|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ALTO Extension documentation | | | | | | | |
| **Suggestions for an extension for ALTO format** | | | | | | | |
|  | | | | | | | |
| Document history | | | | | | | |
| Revisions | | | | | | | |
| Version | Status | Author | | Date | | Changes | |
| 0.1 |  | Lukas Gander, Günter Mühlberger | | 10.04.2010 | |  | |
| 0.2 |  | Lukas Gander | | 03.05.2010 | |  | |
|  |  |  | |  | |  | |
| Approvals | | | | | | | |
| This document requires the following approvals: | | | | | | | |
| Version | Date of approval | | Name | | Role in project | | Signature |
| 0.1 | 16.04.2010 | | All Staff meeting Paris | |  | |  |
|  |  | |  | |  | |  |
|  |  | |  | |  | |  |
| Distribution | | | | | | | |
| This document was sent to: | | | | | | | |
| Version | Date of sending | | Name | | Role in project | | |
| 0.1  0.2 | 12.04.2010  03.05.2010 | | All members of ALTO working group  All members of ALTO working group | |  | | |
|  |  | |  | |  | | |
|  |  | |  | |  | | |

Introduction

Within the framework of the EU R&D project IMPACT several partners from the academic and industrial sector are working on improving OCR recognition. In the scope of this work ALTO was selected as one of the main output formats for the results produced by the several tools and OCR engines. The reasons for this selection was that ALTO is widely used among libraries and that IMPACT wants to meet the needs of this community. Nevertheless it turned out that ALTO in its current version does not meet all the requirements set out by the technical partners of the IMPACT project. Therefore a working group was formed to provide some suggestions how the ALTO format could be extended and improved in the light of recent developments. The working group has listed 11 suggestions, most of them are explained with an xml code snippet. Though all the suggestions are based on sorrow investigations they shall be seen as a first contribution to a discussion process.

A final remark has to be made: In order to improve the general readability of the ALTO schema the consequent usage of attribute groups, xml types and the unified naming of attributes is strongly recommended.

Suggestion 1: History tracking

A lot of software tools and also human interactions are involved in different steps of the digitisation process. Each of them may affect an ALTO file by doing some refinements or corrections. From our point of view it would be desirable to keep track of the changes and verification done by the different agents which are involved in the digitisation process. This would allow a simple kind of a document history and gives also important information about the trustworthily of the whole document. If for example everything was verified by a service provider than we can asume that the quality of the document is very high. Storing the old values as well as the new ones would increase the filesize tremendously. Therefore we suggest to store only the information about what has been changed and by whom without keeping track of the changed values. To realize a tracking of changes and verifications within the ALTO files (without tracking old and new values) two changes would be needed:

1. An ID attribute is added to processingStepType in order to allow a referencing by the attributes “CORRECTEDBY” and “VERIFIEDBY” later on.

CORRECTEDBY and VERIFIEDBY attributes have to be added for all elements. The attributes are holding a List of references (using the ID attribute) to all processStepType entries which have changed the original value. See Figure 2 for an example.

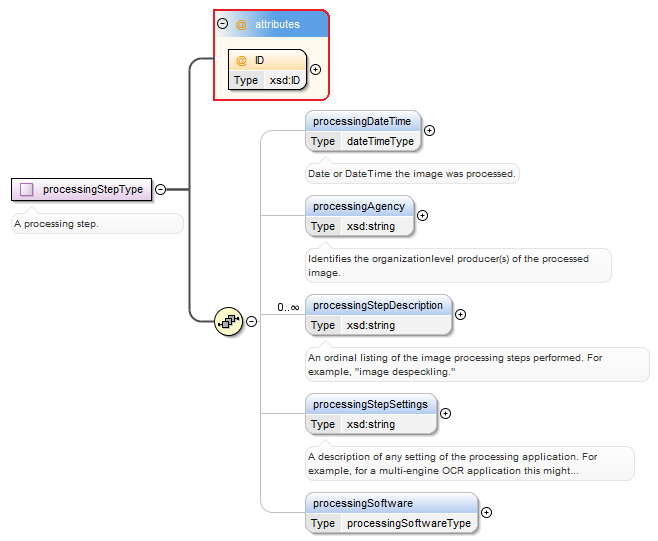


Figure1: Needed changes within the ALTO schema

Impact of suggestion shown by means of an example:



Figure 2: Example shows a TextLine which was corrected by Digitex Service provider

Suggestion 2: Explicit TextStyle encoding

We suggest to use an explicit representation of bold, italic and other style information within the TextStyle element. Currently the style information is represented using a list of constrained strings. ABBYY as well as Omnipage is using an explicit representation for style information. State of the Art used by commercial products should be respected by the ALTO standard.

