

This is an ARCHIVAL DOCUMENT. Please see the current Project website: www.scribeserver.com/NEUMES/

The NEUMES Project *

Interim Project Report (IPR)

11 August 2002 (draft)

by

Louis W. G. Barton

University of Oxford

Software Engineer to the NEUMES Project †

TABLE OF CONTENTS

1. <i>The Project</i>	2
2. <i>Typographic Notes</i>	4
3. <i>Interim Goals</i>	5
4. <i>Simplified Transcription Format</i>	6
5. <i>NeumesXML Design Strategy</i>	8
6. <i>Remarks about the Transcription Examples</i>	9
7. <i>Transcription Example #1</i>	10
8. <i>Transcription Example #2</i>	11
9. <i>Relevant XML Tags from the Digital Scriptorium DTD</i>	12
10. <i>NeumesXML-Specific Tags</i>	15
11. <i>Other XML Tags from the Digital Scriptorium DTD</i>	23
12. <i>How to Read the Context-Free Grammar</i>	28
13. <i>The Neumes Context-Free Grammar in Mnemonics</i>	34
14. <i>Code Assignments in Unicode™</i>	40
15. <i>Eastern Chant Sources</i>	47
16. <i>Technical Documentation</i>	50
17. <i>References Cited</i>	57

* This Project is funded by a grant from the Andrew W. Mellon Foundation to Harvard University, Thomas Forrest Kelly, Principal Investigator.

Copyright © 2002 by the President and Fellows of Harvard College. Contains software or other intellectual property licensed from Louis W. G. Barton, copyright © 1995-2001 by Louis W. G. Barton. The Digital Scriptorium XML tag lists are copyright © 2001-02 UC Regents.

† This report includes emendations by John Caldwell, Jim Grier, Tom Kelly, Brad Maiani, and Clare McInerney.

1. The Project

‘NEUMES’ is an acronym for NEumed Unicode Manuscript Encoding Standard. The purpose of the Project is to design a data representation for digital transcription of medieval neumed manuscripts, and to create key software components that support the use of this encoding. The Project is fortunate to have received financial support from the Andrew W. Mellon Foundation in the form of a research grant to Harvard University (Thomas Forrest Kelly, Principal Investigator).

The Software Engineer for the Project is Louis Barton, who is responsible for the design and programming work, assisted by technicians working under his supervision. The Project’s Advisory Board, made up of medieval musicologists and computer scientists, is responsible for assuring that the work is sound from the point of view of their respective disciplines. Tom Kelly heads the Advisory Board.

The Project Advisory Board consists of the following members: Prof. Thomas Forrest Kelly, Principal Investigator (Department of Music, Harvard University); Prof. Louis W. G. Barton, Software Engineer (University of Oxford); Prof. John A. Caldwell (Faculty of Music, University of Oxford); Dr. Annalisa Doneda (Scuola di Paleografia e Filologia Musicale, University of Pavia, Italy); Prof. Ugo O. Gagliardi (Division of Engineering and Applied Sciences, Harvard University, retired); Prof. James Grier (Faculty of Music, University of Western Ontario); Prof. Andreas Haug (Institut für Musikwissenschaft, Universität Erlangen, Germany); Dr. Peter G. Jeavons (Computing Laboratory, University of Oxford); Dr. Brad Maiani (Technology Services, University of North Carolina–Chapel Hill); Dr. Ruth Ripley (Department of Statistics, University of Oxford).

The Mellon grant was awarded in September 2001 for a Phase One initiative to run for eighteen months, from October 2001 to March 2003. A very productive meeting of the NEUMES Advisory Board was held at Harvard in October 2001. Work on the Project halted until February 2002 due to a contract dispute between Barton and Harvard University; the March 2003 ending date, however, remains in effect.

As a practical matter, a benchmark of progress must be reached by August 2002, so that a Phase Two grant proposal can be submitted timely.

Currently assisting Barton on his technical team are Clare McInerney (XML design and coding), Vani Murthy (clerical assistance), and Barry Ng (graphics and Java programming). Additional technical input is being provided by Bob Garvey, Matthew Hartley, and Brad Maiani. Fernando Viesca of the Harvard Music Department is providing some hardware support for the Scribe server, on which the Project's software is hosted.

Barton has presented papers relating to the Project at the International Conference on Artificial Intelligence (*Knowledge Representation & Reasoning* session, June 2002) and the International Musicological Society 2002 Congress (IMS Study Group on Musical Data and Computer Applications, and Cantus Planus *Large-scale editions and reference tools* session, August 2002). He will also be presenting a paper at the International Conference on Web Delivery of Music 2002 (December 2002).

Additional information about the Project can be found on the Project's Web site at the following address: <http://scribe.fas.harvard.edu/NEUMES/>. [**Current website:** www.scribserver.com/NEUMES/]]

2. Typographical Notes

- Everywhere the name “Unicode” is mentioned, it should be understood as “Unicode™ Standard.” This name is a registered trademark of the Unicode® Consortium.
- Through out this document, square brackets ‘[]’ denote a Unicode™ character. Typically, a mnemonic for the character appears within the brackets, such as “[STA]” for the Start Neume character, but, in actuality, each such item refers to a Unicode character.

3. Interim Goals

Each musicologist on the NEUMES Project Advisory Board will transcribe one nocturn from the office of the Trinity in the Sarum antiphonal. (One advantage of this choice is that exists in the CANTUS database [1].) The purpose of this exercise is to test whether the current draft of the NEUMES data representation will be adequate.

I have written a context-free grammar (*see* definition in Section 13) for the current draft of the NEUMES data representation, which I shall continue to refine. We shall create a simple Java *applet* (executable over the Web in a Java-enabled browser) for data entry. Initially, this program will have pull-down, text menus for selecting options for each data ‘slot’ of NEUMES data. The output will be a text display (similar to the two Transcription Example #1 and #2), which can be printed from one’s browser. The immediate purpose is to help us evaluate the encoding scheme, rather than to make it easy for novices to enter data.

Simultaneously, we are working on an XML ‘wrapper’ (called *NeumesXML*) for transmission and storage of NEUMES data, and for descriptive *metadata* about the source and the transcription. The low-level NEUMES data are in Unicode, and the NeumesXML tags and chant texts are in ASCII. Our schedule calls for us to have an alpha-version of the transcription scheme ready in the coming months.

4. Simplified Transcription Format

4.1. Remarks

Here is a simplified pattern for NeumesXML documents. Items in angle-brackets ‘<’ ‘>’ are NeumesXML tags. Contiguous sequences of NEUMES data are always surrounded by the NeumesXML tags <neumes> and </neumes>; these tags serve to alert programs unable to handle NEUMES data that they should ignore everything between these tags.

Where the tag text is *italicized*, this text only describes what the tag (or sequence of tags) would be. An italicized *syllable* (like “*Syllable_A*”) stands for an actual chant-text syllable. The notion of “syllable” is meant to include phonemes (such as ‘repeated letters’ in Eastern sources) and syllables implied by abbreviated text (such as “*Euouae*”). In the simplified paradigm, below, syllables of a text are labelled sequentially as “*Syllable_A*”, “*Syllable_B*”, and so on.

Because a neume can consist of several neumatic symbols, the NEUMES characters [Start_Neume] and [End_Neume] are always used to mark a neume’s boundaries. A syllable can have one or more neumes. Where constructs such as “A.1.2” appear, “A” represents the current syllable; “1” the number of the neume within that syllable; and “2” the element of a compound neume. Neumed syllables are unambiguously separated by the NEUMES characters appearing between them in the data. Syllables may, optionally, be followed immediately by a hyphen or dash, which may serve as a textual separator between syllables. Syllables may appear without neumes in the case of declaimed text. Words are separated by the *space* character in the data.

The subscript “_{OPTIONAL}” means that this NEUMES character can appear in the data, but is not required. Ellipses ‘...’ denote that more data can continue in a similar pattern. The double-slash “//” starts a comment (until end-of-line).

The simplified transcription

```

<notational family and other descriptive information in NeumesXML>
Syllable_A
<neumes> // the <neumes> and </neumes> tags bound the NEUMES data
  [Start_Neume] // this is an example of a simple neume
    [Neume_Form#A.1OPTIONAL][Height#A.1.1OPTIONAL][Pen_Movement#A.1.1]
  [End_Neume]
  [Start_Neume] // this is an example of a compound neume
    [Neume_Form#A.2OPTIONAL][Height#A.2.1OPTIONAL][Pen_Movement#A.2.1]
    [Ligated][Height#A.2.2OPTIONAL][Pen_Movement#A.2.2]
    [Ligated][Height#A.2.3OPTIONAL][Pen_Movement#A.2.3]
  [End_Neume]
  [Start_Neume]
    [Neume_Form#A.3OPTIONAL][Height#A.3.1OPTIONAL][Pen_Movement#A.3.1]
  [End_Neume]
  ...
  [Start_Neume]
    [Neume_Form#A.nOPTIONAL][Height#A.n.1OPTIONAL][Pen_Movement#A.n.1]
  [End_Neume]
</neumes>
Syllable_B
<neumes>
  [Start_Neume]
    [Neume_Form#B.1OPTIONAL][Height#B.1.1OPTIONAL][Pen_Movement#B.1.1]
  [End_Neume]
</neumes>
Syllable_C
<neumes>
  ...
</neumes>
...
<NeumesXML tail information>

```

5. NeumesXML Design Strategy

A neumed transcription is an XML (Extensible Markup Language) file. Any XML file consists entirely of *tags* and *data*. Tags are delimited by angle brackets ‘<’ ‘>’, which are reserved in XML for this purpose. XML data typically are encoded as ASCII text, but XML also allows Unicode characters. (Actually, Unicode is the default encoding for XML documents.) NeumesXML files are coded in *UTF-8* (Unicode Text Format), such that tags and chant text are in ASCII, while NEUMES data are 24-bit values in Unicode, mostly in the Unicode Private Use Area. *UTF-8* reading is unambiguous.

XML is similar to HTML (the markup language of Web pages), except that XML is extensible whereas HTML is fixed. XML can be extended to suit the needs of particular types of documents. The mechanism of extension in XML is the DTD (Document Type Definition) or Schema, which specify the syntax of extension tags. Every XML document declares the URI (Uniform Resource Identifier) of the applicable DTD or Schema file; a URL (or, Internet address) can be used as the URI to specify the location of this file. Such files can be ‘cascaded’ so that one DTD or Schema can extend another.

