# WandaXML

A data standard for the annotation and storage of
handwriting samples in the context of (computer-based)
forensic handwriting analysis and
writer identification.

Version 1.0

Mai, 2003

# Contents

**Abstract**

This document is a handbook for the WANDA Annotation language WANDAXML . WANDAXML was designed to address the needs of forensic handwriting data annotation. It allows experts to enter information about writer, material (pen, paper), script and content. Annotations may be organized in a structure that reflects the document layout via a hierarchy of document regions. Image processing operations and feature extraction operations can also be recorded in the defined format.

For more information about the WANDA project please visit: *http://pentel.ipk.fhg.de.*

# 1 Introduction

This document is about a conceptional framework and data model for processing, analyzing and storing of handwriting samples in the context of (computer-based) forensic handwriting examination and writer identification. The objective of framework's application is

- to *Ensue objectivity and reproducibility* of processing steps and examination results,
- to *Promote research and development* of sophisticated handwriting analysis methods, and
- to *Establish a common ground for international exchange* of handwritten data samples.

The addressed readership of this document comprise:

- *Forensic handwriting experts* who are willing to support interoperability between data and system as well as to follow standardizations and quality assurance of investigations reports,
- *Researchers, developers and industrial entities* who are going to design and implement (computer-based) algorithms and systems for forensic handwriting examination.

## 1.1 Background

In order to consider an investigation report as scientifically proved, the official expert report has to fulfill certain requirements (e.g. BGH [1]). Therefore, experts from the field of forensic handwriting examination put a lot of efforts in the harmonization and objectification of verbal categories on final conclusions of a handwriting's authorship (e.g. Köller et. al [6]). These activities can be further promoted if not only the final given evidence but also the terminology for describing the investigation object and the supporting arguments during the validation on an hypotheses became standardized. Moreover, the explicit specification of an application domain is essential for the development and the use of intelligent computer-based systems: The implications include not only providing system developers with the vocabulary to represent domain knowledge, but even more important to record inter-operation of humans (handwriting experts) and machines in daily forensic casework. For example the current FISH system [4, 10, 9, 7] considers not only mathematically-based handwriting features, but also very conventional handwriting characteristics as writing style or connectivity (e.g. Hecker 1993 [5]).

## 1.2 Purpose

The conceptional framework and its implemented data specification are intended for dealing with the varieties of handwriting characteristics, their verbal descriptions as well as their computer-based analysis in forensic context.

The framework's underlying domain and data model is the result of joint efforts of forensic handwriting experts as well as researchers and developers in the field of computer-based handwriting processing and analysis. Well-established termini and procedures of forensic science inspired many aspects of the present framework. To fulfill the challenging demands of computer-based handwriting analysis and recherch, further extensions had to be completed.

WANDAXML - the data specification, which will be also presented here, incorporates expertise on the level of

- Handwriting and document related forensic knowledge,

– Digital handwriting images and image processing,

– Digitized handwriting movements and their processing as time series.

Widely spread standards for the management of (handwriting) data are considered. Particularly, to ensure long-term operation the framework invokes the e*X*tensible *M*arkup *L*anguage (XML) [3] that follows world-wide standardized syntax rules, recommended by the *W*orld *W*ide *W*eb *C*onsortium (W3C) [12].

This document is not intended to teach and to promote XML itself. Just a brief introduction will be given for the interested reader and to bring out the potential of the presented approach. Also, this report is not intended to fundamentally question the levels and categories for describing handwriting data. However, on the level of forensic handwriting and document related knowledge most implementations base on the current forensic practice in Germany, primary according to systems operating in forensic labs [2, 4, 11], forensic literature used for the training of experts [8, 5] as well as according the expertise of (police) experts that were interviewed during the specification phase. Further extensions might be required in order to meet demands and forensic standards in other countries in Europe, America and Asia.

Please note that the current version is further evolving. Remarks, recommendation and demands by readers are highly appreciated and might be considered in further developments of the WandaXML specification and in next versions of this document.

### 1.3 Outline

This document is divided into two parts. Part1, section 1 through section 6 deals with the derivation and explanation of the conceptional framework, while Part2 section A through section I, covers the reference description, definitions and samples for the proposed levels and categories.

The goal of the Part 1 is to explain the general idea, to review existing data standards, to motivate the proposed approach and to present applications of the framework. For reading this part of the document it is not required to have knowledge on XML.

Part 1 consist of following sections:

**Section 1** The rest of this section briefly introduces XML, so the reader gets familiar with the syntax applied later.

**Section 2** Recalls existing data standards for the storage and description of (handwriting) data and concludes on the requirements for an handwriting data specification in forensic context.

**Section 3** Presents the WandaXML specification. Starting with some design considerations the generic structure will be derived. Furthermore there are categories for the description of handwriting characteristics, employed materials and content related information presented. Also, categories for the computer-based processing will be derived and explained in more detail.

**Section 4** Shows the wide variety of applications for WandaXML . Besides the usage in current setups of daily forensic casework, opportunities for the improvement of computer-based handwriting feature classification will be outlined. Moreover, it will be shown how upcoming research and development activities might profit from the usage of the here presented framework.

**Section 5** Provides recommendations to further extensions, supplies practical considerations as well as introduces useful tools for further developments and extensions of WandaXML .

**Section 6** Summarizes the currents achievements and provides anchors for interested readers who want to further contribute to WandaXML the conceptional framework for storing and describing handwriting samples in the context of (computer-based) forensic handwriting and writer identification

Part2, consisting of section A through section I, is organized around the implementation of the framework. It contains the complete reference lists, type definitions and examples. Therefore, Part2 is of great interest for those who are planning to implement the current version of the WandaXML specification and/or who are intend to extend it. More detailed, Part2 comprise the following sections:

**Section A** Wandoc being the generic wrapper for all upcoming descriptions and formats of handwriting data and their processing.

**Section B** Writer covering those particularities that might be interesting in the determination of the authorship of a handwriting.

**Section C** Material supplying the qualitative and quantitative annotations on the employed writing material as pen, ink, paper and supposed writing pad.

**Section D** Script supporting the verbal descriptions of handwriting characteristics as currently used in forensic practice.

**Section E** Content enabling the verbal description of the handwriting document itself as well as the supposed message and information needed for later linguistic analysis.

**Section F** Scan documenting the digitalization of a paper-based handwriting document.

**Section G** Proper documenting the computer-assisted preprocessing particularly removal of document backgrounds and imprints.

**Section H** Nicifeat being a format for the storage of interactively measured handwriting characteristics, which primarily follow well established features implemented in the Fish system [4, 10, 9, 7, 11].

**Section I** Wink a format to document recorded handwriting movements. This format is strongly inspired by the inkXML data format that is also a recommendation by the W3C.

### 1.4 What is XML?

Some understanding of XML may help reading this document. We provide in this section enough elements about XML that it should be possible to follow the logic of the design of WandaXML . For a more comprehensive tutorial, see for instance http://www.w3schools.com/xml/default.asp.

Extensible Markup Language (XML) is a text format derived from SGML markup language used for many years in the publishing industry. It is a W3C recommendation (see http://www.w3.org/XML/). It was designed to meet the challenges of large-scale electronic publishing. It is now playing an increasingly important role in the exchange of a wide variety of data.

XML looks very similar to HTML that is also derived from SGML. It is composed of a number of"tags" or "elements" enclosed in brackets that delimit a portion of text. For example, the following XML excerpt could define a title:

```
<title>
    The Extensible Markup Language
</title>
```

In this example, `<title>` is the opening tag and `</title>` is the end tag. Unlike in HTML, in XML the tags are case sensitive, i.e. `<title>` is different from `<Title>` and `<TITLE>`. Another difference is that XML does not tolerate opening tags that are not followed by end tags. In HTML, for instance, the tag `<br>` indicating a line break is not followed by `</br>`. XML possesses a shorthand for empty tags that does not exist in HTML to avoid having to write something like `<br></br>`, the notation:

```
<br/>
```

Like HTML, XML is hierarchical: tags may contain other tags. The following HTML example is also valid XML:

```
<html>
    <head>
        <title>
            The Extensible Markup Language
        </title>
        <!-- ... other head elements ... -->
    </head>
    <body>
        ... document body...
    </body>
</html>
```

Like in HTML, comments start with `<!--` and end with `-->`. Multiple spaces and line breaks are ignored.

XML also supports tag attributes. For example, titles may have two attributes: language and reading direction:

```
    <title lang="he" dir="rtl">...a Hebrew title...</title>
```

XML allows users to define their own markup language respecting the basic XML syntax and thus benefit from a large body of existing software, which manipulates XML (see Section 5.4). In particular, XML and Java are very complementary: XML is a widely accepted means of creating portable data, and the Java programming language provides portable code abilities.

A particular XML markup language such as WandaXML can be defined in a standard way using the document type definition language (DTD) (see http://www.w3.org/TR/REC-xml) or XML schemas (http://www.w3.org/XML/Schema). In the following sections, we define WandaXML using DTDs.

Here is an example of an XML document with a document type declaration:

```
<?xml version="1.0"?>
<!DOCTYPE mydoc SYSTEM "mydoc.dtd">
<mydoc>
    <title lang="he" dir="rtl">...a Hebrew title...</title>
    <body>... document body...</body>
</mydoc>
```

The system identifier "mydoc.dtd" gives the address (a URI – Uniform Resource Identifier) of a DTD for the document.

Here is an example of DTD for that document:

```
<!-- Language names follow the ISO 639-1 recommendation -->
<!ENTITY % LanguageCode "*|aa|ab|af|en|he">
<!-- Exactly one title is required per document. -->
<!ELEMENT mydoc (title+, body?)>
<!ELEMENT title (#PCDATA)>
<!ATTLIST title
    lang     (%LanguageCode;) #IMPLIED
    dir      (ltr|rtl)        #IMPLIED
>
<!ELEMENT body (#PCDATA)>
```

The DTD specifies three types: entities, elements, and attribute lists. Entities allow to conveniently define lists of valid attribute values. Elements correspond to XML tags. Attribute lists define lists of attributes. For a DTD tutorial, see for instance: http://www.w3schools.com/dtd/default.asp.

### 1.5 WandaXML requirements

The XML language developed for Wanda must consist of data structures flexible enough to implement state-of-the art handwriting and document analysis methods. These data structures must also be specific enough to fulfill the requirements of the BKA concerning, in particular, backward compatibility with the existing FISH system.

As part of the design process of the Wanda framework, we identified some requirements for WandaXML .

WandaXML needs to support a variety of data, but more particularly raster images of handwritten documents and virtual ink data. Virtual ink data is used in Forensic applications in the analysis process. For instance, an expert may use an touch sensitive tablet to retrace a handwritten document and obtain a sequence of pen coordinates. A standard way of overlaying virtual ink on top of raster images is highly desirable.

WandaXML needs to support a variety of document structures: forms and other documents mixing handwriting with machine printed text and other elements (stamps, pictures). A notion of "region" in raster images needs to be introduced. It should be possible to describe logical relations between regions. Further down the road, it may also be useful to introduce a corresponding notion of "segment" in virtual ink data and define logical relationshipd between segments and between segments and regions.

WandaXML should provide the possibility of entering detailed annotations about documents. The annotation scheme should be extensible. We have identified five types of annotations that are required for Forensic applications:

- Device: Information about the device that recorded the data (scanner or digitizing tablet).
- Writer: Information about the writer who wrote the document.
- Script: Information about the type of script used.
- Material: Information about the paper and pen used.

– Content: Information about the document content, including the verbatim transcription.

In addition, it should be possible to enter meta data recording information about the forensic expert.

WandaXML should provide users with the possibility of recording operations performed on the data, for instance, image filtering or feature extraction (including "measurements" performed on the handwriting). Optionally, WandaXML could provide users with means of recording the interaction of the user with the computer and play back the sequence of events.

WandaXML should provide users with the possibility of decomposing the annotations into separate files and include a file into another one using a link.

Other desirable features for WandaXML that could be included further down the road include the notion of "layers" in the annotations allowing to visualize subsets of the annotations and the notion of "styles" allowing to customize the rendering.

# 2 Review of existing standards

Before undertaking the task of defining a new XML language for Forensic applications, we reviewed existing standards that are in use in the handwriting recognition and document analysis community. We briefly summarize here the main features of the standards that inspired WandaXML and outline their advantages and limitations to motivate our approach.

We distinguish between formats to *encode* data and formats to *annotate* data. WandaXML is essentially a data annotation format. It is compatible with a variety of data encoding formats. For image data, typical examples would be JPEG, GIF, and TIFF. For vectorial data (graphics represented by their coordinate points, not by pixel values) we can cite SVG.

Several handwritten document annotation formats predate WandaXML . Image annotation formats include IAM, Xmillum and Trueviz. Formats for virtual ink or "electronic ink" combining both data *encoding* and data *annotation*, include Unipen and InkXML. [1]

We do not review below image *encoding* formats because WandaXML simply supports existing formats. We review image *annotation* formats from which WandaXML borrows. We review virtual ink formats because a subset of WandaXML (called wink) builds on top of such existing standards.

## 2.1 Vector data formats

In this section we review formats encoding data that was recorded as a sequence of pen coordinates. We consider two virtual ink formats (InkXML and Unipen) and one vectorial data encoding format (SVG). Owing to the limitations of these formats and the large number of features not directly relevant to our application domain, WandaXML borrows from all three formats but adopted neither in whole.

### 2.1.1 InkXML

InkXML is a format under development that was recently proposed to the W3C consortium by IBM, Intel, Motorola, and the International Unipen Foundation (see http://www.w3.org/TR/inkreqs/). InkXML is a markup language for the exchange of virtual ink, conveying such information as the kind of pen, the color of the ink and the nature of the medium, the pressure applied to the pen, its position and speed. InkXML can be used to exchange virtual ink among devices, such as handhelds,

[1] This outlines the difficulty of separation data encoding and annotations in the case of "electronic ink".

laptops, desktops, and servers. InkXML is intended to provide the ink component of Web-based multimodal applications.

The wink format that is part of WandaXML draws heavily on InkXML. Both in InkXML and in wink, the `<trace>` tag is used to record the data captured by a digitizer. It contains a sequence of points encoded according to the specification given by the `<traceFormat>` tag. A lot of complexity can be supported via `<traceFormat>` in InkXML. In wink, a single trace format per vector image is sufficient, and the unnecessary optional channels and wildcards are supressed. A trace is a complete pen-down movement bounded by two pen-up movements or a complete pen-up movement. InkXML pen up/pen down events are supported in wink, but pen color and brush size changes are not encoded as part of a trace. The InkXML `<chunk>` tags, which group traces but do not carry semantic meaning, are not supported in wink.

In this example, InkXML first gives an absolute position followed by deltas, which are relative positions referenced to the first absolute position. The third coordinate encodes pressure.

```
<traceFormat sampRate="100pps">
     <channel name="X" type="decimal" resolution="10dpmm" />
     <channel name="Y" type="decimal" resolution="12dpmm" />
     <channel name="P" type="decimal" resolution="12ppg" />
</traceFormat>
<trace type="penDown" color="0 0 255" brushSize="3">
    234  122  12
    2  12  14
    -3  0  15
</trace>
```

In wink absolute coordinates are recorded. There are additional provisions in wink that are not part of InkXML like numbering traces by adding an id attribute to `<trace>`.

The InkXML tag `<screenContext>` describes the input conditions when digital ink data is written and allows applications to reconstruct the input area under which the digital ink was captured. It can be used to superimpose virtual ink on top of an image. In wink we generalized the notion of screen context and introduced a trace context to encode easily different image and tablet scales. We introduced in wink a bounding box for the active tablet area visible by the user during tracing.

The InkXML standard does not specify where the origin is. The wink format uses the upper left corner.

### 2.1.2 Unipen

Several members of the Wanda team were involved in the definition of the Unipen format of on-line handwriting data exchange (see http://unipen.nici.kun.nl/). On-line handwriting refers to handwriting recorded on a digitizing tablet or pen computer that consists of a time ordered sequence of pen coordinates also referred above as virtual ink or electronic ink. The Unipen data format has served between over 40 institutions to exchange large bodies of handwritten data (the equivalent of five million characters). It is an ASCII format that in many ways transpires the same philosophy as XML: On-line handwritten documents are organized in pages, blocks, lines, and segments. Such notions have been generalized in WandaXML by the “regions” (also referred to as Regions of Interest, or ROI).

The equivalent of a trace in InkXML is called a component in Unipen. Unipen distinguishes between pen-up and pen-down components, separated by time intervals ”dt”. The coordinates

of the points are provided, not the deltas. The equivalent of `<traceFormat>` is provided by the "coord" keyword. There are specific keywords for the various units. There are no provisions for color, brush shape and brush size. We translate below the InkXML example into Unipen:

```
.COORD X Y P
.X_POINTS_PER_MM 10
.Y_POINTS_PER_MM 12
.POINTS_PER_GRAM 12
.POINTS_PER_SECOND 100
.PEN_DOWN
234  122  12
236  134  26
233 134  41
```

Note that in Unipen the origin of the coordinate system is in the lower left corner.

Unlike InkXML, Unipen was designed for handwriting research, so it has data annotation tags, including verbatim text transcription, device, writer, institution or origin information.

We initially envisioned that WANDA application programs would support the native Unipen format to encode virtual ink. While we still think that is will be important in the future to add a filter that allows importing Unipen data since a large body of data is encoded in that format, we have noted some of the limitations of Unipen that makes it unfit for WANDA :

- Unipen is not XML-based and its conversion of XML is awkward.
- Unipen does not provide detailed annotation capabilities (for writers, devices, ink, etc.) so it would need to be greatly expanded.
- Unipen does not provide with a detailed mechanism to overlay "electronic ink" on top of a "screen context".

Therefore, we built on top of our Unipen experience to define WANDAXML , but also borrowed ideas from other standards like InkXML to which the Unipen foundation has been participating.

### 2.1.3 SVG

Virtual ink can be considered a general framework for any vector image data. Still, some basic shapes such as rectangles, polygonal lines, and polygons, are more simply described with specific tags suitable to encode mouse events. We naturally thought of supporting a vector image standard such as SVG (see http://www.w3.org/TR/SVG/). However, SVG by itself would be insufficient to encode virtual ink since, for instance, it has no provisions for encoding pressure.

In SVG, the equivalent of an InkXML trace or a Unipen component would be a polyline. We translate again the same example:

```
<polyline fill="none" stroke="blue" stroke-width="3"
            points="234,122 236,134 233,134" />
```

In WANDAXML , we defined several tags to encode mouse events that are inspired by SVG that complement the virtual ink set of tags inspired by InkXML. However, the intersection of the needs of WANDA and the features of SVG was too small to justify supporting the full SVG standard.

## 2.2 Image data formats

### 2.2.1 IAM

The IAM data format was designed by M. Zimmerman and collaborators at the Institute of Computer Science and Applied Mathematics of the University of Bern, Switzerland. It was designed to encode the meta data of the IAM database for off-line handwriting recognition research (http://www.iam.unibe.ch/ zimmerma/iamdb/iamdb.html). The database contains 1500 pages of scanned text by 500 writers. The XML provides information about:

- form and writer IDs,
- form skew and line slant and skew,
- line bounding box,
- line verbatim transcription,
- word bounding box,
- word verbatim transcription and grammatical tag,
- connected components of handwritten words.

The segmentation was performed automatically and the bounding boxes are provided to verify the accuracy of the segmentation visually. We show below an example of IAM XML. Without getting into the details, the tag `<frm>` is the root element ("form"). A form can contain machine printed text encoded by the tag `<mp>` or handwritten text encode by `<hw>`. Such block can contain lines (`<lin>`), which in turn can contain words (`<wrd>`), themselves composed of components (`<cmp>`).

```
<?xml version="1.0" encoding="UTF-8"?>
<frm crd="020114" h="3542" id="a01-000u"
     mod="020717" skw="287" sta="final"
     ver="2.0" w="2479" wid="000" >
    <mp>
        <lin id="a01-000u-mp00"
            txt="A MOVE to stop Mr. Gaitskell from nominating ...">
        <!-- ... -->
        <lin/>
    </mp>
    <hw>
        <lin ass="-187" asx="0" asy="739" c="19" chw="999"
             dss="-187" dsx="0" dsy="831"
             fd0="12129" fd1="2994" fd2="-339" ftw="5" h="91"
             id="a01-000u-00" lbs="-108" lbx="0" lby="810"
             seg="ok" slt="90000" stw="7333" ths="154"
             txt="A MOVE to stop Mr. Gaitskell from"
             ubs="-347" ubx="0" uby="769" w="1663" x="408" y="746" >
            <wrd id="a01-000u-00-00" s="y" tag="AT" txt="A" c="1"
                 x="408" y="768" w="27" h="51" >
                <cmp id="00" x="408" y="768" w="27" h="51" />
```

```
            </wrd>
            <!-- ... -->
            </wrd>
            </wrd>
            </wrd>
        </lin>
        <!-- ... -->
        <lin/>
        <lin/>
        <lin/>
    </hw>
</frm>
```

As it can be seen, the IAM format provides a wealth of details on the segmentation parameters, but provides minimal annotations besides the verbatim transcription and the grammatical tags. It is a good *ad hoc* format to encode the meta data of the IAM database, but it does not provide a framework that would allow us to extend it for the needs of the WANDA project. In WANDAXML , we retained the idea of a hierarchy similar to the form-line-word-component hierarchy by introducing the concept of nested regions. However, we did not limit ourselves to a fixed pre-determined hierarchy because we anticipated that we would have to analyze documents having a variety of structures, in which other semantic blocks might make more sense (e.g. "address-block", "signature", "character"). In WANDAXML , we put a lot more emphasis on human-generated annotations (on writer, script, material and content) and allowed recording a wider variety of machine-generated annotations (not solely segmentation annotations) by introducing the notion of "filters".

### 2.2.2 XMillum

Xmillum was developed by the Department of Informatics at the University of Fribourg, Switzerland (see http://xmillum.sourceforge.net/). It is a framework for visualization of document recognition data and creation of interactive document recognition applications. It is not a document annotation format by itself. Rather, it works in conjunction with any other document recognition data available in XML format. The modalities of Xmillum data visualization and interaction of a given format are specified using XSLT stylesheets (for a tutorial, see e.g. http://www.w3schools.com/xsl/). The Xmillum Java software can be extended using Java plugins.

To illustrate how Xmillum works, we use an XML example from the University of Washington database that annotates a machine printed document. The tag root element `<uwdoc/>` starts the document, which is composed of zones. The element `<zone/>` may contain the verbatim transcription of "ground truth" `<gt/>`.

```
<?xml version="1.0" encoding="UTF-8"?>
<uwdoc image="cvpr-19920615-154.png">
    <zone fontsize="4-8" fontspacing="proportional"
        fontstyle="plain" fonttype="serif"
        height="64" id="00H"
        width="552" x="236" y="3120">
        <gt xml:space="preserve">
            0-8186-2855-3/92 $3.00 \copyright 1992 IEEE
        </gt>
```

```
    </zone>
    <zone fontsize="9-12" fontspacing="proportional"
        fontstyle="plain" fonttype="serif"
        height="268" id="00G"
        width="1000" x="1324" y="2752">
        <gt xml:space="preserve">
            Due to the geometric reasoning capability,
            the APO system understands the operations ...
        </gt>
    </zone>
    <zone/>
    <!-- ... -->
    <zone/>
</uwdoc>
```

What Xmillum does is to define a stylesheet for that document that translates its XML. We show below an example that would display two layers, one for the image and one for the zones, and would show zones as yellow rectangles:

```
<?xml version="1.0"?>
<!-- Define the name spaces for style sheets
and Xmillum and the corresponding prefixes xsl and xmi -->
<xsl:stylesheet version="1.0"
  xmlns:xmi="http://www-iiuf.unifr.ch/~hitz/xmillum"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <!-- Start a template that matches to the root of the document -->
  <xsl:template match="uwdoc">
    <xmi:document>
      <!-- Style for solid, yellow, transparent blocks -->
      <xmi:style name="yellow-style">
        <param name="foreground" value="yellow"/>
        <param name="transparency" value="0.4"/>
        <param name="fill" value="true"/>
      </xmi:style>
      <!-- Rectangular yellow blocks -->
      <xmi:object name="yellow-block" class="iiuf.xmillum.displayable.Block">
        <param name="style" value="yellow-style"/>
      </xmi:object>
      <!-- Images -->
      <xmi:object name="image" class="iiuf.xmillum.displayable.Image"/>
      <!-- First layer: the image -->
      <xmi:layer name="Image">
        <image src="{@image}"/>
      </xmi:layer>
      <!-- Second layer: the different zones -->
      <xmi:layer name="Zones">
        <xsl:for-each select="zone">
            <yellow-block x="{@x}" y="{@y}" w="{@width}" h="{@height}"/>
```

```
        </xsl:for-each>
      </xmi:layer>
    </xmi:document>
  </xsl:template>
</xsl:stylesheet>
```

Therefore, Xmillum is complementary to formats such as IAM and the University of Washington format that are strictly data formats. It provides styling and plugin capabilities to enable data visualization and user interactions. However it is biased towards machine printed document analysis and does not provide with the necessary annotation XML that WANDA needs. It could be used in conjunction or as a complement to WANDAXML (see Section 3.3).

### 2.2.3 TrueViz

The TrueViz format was designed to support the TrueViz Java platform developed by Tapas Kanungo and collaborators (see http://citeseer.nj.nec.com/384635.html and http://www.cfar.umd.edu/~kanungo/software/trueviz-$1_0$2.tar.gz). TrueViz is a tool for visualizing and editing ground truth and meta data for OCR written in Java.

We evaluated TrueViz carefully and retained large parts of its design in WANDAXML . In fact, some of the Java code of TrueViz was partially re-used in the WANDA platform.

