# MathQTI Draft Specification

Manolis Mavrikis*
m.mavrikis@ed.ac.uk

3 Apr 2005

## 1 Introduction

This document (and the dtd, schema that accompanies it[1]) extends the IMS Question & Test Interoperability (QTI) specification[2] in an attempt to enable the exchange of questions with mathematical context between question engines and authoring tools. The motive behind this and the limitations of QTI have been described in previous version of this document[3] and in several meetings and workshops around this matter.

We assume here that the reader is familiar with QTI v2 and present the main extensions (or changes) which accommodate the existence of mathematical questions and more advanced response processing.

## 2 Including mathematical expressions

### 2.1 itembody

The presentation model of QTI v2 has improved and the final release includes the MathML schema allowing therefore one to use either content or presentation MathML. This was a significant improvement from the draft version and hence the discussion in our draft specification is not redundant. Nevertheless, the example from the MathQTI draft is repeated here to close the issue:

#### 2.1.1 Example

We provide an example here and further examples of the use of MathML will be made available as more examples of this specification appear.

---

*Thanks to everyone for their contributions and discussions and mostly Alberto González Palomo, Antony Maciocia, Georgi Goguadze, Helen Aston, Graham Smith, Dick Bacon, Steve Lay and all the members of the Serving Maths project.

[1]see http://www.maths.ed.ac.uk/mathqti/

[2]http://www.imsglobal.org/question/qti_v2p0/imsqti_infov2p0.html

[3]available at http://www.maths.ed.ac.uk/mathqti/documents.shtml

```
<itembody>
      <p>Find the derivative of the following function</p>
      <p><m:math>
          <m:semantics>
           <m:mrow>
            <m:mi>f</m:mi>
            <m:mfenced>
             <m:mi>x</m:mi>
            </m:mfenced>
            <m:mo>=</m:mo>
            <m:mi>x</m:mi>
            <m:mo>+</m:mo>
            <m:mn>1</m:mn>
          </m:mrow>
          <m:annotation-xml encoding="MathML-Content">
            <m:apply>
             <m:eq/>
             <m:apply>
               <m:ci type='fn'>f</m:ci>
               <m:ci>x</m:ci>
             </m:apply>
             <m:apply>
               <m:plus/>
               <m:ci>x</m:ci>
               <m:cn>1</m:cn>
             </m:apply>
            </m:apply>
          </m:annotation-xml>
        </m:semantics>
        </m:math>
     </p>
</itembody>
```

Other applications may prefer to annotate their expressions with LaTeX, for example:

```
<m:annotation encoding="TeX">f(x)=x+1</m:annotation>
```

or with the corresponding OpenMath.

## 3 Cloning for mathematical questions

The overall template mechanism and the `templateDeclaration` element is one of the most useful features of v2. Especially for mathematical questions randomisation is easier and leads to easily maintained items. On the other hand, in order to achieve its full functionality we also need the ability to construct constants (or *templateVariables* as they are called in QTI) using mathematical expressions.

We could, of course, adopt any syntax for these expressions (e.g Maple, C-like) but since the standard is XML based, using a string format for the expressions (e.g 2*x+4) would only complicate the situation. The use of a standard-based format will facilitate interoperability and maintenance through common acceptable tools. In this case though, because we are more concerned with the meaning of the mathematical expression rather than its visual representation we decided to use OpenMath. We believe that OpenMath and MathML can happily live together in the same specification as long as they address different goals. The use of OpenMath in this case seems a solution that will accommodate most (if not all) projects which need to do some mathematical processing whether this is advanced or just very trivial.

We appreciate that from one point of view, the choice of OpenMath, increases the complexity of the specification and its implementation but on second thought OpenMath's simplicity with respect to how many elements it involves is a great advantage. As we will describe later in this specification OpenMath's Content Dictionary mechanism can accommodate simple as well as advanced question engines.