XML tags are ‘human-readable’ when they are ASCII-coded and based on a natural language (usually English), e.g., “<handShift>”. ‘Readability’, however, depends on one’s understanding of the language. English is the *lingua franca* of computers, but perhaps Latin should be used for NeumesXML tags. (By contrast, NEUMES data represent concepts as Unicode characters without language-centricity.)

6. Remarks about the Transcription Examples

1. In Transcription Example #1 and #2, line breaks have been inserted only for legibility.
2. Programs that can read only ASCII text will see NEUMES data as gibberish. The NeumesXML tags `<neumes>` and `</neumes>` surround NEUMES data to alert such programs to ignore our data.
3. The characters '+' and '-' denote direction up and down. "**Ma**j" and "**mi**n" denote major and minor intervals, respectively. '[+l**i**t**t**l**e**]' is read as "up a little," and so on.
4. The character '[**h**]' denotes a b-natural.

7. Transcription Example #1

7.1. Source

This is the verse of the Gradual *Haec dies* when sung on Wednesday of Easter week. This Gradual is sung throughout the week beginning on Easter Sunday, but with different verses on each day.

7.2. Diplomatic facsimile of part of the source (in square neumes)



7.3. NEUMES data transcription (in mnemonics)


```
<XML header information>
Dex<neumes>[STA][Punctum][a][no preced][END]</neumes>
te<neumes>[STA][Punctum][c][+min3][END]</neumes>
ra<neumes>[STA][Pes/Podatus][d][+Maj2][LIG][e][+Maj2][END]</neumes>
space
Do<neumes>[STA][Climacus][e][eq/unison][d][-Maj2][c][-Maj2][h]
[-min2][END]
[STA][Climacus][c][+min2][a][-min3][G][-Maj2][END]
[STA][Clivis/Flexa][h][+min3][LIG][G][-min3][END]
[STA][Punctum][a][+Maj2][END]
...</neumes>
mi<neumes>...<neumes>
ni<neumes>...<neumes>
space
fe<neumes>...<neumes>
cit<neumes>...<neumes>
space
...
<XML tail information>
```

8. Transcription Example #2

8.1. Source

Same as Transcription Example #1.

8.2. Diplomatic facsimile of part of the source (in Aquitanian neumes)



 Dex- te- ra Do- mi- ni fe- cit

8.3. NEUMES data transcription (in mnemonics)

```

<XML header information>
Dex<neumes>[STA][Punctum][unk][no preced][END]</neumes>
te<neumes>[STA][Punctum][unk][+][END]</neumes>
ra<neumes>[STA][Pes/Podatus][unk][+little][LIG][unk][+][END]</neumes>
space
Do<neumes>[STA][Climacus][unk][eq/unison][unk][-][unk][-][unk][-][END]
[STA][Climacus][unk][+][unk][-][unk][-][END]
[STA][Clivis/Flexa][unk][+][LIG][unk][-][END]
[STA][Punctum][unk][+little][END]
...</neumes>
mi<neumes>...<neumes>
ni<neumes>...<neumes>
space
fe<neumes>...<neumes>
cit<neumes>...<neumes>
space
...
<XML tail information>

```

9. Relevant XML Tags from the Digital Scriptorium DTD

The Digital Scriptorium ('DS') project [2] is working on an XML DTD (Document Type Definition) for transcription and critical markup of medieval and early-Renaissance text manuscripts, as well as a description DTD for electronic library catalogs. The DS DTDs are intended to be 'compatible' with that of the well-known Text Encoding Initiative [4]. We are using the DS transcription tags list as a starting point for creating a NeumesXML tags list, since a consistent vocabulary for transcriptions will likely benefit users. Below are listed the DS transcription tags that we think might be useful in neumed transcriptions. Note that the DS convention for specifying *values* for tag *attributes* (cf., Section 10.2.) is currently to give an example rather than a formal definition.

```

<abbr> </abbr>           //abbreviation
  expan="Anno"           //can give full expansion in the expan attribute
<add> </add>            //addition or insertion, normally scribal
  rend="pencil"
  place="right"         //values: inline, supralinear, infralinear, left, right, top, bottom,
                        // opposite, verso, mixed.
  hand="C19"
<anchor />              //identifier one can attach to a point within a text, whether or not it
                        // corresponds with a textual element.
  rend="b"
  id="f56v18"
<authority> </authority> //name of person or other agent responsible for making an
                        // electronic file available, other than a publisher or distributor.
<cb />                 //column-boundary: marks beginning of a new column and gives the
                        // column number and total number of columns in the n attribute.
  n="1/2"               //indicates column #1 of 2.
<corr> </corr>         //correct form of a passage apparently erroneous in the copy text;
                        // the sic attribute gives the original form; cf. <sic>.
  sic="&longs;"
<damage> </damage>    //damaged portion of manuscript; may contain unclear passages

```

// (tagged `<unclear>`) or lacunae (marked with `<gap>`), as well as
// damaged-but-legible text.

`<dateRange>` `</dateRange>` //two dates or another phrase delimiting a time period.
 `from="1450"`
 `to="1480"`
 `exact="none"`

`` `` //deletion or cancellation, normally scribal.
 `rend="dot"`

`<div1>` `</div1>` //formal division of text (book, canto, section, chapter) larger than a
// paragraph [`<div1>` through `<div8>`].
 `type="section"`

`<docTitle>` `</docTitle>` //title of text on MS title page.

`<edition>` `</edition>` //particularities of one edition of a text [Appendix A].

`<editorialDecl>` `</editorialDecl>` //editorial principles and practices applied
// during the encoding of a text.

`<encodingDesc>` `</encodingDesc>` //documents relationship between an electronic
// text and its source(s).

`<expan>` `</expan>` //expansion of an abbreviation (typically used to mark up only
// letters not written as part of the abbreviation).

`<figDesc>` `</figDesc>` //description of a figure, for use when the image cannot be
// presented in graphic form, e.g. in a text-only interface.

`<figure>` `</figure>` //figure or graphic image, whether illumination, miniature, or
// diagram.

`<fileDesc>` `</fileDesc>` //full bibliographic description of an electronic file.

`<gloss>` `</gloss>` //a gloss in the text (enclosed in `<add>` when glosses are
// additions to the base text).

`<handList>` `</handList>` //in the `<teiHeader>`, lists hands in the MS.

`<handShift/>` //in transcription, marks point at which running text changes hands (do
// not use for additions and deletions in later hands: use the *hand* attribute
// on the relevant element).

`<idno>` `</idno>` //identifying number, in bibliographic citations in header.

`<lb/>` //line-boundary: marks beginning of a new line (where necessary to avoid
// confusion, or when a transcription is line-for-line) and optionally gives
// the line number in the *n* attribute.

`<note>` `</note>` //note or annotation; the *n* attribute can specify a number, and *target* can

```

// indicate the note's point of attachment to the electronic text.
<pb/> //page-boundary: marks beginning of a new page and gives the foliation in
// the n attribute; cf. <fw> for coding of old foliation.
    n="185r"
<principal> </principal> //name of the principal researcher responsible for creating
// an electronic text.
<profileDesc> </profileDesc> //detailed description of a text's non-bibliographic
// aspects, specifically the languages and sublanguages
// used, the situation in which it was produced, and the
// participants and their setting.
<publicationStmt> </publicationStmt> //information concerning publication or
// distribution of an electronic or other text.
<series> </series> //information about the series in which a book or other bibliographic item
// has appeared.
<seriesStmt> </seriesStmt> //information about the series, if any, to which a publication
// belongs.
<sic> </sic> //text reproduced as it appears in the source although apparently incorrect
// or inaccurate; the corr attribute gives a correction for the apparent error;
// cf. <corr>.
    corr="f"
<sourceDesc> </sourceDesc> //bibliographic description of the copy text(s) from which an
// electronic text was derived or generated.
<tagUsage> </tagUsage> //information about a specific element's usage within a text.
<teiHeader> </teiHeader> //descriptive and declarative information that make up an
// "electronic title page."
<xref> </xref> //reference to another location in the current document or in an external
// one using an extended pointer notation, possibly modified by additional
// text.

```

10. NeumesXML-Specific Tags

Below are listed the NeumesXML tags we expect to add to the list of Relevant XML Tags from the Digital Scriptorium DTD [2]. These NeumesXML-Specific Tags are intended to accommodate the special needs of neumed transcriptions.

To review, an XML file consists entirely of *tags* and *data*, where tags are differentiated from data by being set off with the reserved, angle-bracket characters ‘<’ and ‘>’. One can think of tags generally as containing information *about* the transcription, whereas the data are the transcription *per se*.

10.1. Design Decisions

So that search and comparison between transcriptions may be successful, we should reduce to a minimum any *redundancy* in the NeumesXML coding scheme. In other words, we should avoid giving transcribers the capacity to transcribe a unit of information in more than one manner.

We anticipate that some people might have occasion to use both text-only manuscript transcriptions in Digital Scriptorium format and neumed transcriptions in NeumesXML format. So that such users may have a consistent vocabulary, tag names that we invent should not be the same as any Digital Scriptorium tag (*see* Sections 9 and 11). Also, coding conventions used by the Digital Scriptorium initiative (such as use of “n” for “number”) should be followed unless there is good reason to do otherwise.

Toward a consistent vocabulary, attribute values (or, tag names) should, wherever possible, be compatible with *field* names used by CANTUS [1] or other conventional practices.

We should rigorously separate, when known, elements of Mass and Office. We should make the verse a sub-classification under the genre. Example:

```
type="Cm" //Communion
```

```
subtype="CmV" //Versus Communiois (Communion Verse)
```

Justification [Grier]: The verses are not subgenres of their own; they invariably belong to a genre. They tend to have an identity specific to the genre. So, we should tie the respective verses to the genres in which they function.

The term “repertory” is used in such a wide variety of ways, that we shall use “liturgical rite” instead in this taxonomy.

Interlinear material tends to be highly variable, and so the greater flexibility we have the better. We should provide separate categories for variants, glosses, and additions, plus allow interlinear variants of unspecified type.

