you cannot really share your formulas with others. If you want to reuse your Web page math objects in other applications, you should conform to a standard markup style. *OpenMath* already provides such a standard.

An interesting question to ask is whether semantic markup is sufficient for all representations of mathematical objects. After all, knowing the meaning of a formula should imply that at least one (reasonable) presentation can be produced. Isn't that exactly what a computer algebra system does when it prints its results? Unfortunately, there are two problems with this simple approach:

- Mathematics is a large and living discipline. That is why *OpenMath* is itself extensible and new content dictionaries can be introduced. If a presentation agent has static display knowledge, it won't be able to cope with new notations. It also would have a very big knowledge base or will cover only a few parts of mathematics.

- There can be different notations for the same objects, depending on the taste of the author, the intended audience (high school students

vs. professional mathematicians vs. theoretical physicists, etc.) or the field of mathematics.

So we must provide a way to specify some visual information and this must be at the formula-level of granularity. We can

- attach some presentation information to an operator, or
- attach an expression representing the presentation to an entire expression.

The second way is the simplest and it is just a matter of defining a set of "presentation operators" in the spirit of TEX or the Math DTD fragment of ISO 12083 and mix them with the "semantical" operators. Note that it is possible (and sometimes unavoidable) that the only operators used to represent a formula are presentation operators.

The first technique poses some real problems as it implies that some kind of programming facilities should be available to describe the presentation. This issue is discussed in the next section.

Note that "presentation" in the above discussion is not limited to "visual presentation". The previous considerations largely apply to the issue of audio rendering.

2 Web-oriented implications

2.1 Speed of transmission

The challenge facing any web standard is to strike a balance between content and speed of transmission. HTML has addressed this issue by providing a mechanism for specifying a document structure and delegating formatting issues to the browser. Our approach follows the same philosophy. Only the overall structure of a math equation is specified, with the rendering of this equation performed by the client. This is in contrast to a server-side image processing solution such as the pre-rendering of equations as GIF images.

2.2 Dynamic reformatting

Given that web pages are displayed on a variety of systems, any mathematical HTML system must be able to adjust the characteristics of the mathematics display to user specified characteristics of the client. This need for dynamic display ranges all the way from adjusting font sizes and baselines of expressions such as $\frac{1}{x+1+\frac{2}{x+1}}$ to the line-breaking long mathematical expressions such as

$$1+\frac{41}{2}\,\frac{1}{(x-2)^2}+\frac{225}{4}\,\frac{1}{x-2}+\frac{363}{4}\,\frac{1}{x-4}-\frac{271}{2}\,\frac{1}{x-3}+\frac{1}{2}\,\frac{1}{x-1}$$

which, on a narrow screen, might better be rendered as

$$\begin{aligned}1+&\frac{41}{2}\,\frac{1}{(x-2)^2}+\frac{225}{4}\,\frac{1}{x-2}\\&+\frac{363}{4}\,\frac{1}{x-4}-\frac{271}{2}\,\frac{1}{x-3}\\&+\frac{1}{2}\,\frac{1}{x-1}\end{aligned}$$

Our specification implicitly contains the expression structure of a given mathematical equation. This structure can be easily extracted and used to intelligently line break and possibly reformat an equation.

2.3 Embedded HTML tags

It is essential that existing HTML capabilities be usable in a mathematical context. Unlike other approaches, which represent mathematical content as an atomic block employing a foreign syntax, (eg. TEX), our approach is a true SGML-based extension to HTML. This approach allows HTML tags, both present and future, to be embedded directly within the mathematical content, providing additional functionality and flexibility.

For example, the tags for creating hyperlinks between documents could be incorporated directly on parts of the mathematical expression to allow the user to click on a mathematical function name in order to view its definition elsewhere. The following HTML source

```
<math>
<GCD><a href=#GCD_DEF>GCD</a>{{x^2}-1<sep>x+1}</GCD>
</math>
.
.
.
<b><a name=GCD_DEF>Definition of GCD:</b> ...
```

would produce the equation $\underline{\mathrm{GCD}}(x^2 - 1, x + 1)$ with the GCD characters underlined as a link to the definition. (See the section on Markup for a discussion of the use of shortrefs.)

To display $a + b$ with the b in red, the HTML would look something like

```
<math>
{a + <color red>b</color>}
</math>
```

By facilitating the re-use of existing (and future) HTML structures, supplementary software tools such as spell checkers, search engines, structure verifiers and URL checkers either remain valid or can easily be adapted.

2.4 Acceptance by browser developers

The eventual goal of any HTML standard is that support for it eventually be incorporated into the major commercial web browsers.

If we are being pragmatic, we must accept that successful adoption must be based on a phased implementation. The first phase would be based on plug-ins or other components such as ActiveX controls. This allows for the market research and response that would be required before acceptance by the major browser vendors.

2.5 Information transfer with other applications

While the current use of HTML math in web browsers is for the most part restricted to display, the design must take into account information transfer to and from other packages.

These packages include document preparation involving tools such as MathType, Expressionist, and Frame.