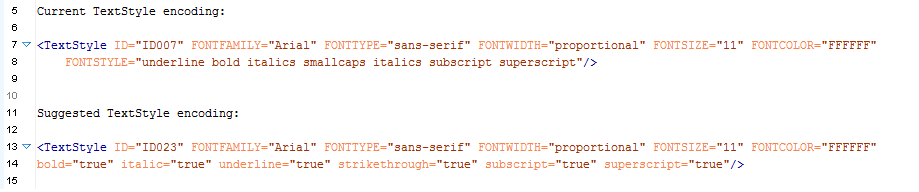
****

Figure 3: Comparison between current and suggested encoding

Suggestion 3: Polynomial representation for segmentation

Currently ALTO allows only a rectangular representation of the segmentation on line and string elements. Also for the print space only a rectangular representation is possible. Polynomial representation is possible only on block level. On Block level the “*Shape” element* type can be used for polynomial representation. Work done in Impact has shown that a rectangular representation is sometimes to weak (overlapping regions). Suggested Solution: Add the “*Shape*” element on all elements including print space. In order to avoid ambiguity also the rectangular representation of the segmentation is transferred to the Shape element.

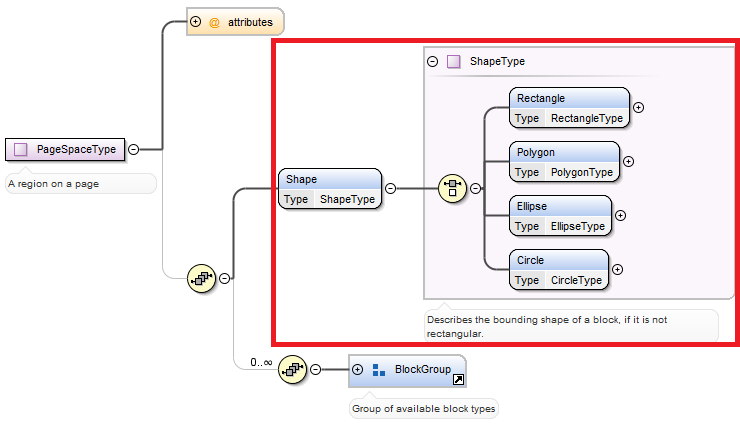


Figure 4: Needed changes for suggestion 3 shown for the Page Space type.



Figure 5: Comparison between current and suggested encoding

Suggestion 4: Parameterization of image transformation.

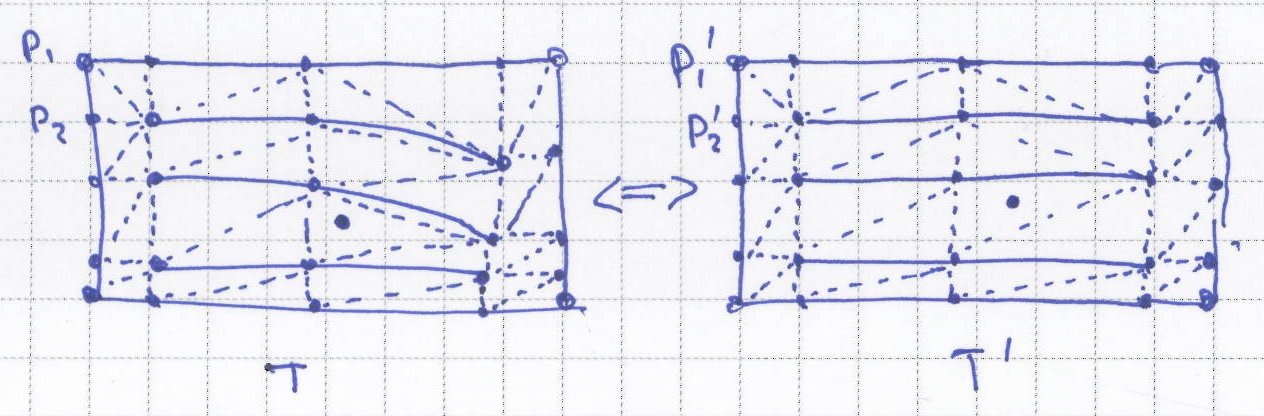
During the OCR processing the input image often is enhanced (e.g. deskewed) by the OCR engine. In such a case there are two images available after the OCR processing: An original image and an enhanced image. Now: in order to avoid to write the coordinates of blocks, lines, strings and characters for the orignal as well as the rendered image, the image transformation should be stored in the ALTO file. The proposed method is flexible and highly efficient.