In TrueViz, a given image is associated with a corresponding XML file providing annotations. A typical TrueViz XML file looks like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Page SYSTEM "Trueviz.dtd">
<Page>
    <PageID Value="my_page1"> </PageID>
    <PageType Value="Letter"> </PageType>
    <!-- more page annotations here -->
    <GT_Text Value="This is the text of the document..."> </GT_Text>
    <Line>
        <LineID Value="my_line1"> </LineID>
        <LineCorners>
            <Vertex x="15" y="45"> </Vertex>
            <Vertex x="1670" y="45"> </Vertex>
            <Vertex x="1670" y="167"> </Vertex>
            <Vertex x="15" y="167"> </Vertex>
        </LineCorners>
        <LineNext Value="my_line2"> </LineNext>
        <LineNumChars Value="13"> </LineNumChars>
        <GT_Text Value="This is the text"> </GT_Text>
        <Word>
            <WordID Value="my_word1"> </WordID>>
            <WordCorners>
                <!-- vertices -->
            </WordCorners>
            <!-- some word annotations here -->
            <Character>
```

```
                <CharacterID Value="my_char1"> </CharacterID>
                <CharacterCorners>
                    <!-- vertices -->
                </CharacterCorners>
                <!-- some character annotations here -->
            </Character>
            <Character/>
            <!-- more characters -->
            <Character/>
        </Word>
        <Word/>
        <!-- more words -->
        <Word/>
    </Line>
    <Line/>
    <!-- more lines -->
    <line/>
</Page>
```

TrueViz has been inspirational for WandaXML . It provides the idea of regions that Wanda needs. In TrueViz, regions may be specialized (e.g. pages, lines, words and characters), or generic (zones). The regions are nested using the XML hierarchy, but TrueViz also allows the user to define a logical order between regions that are at that are at same level of the hierarchy, using the elements `<PageNext>`, `<LineNext>`, etc. We retained the same concept in WandaXML while simplifying the XML syntax. In particular, there is an unnecessary abundance of specific XML tags for the different type of regions like `<LineCorners>`, `<WordCorners>`, `<CharacterCorners>`. In WandaXML , there is only one region tag, but regions may be of different types. Several document analysis annotations are defined in TrueViz, but do not suffice for the purpose of Wanda .

### 2.3 Comparison

We summarize the features of the existing standards in Table 1. We outline in bold the features that are required for Wanda . Other features that may be of interest are shown in regular type. Even without making a statement of quality, it is apparent that no standard predating WandaXML meets all of our requirements.

## 3 Conceptional framework for forensic handwriting annotation

I should be apparent from the previous section, which reviews existing standards, that there is a need for a more general framework that encompasses the notions of on-line and off-line handwritten documents and their possible dependencies, includes detailed annotations capabilities, and provides users with with the possibility of recording data processing operations.

### 3.1 Design considerations and adopted philosophy

We adopted a modular design for WandaXML . Presently, the WandaXML language is decomposed into the following subsets:

| | InkXML | Unipen | SVG | IAM | XMillum | TrueViz | WANDA |
|---|---|---|---|---|---|---|---|
| **XML-based** | √ | | √ | √ | √ | √ | √ |
| **raster images** | | | √ | √ | √ | √ | √ |
| **vector images** | √ | √ | √ | | | | √ |
| **virtual ink** | √ | √ | | | | | √ |
| v-ink segments | √ | √ | | | | | |
| **image regions** | | | | √ | √ | √ | √ |
| **ink/image overlay** | √ | | | | | | √ |
| **device annot.** | √ | √ | | | | | √ |
| **writer annot.** | | √ | | √ | | | √ |
| **script annot.** | | √ | | | | | √ |
| **material annot.** | | | | | | | √ |
| **content annot.** | | √ | | √ | | | √ |
| **filters/plugins** | | | √ | | √ | | √ |
| interactivity | | | √ | | √ | | |
| layers | | | | | √ | √ | |
| styles | | | √ | | √ | | |
| **external link** | | √ | √ | | | | √ |

Table 1: **Existing standard comparison.**

- `wandoc`: subset describing regions of interest and filters (operations applied to the documents), and annotation encapsulation;
- `writer`: subset describing writer annotations;
- `material`: subset describing "material" (e.g. paper, pen) annotations;
- `script`: subset describing handwriting script;
- `content`: subset describing document content annotations;
- `scan`: subset encoding scanner operations;
- `proper`: subset encoding image preprocessing;
- `nicifeat`: subset encoding measurements;
- `wink`: subset encoding virtual ink.

Planned additions of other subsets will be described in Section 3.3.

All subsets are independent, except for `wandoc`, which acts as a "wrapper".

The WANDAXML design revolves around a small number of key concepts: annotations, filters, and regions.

- **Annotations:** Within `wandoc` we have defined a generic tag `<annotation/>`, which wraps around annotations of different types. Presently, we have defined four types of annotations: `writer`, `material`, `script`, and `content`. We make use of XML name spaces (as defined by the W3C recommendation http://www.w3.org/TR/REC-xml-names/) to nest the annotation subsets in `wandoc`. This framework allows us to make detailed annotations of forensic documents in the categories already defined (writer, content, material, and script) and leaves the door open for further annotation extensions that would be implemented as separate subsets.

– **Filters:** Another feature of the WandaXML language is that it provides users with a flexible way of recording and eventually playing back operations on the data. This is achieved through the notion of "filter". A filter can be understood as a computer program that processes the document and returns either a new transformed image or a set of features. The `wandoc` language subset possesses a tag `<filter/>` which wraps around any type of user defined filter. Examples of such filters are defined by the `proper` and `nicifeat` language subsets (see appendix). Again, name spaces are used to allow designers of application specific subsets of WandaXML to extend the language and specify new types of filter inputs and outputs.

– **Regions:** Importantly, WandaXML allows the forensic expert to describe the document structure via a hierarchy of regions. The `wandoc` language subset possesses a tag `<region/>` that defines regions of rectangular or polygonal shapes. Annotations and filters can apply to the whole document of to regions. Examples of regions include paragraphs, address blocks, lines, words, and characters. The region types can be defined by the users.

Finally, WandaXML allows superimposing virtual ink (electronic ink) as entered, for instance, using a digitizing tablet, on top of a document image. This is achieved with the `wink` subset.

## 3.2 Implementation of the defined framework

In the remainder of the Section, we describe in more details the `wandoc` subset, which conveys all document annotations. We also elaborate on the specific annotations that have been defined for forensic applications: `writer`, `content`, `material`, and `script`. We briefly describe the `wink` subset, which allows us to encode virtual ink. For the other language subsets, we refer the user to the reference manual, which is appended.

### 3.2.1 Wandoc skeleton

We adopted the following skeleton for `wandoc`:

```
<wandoc>
    <filters/> [?]
    <annotations/> [?]
    <wanda_link/> [*]
    <meta/> [?]
    <pages> [?]
        <page> [+]
            <filters/> [?]
            <annotations/> [?]
            <regions/> [?]
            <wanda_link/> [*]
            <meta/> [?]
        </page>
    </pages>
</wandoc>
```

We use the following convention to describe tag requirements: [+] at least one; [?] zero or one; [*] any number including zero.

In the `wandoc` framework, a document consists of one or several pages, each of which may be represented by an image. Notice that there is no tag `<image/>`. Images are imported through specially defined filters, e.g. the `scan` filter.

Filters may be applied at various levels of the hierarchy: at the top level to act on the collection of pages, at the page level, or, as we shall see below, at the region level. Similarly, annotations may be attached to all of these different levels.

We introduced a `<meta/>` tag that regroups information about how the particular annotation was generated (author, date created, contact email, author's affiliation, etc.) and pertains to all WandaXML subsets.

We also introduced a generic `<wanda_link/>` tag: it refers to a file locator and plays a role similar to an HTML hyperlink or an Xlink.[2]

### 3.2.2 Wanda regions

Wanda regions provide the possibility of multiple levels of annotations and local filtering. Regions may either be manually extracted or be the result of an automatic document segmentation. An example of manual extraction of regions is shown in Figure 1.

We summarize in Table 2 the essential region properties. Regions possess a unique identifier. This provides users with the flexibility of referring to regions from other parts of the document and, for instance, applying a filter to multiple regions. The label is a user defined string that may refer to the content of the region and allows the user to retrieve that region easily by performing a search. The attribute next allows the user to link the region to another region and therefore provide a logical order of the regions (e.g. a reading order).

We adopted the following skeleton for regions:

```
<region> [+]
    <points> [+]
        <point/>
    </points>
    <filters> [?]
        <filter/>
    </filters>
    <annotations> [?]
        <annotation/>
    </annotations>
    <regions> [?]
        <region/>
    </regions>
    <wanda_link/> [*]
</region>
```

The tags `<filter/>` and `<annotation/>` are wrapper tags. Via the use of name spaces, they allow users to describe new kinds of filters and annotations without modifying the core WandaXML specification, as we will explain in the following Sections.

[2]The definition of a `wanda_link` in terms of Xlink has not been firmed up yet but there are provisions for it in the DTD.

| | |
|---|---|
| region | - attributes: id, label, next<br>- delineated by a set of points (rectangle or polygon)<br>- content: filters, annotations, other regions, wanda link, meta |

Table 2: **Wanda region summary.**

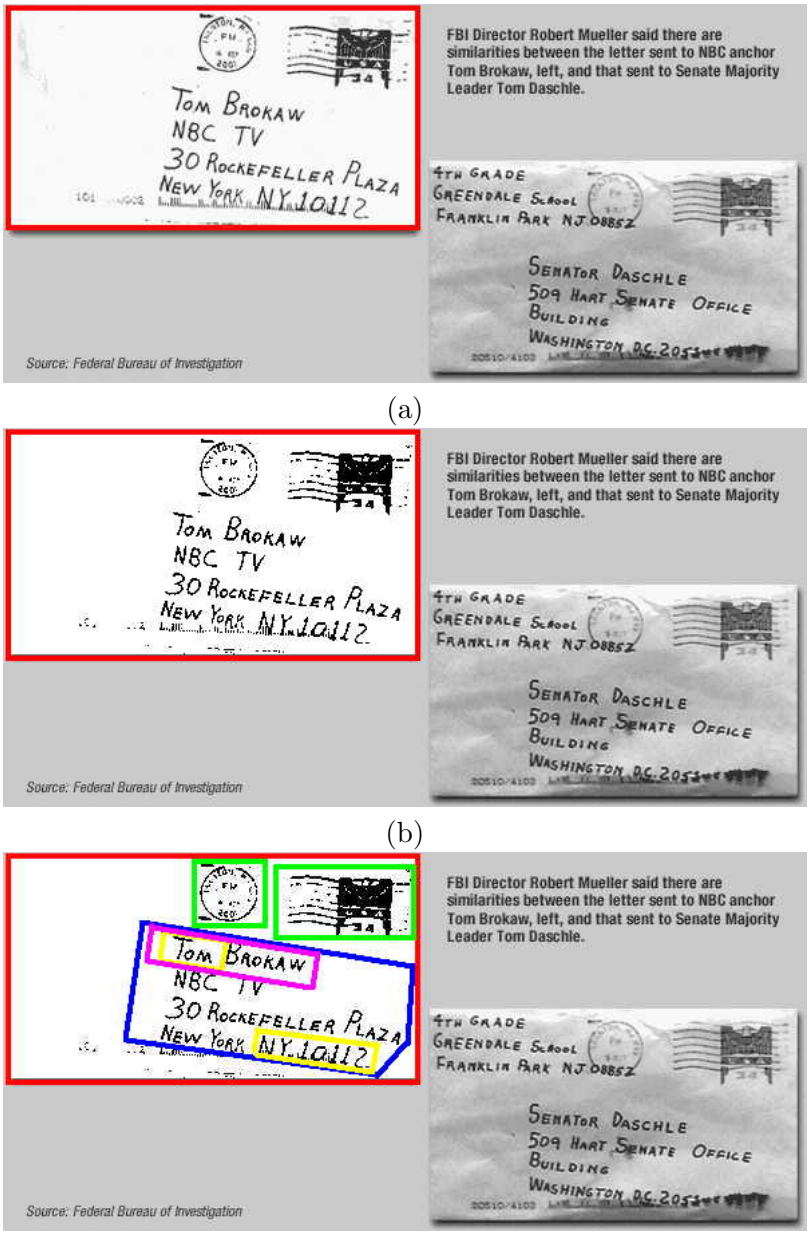



Figure 1: **Wanda document annotation using regions .** These envelopes from actual suspects in the recent anthrax criminal case provide us with an example of possible use of our WANDA regions. (a) A region of interest can be delineated, isolating one envelope. (b) A filter can be applied to the region to clean the data. (c) A hierarchy of regions (e.g. in address block, lines, words, and characters) can be defined and annotated.

| | |
|---|---|
| writer | - person (first and last name, gender, year of birth)<br>- properties (handedness, skill)<br>- education (country, level)<br>- language (native) |
| material | - paper (type, size, material, weight, product, absorbency)<br>- pen (type, product, tip features, ink features)<br>- pad (type, surface, hardness) |
| script | - type (e.g. Latin, Greek, Arabic, etc.)<br>- language (language in which the script is written)<br>- style (major style, connection, capitalization, consistency, stroke quality embellishment, connectivity, speed) |
| content | - document information (type, intent)<br>- text block attributes (type, length)<br>- text block content analysis (tone, grammar, spelling, verbatim transcription) |

Table 3: **Wanda annotation categories.** We summarize in this table the attributes collected in the various annotation categories. See the appendix for the XML syntax.

### 3.2.3 Wanda annotations

Annotations are inserted in a `wandoc` XML document by making use of name spaces. For example, one would write:

```
<annotation type="content" xmlns="../content.dtd">
  <document type="envelope" intent="personal" />
</annotation>
```

In this example, `<annotation/>` is a generic tag of `wandoc` while `<document/>` is specific of the `<content/>` annotation subset. The attribute `xmlns` stands for XML name space. Its value is a unique name reserved to identify the definition of a particular XML. In this example, we use the URI (Uniform Resource Identifier) of a DTD file. URLs (Uniform Resource Locators) are also frequenly used for that purpose. For a tutorial on name spaces, see http://www.zvon.org/xxl/NamespaceTutorial/Output/.

Four categories of annotations that are specific of forensic applications have been defined: `writer`, `material`, `script`, and `content`. All of them are described in details in appendix. We summarize the content of these annotations in Table 3.

The annotations we propose may be used as a simple checklist for the forensic expert: These are information that document the case and may be used for further reporting.

Another intent is to use such annotations to facilitate retrieving documents by database queries. Examples of queries may include:

- Find the writers within a given age range of a given gender.
- Find writers who wrote with a similar pen.
- Find writers who wrote in a particular language and/or with a particular script.

Finally, some annotations will also be used for research purpose. Using data collected from a large variety of subjects, it will be possible to compute relevant statistics about the data that will enrich the panel of available forensic techniques. Examples of such statistics may include:

– Determine whether the use of a given script (e.g. cursive) is biased by a given writer population attribute (e.g. gender).

– Determine whether the use of certain writing material (e.g. recycled paper) is characteristic of a certain writer population attribute (e.g. age).

### 3.2.4 Wanda filters

Filters play an essential role in our framework since they allow describing data processing. A filter acts on the region in which it is defined and may take a number of additional inputs. Those inputs may be parameters provided by a user interacting with a graphical user interface (GUI). The filter is identified by its module locator, which does not preclude of any programming language. Finally, the filter returns outputs of various types. Two of those types are predefined in WandaXML : features and links to files returning the result of filtering. Such files may be images of XML documents.

We adopted the following skeleton for filters:

```
<filter> [+]
    <inputs> [+]
        <input/> [+]
    </inputs>
    <module/> [+]
    <outputs> [+]
        <output/> [+]
    </outputs>
</filter>
```

The tags `<input/>` and `<output/>` wrap around application specific tags defining inputs and outputs. Via the use of name spaces, new application specific types of inputs and outputs can be defined to extend WandaXML , without changing the core of the language. We include in appendix the documentation of two application specific formats: `proper`, the format encoding image preprocessing (e.g., background removal, denoising, contrast enhancement), and `nicifeat`, the format encoding feature measurements (e.g., length of ascenders and descenders, handwriting slant, character height, loop area). The details of their use in the context of the application they support can be found in the application manuals.

### 3.2.5 Wanda virtual ink format

The `wink` format was designed to record and annotate virtual ink for Wanda applications. Particular attention was given to facilitate superimposing virtual ink on top of a background image. In forensic applications, this feature may be used, for instance, to allow forensic experts to retrace characters to reconstitute a plausible handwriting sequence.

However, the `wink` format may also be used independently of the context of a background image. For instance, virtual ink may be recorded by an opaque tablet and encoded with `wink`.

A `wink` block consists of device information, as described in Table 4, information about the trace format (tablet orientation and channels used), and one or several data blocks. Data blocks contain one or several traces that are sequences of pen coordinates. Within each data block, the origin and size of the active bounding box may be redefined. This allows to account for eventual changes in screen context when tablets possessing a display are used. A wink block has the following skeleton:

| | |
|---|---|
| device_info | - General: id, manufacturer, model, size of the sensitive area,<br>sample frequency, sample uniformity, empiric displacement error.<br>- Channel info: name, type, min value, max value, accuracy, resolution. |

Table 4: **Device information summary.**