All these underlying representations are based on the mathematical structure of the expression. For example, the expression

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

is represented in Frame's MIF format as

```
<MathFullForm
‘equal[char[x],over[plus[minus[char[b]],pm[sqrt[plus[indexes[1,0,char[b],
```

```
num[2.00000000,"2"]],minus[times[num[4.00000000,"4"],char[a],char[c]]]]]],
times[num[2.00000000,"2"],char[a]]]]'
>
```

An ISO-12083-based representation is used by several technical publishers. Other software packages such as computer algebra systems provide a much more sophisticated set of manipulations on equations.

Our proposal allows including semantic content within the mathematics via the OpenMath standard. This permits any software package which can understand OpenMath to represent the actual meaning of the math, not just its displayed form. In the case where only static math content is desired this additional markup need not be included. Furthermore our "base" markup-up can be associated with a default OpenMath content dictionary, consequently providing default semantic information.

3 Mathematical software implications

Although the first goal of HTML Math is to provide the means to write and display equations and formula inside Web pages, the ability to send the underlying mathematical object to some other programs requires it to support the semantics of these objects. The typical example of such a situation is illustrated by cut-and-paste from a Web page to a computer algebra system or a spreadsheet.

Representing the semantics of mathematical objects is a difficult problem that has been addressed by many people coming from various scientific communities and clearly remains beyond the scope of this working group. There is an international effort, the *OpenMath* Consortium, that is currently addressing this issue. As one of the encoding formats of *OpenMath* is "SGML compliant" its integration within HTML Math is fairly straightforward.

There are several clear advantages to using *OpenMath* to describe the semantics of HTML Math objects:

- The problem is solved "for free" by another group.
- This *OpenMath* encoding provides a natural (vendor-neutral) format for cut-and-paste, drag-and-drop, etc.
- There is an immediate compliance with *OpenMath* applications. Today, three computer algebra vendors have already expressed a strong interest in supporting *OpenMath* and several universities and research laboratories are planning to use *OpenMath*. This includes the ability to use some *OpenMath* programs to easily generate this part of HTML Math using a syntax with which the user is familiar.

The issue is then how to embed an *OpenMath* object in an HTML document and to associate it with an HTML Math expression.

3.1 OpenMath

The goal of the *OpenMath* effort is to define a platform-independent, vendor-neutral standard for the representation of mathematical objects so that they may be exchanged in a meaningful way between various software tools.

The origin of *OpenMath* dates to 1992 and discussions among Maple developers as to how best organize communication among mathematical software packages. Professor Gaston Gonnet organized a workshop at the ETH Zurich in December 1993, and sought the participation of potentially interested members of the computer algebra community.

On December 15, 1996, the first official version of *OpenMath* was released, and is available through the *OpenMath* web site `http://www.openmath.org/OpenMath`. This release consists of a set of *content dictionaries* (CDs), covering basic mathematical concepts and two prototype libraries in C and Aldor.

In a nutshell, a content dictionary defines a set of related concepts and operations as a set of symbols. It can also define one or more representations for objects (such as multivariate polynomials). Content dictionaries are written in a specific syntax and can be viewed as SGML documents (a DTD is available). It is possible to translate this form to *OpenMath*, thereby making a content dictionary an *OpenMath* object. The structure of a content dictionary is specified by a special content dictionary, called *Meta*.

3.2 HTML Math examples per this proposal

The simple expression $a^2 + 1$ can be encoded in HTML Math as:

```
{ { a ^ 2 } + 1 }
```

The equivalent SGML encoding in *OpenMath* annotates the symbols with the Content Dictionary to which they belong (here, it is *Basic*):

```
<OMapp>
  <OMsymb name="+" cd="Basic">
  <OMapp>
    <OMsymb name="^" cd="Basic">
    <OMvar name="a">
    2
  </OMapp>
  1
</OMapp>
```

The expression $\cos^2 x$ is encoded in *OpenMath* as

```
<OMapp>
  <OMsymb name="^" cd="Basic">
  <OMapp>
    <OMsymb name="cos" cd="Basic">
    <OMvar name="x">
  </OMapp>
  2
</OMapp>
```

3.3 Putting things together

The *OpenMath* encoding is not intended to be hand-entered but rather generated by a program, possibly directly from the HTML Math format given a set of suitable defaults. Various solutions can be envisioned to mix these two notations, the simplest being to wrap both inside the same math tag:

```
<math>
{ { a ^ 2 } + 1
<OM> <OMapp> ... </OMapp> </OM>}
</math>
```

The major drawback of this is that it is difficult to recover the meaning of a sub-expression because the HTML Math and *OpenMath* trees may be structurally different.

For example, the expression C^2 (the space of functions whose second derivatives are continuous) could have the same HTML Math structure as the expression x^2 though they are completely different mathematical objects and consequently have different *OpenMath* representations. `<INFIX>C<EXP>2</INFIX>`.