10.2. Syntax

The NeumesXML tags in the Detailed Tag List (Section 10.4. , below) are aligned toward the left margin. Listed below many tags, and indented from the tag name, are one or more *attributes* (such as “type”). Each attribute is followed by an equals-sign, then a *value* that is allowable with this attribute. In actual tags, values are always in quotation marks. When an attribute is allowed only if some other attribute is present, then the dependent attribute is further indented. Example: the Tag List entry for,

```
<genre/>      //the liturgical role of the chant
  type="A"     //Antiphona (Antiphon; “An”)
    subtype="AaB" //Antiphona ad benedictus
```

would appear in an actual NeumesXML tag as, `<genre type="A" subtype="AaB"/>` .

Typically, an attribute is allowed to take one of many values. In the Tag List, this is indicated by repeating the *attribute-value* pair for each allowable value. Example:

```
<genre/>      //the liturgical role of the chant
  type="A"     //Antiphona (Antiphon; “An”)
  type="R"     //Responsorium (Responsory; “Re”)
```

Note, however, that a particular attribute can appear only once in an actual tag, and it can have exactly one value in that tag.

The following examples are permissible in a NeumesXML document:

```
<title>Haec dies</title>
<genre type="Gr" subtype="GrV"/>
<liturgicalRite type="ROM"/>
```

In XML, some tags come in pairs (i.e. a beginning tag and an ending tag) that surround *data*, whereas others appear singly (obtusely called *empty tags* in XML). Tags that appear singly are required to *end* in the forward-slash '/'. With paired tags, the ending tag has the same name as the beginning tag, except that the ending tag begins with a forward-slash '/'. Beginning and ending tags delineate the scope of a paired tag. The attributes of a paired tag always appear in the beginning tag; ending tags never have attributes. Example:

```
<initial color="red"/>H</initial>
```

Comments may appear after a tag declaration, preceded by a double-slash "//". Comments are in effect until the end of the current line. My requests for help appear in comments and are **bold** in brackets "[**help request**]" or are preceded by a "**Q:**".

I use three special place-holders for attribute values. " ? " indicates that the attribute value has not yet been defined and will require further discussion. The attribute "\rho" denotes that, in an actual tag, a text string would be inserted within the quotation marks. "\pi" denotes that, in an actual tag, a string of one or more digits would be inserted within the quotation marks.

Some tags have more than one attribute name listed at a given level of indentation. Example:

```
<genre/> //the liturgical role of the chant
  type="A" //Antiphona (Antiphon; "An")
    n="\pi" //\pi is a digit between 1 and 15 (e.g. for 5 psalm-antiphons)
    subtype="a" //antiphona unica super psalmos
```

where `n="\pi"` and `subtype="a"` are at the same level.

In a subsequent iteration, we shall need a more rigorous specification for these tags, in which we declare which attributes are *required* for each tag, and define rules governing the allowed or required combinations of attributes within a tag.

10.3. Brief Tag List

```

<differentia> </differentia>
<celebratio/>
<folioNumber/>
<genre/>
<incipit> </incipit>
<initial> </initial>
<link/>
<liturgicalRite/>
<neumes> </neumes>
<notationalFamily/>
<phrase> </phrase>
<rubric> </rubric>
<service/>

```

10.4. Detailed Tag List

```

<differentia> </differentia>      // differentia to be sung at the end of each Psalm verse.
// Typically occurs at end of Office Antiphons or the Psalm verse for the Introit of the Mass.
<celebratio/>
  nomen="ρ"    //ρ is a free-form, text identification of the liturgical celebration or feast day.
               // Usually this is a citation of the rubrical name of the feast if it appears in the MS,
               // or what the transcriber thinks the feast is when its name does not appear.
               // [Differences between rites and local practices make it infeasible to regularize the
               // celebratio identifications for purposes of search or comparison.]
<folioNumber/>
  value="ρ"    //ρ is a free-form identification of the folio number
//Q: Is this tag needed? If so, how is it different from <pb/> in Section 9, which "marks beginning
// of a new page and gives the foliation"?
<genre/>      // the liturgical role of the chant.
// Of the Office:
  type="A"    //Antiphona (Antiphon; alternate: "An")
  n="π"       //π is a digit between 1 and 15 (e.g. for 5 psalm-antiphons)
  subtype="a" //antiphona unica super psalmos
  subtype="a1" //antiphona prima super psalmos
  subtype="AaB" //Antiphona ad benedictus
               // [Same as "antiphona super canticum Benedictus"?]

```

subtype="AaE" //Antiphona ad evangelium
subtype="AaP" //Antiphona ad processionem
subtype="Ab+" //antiphona superaddita ad canticum
subtype="Am" //antiphona super canticum Magnificat
subtype="An" //antiphona super canticum Nunc dimittis
subtype="AsB" //Antiphona super benedicite
subtype="AV" //Versus Antiphonii (Antiphon Verse)
type="R" //Responsorium (Responsory; alternate: "Re")
n="π" //π is a digit between 1 and 30 (e.g. responsorium primum nocturni)
// (e.g., versus ad responsorium primum pertinens)
// Without **subtype**, the default is the Responsorium Prolixum (i.e., 'normal' responsory)
subtype="Rb" //Responsorium Breve (short reponsory)
subtype="RbV" //Versus Responsorii Brevis (verse of the short responsory)
subtype="RV" //Versus Responsorii Prolixi (verse of the 'normal' responsory)
type="W" //Versiculus (Versicle)
type="H" //Hymnus (Hymn)
type="I" //Invitatory antiphon
type="P" //Psalm (that is, the invitatory psalm, when it is written out in full or in substantial
//part with musical notation)
type="M" //Miscellaneous (chants in this group include the Te Deum, Versus in Triduo,
//prosulae, and the Mass chants that are on rare occasions incorporated into
//the Office)
// Of the Mass:
type="Ag" //Agnus Dei
type="Al" //Alleluia
subtype="AlV" //Versus ad Alleluia (Alleluia Verse, extra verse for Alleluia)
type="An" //Antiphona
subtype="Ap" //antiphona "pro populo" [Same as "antiphona processionalis"?]
type="AnP" //Psalmus Antiphonae
type="AP" //Antiphona ad processionem [Same as antiphona "pro populo"?]
subtype="APV" //Versus Antiphonae ad processionem (Procession Antiphon Verse)
type="BD" //Benedicamus Domino
type="Can" //Cantio
type="Cm" //Communion

```

    subtype="CmV" //Versus Communionis (Communion Verse)
    subtype="CmP" //Psalm verse for the communion
    subtype="CmR" //Versus ad repetendum for the communion
type="Cr" //Credo
type="Gl" //Gloria in excelsis Deo
type="Gr" //Graduale (Gradual)
    subtype="GrV" //Versus Gradualis (versus ad graduale, Gradual Verse)
type="In" //Introitus (Introit)
    subtype="InP" //Psalmus Introiti (Psalm verse for the Introit)
    type="InR" //Versus ad repetendum for the Introit
type="Ite" //Ite missa est
type="Ky" //Kyrie
type="Le" //Lectio
type="Let" //Letania
type="Of" //Offertorium (Offertory)
    n="π" //π is a digit between 1 and 15 designating the verse number
    subtype="OfV" //Versus Offertorii (Offertory Verse)
type="Ru" //Rubric (space for a rubric)
type="Sa" //Sanctus
type="Sq" //Sequentia
type="Tc" //Tractus
    n="π" //π is a digit between 1 and 15 designating the verse number
    subtype="TcV" //Versus Tractus (Tract Verse)
type="Tp" //Tropus
type="Va" //Varia
<incipit> </incipit> // typically the cue for the Psalm.
//Q: I'm not sure I have the idea quite right about Psalm incipits and differentia. Example:
// <genre type="In" subtype="InP"/><incipit> ... text of the Psalm incipit ...
// </incipit><differentia> <neumes> ... transcription of "EUOUAE" and NEUMES
// chant data ... </neumes></differentia>
<initial> </initial> // color/size of capitals/initials.
    color="black"
    color="blue"
    color="brown"

```

```

color="green"
color="orange"
color="purple"
color="red"
color="white"
color="yellow"
size=" ? " //Q: define.
<link/> // URL link to CAO, AMS, CANTUS, and so on, for texts, descriptions, or images.
<liturgicalRite/> // identifies the major Christian liturgy.
// Of Western rites:
type="BEN" // [Old] Beneventan
type="GALL" // [Old] Gallican
type="GREG" // Gregorian or Frankish (Carolingian-Roman)
type="MED" // Ambrosian (Milanese, i.e. Mediolanensis)
type="MOZ" // Mozarabic (Old Spanish, Visigothic)
type="ROM" // Old Roman (Urban Roman, or City Roman, etc.)
// Of Eastern rites:
type="BYZ" // Byzantine
type="COP" // Coptic
type="SLA" // Slavonic
<neumes> </neumes> // delineates NEUMES transcription data.
<notationalFamily/>
lines="lined" // number and ink color of lines is recorded in NEUMES data
lines="unlined" // no guidelines
// Western sources:
type="Ambrosian"
type="Anglo-Saxon"
type="Aquitanian"
type="Beneventan"
type="Bolognese"
type="Breton"
type="Catalan"
type="Central-Frankish"
  subtype="Chartres"
  subtype="Nevers"
  subtype="Norman"
  subtype="StBenigne"
type="Dasian"

```

```

type="German"
type="Hufnagel"
type="Messine"
type="Mozarabic"
type="Nonantolan"
type="Novalese"
type="Paleo-Frankish"
type="Square"
type="StGall"
type="Tied-Dot"
// Eastern sources:
type="MiddleByzantine"
type="NeoByzantine"
type="PaleoByzantine"
type="PaleoSlavonic"
<phrase> </phrase> // delineates a phrase; boundaries are a judgment call by the transcriber.
<rubric> </rubric> // everything that the original scribe writes that is not part of the music
// and its text, such as: labels for feast days, offices and masses; types of
// pieces; verses; and so on.
<service/>
// Of the Office: An abbreviation for the Office in which the chant is sung (CANTUS codes used)
type="V" //First Vespers
type="C" //Compline
type="M" //Matins
type="L" //Lauds
type="D" //Day Hours [kept for compatibility with the CANTUS scheme]
type="P" //Prime
type="T" //Terce
type="S" //Sext
type="N" //None
type="V2" //Second Vespers
type="E" //Antiphons for the Magnificat or Benedictus ("in evangelio")
type="H" //Antiphons or responsories based on texts from the Historia
type="R" //Memorial
type="X" //Supplementary chants