```
<wink>
    <device_info/>
    <trace_format>
        <tablet_orientation/>
        <channels/>
    <trace_format/>
    <data_block>
        <trace_context>
            <image_scale/>
            <image_offset/>
            <tablet_scale/>
            <tablet_offset/>
            <bounding_box/>
            <brush_size/>
        </trace_context>
        <trace/>
        <!-- ... -->
        <trace/>
    </data_block>
</wink>
```

To import a `wink` segment of handwritten data into a `wandoc` XML document, a filter needs to be written, playing the same role the `scan` plays for raster image data. The technical details of the trace format and the wink coordinate system allowing the superposition of virtual ink and images are described in Section 5.3. Details about XML format are found in appendix.

### 3.3 Recommendations for further extensions

By design, WandaXML revolves around a small number of concepts and is extensible, via the use of name spaces. New filters and annotation types can therefore be easily incorporated. However, WandaXML will evolve and it may be beneficial in the future to incorporate some features that we purposely did not include in this first version.

One such feature is the introduction of segments in virtual ink annotation. Segments are the equivalent of regions for virtual ink. They delineate a chunk of virtual ink in the time (or sequence) domain. Having a notion of segment may turn out to be useful in handwriting analysis. However, at this stage, we found that it would introduce another level of complexity. Rather, virtual ink is presently segmented only by regions defined on the page in the spacial domain.

Another feature is the notion of layers. Presently, the annotations and the filters apply to regions. One could imagine to group regions into layers and allow annotations and filters to apply to layers. Such definition of subsets of regions violates the XML hierarchy. A simple implementation of layers using the existing format is to make use if the region `label`. The label can encode a layer

membership.

Yet another feature is the possibility of dynamically recomputing the results of data filtering. Presently, we store the result of filtering as a filter `<output/>`. The output could be omitted and recomputed on-the-fly. A mechanism for chaining filters, perhaps using output identifiers, would be required.

The present format focuses essentially on encoding the results of annotations, not the process of annotating. Therefore, there are no provisions for un-doing actions. Keeping track of the order in which the regions are defined, the annotations entered, and the filters applied would allow users to undo their actions. In a more elaborate framework, one could even have branching series of actions. Additionally, recording user interaction and being able to play them back would be useful.

The present format implicitly makes the simplifying assumption: one page, one image. However, a page may be defined by multiple images, e.g. at different resolutions. A notion of underlying canvas is needed for such more elaborate applications.

WandaXML can be extended to use styles like Xmillum to encode annotation rendering. Alternatively, Xmillum could be used in conjunction with WandaXML .

# 4 Application of the conceptional framework in forensic handwriting analysis

The WandaXML standard might be the essential underpinning in diverse areas of daily forensic casework as well as in research/development of methods and systems for forensic analysis. In the following we will briefly describe the usage of WandaXML for the report generation and documentation of handwriting characteristics obtained by expert rating and/or by technical equipment (section 4.1). Then, we will explain how the characteristics stored in the Xml format might be transferred and applied in large-scale writer identification (section 4.2). Finally, we point to potential applications of WandaXML and its impact in research and development.

## 4.1 Quality control and expert opinions

WandaXML provides a conceptional framework for dealing with the varieties of handwriting characteristics and their literal and/or numerical descriptions. With the introduction of feature categories and admissible category elements, WandaXML will help to overcome lexical ambiguities and imprecision in forensic science. Moreover, due to the standardization of the data format, which is both human readable and can be automatically processed by computer systems, the collaborative human-machine processing of specimens in forensic handwriting analysis will be further promoted.

The analysis performed on a questioned handwriting sample, e.g. a signature on a bank check or a last will, can be primarily grouped in (a) physical-technical analysis, as for example the spectral analysis of the employed ink (compare figure2), and, (b) analysis of handwriting's graphical components (compare figure3). Further on, the graphic-related handwriting examinations are distinguished into (b.1) shape-based, and (b.2) stroke-morphology-based analysis [8, 5].

WandaXML is able to implement results of analysis performed on a handwriting specimen as well as to communicate those with a general casework management system. Due to the open concept using <filters/ > and <annotations/ > WandaXML is also capable to fulfill upcoming demands.

In the current version WandaXML elaborates on the Fish [4, 10, 9, 7] implementation, and, introduces further categories and admissible values. As mentioned above, the categories implemented in WandaXML are related to (a) the particularities of a disposer of an handwriting - in

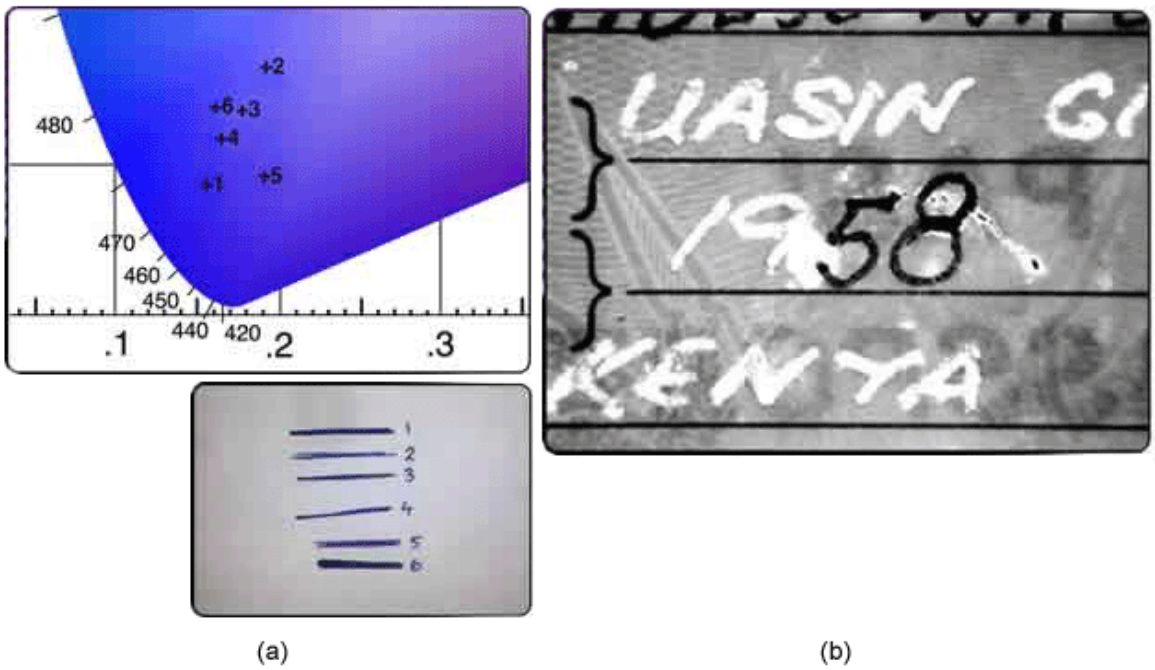



Figure 2: **Physical-technical handwriting analysis** e.g. using (a) chromaticity diagrams and (b) spectral analysis for characterizing and comparing inks on documents - Source: http://www.fosterfreeman.co.uk/

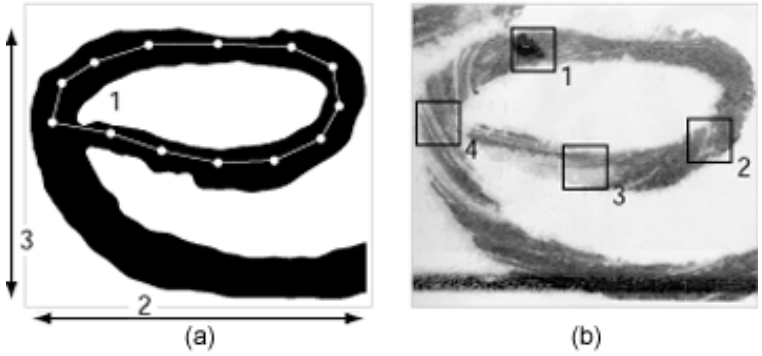



Figure 3: **Graphical handwriting characteristics.** (a) shape-related e.g. (1) loop size, (2) horizontal and (3) vertical extension; (b) stroke-morphology-based e.g. (1) ink drops, (2) ink gaps, (3) ink-free beginnings or (4) centrifugal lines.

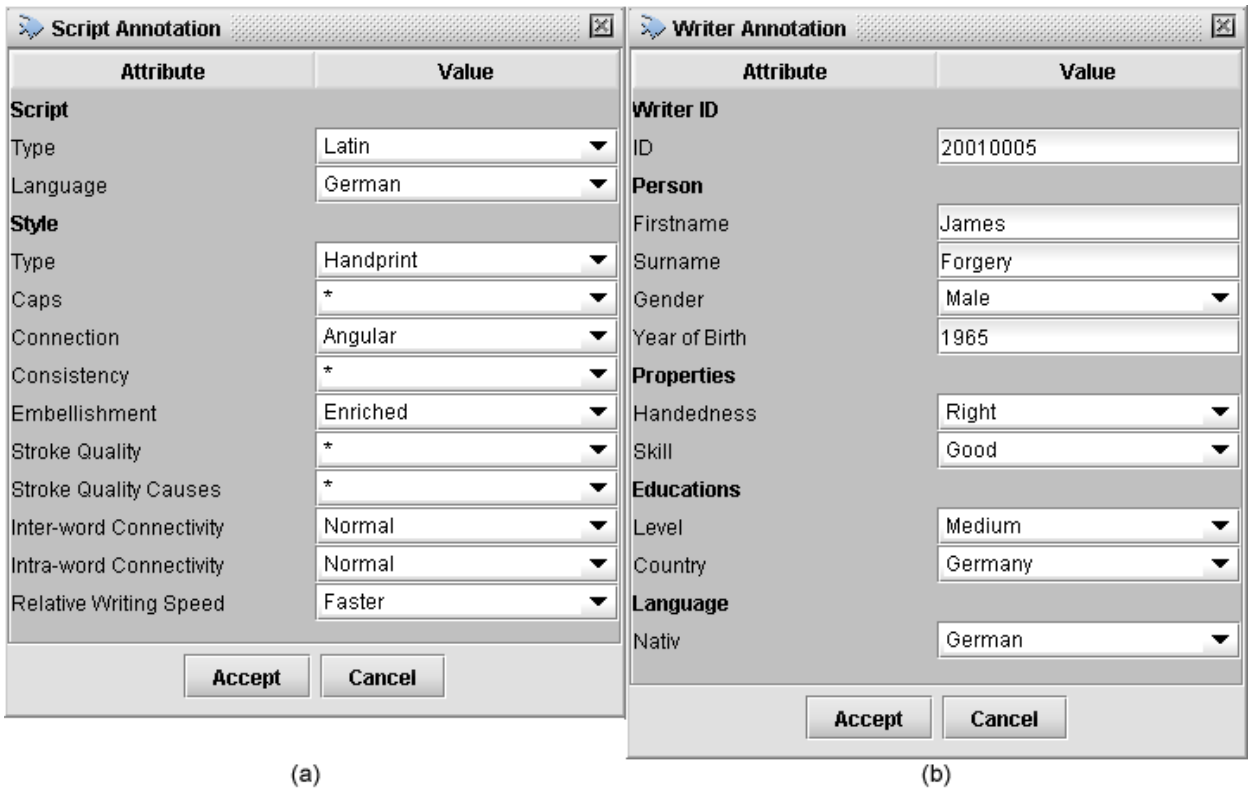



Figure 4: **Graphical user interface for annotations.** (a) graphical handwriting characteristics and (b) writer characteristics.

case he/she is known; (b) the graphical characteristics of the examined handwriting [5]; (c) the material characteristics that might be analyzed/determined by other examination procedures, and (d) the context of the handwriting as well as content aspects which might be of interest in a linguistic analysis. These four categories are well established in forensic science. Other categories as the one supplied for (e) digitalization, (f) cleaning and (g) interactive measurements are of great importance for a ongoing computer-based examination.

According to current forensic practice and the required quality assurance, such called feature protocols have to be provided for each handwriting investigation/comparison. Those protocols are being supplied for the questioned handwriting sample as well as for the reference handwritings. This procedures are extremely time consuming. So, interactive and semi-automated computer procedures might partly support the complete documentation of frequently needed aspects of handwriting.

For the semi-automatic generation of an expert report one can think of an easy-to-use graphical user interface where the expert only selects the specific categories and values that describe the handwriting product (see for an example figure 4). On users demand the annotated data might be transferred into the case management system. Also, a text document might be generated, which can be further completed by using a standard text processor as for example MS Word. In this way it will be possible to speed up the documentation of forensic evidences as well as to harmonize those documentations.

The internal representation of handwritings features, using a standardized XML format, supports the exchange of examination result between different laboratories and/or governmental entities. The chosen XML-format will ensure that data can be rapidly imported into another computer

| Stage | Description | Processing mode |
|---|---|---|
| 1.a | case entry | interactive |
| 1.b | scanning | interactive |
| 1.c | preprocessing | interactive |
| 2.a | annotation | interactive |
| 2.b | measurement | interactive |
| 3 | normalization* | batch |
| 4 | pre-selection | interactive |
| 5 | identification matching | batch |
| 6 | hit-list inspection | interactive |

Table 5: **Stages in the forensic processing of handwritten samples.** The normalization stage (*) is not covered in the Wanda project.

system that does not need to be provided by the same manufacturer. So, WandaXML promotes interoperability, and, with the anchors for further extensions, long-term usability.

### 4.2 From XML file directory to Working-Set database: xml2ws

The concept behind the Wanda system architecture is inspired by a number of insights. First, the need for longitudinal use of forensic annotated handwritten samples requires a stable and open data representation. The extended markup language XML has emerged as an ideal format. It is based on the basic but convenient and legible XML text file. Formal syntax requirements allow for parsing and utilization of the XML content, depending on the changing application demands. This is opposed to long-term use of commercial and legacy databases, where the actual data formats are binary, obscure and subject to version changes initiated by the software industry. Economic instability has revealed that a high dependence on industrial technology entails risks.

The second insight is that in the actual use of forensic handwriting systems, there is a clear work flow, consisting of free interactive computer use as well as formalized, constrained working stages. Table 5 shows the stages in the work flow of handling forensic handwritten samples. Each stage is characterized by its typical requirements.

Users generate knowledge concerning the handwritten samples, by selecting regions of interest, annotating the content symbolically by using word categories and performing qualitative and quantitative measurements. However, the goal of these activities is to create an internal representation for each sample, which allows for fast and effective selection and matching within a large set of cases. In the work flow, interactivity and dependency on a graphical user interface plays a dominant role in the initial stages (1, 2). Batch processing on large data sets plays a dominant role in the late stages (3 and 5). Stages 4 and 6 (pre-selection and hit-list inspection) are interactive. However, the interaction complexity is much lower here than is the case in the initial stages, such that the interface can be realized by using a standard internet browser.

During the development, it became apparent that the requirements of fast and effective case selection (stage 4) cannot be met by the current XML software, i.e., at the scale which is required by the application domain ($>>$ 20000 cases may be present).

The challenge then is to combine (a) the advantages of XML transparency and data handling convenience with (b) the advantages of traditional database methods, which excel in fast data

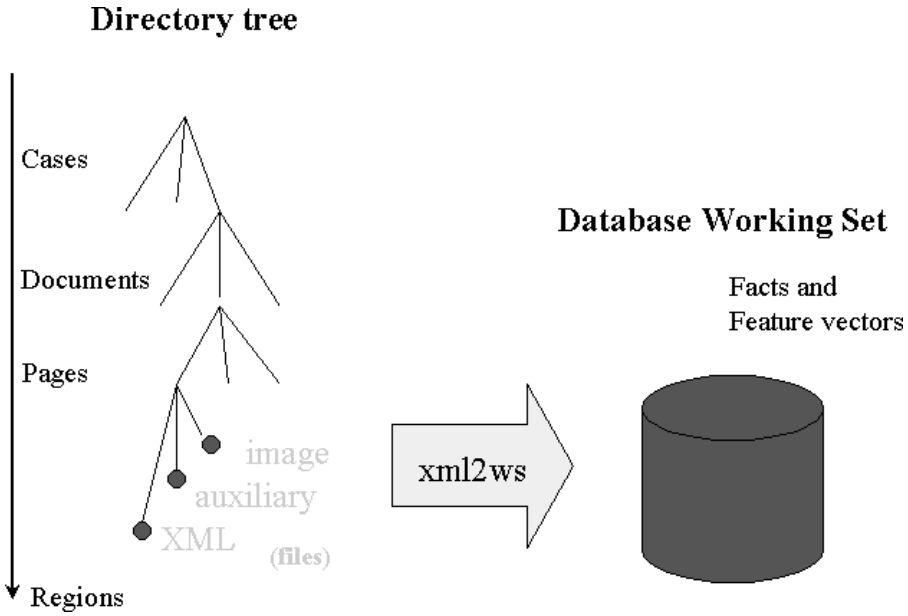



Figure 5: **The function of the xml2ws tool depicted schematically.** The content of the XML files, i.e., relevant facts and feature vectors, are transported into the **Working-Set** database.

handling and SQL-based logical subset selection.

The proposed solution is to assume that a directory tree of XML files and their corresponding image files represents the core data, for interactive handling but also for long-term storage. During the coming decades, programmers will be able to recognize the structure of the XML. The image format (TIFF) is a de facto standard, for which open-source access and visualization methods exist. In order to achieve the goal of fast and effective case selection and matching, it is proposed to utilize a **Working-Set** database. The **Working Set** is a traditional database, which is volatile and replaceable. Fast search and matching is performed on the data in the working set, not on the individual XML files. The **Working-Set** database is generated from the XML data sets. This can be done in two ways:

1. Interactive storage of an individual case in the **Working Set** upon completion of measurement. This is called a "commit" action. A sample is committed to the database.

2. Batch commitment of a complete XML directory tree or sub tree to the **Working-Set** database.

In order to realize an open implementation which can be maintained easily for a longer period in time, we chose to use java as a language for an XML-to-Working-Set (**xml2ws**) converter. This program utilizes existing free class libraries for database access. The proof-of-concept tool **xml2ws** was written for Postgres SQL, but other implementations such as MySQL are not very different from this approach. Figure 5 shows the function of the **xml2ws** tool which is located between the XML directory tree and the database.

The commitment of a single case to the database can be performed by embedding **xml2ws** as a Wanda server plugin which is called from the GUI when the user hits a "Commit" button. However, since **xml2ws** is a stand-alone executable program, it can also be used from the command line and

```
> find ./docs_Firemaker | grep xml | more
./docs_Firemaker/case0152/docp1copy-normal/page0001/0152_docp1copy-normal_0001.xml
./docs_Firemaker/case0152/docp2copy-upper/page0001/0152_docp1copy-normal_0001.xml
./docs_Firemaker/case0152/docp3copy-forged/page0001/0152_docp1copy-normal_0001.xml
./docs_Firemaker/case0152/docp4self-natural/page0001/0152_docp1copy-normal_0001.xml
./docs_Firemaker/case0153/docp1copy-normal/page0001/0153_docp1copy-normal_0001.xml
./docs_Firemaker/case0153/docp2copy-upper/page0001/0153_docp1copy-normal_0001.xml
./docs_Firemaker/case0153/docp3copy-forged/page0001/0153_docp1copy-normal_0001.xml
./docs_Firemaker/case0153/docp4self-natural/page0001/0153_docp1copy-normal_0001.xml
./docs_Firemaker/case0154/docp1copy-normal/page0001/0154_docp1copy-normal_0001.xml
./docs_Firemaker/case0154/docp2copy-upper/page0001/0154_docp1copy-normal_0001.xml
./docs_Firemaker/case0154/docp3copy-forged/page0001/0154_docp1copy-normal_0001.xml
./docs_Firemaker/case0154/docp4self-natural/page0001/0154_docp1copy-normal_0001.xml
./docs_Firemaker/case0155/docp1copy-normal/page0001/0155_docp1copy-normal_0001.xml
./docs_Firemaker/case0155/docp2copy-upper/page0001/0155_docp1copy-normal_0001.xml
./docs_Firemaker/case0155/docp3copy-forged/page0001/0155_docp1copy-normal_0001.xml
./docs_Firemaker/case0155/docp4self-natural/page0001/0155_docp1copy-normal_0001.xml
. . .
```

Table 6: An example of the Unix **find** output used together with **grep** to obtain the XML file names. Such a list can be sent to **xml2ws** in order to transport the XML tag contents to "SQL" database fields.

in scripts. Therefore, the commitment of a complete directory tree to the **Working-Set** database can be realized using generic operating-system tools such as the Unix 'find' function. An example Linux script of this method is provided. In principle, a complete java implementation of a batch approach is possible, which relaxes the dependency on Unix/Linux in future implementations, if needed.

As a consequence, this output can be sent to **xml2ws** by prefixing each of the lines with the xml2ws command. An example is:

```
find . | grep '\.xml' \
       | sed "s=^=xml2ws jdbc:postgresql://localhost:5432/wanda $user=" \
       | csh -s
```

The streaming editor **sed** puts the commandname xml2ws and a database identifier (with ipname:portnr) at the beginning of each line. The variable `$user` contains the name of an authorized user. The resulting filtered output is sent to the commandline-interpreter shell **csh**. The reader is referred to the script **all-xml-to-working-set** for an actual example in the software directory. Please note that for starting a java command, the "class path" must be correct.

Figure 6 shows the work flow, from the system's perspective. After **xml2ws** has added a case to the **Working Set**, the data can be normalized. On the basis of a regular internet browser, the user can then ask for a reduced set of potentially relevant cases (Preselection). After the subsequent identification or matching process, a hit list is generated, which may be browsed by the user.

In conclusion, the **Working-Set** concept allows for an exploitation of the best of two worlds. Rather than viewing the database as the oracle which stores information for a prolonged period of time, the working set is a more volatile object, for fast and effective data-set selections. As database technology progresses, such implementations may be adopted, with limited consequence for the overall software scheme. The open setup of the XML directory tree guarantees that the commitment of existing data to a new and possibly improved type of working set database is a feasible option.

### 4.3 Research and Development

Existing systems of forensic data analysis can already benefit from WandaXML : an exemplified in the previous sections, having such a format facilitates batch processing of data and can be combined with any application.

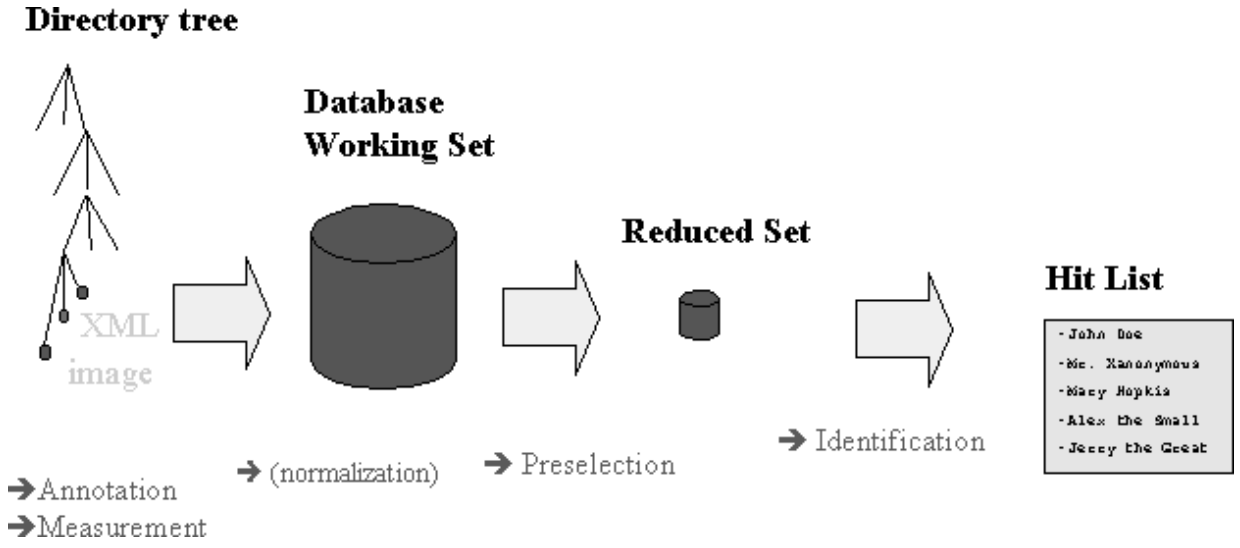



Figure 6: **The work flow depicted from the system perspective.**

Besides representing existing practices in forensic data analysis in a standard way, WandaXML can facilitate experimenting with new ideas. For instance, the analysis and annotation of complex documents with structured layouts such as envelopes is supported by the notion of regions (see our example of Figure 1). Also, regions do not need to be limited to handwriting contents. Stamps, printed data, and various other marks can be conveniently annotated.

Another possible use of the framework that we have not fully explored yet is the definition of templates or XML empty skeletons. Structured documents such as forms and envelopes have a well defined hierarchy of regions that can be pre-defined in a template or skeleton. Users would then have to fill the skeleton with annotations for particular images.

In the years to come, the WandaXML format will be particularly useful for research and development since it will allow researchers to parse and annotate large bodies of data, partially manually, partially automatically. Such data will lend themselves to statistical data analysis to automate writer identification, extract new features of interest, infer new correlations between handwriting attributes, and improve handwriting recognition.

The existence of publicly available data formatted in a standard way will stimulate research for forensic applications. We foresee the possibility of organizing benchmarks based on such data. A standard data format also facilitates and stimulates data exchange. We anticipate that their will be a growing body of data incorporating contributions of many institutions.

Additionally, the WandaXML format will allow researchers to track their experiments. During data exploration, WandaXML enables recording the application of particular sequences of filters. This makes it possible to play back sequences of data processing and eventually modify them. This also ensures the reproducibility of results.

Finally, the format will allow researchers to refine the sequence of routine working stages in order to obtain consistent forensic analyzes. The defined protocol will then be cast into an analysis wizard, which will guide the forensic expert step by step.

# 5 How to use and expand WandaXML

This Section provides more technical details about WandaXML . It also provides the reader with practical tips and tools to define his/her own XML language for the purpose of describing document annotations and filtering derived from WandaXML . We do not limit ourselves to forensic applications. Documents have a more encompassing meaning, including:

- Typed, machine generated or machine printed documents (e.g. forms, web pages) possibly containing tables.
- Images lending themselves to pattern analysis or recognition (e.g. fingerprints, DNA microarrays), or images lending themselves to scene analysis (e.g. license plate recognition).
- Time dependent signals lending themselves to pattern recognition and making predictions, like speech, video, spectrograms, financial data.
- Multimodal data combining several of the above (e.g. combinations of on-line (time series) and off-line (image) handwriting data or combinations of speech and gesture data).

## 5.1 Conventions

Following the general XML nomenclature, in XML annotation we have:

```
<my_tag my_attribute="my_value">
    <my_element/>
</my_tag>
```

where, `<my_tag/>` and `<my_element/>` are tags or "elements", and `my_attribute` is an attribute of `my_tag`, having value `my_value`. In addition, in Document Type Definitions (DTDs), "entities" are defined, which may be used as lists of admissible attribute values.

We adopted a minimum set of conventions carried throughout WandaXML :

- Entities, attributes and elements contain only lowercase and underscore characters (some DTDs that were defined before adopting this convention may still contain capital letters, but they will be eliminated in the future). This facilitates constructing form filling graphical user interfaces directly from the DTD.
- Certain attributes have special meanings: `id` a unique identifier, `type` a type from a predefined list, `label` an optional user-defined string that can be used for search purpose.
- To simplify parsing, we have defined a number of "containers" (`<pages/>`, `<filters/>`, `<annotations/>`, `<inputs/>`, and `<outputs/>`), which contain elements of the same name (`<page/>`, `<filter/>`, `<annotation/>`, `<input/>`, and `<output/>`) and have the optional attribute `number_of`.

## 5.2 An simple example of wandoc XML

To best describe the Wanda annotation mechanisms and concepts, we describe step by step the example found in the `wandoc` reference (see appendix).

By convention, we use the short hand `<my_tag/>` (terminated by `/>`) to designate a tag that may contain other elements, but is not expanded at this stage of the description of the example. At later stages, we may omit certain attributes and replace them by `...`.

A document annotation file always start with a root element `<wandoc/>`:

```
<wandoc
    id="20032004_0001"
    label="WANDA test and development sample"
    xmlns="http://pentel.ipk.fhg.de/\wandaxml/wandoc/wandoc.dtd"
/>
```

The `id` is a machine generated unique document annotation identifier, while the `label` is provided by the user. The name space `xmlns` points to the `wandoc` language definition.

In our example, the `<wandoc/>` element contains a single page:

```
<wandoc ...>
    <pages numberOf="1">
        <page id="20032004_0001_copy51" label="frontpage" next=""/>
    </pages>
</wandoc>
```

The page contains one call to a filter, three annotations, and one region:

```
<wandoc ...>
    <pages numberOf="1">
        <page ...>
            <filters number_of="1"/>
            <annotations number_of="3"/>
            <regions number_of="1"/>
        </page>
    </pages>
</wandoc>
```

We detail below the filter content. This filter is importing an image from a scanner using the IBIS scan software and returning a link to the resulting image.

```
<wandoc ...>
    <pages numberOf="1">
        <page ...>
            <filters number_of="1">
                <filter type="import" label="ibisScan">
                    <inputs>
                        <input
                            type="stream"
                            number="1"
                            xmlns="../scan.dtd">
                            <scan/>
                        </input>
                    </inputs>
                    <module type="extern" exec="ibis.exe">
                       <meta version="3.51" />
                    </module>
                    <outputs>
```

```
                        <output type="file">
                            <wanda\_link href="copy51.tif" />
                        </output>
                    </outputs>
                </filter>
            </filters>
            <annotations number_of="3"/>
            <regions number_of="1"/>
        </page>
    </pages>
</wandoc>
```

We do not develop the annotations, see the appendix for details. We develop the region content:

```
<wandoc ...>
    <pages numberOf="1">
        <page ...>
            <filters number_of="1"/>
            <annotations number_of="3"/>
            <regions number_of="1">
                <region
                    id="20032004\_0001\_copy51\_0001\_213746432"
                    label="Hello...happy with it."
                    next="2"
                >
                <points>
                    <point x="0" y="0" />
                    <point x="10" y="0" />
                    <point x="0" y="10" />
                    <point x="10" y="10" />
                </points>
                <annotations number_of="3"/>
                <filters number_of="1"/>
            </regions>
        </page>
    </pages>
</wandoc>
```

A region is defined by a unique id (presumably machine generated). Id's allow users to refer to regions for instance to pass them as filter argument. Otherwise, by default, filters apply their parent region, page or document. Additionally, regions may possess a user-defined label, the intend of which is to facilitate searching through regions. The attribute "next" is used to indicate a logical ordering of the regions, which are at the same hierarchical level. Such ordering is used, for instance, to indicate reading order.

Regions are delineated by a set of points. The origin is at the upper left corner. The unit, if not specified, is the pixel.

The region of our example possesses three annotations and one filter. The filter corresponds to some measurements and returns features (not shown). The interested reader can look at the full example in appendix.

## 5.3 Guidelines to use the Wanda virtual ink format (wink)

The `wink` format is used in WANDA to encode virtual (electronic) ink. In this section, we describe the `wink` trace recording mechanism and the `wink` coordinate system.

### Trace format

We recall that `wink` encodes virtual ink according to the following skeleton:

```
<wink>
    <device_info/>
    <trace_format>
        <tablet_orientation/>
        <channels/>
    <trace_format/>
    <data_block>
        <trace_context>
            <image_scale/>
            <image_offset/>
            <tablet_scale/>
            <tablet_offset/>
            <bounding_box/>
            <brush_size/>
        </trace_context>
        <trace/>
        <!-- ... -->
        <trace/>
    </data_block>
</wink>
```

A trace is simply recorded as a sequence of pen coordinates. Pen coordinates consist of several channels, including:

- x position (in unit "u"),
- y position (in unit "u"),
- z (altitude above the tablet in mm),
- pressure levels (in arbitrary device dependent unit),
- pen azimuth (in degrees),
- pen tilt (in degrees),
- time (in seconds).

The unit "u" is the `wink` canvas unit defined in the next section. A sequence of x y p t coordinates would be recorded as:

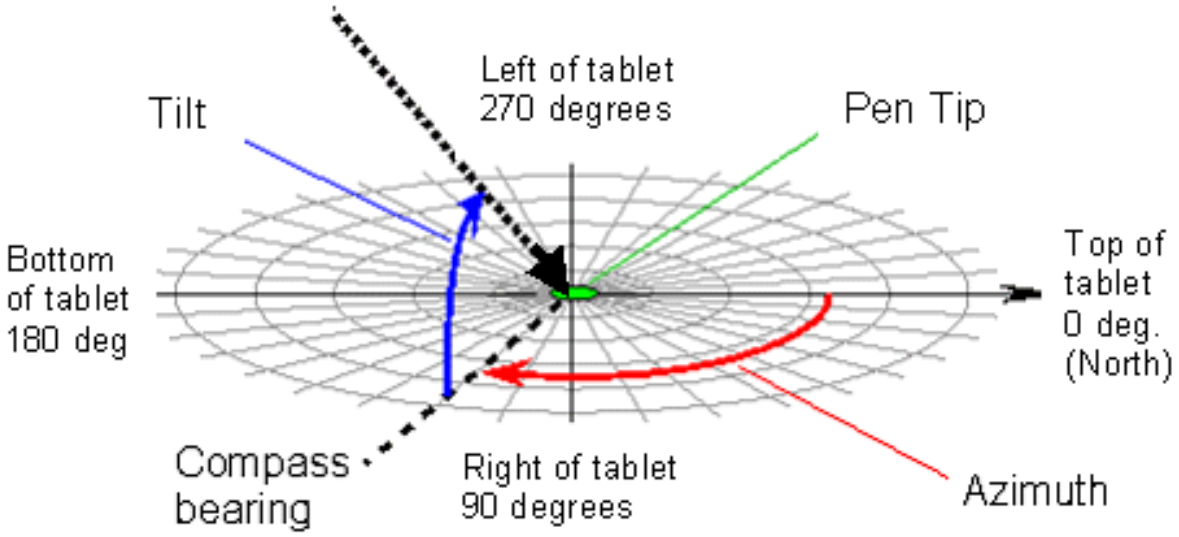



Figure 7: **Pen orientation encoding.** The figure shows how pen orientation is recorded using tilt and azimuth information.