During image processing there can be a complex modification such as line dewarping. So arises the question of coordinates transformation between original and modified images. Our idea is to define this transformation using a mesh.

This transformation is of course approximate, but for any given desired precision one can define a mesh so that all coordinate transformation will have this precision.

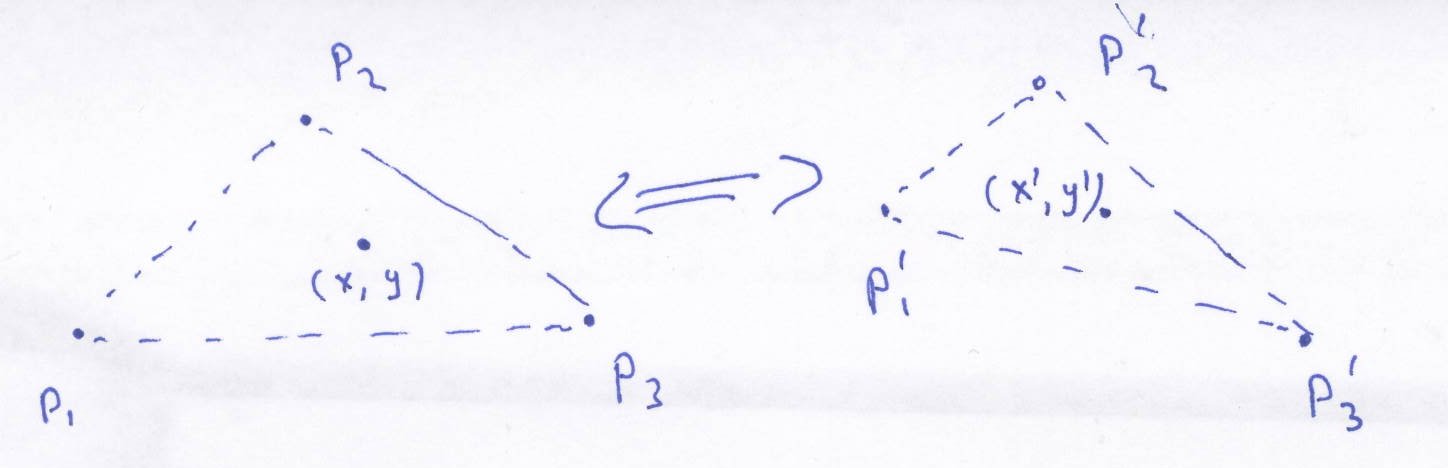
Mesh is defined as a set of n points .. which can have any values, including negative ones. The whole original image should be contained inside of the convex hull of . There is a similar set of points, on a modified image, such that

To build a one-to-one mapping let’s build a triangulation of a set of points on the original image. The triangulation can be built in different ways, we need to choose a deterministic algorithm. Automatically a triangulation is built on a modified image.



Now any point of the original image is contained in a triangle of . Let this triangle be created by points and it corresponds to triangle on a modified image.

Let’s build a mapping for a point inside this triangle.



values are known, so for any given x and y inside p1, p2, p3 triangle we can calculate x’ and y’ and vice versa.

So for any given point (x,y) on original image we can get coordinates (x’, y’) of the same point on transformed image. Also, for any point (x’, y’) on a modified image we can get coordinates (x, y) on original image.