```

11. Other XML Tags from the Digital Scriptorium DTD

Below are listed the rest of the XML tags currently defined by the Digital Scriptorium transcription DTD [2]. We do not currently expect to use these. The list is reproduced here in case we later decide we need to use some of these, and to avoid naming conflicts with new NeumesXML tags.

```

<addspan/>    //beginning of a long textual sequence added by an author, scribe, annotator,
               // or corrector.

<argument> </argument>    //formal list or prose description of topics addressed by a
                           // subdivision of a text.

<author> </author>    //within bibliographic citations, contains the name of a work's author(s)
                       // (primary statement of responsibility).

<availability> </availability>    //information about a text's availability (e.g. restrictions
                                   // on its use or distribution, its copyright status).

<back> </back>    //appendices and such that follow a text's body.

<bibl> </bibl>    //bibliographic citations, including in the header.

<body> </body>    //whole body of a single unitary text, excluding any front or back matter.

<closer> </closer>    //groups together dateline, byline, salutation, and similar phrases that
                       // appear as a final group at the end of a division, esp. of a letter.

<correction> </correction>    //how and under what circumstances corrections have
                               // been made to the electronic text.

<creation> </creation>    //information about a text's creation (e.g. date;
                           // cf. <publicationstmt>).

<date> </date>    //a date in any format; the calendar attribute can specify the system in
                   // which the date is given, and the value attribute notes the date in a
                   // standard format.

<dateline> </dateline>    //brief description of the place, date, and time when a letter,
                           // newspaper story, or similar work was produced.

<delspan/>    //beginning of a long textual sequence marked for deletion by an author, scribe,
               // annotator, or corrector.

               type="X"
               to="f56v18"

<distributor> </distributor>    //name of person or other agent responsible for

```

```

// distributing a text.
<docAuthor> </docAuthor> //author of text on MS title page to be described.
<docDate> </docDate> // date of work on MS title page to be described.
<editionStmt> </editionStmt> // information relating to one edition of a text.
<editor> </editor> //secondary statement of responsibility for a bibliographic item
<epigraph> </epigraph> // quotation, anonymous or attributed, that appears at the
// start of a section or chapter or on a title page.
<extent> </extent> //approximate size of the electronic text as stored, specified in any
// convenient units (e.g. sentences, MB).
<foreign> </foreign> //word, phrase, or passage in a language other than the surrounding
// language; the language of the foreign material should be identified
// using the lang attribute; whole elements (whether paragraphs,
// text divisions, or glosses) in a foreign language need not be tagged
// <foreign>; instead, the lang attribute should be used on the element
// (<div>, <p>, <gloss>, etc.).
    lang="lat"
<front> </front> //prefatory matter (e.g. header, title page, preface, dedication) found
// before the start of a text proper.
<funder> </funder> //name of individual, institution, or organization responsible for funding
// a project or text.
<fw </fw> //("forme work") used to transcribe material not part of the running text;
// the type attribute indicates the kind of material:
    type="runhead" //running head.
    type="runfoot" //running foot.
    type="colhead" //column heading.
    type="fol" //old foliation (distinct values of the type attribute may be used if there
// are several old foliations: type="fol A", type="fol B", etc.)
<gap/> //marks a gap in the MS or in the transcription; distinguish cases using the desc and
// reason attributes.
    extent="0.5 inch"
    hand="HAND1"
    reason="erased"
<hand/> //in <teiHeader>, describes one or more hands in the MS.
    id="t1"
    hand="t1"

```

```

    scribe="Thomas London"
<head> </head>          //any heading, e.g. a section title, the heading of a list or a glossary.
<hi> </hi>             //text graphically highlighted in some way (e.g. rubrication, special ductus).
    rend="b"
<hyphenation> </hyphenation> //how an encoded text treats its source's hyphenation.
<interpretation> </interpretation> //scope of any analytic or interpretative
                                     // information added to the text aside from the
                                     // transcription.
<item> </item>         //one component of a list.
<l> </l>               //line of verse (i.e. the metrical line, not the physical line).
<label> </label>      //label associated with an item; used most commonly for the headwords
                                     // in glossary lists.
<language> </language> //characterizes a single (sub)language used within a text.
<langUsage> </langUsage> //languages, sublanguages, registers, and dialects represented
                                     // within a text.
<lg> </lg>            //group of verse lines: stanza, strophe, verse paragraph, or any metrical
                                     // unit larger than a line and smaller than a <div>.
    type="verse"
<list> </list>        //any sequence of items organized as a list.
<milestone/>          //boundary between sections of text.
<name> </name>        //proper noun or noun phrase.
<normalization> </normalization> //extent that the original source has been regularized in
                                     // being converted to electronic form.
<notesStmt> </noteStmt> //groups any notes providing information about a text additional
                                     // to that recorded elsewhere in the bibliographic description.
<num> </num>          //number (recommended in P-transcriptions to allow clean distinction
                                     // between numbers and other lexical items); the numeric value should
                                     // be given in the value attribute.
<opener> </opener>   //groups together dateline, byline, salutation, and similar phrases that
                                     // appear as a preliminary group at the start of a division, esp. of a letter.
<orig> </orig>        //original form of a reading; the reg attribute provides a regularized or
                                     // normalized form; cf. <reg>.
<p> </p>              //marks paragraphs in prose.
<projectDesc> </projectDesc> //aim or purpose for which an electronic file was

```

```

// encoded.

<ptr/> //pointer to another location in the current document in terms of one or more
// identifiable elements; cf. <ref>.

<publisher> </publisher> //name of the organization responsible for publication or
// distribution of a bibliographic item.

<pubPlace> </pubPlace> //name of the place where a bibliographic item was published

<q> //quotation or apparent quotation: representation of speech or thought marked as
// quoted from someone else (whether in fact quoted).

<quotation> </quotation> //editorial practice with respect to the original text's quotation
// marks.

<ref> </ref> //reference to another location in the current document in terms of one or
// more identifiable elements, possibly modified by additional text, e.g.
// for "see also" notes; cf. <ptr>.

    target="gloss2"

<reg> </reg> //reading which has been regularized or normalized in some way; the
// orig attribute gives the unregularized form; cf. <orig>.

<rendition> </rendition> //information about the intended rendition of one or more
// elements (presently only a hook for future DSSSL support).

<resp> </resp> //in statements of responsibility in header, phrase identifying the function
// of the individual(s) named.

<respStmt> </respStmt> //in bibliographic citations in header, statement of responsibility.

<restore> </restore> //canceled deletion (or portion of a deletion), i.e. a passage marked first
// as deleted, then later as not to be deleted after all.

<revisionDesc> </revisionDesc> //summary of file's revision history.

<s> </s> //sentence-like division of a text; can mark orthographic sentences or
// other segmentation provided as long as the segmentation is end-to-end,
// complete, and non-nesting; cf. <seg>.

<salute> </salute> //greeting prefixed to a foreword, dedicatory epistle, or other division of
// a text, or the salutation in the closing of a letter, preface, etc.

<samplingDecl> </samplingDecl> //prose description of the rationale and methods used in
// sampling texts when creating a corpus or collection.

<seg> </seg> // in transcriptions of text within diagrams, marks distinct segments of the text.

<segmentation> </segmentation> //principles according to which the text has been
// segmented (e.g. sentences, tone-units, graphemic strata).

```

`<signed> </signed>` //closing salutation appended to a foreword, dedicatory epistle, or
// other division of a text.

`<space> </space>` //abnormal white space, e.g. blank left for word not filled in or space
// left for image not drawn.

`<sponsor> </sponsor>` //name of sponsoring organization or institution.

`<stdVals> </stdVals>` //format used when standardized date or number values are supplied.

`<supplied> </supplied>` //text supplied by the editor in place of wholly illegible text.

`<tagsDecl> </tagsDecl>` //detailed information about the tagging applied to an SGML
// document.

`<TEI.2> </TEI.2>` //top-level tag containing a single TEI-conformant document that
// comprises a TEI header and a text.

`<text> </text>` //single text of any kind, whether unitary or composite.

`<title> </title>` //in bibliographic citations in header, contains a work's title.

`<titlePage> </titlePage>` //title page of a text found in the front or back matter

`<titlePart> </titlepart>` //subsection or division of a work's title as indicated on a
// title page.

`<titleStmt> </titleStmt>` //information about the title of a work and those responsible
// for its intellectual content.

`<trailer> </trailer>` //closing title or footer appearing at the end of a division or text.

`<unclear> </unclear>` //word, phrase, or passage which cannot be transcribed with
// certainty due to illegibility.

`<w> </w>` //in philological transcriptions, marks a unit which is to be treated as a word, but
// which might not be recognized as such by software relying solely on white-space.

`<xptr> </xptr>` //pointer to another location in the current document or an external one.

12. How to Read the Context-Free Grammar

12.1. General Remarks

A context-free grammar (CFG) is commonly used in computer science as the tool-of-choice for writing the formal specifications of a computer language or a complex data format. In particular, a CFG allows one to unambiguously specify a data format, in a manner that can be easily translated into computer programs for verifying well-formedness of data streams. Furthermore, a formal grammar is an important design tool, because it forces one to work through the details of the data representation clearly and in detail. Typically, flaws or ambiguities that might lie hidden in an English-language specification become obvious when the specification is written in a formal grammar.

The CFG that follows here defines the layout we discussed for the low-level data of neumed transcriptions. In this Section, my explanations for reading the CFG might seem intimidating for non-computer scientists, but CFGs are actually easy to follow once one gets the knack of it.

The low-level data representation might seem too complicated for practical use, but one should bear in mind that NEUMES data are not intended to be ‘human readable’. The detail and potential redundancy in the data will be handled by high-level software to insulate the user from this complication. The complication in the data is necessary (a) to be able to transcribe all neumed documents from *ca.* 800-1600 C.E. in a unified and consistent manner, and (b) to support both the analytical uses of the data and the visualization of a stylized facsimiles of the source documents.

Note that if a data file passes a test for well-formedness against the grammar, this does not, in itself, prohibit certain ‘impossible’ combinations in the data (for example, the appearance of a liquescent qualifier on the first tone of a compound neume). Such prohibitions must be expressed separately as a set of *rules* (or, *if-then* propositions). The set of rules applicable to a data stream might differ according to the notational family. So that fairly massive amounts of data can be processed efficiently, this more thorough,

rule-based checking should be done at data-entry time (viz., when the slowness of human interaction allows plenty of time for background processing).