### 3.1 Example

Lets start from an example. The question statement at "Digging a hole" sample (see the example `template.xml`) reads (a bit simplified):

```
 <p>If it takes <printedVariable identifier="A"/> men
<printedVariable identifier="MIN"/> minutes to dig a hole,
how long would it take <printedVariable identifier="B"/>
men to dig a  similar hole?</p>
```

where *A, MIN* are variables the values of which get substituted in the question. We believe that it is often easier, more general and helps reusability to construct such variables with proper mathematical expressions. For instance after randomising *A* (i.e the number of initial people) we could write the following :

```
<templateProcessing>
      ...
      <setTemplateValue identifier="B">
         <OMOBJ xmlns="http://www.openmath.org/OpenMath">
          <OMA>
            <OMS cd="arith1" name="times"/>
            <OMV name="A"/> <!-- A is declared before -->
            <OMI>2</OMI>
          </OMA>
         </OMOBJ>
      </setTemplateValue>
      ...
</templateProcessing>
```

The OpenMath expression would in effect substitute the corresponding `times` element of QTI.

### 3.2 Implementation

As already mentioned, there are quite a few (mainly algebraic) mathematical constructs in QTI v2. However, the use of a mathematical encoding would allow any kind of expression to be applied to the variables and enable more advanced processing engines to process them.

Let us emphasise here that although the use of the OpenMath expressions seems more verbose than the corresponding QTI one from an implementation point of view this is insignificant. The same (or similar) code that one has to write to process the expressions with QTI v2 elements has to be developed to process OpneMath expressions. The only difficulty arises from the fact that they belong in a different namespace. In fact, as more and more projects employ OpenMath the easier this will become. There are already some tools and more will be soon available. We strongly believe that the minor disadvantage of using a more verbose representation comes with the benefit of extensibility, scalability, reusability not to mention systems that can be pedagogically more sound.

Based on all the above MathQTI permits the use of OpenMath expressions for variables [4]. Engines that deliver MathQTI are responsible for dealing with the semantics of the mathematical objects and clarify the type of the variable and its use. In addition, one has to specify whether the OMOBJ elements that are encountered in the `setTemplate` are evaluated or not. It was decided that the default action would be to 'evaluate numerically' as this is the most common use. The attribute (eg. `evaluate` serve the purpose of clarifying the action. For now only `numerically` and `expression` are allowed but further values will be added if necessary.

## 3.3 Content Dictionaries (CDs) and CD Groups

Realising that there is a fear of risking how interoperable a specification which allows the use of arbitrary any mathematical expression is and in order to simplify this complexity we propose the use of appropriate Content Dictionaries.

Content dictionaries (CDs) are XML files in OpenMath that contain descriptions, mathematical properties, and examples of the concepts represented by the symbols in the CD to make their meaning unambiguous. This provides means of grouping the elements that one uses. For example, the OpenMath Society has developed a set of official CDs for central mathematical concepts and they contain a subset, that is isomorphic to the MathML concepts. This mechanism on its own is a powerful tool and adds to the benefits of employing OpenMath instead of (for example) Content MathML. We will emphasise here (as the OpenMath standard does as well) that it is not Content Dictionaries themselves which are being transmitted, but some "mathematics" whose definitions are held within the Content Dictionaries. This means that the applications must have already agreed on a set of Content Dictionaries which they "understand" (i.e., can cope with to some degree) and do not really have to be able to process the formal meaning which is described in the CD[5].

[4] We would like to extend QTI's type system to include an `omobj` type but with the current state of the XSD this seems very difficult and requires changes to the original QTI. Therefore variables that contain OpenMath expressions will not have a base type

[5] see Section 4 of The OpenMath Standard v2 at http://www.openmath.org/documents for more information

Content Dictionaries can be grouped in what is called a CD Group. As the OpenMath specification describes 'the CD Group mechanism is a convenience mechanism for identifying collections of CDs. A CD Group file is an XML document used in the (static or dynamic) negotiation phase where communicating applications declare and agree on the Content Dictionaries which they process. It is a complement, or an alternative, to the individual declaration of Content Dictionaries understood by an application'. The implication for MathQTI is that we can group CDs in a coherent manner. Systems which can understand certain groups will be able to share assessment items which contain mathematics, the semantics of which are described in the CDs of this group.