```
<trace>
    x y p t
    x y p t
    ...
    x y p t
</trace>
```

where x, y, p, and t are absolute coordinates (not deltas), relative to an origin placed in the upper left corner of the bounding box defined in the next section. The subset of channels used in the trace would be declared in the `<traceformat/>` as:

```
<channels number_of=3>
    <channel name="x"/>
    <channel name="y"/>
    <channel name="p"/>
    <channel name="time"/>
<channels/>
```

Traces also have an optional attribute "type" that specifies whether the pen is up or down for devices that record both pressure levels and whether the pen is touching. To precise that the pen was lifted between two pen down traces, one may want to impose the insertion of a pen up trace, even if the device does not record pen up trajectories. In such a case, the pen up trace would simply always be empty. Alternatively, if the time stamps are recorded, in order to record the elapsed time between two consecutive pen down traces, one could include the last point of the previous pen down trajectory and the first point of the next pen down trajectory. The pen orientation encoding conventions are shown in Figure 7.

The trace format also contains tablet orientation information (see the appendix for details).

### The wink coordinate system

A `wink` record possesses an invisible canvas on top of which the virtual ink and optionally a background image are placed. All measurements on the canvas, including the pen tip coordinates forming the trace, are recorded in **the canvas unit "u"**. This unit is defined relatively to tablet and image units via two scaling factors `tablet_scale` and `image_scale`. A trace is recorded within the boundaries of a `bounding_box`, which must be contained both within the tablet sensitive area and within the image area, if there is a background image. The size of the bounding box is recorded in unit "u". The position of upper left corner of the bounding box with respect to the upper left corner of the tablet called `tablet_offset` is also recorded in unit "u". Similarly, the position of the upper left corner of the bounding box with respect to the upper left corner of the image (or active image region) called `image_offset` is recorded in unit "u".

Thus, in the case where a `wink` record is embedded within a `wandoc` region, the upper left corner of the region is retrieved by applying `image_offset`. But when `wink` is used to encode virtual ink written on top of a raster image outside of the context of `wandoc`, no region is defined and the origin of the bounding box is defined relatively to the origin of the image. A possible use of wink would be to pair an image file (e.g. `my_image.png`) to a wink file (e.g. `my_image.wink.xml`).

Note that at this stage we assume that both pixels and tablet dots are squared and we provide only one scaling factor for both x and y directions.

We illustrate the coordinate system in Figure 8. To summarize the arithmetic:

$$tablet\_coordinates = (canvas\_coordinates + tablet\_offset) * tablet\_scale \quad (1)$$

$$image\_coordinates = (canvas\_coordinates + image\_offset) * image\_scale \quad (2)$$

### Unit choice scenarios

The wink coordinate system gives a lot of flexibility. Typical choices for the wink canvas unit "u" include:

1. The unit "u" is the pixel size of the background image.
2. The unit "u" is the digitizing tablet dot size.
3. The unit "u" is the millimeter on the scanned document.

All three choices allow users to represent virtual ink that is overlayed on top of an image. It is possible to easily convert from one unit to another. The second choice permits users to represent data not associated with an image (`<image_scale/>` and `<image_offset/>` would be omitted from the `<trace_context/>`). This unit choice is also the default choice if no `<trace_context/>` is provided. The other choices are meaningful only when the virtual ink is overlayed on top of an image. Note in the particular case when the image is retraced with a mouse, `<tablet_scale/>` and `<tablet_offset/>` would be omitted. We have no provision at this time to reconstitute the user's experience with the mouse (size of the picture on screen, mouse parameters).

To illustrate the three example unit choices, we treat a small example in which a trace consisting of only two dots is formatted with wink. Imagine that we have scanned an image at 10 dpmm and that the tablet resolution is of 100 dpmm. At a zoom of 1, the tablet has a resolution of 10 dots per image pixel. At a zoom of 2, it has a resolution of 20 dots per image pixel. Assume that the wink bounding box origin is at a distance x = 300 pixels and y = 200 pixels of the image region origin and a distance x = 1500 dots and y = 2600 dots of the tablet origin. The bounding box has

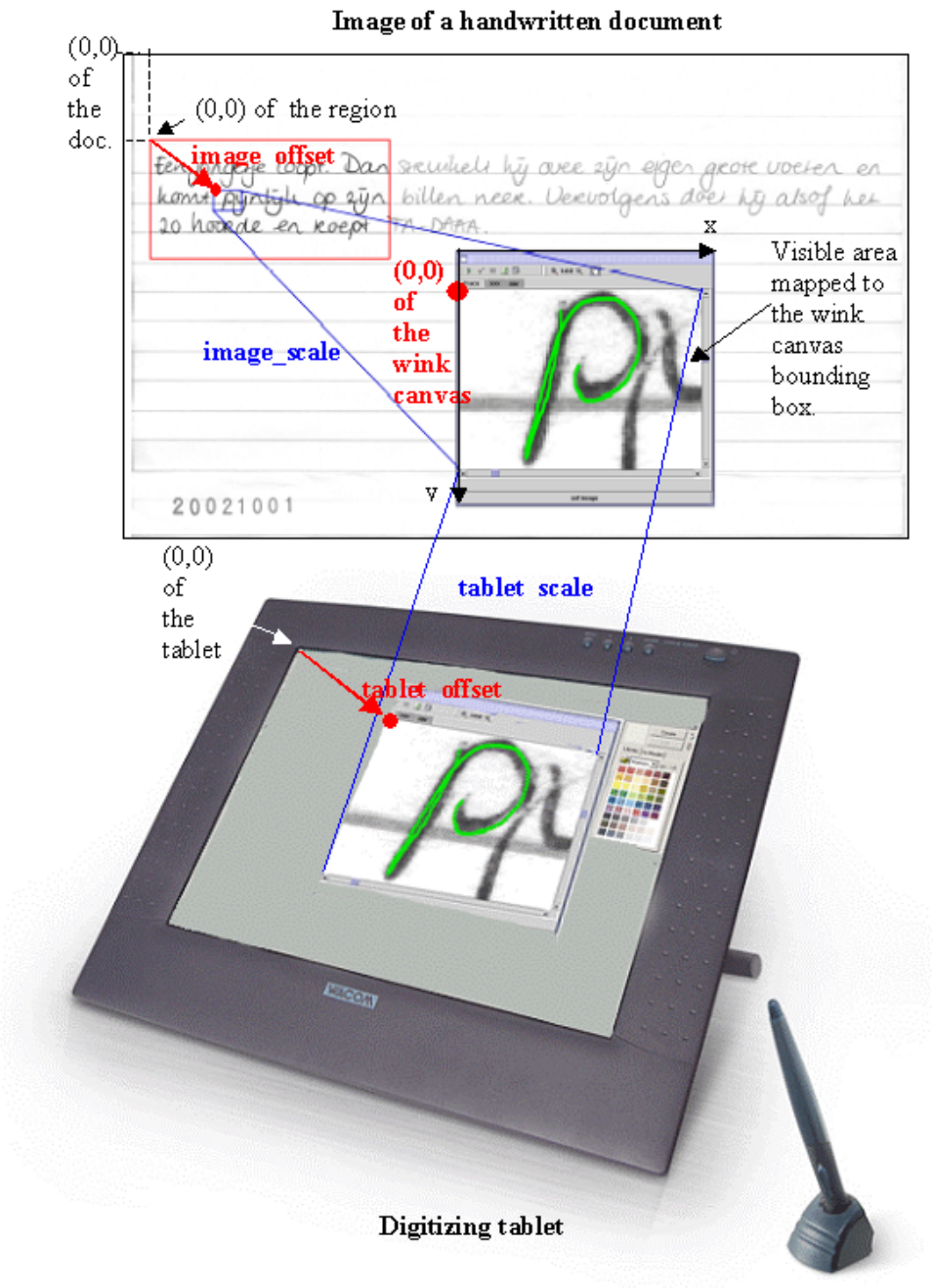



Figure 8: **Wink coordinate system.** The figure illustrates how virtual ink can be overlayed on top of a raster image using the wink coordinate system. Virtual ink is recorded in the wink canvas unit "u" defined relatively to the tablet unit and the image unit via a scaling factor. The canvas offset with respect to the tablet origin and the image origin are also expressed in the unit "u".

a width of 150 pixels and a height of 100 pixels.

Case1: 1 u = 1 pixel

At zoom 1, we have:

```
<trace_context>
    <image_scale value="1" unit="dpu" />
    <image_offset x="300" y="200" />
    <tablet_scale value="10" unit="dpu" />
    <tablet_offset x="150" y="260" />
    <bounding_box width="150" height="100" />
    <brush size="2" />
</trace_context> <trace>
    330 250
    400 280
</trace>
```

At zoom 2, we have:

```
<trace_context>
    <image_scale value="1" unit="dpu" />
    <image_offset x="300" y="200" />
    <tablet_scale value="20" unit="dpu" />
    <tablet_offset x="75" y="130" />
    <bounding_box width="150" height="100" />
    <brush size="2" />
</trace_context> <trace>
    330 250
    400 280
</trace>
```

Case2: 1 u = 1 tablet dot size

At zoom 1, we have:

```
<trace_context>
    <image_scale value="0.1" unit="dpu" />
    <image_offset x="3000" y="2000" />
    <tablet_scale value="1" unit="dpu" />
    <tablet_offset x="1500" y="2600" />
    <bounding_box width="1500" height="1000" />
    <brush size="20" />
</trace_context> <trace>
    3300 2500
    4000 2800
</trace>
```

At zoom 2, we have:

```
<trace_context>
    <image_scale value="0.05" unit="dpu" />
    <image_offset x="6000" y="4000" />
    <tablet_scale value="1" unit="dpu" />
    <tablet_offset x="1500" y="2600" />
    <bounding_box width="3000" height="2000" />
    <brush size="40" />
</trace_context> <trace>
    6600 5000
    8000 5600
</trace>
```

Case3: 1 u = 1 mm

At zoom 1, we have:

```
<trace_context>
    <image_scale value="10" unit="dpu" />
    <image_offset x="30" y="20" />
    <tablet_scale value="100" unit="dpu" />
    <tablet_offset x="15" y="26" />
    <bounding_box width="15" height="10" />
    <brush size="0.2" />
</trace_context> <trace>
    33 25
    40 28
</trace>
```

At zoom 2, we have:

```
<trace_context>
    <image_scale value="10" unit="dpu" />
    <image_offset x="30" y="20" />
    <tablet_scale value="200" unit="dpu" />
    <tablet_offset x="7.5" y="13" />
    <bounding_box width="15" height="10" />
    <brush size="0.2" />
</trace_context> <trace>
    33 25
    40 28
</trace>
```

### 5.4 Useful tools

In this Section, rather than providing a long list of potentially useful tools, we provide a list of tools we have most extensively used and have delivered on their promises. For a comprensive list of free XML tools, see:
http://www.garshol.priv.no/download/xmltools/.

### XML viewers and editors

The most widely available XML viewer is probably **Microsoft Internet explorer**. Just open and XML file with Microsoft Internet explorer to obtain a nice colored display of the XML file. The XML tree is "active": tags can be expanded of contracted like in a directory tree. Microsoft Internet explorer supports the documents defined by DTDs via the declaration DOCTYPE (e.g. `<!DOCTYPE writer SYSTEM "script_.dtd">`) but does not check the consistency of the XML file with the DTD.

Text editors are unsophisticated tools to edit XML, but are sometimes very convenient. Tools that color tags in a meaningful way and indent the XML tree properly are most useful. We do not have a particular recommendation. Many people use **PSGML**, a GNU emacs mode, see:
http://www.lysator.liu.se/~lenst/about_psgml/.

### XML parsers

XML files are often (but not necessarily) associated with files providing precisions about allowable elements, attributes, and entities. Two types of standards are commonly used: Document Type Descriptions (DTDs) and schemas. To view and edit XML and produce DTDs and/or schemas, there exist integrated environments that are more sophisticated than simple editors. These include **XMLSpy** (http://www.xmlspy.com/). Free trial licenses of one month are available. Some features of XMLSpy are particularly useful: A DTD or a schema can be generated from and XML example. Conversely, an XML example can be generated from a DTD or a schema. The syntax of all files are automatically checked and the consistency of XML files with their associated DTD or schema is verified. DTDs can be generated from schemas and vice versa. Schemas can be automatically documented (but not DTDs). For documentation purpose however, we find XMLSpy sub-optimal and prefer using another tool (see the next Section).

### DTD documentation

We documented the XML Document Type Definitions (DTDs) with **dtd2html**. Most of WandaXML is defined through DTDs. Only "proper" uses a schema. For consistency, we converted the proper schema into a DTD using XMLSpy and then documented the DTD.

The program dtd2html is a Perl script that is part of the **PerlSGML** package of Earl Hood. The script is easy to customize, which allowed us to also produce a printable documentation attached in appendix.

To use dtd2html, first install the package:
http://www.nacs.uci.edu/indiv/ehood/perlSGML.html

Then replace the routines dtd2html.pl and dtd.pl by their customized version found on the Wanda server.

If you have access to the on-line WandaXML documentation directory you can try some examples of usage:

- Go to a subdirectory: `cd script`.
- Run at the prompt: `dtd2html.pl script.dtd`. This re-generates the documentation for `script.dtd`. The program looks for `DESCRIPTION.dsc` and `EXAMPLE.xml` in the same directory. The program also creates automatically the file `script_.dtd` that is identical to `script.dtd` with all star characters replaced by "X". The file `EXAMPLE.xml` should point

to `script_.dtd` not to `script.dtd` if you want Microsoft explorer to parse `EXAMPLE.xml` correctly (stars generate errors).

- To create a new `DESCRIPTION.dsc` backbone file, run at the prompt: `dtd2html.pl -elemlist script.dtd > DESCRIPTION.dsc`. The resulting file needs then to be filled out with your annotations. Be careful not to overwrite a filled out description file with an empty backbone `:=)`.
- To update a description file if the dtd has changed, run at the prompt: `dtd2html.pl -updateel OLD_DESCRIPTION.dsc script.dtd > DESCRIPTION.dsc` where `OLD_DESCRIPTION.dsc` is file with old tags filled out with your precious annotations and `DESCRIPTION.dsc` copies whatever did not change, removes old tags and adds new ones.

The appendices of the present document were produced by converting the HTML documentation to Latex using the program html2latex: http://html2latex.sourceforge.net/ that was slightly customized. The libraries TreeBuilder.pm and Latex.pm of the HTML package were slightly modified. The script html2latex.pl was slightly modified and a new configuration file html2latex.xml was written. The modified code is found on the WANDA server with the documentation. To convert the HTML documentation, execute at the prompt:

```
html2latex\html2latex.pl -conf html2latex\html2latex.xml PRINTABLE.html
```

A script `make_doc.bat` is provided to recreate the full documentation.

### 5.5 Examples of other application domains

Straightforward additions can be made to the existing framework so that it can handle images for other application domains, including:

- Printed document analysis.
- Scene analysis (e.g. license plate reading, see Figure 9, face recognition).
- Biometrics (e.g. fingerprint verification, see Figure 10, signature verification).
- Bioinformatics (e.g. gene expression microarrays, protein assays, see Figure 11, microscope images).

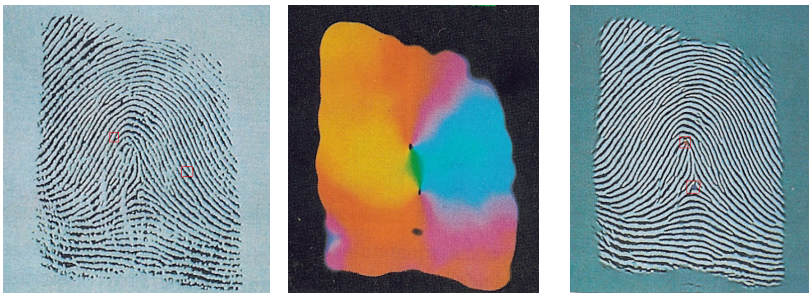
Also one dimensional signals including speech and spectral data could be handled in a similar way. Further down the road, extensions to video signals could be envisioned.

## 6 Conclusion

This document described a new data format WandaXML to annotate forensic handwriting data. The choice of XML makes it intrinsically extensible. Our design is centered around a small number of concepts and conventions. Regions, annotations, and filters are the three essential elements of all WANDAXML annotated documents. Regions are parts of documents delineated with a rectangular of polygonal shape. Regions are naturally organized by the XML hierarchy. This reflects well the hierarchical nature of documents' organization (e.g., page, paragraph, line, word, character). In addition, within a given hierarchical level, regions are organized in linked lists to encode logical

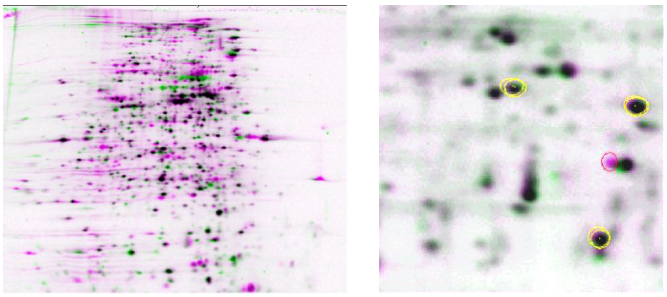

(a) (b)

Figure 9: **License plate reading.** This example shows a potential application of our framework to license plate reading . (a) A car approaches a check point. A camera takes a frontal picture. (b) The license plate region of interest can be outlined and annotated. Other annotation on the car model could be added to the full image. Source: http://www.licenseplaterecognition.com/.



(a) (b) (c)

Figure 10: **Fingerprint analysis.** This example shows how our annotation framework can be applied to the domain of fingerprint analysis. (a) An original image. Regions can be defined to outline minutiae, possibly annotated by hand. (b) A filter can be applied to detect line orientation, which appear color coded on the resulting image. (c) The line orientation map may be used for filtering purpose and we show here the resulting image. Two regions are outlined corresponding to singularities in the orientation map that are automatically detected. Source: Digital Image Processing. Bernd Jahne. Springer Verlag. Second Edition. 1993.



(a) (b)

Figure 11: **Two dimensional gel analysis.** This example shows protein gels obtained by two-dimensional polyacrylamide gel electrophoresis. The separation is achieved with an electric field in one direction and a pH gradient in the other direction. (a) For analysis, two gels obtained with different samples are superimposed and pseudo-colored, showing differences in protein composition of the two samples. (b) A program outlines regions of interest on a detail of the gel. Source: http://www.cgen.com/.

relations such as reading order. Region annotations are encapsulated within a generic annotation tag. Via the mechanism of XML name spaces, this provides users with the flexibility of defining their own types of annotations to supplement the types already defined: writer, material, script, and content. WandaXML has a simple syntax to format filters. New types of filter inputs and outputs may be defined via the use of name spaces. WandaXML also defines some application specific filter formats, and a virtual ink encoding format. We foresee that the simplicity and extensibility of the framework will encourage its wide spreading within the forensic community and in the document analysis community at large.

## Acknowledgements

The work described in this document is the result of a joint collaboration of forensic experts from the Bundeskriminalamt in Wiesbaden, Germany (BKA) and the Landeskriminalamt in Berlin, Germany (LKA) as well as researchers and developers from the Fraunhofer Institute for Production Systems and Design Technologies, Germany (FhG-IPK), the international Unipen Foundation, the Rijksuniversiteit Groningen, The Netherlands (RUG), the Nijmegen Institute for Cognition and Information, The Netherlands (NICI), Clopinet Consulting, USA and the Giesler-Software Entwicklung, Germany (SWE).
The present version of this document was prepared by Katrin Franke, Isabelle Guyon, and Lambert Schomaker. Although many persons have made direct or indirect contributions, special thanks go to those who were actively involved, particularly:
Axel Kerkhoff and Werner Kuckuck (BKA), Gerhard Grube (LKA), Altug Metin, Tomas Kühn, Martin Penk, and Steffen Rose (FhG-IPK), Johan Everts and Geertje Zwarts (RUG), Louis Vuurpijl and Merijn van Erp (NICI), Stefan Giesler (SWE).

## Appendices - WANDA XML Reference manual

We present the detailed specifications of Wanda XML in the following sections. We use the following conventions for element requisites: [+] at least one; [?] zero or one; [*] any number including zero.

# A WANDOC DTD

The wandoc XML regroups the Wanda XML tags devoted to document annotation. A wandoc document consists of one or several pages, each of which is represented by an image (allowed formats include gif, tif and jpg). Within each page, annotations may include:

- Textual annotations, such as information about the page content, writer, pen and writing surface used, type of script and language used.
- Filter declarations corresponding to operations applied to the image that can either be re-computed on-the-fly, or played back using a cached result.
- Regions, which are user-defined polygons or rectangles delineating a particular region of interest on the image. Regions may be recursively defined and are themselves annotated.

In order to accomodate user-defined filters and annotations, generic tags <input/> <output/>, and <annotation/> are introduced. Elements enclosed withing these tags will be application specific and defined in separate DTDs. See, for instance, the following annotation DTDs: writer , script , material , content . The tag <comment/> however is a generic annotation. For particular inputs and outputs, see the NICI measurement filter DTD: nicifeat . The tags <features/> and <wanda_link/> enclose generic types of outputs. In several data blocks (<page/>, <region/>, <annotation/>, <filter/>, <input/>, and <output/>, links can be specified to point to a file the content of which replaces the element <wanda_link/>.

### wandoc DTD Tree

```
wandoc
 |_(filters?,
 |   |_(filter+)
 |       |_(inputs+,
 |       |   |_(input+,
 |       |   |   |_(#PCDATA)
 |       |   |
 |       |   |__wanda_link*) ...
 |       |
 |       |__module+,
 |       |   |_(#PCDATA)
 |       |
 |       |__outputs+,
 |       |   |_(output+,
 |       |   |   |_(features?)
 |       |   |       |_(feature+)
 |       |   |           |_(#PCDATA)
 |       |   |
 |       |   |
 |       |   |
 |       |   |__wanda_link*) ...
 |       |
 |       |__wanda_link*) ...
 |
 |
 |__annotations?,
 |   |_(annotation+)
 |       |_(comment?,
 |       |   |_(#PCDATA)
 |       |
 |       |__wanda_link*) ...
 |
 |
 |__pages?,
```

```
|   |_(page+)
|       |_(filters?, ...
|       |__annotations?, ...
|       |__regions?,
|       |   |_(region+)
|       |       |_(filters?, ...
|       |       |__annotations?, ...
|       |       |__points+,
|       |       |   |_(point+)
|       |       |       |_EMPTY
|       |       |
|       |       |
|       |       |__regions?, ...
|       |       |__wanda_link*) ...
|       |
|       |
|       |__wanda_link*, ...
|       |__meta?) ...
|
|
|__wanda_link*,
|   |_EMPTY
|
|__meta?)
    |_EMPTY
```

### wandoc XML example

```
<?xml version="1.0" encoding="UTF-8"?>
<!--Sample XML file generated by XMLSPY v5 rel. 3 U
                                    (\url{http://www.xmlspy.com})-->
<!DOCTYPE wandoc SYSTEM "wandoc_.dtd">
<wandoc
  id="20032004_0001"
  label="WANDA test and development sample"
  xmlns="\url{http://pentel.ipk.fhg.de/wandaXML/wandoc/wandoc.dtd}">
  <pages number_of="1">
    <page id="20032004_0001_copy51" label="frontpage" next="">
      <filters number_of="1">
        <filter type="import" label="ibisScan" >
          <inputs>
            <input type="stream" number="1" xmlns="../scan.dtd">
              <scan/>
            </input>
          </inputs>
```

```
    <module type="extern" exec="ibis.exe">
      <meta version="3.51"/>
    </module>
    <outputs>
      <output type="file">
        <wanda_link
          href="docServer://wanda/database/.../0001.tif"/>
      </output>
    </outputs>
  </filter>
</filters>
<annotations number_of="3">
  <annotation type="content" xmlns="../content.dtd">
    <whole_document
      type="envelope"
      intent="personal"/>
  </annotation>
  <annotation type="writer" xmlns="../writer.dtd">
    <writer id="2015">
      <person>
        <name firstname="Altug" surname="Metin"/>
        <gender type="male"/>
        <born year="1978"/>
      </person>
      <properties handedness="left" skill="ok"/>
      <education country="France" level="high"/>
      <language native="French"/>
      <meta
        author="SIC Writer"
        email="sic@here.we.are"
        institution="Fraunhofer
        IPK" version="1.0"
        created="2003-03-17" />
    </writer>
  </annotation>
  <annotation type="material" xmlns="../material.dtd">
    <material>
      <paper
        type="writing"
        size="less_A6"
        material="woodfree"
        weight="less_60"
        product="IBM_copy_paper"
        absorbency="low"/>
      <pad type="paper" surface="even" hardness="soft"/>
    </material>
  </annotation>
```

```
</annotations>
<regions number_of="1">
  <region id="20032004_0001_copy51_0001_213746432"
      label="Hello...happy with it." next="">
     <points>
       <point x="0" y="0"/>
       <point x="10" y="0"/>
       <point x="0" y="10"/>
       <point x="10" y="10"/>
     </points>
     <annotations number_of="3">
       <annotation type="material" xmlns="../material.dtd">
         <material>
           <pen type="fountain_pen"
                product="waterman_serenite">
             <tip
               type="metal_nib"
               material="gold"
               diameter="medium"
               flexibility="high"/>
             <ink
               viscosity="low"
               transparence="opaque"
               color="gray"
               product="waterman"/>
           </pen>
         </material>
       </annotation>
       <annotation type="script" xmlns="../script.dtd">
          <script type="latin" language="english">
            <style
              major="cursive"
              connection="arced"
              caps="lower_upper"
              consistency="high"
              stroke_quality="disturbed"
              embellishment="simplified"
              stroke_quality_causes="ink_aging"
              inter-word_connectivity="low"
              intra-word_connectivity="high"
              relative_writing_speed="faster"/>
            </script>
         </annotation>
         <annotation type="content" xmsl="../content.dtd">
            <content>
              <text_block
                type="addressee_address_block"
```