12.2. Conventions Used in the Grammar

The notation I am using in this grammar is a hybrid of Backus-Naur form (BNF) and Z notation. Prior knowledge of these is unnecessary, as the following explanations cover all the relevant conventions.

Terms in narrow angle-brackets, e.g. “< neume specifier >”, are *non-terminals* in the grammar. That is to say, they receive further definition (called, *decomposition*) culminating finally in *terminal* characters. All terminal characters (called, *primitives*) in this grammar are Unicode™ characters.

☞ N.B.: Although both XML and BNF use narrow angle-brackets, these two usages are entirely different. Thus, non-terminals in the grammar, such as “< neume specifier >” should not be construed as NeumesXML tags. In the few places where a NeumesXML tag is mentioned in the grammar, it is written in bold within quotes, e.g., ‘ “<neumes>” ’.

The *production operator* “ ::= ” can be read as “consists of” (or, “is defined as”), such that the left-hand side (or, *LHS*) of the production gives the name of the non-terminal that is being defined, and the right-hand side (*RHS*) of the production gives the definition. Following each production I have added a plain English explanation in *italics*.

The vertical bar ‘ | ’ indicates an Exclusive-OR choice in a production. When a vertical bar is present in a production, the LHS is defined to be any *one* of the items on the RHS that are separated by the vertical bar. In the Neume Form, Interval, and Qualifier definitions there are various categories that (in the view of some users) may overlap. For example, one might wish to find occurrences of a melodic pattern notated in single-note neumes as opposed to compound neumes, regardless of whether a single-note neume is a Virga, Traculus, or Apostropha. This could be done by performing a search using a compound expression with the Boolean OR operator; for example, “[Virga] OR [Traculus] OR [Apostropha]” could produce a ‘hit’ on any one of these. To avoid repetitive typing or the risk of

forgetting, one could create ‘macros’ that expand automatically to their full form. For example, one might store “Single” as a macro name for the string “[Virga] OR [Traculus] OR [Apostropha]”.

The subscript “_{opt}” indicates an optional element that may, but need not, be present in that ‘slot’ of the data stream. If an optional element is absent, then it is just that, *absent* from the data stream; no ‘place holder’ is used. Spaces appearing in *productions* are non-significant, being used only for legibility.

The RHS can include wide angle brackets, “ $\langle \rangle$ ”, which denote a *sequence*. (This is the technical term used in computer science, and should not be confused with “sequence” in chant terminology.) In a sequence, the stuff between brackets can be repeated in the *production*; for example, “ $\langle \alpha \beta \rangle$ ” indicates that “ $\alpha \beta$ ” is allowed, as well as “ $\alpha \beta \alpha \beta$ ”, “ $\alpha \beta \alpha \beta \alpha \beta$ ”, and so on.

All *terminals* are Unicode characters, which I denote by square brackets “[]”. Inside the brackets may be a character mnemonic, or a character literal, or a Unicode value. For example, “[STA]” is a mnemonic for the Start Neume character in NEUMES encoding; it stands for an actual code in the Private Use Area of Unicode. A NEUMES code is a unique, binary pattern that cannot be confused with anything else in the transcription. Character literals are indicated in **courier bold** and are enclosed in single quotes; for example, [**a**] denotes the character ‘a’, in this case, an ASCII character.

Actual Unicode values are written in the standard notation for Unicode, such as “U+F8FF”. (‘U+’ indicates Unicode, and ‘F8FF’ is the hexadecimal value, or bit pattern, of the character.) There are 6,400 code points available for our use in the Private Use Area of Unicode. The code values in the Private Use Area range from U+E000 to U+F8FF.

Two clarifications may be needed concerning my use of *set* notation. Braces “{ }” denote a set, and the *members* of the set are enumerated within the braces. I use ‘ π ’ as a variable. The symbol ‘ \in ’ denotes *set membership*. And so, “ $\pi \in \{\mathbf{a}, \dots, \mathbf{z}\}$ ” means “an ASCII character between ‘a’ and ‘z’”; and, “[$\pi \in \{U+E000, \dots, U+F8FF\}$]” means “a Unicode character in the Private Use Area”; furthermore, “ $\langle \pi \in \{\mathbf{a}, \dots, \mathbf{z}\} \rangle$ ” means “a sequence of ASCII characters from the set of ‘a’ to ‘z.’” We have yet to *assign* (or, specify) a specific code values for the NEUMES characters.

Rules may be needed to define what constitutes a valid syllable (or phoneme in the case of ‘repeated letters’ in Eastern sources). The CFG merely asserts that there will be some string of ASCII characters in the ‘slot’ for chant syllable, but it cannot say what constitutes a *valid* syllable. If any check for validity is possible (such as dictionary lookup) it would be most efficiently done by the data-entry program.

12.3. Examples

The production,

$$\langle \text{neumes data} \rangle ::= \text{“}\langle \text{neumes} \rangle \text{”} \langle \text{neumes special character} \rangle \text{“}\langle / \text{neumes} \rangle \text{”}$$

should be read as follows:

The grammatical entity *neumes data* is defined as the NeumesXML tag $\langle \text{neumes} \rangle$, followed by a sequence of grammatical entities called *neumes special character*, followed by the NeumesXML tag $\langle / \text{neumes} \rangle$.

The production,

$$\langle \text{neume} \rangle ::= [\text{STA}] \langle \langle \text{abstract neume} \rangle_{\text{opt}} \langle \text{neume descriptor list} \rangle \rangle [\text{END}]$$

should be read as follows:

The grammar entity *neume* is defined as the NEUMES Unicode *Start Neume* character, followed by a sequence of pairs of grammatical entities called *abstract neume* and *neume descriptor list* (where the *abstract neume* is optional in each pair, and may be absent), followed by the NEUMES Unicode *End Neume* character.

12.4. Guidelines Used in Designing the Grammar

The grammar is organized broadly by ‘interruptions’ in the scribal writing process. Each interruption is caused by the scribe lifting his pen. We identify three types of interruption, in descending significance.

- (1) Pen lifting between syllables (also phonemes in the case of ‘repeated letters’ in Eastern sources) are the most significant breaks. The neumes chanted on a syllable appear in the data

- immediately after the syllable, and they end with the next syllable. Syllables are encoded as ASCII text, and so they cannot be confused with Unicode NEUMES data.
- (2) In the course of a single syllable there can be a sequence of neumes, where a *neume* is a logical entity (or, “musical gesture”) that always terminates in the lifting of the scribal pen. Neumes are recorded in the data stream from the end of one syllable to the start of the next (or to the end of transcription if it is the last syllable). Unchanted syllables simply appear in the data without any neumes between them. Whether there is a single neume or a sequence of neumes on a given syllable, these intra-syllable interruptions are always delineated in the data stream by the Start Neume [STA] and End Neume [END] characters.
- (3) Within one neume there can be a sequence of one or more neumatic symbols and special symbols, where each such symbol is written with an unbroken line of ink. We refer to these marks on the parchment generally as *glyphs*. Glyphs are identified taxonomically and recorded in the data stream as Glyph Forms.
- (α) For neumatic symbols, identification of the Abstract Neume is *optional*, when known.
- (4) A neumatic symbol can have several tones. For each tone the following is recorded:
- (β) an *optional* specification of the Pitch, or degree of the tonal scale, when known; and
- (γ) a *required* description of the direction of pen movement (up or down) from the previous tone or neumatic symbol, plus *optional* qualifiers and/or ligated-to-next indicator.

The encoding scheme should stay as close as possible to the *prima facie* contents of the parchment.

There is some confusion over whether Punctum and Virga should be encoded as distinct neumatic symbols, and/or whether one of them should be *qualifier*. I have chosen to code them as distinct symbols.

Finding a concise coding is not a design priority for this Project. The main rationale for using Unicode characters for Glyph Forms, rather than using XML tags (such as “<neume type=“virga”>”), is not to save space, but rather to enable efficient *complex traversal* of data. Whereas XML files must be processed sequentially from the beginning, “complex traversal” involves reading data in reverse order, in

‘random access’, and so on. By coding NEUMES data as Unicode characters in the Private Use Area, the meaning of each NEUMES character is unambiguous. In principle, disambiguation is not possible in XML without reading the file sequentially from the beginning to the point of inspection.

13. The Neumes Context-Free Grammar in Mnemonics

13.1. Notes

- For easier reading, this Section uses mnemonics (such as “[STA]”) to denote Unicode characters.
- All NEUMES characters are assigned to code points in the Private Use Area of the Unicode Standard.
- Character literals appear in single quotes ‘ ’, and string literals are in double quotes “ ”.
- Wide angle brackets, “⟨⟩” denote a sequence; viz., the item(s) within the brackets can repeat.
- Known to be missing from this CFG are characters for X-Y coordinates of a glyph in a digital photo.
- To-do list: staff; clef; custos; equaliter; letter marks.

13.2. The Grammar

< transcription > ::= ⟨< chant syllable > < neumes data >_{opt}⟩

Explanation: a transcription consists of a sequence of text syllables of the chant text, where each syllable may optionally be followed by neumes data.

< chant syllable > ::= ⟨ $\pi \in \{\text{space}, \text{' , '}, \text{' - '}, \text{' . '}, \text{' A '}, \dots, \text{' Z '}, \text{' a '}, \dots, \text{' z '}\}$ ⟩

Explanation: a chant syllable consists of a sequence of ASCII characters for a syllable of chant text. Characters can include the space character, comma, hyphen, period, and the letters ‘A’ through ‘Z’ in upper-case or lower-case.

< neumes data > ::= “<neumes>” ⟨< neumes special character >⟩ “</neumes>”
 | “<neumes>” ⟨< neume >⟩ “</neumes>”

Explanation: neumes data can consists either of a sequence of neumes special characters or a sequence of neumes. In either case, the sequence is surrounded by the XML tags “<neumes>” and “</neumes>” in ASCII.

(These XML tags are present to alert non NEUMES-capable programs to ignore the neumes data.)