The semantics of `{ C ^ 2 }` is denoted by

```
<OMapp>
  <OMsymb name="C" cd="FuncAnalysis">
  2
</OMapp>
```

(where *FuncAnalysis* is a suitable content dictionary dealing with functional analysis). The semantics of `{ x ^ 2 }` is encoded as

```
<OMapp>
  <OMsymb name="^" cd="Basic">
  <OMvar name="x">
  2
</OMapp>
```

There are several ways to circumvent these problems:

- use of *OpenMath* markup inside sub-expressions
- use of the attribute feature of *OpenMath* to specify the way certain parts are represented (thus embedding the presentation inside the semantics)
- provide a default mapping from "pure" HTML Math (HTML Math without *OpenMath* markup) to *OpenMath*.

4 The proposed HTML DTD with math

This section describes the various parts which make up the proposed mathematics fragment for HTML. It also describes the SGML structures which underly the proposal. In preparing these definitions, the following aims have been kept in mind:

- To allow the user to enter simple math expressions in as natural and intuitive a manner as possible.
- To mingle the overhead of the HTML structure whilst retaining all of the information content of the math.
- To simplify the processing required by providing mechanisms for explicit grouping, thus avoiding issues of operator precedence.
- To maintain full compatability with the existing HTML structures, and with the SGML and XML standards. However, no knowledge of SGML is required in order to use the math fragment.
- To support the use of the various level of math functionality proposed in a user-selectable way.
- To ensure that there is no impact on an HTML user not wishing to use any of the math facilities.

4.1 Components of the HTML DTD with math

The HTML DTD with Math consists of

- The standard HTML DTD (3.2 at the moment of writing). No modifications are required to any of the elements in this definition.
- The proposed math fragment. This is a set of additional HTML elements for use when encoding mathematics in HTML pages.

The elements of the math fragment may be included directly in the HTML DTD, or by use of an SGML *External Entity.* This is a mechanism for including a reference to an external file within a DTD, so that a parser will treat the contents of the file referenced as part of the main file. A typical external entity declaration and reference would be:

```
< !ENTITY  \% HTMLMath PUBLIC ''W3C - 1997//DTD Math//EN''>

\%HTMLMath;
```

"`W3C - 1997//DTD Math//EN`" is the *Formal Public Identifier* which is the published standard name for referring to this fragment. The location of the fragment file may be a local file, the target of a hyperlink or some other

reference. The resolution of the reference is a task for the browser and is not defined by the standard. One approach would be to provide a reference version on a publically accessible site which could be accessed, hyperlinked to, or downloaded as required.

The math fragment

The math fragment itself consists of the following:

- The math, operator and scope elements described below
- An AMS-LaTeX derived element set
- An OpenMath element set
- Additional element sets as required

These elements may themselves be located in different fragments. The only requirement placed on the math elements is that they must not conflict in name with any element elsewhere in the HTML definition.

4.2 Mathematics notation in HTML

Here we describe the notation used to describe math in HTML. HTML math is like the rest of HTML, ie. the user content is contained within elements. We will define some specific elements for mathematical use. Elements in HTML (as in any SGML) are of two types:

- Elements with content. These have start and end tags surrounding the content. For example `<SQRT> x </SQRT>` represents "the square root of x".
- Canonically empty elements. These have start tags only, and no content. For example `<PLUS>` represents the operator "plus".

We define a top level element `<MATH>` which serves to delimit math structures embedded in an HTML page. All other math elements are embedded within `<MATH>`. `<MATH>` will be defined in such a way as to include the user content elements of existing HTML, such as anchors, text formatting, insertion of images etc.

Operator elements

This proposal suggests the definition of the major math operators as canonically empty elements. For example:

Operator	Element
+	<PLUS>
-	<MINUS>
/	<DIVIDE>
*	<TIMES>
^	<EXP>

(list to be completed).

So "`x + y`" becomes "`x <PLUS> y`". However, this looks unnatural for non-HTML experts and is also verbose to write. Therefore we define two further SGML constructs: a general entity and a shortref declaration.

An entity is a way of defing a name for a text substitution string such that, wherever the entity name is encountered, the parser will substitute the text string (This feature is already used in standard HTML DTD). We define:

```
<!ENTITY pl "<PLUS>"   >
```

So wherever the parser encounters `&pl;` it understands `<PLUS>`.

Now we can write "`x &pl; y`" and the parser understands it as "`x <PLUS> y`".

We then define:

```
<!SHORTREF PLUSMAP     '+'     pl      >

<!USEMAP PLUSMAP       MATH    >
```

The shortref declaration defines the `'+'` character as a reference to the entity `pl` we have defined. The `USEMAP` defines the scope of the association: we restrict it to just the elements within `<MATH>`, so that elsewhere in an HTML page it has its usual character interpretation.

Now we can write "`x + y`" and the parser understands it as "`x <PLUS> y`". In the same way we can define entities and shortrefs for each of the operator elements.

Infix elements

The definition of the mathematical operators above as canonically empty elements relies on the idea of infix operators. We define an infix element whose mathematical meaning is governed by the embedded operator. For example:

```
<INFIX>w <TIMES> z</INFIX>
```

or equivalently using a shortref for `<TIMES>`

```
<INFIX>w * z</INFIX>
```

The operator `<TIMES>` serves both to define the purpose and to separate the two arguments of the operation.