```
                    length="paragraph" >
                    <properties
                      tone="neutral"
                      grammar="ok"
                      spelling="bad"/>
                    <verbatim>
                      Alfred Joe
                      223 daoun the rode
                      The Bled
                      USA
                    </verbatim>
                  </text_block>
                </content>
              </annotation>
            </annotations>
            <filters number_of="1">
              <filter type="feature_extract" label="nicifeat">
                <inputs>
                  <input type="stream" number="1"
                         xmlns="../nicifeat.dtd">
                    <nicifeat/>
                  </input>
                </inputs>
                <module type="client" exec="nicifeatmeasurement"/>
                <outputs>
                  <output type="stream">
                    <features number_of="4">
                    <!-- We do not need to precise the
                         name space because the feature tag
                         is a generic wandoc tag -->
                      <feature name="first" type="decimal"
                               unit="pixel" value="1.0"/>
                      <feature name="second"  type="decimal"
                               unit="pixel" value="2.0"/>
                      <feature name="third"  type="decimal"
                               unit="pixel" value="3.0"/>
                      <feature name="fourth"  type="decimal"
                               unit="pixel" value="4.0"/>
                    </features>
                  </output>
                </outputs>
              </filter>
            </filters>
          </region>
        </regions>
      </page>
    </pages>
```

```
</wandoc>
```

### wandoc DTD File

```
<!-- ============================================================

    wandoc  (Wanda DOCUMENT  Modeling Language)

    author: Katrin Fanke, Lambert Schomaker, Isabelle Guyon
    institution: Fraunhofer IPK, Rijksuniversiteit Groningen, Clopinet
    version: 2.0
    created: 2002-04-11-00-00
    modified: 2003-05-08-00-00

 ============================================================ -->
<!ENTITY % anno_types
    " * | document | writer | script | material | content ">
<!ENTITY % filter_types
    " * | import | processing | feature_extract ">
<!ENTITY % input_types " * | stream | file | var ">
<!ENTITY % module_types " * | client | server | extern ">
<!ENTITY % output_types " * | stream | file | var ">
<!ENTITY % feature_types " * | boolean | decimal ">

<!-- ======================================================== -->
<!ELEMENT wandoc
    (filters?, annotations?, pages? , wanda_link*, meta?)>
<!ATTLIST wandoc
    id CDATA #REQUIRED
    label CDATA "Enter label !"
    xmlns CDATA #IMPLIED
>

<!-- ======================================================== -->
<!ELEMENT pages (page+)>
<!ATTLIST pages
    number_of CDATA #IMPLIED
>
<!ELEMENT page
    (filters?, annotations?, regions?, wanda_link*, meta?)>
<!ATTLIST page
    id CDATA #REQUIRED
    label CDATA #IMPLIED
    next CDATA #IMPLIED
```

```
>

<!-- ======================================================== -->
<!ELEMENT regions (region+)>
<!ATTLIST regions
    number_of CDATA #IMPLIED
>
<!ELEMENT region
    (filters?, annotations?,  points+, regions?, wanda_link*)>
<!ATTLIST region
    id CDATA  #REQUIRED
    label CDATA #IMPLIED
    next CDATA #IMPLIED
>

<!-- ======================================================== -->
<!ELEMENT annotations (annotation+)>
<!ATTLIST annotations
    number_of CDATA #IMPLIED
>
<!ELEMENT annotation (comment?, wanda_link*)>
<!ATTLIST annotation
    type (%anno_types;) #IMPLIED
    xmlns CDATA #IMPLIED
>
<!ELEMENT comment (#PCDATA)>

<!-- ======================================================== -->
<!ELEMENT filters (filter+)>
<!ATTLIST filters
    number_of CDATA #IMPLIED
>
<!ELEMENT filter (inputs+, module+, outputs+, wanda_link*)>
<!ATTLIST filter
    type (%filter_types;) #IMPLIED
    label CDATA #IMPLIED
    xmlns CDATA #IMPLIED
>
<!ELEMENT inputs (input+, wanda_link*)>
<!ATTLIST inputs
    number_of CDATA #IMPLIED
>
<!ELEMENT input (#PCDATA)>
<!--type and  the n-th input for the module -->
<!ATTLIST input
    type (%input_types;) #REQUIRED
    number CDATA #IMPLIED
```

```
    label CDATA #IMPLIED
    xmlns CDATA #IMPLIED
>
<!ELEMENT module (#PCDATA)>
<!ATTLIST module
    type (%module_types;) #REQUIRED
    exec CDATA #REQUIRED
>
<!ELEMENT outputs (output+, wanda_link*)>
<!ATTLIST outputs
    number_of CDATA #IMPLIED
>
<!ELEMENT output (features?)>
<!ATTLIST output
    type (%output_types;) #REQUIRED
    number CDATA #IMPLIED
    label CDATA #IMPLIED
    xmlns CDATA #IMPLIED
>

<!-- ======================================================== -->
<!ELEMENT features (feature+)>
<!ATTLIST features
    number_of CDATA #IMPLIED
>
<!ELEMENT feature (#PCDATA)>
<!ATTLIST feature
     name CDATA #REQUIRED
     type (%feature_types;) #IMPLIED
     unit CDATA #IMPLIED
     value CDATA #REQUIRED
     number CDATA #IMPLIED
 >
 <!-- ======================================================== -->
<!ELEMENT points (point+)>
<!ELEMENT point EMPTY>
<!ATTLIST point
    x CDATA #REQUIRED
    y CDATA #REQUIRED
>

<!-- ======================================================== -->
<!ELEMENT wanda_link EMPTY>

<!ATTLIST wanda_link
    href    CDATA    #REQUIRED
>
```

```
<!-- This is a proposition of definition of wanda_link as xlink:
<!ATTLIST wanda_link
    xmlns:xlink   CDATA    #FIXED   "\url{http://www.w3.org/1999/xlink}"
    xlink:type    (simple) #FIXED   "simple"
    xlink:href    CDATA    #REQUIRED
    xlink:show    (embed)  #FIXED   "embed"
    xlink:actuate (onLoad) #FIXED   "onLoad"
>
We made a fix choice for the xlink behavior that will carry over
for the entire wandoc document. If attribute show is set to
embed then the xlink replaces the element link. When attribute
actuate is set to onLoad, replacement is immediate
(similarly to well-known element <html:img src=... >).
For other options see:
\url{http://www.zvon.org/xxl/xlink/OutputExamples/frame_xlinksimple_html.html}
or \url{http://www.w3.org/TR/REC-xml-names/} -->

<!-- ======================================================= -->
<!ELEMENT meta EMPTY>
<!ATTLIST meta
    author CDATA #IMPLIED
    email CDATA #IMPLIED
    institution CDATA #IMPLIED
    version CDATA #IMPLIED
    created CDATA #IMPLIED
    modified CDATA #IMPLIED
>
```

### wandoc XML tag reference

**<annotation**
type = "*—document—writer—script—material—..." – Type of annotation.
xmlns = `CDATA` – XML name space (e.g. a DTD name or url) >
**<comment/>** – Free text comment. [?]
**<wanda_link/>** – Element that will be replaced by the file it points to. [*]
**</annotation>**
Description: Tag enclosing a user-defined annotation (an application specific DTD is required). See, for instance, the following annotation DTDs: writer , script , material , content .
Parent Element(s): **<annotations** /> – Container of annotations.

**<annotations**
number_of = `CDATA` – Number of annotations.
>
**<annotation/>** – Tag enclosing a user-defined annotation. [+]
**</annotations>**

Description: Container of annotations.
Parent Element(s):
<**page** /> – Document page containing a single image and annotations.
<**region** /> – Region of interest defined by a polygon or rectangle and annotated.
<**wandoc** /> – Root element of a Wanda document annotation.

<**comment**
</**comment**>
Description: Free text comment.
Parent Element(s): <**annotation** /> – Tag enclosing a user-defined annotation.

<**feature**
name = `CDATA` – Feature name.
number = `CDATA` – Number of the feature.
type = "*—boolean—decimal" – Feature value type.
unit = `CDATA` – Feature unit.
value = `CDATA` – Feature value.
</**feature**>
Description: Tag to record a statistic about the document, e.g. average ascender length.
Parent Element(s): <**features** /> – Container of features.

<**features**
number_of = `CDATA` – Number of features.
>
<**feature/**> – Tag to record a statistic about the document. [+]
</**features**>
Description: Container of features.
Parent Element(s): <**output** /> – Tag enclosing user-defined outputs.

<**filter**
label = `CDATA` – Label of filter (any text).
type = "*—import—processing—feature_extract" – Filter type.
xmlns = `CDATA` – XML name space (e.g. a DTD name or url) >
<**inputs/**> – Container of inputs. [+]
<**module/**> – Executable module. [+]
<**outputs/**> – Container of outputs. [+]
<**wanda_link/**> – Element that will be replaced by the file it points to. [*]
</**filter**>
Description: Generic filter definition (a function/plugin call).
Parent Element(s): <**filters** /> – Container of fiters.

<**filters**
number_of = `CDATA` – Number of filters.
>
<**filter/**> – Generic filter definition. [+]
</**filters**>
Description: Container of filters.
Parent Element(s):
<**page** /> – Document page containing a single image and annotations.

<**region** /> – Region of interest defined by a polygon or rectangle and annotated.
<**wandoc** /> – Root element of a Wanda document annotation.

<**input**
label = CDATA – Label identifying the input for search purpose.
number = CDATA – Optional input order number.
type = "*—stream—file—var" – Input type.
xmlns = CDATA – XML name space (e.g. a DTD name or url)
</**input**>
Description: Tag enclosing user-defined inputs (an application specific DTD is required). See for instance the NICI measurement filter DTD: nicifeat .
Parent Element(s): <**inputs** /> – Container of inputs.

<**inputs**
number_of = CDATA – Number of inputs.
>
<**input/**> – Tag enclosing user-defined inputs. [+]
<**wanda_link/**> – Element that will be replaced by the file it points to. [*]
</**inputs**>
Description: Container of inputs.
Parent Element(s): <**filter** /> – Generic filter definition.

<**meta**
author = CDATA – Author of the annotations of this part of the document.
created = CDATA – Date created in the YYYY-MM-DD-mm-ss format.
email = CDATA – Contact email.
institution = CDATA – Author's affiliation.
modified = CDATA – Date last modified in the YYYY-MM-DD-mm-ss format.
version = CDATA – DTD version number.
/>
Description: Information about how the document annotations were generated.
Parent Element(s):
<**page** /> – Document page containing a single image and annotations.
<**wandoc** /> – Root element of a Wanda document annotation.

<**module**
exec = CDATA – The function call itself.
type = "*—client—server—extern" – Type of module.
</**module**>
Description: Executable module (function/plugin call).
Parent Element(s): <**filter** /> – Generic filter definition.

<**output**
label = CDATA – Label identifying the output for search purpose.
number = CDATA – Optional output order number.
type = "*—stream—file—var" – Type of output.
xmlns = CDATA – XML name space (e.g. a DTD name or url) >
<**features/**> – Container of features. [?]
</**output**>

Description: Tag enclosing user-defined outputs (an application specific DTD is required). See for instance the NICI measurement filter DTD: nicifeat .
Parent Element(s): <**outputs** /> – Container of outputs.

<**outputs**
number_of = `CDATA` –
>
<**output/**> – Tag enclosing user-defined outputs. [+]
<**wanda_link/**> – Element that will be replaced by the file it points to. [*]
</**outputs**>
Description: Container of outputs.
Parent Element(s): <**filter** /> – Generic filter definition.

<**page**
id = `CDATA` – Unique page id number.
label = `CDATA` – A user-defined string used to retrieve easily the page by searching (e.g., part of the page content).
next = `CDATA` – Next page identifier. >
<**filters/**> – Container of fiters. [?]
<**annotations/**> – Container of annotations. [?]
<**regions/**> – Container of regions. [?]
<**wanda_link/**> – Element that will be replaced by the file it points to. [*]
<**meta/**> – Information about how the document annotations were generated. [?]
</**page**>
Description: Document page containing a single image and annotations.
Parent Element(s): <**pages** /> – Container of pages.

<**pages**
number_of = `CDATA` – Number of pages.
>
<**page/**> – Document page containing a single image and annotations. [+]
</**pages**>
Description: Container of pages.
Parent Element(s): <**wandoc** /> – Root element of a Wanda document annotation.

<**point**
x = `CDATA` – x coordinate.
y = `CDATA` – y coordinate.
/>
Description: Cartesian coordinate point, in pixel units on the image.
Parent Element(s): <**points** /> – Container of points.

<**points** >
<**point/**> – Cartesian coordinate point. [+]
</**points**>
Description: Container of points.
Parent Element(s): <**region** /> – Region of interest defined by a polygon or rectangle and annotated.

<**region**
id = `CDATA` – Unique region of interest id.
label = `CDATA` – A user-defined string used to retrieve easily the region by searching (e.g., part of the region content).
next = `CDATA` – Next region identifier. >
<**filters**/> – Container of fiters. [?]
<**annotations**/> – Container of annotations. [?]
<**points**/> – Container of points. [+]
<**regions**/> – Container of regions. [?]
<**wanda_link**/> – Element that will be replaced by the file it points to. [*]
</**region**>
Description: Region of interest.
Parent Element(s): <**regions** /> – Container of regions.

<**regions**
number_of = `CDATA` – Number of regions.
>
<**region**/> – Region of interest defined by a polygon or rectangle and annotated. [+]
</**regions**>
Description: Container of regions.
Parent Element(s):
<**page** /> – Document page containing a single image and annotations.
<**region** /> – Region of interest defined by a polygon or rectangle and annotated.

<**wanda_link**
href = `CDATA` – Link reference (URL or file name).

/>
Description: Element that will be replaced by the file it points to.
Parent Element(s):
<**annotation** /> – Tag enclosing a user-defined annotation.
<**filter** /> – Generic filter definition.
<**inputs** /> – Container of inputs.
<**outputs** /> – Container of outputs.
<**page** /> – Document page containing a single image and annotations.
<**region** /> – Region of interest defined by a polygon or rectangle and annotated.
<**wandoc** /> – Root element of a Wanda document annotation.

<**wandoc**
id = `CDATA` – Wanda document unique id.
label = `CDATA` – A user-defined string used to retrieve easily the document by searching (e.g., part of the document content).
xmlns = `CDATA` – XML name space (e.g. a DTD name or url) >
<**filters**/> – Container of fiters. [?]
<**annotations**/> – Container of annotations. [?]
<**pages**/> – Container of pages. [?]
<**wanda_link**/> – Element that will be replaced by the file it points to. [*]
<**meta**/> – Information about how the document annotations were generated. [?]

</**wandoc**>
Description: Root element of a Wanda document annotation.
Parent Element(s): None

# B WRITER DTD

The writer description is part of the Wanda XML language. It records information on the writer that may be relevant to the writer identification process from sample handwriting. It excludes other personal information. Therefore, only year of birth, education, gender, and language information are considered.

### writer DTD Tree

```
writer
 |_(person,
 |   |_(name,
 |   |   |_EMPTY
 |   |
 |   |__gender,
 |   |   |_EMPTY
 |   |
 |   |__born)
 |       |_EMPTY
 |
 |
 |__properties?,
 |   |_EMPTY
 |
 |__education?,
 |   |_EMPTY
 |
 |__language?,
 |   |_EMPTY
 |
 |__meta?)
     |_EMPTY
```

### writer XML example

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE writer SYSTEM "writer_.dtd">
```

```
<writer id="2015">
    <person>
        <name firstname="Altug" surname="Metin"/>
        <gender type="male"/>
        <born year="1978"/>
    </person>
    <properties handedness="left" skill="ok"/>
    <education country="France" level="high"/>
    <language native="French"/>
    <meta author="SIC Writer"
          institution="Fraunhofer IPK"
          version="1.0"
          created="2003-03-17"
          modified=""/>
</writer>
```

### writer DTD File

```
<!-- edited with XMLSPY v5 rel. 3 U (\url{http://www.xmlspy.com})
                            by Isabelle Guyon (Clopinet) -->
<!-- ============================================================

    writer (Wanda WRITER Modeling Language)

    author: Altug Metin, edited by Isabelle Guyon
    institution: Fraunhofer IPK, Clopinet
    version: 3.0
    created: 2002-02-18-00-00
    modified: 2003-05-08-00-00

 ============================================================ -->
<!ENTITY % gender_types " * | female | male ">
<!ENTITY % handedness_types " * | left  | right ">
<!ENTITY % skill_types " * | bad  | ok | good ">
<!ENTITY % education_level_types
        " * | elementary | medium | high ">
<!ENTITY % education_country_types " * |
albania |
alberta |
ashmore |
algeria |
afghanistan |
argentina |
anguilla |
```

armenia |
azerbaijan |
alaska |
alabama |
anguilla |
andorra |
angola |
antigua_barbuda |
arkansas |
american_samoa |
australia |
austria |
aruba |
antarctica |
arizona |
bahrain |
barbados |
british_columbia |
burundi |
belgium |
bahamas |
bangladesh |
belize |
brazil |
bermuda |
bosnia_hercegovina |
bolivia |
burma |
botswana |
bhutan |
bulgaria |
bouvet_island |
belarus |
byelorussia |
brunei |
california |
cambodia |
china |
chad |
sri_lanka |
congo |
croatia |
cayman_islands |
colombia |
chile |
cameroon |
canada |

```
colorado |
comoros |
costa_rica |
czech_republic|
slovakia |
connecticut |
cuba |
cape_verde |
cook_islands |
central_african_republic |
cyprus |
district_of_columbia |
delaware |
denmark |
benin |
dominica |
dominican_republic |
eritrea |
ecuador |
equatorial_guinea |
easttimor |
england |
estonia |
elsalvador |
ethiopia |
faroe_islands |
french_guiana |
finland |
fiji |
falkland_islands |
florida |
micronesia |
french_polynesia |
france |
djibouti |
georgia |
kiribati |
grenada |
germany |
ghana |
gibraltar |
greenland |
gambia |
gabon   |
guadeloupe |
greece |
georgia |
```

```
guatemala |
guam |
guinea |
guyana |
hawaii |
hongkong |
honduras |
haiti |
hungary |
iowa |
iceland |
idaho |
ireland |
india |
illinois |
indiana |
indonesia |
iraq |
iran |
israel |
italy |
cote_d_ivoire |
japan |
johnston_atoll |
jamaica |
jan_mayen |
jordan |
kenya |
kyrgyzstan |
kirghiz |
north_korea |
south_korea |
kansas |
kuwait |
kentucky |
kazakhstan |
louisiana |
liberia |
lebanon |
liechtenstein |
lithuania |
lesotho |
laos |
luxembourg |
latvia |
libya |
massachusetts |
```

```
manitoba |
monaco |
maryland |
maine |
mauritius |
madagascar |
macao |
michigan |
montserrat |
oman |
mali |
malta |
minnesota |
missouri |
mongolia |
martinique |
morocco |
mississippi |
montana |
mauritania |
moldova |
moldavia |
malawi |
mexico |
malaysia |
mozambique |
netherlands_antilles |
nebraska |
north_carolina |
north_dakota |
netherlands |
newfoundland_labrador |
niger |
new_hampshire |
northern_ireland |
new_jersey |
new_brunswick |
new_caledonia |
northern_mariana_islands |
new_mexico |
vanuatu |
norway |
nepal |
nicaragua |
nigeria |
nova_scotia |
northwest_territories |
```

```
nauru |
nunavut |
nevada |
northern_mariana_islands |
norfolk_island |
new_york_state |
new_zealand |
ohio |
oklahoma |
ontario |
oregon |
mayotte |
pennsylvania |
pitcairn_island |
peru |
paracel_islands |
guinea_bissau |
philippines |
prince_edward_island |
pakistan |
poland |
panama |
portugal |
papua_new_guinea |
puerto_rico |
portuguese_timor |
palau |
paraguay |
qatar |
quebec_province |
reunion |
zimbabwe |
rhode_island |
romania |
russia |
rwanda |
ryukyu_islands_southern |
south_africa |
svalbard |
south_carolina |
south_dakota |
seychelles |
sao_tome_and_principe |
senegal |
spanish_north_africa |
singapore |
sudan |
```

```
sikkim |
sierra_leone |
san_marino |
saskatchewan |
somalia |
spain |
swaziland |
surinam |
western_sahara |
scotland |
saudi_arabia |
swan_islands |
sweden |
namibia |
syria |
switzerland |
tajikistan |
togo |
thailand |
tunisia |
turkmenistan |
tokelau |
tennessee |
tonga |
trinidad_and_tobago |
united_arab_emirates |
turkey |
tuvalu |
texas |
tanzania |
egypt |
united_states_caribbean_islands |
uganda |
unitedkingdom |
ukraine |
united_states_pacific_islands  |
soviet_union |
united_states |
utah |
burkina_faso |
uruguay |
uzbekistan |
virginia |
british_virgin_islands |
vaticancity |
venezuela |
united_states_virgin_islands |
```

```
vietnam |
vermont |
washington_state |
wallis_and_futuna |
wisconsin |
wake_island |
wales |
samoa |
west_virginia |
wyoming |
christmas_island |
cocos_keeling_islands |
maldives |
saint_kitts-nevis |
marshall_islands |
midway_islands |
niue |
saint_kitts-nevis-anguilla |
saint_helena |
saint_lucia |
saint_pierre_and_miquelon |
saint_vincent_and_the_grenadines |
macedonia |
slovakia |
spratly_island |
czech_republic |
south_georgia |
slovenia |
canada |
united_kingdom |
yemen |
yukon_territory |
yemen |
yugoslavia |
zambia ">
<!-- source: \url{http://www.loc.gov/marc/countries/} -->
<!ENTITY % native_language_types " * |
afrikaans |
akkadian |
albanian |
arabic |
armenian |
assyrian |
aymara |
malay |
bangala |
basque |
```

```
bavarian |
belorusian |
bengali |
berber_tamazight |
breton |
bulgarian |
burmese |
cambodian |
cantonese |
catalan |
cherokee_tsalagi |
croatian |
czech |
dakota |
danish |
dauphinois |
dutch |
egyptian |
english |
estonian |
finnish |
flemish |
french |
frisian |
fukienese |
gaelic |
galician |
georgian |
german |
greek |
guarani |
gujarati |
hakka |
halaka |
hausa |
hawaiian |
hebrew |
hundustani_hindi |
hungarian |
icelandic |
indonesian |
italian |
japanese |
javanese |
kamilaroi |
korean |
kurdish |
```

ladino |
latvian |
lithuanian |
lojban |
luganda |
macedonian |
malayalam |
maltese |
mandarin_chinese |
manx |
maori |
mohawk |
mon |
mongolian |
myanmar |
navajo |
nepalese |
norwegian |
occitan |
ojibwe |
oneida |
papiamentu |
persian |
pidgin |
pitcairn |
polish |
portuguese |
punjabi |
quechua |
rasta |
romanian |
romansch |
romany |
russian |
sardinian |
scots |
serbian |
sinhalese |
slovak |
slovenian |
spanish |
sranan |
sudanese |
swabian |
swahili |
swedish |
tagalog |

```
tamil |
telugu |
thai |
tlingit |
turkish |
ukranian |
viennese |
vietnamese |
welsh |
wu |
yiddish ">
<!-- source: \url{http://www.tesarta.com/www/resources/languages.html} -->
<!-- ======================================================= -->
<!ELEMENT writer
    (person, properties?, education?, language?, meta?)>
<!-- root tag -->
<!ATTLIST writer
    id CDATA #REQUIRED
>
<!-- ======================================================= -->
<!-- every writer is a person ... -->
<!ELEMENT person (name, gender, born)>
<!-- person's name -->
<!ELEMENT name EMPTY>
<!ATTLIST name
    firstname CDATA #IMPLIED
    surname CDATA #IMPLIED
>
<!ELEMENT gender EMPTY>
<!-- person's gender -->
<!ATTLIST gender
    type (%gender_types;) "*"
>
<!ELEMENT born EMPTY>
<!-- person's birth -->
<!ATTLIST born
    year CDATA "*"
>
<!-- ======================================================= -->
<!ELEMENT properties EMPTY>
<!-- writer properties -->
<!ATTLIST properties
    handedness (%handedness_types;) "*"
    skill (%skill_types;) "*"
>
<!-- ======================================================= -->
<!ELEMENT education EMPTY>
```

```
<!-- writer's (handwrting) education -->
<!ATTLIST education
    country (%education_country_types;) "*"
    level (%education_level_types;) "*"
>
<!-- ======================================================= -->
<!ELEMENT language EMPTY>
<!ATTLIST language
    native (%native_language_types;) "*"
>
<!-- ======================================================= -->
<!ELEMENT meta EMPTY>
<!-- author of the writer info and location of writer
                                    info's entering/storage -->
<!ATTLIST meta
    author CDATA #REQUIRED
    institution CDATA #REQUIRED
    version CDATA #REQUIRED
    created CDATA #REQUIRED
    modified CDATA #REQUIRED
>
```

**writer XML tag reference**

<**born**
year = CDATA – Year born in four digit format.

/>
Description: Year born.
Parent Element(s): <**person** /> – Civil state information.

<**education**
country = "*—albania—alberta—ashmore—algeria—..." – Country of education.
level = "*—elementary—medium—high" – Level of education.
/>
Description: Education received that may affect handwriting.
Parent Element(s): <**writer** /> – Writer root element.