< neumes special character > ::= [$\pi \in \{ [ij.], [iij.], [V], [Doh], [Fah], \dots \}]$

Explanation: a neumes special character is a NEUMES character that may be used for non-standard marks in the chant text (such as the “ij.”, “iij.”, and “V ”), or for staff and clef marks, or for other special symbols that lie outside the scope of neumatic symbols per se.

< neume > ::= [STA] < < abstract neume > _{opt} < neume descriptor list > > [END]

Explanation: a neume consists of a sequence bounded by the ‘Start neume’ and ‘End neume’ NEUMES characters. Each element of the sequence is an optional abstract neume followed by a neume descriptor list.

< abstract neume > ::= < neume form > < certainty factor > _{opt}

Explanation: an abstract neume is a neume form, followed optionally by a certainty factor (CF).

< neume form > ::=

A NEUMES character giving the neume’s abstract taxonomic identification.

[Punctum] | [Virga] | [Tractulus] | [Uncinus] | [Apostropha] |
[Gravis] | [Oriscus, detached]

Explanation: 1-note neume forms.

[Pes/Podatus] | [Clivis/Flexa] | [Pes quassus] |

Explanation: 2-note neume forms.

[Torculus/Pes flexus] | [Porrectus/Flexa resupina] |
[Trigon/Tripunctum] | [Pressus] |

Explanation: 3-note neume forms.

[Climacus] | [Scandicus] | [Salicus] |

[Pes/Podatus subpunctis] | [Porrectus/Flexa resupina subpunctis] |

[Torculus resupinus subpunctis]

Explanation: 3+ note neume forms, including Climacus and neume forms ending upward that have puncta going down afterwards.

[Porrectus flexus] | [Scandicus flexus] | [Torculus resupinus]

Explanation: 4-note neume forms.

< certainty factor >

::=

[CF 1.0] | [CF 0.9] | [CF 0.8] | [CF 0.7] |

Explanation: "Definitely is."

[CF 0.6] | [CF 0.5] | [CF 0.4] | [CF 0.3] |

Explanation: "Probably is."

[CF 0.2] | [CF 0.1] |

Explanation: "Might be."

[CF 0.0] | [CF -0.1] |

Explanation: "Completely uncertain."

[CF -0.2] | [CF -0.3] | [CF -0.4] | [CF -0.5] |

Explanation: "Might be wrong."

[CF -0.6] | [CF -0.7] | [CF -0.8] | [CF -0.9] |

Explanation: "Probably is wrong."

[CF -1.0]

Explanation: "Definitely wrong."

A NEUMES character specifying the certainty of transcription; it immediately follows the transcription element in question.

< neume descriptor list >

::= < < pitch >_{opt} < pen movement > < qualifier list >_{opt} < ligation >_{opt} >

Explanation: a neume descriptor list is a sequence, where each element in the sequence consists of an optional pitch specifier, followed by the obligatory pen movement specifier, followed optionally by a qualifier list, followed optionally by Ligation to the next glyph.

< ligation > ::= [LIG]

Explanation: character denoting ligation to the next glyph.

< pitch > ::= [Γ] | [A] | [B] | [C] | [D] | [E] | [F] | [G] |

[a] | [b] | [h] | [c] | [d] | [e] | [f] | [g] |

[aa] | [bb] | [hh] | [cc] | [dd] | [ee] | [ff] | [gg]

Explanation: a pitch specifier is a NEUMES character for a degree of the diatonic scale. ('h' denotes a b-natural.) [Note: The mnemonics correspond to Odo's (or, Guidonian) letter system. Some special cases need to be accommodated, e.g.: low B̄ indicated by 'B' in some MSS; 'quarter-tones' in Montpellier, (H. 159, Antiphonarium) 24/octave.]

< pen movement > ::= < interval > < certainty factor >_{opt}

Explanation: pen movement consists of an interval specifier plus an optional certainty factor.

< interval > ::= [no preced] | [unk] | [eq/unison] |

[+] | [+little] | [+lot] |

[+undifferentiated2] | [+min2] | [+Maj2] |

[+undifferentiated3] | [+min3] | [+Maj3] |

[+4] |

[+aug4] | [+dim5] //for cases in MS of B-F or F-b

[+5] |

[+undifferentiated6] | [+min6] | [+Maj6] |

[+undifferentiated7] | [+min7] | [+Maj7] |
 [+8] |
 [-] | [-little] | [-lot] |
 [-undifferentiated2] | [-min2] | [-Maj2] |
 [-undifferentiated3] | [-min3] | [-Maj3] |
 [-4] |
 [-aug4] | [-dim5] //for cases in MS of F-B or b-F
 [-5] |
 [-undifferentiated6] | [-min6] | [-Maj6] |
 [-undifferentiated7] | [-min7] | [-Maj7] |
 [-8]

Explanation: an interval is a NEUMES character specifying the apparent direction of scribal pen movement. [Note: more than one interval character could be used in a search- or compare-operation by combining them with a Boolean ‘OR’ operator, such as, “[-] OR [-undifferentiated3] OR [-Maj3]”.)

Q2: *Should we leave code space open for quarter-tones in case they are ever needed?*

< qualifier list > ::= < < qualifier > < certainty factor >_{opt} >

Explanation: a qualifier list is a sequence of qualifiers, each of which can optionally be followed by a certainty factor.

< qualifier > ::= [Liquescent] | [Episema] | [Quilisma] | [Oriscus] |
 [Angular/Quadratus] | [Substitute style] | [Strata/Stratus] |
 [Short stroke] | [Normal stroke] | [Long stroke]

Explanation: a qualifier is a NEUMES character that modifies the preceding pen movement in a glyph. [Note: More than one qualifier could be searched for or compared by means of a complex expression using the Boolean 'OR' operator.]

14. Code Assignments in Unicode™

14.1. Notes

- Actual Unicode™ character values are written in the standard notation for Unicode, such as “U+F8FF” (where ‘U+’ indicates Unicode, and ‘F8FF’ is the hexadecimal value, or bit pattern, of the character).
- Various musical symbols have recently been adopted in the Unicode Standard, version 3.1 [5], despite the Unicode Consortium’s stated position against including musical symbols in the Standard. Some of the new symbols are for ‘Gregorian chant’, and others for music generally (such as *staff*) have application to chant. The relevant codes are listed in Section 14.2. . It is the position of the Consortium that a given symbol should have only one codepoint in the *Standard*. (For example, English ‘a’ and French ‘a’ must be the same codepoint.) The set of chant symbols adopted by the Consortium is, however, entirely inadequate to this Project, and mixing their codes and ours is likely to cause confusion for programmers. I have decided to provide a full complement of symbols for our Project. If the Consortium decides at some time to adopt our symbol set, then the codes can be combined then.
- All NEUMES characters are assigned to codepoints in the Private Use Area of the Unicode Standard. The Private Use Area covers the range of values between U+E000 and U+E8FF (comprising 6,400 *codepoints*, or distinct bit patterns). Some care must be exercised to avoid conflicting uses of particular Private Use Area character values, for example by programs like Microsoft *Word* and for system use.

14.2. Existing Unicode™ Assignments for Symbols Used in Chant

Unicode version 3.1, defines a ‘block’ of musical symbols (code points U+1D100 through U+1D1FF, as defined in Section 12.11 of the Standard, *see* www.unicode.org/charts/PDF/U1D100.pdf).

This block includes some ‘Gregorian Chant’ symbols. Note that indicators of verticality and pitch representations are not covered by these additions to the Standard. They were adopted only so that musical symbols could be included in a line of text, without intending to represent a musical ‘score’.

The block description includes subsections on processing, input methods, directionality, format characters, precomposed note characters, alternative noteheads, augmentation dots and so on. We shall be examining these to determine to what extent these concepts might be reused in the NEUMES data coding.

Below are codepoints and symbol names for musical symbols currently defined in the Standard that are relevant to our work. The hexadecimal numbers in the left column should be read as Unicode values.

Note that these are 5-digit hexadecimal numbers rather than the usual 4-digits.

Bars

1D100	MUSICAL SYMBOL SINGLE BARLINE
1D101	MUSICAL SYMBOL DOUBLE BARLINE
1D105	MUSICAL SYMBOL SHORT BARLINE
1D112	MUSICAL SYMBOL BREATH MARK

Accidentals

1D12C	MUSICAL SYMBOL FLAT UP
-------	------------------------

Staves

1D116	MUSICAL SYMBOL ONE-LINE STAFF
1D117	MUSICAL SYMBOL TWO-LINE STAFF
1D118	MUSICAL SYMBOL THREE-LINE STAFF
1D119	MUSICAL SYMBOL FOUR-LINE STAFF
1D11A	MUSICAL SYMBOL FIVE-LINE STAFF
1D11B	MUSICAL SYMBOL SIX-LINE STAFF

Gregorian notation

1D1D0	MUSICAL SYMBOL GREGORIAN C CLEF
1D1D1	MUSICAL SYMBOL GREGORIAN F CLEF
1D1D2	MUSICAL SYMBOL SQUARE B
1D1D3	MUSICAL SYMBOL VIRGA
1D1D4	MUSICAL SYMBOL PODATUS
1D1D5	MUSICAL SYMBOL CLIVIS
1D1D6	MUSICAL SYMBOL SCANDICUS

1D1D7 MUSICAL SYMBOL CLIMACUS
 1D1D8 MUSICAL SYMBOL TORCULUS
 1D1D9 MUSICAL SYMBOL PORRECTUS
 1D1DA MUSICAL SYMBOL PORRECTUS FLEXUS
 1D1DB MUSICAL SYMBOL SCANDICUS FLEXUS
 1D1DC MUSICAL SYMBOL TORCULUS RESUPINUS
 1D1DD MUSICAL SYMBOL PES SUBPUNCTIS

14.3. Code Assignments for NEUMES Data

Specific NEUMES codepoint assignments will be made at a later date. We expect to use codepoints of the Private Use Area in the range U+E000 to U+E1DD (as has been done by others for cuneiform symbols). We hope to avoid conflicts in codepoint use by other programs that might be running simultaneously with NEUMES applications, especially *Windows* system routines and the Microsoft *Word* program. [Remark: the two code locations at the end of each plane are designated non-characters.]

Following the same order and layout as the context-free grammar (see Section 13), here follows a roster of symbols we expected to use for NEUMES data. In the right-hand column are given some temporary codepoints assignments for use by the Project technicians.