Note here that the multiplication operation is explicit:

"2b" is a single text string:

"2*b" is an operation

A rendering engine may of course display "2*b" as "2b". The presence of the operator is required in the encoded math, not necessarily on the screen.

Scope elements

In order to allow a user to specify explicit element groupings, and also to avoid problems of operator precedence, we define a scope element which serves to delimit the scope of a mathematical structure (operation, function, ...). This element serves the same role as parentheses frequently do in traditional math notation. For example:

```
<COS><M>x</M></COS>
```

for $\cos(x)$ where `<M>` is the scope delimiter element.

We can further define shortrefs {} for the start and end tags of the scope element, allowing us to write, for example:

```
<COS>\{x\}</COS>
```

It is possible to use the same element as both the infix element and the scoping element, giving:

```
\{\{a + b\} * c\}
```

Note that if it is necessary to use the shortref character in its original meaning without interpretation, we can create another entity to represent the uninterpreted character. This is analogous to writing `&LT;` for '<' in standard HTML.

4.3 OpenMath elements

The *OpenMath* elements are a set of HTML math elements included in order to allow the user to encode additional information related to the underlying meaning of the math. These elements will allow interchange of information between math-aware HTML applications (eg. browsers) and more math-specific applications such as Computer Algebra packages. This is discussed in greater detail in the section on "Mathematical software implications".

From the structural viewpoint, the *OpenMath* elements are all defined as available inside the `MATH` element and its children. *OpenMath* elements are always optional: if the additional level of semantics is not required, it may be omitted. *OpenMath* elements, if used, are simply embedded at relevant place in the math element structure: no parsing of this structure is required to retrieve the openmath content of the page.

Note that if a presentation-based approach has been taken to the tagging, it may be the case that the HTML and *OpenMath* structural decompositions of the math do not match. In this case it will not be possible to embed the *OpenMath* information.

For example, consider "the integral from 0 to 1 of cos-squared x with respect to x."

"Presentation" markup (importing TeX constructs):

```
<E><INT><SUB>0</SUB><SUP>1</SUP><COS><SUP>2</SUP>x dx</E>
```

There is no part of the structure whose scope is the concept "cos x": there is nowhere to attach *OpenMath* information at this level.

"Semantic" markup (using ',' as a shortref for ¡SEP¿ a generic separator element):

```
<E><INT>0,1,x <SQR><COS>{x}</COS></SQR></INT></E>
```

Here "cos x" corresponds to a defined part of the structure, and the *OpenMath* information may be embedded at the level required.

OpenMath scope and operator elements

OpenMath structures may require more information about the operators than is described in the section on operator elements above. We can define a separate scope element for *OpenMath* structures, and additional elements carrying this extra information. Within the context of the *OpenMath* scope element, we can define shortrefs and entities for these elements. We can (optionally) reuse the same characters as for the 'standard' operators.

For example: `<OM>` is the *OpenMath* scope element, having shortrefs `[ ]` for its start and end tags. We redefine '+' to correspond to `<OM-PLUS CD="basic">` within the `<OM>` element.

`[x + y]` corresponds to `<OM>x <OM-PLUS CD="basic"> y </OM> </OM>` but within the normal math scope element the existing definitions still apply:

{`x + y`} corresponds to `<M>x <PLUS> y </M>`.

4.4 Compatibility Issues - HTML, SGML, XML

One key feature of any proposal is that it should impose few or no restrictions on users of the existing parts of HTML.

HTML compatibility

There is no impact for users of existing HTML not wishing to use the math features. As the math elements proposed are normal SGML (ie. HTML) elements, it is possible to intersperse existing HTML elements such as anchors and text formatting elements within the math to enhance the readability and provide access to existing HTML functionality.

SGML compatibility

All the notation techniques described in this proposal are valid in standard SGML.

XML compatibility

Using the approach defined in this proposal, it is possible to write valid XML, subject to the restriction that the use of shortrefs is not supported in the current XML draft. It is therefore necessary, if writing the XML directly, to use the entity names or element tags themselves.

However, SGML normalization tools exist (eg. James Clark's SPAM), and we can expect tools which will convert the SGML form to an XML form to be available soon. With some technical caveats (which have no bearing on the HTML DTD or on any part of this proposal) any valid SGML document can be converted automatically to an XML document without loss of information in the markup.

5 User-defined elements and macro substitution

Two of the issues frequently raised in the context of SGML-like notations for mathematics are the questions of author-preference and extensibility. It must be possible for an author to choose the notation used for a particular function, and to create new functions not explicitly provided for in the specification.

`<FN name="`*user-defined*`" layoutinfo="`*more user info*`">` is a user-definable math element However, if the browser is unaware of this element, there will be no special rendering. The layout attribute gives presentation information provided by the user, possibly in the form of a glyph definition, or a reference to an image file. This presentation may be ignored by the browser. The `layoutinfo` attribute can be of type `#IMPLIED`, making it optional for the user.