<**gender**
type = "*—female—male" – Male/female.
/>
Description: Male/female.
Parent Element(s): <**person** /> – Civil state information.

<**language**
native = "*—afrikaans—akkadian—albanian—arabic—..." – Language of educa-

tion.
/>
Description: Language(s).
Parent Element(s): <**writer** /> – Writer root element.

<**meta**
author = CDATA – Person who collected the data.
created = CDATA – Date of creation.
institution = CDATA – Place of creation.
modified = CDATA – Date last modified.
version = CDATA – Version number.
/>
Description: How this information was collected.
Parent Element(s): <**writer** /> – Writer root element.

<**name**
firstname = CDATA – First name or given name.
surname = CDATA – Surname, family name, last name.
/>
Description: Person name.
Parent Element(s): <**person** /> – Civil state information.

<**person** >
<**name/**> – Writer name.
<**gender/**> – Male/female.
<**born/**> – Year born.
</**person**>
Description: Civil state information.
Parent Element(s): <**writer** /> – Writer root element.

<**properties**
handedness = "*—left—right" – Hand usually used for writing.
skill = "*—bad—ok—good" – Handwriting skill.
/>
Description: Writing ability properties.
Parent Element(s): <**writer** /> – Writer root element.

<**writer**
id = CDATA – Writer id number.
>
<**person/**> – Civil state information.
<**properties/**> – Writing ability properties. [?]
<**education/**> – Education received that may affect handwriting. [?]
<**language/**> – Language(s). [?]
<**meta/**> – How this information was collected. [?]
</**writer**>
Description: Writer root element.
Parent Element(s): None

# C MATERIAL DTD

The material description is part of the Wanda XML language to annotate forensic documents. Writing utensil, ink and surface material are recorded.

### material DTD Tree

```
material
 |_(paper?,
 |   |_EMPTY
 |
 |__pen?,
 |   |_(tip?,
 |   |   |_EMPTY
 |   |
 |   |__ink)
 |       |_EMPTY
 |
 |
 |__pad?,
 |   |_EMPTY
 |
 |__meta?)
     |_EMPTY
```

### material XML example

```
<?xml version="1.0" encoding="UTF-8"?>
<!--Sample XML file generated by XMLSPY v5 rel. 3 U
    (\url{http://www.xmlspy.com})-->
<!DOCTYPE material SYSTEM "material_.dtd">
<material>
    <paper  type="writing"
            size="less_A6"
            material="woodfree"
            weight="less_60"
            product="IBM_copy_paper"
            absorbency="low"/>
    <pen type="fountain_pen" product="waterman_serenite">
        <tip    type="metal_nib"
                material="gold"
                diameter="medium"
```

```
                flexibility="high"/>
        <ink    viscosity="low"
                transparence="opaque"
                color="gray"
                product="waterman"/>
    </pen>
    <pad type="paper" surface="even" hardness="soft"/>
</material>
```

### material DTD File

```
<!-- ============================================================

    material (Wanda MATERIAL Modeling Language)
    author: Katrin Fanke, Gerhard Grube, Lambert Schomaker,
            Louis Vuurpijl, Isabelle Guyon
    institution: Fraunhofer IPK, Landeskriminalamt Berlin,
            Rijksuniversiteit Groningen, Nijmegen University,
            Clopinet
    version: 3.0
    created: 2002-03-18-00-00
    modified: 2003-05-08-00-00

 ============================================================ -->
<!ENTITY % pen_type_types
    " * | pencil | mechanical_pencil | propelling_pencil |
    ball_point_pen | roller_ball_pen | gel_ink_pen |
    porous_point_pen | fine_line_pen | fountain_pen |
    steel_nib_pen | calligraphic_pen | quill">
<!ENTITY % tip_type_types
    " * | pencil | ball_pen | metal_nib | plastic_nib |
    flexible_plastic_nib | stiff_perforated_plastic_point |
    fiber_tip | felt_tip | ceramic_tip ">
<!ENTITY % tip_material_types
  " * | metal | plastic | ceramic | fibers | felt | gold | steel ">
<!ENTITY % tip_flexibility_types " * | low | normal | high ">
<!ENTITY % ink_viscosity_types " * | low | medium | high ">
<!ENTITY % ink_transparence_types
    " * | opaque | semi_opaque | transparent ">
<!ENTITY % ink_color_types
    " * | white | black | gray | purple | blue | green |
    yellow | orange | red | others ">
<!ENTITY % paper_type_types
    " * | writing | newspaper | concept | check | ledger |
```

```
    flint_glazed | board | filter | embossed | creped | grained
    | mottled | cast_coated | absorbent | bleached ">
<!ENTITY % paper_size_types
    " * | less_A6 | A6 | A5 | US_letter | A4 | larger_than_A4 ">
<!ENTITY % paper_material_types " * | woodfree | rice ">
<!ENTITY % paper_weight_types
    " * | less_60 | 60 | 70 | 80 | 90 | 100 | 110 | 120 |
    more_than_120 ">
<!ENTITY % paper_absorbency_types " * | low | normal | height ">
<!ENTITY % pad_type_types
    " * | electronic_tablet | paper | glass | wood | metal |
    hard_pad | soft_pad ">
<!ENTITY % pad_hardness_types " * | soft | normal | hard ">
<!ENTITY % pad_surface_types " * | even | structured ">

<!-- ======================================================== -->

<!ELEMENT material (paper?, pen?, pad?, meta?)>

<!-- Paper specific information ============================ -->

<!ELEMENT paper EMPTY>

<!ATTLIST paper
    type (%paper_type_types;) #IMPLIED
    size (%paper_size_types;) #REQUIRED
    material (%paper_material_types;) #IMPLIED
    weight (%paper_weight_types;) #IMPLIED
    product CDATA #IMPLIED
    absorbency (%paper_absorbency_types;) #REQUIRED
>

<!-- Pen and Ink specific information ====================== -->

<!ELEMENT pen (tip?, ink)>
<!ATTLIST pen
    type (%pen_type_types;) #REQUIRED
    product CDATA #IMPLIED
>

<!ELEMENT tip EMPTY>
<!ATTLIST tip
    type (%tip_type_types;) #IMPLIED
    material (%tip_material_types;) #IMPLIED
    diameter CDATA #IMPLIED
    flexibility (%tip_flexibility_types;) #REQUIRED
>
```

```
<!ELEMENT ink EMPTY>
<!ATTLIST ink
    viscosity (%ink_viscosity_types;) #REQUIRED
    transparence (%ink_transparence_types;) #IMPLIED
    color (%ink_color_types;) #REQUIRED
    product CDATA #IMPLIED
>

<!-- Pad specific information ============================== -->

<!ELEMENT pad EMPTY>
<!ATTLIST pad
    type (%pad_type_types;) #IMPLIED
    hardness (%pad_hardness_types;) #REQUIRED
    surface (%pad_surface_types;) #REQUIRED
>

<!-- ======================================================= -->

<!ELEMENT meta EMPTY>

<!ATTLIST meta
    author CDATA #IMPLIED
    email CDATA #IMPLIED
    institution CDATA #IMPLIED
    version CDATA #IMPLIED
    created CDATA #IMPLIED
    modified CDATA #IMPLIED
>
```

### material XML tag reference

**<ink**
color = "*—white—black—gray—purple—..." – Ink color.
product = CDATA – Ink product/manufacturer name.
transparence = "*—opaque—semi_opaque—transparent" – Ink transparence.
viscosity = "*—low—medium—high" – Ink viscosity.
**/>**
Description: Type and color of ink.
Parent Element(s): <**pen** /> – Writing pen.

**<material >**
<**paper/**> – Writing paper. [?]
<**pen/**> – Writing pen. [?]

<**pad/**> – Writing support. [?]
<**meta/**> – How this information was collected. [?]
</**material**>
Description: Material used for writing.
Parent Element(s): None

<**meta**
author = `CDATA` – Person who collected the data.
created = `CDATA` – Date of creation.
email = `CDATA` – Contact email.
institution = `CDATA` – Place of creation.
modified = `CDATA` – Date last modified.
version = `CDATA` – DTD version number.
/>
Description: How this information was collected.
Parent Element(s): <**material** /> – Material used for writing.

<**pad**
hardness = "*—soft—normal—hard" – Pad surface hardness.
surface = "*—even—structured" – Pad surface irregularities.
type = "*—electronic_tablet—paper—glass—wood—..." – Pad type.
/>
Description: Writing support.
Parent Element(s): <**material** /> – Material used for writing.

<**paper**
absorbency = "*—low—normal—height" – Paper absorbemcy.
material = "*—woodfree—rice" – Paper material.
product = `CDATA` – Paper product/manufacturer name.
size = "*—less_A6—A6—A5—US_letter—..." – Paper size.
type = "*—writing—newspaper—concept—check—..." – Paper type.
weight = "*—less_60—60—70—80—..." – Paper weight.
/>
Description: Writing paper.
Parent Element(s): <**material** /> – Material used for writing.

<**pen**
product = `CDATA` – Pen product/manufacturer name.
type = "*—pencil—mechanical_pencil—propelling_pencil—ball_point_pen—..." – Pen type. >
<**tip/**> – Pen tip. [?]
<**ink/**> – Type and color of ink.
</**pen**>
Description: Writing pen.
Parent Element(s): <**material** /> – Material used for writing.

<**tip**
diameter = `CDATA` – Tip diameter.
flexibility = "*—low—normal—high" – Tip flexibility.

material = “*—metal—plastic—ceramic—fibers—...” – Tip material.
type = “*—pencil—ball_pen—metal_nib—plastic_nib—...” – Tip type.
/>
Description: Pen tip.
Parent Element(s): <**pen** /> – Writing pen.

# D SCRIPT DTD

The script description is part of the Wanda XML language to annotate forensic documents. Language, type of script and style are recorded.

### script DTD Tree

```
script
 |_(style,
 |   |_EMPTY
 |
 |__meta?)
     |_EMPTY
```

### script XML example

```
<?xml version="1.0" encoding="UTF-8"?>
<!--Sample XML file generated by XMLSPY v5 rel. 3 U
                            (\url{http://www.xmlspy.com})-->
<!DOCTYPE script SYSTEM "script_.dtd">
<script type="latin" language="afrikaans">
    <style
        major="cursive"
        connection="arced"
        caps="lower_upper"
        consistency="high"
        stroke_quality="disturbed"
        embellishment="simplified"
        stroke_quality_causes="ink_aging"
        inter-word_connectivity="low"
        intra-word_connectivity="high"
        relative_writing_speed="faster"/>
</script>
```

**script DTD File**

```
<!-- ============================================================

    script  (Wanda SCRIPT Modeling Language)

    author: Katrin Fanke, Lambert Schomaker,
            Gerhard Grube, Louis Vuurpijl, Isabelle Guyon
    institution: Fraunhofer IPK, Rijksuniversiteit Groningen,
        Landeskriminalamt Berlin, Nijmegen University, Clopinet
    version: 3.0
    created: 2002-03-18-00-00
    modified: 2003-05-08-00-00

 ============================================================ -->
<!-- source: Writing Systems of the World:
             Alphabets, Syllabaries, Pictograms
             by Nakanishi Akira and Akira Nakanishi -->
<!ENTITY % script_type_types
" * |
latin |
greek |
russian_cyrillic |
georgian |
armenian |
other_european |
hebrew |
arabic |
farsi |
urdu |
maldivian |
other_west_asiatic |
devanagari |
gurmukhi |
gujarati |
oriya |
bengali |
tamil |
telugu |
kannanda |
malayalam |
sinhalese |
other_indian |
burmese |
khmer |
thai |
```

```
lao |
other_southeast_asia |
chinese_hanzi |
tibetan |
mongolian |
korean_hangul |
japanese_kanji |
japanese_kana |
other_east_asia |
amharic |
other_african |
south_american |
other_american ">

<!-- source: \url{http://www.tesarta.com/www/resources/languages.html} -->
<!-- NOTE: The script language is different
           from the native language of the writer. -->
<!ENTITY % language_types
"* |
afrikaans |
akkadian |
albanian |
arabic |
armenian |
assyrian |
aymara |
malay |
bangala |
basque |
bavarian |
belorusian |
bengali |
berber_tamazight |
breton |
bulgarian |
burmese |
cambodian |
cantonese |
catalan |
cherokee_tsalagi |
croatian |
czech |
dakota |
danish |
dauphinois |
dutch |
egyptian |
```

```
english |
estonian |
finnish |
flemish |
french |
frisian |
fukienese |
gaelic |
galician |
georgian |
german |
greek |
guarani |
gujarati |
hakka |
halaka |
hausa |
hawaiian |
hebrew |
hundustani_hindi |
hungarian |
icelandic |
indonesian |
italian |
japanese |
javanese |
kamilaroi |
korean |
kurdish |
ladino |
latvian |
lithuanian |
lojban |
luganda |
macedonian |
malayalam |
maltese |
mandarin_chinese |
manx |
maori |
mohawk |
mon |
mongolian |
myanmar |
navajo |
nepalese |
norwegian |
```

```
occitan |
ojibwe |
oneida |
papiamentu |
persian |
pidgin |
pitcairn |
polish |
portuguese |
punjabi |
quechua |
rasta |
romanian |
romansch |
romany |
russian |
sardinian |
scots |
serbian |
sinhalese |
slovak |
slovenian |
spanish |
sranan |
sudanese |
swabian |
swahili |
swedish |
tagalog |
tamil |
telugu |
thai |
tlingit |
turkish |
ukranian |
viennese |
vietnamese |
welsh |
wu |
yiddish ">


<!ENTITY % major_style_types
    " * | cursive | mixed | handprint | blockprint ">
<!ENTITY % cap_type_types
    " * | lower_upper | all_caps | lowercase | uppercase ">
<!ENTITY % connection_types
```

```
    " * | arced | garland | angular | straight | indetermined ">
<!ENTITY % consistency_types
    " * | high | normal | low ">
<!ENTITY % embellishment_types
    " * | simplified | normal | enriched ">
<!ENTITY % stroke_quality_types
    " * | smooth | disturbed | highly_disturbed ">
<!ENTITY % stroke_quality_causes_types
    " * | ink_aging | pen_defect | paper_absobency |
    pad_hardness | pad_surface | writer_state | pen_grip ">
<!ENTITY % inter-word_connectivity_types
    " * | high | normal | low ">
<!ENTITY % intra-word_connectivity_types
    " * | high | normal | low ">
<!ENTITY % relative_writing_speed_types
    " * | faster | normal | slower ">

<!--Script specific information ============================ -->
<!ELEMENT script (style, meta?)>
<!ATTLIST script
    type (%script_type_types;) #REQUIRED
    language (%language_types;) #REQUIRED
>

<!ELEMENT style EMPTY>
<!ATTLIST style
    major (%major_style_types;) #REQUIRED
    connection (%connection_types;) #IMPLIED
    caps (%cap_type_types;) #IMPLIED
    consistency (%consistency_types;) #IMPLIED
    stroke_quality (%stroke_quality_types;) #IMPLIED
    embellishment (%embellishment_types;) #IMPLIED
    stroke_quality_causes (%stroke_quality_causes_types;) #IMPLIED
    inter-word_connectivity (%inter-word_connectivity_types;) #IMPLIED
    intra-word_connectivity (%intra-word_connectivity_types;) #IMPLIED
    relative_writing_speed (%relative_writing_speed_types;) #IMPLIED
>
<!-- ======================================================= -->
<!ELEMENT meta EMPTY>
<!ATTLIST meta
    author CDATA #IMPLIED
    email CDATA #IMPLIED
    institution CDATA #IMPLIED
    version CDATA #IMPLIED
    created CDATA #IMPLIED
    modified CDATA #IMPLIED
>
```

**script XML tag reference**

<**meta**
author = `CDATA` – Person who collected the data.
created = `CDATA` – Date of creation.
email = `CDATA` – Contact electronic mail.
institution = `CDATA` – Place of creation.
modified = `CDATA` – Date last modified.
version = `CDATA` – DTD version number.
/>
Description: How this information was collected.
Parent Element(s): <**script** /> – Script root element.

<**script**
language = "*—afrikaans—akkadian—albanian—arabic—..." – Language used.
type = "*—latin—greek—russian_cyrillic—georgian—..." – Script type used. >
<**style/**> – Writing style.
<**meta/**> – How this information was collected. [?]
</**script**>
Description: Script root element.
Parent Element(s): None

<**style**
caps = "*—lower_upper—all_caps—lowercase—uppercase" – Capitalization: all_caps differs from uppercase in that first letters may have a larger size.
connection = "*—arced—garland—angular—straight—..." – Type of connectors.
consistency = "*—high—normal—low" – Consistency of style.
embellishment = "*—simplified—normal—enriched" – Degree of embellishement.
inter-word_connectivity = "*—high—normal—low" – Connectivity between words.
intra-word_connectivity = "*—high—normal—low" – Connectivity within word.
major = "*—cursive—mixed—handprint—blockprint" – Major style.
relative_writing_speed = "*—faster—normal—slower" – Writing speed compared to average.
stroke_quality = "*—smooth—disturbed—highly_disturbed" – Regularity of the baseline.
stroke_quality_causes = "*—ink_aging—pen_defect—paper_absobency—pad_hardness—..." – Possible causes of stroke defects.
/>
Description: Writing style.
Parent Element(s): <**script** /> – Script root element.

# E CONTENT DTD

The part of Wanda ML concerns the document content annotations.

### content DTD Tree

```
content
 |_(document?,
 |   |_EMPTY
 |
 |__text_block?,
 |   |_(properties?,
 |   |   |_EMPTY
 |   |
 |   |__verbatim?)
 |       |_(#PCDATA)
 |
 |
 |__misc_block?,
 |   |_EMPTY
 |
 |__meta?)
     |_EMPTY
```

### content XML example

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE content SYSTEM "content_.dtd">
<content>
    <document
        type="envelope"
        intent="personal"/>
    <text_block
        type="addressee_address_block"
        length="paragraph"
    >
        <properties tone="neutral" grammar="ok" spelling="bad"/>
        <verbatim>
            Alfred Joe
            223 daoun the rode
            The Bled
            USA
```

```
        </verbatim>
    </text_block>
    <meta
        author="Isabelle Guyon"
        email="isabelle@clopinet.com"
        institution="Clopinet"
        version="1.0"
        created="2003-04-18-00-00"
        modified="2003-04-24-00-00"/>
</content>
```

### content DTD File

```
<!-- ============================================================

    content  (Wanda CONTENT  Modeling Language)

    author: Isabelle Guyon, Katrin Fanke, Lambert Schomaker
    institution: Clopinet,  Fraunhofer IPK,
                 Rijksuniversiteit Groningen
    version: 1.0
    created: 2002-03-25-00-00
    modified: 2003-05-08-00-00

============================================================ -->
<!ENTITY % document_type_types
            " * | writing_product | note | letter | check |
            greeting_card | envelope | postcard | bank_transfer
            | post_sticker | hotel_registration |
            foreigner_script | contract | last_will |
            work_permit | tax_form | registration_form |
            excerpt ">
<!ENTITY % intent_types
            " * | message | bomb_threat | threat | extortion |
            blackmail | pornography | child_porn |
            drugs_related | financial_fraud | terrorism |
            racism | personal | public ">
<!ENTITY % text_block_type_types
            " * | whole_form | addressee_address_block |
            sender_address_block | personal_name |
            geographical_name | zipcode | legal_amount |
            currency_amount | telephone_number | digits |
            abbreviation | signature | paraph | plain_text ">
<!ENTITY % misc_block_type_types
```

```
            " * | postal_stamp | ink_stamp | barcode |
            printed_text | fingerprint | stain | drawing |
            scribble | graffiti ">
<!ENTITY % text_length_types
            " * | character | word | few_words | line |
            paragraph | page | pages ">
<!ENTITY % tone_types
            " * | kind | neutral | threatening | angry |
            sad | happy ">
<!ENTITY % grammar_types " * | bad | ok | good ">
<!ENTITY % spelling_types " * | bad | ok | good ">

<!ELEMENT content (document?, text_block?, misc_block?, meta?)>

<!ELEMENT meta EMPTY>
<!ATTLIST meta
    author CDATA #REQUIRED
    email CDATA #REQUIRED
    institution CDATA #REQUIRED
    version CDATA #REQUIRED
    created CDATA #REQUIRED
    modified CDATA #REQUIRED
>

<!ELEMENT document EMPTY>
<!ATTLIST document
    type (%document_type_types;) #REQUIRED
    intent (%intent_types;) #REQUIRED
>

<!ELEMENT text_block (properties?, verbatim?)>
<!ATTLIST text_block
    type (%text_block_type_types;) #REQUIRED
    length (%text_length_types;) #REQUIRED
>

<!ELEMENT misc_block EMPTY>
<!ATTLIST misc_block
    type (%misc_block_type_types;) #REQUIRED
>

<!ELEMENT properties EMPTY>
<!ATTLIST properties
    tone (%tone_types;) #REQUIRED
    grammar (%grammar_types;) #REQUIRED
    spelling (%spelling_types;) #REQUIRED
>
```

```
<!ELEMENT verbatim (#PCDATA)>
```

### content XML tag reference

<**content** >
<**document**/> – Information on the whole document. [?]
<**text_block**/> – Block of text. [?]
<**misc_block**/> – Miscellaneous non-text block. [?]
<**meta**/> – Information on who created these data annotations. [?]
</**content**>
Description: Root element.
Parent Element(s): None

<**document**
intent = "*—message—bomb_threat—threat—extortion—..." – Document intent.
type = "*—writing_product—note—letter—check—..." – Document type.
/>
Description: Information on the whole document.
Parent Element(s): <**content** /> – Root element.

<**meta**
author = `CDATA` – Person who authored the annotations.
created = `CDATA` – Date created.
email = `CDATA` – Contact email.
institution = `CDATA` – Author's affiliation.
modified = `CDATA` – Date modified.
version = `CDATA` – DTD version.
/>
Description: Information on who created these data annotations.
Parent Element(s): <**content** /> – Root element.

<**misc_block**
type = "*—postal_stamp—ink_stamp—barcode—printed_text—..." – Non-text block type.
/>
Description: Miscellaneous non-text block.
Parent Element(s): <**content** /> – Root element.

<**properties**
grammar = "*—bad—ok—good" – Grammar quality.
spelling = "*—bad—ok—good" – Spelling quality.
tone = "*—kind—neutral—threatening—angry—..." – Tone employed.
/>
Description: Text block content properties.
Parent Element(s): <**text_block** /> – Block of text.

<**text_block**
length = “*—character—word—few_words—line—...” – Text block length.
type = “*—whole_form—addressee_address_block—sender_address_block—personal_name—...” – Text block type. >
<**properties/**> – Text block content properties. [?]
<**verbatim/**> – Verbatim transcription. [?]
</**text_block**>
Description: Block of text.
Parent Element(s): <**content** /> – Root element.

<**verbatim**
</**verbatim**>
Description: Verbatim transcription. Use the ISO 10646 standard to encode special characters. You can refer to characters from in the encoded repertoire by using &#dddd; (decimal character code) or &#xHHHH; (hexadecimal character code, in uppercase).
Parent Element(s): <**text_block** /> – Block of text.

# F SCAN DTD

Wanda XML for scanned image annotation.

### scan DTD Tree

```
scan
 |_(scanner?,
 |   |_EMPTY
 |
 |__image+,
 |   |_(lcms)
 |       |_EMPTY
 |
 |
 |__meta?)
     |_EMPTY
```

### scan XML example

```
<?xml version="1.0"?>
<!DOCTYPE scan SYSTEM "scan_.dtd">
<scan name="A4 Farbe quer">
    <scanner driver="HP ScanJet"
```

```
            feeder="flatbed"
            pages="frontside"
            format="A4"
            orientation="landscape"/>
    <image   name="Archivbild"
            filename="FSA%05u.jpg"
            camera="frontside"
            colors="rgb24"
            compress="jpeg"
            quality="30"
            xresolution="100"
            yresolution="100">
         <lcms incolor="ct_lcms_rgb"
         inprofile="\url{file://c}$/sg_pDir/ibis/sRGB/ibs_9403_300.icc"/>
    </image>
    <image name="Detailbild"
           filename="FSD%05u.tif"
           camera="frontside"
           colors="rgb24"
           compress="none"
           xresolution="200" yresolution="200">
        <lcms incolor="ct_lcms_rgb"
        inprofile="\url{file://c}$/sg_pDir/ibis/sRGB/ibs_9403_300.icc"/>
    </image>
    <meta
         author="Stefan Giesler"
         email="st.giesler@t-online.de"
         institution="Giesler Software Entwicklung"
         version="1.0"
         created="2003-03-10-00-00"
         modified="2003-04-11-09-00"/>
</scan>
```

### scan DTD File

```
<!-- ============================================================

    scan(profile) (Wanda SCAN Modeling Language)

    author: Stefan Giesler <st.giesler@t-online.de>
    edited by: Isabelle Guyon <isabelle@clopinet.com>
    Grouped attlist args in one attlist.
    Changed some default values.
    Changed <info> to <meta> and put it under root element.
```

```
    Added email in <meta>.
    institution: Giesler Software Entwicklung
    version: 1.0
    created: 2003-03-10-00-00
    modified: 2003-04-11-09-00

 ========================================================== -->
<!ENTITY % orient "portait | landscape" >
<!ENTITY % side "frontside | backside | bothsides" >
<!ENTITY % feed "flatbed | feeder | other" >
<!ENTITY % size "A6 | A5 | A4 | A3" >
<!ENTITY % camera "frontside | backside" >
<!ENTITY % colors "bitonal | gray | rgb24 | rgb36" >
<!ENTITY % rotate "none | left | centre | right" >
<!ENTITY % compress "none | faxg3 | faxg4 | jpeg" >
<!ENTITY % xsection "none | left | right | user" >
<!ENTITY % ysection "none | top | bottom | user" >
<!-- ========================================================= -->
<!ELEMENT scan (scanner?, image+, meta?) > <!-- root tag -->