## 3.4 The mathematics of QTI v2

To convey our message a group of CDs with the mathematics that are allowed currently in QTI v2 is developed. The group is accessible here:

`http://www.maths.ed.ac.uk/mathqti/cdgroups/mathqti_basic.cdg`

it groups the `arith1, integer1, fns1, mathmltypes, relation1, set1` CDs some of which are redefined by adding the `mathQTI_` prefix removing the symbols and constructs that QTI v2 compatible engines would not be able to process.

In addition, we provide a CD Group which contains CDs that we believe represent a reasonable subset of mathematics that even not very sophisticated engines can achieve. The groups is accessible here:

`http://www.maths.ed.ac.uk/mathqti/cdgroups/mathqti_core.cdg`

Engines which "understand" this CD Group can understand the CDs described in it. These are basically the CDs that correspond to the same subset of mathematics that MathML targets. We expect systems such as AIM, STACK, WaLLiS, METRIC to be able to understand this core subject of mathematics.

In fact, we expect that the CD and CD Group mechanism will allow for more sophisticated engines to declare the mathematics that they understand and agree on other Content Dictionaries. This will allow the sharing of questions between these tools without having to define their own specification but relying on QTI (or mathQTI) for the overall structure.

## 3.5 Rendering issues stemming from cloning

As discussed in the forum[6] randomised question generation might, potentially, give rise to mathematical expressions like $x^5 + 0 * x^4 + 1 * x^2 + (-4) * x^1 + 3 * x^0$ which (depending on the circumstances) may or may not be desirable and some control should (perhaps) be allowed to authors and educators.

First of all, let us clarify here that although randomisation is a convenient mechanism for producing large numbers of similar items it should be used with

[6] http://mantis.york.ac.uk/serving_maths/mod/forum/discuss.php?d=321

caution. From a pedagogical point of view there may be some point in being really careful in what expressions are randomised and how. One cannot expect (at least with the current state of the specifications) that cloning will simplify authoring to that level. If an author wants more control over the output all they have to do is to be more careful with the randomisation process (eg. do not produce negatives or zeros) and provide different templates for different items. Although it is very tempting to use template for items that seem similar the measure of similarity should be considered twice. For example, $2x^2 - 3x$ and $2x^2$ are similar from a mathematical and template processing point of view but depending on the context and the question asked it could be argued that the first produces a more difficult questions than the second.

As the issue is not so trivial to resolve the approach that we adopted is that one should create expressions that he or she wants to simplify as OpenMath objects. When these objects are printed it will be up to the engine to simplify or not[7]. For example, the `factorise.xml` example includes a polynomial the coefficients of which are randomly generated. Since it is defined as an OpenMath object when this is printed the engine is in charge to handle the issue of how it will be displayed[8].

# 4 `responseProcessing` for mathematical questions

Similarly to the discussion for the template processing we allow the use of OpenMath expressions in the *responseProcessing*. Such use covers the need of performing more advanced tests and providing more detailed feedback to the student.

## 4.1 Numerical equivalence

Dick Bacon's proposal[9] for QTI v1.2 contained a lot of aspects some of which have been incorporated in QTI v2. In addition Graham Smith and Dick Bacon have been working towards making further changes to QTI v2 to convert the Maths/Psysics extensions of version 1 to cersion 2 particularly targeting the use of significant figures and decimal places to enable authoring of questions that test the ability of the student to round numbers to a given number of significant fiqures or decimal places. MathQTI incorporates the additional elements that they have introduced in order to ease interoperability and wide spread adoption. Examples of these will be made availabe and an implementation provided in

[7]In the future one could imagine a mechanism of controlling this process with appropriate attributes in the `printedVariable` element

[8]see http://mantis.york.ac.uk/tomcat/mathqti/out/factorise.jsp for the output of the http://mantis.york.ac.uk/tomcat/mathqti/xml/factorise.xml where the polynomial is printed. Refresh the page a couple of types to see different questions. The simplification is achieved through the use of YACAS as the backengine but simplifications can be achieved with simpler engines as well.