This proposal further foresees the need for macro support whereby user-defined element names may be used for elements of the math fragment. A trivial example is that the user might wish to write `<SQR>x</SQR>` and have it understood as `<INFIX>x<EXP> 2</INFIX>`.

There are several possible approaches:

- Encode the transformation rules in a format such as CSS (Cascading Style Sheets). Existing browsers have some support for CSS.
- Use a DSSSL-like encoding (DSSSL = **D**ocument **S**tyle and **S**emantics **S**pecification **L**anguage, ISO standard 10179). This specification would be in the form of an SGML document which specifies transformations between branches of a document node tree and another output document. This SGML document is separate from the HTML markup document, and might be referenced from a `<LINK>` or `<META>` element at the head of the HTML page. The SGML document could be processed by a DSSSL application or a Java applet.
- Use a notation such as Architectural Forms from the HyTime (Hypermedia/ Time-Based Structuring Language, ISO 10744). This allows the specification, via HyTime attributes, of a 'meta' element definition (the AF) in a Meta-DTD which can be understood by the HyTime processor but is transparent to the normal SGML processing. Another possibility is to look at the HyTime Function Notation.

The full DSSSL and HyTime specifications are very rich and it is likely, if either of these approaches were adopted, that a restricted subset would be adequate. There is a further issue as to whether the standard (or subset) is formally adopted, or whether we simply use a compatible notation.

6 Examples

Example 1

$$x$$

TEX

```
$$x$$
```

Short HTML

```
<math> x </math>
```

Example 2

$$x^2 + y^2 + z^2$$

TEX

```
$$x^2 + y^2 + z^2$$
```

Short HTML

```
<math>{ {x ^ 2} + {y ^ 2} + {z ^ 2} }</math>
```

Full HTML

```
<math><infix>
  <infix>x <power> 2</infix>
  <plus>
  <infix>y <power> 2</infix>
  <plus>
  <infix>z <power> 2</infix>
</infix></math>
```

Note: `<INFIX>` is the scope element whose function is determined by the embedded operator (here `<TIMES>` `<PLUS>` `<MINUS>` etc.)

Example 3

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

TEX

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

Short HTML

```
<math>
<frac>
- b &pm; <sqrt> {b ^ 2} - {4 * a * c} </sqrt>
<sep>
2 a
</frac>
</math>
```

Example 4

$$\left(\frac{a}{p}\right) \equiv a^{\frac{1}{2}(p-1)} \pmod{p}$$

TEX

```
$$\left(\frac{a}{p}\right) \equiv a^{\frac{1}{2}(p-1)} \pmod p$$
```

Short HTML

```
<math>
<frac>a , p</frac> <equiv>
{a ^ {<frac>1,2 </frac>*{p - 1}}
<pmod> p
</math>
```

Example 5

$$T_{\mu\nu} = F_{\mu}{}^{\kappa} F_{\nu\kappa} - \frac{1}{4} g_{\mu\nu} F^{\kappa\lambda} F_{\kappa\lambda}$$

TEX

```
$$T_{\mu \nu} = {F_{\mu}}^{\kappa} F_{\nu \kappa}
- \frac{1}{4} g_{\mu \nu} F^{\kappa \lambda}F_{\kappa \lambda}$$
```

Short HTML

```
{T_{&mu;&nu;}}
=
{
{{F_{&mu; }^{&kappa;}}*F_{&mu; }^{&kappa;}}
-
{<frac>1,4</frac>{g_{&mu;&nu;}*F^{&kappa;&lambda
}*{F_{&kappa;&lambda;}}}}
}
```

Full HTML

```
<math>
  <m> T<sub>&mu;&nu;</sub> </m>
  =
  <m> F<sub>&mu;</sub><sup>&kappa;</sup> </m>
  -
  <m>
    <infix>
      <m><frac>1<sep>4</frac></m>
      <times>
      <m>g<sub>&mu;^nu;</sub></m>
      <times>
      <m>F<sup>&kappa;&lambda;</sup></m>
      <times>
      <m>F<sup>&kappa;&lambda;</sup></m>
    </infix>
  </m>
```

Example 6

$$\int_C \mathrm{d}\omega = \int_{\partial C} \omega$$

TEX

```
$$\int_{C} {\rm d}{\omega} = \int_{\partial C} \omega$$
```

Short HTML

```
<math>
<int> C,,,&omega;,</int> = <int><partial>C</>,,&omega;,</>
</math>
```

Note: By convention we order the child elements of the `<int>` lower limit: upper limit: bound variable: integrand. There are other ways, for example by using attributes.

Example 7

$$\frac{df}{dx} \neq \frac{f}{x}$$

TEX

```
$$\frac{df}{dx} \neq \frac{f}{x}$$
```

Short HTML

```
<math>
<diff>x, f</> <NEQ> <frac>f, x</>
</math>
```

Example 8

$$\left(\frac{p}{q}\right) \neq \left(\frac{q}{p}\right)^{-1}$$

TEX

```
$$\left(\frac{p}{q}\right) \neq \left(\frac{q}{p}\right)^{-1}$$
```

Short HTML

```
<math>
(<frac>p, q</>) <NEQ> {{(<frac>q,p</>)}^-1}
</math>
```

Note: the left and right parantheses are seen as text and do not contribute to the logical structure. We probably need self-sizing parens (a la TeX)also : these would be a different element.