CFG non-terminal	mnemonic	description	Private Use Area
< neumes special character > ::=	[ij.]		
	[iij.]		
	[V]		
	[Doh]	U+1D1D0	
< neume >	[Fah]	U+1D1D1	
	::= [STA]	Start Neume	U+E000
< neume form >	[END]	End Neume	U+E001
	::= [Punctum]		U+E010
	[Virga]	U+1D1D3	U+E011
	[Tractulus]		
	[Unicus]		

	[Apostropha]		
	[Gravis]		
	[Oriscus, detached]		
	[Pes/Podatus]	U+1D1D4	U+E016
	[Clivis/Flexa]	U+1D1D5	U+E017
	[Pes quassus]		
	[Torculus/Pes flexus]	U+1D1D8	
	[Porrectus/Flexa resupina]	U+1D1D9	
	[Trigon/Tripunctum]		
	[Pressus]		
	[Climacus]	U+1D1D7	U+E01D
	[Scandicus]	U+1D1D6	
	[Salicus]		
	[Pes/Podatus subpunctis]	U+1D1DD	U+E020
	[Porrectus/Flexa resupina subpunctis]		
	[Torculus resupinus subpunctis]		
	[Porrectus flexus]	U+1D1DA	
	[Scandicus flexus]	U+1D1DB	
	[Torculus resupinus]	U+1D1DC	
< certainty factor >	::= [CF 1.0]		U+E030
	[CF 0.9]		
	[CF 0.8]		
	[CF 0.7]		
	[CF 0.6]		
	[CF 0.5]		
	[CF 0.4]		
	[CF 0.3]		
	[CF 0.2]		
	[CF 0.1]		
	[CF 0.0]		
	[CF -0.1]		
	[CF -0.2]		
	[CF -0.3]		
	[CF -0.4]		

		[CF -0.5]		
		[CF -0.6]		
		[CF -0.7]		
		[CF -0.8]		
		[CF -0.9]		
		[CF -1.0]		
< ligation >	::=	[LIG]	ligated to next	U+E005
< pitch >	::=	[Γ]		U+E050
		[A]		
		[B]		
		[C]		
		[D]		
		[E]		
		[F]		
		[G]		U+E057
		[a]		U+E058
		[b]		U+E059
		[h]	square B (natural)	U+E05A
		[c]		U+E05B
		[d]		U+E05C
		[e]		U+E05D
		[f]		U+E05E
		[g]		U+E05F
		[aa]		U+E060
		[bb]		
		[hh]	square B (natural)	
		[cc]		
		[dd]		
		[ee]		
		[ff]		
		[gg]		
< interval >	::=	[no preced]	no preceding tone	U+E070
		[unk]	unknown tone	U+E071
		[eq/unison]	equal tone or unison	U+E072

[+]	up	U+E073
[+little]	up a little	
[+lot]	up a lot	
[+undifferentiated2]		
[+min2]		U+E077
[+Maj2]		U+E078
[+undifferentiated3]		
[+min3]		U+E07A
[+Maj3]		U+E07B
[+4]		
[+aug4]	up augmented 4th	
[+dim5]	up diminished 5th	
[+5]		
[+undifferentiated6]		
[+min6]		
[+Maj6]		
[+undifferentiated7]		
[+min7]		
[+Maj7]		
[+8]	up an octave	
[-]	down	U+E085
[-little]	down a little	
[-lot]	down a lot	
[-undifferentiated2]		
[-min2]		U+E089
[-Maj2]		U+E08A
[-undifferentiated3]		
[-min3]		U+E08C
[-Maj3]		U+E08D
[-4]		
[-aug4]	up augmented 4th	
[-dim5]	up diminished 5th	
[-5]		
[-undifferentiated6]		

		[-min6]	
		[-Maj6]	
		[-undifferentiated7]	
		[-min7]	
		[-Maj7]	
		[-8]	down an octave
< qualifier >	::=	[Liquescent]	
		[Episema]	
		[Quilisma]	
		[Oriscus]	
		[Angular/Quadratus]	
		[Substitute style]	
		[Strata/Stratus]	
		[Short stroke]	short pen stroke
		[Normal stroke]	normal pen stroke
		[Long stroke]	long pen stroke

15. Addenda for Eastern Chant Sources

15.1. Notes

- This section expands text and NEUMES definitions to include Eastern chant, principally Byzantine and Slavonic sources [2].
- Please see Section 12 for an explanation of the typographic conventions used here.
- Here ‘z’ is used as a variable instead of ‘π’.

15.2. Addenda to the Context-Free Grammar

< chant syllable' > ::= < chant syllable > ∪
 < z ∈ {Greek character set} >

Explanation: the definition of chant syllable' is the union of the previous definition of a chant syllable and a sequence of Greek characters. These are Unicode characters for a syllable of chant text or a phoneme in the case of a 'repeated 'letter'.

[Note: Annalisa, please define the set of characters needed to cover Greek and Slavonic texts, and identify which letters can be 'repeated'.]

< neume form' > ::= < neume form > ∪
 [Antikenokilisma] | [Antikenoma] | [Apoderma] | [Apostrophoi] |
 [Apostrophoi Chamele] | [Apostrophos] | [Apostrophos Chamele] |
 [Apostrophos Chamele Petaste] | [Apostrophos Elaphron] |
 [Apostrophos Elaphron Petaste] | [Apostrophos sopra Oxeia] |
 [Apostrophos sopra Petaste] | [Argon] | [Bareia] |
 [Chamele Apostrophoi] | [Chamele Apostrophos] | [Diple] |

[Dyo Kentemata] | [Elaphron] | [Elaphron Apostrophoi] |
[Elaphron Apostrophos] | [Elaphron Apostrophos Petaste] |
[Elaphron Petaste] | [Gorgon] | [Hyporrhoe] | [Hyporrhoe nel Seisma] |
[Hypsele Kentema Oligon] | [Hypsele Oligon] |
[Hypsele Oligon Oxeia] | [Hypsele Oxeia] | [Hypsele Petaste] | [Ison] |
[Ison sopra Kentema sopra Oligon] |
[Ison sopra Kentema sopra Petaste] | [Ison sopra Kouphisma] |
[Ison sopra Oxeia] | [Ison sopra Petaste] | [Kentema] |
[Kentema Oligon Petaste] | [Kentema sopra Oligon] |
[Kentema sopra Oxeia] | [Kentema sopra Petaste] | [Kouphisma] |
[Kratema] | [Kylisma] | [Oligon] | [Oligon e Kentema] |
[Oligon e Oxeia] | [Oligon e Petaste] | [Oligon Hypsele] |
[Oligon Kentema Oxeia] | [Oxeia] | [Oxeia e Kentema] |
[Oxeia Hypsele] | [Parakalesma] | [Parakletike] | [Petaste] |
[Petaste e Kentema] | [Petaste Hypsele] | [Piasma] |
[Piasma e Tzakisma] | [Psephiston] | [Strepton] | [Synagma] |
[Tromikon] | [Tzakisma] | [Xeron Klasma]

Explanation: the definition of neume form ' is the union of the previous definition of a neume form and a list of NEUMES characters for Eastern chant notation.

Q1: *Annalisa, can you arrange the neumes more hierarchically, and perhaps simplify the taxonomy by using qualifiers?*

Q2: *Annalisa, does the range of pitches for Western notation suffice for Eastern sources?*

15.3. Code Assignments for NEUMES Data

Byzantine Musical Symbols (codepoints U+1D000 through U+1D0FF) are described in the new Section 12.10 of the Unicode™ Standard, version 3.1 (*see* www.unicode.org/charts/).

16. Technical Documentation

16.1. XML Schema versus XML DTD

As an extensible markup language, XML must have a means by which to specify the grammars of particular implementations of XML such as NeumesXML. An XML document is ‘validated’ (in software engineering terms, *verified*) for well-formedness against the specified grammar. Each XML document declares which grammar it is intended to conform to. This declaration is made near the top of the XML document as a Uniform Resource Identifier (URI), often in the form of a Uniform Resource Locator (URL), viz., an Internet address. These grammar specifications can be *cascaded*, so that one grammar can refer to another grammar that it extends.

The older mechanism for grammar XML specification is the Document Type Definition (DTD). A DTD specifies which tags are allowed in an XML document and the allowable format of these tags, including allowable *attributes* and *values* (see Section 10.2.). Since XML is a *self-representing language*, DTDs are written in XML.

Notably missing from DTD capabilities is the capacity to enforce grammar rules about the *data* between tags. *Data type checking* is important for ensuring the robustness of files that transport data in XML. Largely to satisfy this need, the XML Schema mechanism [6] was developed as an alternative to DTD. The capabilities of XML Schema are a (non-strict) superset of the DTD capabilities, and Schemas are also written in XML. Nevertheless, for practical purposes DTD and Schema are incompatible; to migrate between them, existing data files would need reformatting, which could become a messy process.

For the purpose of avoiding duplication of effort in creating transcriptions, an early design goal of the NEUMES Project was to make NeumesXML files, in as much as possible, compatible with text-only manuscript transcriptions written under the Digital Scriptorium (DS) transcription DTD [2]. In particular, it was hoped that DS-format transcriptions could easily be imported to NeumesXML programs, and that

NeumesXML documents could somehow be readable by DS-transcription ‘applications’. Barton and McInerney explored various options for realizing ‘compatibility’ with DS, as follows.

(1) The head of the DS transcription initiative, Dr. Charles Faulhaber, was approached with the idea of converting the DS DTD to Schema format. This line of inquiry has gone nowhere, presumably because part of DS’s mission is to remain compatible with the Text Encoding Initiative (TEI) [4], which seems to be firmly in the camp of using DTDs and ASCII-only transcription.

(2) Various ideas were unsuccessfully explored for declaring in the DS DTD that the `<neumes>` `</neumes>` tags should be treated like comment tags, so that all the NEUMES data would be ignored by any DS transcription ‘applications program’.

(3) The currently-available DS transcription software was downloaded and tried. It turns out that the dataflow for a DS transcription document is as follows.

- (a) The user accesses a transcription on the Web using her/his browser.
- (b) The browser has a built-in XML parser, which ‘validates’ the transcription against its declared, DS transcription DTD.
- (c) If the document is ‘valid’, then the browser passes the file to an XML stylesheet (XSL), which formats the transcription as an HTML document.
- (d) The resulting HTML document is displayed in the browser.