[9]http://www.scrolla.hw.ac.uk/qti/stmp_qvarv1p3.doc

due time. Meanwhile the reader can refer to DB's document and the CETIS Assessment-SIG extended examples for QTI v1.2 [10] by Graham Smith[11].

## 4.2 Syntactical equivalence

Apart from allowing the use of OMOBJ inside the response processing we are also adding a test for syntactical equivalence which is achieved by an element called synt_equal. The element accepts two parameters and returns true if the two expressions are syntactically the same. For example:

```
 <responseProcessing>
   <responseCondition>
      <responseIf>
         <synt_equal>
           <OMOBJ xmlns="http://www.openmath.org/OpenMath">
              <OMV name="RESPONSE"/>
           </OMOBJ>
           <OMOBJ xmlns="http://www.openmath.org/OpenMath">
           <OMA>
             <OMS cd="arith1" name="power"/>
             <OMV name="x"/>
             <OMI>3</OMI>
           </OMA>
           </OMOBJ>
         </synt_equal>
      </responseIf>
      ....
  <responseCondition>
</responseProcessing>
```

This checks if the learner's answer (i.e. *RESPONSE*) is $x^3$. Such an element eases other mathematical conditional tests and simplifies the QTI format even more. For example, to test the relation of a response compared to a number in QTI one has to do the following:

```
<responseProcessing>
   <responseCondition>
     <responseIf>
        <lt>
          <variable identifier="RESPONSE"/>
          <baseValue baseType="integer">5</baseValue>
        </lt>
     <responseIf>
   </responseCondition>
   ...
</responseProcessing>
```

in mathQTI:

```
<responseProcessing>
       <responseCondition>
```

[10]http://130.159.236.5/qthtm/cetisexamples3/extended/extended.htm

[11]Contact Graham for access to the latest engine and web site with more information on this issue

```
            <responseIf>
              <synt_equal>
                <OMOBJ xmlns="http://www.openmath.org/OpenMath">
                 <OMA>
                  <OMS cd="logic1" name="true"/>
                </OMA>
                </OMOBJ>
                <OMOBJ xmlns="http://www.openmath.org/OpenMath">
                  <OMA>
                    <OMS cd="relation1" name="lt"/>
                    <OMV name="RESPONSE"/>
                    <OMI>5</OMI>
                 </OMA>
              </OMOBJ>
            </synt_equal>
          <responseIf>
      </responseCondition>
        ...
</responseProcessing>
```

This seems more verbose but is equivalent and mathematically more sound. Most of the tests can be performed similarly instead of the QTI ad-hoc language. This approach renders elements such as `lt, lte, gt, gte` redundant but we will not remove them yet from the specification.

Once again, this approach helps reusability, maintenance and even authoring. Existing tools that export to OpenMath can help us write the expression which will be used for the response processing. In general this will also help in avoiding the need to rewrite tools for expressions in QTI's format.

More importantly, this approach also allows writing other, more complicated questions. Systems which support more advanced processing could even accept questions where the student's input is transformed before the test to conform to some mathematical property that the answer should have. For example,

```
   <responseCondition>
     <responseIf>
       <synt_equal>
          <OMOBJ xmlns="http://www.openmath.org/OpenMath">
             <OMA>
               <OMS cd="logic1" name="true"/>
             </OMA>
          </OMOBJ>
          <OMOBJ xmlns="http://www.openmath.org/OpenMath">
           <OMA>
             <OMS cd="relation1" name="eq"/>
              <OMA>
                <OMS cd="transc1" name="sin"/>
                <OMV name="RESPONSE"/>
              </OMA>
              <OMA>
                <OMV name="a"/>
              </OMA>
           </OMA>
          </OMOBJ>
        </synt_equal>
      </responseIf>
```

```
    ....
  <responseCondition>
```

which checks if the sin of the learner's answer plus one is equal to the value of the identifier a.

The way the test is conducted and (perhaps) algebraic transformations to the user's input (e.g simplification, factorisation) before testing are not (at least yet) the spec's concern but that of individual systems. In the future extensions could be defined to be more rigorous in these tests and allow interoperability between systems that can achieve such processing.