Example 9

$$\left[\begin{array}{cc} 0 & i \\ i & 0 \end{array} \right] \left[\begin{array}{cc} 0 & i \\ i & 0 \end{array} \right] = - \left[\begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array} \right]$$

TEX

```
$$ \left[ \begin{array}{cc} 0 & i \\ i & 0 \\ \end{array} \right]
\left[ \begin{array}{cc} 0 & i \\ i & 0 \\ \end{array} \right] =
- \left[ \begin{array}{cc} 1 & 0 \\ 0 & 1 \\ \end{array} \right]
$$
```

Short HTML

```
<math>
{       <matrix><mrow>0,i</><mrow>i,0</></matrix>
*<matrix><mrow>0,i</><mrow>i,0</></matrix>      }
=
-<matrix><mrow>0,1</><mrow>1,0</></matrix>
</math>
```

Example 10

$$\int_0^{\pi} \cos(\tfrac{1}{2}\cos x)\, J_0\left(\sqrt{0.5 - 0.25\cos^2 x}\right)\, dx$$

IP-HTML fully expanded form

```
<INT>x<SEP>0<SEP><M>&Pi;<SEP>
  <INFIX>
    <Cos><INFIX> 0.5 <TIMES><Cos>x</Cos> </INFIX> </Cos>
    <TIMES>
    <BesselJ0><sqrt>
      <INFIX>
        0.5
        <MINUS>
        <INFIX>0.25 <TIMES> <sqr><cos>x</cos></sqr></INFIX>
      </INFIX>
    </sqrt></BesselJ0>
  </INFIX>
</INT>
```

Note: `<SEP>` is a generic separator element which serves to separate the child elements of `<INT>`

IP-HTML using the shortref definitions

```
<INT>x, 0, &Pi; , {<Cos> {0.5 * <Cos> x </Cos> } </Cos> *
<BesselJ0><Sqrt>{0.5-{0.25 *
<SQR><Cos>x</Cos></Sqr>}}</Sqrt></BesselJ0>}</INT>
```

IP-HTML using the shortref definitions and some end-tag minimization (Standard SGML/HTML technique)

```
<INT>x, 0, &Pi; , {<Cos> {0.5 * <Cos> x  } </Cos> *
<BesselJ0><Sqrt>{0.5-{0.25 * <SQR><Cos>x}}</BesselJ0>}</INT>
```

TEX

```
$$\int_0^\pi\cos({\scriptstyle{1\over2}}\cos x)\,J_0\left(
\sqrt{0.5-0.25\cos^2x}\,\right)\,dx$$
```

Mathematica full form

```
Integrate[Cos[0.5*Cos[x]]*BesselJ[0,Sqrt[0.5-
0.25*Cos[x]^2]],{x,0,Pi}]
```

---

7 Representative Tags

There exist already several different sets of proposed tags for mathematical markup: that proposed by Dave Ragget for an earlier HTML proposal, that of the ISO 12083 working group, that proposed by the EUROMATH consortium.

In addition there is widespread use of the markup commands in various TEX packages, such as AMS-LATEX. The final set of tags will be selected taking the foregoing into consideration. The essential aspects of the current proposal (intermiscibility with other HTML markup, user extensibility for new tags, use of shortrefs for quasi-infix notation) are not affected by the precise set of math tags admitted.

We note that the set of tags will have to include a number of purely notational elements, e.g. diacritical marks, relative object placement, etc. Their composition is "semantic" in the sense that expression trees are produced, but not in the sense of mathematical semantics.

Macros can be used if desired to distinguish similar notations with different mathematical meaning (e.g. (xx/yy) for deriviative, quotient, Legendre symbol). This is a logical extensions of macros being used to form more consise notations in general (e.g. `<cos> x </>` *vs* `<fn> &cos; , x </>`).

To give an indication of the sort of composition tags we envision, we list the following representative set:

```
<infix> a <plus> b <minus> c</>
```
$a + b - c$

```
{a + b - c}
```
$a + b - c$

```
{{a + b} <over> {c + d}}
```
$\frac{a+b}{c+d}$

```
<sqrt> x </>
```
$\sqrt{x}$

```
a <space> b </> c
```
$a\,b\,c$

```
a <space before=10 after=4> 5 </> c
```
$a\;\;5\;\;c$

```
<plex> &Sigma;, {i=0}, &infty; , {{i+1} <over> {i-1}} </>
```

$$\sum_{i=0}^{\infty} \frac{i+1}{i-1}$$

```
<sum> {i=0}, &infty;, {{i+1} <over> {i-1}} </>
```

$$\sum_{i=0}^{\infty} \frac{i+1}{i-1}$$

```
<plex> &lim;, {h &rightarrow; 0},, {{x+h} <over> {x-h}}</>
```