We discovered that there are no DS-transcription ‘applications programs’, and specifically DS does not currently have software that would allow one to view the DS XML tags in a transcription file easily in one’s browser, much less to manipulate them.

After lengthy consideration, a decision was made that the NeumesXML grammar should be specified by an XML Schema, rather than an XML DTD. If NeumesXML is defined by a Schema, NEUMES data sequences (i.e., the “low-level” data) can be automatically checked for well-formedness against our context-free grammar (which is outlined in Section 13). In particular, complex data types can be defined (*see especially the NEUMES data relationships discussed in Section 12.4.*), and constraints can be

enforced on data *values*. By using the more modern, Schema technology, our work might set an example for other transcription initiatives, so that they would eventually adopt the Schema methodology as well.

16.2. Detailed Justification for Using XML Schema

Our intent is that the NEUMES Unicode-character encoding scheme (viz., the “low-level data” for *prima facie* content of manuscripts) should be quite stable, or ‘fixed’. To promote interoperability of these data across time, computer platforms and applications programs, users should be prevented (not merely discouraged) from making custom changes to the encoding scheme. This scheme involves not only the assignment of *codepoints* in the Private Use Area of Unicode, but also syntactic relationships between NEUMES data as they appear in transcription sequences. By contrast, Unicode generally has no capability for checking that character sequences are syntactically well-formed. For example, the string “+\$+;-p”, *qua* nonsense, is permissible in Unicode. Without the capability to enforce the NEUMES grammar, there are likely to be ill-formed or extended-representation transcriptions floating around, with the result that transcription interoperability will break down.

A fundamental notion in my design theory for the Project is that the NEUMES scheme is a *language*. For a string to be in the language, (a) it must be well-formed according to a normative grammar, and (b) it must correspond to a real-world sequence of symbols in an actual manuscript. XML Schema has a mechanism for data type checking that can enforce constraint '(a)'. There is compelling motivation to use '(a)'-type checking before any further processing of a transcription file can be done.

An alternative to XML Schema is to do the grammar checking ‘off-line’, that is, to write a parser program (likely in Java) that users would invoke to check their transcription files for well-formedness. The separate step of grammar checking would, therefore, be optional for users and would likely not be carried out in most cases. Having a separate program would also open the door to others writing variants that would defeat the goal of having a standard. Furthermore, the parser program would be proprietary to us and (even if it were made freely available) would introduce unwanted dependencies.

Doing the grammar checking at the XML layer, however, has a potential drawback. If the grammar specification (which should be ‘fixed’) is mixed with the NeumesXML tag specification (which we intend to be flexible) in a single Schema file, then users (who are invited to extend the tag specification) will find it easy to also modify the grammar specification. Likely the best workaround for us is to write two Schema files, one for the NEUMES grammar and another for the NeumesXML tags. These would *cascade*, such that the tags-definition Schema would inherit (or, refer to) the NEUMES grammar Schema.

Secondary benefits of using Schema are that : at least to me, Schema code reads more cleanly than DTD code; Schema has Java-compatible data types, which will make some programming tasks easier, as our applications programming is being done in Java; a utility is available for Schema data import/export with MS *Access*, which seems to be the database system of choice among medieval musicologists; and in general, Schema seems to be the proverbial ‘wave of the future’, with much more support for it likely to become available in Web browsers and so on.

16.3. Attaining Digital Scriptorium Compatibility with NeumesXML

Despite our decision to specify NeumesXML as a Schema, it still seems possible to share text transcriptions between DS and NEUMES.

(1) Conversion of files from NEUMES format to DS format by a utility program should be fairly straightforward. The program would strip out all NEUMES data by deleting everything between the `<neumes>` `</neumes>` tags, then delete all non-DS tags, and replace the header information with the appropriate DS information.

(2) Importing DS files to NEUMES programs could be even simpler, as only the header information might need to be changed. We must make a design decision about whether we will allow the DS tags that we are not using (*see* list in Section 11) to appear in a NeumesXML document. If we allow these tags, then no further conversion is necessary. If they are not allowed, then we must investigate whether there might be, from our viewpoint, any side-effects to simply stripping these tags out.

These conversion utilities could be run offline by users, and might be provided free-of-charge for downloading from the NEUMES Web site.

It may be worthwhile to continue exploring means by which we could effect file conversions more transparently from the user's viewpoint. DS currently uses an XML stylesheet (XLS) to create a visualization of transcription files. Unfortunately, XLS offers no capacity for programming *conditional* (or, branching) instructions. A new technology called Extensible Stylesheet Language Transformations (XSLT) appears to allow remote procedure calls during dynamic building of a parse tree. We could provide Java routines on the Scribe server that could be invoked by remote procedure call from an XSLT document. Once control has been passed to a Java program, any type of conversion operation would be feasible. It may be that the DS stylesheet might allow us a way to 'hook' to an XSLT Transformer. The XSLT declaration in the XML document would be performing the transformation; if the stylesheet declaration were removed, one would be able to view the document without transformation.

16.4. NeumesXML Header Information

An actual NeumesXML transcription file begins with heading information that specifies the encoding scheme, an XML Stylesheet (for transformation of the XML document for Web display), and the NeumesXML Schema. The code is as follows [McInerney].

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl"
href="http://hul.harvard.edu/~clare/neumesXML/NeumesXML.xslt"?>
<NeumesXML xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://hul.harvard.edu/~clare/neumesXML/NeumesX
ML.xsd">
```

A NeumesXML transcription file ends with,

```
</NeumesXML>
```

Note that the path given here (<http://hul.harvard.edu/~clare/>) will be replaced by the Scribe server path to the NEUMES Project (<http://scribe.fas.harvard.edu/NEUMES/>) **[this URL is now outdated]**.

16.5. NeumesXML .pen File

The Schema should refer to NEUMES codepoints as entity names. These entities should be defined in a separate *.pen* file. By using names instead of Unicode values in the Schema, it is easier to maintain. If we decide to change the NEUMES Unicode assignments during development, one simply has to modify the *.pen* file. A *.pen* file simply consists of a list of entity definitions, for example,

```
<!ENTITY STA "&#E000;" ><!-- [STA] Start Neume -->
<!ENTITY END "&#E001;" ><!-- [END] End Neume -->
```

16.6. Complex Entities in XML Schema

The following hypothetical Schema definition in pseudo-code is intended mainly to illustrate the basic design idea of mapping the NEUMES context-free grammar to complex entities in XML Schema. The main purpose of doing so is so that an XML parser will effectively do NEUMES data-type checking on a NeumesXML transcription file. Complex entities can be made up of entities that are, themselves, complex entities. Thus kind of recursive entity definition mirrors the recursiveness of the NEUMES CFG. And so, the entire CFG can be translated to Schema format, down to the level of the CFG *terminals*, which are actual Unicode values in the Private Use Area.

```
<schema xmlns="http://www.w3.org/2000/08/XMLSchema" // default namespace
  xmlns:nx="http://scribe.fas.harvard.edu/NEUMES/NeumesXML"
  targetNamespace="http://scribe.fas.harvard.edu/NEUMES/NeumesXML"
  elementFormDefault="unqualified" // is default
  attributeFormDefault="unqualified"> // is default
<!-- Schema for NeumesXML data types -->
<element name="NeumeElement" type="nx:NeumeElementType"/>
<element name="CF" type="nx:CFTYPE"/>
<element name="NeumeChar" type="nx:NeumeCharType"/>
<!-- Etc. -->
<complexType name="NeumeElementType">
```

```

<sequence>
  <nx:element name="neumeChar" type="NeumeCharType"/>
  <nx:element name="neumeCF" type="CF" minOccurs="0"/>
  <!-- Etc. -->
</sequence>
</complexType>
<simpleType name="CFType">
</simpleType>
<simpleType name="NeumeCharType">
  <restriction base="">
    <minInclusive value=""/>
    <maxInclusive value=""/>
  </restriction>
</simpleType>
<!-- Etc. -->
</schema>

```

16.7. UTF-8 Encoding

At the binary level of encoding, NeumesXML files are encoded in *UTF-8* (Unicode Text Format). UTF-8 is the default encoding for XML documents (Unicode coding can be optionally specified). UTF-8 is a strict superset of plain (non-extended) ASCII, and so XML files in normal ASCII are automatically readable. UTF-8 uses a variable-length coding scheme, by which one can mix together 7-bit ASCII characters and multi-byte Unicode characters in the same file. In NeumesXML, we are mixing ASCII characters for chant text and XML tags with Unicode Private Use Area characters for NEUMES data.

In UTF-8, each ASCII character is limited to 7 bits of an 8-bit byte, viz., extended-ASCII characters are not allowed. The *high-order* bit of each byte is always cleared to 0, which is the indicator that this byte is an ASCII character. In the UTF-8 coding of Unicode characters, the high-order bit of each byte is always set to 1, so that their distinction from ASCII characters is guaranteed. Unicode Standard characters are represented in 16 bits (i.e., two bytes), and Unicode Private Use characters are represented in 24 bits (i.e., three bytes). Reading data at an arbitrary position in a UTF-8 file is always unambiguous.

17. References Cited

- [1] CANTUS database project, online at <http://publish.uwo.ca/~cantus/fdes80.html>.
- [2] Digital Scriptorium, “The Digital Scriptorium: A Prototype Image Database & Visual Union Catalog Of Medieval And Renaissance Manuscripts,” online at <http://sunsite.berkeley.edu/scriptorium/>.
- [3] Doneda , Annalisa, “Computer Applications to Byzantine Chant,” *Cantus Planus, Papers Read at the 10th Meeting, Visegrád Hungary 2000*, (forthcoming), online at <http://scribe.fas.harvard.edu/medieval/Doneda.htm>.
[Current URL is <http://www.scribserver.com/medieval/doneda.htm>]
- [4] “TEI: The Text Encoding Initiative,” online at <http://www.tei-c.org/>.
- [5] Unicode Consortium, “Musical Symbols. Range: 1D100-1D1FF. The Unicode Standard 3.1,” online at <http://www.unicode.org/charts/PDF/UID100.pdf>.
- [6] W3C (World Wide Web Consortium), “XML Schema Requirements,” online at <http://www.w3.org/TR/1999/NOTE-xml-schema-req-19990215>.