 <!-- name of the scan profile -->
<!ATTLIST scan name CDATA #REQUIRED >

<!-- ========================================================= -->
<!ELEMENT scanner EMPTY >

<!ATTLIST scanner
    driver CDATA "driverName"
    feeder (%feed;) "flatbed"
    pages (%side;) "frontside"
    orientation (%orient;) "portait"
    format (%size;) "A4" >


<!-- ========================================================= -->
<!ELEMENT image (lcms) >

<!ATTLIST image
    name CDATA #REQUIRED
    filename CDATA #REQUIRED
    camera (%camera;) "frontside"
    colors (%colors;) "rgb24"
    rotate (%rotate;) "none"
    compress (%compress;) "none"
    quality CDATA "100"
    xresolution CDATA "100"
    yresolution CDATA "100"
```

```
    xpixels CDATA "0"
    ypixels CDATA "0"
    xsection (%xsection;) "none"
    xsecsize CDATA "0"
    xseccoord CDATA "0"
    ysection (%ysection;) "none"
    ysecsize CDATA "0"
    yseccoord CDATA "0">

<!-- ======================================================== -->
<!-- little color management system -->
<!ELEMENT lcms EMPTY>

<!ATTLIST lcms
    inProfile CDATA #REQUIRED
    inColor CDATA #REQUIRED
    outProfile CDATA #REQUIRED
    outColor CDATA #REQUIRED>

<!-- ======================================================== -->
<!ELEMENT meta EMPTY>
<!ATTLIST meta
    author CDATA "anonymous"
    email CDATA "anonymous@somewhere.abc"
    institution CDATA "anywhere"
    version CDATA "00.00"
    created CDATA "YYYY-MM-DD-hh-mm-ss"
    modified CDATA "YYYY-MM-DD-hh-mm-ss">
```

### scan XML tag reference

**<image**

camera = “frontside—backside” – Camera type.
colors = “bitonal—gray—rgb24—rgb36” – Color space.
compress = “none—faxg3—faxg4—jpeg” – Compression type.
filename = `CDATA` – Image file name.
name = `CDATA` – Image name.
quality = `CDATA` – JPEG compression quality.
rotate = “none—left—centre—right” – Rotation type.
xpixels = `CDATA` – Image width in pixels.
xresolution = `CDATA` – Image x resolution in dpi.
xseccoord = `CDATA` – Upper-left coordinate of scanned section in x.
xsecsize = `CDATA` – Width of scanned section in x.
xsection = “none—left—right—user” – Type of section in x. Meaning?
ypixels = `CDATA` – Image height in pixels.

yresolution = `CDATA` – Image y resolution in dpi.
yseccoord = `CDATA` – Upper-left coordinate of scanned section in y.
ysecsize = `CDATA` – Width of scanned section in y.
ysection = "none—top—bottom—user" – Type of section in y. Meaning? >
<**lcms/**> – Little color management system.
</**image**>
Description: Image definition block.
Parent Element(s): <**scan** /> – Root element.

<**lcms**
inColor = `CDATA` – What?
inProfile = `CDATA` – What?
outColor = `CDATA` – What?
outProfile = `CDATA` – What?
/>
Description: Little color management system.
Parent Element(s): <**image** /> – Image definition block.

<**meta**
author = `CDATA` – Author of the scan profile.
created = `CDATA` – Date created in YYYY-MM-DD-hh-mm-ss format.
email = `CDATA` – Email of the author.
institution = `CDATA` – Author's affiliation.
modified = `CDATA` – Date created in YYYY-MM-DD-hh-mm-ss format.
version = `CDATA` – Version of the scan profile DTD.
/>
Description: Information about who generated this document.
Parent Element(s): <**scan** /> – Root element.

<**scan**
name = `CDATA` – Name of the scan profile.
>
<**scanner/**> – Scanner information. [?]
<**image/**> – Image definition block. [+]
<**meta/**> – Information about who generated this document. [?]
</**scan**>
Description: Root element.
Parent Element(s): None

<**scanner**
driver = `CDATA` – Scanner driver name.
feeder = "flatbed—feeder—other" – Document feeder.
format = "A6—A5—A4—A3" – Document format.
orientation = "portait—landscape" – Document orientation.
pages = "frontside—backside—bothsides" – Document pages.
/>
Description: Scanner information.
Parent Element(s): <**scan** /> – Root element.

# G PROPER DTD

This part of Wanda XML describes preprocessing operations.

### proper DTD Tree

```
proper
 |_(roi*,
 |   |_(vertex*)
 |       |_EMPTY
 |
 |
 |__properParam+,
 |   |_(background,
 |   |   |_(none?,
 |   |   |   |_ANY
 |   |   |
 |   |   |__homogenous?,
 |   |   |   |_ANY
 |   |   |
 |   |   |__textured?,
 |   |   |   |_(low?,
 |   |   |   |   |_EMPTY
 |   |   |   |
 |   |   |   |__high?,
 |   |   |   |   |_EMPTY
 |   |   |   |
 |   |   |   |__peaks?,
 |   |   |   |   |_EMPTY
 |   |   |   |
 |   |   |   |__amount?)
 |   |   |       |_EMPTY
 |   |   |
 |   |   |
 |   |   |__userdefined?,
 |   |   |   |_ANY
 |   |   |
 |   |   |__formdropout?,
 |   |   |   |_(image,
 |   |   |   |   |_ANY
 |   |   |   |
 |   |   |   |__method,
 |   |   |   |   |_ANY
 |   |   |   |
 |   |   |   |__offset)
```

```
|   |   |       |_ANY
|   |   |
|   |   |
|   |   |__colordropout?,
|   |   |   |_ANY
|   |   |
|   |   |__colorconvert?)
|   |       |_(channelcomp?)
|   |           |_(image, ...
|   |           |__method, ...
|   |           |__offset) ...
|   |
|   |
|   |
|   |__foreground?,
|   |   |_(reconst?,
|   |   |   |_ANY
|   |   |
|   |   |__noise?,
|   |   |   |_EMPTY
|   |   |
|   |   |__line?,
|   |   |   |_(direction?,
|   |   |   |   |_(rows?,
|   |   |   |   |   |_ANY
|   |   |   |   |
|   |   |   |   |__columns?,
|   |   |   |   |   |_ANY
|   |   |   |   |
|   |   |   |   |__angle?)
|   |   |   |       |_EMPTY
|   |   |   |
|   |   |   |
|   |   |   |__((average? |
|   |   |   |     |_ANY
|   |   |   |
|   |   |   |____trace? |
|   |   |   |     |_EMPTY
|   |   |   |
|   |   |   |____morpho? |
|   |   |   |     |_EMPTY
|   |   |   |
|   |   |   |____gabor?)?))
|   |   |   |         |_ANY
|   |   |   |
|   |   |   |
|   |   |__stamp?,
```

```
|   |   |   |_((segment |
|   |   |   |    |_ANY
|   |   |   |
|   |   |   |___contrast),
|   |   |   |    |_ANY
|   |   |   |
|   |   |   |__noise) ...
|   |   |
|   |   |__framein?,
|   |   |   |_(share? |
|   |   |   |   |_EMPTY
|   |   |   |
|   |   |   |__rect?)
|   |   |       |_(extension |
|   |   |       |   |_EMPTY
|   |   |       |
|   |   |       |__frame)
|   |   |           |_EMPTY
|   |   |
|   |   |
|   |   |
|   |   |__border?)
|   |       |_(rect?) ...
|   |
|   |
|   |__inout?,
|   |   |_(pre? |
|   |   |   |_(scale?)
|   |   |       |_(resolution?,
|   |   |       |   |_EMPTY
|   |   |       |
|   |   |       |__(bilinear? |
|   |   |       |    |_ANY
|   |   |       |
|   |   |       |___repeat?))
|   |   |            |_ANY
|   |   |
|   |   |
|   |   |
|   |   |__post?)
|   |       |_(scale? | ...
|   |       |__colorout?)
|   |           |_(color? |
|   |           |   |_EMPTY
|   |           |
|   |           |__pseudo?)
|   |               |_ANY
```

```
|   |
|   |
|   |
|   |
|   |__desc?)
|       |_ANY
|
|
|__meta?)
    |_EMPTY
```

### proper XML example

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE content SYSTEM "proper_.dtd">
    <roi id="0">
        <vertex x="20" y="50"/>
        <vertex x="50" y="50"/>
        <vertex x="20" y="100"/>
    </roi>
    <roi id="1">
        <vertex x="306" y="97"/>
        <vertex x="377" y="44"/>
        <vertex x="454" y="100"/>
        <vertex x="477" y="61"/>
        <vertex x="474" y="147"/>
        <vertex x="454" y="108"/>
        <vertex x="376" y="143"/>
    </roi>
    <properParam id="0">
        <background>
            <homogenous/>
        </background>
        <foreground>
            <line>
                <direction>
                    <rows/>
                </direction>
                <morpho ze_horizontal="0.5" ze_vertical="20"/>
            </line>
        </foreground>
    </properParam>
    <properParam id="1">
        <background>
            <homogenous/>
```

```
        </background>
        <foreground>
            <line>
                <direction>
                    <rows/>
                </direction>
                <morpho ze_horizontal="22" ze_vertical="200"/>
            </line>
        </foreground>
    </properParam>
    <meta author="Martin Peng" institution="fraunhofer IPK"
          version="2.0" created="2003-03-17" modified=""/>
</proper>
```

### proper DTD File

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- ============================================================

    proper (SIC PROPER Modeling Language)

    author: Martin Peng, Katrin Franke
    institution: Fraunhofer IPK
    version: 2.0
    created: 2002-11-18-00-00
    modified: 2003-02-23-00-00

 ============================================================ -->
<!-- edited with XMLSPY v5 rel. 3 U
    (\url{http://www.xmlspy.com}) by Isabelle Guyon (Clopinet) -->
<!ELEMENT proper (roi*, properParam+, meta?)>
<!ELEMENT roi (vertex*)>
<!ATTLIST roi
    id CDATA #IMPLIED
>
<!ELEMENT properParam (background, foreground?, inout?, desc?)>
<!ATTLIST properParam
    id CDATA #IMPLIED
>
<!ELEMENT meta EMPTY>
<!ATTLIST meta
    author CDATA #IMPLIED
    institution CDATA #IMPLIED
    version CDATA #IMPLIED
```

```
    created CDATA #IMPLIED
    modified CDATA #IMPLIED
>
<!ELEMENT vertex EMPTY>
<!ATTLIST vertex
    x CDATA #REQUIRED
    y CDATA #REQUIRED
>
<!ELEMENT background
    (none?, homogenous?, textured?, userdefined?,
    formdropout?, colordropout?, colorconvert?)>
<!ELEMENT foreground
    (reconst?, noise?, line?, stamp?, framein?, border?)>
<!ELEMENT inout (pre? | post?)>
<!ELEMENT desc ANY>
<!ELEMENT none ANY>
<!ELEMENT homogenous ANY>
<!ELEMENT textured (low?, high?, peaks?, amount?)>
<!ELEMENT userdefined ANY>
<!ELEMENT formdropout (image, method, offset)>
<!ELEMENT colordropout ANY>
<!ELEMENT colorconvert (channelcomp?)>
<!ELEMENT reconst ANY>
<!ELEMENT noise EMPTY>
<!ATTLIST noise
    size CDATA #IMPLIED
>
<!ELEMENT line
    (direction?, ((average? | trace? | morpho? | gabor?)?))>
<!ELEMENT stamp ((segment | contrast), noise)>
<!ELEMENT framein (share? | rect?)>
<!ELEMENT border (rect?)>
<!ELEMENT pre (scale?)>
<!ELEMENT post (scale? | colorout?)>
<!ELEMENT low EMPTY>
<!ATTLIST low
    value CDATA #IMPLIED
>
<!ELEMENT high EMPTY>
<!ATTLIST high
    value CDATA #IMPLIED
>
<!ELEMENT peaks EMPTY>
<!ATTLIST peaks
    value CDATA #IMPLIED
>
<!ELEMENT amount EMPTY>
```

```
<!ATTLIST amount
    value CDATA #IMPLIED
>
<!ELEMENT image ANY>
<!ELEMENT method ANY>
<!ELEMENT offset ANY>
<!ELEMENT channelcomp (image, method, offset)>
<!ELEMENT direction (rows?, columns?, angle?)>
<!ELEMENT average ANY>
<!ELEMENT trace EMPTY>
<!ATTLIST trace
    grayvariance CDATA #IMPLIED
    derivation CDATA #IMPLIED
    limit CDATA #IMPLIED
    threshold CDATA #IMPLIED
>
<!ELEMENT morpho EMPTY>
<!ATTLIST morpho
    se_horizontal CDATA #IMPLIED
    se_vertical CDATA #IMPLIED
    ze_horizontal CDATA #IMPLIED
    ze_vertical CDATA #IMPLIED
>
<!ELEMENT gabor ANY>
<!ELEMENT segment ANY>
<!ELEMENT contrast ANY>
<!ELEMENT share EMPTY>
<!ATTLIST share
    amount CDATA #IMPLIED
>
<!ELEMENT rect (extension | frame)>
<!ELEMENT scale (resolution?, (bilinear? | repeat?))>
<!ELEMENT colorout (color? | pseudo?)>
<!ELEMENT rows ANY>
<!ELEMENT columns ANY>
<!ELEMENT angle EMPTY>
<!ATTLIST angle
    degree CDATA #REQUIRED
>
<!ELEMENT extension EMPTY>
<!ATTLIST extension
    offset_x CDATA #IMPLIED
    offset_y CDATA #IMPLIED
    extension_x CDATA #IMPLIED
    extension_y CDATA #IMPLIED
>
<!ELEMENT frame EMPTY>
```

```
<!ATTLIST frame
    left CDATA #IMPLIED
    top CDATA #IMPLIED
    bottom CDATA #IMPLIED
    right CDATA #IMPLIED
>
<!ELEMENT resolution EMPTY>
<!ATTLIST resolution
    dpi (75 | 150 | 300 | 600 | 1200) #IMPLIED
>
<!ELEMENT bilinear ANY>
<!ELEMENT repeat ANY>
<!ELEMENT color EMPTY>
<!ATTLIST color
    model (rgb | cmyk | gray | bw) #IMPLIED
>
<!ELEMENT pseudo ANY>
```

**proper XML tag reference**

<**amount**
value = CDATA – Amount of texture value.

/>
Description: Amount of texture.
Parent Element(s): <**textured** /> – Textured background removal.

<**angle**
degree = CDATA – Maximum slant angle. By default: 20 degrees.

/>
Description: Maximum slant angle in degrees.
Parent Element(s): <**direction** /> – Direction of lines.

<**average**
/>
Description: Average line cleaning method. Use this procedure for the cleaning of lined and squared paper.
Parent Element(s): <**line** /> – Line removal (e.g. handwriting guide lines).

<**background** >
<**none**/> – No background removal. [?]
<**homogenous**/> – Homogeneous background removal. [?]
<**textured**/> – Textured background removal. [?]
<**userdefined**/> – User defined background removal. [?]
<**formdropout**/> – Background form removal (under construction). [?]

<**colordropout/**> – Colors to be ignored (under construction). [?]
<**colorconvert/**> – Color conversion (under construction). [?]
</**background**>
Description: Background removal.
Parent Element(s): <**properParam** /> – Parameters of the proper preprocessing.

<**bilinear**
/>
Description: What is that?
Parent Element(s): <**scale** /> – Image scaling.

<**border** >
<**rect/**> – Inner rectangle of a frame. [?]
</**border**>
Description: Border removal.
Parent Element(s): <**foreground** /> – Container of foreground filtering parameters.

<**channelcomp** >
<**image/**> – Image place holder (under construction).
<**method/**> – Method place holder (under construction).
<**offset/**> – Offset place holder (under construction).
</**channelcomp**>
Description: Channel comparison (under construction).
Parent Element(s): <**colorconvert** /> – Color conversion (under construction).

<**color**
model = "rgb—cmyk—gray—bw" – Model type.
/>
Description: Color model.
Parent Element(s): <**colorout** /> – Output image color coding.

<**colorconvert** >
<**channelcomp/**> – Channel comparison (under construction). [?]
</**colorconvert**>
Description: Color conversion (under construction).
Parent Element(s): <**background** /> – Container of background filtering parameters.

<**colordropout**
/>
Description: Colors to be ignored (under construction).
Parent Element(s): <**background** /> – Container of background filtering parameters.

<**colorout** >
<**color/**> – Color model. [?]
<**pseudo/**> – Rendering in pseudo color. [?]
</**colorout**>
Description: Output image color coding.
Parent Element(s): <**post** /> – Image formating after preprocessing.

<**columns**
/>
Description: Lines are vertical.
Parent Element(s): <**direction** /> – Direction of lines.

<**contrast**
/>
Description: Stamp contrast.
Parent Element(s): <**stamp** /> – Removal of stamps.

<**desc**
/>
Description: Textual description?
Parent Element(s): <**properParam** /> – Parameters of the proper preprocessing.

<**direction** >
<**rows/**> – Lines are horizontal. [?]
<**columns/**> – Lines are vertical. [?]
<**angle/**> – Maximum slant angle. [?]
</**direction**>
Description: By choosing <rows/> and <columns/> the direction of the lines can be set. Lines with slant larger than specified by <angle/> compared to the horizontal or vertical are not cleaned.
Parent Element(s): <**line** /> – Line removal (e.g. handwriting guide lines).

<**extension**
extension_x = `CDATA` – ???
extension_y = `CDATA` – ???
offset_x = `CDATA` – ???
offset_y = `CDATA` – ???
/>
Description: Frame extension.
Parent Element(s): <**rect** /> – Inner rectangle of a frame.

<**foreground** >
<**reconst/**> – Reconstruct image segments after background removal. [?]
<**noise/**> – Noise filter. [?]
<**line/**> – Line removal (e.g. handwriting guide lines). [?]
<**stamp/**> – Removal of stamps. [?]
<**framein/**> – Foreground frame removal. [?]
<**border/**> – Border removal (under construction). [?]
</**foreground**>
Description: Container of foreground filtering parameters.
Parent Element(s): <**properParam** /> – Parameters of the proper preprocessing.

<**formdropout** >
<**image/**> – Image place holder (under construction).
<**method/**> – Method place holder (under construction).
<**offset/**> – Offset place holder (under construction).
</**formdropout**>

Description: Background form removal (under construction).
Parent Element(s): <**background** /> – Container of background filtering parameters.

<**frame**
bottom = `CDATA` – Bottom y coord?
left = `CDATA` – Left x coord?
right = `CDATA` – Right x coord?
top = `CDATA` – Top y coord?
/>
Description: Frame rectangle vertices.
Parent Element(s): <**rect** /> – Inner rectangle of a frame.

<**framein** >
<**share**/> – ??? [?]
<**rect**/> – Inner rectangle of a frame. [?]
</**framein**>
Description: Radical cleaning of the outer frame with a following reconstruction on the basis of the inner segments. Hereby the parameter Max indicates the size (percent) of the frame width. Furthermore, the coordinates of the inner rectangle might be explicit specified. FrameOut is under construction.
Parent Element(s): <**foreground** /> – Container of foreground filtering parameters.

<**gabor**
/>
Description: Gabor filter.
Parent Element(s): <**line** /> – Line removal (e.g. handwriting guide lines).

<**high**
value = `CDATA` – Value of high level texture?

/>
Description: Value of high level texture?
Parent Element(s): <**textured** /> – Textured background removal.

<**homogenous**
/>
Description: In case the input image is a greyscale picture (8 bit) without background pattern, e.g. a monochrome image (not necessarily white), the background will be deleted and the image becomes binary. If the image is already binary (black/white image) no image adjustments will take place.
Parent Element(s): <**background** /> – Container of background filtering parameters.

<**image**
/>
Description: Image place holder (under construction).
Parent Element(s):
<**channelcomp** /> – Channel comparison (under construction).
<**formdropout** /> – Background form removal (under construction).

**<inout >**
**<pre/>** – Image formating before preprocessing. [?]
**<post/>** – Image formating after preprocessing. [?]
**</inout>**
Description: Input and output formatting.
Parent Element(s): **<properParam />** – Parameters of the proper preprocessing.

**<line >**
**<direction/>** – Direction of lines. [?]
**<average/>** – Average line cleaning method. [?]
**<trace/>** – Trace line removal method. [?]
**<morpho/>** – Morphological operation. What do se and ze stand for? [?]
**<gabor/>** – Gabor filter (under construction). [?] [?]
**</line>**
Description: Line removal. Horizontal and/or vertical lines (e.g. handwriting guide lines) are deleted.
Parent Element(s): **<foreground />** – Container of foreground filtering parameters.

**<low**
value = `CDATA` – Value of low level texture?

**/>**
Description: Value of low level texture?
Parent Element(s): **<textured />** – Textured background removal.

**<meta**
author = `CDATA` – Author of the proper profile.
created = `CDATA` – Date created in YYYY-MM-DD-hh-mm-ss format.
institution = `CDATA` – Author's affiliation.
modified = `CDATA` – Date created in YYYY-MM-DD-hh-mm-ss format.
version = `CDATA` – Version of the proper profile DTD.
**/>**
Description: Information about who generated this document.
Parent Element(s): **<proper />** – Root element.

**<method**
**/>**
Description: Method place holder (under construction).
Parent Element(s):
**<channelcomp />** – Channel comparison (under construction).
**<formdropout />** – Background form removal (under construction).

**<morpho**
se_horizontal = `CDATA` – ???
se_vertical = `CDATA` – ???
ze_horizontal = `CDATA` – ???
ze_vertical = `CDATA` – ???
**/>**
Description: Morphological operator. This method has been developed for thin and medium

writing lines.
Parent Element(s): <**line** /> – Line removal (e.g. handwriting guide lines).

<**noise**
size = `CDATA` – Size of a noise segment.

/>
Description: Noise filter. The fine cleaning eliminates separate segments without reconstruction. By using the parameter Max Do you mean "size"? the segment size of the noise that has to be cleaned can be directed. For example Noise- Max = 1 deletes a single image pixel.
Parent Element(s):
<**foreground** /> – Container of foreground filtering parameters.
<**stamp** /> – Removal of stamps.

<**none**
/>
Description: In case of an 8 bit-binary input image no background removal or manipulation will be performed. However, is contains the image more than two image values a simple threshold binarization will be carried out.
Parent Element(s): <**background** /> – Container of background filtering parameters.

<**offset**
/>
Description: Offset place holder (under construction).
Parent Element(s):
<**channelcomp** /> – Channel comparison (under construction).
<**formdropout** /> – Background form removal (under construction).

<**peaks**
value = `CDATA` – Value of peak level texture?

/>
Description: Value of peak level texture?
Parent Element(s): <**textured** /> – Textured background removal.

<**post** >
<**scale/**> – Image scaling. [?]
<**colorout/**> – Output image color coding. [?]
</**post**>
Description: Image formating after preprocessing.
Parent Element(s): <**inout** /> – Input and output formatting.

<**pre** >
<**scale/**> – Image scaling. [?]
</**pre**>
Description: Image formating before preprocessing.
Parent Element(s): <**inout** /> – Input and output formatting.

<**proper** >

<**roi/**> – Region of interest. Replace by region? [*]
<**properParam/**> – Parameters of the proper preprocessing. [+]
<**meta/**> – Information about the XML document generation. [?]
</**proper**>
Description: Root element.
Parent Element(s): None

<**properParam**
id = CDATA – Proper parameters unique id.
>
<**background/**> – Container of background filtering parameters.
<**foreground/**> – Container of foreground filtering parameters. [?]
<**inout/**> – Input and output formatting. [?]
<**desc/**> – Textual description? [?]
</**properParam**>
Description: Parameters of the proper preprocessing.
Parent Element(s): <**proper** /> – Root element.

<**pseudo**
/>
Description: The remaining handwriting is converted to a pseudo-color image representing the relative ink intensity distribution along the writing trace.
Parent Element(s): <**colorout** /> – Output image color coding.

<**reconst**
/>
Description: An intense cleaning will be performed and only those image segments, which are most likely part of the writing product, will be (morphologically) reconstructed.
Parent Element(s): <**foreground** /> – Container of foreground filtering parameters.

<**rect** >
<**extension/**> – Frame extension. What is that?
<**frame/**> – Rectangle frame.
</**rect**>
Description: Inner rectangle of a frame.
Parent Element(s):
<**border** /> – Border removal (under construction).
<**framein** /> – Foreground frame removal.

<**repeat**
/>
Description: What is that?
Parent Element(s): <**scale** /> – Image scaling.

<**resolution**
dpi = "75—150—300—600—1200" – Image resolution in dots per inch.
/>
Description: Image resolution.
Parent Element(s): <**scale** /> – Image scaling.

<**roi**
id = CDATA – Unique ROI identifier.
>
<**vertex/**> – Vertices defining the ROI contour. What are x and y? [*]
</**roi**>
Description: Region of interest.
Parent Element(s): <**proper** /> – Root element.

<**rows**
/>
Description: Lines are horizontal.
Parent Element(s): <**direction** /> – Direction of lines.

<**scale** >
<**resolution/**> – Image resolution. [?]
<**bilinear/**> – What is that? [?]
<**repeat/**> – What is that? [?]
</**scale**>
Description: Image scaling.
Parent Element(s):
<**post** /> – Image formating after preprocessing.
<**pre** /> – Image formating before preprocessing.

<**segment**
/>
Description: Segment size in stamps (similar to noise cleaning).
Parent Element(s): <**stamp** /> – Removal of stamps.

<**share**
amount = CDATA – Amount of share :=)

/>
Description: ???
Parent Element(s): <**framein** /> – Foreground frame removal.

<**stamp** >
<**segment/**> – Segment size in stamps (similar to noise cleaning).
<**contrast/**> – Stamp contrast. Does this take a value?
<**noise/**> – Noise filter.
</**stamp**>
Description: Sequences of typewriting or stamp prints can be deleted. The parameter Max What is that max? controls the segment size similar to the Noise-Cleaning. One may encounter difficulties when the handwriting and the typewriting segments merge. In this case it is necessary to finish the adjustment manually with an Eraser Tool.
Parent Element(s): <**foreground** /> – Container of foreground filtering parameters.

<**textured** >
<**low/**> – Value of low level texture? [?]
<**high/**> – Value of high level texture? [?]

<**peaks**/> – Value of peak level texture? [?]
<**amount**/> – Amount of texture. [?]
</**textured**>
Description: Requirement is a greyscale image (8 bit). The elimination of the background texture will be supported if the line width of the background pattern is essentially smaller than the line width of the handwriting.
Parent Element(s): <**background** /> – Container of background filtering parameters.

<**trace**
derivation = `CDATA` – Amount/use of derivative.
grayvariance = `CDATA` – Gray level variance.
limit = `CDATA` – ???.
threshold = `CDATA` – ???.
/>
Description: Trace line removal method. This method is suitable for medium and thick writing lines.
Parent Element(s): <**line** /> – Line removal (e.g. handwriting guide lines).

<**userdefined**
/>
Description: Planned form of background removal (future implementation).
Parent Element(s): <**background** /> – Container of background filtering parameters.

<**vertex**
x = `CDATA` – ???
y = `CDATA` – ???
/>
Description: Vertices defining the ROI contour.
Parent Element(s): <**roi** /> – Region of interest. Replace by region?

# H NICIFEAT DTD

This part of the Wanda XML language groups the tags related to measurements implemented by the NICI group. Those application specific tags are customizing the <filter/> of <wandoc/>) (see wandoc documentation). <nicifeat/> is a filter input tag.

### nicifeat DTD Tree