$$\lim_{h \to 0} \frac{x+h}{x-h}$$

```
<lim> {h &rightarrow; 0}, {{x+h} <over> {x-h}}</>
```

$$\lim_{h \to 0} \frac{x+h}{x-h}$$

```
<fn> &cos; , (x) </>
```
$\cos(x)$

```
<cos> (x) </>
```
$\cos(x)$

```
<hat> x </>
```
$\hat{x}$

```
<bar> x </>
```
$\bar{x}$

```
<tilde> x </>
```
$\tilde{x}$

```
<vec> x </>
```
$\vec{x}$

```
<dot> x </>
```
$\dot{x}$

```
<ddot> x </>
```
$\ddot{x}$

```
<overline> {x + y} </>
```
$\overline{x+y}$

```
<underline> {x + y} </>
```
$\underline{x+y}$

```
<widehat> {x + y} </>
```
$\widehat{x+y}$

```
<widetilde> {x + y} </>
```
$\widetilde{x+y}$

```
<overbrace> n <sep> {x x &cdots; x} </>
```
$\overbrace{x\,x\cdots x}^{n}$

```
<underbrace> n <sep> {x x &cdots; x} </>
```
$\underbrace{x\,x\cdots x}_{n}$

```
<paren> b </>
```
(b)

```
( b )
```
(b)

```
<bracket> x </>
```
$[x]$

```
[ x ]
```
$[x]$

```
<brace> x </>
```
$\{x\}$

8 Representative Entities

Herein we include a "working" set of HTML entities. It should be noted that there exist a number of entity set definitions, for example : ISO 8879, ISO AMSO (Added Maths Symbols), ISO GRK1 (Greek letters). These, along with the AMS-Latex and WRI sets will be an input to our final definition.

Symbol	TEX	Entity
Å	\AA	&AA;
Æ	\AE	&AE;
ℵ	\aleph	ℵ
α	\alpha	α
⨿	\amalg	⨿
∠	\angle	∠
≈	\approx	≈
*	\ast	*
≍	\asymp	≈
\	\backslash	&backslash;
β	\beta	β
⋂	\bigcap	⋂
◯	\bigcirc	◯
⋃	\bigcup	⋃
⨀	\bigodot	⨀
⨁	\bigoplus	⨁
⨂	\bigotimes	⨂
⨆	\bigsqcup	⨆
▽	\bigtriangledown	▽
△	\bigtriangleup	△
⨄	\biguplus	⨄
⋁	\bigvee	⋁
⋀	\bigwedge	⋀
⊥	\bot	⊥
⋈	\bowtie	⋈
□	\Box	&Box;
•	\bullet	•
∩	\cap	∩
·	\cdot	ċ
χ	\chi	χ
∘	\circ	ˆ
♣	\clubsuit	♣
≅	\cong	≅
∐	\coprod	∐
∪	\cup	∪
†	\dag	&dag;
†	\dagger	†
⊣	\dashv	⊣
‡	\ddag	&ddag;
‡	\ddagger	‡
Δ	\Delta	Δ
δ	\delta	δ
⋄	\diamond	⋄
◊	\Diamond	⋄
♢	\diamondsuit	♦
÷	\div	÷
≐	\doteq	≐

Symbol	TEX	Entity
$\dots$	\dots	&dots;
$\downarrow$	\downarrow	↓
$\Downarrow$	\Downarrow	⇓
ℓ	\ell	ℓ
$\emptyset$	\emptyset	∅
ϵ	\epsilon	ε
$\equiv$	\equiv	≡
η	\eta	η
$\exists$	\exists	&exists;
$\flat$	\flat	♭
$\forall$	\forall	∀
$\frown$	\frown	⌢
Γ	\Gamma	Γ
γ	\gamma	γ
$\geq$	\ge	≥
$\geq$	\geq	≥
$\gg$	\gg	≫
$\hbar$	\hbar	ℏ
$\heartsuit$	\heartsuit	♥
$\hookleftarrow$	\hookleftarrow	↩
$\hookrightarrow$	\hookrightarrow	↪
$\imath$	\i	&i;
$\Im$	\Im	ℑ
$\imath$	\imath	ı
$\in$	\in	∈
∞	\infty	&infty;
$\int$	\int	∫
ι	\iota	ι
$\jmath$	\j	&j;
$\jmath$	\jmath	ȷ
$\bowtie$	\Join	&Join;
κ	\kappa	κ
Λ	\Lambda	Λ
λ	\lambda	λ
$\wedge$	\land	&land;
$\langle$	\langle	⟨
$\lceil$	\lceil	⌈
$\dots$	\ldots	&ldots;
$\leq$	\le	≤
$\leadsto$	\leadsto	&leadsto;
$\leftarrow$	\leftarrow	←