**Example for better understanding:**

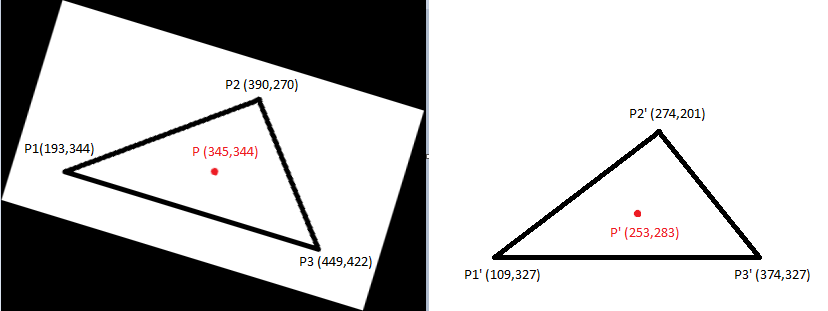


Figure 6: Original triangle on the left side and the modified triangle on the right side

The left triangle shows a calculated triangle from the original Image T (P1,P2,P3) and the right triangle is the corresponding triangle T’ (P1’,P2’,P3’) from the modified image. The Points P1,P2,P3 and P1’,P2’,P3’ are embedded within the Alto file. We want to find now the coordinates of the Point P’(253,283) from the modified image on the original image. We insert now all known variables in the system of linear equations given above. Then we get.

To realize the suggestion the following section is added subsequent to the Style element to the existing ALTO schema:

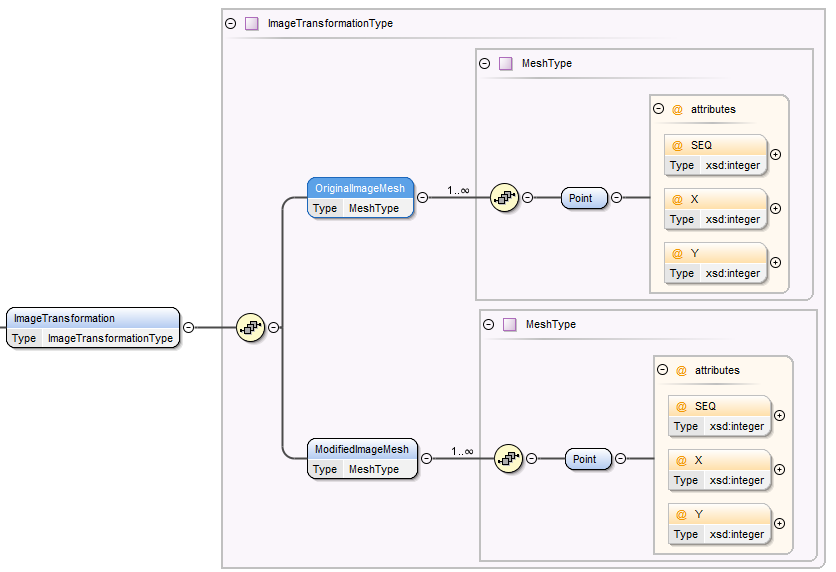


Figure 7: Meshes for original and modified Image

Suggestion 5: Block type expansion

The rationale of the expansion of block types is that different regions are to be recognized or further processed by separate tools. We suggest to add the following block types to the ALTO format:

TableBlock

ChartBlock

GraphicBlock

MathsBlock

NoiseBlock

LineDrawingBlock

UnknownBlock

MusicScoreBlock

ManuscriptBlock

IllegibleBlock

ChemBlock

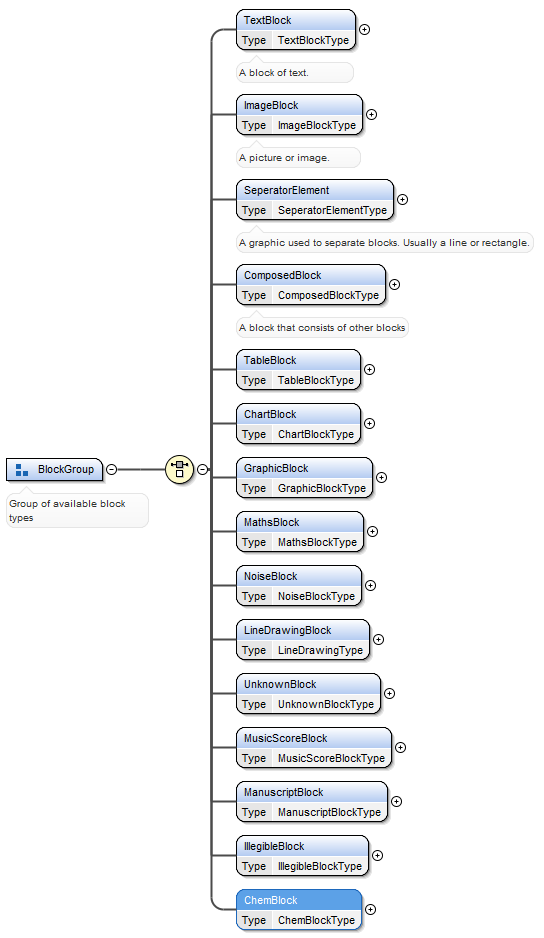


Figure 8: Suggested expansion of block types

Suggestion 6: Segmentation hierarchy extension

Hierarchy of segmentation within ALTO is not detailed enough. Compared to state of the art commercial products a paragraph and character level is missing. The transformation of the output from ABBYY or Omnipage to ALTO should be possibly lossless.

In addition the attributes “Valid” and “KEYIN” could be added to the Glyph element. We are indicating suspicious characters by the attribute VALID=1. The KEYIN=# is relevant to the KEYIN stage. For glyph variants the attributes VS and VC are added.