In the meantime simple QTI conformance would only require a system to comply with the basic QTI CD Group. This project will provide an implementation of this. On the other hand, more sophisticated systems would still conform to the specification while having more capabilities in terms of mathematical processing.

## 4.3 Other transformations and equivalence

Semantical equivalence and transformations to the user input are also very important for systems that have access to Computer Algebra Systems (like AIM, WaLLiS, ActiveMath). Therefore, MathQTI includes an element `sem_equal` for semantical comparison. For example the following XML fragment from the response processing:

```
<responseCondition>
    ...
    <responseElseIf>
        <sem_equal>
            <variable identifier="RESPONSE"/>
            <variable identifier="poly"/>
        </sem_equal>
        <setOutcomeValue identifier="SCORE">...</setOutcomeValue>
     </responseElseIf>
     ...
</responseCondition>
```

checks if the student's response is semantically equivalent to the OpenMath object declared at the declaration stage.

Finally, during workshops and meeting there were several ideas that were discussed for other kinds of transformations and equivalence. For example, it was agreed that pattern matching and regular expressions can be used effectively even to replace the need for syntactical equivalence which some times can be very strict and difficult to represent what an author wants. Fortunately the final release of QTI v2 supports regular expressions and provides an adequate (for the moment) solution. Of course writing effective regular expressions can be a daunting task. One would hope that simplifications particularly targeting mathematical expressions will be developed and perhaps ease this process[12]. Another approach would be to employ pattern matching on the MathML or OpenMath

[12] see http://mantis.york.ac.uk/serving_maths/mod/forum/discuss.php?d=325

representation of the user's input (using XPath expressions for example). This is of course not so trivial and beyond the scope of this project.

In addition, it was made clear (especially from the Sep. workshop) that we could employ the power of CAS more explicitly in the question format by using CAS queries represented in some language. The MONET[13] Query Language would be particularly useful. This is also left open to discussion for later versions of the specification and could be achieved by using external `responseProcessing` and some other element for queries that are not necessarily related to response processing (eg. template variables).

# 5 Inline Interaction Elements

In mathematical questions it is often the case that authors want to provide some structure for the students to give their reply. That is, instead of just asking the question and providing a blank box for the answer some questions are easier when set as in the following example:

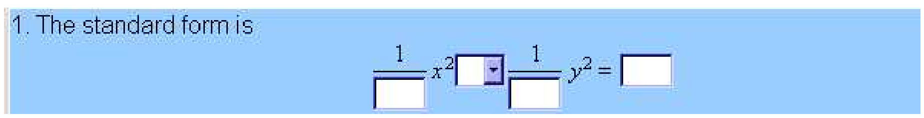


Figure 1: Part of an exercise which expects students to input the missing parts of an equation's standard form

Although it seems possible that something like this could be achieved with the current specification, the binding process to allow the interactive elements inside the mathematical encoding is really tedious. A solution that separates content and presentation helps authoring, reusability and maintenance. WaLLiS[14] and ActiveMath[15] formats achieve this by using an extra attribute called *for*[16] at the interaction elements. An example of how this is done follows.

For the expression in figure 1 the presentation MathML which corresponds to $\frac{1}{4}x^2 + \frac{1}{3}y^2 = 4$ is

[13] http://monet.nag.co.uk

[14] see http://www.maths.ed.ac.uk/wallis/format for details and publications

[15] description coming soon

[16] the name is not important, with *for* we mean 'for which' element the interaction element will take its position

```
<math>
  <mrow>
    <mfrac>
      <mrow>
        <mn>1</mn>
      </mrow>
      <mrow>
        <mn>4</mn>
      </mrow>
    </mfrac>
    <msup>
      <mrow>
        <mi>x</mi>
      </mrow>
      <mrow>
        <mn>2</mn>
      </mrow>
    </msup>
    <mo>+</mo>
    <mfrac>
      <mrow>
        <mn>1</mn>
      </mrow>
      <mrow>
        <mn>3</mn>
      </mrow>
    </mfrac>
    <msup>
      <mrow>
        <mi>y</mi>
      </mrow>
      <mrow>
        <mn>2</mn>
      </mrow>
    </msup>
    <mo>=</mo>
    <mn>4</mn>
  </mrow>
</math>
```