Symbol	TEX	Entity
$\Leftarrow$	\Leftarrow	⇐
$\leftharpoondown$	\leftharpoondown	↽
$\leftharpoonup$	\leftharpoonup	↼
$\leftrightarrow$	\leftrightarrow	↔
$\Leftrightarrow$	\Leftrightarrow	⇔
$\leq$	\leq	≤
$\lfloor$	\lfloor	⌊
$\lhd$	\lhd	&lhd;
$\ll$	\ll	≪
$\longleftarrow$	\longleftarrow	⟵
$\Longleftarrow$	\Longleftarrow	⟸
$\longleftrightarrow$	\longleftrightarrow	⟷
$\Longleftrightarrow$	\Long leftrightarrow	⟺
$\longmapsto$	\longmapsto	⟼
$\longrightarrow$	\longrightarrow	⟶
$\Longrightarrow$	\Longrightarrow	⟹
$\vee$	\lor	&lor;
‘	\lq	&lq;
$\mapsto$	\mapsto	↦
$\mho$	\mho	℧
$\mid$	\mid	∣
$\models$	\models	⊧
$\mp$	\mp	∓
μ	\mu	μ
∇	\nabla	∇
$\natural$	\natural	♮
$\neq$	\ne	≠
$\nearrow$	\nearrow	↗
$\neg$	\neg	&neg;
$\neq$	\neq	&neq;
$\ni$	\ni	∋
ν	\nu	ν
$\nwarrow$	\nwarrow	↖
Ø	\O	&O;
ø	\o	&o;
$\odot$	\odot	⊙
Œ	\OE	&OE;
œ	\oe	&oe;
$\oint$	\oint	∮
Ω	\Omega	Ω
ω	\omega	ω
$\ominus$	\ominus	⊖
$\oplus$	\oplus	⊕
$\oslash$	\oslash	ø
$\otimes$	\otimes	⊗
¶	\P	&P;
$\parallel$	\parallel	∥

Symbol	TEX	Entity
∂	\partial	&partial;
$\perp$	\perp	⊥
Φ	\Phi	Φ
ϕ	\phi	φ
Π	\Pi	Π
π	\pi	π
$\pm$	\pm	±
$\pounds$	\pounds	&pounds;
$\prec$	\prec	≺
$\preceq$	\preceq	⪯
$\prime$	\prime	′
$\prod$	\prod	∏
$\propto$	\propto	∝
Ψ	\Psi	Ψ
ψ	\psi	ψ
$\rangle$	\rangle	⟩
$\rceil$	\rceil	⌉
$\Re$	\Re	ℜ
$\rfloor$	\rfloor	⌋
$\rhd$	\rhd	&rhd;
ρ	\rho	ρ
$\rightarrow$	\rightarrow	→
$\Rightarrow$	\Rightarrow	⇒
$\rightharpoondown$	\rightharpoondown	⇁
$\rightharpoonup$	\rightharpoonup	⇀
$\rightleftharpoons$	\rightleftharpoons	⇌
$\rq$	\rq	&rq;
$\S$	\S	&S;
$\searrow$	\searrow	↘
$\setminus$	\setminus	∖
$\sharp$	\sharp	♯
Σ	\Sigma	Σ
σ	\sigma	σ
$\sim$	\sim	∼
$\simeq$	\simeq	≃
$\smile$	\smile	⌣
$\spadesuit$	\spadesuit	♠
$\sqcap$	\sqcap	⊓
$\sqcup$	\sqcup	⊔
$\sqsubset$	\sqsubset	⊏
$\sqsubseteq$	\sqsubseteq	⊑
$\sqsupset$	\sqsupset	⊐
$\sqsupseteq$	\sqsupseteq	⊒
ß	\ss	&ss;
$\star$	\star	☆
$\subset$	\subset	⊂
$\subseteq$	\subseteq	⊆

Symbol	TEX	Entity
$\succ$	\succ	≻
$\succeq$	\succeq	⪰
$\sum$	\sum	∑
$\supset$	\supset	⊃
$\supseteq$	\supseteq	⊇
$\surd$	\surd	&surd;
$\swarrow$	\swarrow	↙
τ	\tau	τ
Θ	\Theta	Θ
θ	\theta	θ
$\times$	\times	×
$\rightarrow$	\to	&to;
$\top$	\top	⊤
$\triangle$	\triangle	▵
$\triangleleft$	\triangleleft	◃
$\triangleright$	\triangleright	▹
$\trianglelefteq$	\unlhd	&unlhd;
$\trianglerighteq$	\unrhd	&unrhd;
$\uparrow$	\uparrow	↑
$\Uparrow$	\Uparrow	⇑
$\updownarrow$	\updownarrow	↕
$\Updownarrow$	\Updownarrow	⇕
$\uplus$	\uplus	⊎
υ	\upsilon	υ
Υ	\Upsilon	Υ
ε	\varepsilon	ϵ
φ	\varphi	ϕ
ϖ	\varpi	ϖ
ϱ	\varrho	ϱ
ς	\varsigma	ς
ϑ	\vartheta	ϑ
$\vdash$	\vdash	⊢
$\vee$	\vee	∨
$\Vert$	\Vert	‖
$\vert$	\vert		
$\wedge$	\wedge	∧
$\wp$	\wp	℘
$\wr$	\wr	≀
Ξ	\Xi	Ξ
ξ	\xi	ξ
ζ	\zeta	ζ
ζ	\zeta	ζ