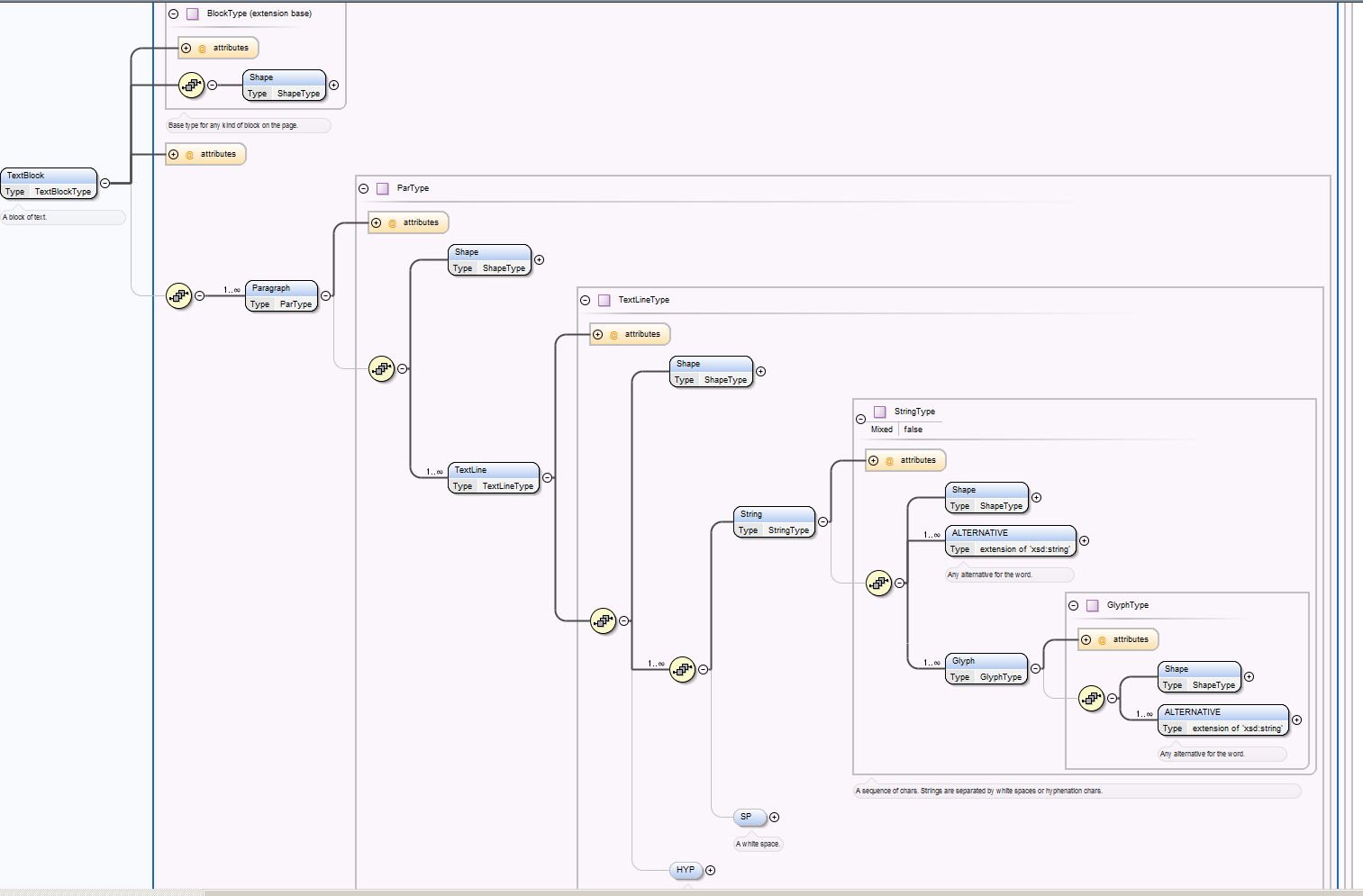


Figure 9: Suggested hierarchy for segmentation for ALTO

Suggestion 7: Logical labelling of structural elements

Not all logical structural elements can easily and meaningfully be stored in METS. From our point of view these are especially structural elements which do not affect the reading order like:

PageHeaders (also called running titles)

Signature marks

Page numbers

Catch words (preview words)

The llabeling should be possible for paragraph (text blocks), line and string elements. The example implementation is only considering typical structural elements from books. We suggest to develop specific schemas for different print types (books, newspapers) in order to force the author to use standardized denotations.

Suggestion 8: Reading Order

Reading order in ALTO is very limited. It only allows simple sequences and not even the starting point is clearly defined. It would be desirable to be able to store more detailed information like potentially nested groups of elements which can be either ordered or unordered.

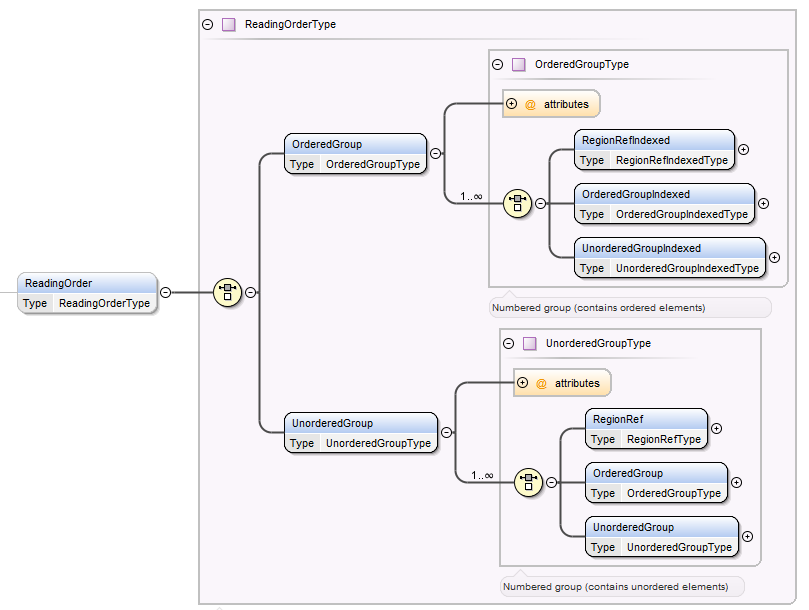
****

Figure 10: Suggested Implementation for reading order

Suggestion 9: Language marking extension

Language should also be available on Line and Word/String level. Currently it is not possible to correctly mark e.g. a Latin citation in an English text. Language could be inherited from parent elements and only be overwritten locally if required (like Cascading Style Sheets…).

Suggestion 10: Special glyph encoding

An indicator for the used encoding of special characters would be very helpful (e.g. ASCII + escape sequences, Unicode – combined symbols, Unicode – base character plus combining characters, individually defined characters from Unicode private use areas etc.). If this information is not available it cannot be guaranteed that different tools interpret the content in the same way.