In order to provide the structure seen in figure 1 we want to leave blank the denominators, the operator and the right hand side. To achieve that we set *id*s to the elements that we want to leave as blanks when the question is presented[17]

```
<math>
  <mrow>
    <mfrac>
      <mrow>
        <mn>1</mn>
      </mrow>
      <mrow>
        <mn id="d1">4</mn>
      </mrow>
    </mfrac>
    <msup>
      <mrow>
        <mi>x</mi>
      </mrow>
      <mrow>
        <mn>2</mn>
      </mrow>
    </msup>
    <mo id="operator">+</mo>
    <mfrac>
      <mrow>
        <mn>1</mn>
      </mrow>
      <mrow>
        <mn id="d2">3</mn>
      </mrow>
    </mfrac>
    <msup>
      <mrow>
        <mi>y</mi>
      </mrow>
      <mrow>
        <mn>2</mn>
      </mrow>
    </msup>
    <mo>=</mo>
    <mn id="rhs">4</mn>
  </mrow>
</math>
```

The next step is to put the interaction layer over it.

```
<textEntryInteraction responseIdentifier="STD_FORM_1" for="d1"/>
<textEntryInteraction responseIdentifier="STD_FORM_2" for="d2"/>
<textEntryInteraction responseIdentifier="STD_FORM_3" for="rhs"/>
```

[17]whether the values of the elements to be replaced are emptied or not is not the specification's concern. Our suggestion is that leaving the original content intact helps reusing the expression with the least amount of editing

```
<choiceInteraction for="operator" responseIdentifier="STD_FORM_4">
  <simpleChoice identifier="PLUS">-</simpleChoice>
  <simpleChoice identifier="MINUS">+</simpleChoice>
</choiceInteraction>
```

This approach is valid XML and it allows for the mathematical content to be written easily by an equation editor and copied/pasted. It also allows for it to be reused later (even if the format changes) in an easier way. Finally, the proposed way provides enough flexibility for complicated questions that, for example, progressively diminish the difficulty of the formula presented by revealing parts of the expression. Of course, the proper output in these situations is a bit more complicated. Unfortunately, it is impossible to embed other languages in MathML. For example, it it invalid to add XHTML input elements as part of a MathML construct and hence it is not clear what would be the recommended way to achieve the effect described previously. It is expected that depending on the context, the delivery medium and the purposes of the project ad-hoc solutions can be found. For example, WaLLiS transforms MathML with embed XHTML to XHTML elements that maintain the mathematical structure. Others could use the `maction` element etc. It is hoped that this will be one of the issues addressed in following versions of MathML. Nevertheless, the proposed representation with the use of `for` attribute is general and implementation independent.

In fact, the solution with the *for* attribute is consistent with other questions as well. For example:

```
 <p>Now is the winter of our discontent<br/> Made glorious summer
by this sun of <span id=''blank1''>York</span>.<br/> And all the
clouds that ... </p>
                      ...
<textEntryInteraction responseIdentifier="RESPONSE"
                        for=''blank1'' expectedLength="10"/>
```

and again makes the text of the question more reusable than the case where it would be broken by the *textEntryInteraction* elements in between. Nevertheless *for* will be an optional attribute and it is up to individual authors if they use it or not.

## 6 Adaptive Items, Hints and other Events

The final release of QTI v2 enables the creation of adaptive items and the inclusion of hints and other events (eg. solutions) with using the `endAttemptInteraction` element. Although the solution does not adequately meet the expectations and the needs of mathematical exercises that ofter require multiple steps, adaptive hints etc. a solution that could conform both to QTI and take into account other approaches like ExerciseML or ActiveMath's exercise format is considered out of this project's scope.

Your comments with regards to all the issues discussed here are more than welcome.