```
nicifeat
 |_(character?,
 |   |_EMPTY
 |
 |__ruler?,
 |   |_(point+)
```

```
 |       |_EMPTY
 |
 |
 |__strokes?,
 |   |_EMPTY
 |
 |__polyline?,
 |   |_(point+) ...
 |
 |__polygon?)
     |_(point+) ...
```

### nicifeat XML example

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- ============================================================
    author: Louis Vuurpijl <vuurpijl@nici.kun.nl>,
            Merijn van Erp <M.vanErp@nici.kun.nl>,
            edited by Isabelle Guyon <isabelle@clopinet.com>
    institution: NICI/Cognitive Engineering, Clopinet
    version: 0.9
    created: 2003-04-07-00-00
    modified: 2003-05-08-00-00
============================================================ -->
<!DOCTYPE nicifeat SYSTEM "nicifeat_.dtd">
<filter name="nicifeat" type="measurement"
        xmlns="pentel.ipk.fhg.de/wanda/nicifeat.dtd">
<inputs number_of="10">
        <input type="stream" number="0" label="Descender">
             <nicifeat name="descender">
                   <character code="j"/>
                   <ruler type="vertical">
                       <point type="top" x="142.0" y="91.0"/>
                       <point type="bottom" x="142.0" y="123.0"/>
                   </ruler>
             </nicifeat>
        </input>
        <input type="stream" number="1" label="Ascender">
             <nicifeat name="ascender">
                   <character code="l"/>
                   <ruler type="vertical">
                       <point type="top" x="194.0" y="92.0"/>
                       <point type="bottom" x="197.0" y="120.0"/>
                   </ruler>
             </nicifeat>
```

```
</input>
<input type="stream" number="2" label="CHeight">
      <nicifeat name="cheight">
            <character code="i"/>
            <ruler type="vertical">
                <point type="top" x="185.0" y="38.0"/>
                <point type="bottom" x="185.0" y="45.0"/>
            </ruler>
      </nicifeat>
</input>
<input type="stream" number="3" label="CWidth">
      <nicifeat name="cwidth">
            <character code="u"/>
            <ruler type="horizontal">
                 <point type="top" x="164.0" y="43.0"/>
                 <point type="bottom" x="180.0" y="43.0"/>
            </ruler>
      </nicifeat>
</input>
<input type="stream" number="4" label="Allo">
        <nicifeat name="allograph">
                <character code="B"/>
                <strokes type="allograph" number_of="2"/>
                <polyline type="stroke" npoints="13">
                        <point x="96.0" y="12.0"/>
                        <point x="96.0" y="20.0"/>
                        <!-- ... -->
                        <point x="99.0" y="44.0"/>
                </polyline>
                <polyline type="stroke" npoints="31">
                        <point x="84.0" y="10.0"/>
                        <point x="99.0" y="10.0"/>
                        <!-- ... -->
                        <point x="80.0" y="64.0"/>
                </polyline>
                <strokes type="true_allograph"
                         number_of="2"/>
                <polyline type="stroke" npoints="13">
                        <point x="96.0" y="12.0"/>
                        <point x="96.0" y="20.0"/>
                        <!-- ... -->
                        <point x="99.0" y="44.0"/>
                </polyline>
                <polyline type="stroke" npoints="31">
                        <point x="84.0" y="10.0"/>
                        <point x="99.0" y="10.0"/>
                        <!-- ... -->
```

```
                        <point x="80.0" y="64.0"/>
                </polyline>
        </nicifeat>
</input>
<input type="stream" number="5" label="Slant">
      <nicifeat name="slant">
           <character code="l"/>
           <sruler>
               <point type="top" x="193.0" y="80.0"/>
               <point type="bottom" x="172.0" y="127.0"/>
           </sruler>
      </nicifeat>
</input>
<input type="stream" number="6" label="ULoop">
        <nicifeat name="uloop">
                <polygon type="pmeter" npoints="50">
                        <point x="45.0" y="38.0"/>
                        <point x="44.0" y="39.0"/>
                        <!-- ... -->
                        <point x="46.0" y="37.0"/>
                </polygon>
        </nicifeat>
</input>
<input type="stream" number="7" label="LLoop">
        <nicifeat name="lloop">
                <polygon type="pmeter" npoints="82">
                        <point x="53.0" y="71.0"/>
                        <point x="52.0" y="72.0"/>
                        <!-- ... -->
                        <point x="54.0" y="70.0"/>
                </polygon>
        </nicifeat>
</input>
<input type="stream" number="8" label="LHeight">
        <nicifeat name="lheight">
                <polyline type="pruler" npoints="3">
                        <point x="201.0" y="11.0"/>
                        <point x="200.0" y="40.0"/>
                        <point x="216.0" y="95.0"/>
                </polyline>
        </nicifeat>
</input>
<input type="stream" number="9" label="Oval">
        <nicifeat name="oval">
             <character code="g"/>
             <ruler type="vertical">
                 <point type="top" x="53.0" y="124.0"/>
```

```
                        <point type="bottom" x="56.0" y="133.0"/>
                    </ruler>
                </nicifeat>
        </input>
</inputs>
<module exec="nicMeasure" type ="client">
        <meta   author="Merijn van Erp"
                institution="NICI"
                version="1.0"/>
</module>
<outputs>
    <output type="stream">
        <features number_of="3">
            <feature name="descender-j"
                     number="1" value="110.0" unit="mm"/>
            <feature name="ascender-k"
                     number="2" value="210.0" unit="mm"/>
            <feature name="width-l"
                     number="3" value="310.0" unit="mm"/>
        </features>
    </output>
</outputs>
</filter>
```

### nicifeat DTD File

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSPY v5 rel. 3 U
    (\url{http://www.xmlspy.com}) by Isabelle Guyon (Clopinet) -->
<!-- ============================================================
    nicifeat  (Wanda measurement feature DTD)

    author: Louis Vuurpijl <vuurpijl@nici.kun.nl>,
            Merijn van Erp <M.vanErp@nici.kun.nl>,
            edited by Isabelle Guyon <isabelle@clopinet.com>
    institution: NICI/Cognitive Engineering, Clopinet
    version: 0.9
    created: 2003-04-07-00-00
    modified: 2003-05-08-00-00
============================================================ -->
<!ENTITY % nicifeat_name_types
            "allograph | ascender | cheight | cwidth |
            descender | lheight | lloop | oval | slant | uloop">
<!ENTITY % nicifeat_ruler_types
```

```
            "horizontal | vertical | sloped">
<!ENTITY % nicifeat_point_types "bottom | left | right | top">
<!ENTITY % nicifeat_char_types
"a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p |
 q | r | s | t | u | v | w | x | y | z |
 A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P |
 Q | R | S | T | U | V | W | X | Y | Z |
 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 ">
<!ENTITY % nicifeat_stroke_types "allograph | true_allograph">
<!ENTITY % nicifeat_polyline_types "stroke | pruler">
<!ENTITY % nicifeat_polygon_types "pmeter">
<!-- ======================================================== -->
<!ELEMENT nicifeat
    (character?, ruler?, strokes?, polyline?, polygon?)>
<!ATTLIST nicifeat
    name (%nicifeat_name_types;) #REQUIRED
>
<!ELEMENT character EMPTY>
<!ATTLIST character
    code (%nicifeat_char_types;) #REQUIRED
>
<!ELEMENT ruler (point+)>
<!ATTLIST ruler
    type (%nicifeat_ruler_types;) #REQUIRED
>
<!ELEMENT strokes EMPTY>
<!ATTLIST strokes
    type (%nicifeat_stroke_types;) #IMPLIED
    number_of CDATA #REQUIRED
>
<!ELEMENT point EMPTY>
<!ATTLIST point
    x CDATA #REQUIRED
    y CDATA #REQUIRED
    type (%nicifeat_point_types;) #IMPLIED
>
<!ELEMENT polygon (point+)>
<!ATTLIST polygon
    type (%nicifeat_polygon_types;) #REQUIRED
    npoints CDATA #REQUIRED
>
<!ELEMENT polyline (point+)>
<!ATTLIST polyline
    type (%nicifeat_polyline_types;) #REQUIRED
    npoints CDATA #REQUIRED
>
```

**nicifeat XML tag reference**

<**character**
code = "a—b—c—d—e—..." – Character code (ASCII or other).
/>
Description: Code of a handwritten character.
Parent Element(s): <**nicifeat** /> – Input of a measurement filter.

<**nicifeat**
name = "allograph—ascender—cheight—cwidth—descender—..." – Name of the nicifeat. >
<**character/**> – Code of a handwritten character. [?]
<**ruler/**> – Ruler measurement defined by two boundary points. [?]
<**strokes/**> – Container of strokes. [?]
<**polyline/**> – Open polygonal line described by its vertices. [?]
<**polygon/**> – Polygon described by its vertices. [?]
</**nicifeat**>
Description: Input of a measurement filter.
Parent Element(s): None

<**point**
type = "bottom—left—right—top" – Point type.
x = `CDATA` – Point x coordinate.
y = `CDATA` – Point y coordinate.
/>
Description: Point in cartesian coordinates.
Parent Element(s):
<**polygon** /> – Polygon described by its vertices.
<**polyline** /> – Open polygonal line described by its vertices.
<**ruler** /> – Ruler measurement defined by two boundary points.

<**polygon**
npoints = `CDATA` – Number of vertices
type = pmeter – Type of polygon. >
<**point/**> – Point in cartesian coordinates. [+]
</**polygon**>
Description: Polygon described by its vertices.
Parent Element(s): <**nicifeat** /> – Input of a measurement filter.

<**polyline**
npoints = `CDATA` – Number of vertices.
type = "stroke—pruler" – Type of polyline. >
<**point/**> – Point in cartesian coordinates. [+]
</**polyline**>
Description: Open polygonal line described by its vertices.
Parent Element(s): <**nicifeat** /> – Input of a measurement filter.

<**ruler**

type = “horizontal—vertical—sloped” – Ruler type. >
<**point/**> – Point in cartesian coordinates. [+]
</**ruler**>
Description: Measurement interactive tool.
Parent Element(s): <**nicifeat** /> – Input of a measurement filter.

<**strokes**
number_of = `CDATA` – Number of strokes.
type = “allograph—true_allograph” – Type of strokes.
/>
Description: Container of strokes.
Parent Element(s): <**nicifeat** /> – Input of a measurement filter.

# I WINK DTD

The wink XML language is part of Wanda XML. It encodes virtual ink (also called electronic ink), as recorded, for instance, by a digitizing tablet. It is inspired by the InkXML standard and by the Unipen standard. Using wink, one can encode standalone virtual ink documents or superimpose virtual ink on top of raster images. In this last case, a wink document may be included into a wandoc document. The modalities of superposition of the ink on top of the image are defined by the tag <trace_context/> tag. A wink encoded handwriting sample may be divided into subsets grouped by the tag <data_block/>, each having a different <trace_context/>.

### wink DTD Tree

```
wink
 |_(device_info?,
 |   |_(id,
 |   |   |_EMPTY
 |   |
 |   |__manufacturer,
 |   |   |_EMPTY
 |   |
 |   |__model,
 |   |   |_EMPTY
 |   |
 |   |__sensitive_area,
 |   |   |_EMPTY
 |   |
 |   |__sample_frequency,
 |   |   |_EMPTY
 |   |
 |   |__sampling_uniformity,
 |   |   |_EMPTY
```

```
|   |
|   |__empiric_displacement_error,
|   |   |_(absolute_error,
|   |   |   |_EMPTY
|   |   |
|   |   |__relative_error)
|   |       |_EMPTY
|   |
|   |
|   |__channel_info_list)
|       |_(channel_info*)
|           |_EMPTY
|
|
|
|__trace_format+,
|   |_(tablet_orientation,
|   |   |_EMPTY
|   |
|   |__channels+)
|       |_(channel+)
|           |_EMPTY
|
|
|
|__data_block?,
|   |_(trace+,
|   |   |_(#PCDATA)
|   |
|   |__trace_context?)
|       |_(image_scale,
|       |   |_EMPTY
|       |
|       |__image_offset,
|       |   |_EMPTY
|       |
|       |__tablet_scale,
|       |   |_EMPTY
|       |
|       |__tablet_offset,
|       |   |_EMPTY
|       |
|       |__bounding_box,
|       |   |_EMPTY
|       |
|       |__brush)
|           |_EMPTY
```

```
 |
 |
 |
 |__wanda_link*,
 |   |_EMPTY
 |
 |__meta?)
     |_EMPTY
```

### wink XML example

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE wink SYSTEM "wink_.dtd">
<wink>
    <device_info>
        <id value="0025Z3004AB"/>
        <manufacturer name="wacom"/>
        <model label="Cintiq" name="ET-0405A-UV2.0-3" code="9" type="81"/>
        <sensitive_area height="92.8" width="127.6"/>
        <sample_frequency value="200"/>
        <sampling_uniformity value="true"/>
        <empiric_displacement_error>
            <absolute_error value="1.5" unit="mm" ref_angle="55" />
            <relative_error value="0.2" unit="mm" delta_angle="5"/>
        </empiric_displacement_error>
        <channel_info_list>
            <channel_info name="pressure_levels" type="decimal"
                        minimum="0" maximum="512"/>
            <channel_info name="x" type="decimal"
                        minimum="0" maximum="127"
                        accuracy="0.5" resolution="1000" />
            <channel_info name="y" type="decimal"
                        minimum="0" maximum="92"
                        accuracy="0.5" resolution="1000" />
            <channel_info name="z" type="decimal"
                        minimum="0" maximum="10"
                        accuracy="0.5" resolution="100" />
            <channel_info name="penAzimuth" type="decimal"
                        minimum="0" maximum="60" />
            <channel_info name="pen_tilt" type="decimal"
                        minimum="0" maximum="60" />
            <channel_info name="time" type="decimal"
                        minimum="0" maximum="1000"/>
        </channel_info_list>
    </device_info>
```

```
    <trace_format>
        <tablet_orientation value="straight"/>
        <channels number_of="7">
            <channel name="pressure_levels"/>
            <channel name="x"/>
            <channel name="y"/>
            <channel name="z"/>
            <channel name="pen_azimuth"/>
            <channel name="pen_tilt"/>
            <channel name="time"/>
        </channels>
    </trace_format>
    <data_block>
        <trace type="pen_down">
            24024.0000 8492.0000 41.0000 0.0000 2250.0000 800.0000
            18980.0000 10399.0000 492.0000 1468.2927 1980.0000 580.0000
            <!-- ... -->
            20678.0000 11345.0000 585.0000 1546.3415 1940.0000 560.0000
        </trace>
        <trace type="pen_up">
            19525.0000 12583.0000 0.0000 1756.0976 1980.0000 540.0000
            19175.0000 12749.0000 0.0000 1760.9756 1960.0000 530.0000
            <!-- ... -->
            17847.0000 13613.0000 0.0000 1800.0000 1960.0000 530.0000
        </trace>
        <trace_context>
            <image_scale value="22" type="dpu"/>
            <image_offset x="50" y="50"/>
            <tablet_scale value="1" type="dpu"/>
            <tablet_offset x="3020.0000" y="1550.0000"/>
            <bounding_box width="100" height="100" />
            <brush size="100"/>
        </trace_context>
    </data_block>
</wink>
```

### wink DTD File

```
<!-- ============================================================

    wink  (Wanda Ink XML language)

    author: Katrin Fanke, Lambert Schomaker, Isabelle Guyon,
            Louis Vuurpijl and the Wanda team
```

```
    institution: Fraunhofer IPK, Rijksuniversiteit Groningen,
                 Clopinet, NICI
    version: 0.9
    created: 2002-03-25-00-00
    modified: 2003-06-13-00-00
-->
<?xml version="1.0" encoding="UTF-8"?>
<!-- ENTITIES============================================== -->
<!ENTITY % bool " true | false ">
<!ENTITY % trace_type "pen_up | pen_down">
<!ENTITY % scale_unit "mmpu | dpu">
<!ENTITY % unit_types "mm | px | u | dot">
<!ENTITY % position " straight | upside_down | rotated_90_clockwise
| rotated_90_counter_clockwise">
<!ENTITY % channel_name
    "* | pressure_levels | x | y | z | pen_azimuth | pen_tilt | time">
<!--======================================================== -->
<!ELEMENT wink (device_info?, trace_format+, data_block?, wanda_link*, meta?)>
<!ELEMENT device_info
(id, manufacturer, model, sensitive_area, sample_frequency,
sampling_uniformity, empiric_displacement_error, channel_info_list)>
<!ELEMENT id EMPTY>
<!ATTLIST id
    value CDATA #REQUIRED
>
<!ELEMENT manufacturer EMPTY>
<!-- manufacturer of device e.g. "wacom"-->
<!ATTLIST manufacturer
    name CDATA #REQUIRED
>
<!ELEMENT model EMPTY>
<!-- model number of device
     e.g. "ET-0405A-UV2.0-3" for graphire2-->
<!ATTLIST model
    label CDATA #IMPLIED
    name CDATA #REQUIRED
    code CDATA #IMPLIED
    type CDATA #IMPLIED
>
<!ELEMENT sensitive_area EMPTY>
<!ATTLIST sensitive_area
    height CDATA #REQUIRED
    width CDATA #REQUIRED
>
<!ELEMENT sample_frequency EMPTY>
<!-- sample_frequency / data_rate -->
<!ATTLIST sample_frequency
```

```
    value CDATA #REQUIRED
>
<!ELEMENT sampling_uniformity EMPTY>
<!ATTLIST sampling_uniformity
    value (%bool;) #REQUIRED
>
<!ELEMENT empiric_displacement_error (absolute_error, relative_error)>
<!ELEMENT absolute_error EMPTY>
<!ATTLIST absolute_error
    value CDATA #REQUIRED
    unit (%unit_types;) #REQUIRED
    ref_angle CDATA #REQUIRED
>
<!ELEMENT relative_error EMPTY>
<!ATTLIST relative_error
    value CDATA #REQUIRED
    unit (%unit_types;) #REQUIRED
    delta_angle CDATA #REQUIRED
>
<!ELEMENT channel_info_list (channel_info*)>
<!ELEMENT channel_info EMPTY>
<!ATTLIST channel_info
    name (%channel_name;) #REQUIRED
    type (decimal | boolean) #IMPLIED
    minimum CDATA #IMPLIED
    maximum CDATA #IMPLIED
    accuracy CDATA #IMPLIED
    resolution CDATA #IMPLIED
>
<!ELEMENT trace_format (tablet_orientation, channels+)>
<!ELEMENT tablet_orientation EMPTY>
<!ATTLIST tablet_orientation
    value (%position;) #REQUIRED
>
<!ELEMENT channels (channel+)>
<!ATTLIST channels
    number_of CDATA #IMPLIED
>
<!ELEMENT channel EMPTY>
<!ATTLIST channel
    name (%channel_name;) #REQUIRED
>
<!ELEMENT data_block (trace+, trace_context?)>
<!ELEMENT trace (#PCDATA)>
<!ATTLIST trace
    type (%trace_type;) #REQUIRED
>
```

```
<!ELEMENT trace_context
    (image_scale, image_offset, tablet_scale, tablet_offset,
    bounding_box, brush)>
<!ELEMENT image_scale EMPTY>
<!ATTLIST image_scale
    value CDATA #REQUIRED
    unit (%scale_unit;) #REQUIRED
>
<!ELEMENT image_offset EMPTY>
<!ATTLIST image_offset
    x CDATA #REQUIRED
    y CDATA #REQUIRED
>
<!ELEMENT tablet_scale EMPTY>
<!ATTLIST tablet_scale
    value CDATA #REQUIRED
    unit (%scale_unit;) #REQUIRED
>
<!ELEMENT tablet_offset EMPTY>
<!ATTLIST tablet_offset
    x CDATA #REQUIRED
    y CDATA #REQUIRED
>
<!ELEMENT bounding_box EMPTY>
<!ATTLIST bounding_box
    height CDATA #REQUIRED
    width CDATA #REQUIRED
>
<!ELEMENT brush EMPTY>
<!ATTLIST brush
    size CDATA #REQUIRED
>
<!ELEMENT meta EMPTY>
<!ATTLIST meta
    author CDATA #IMPLIED
    email CDATA #IMPLIED
    institution CDATA #IMPLIED
    version CDATA #IMPLIED
    created CDATA #IMPLIED
    modified CDATA #IMPLIED
>
<!ELEMENT wanda_link EMPTY>
<!ATTLIST wanda_link
    href    CDATA    #REQUIRED
>
```

**<absolute_error**
ref_angle = CDATA – Absolute error reference angle in degrees.
unit = "mm—px—u—dot" – Absolute error unit.
value = CDATA – Absolute error value.
/>
Description: Absolute displacement error.
Parent Element(s): <**empiric_displacement_error** /> – Empirical displacement error.

**<bounding_box**
height = CDATA – Height of the bounding box.
width = CDATA – Width of the bounding box.
/>
Description: The bounding box of the wink canvas.
Parent Element(s): <**trace_context** /> – Information allowing to superimpose the trace on top of an image.

**<brush**
size = CDATA – Brush size in unit "u".

/>
Description: Brush parameters.
Parent Element(s): <**trace_context** /> – Information allowing to superimpose the trace on top of an image.

**<channel**
name = "*—pressure_levels—x—y—z—..." – Name of the channel (e.g., x, y, z). Should be one of the names declared in channel_info_list.
/>
Description: All the coordinates are stored at maximum sampling rate, no sub-sampling. x, y (and z if recorded) are recorded in unit u (see trace_context). The pressure pressure_level, is in arbitrary units. The azimuth pen_azimuth is in degrees. The tilt pen_tilt is in degrees.
Parent Element(s): <**channels** /> – Container of channel entries that are currently used to record traces.

**<channel_info**
accuracy = CDATA – Accuracy with wich the channel is measured in the original tablet unit.
maximum = CDATA – Maximum value this channel can take in the original tablet unit.
minimum = CDATA – Minimum value this channel can take in the original tablet unit.
name = "*—pressure_levels—x—y—z—..." – Name of the channel. See also channel in traceFormat.
resolution = CDATA – Resolution of that channel in the original tablet unit.
type = "decimal—boolean" – Boolean or decimal.
/>
Description: A channel is a recorded coordinate (position, pressure, pen orientation). channelInfo declares all the channels supported by the tablet and their parameters.

Parent Element(s): <**channel_info_list** /> – Container of channel info entries.

<**channel_info_list** >
<**channel_info/**> – Declaration of a particular channel and channel information. [*]
</**channel_info_list**>
Description: List of available channels.
Parent Element(s): <**device_info** /> – Information relating to the tablet as precised by the manufacturer.

<**channels**
number_of = `CDATA` – Number of channels.
>
<**channel/**> – One of the recorded channels in a trace. [+]
</**channels**>
Description: Container of channel declarations.
Parent Element(s): <**trace_format** /> – Format of a trace listing channels used and tablet orientation.

<**data_block** >
<**trace/**> – The actual ink recorded as a sequence of coordinate points. [+]
<**trace_context/**> – Information allowing to superimpose the trace on top of an image. [?]
</**data_block**>
Description: Block grouping several traces having same trace_context.
Parent Element(s): <**wink** /> – Root element of the Wanda ink XML.

<**device_info** >
<**id/**> – A unique id such as the serial number.
<**manufacturer/**> – Tablet manufacturer.
<**model/**> – Tablet model information.
<**sensitive_area/**> – Dimensions of the tablet sensitive area.
<**sample_frequency/**> – Sampling rate in points per second.
<**sampling_uniformity/**> – Points regularly sampled or not.
<**empiric_displacement_error/**> – Empirical displacement error.
<**channel_info_list/**> – Container of channel info entries.
</**device_info**>
Description: The device (tablet) information may be stored with the data or separately using wanda_link. It should not affect the rendering of the ink over the image, which is described by trace_context.
Parent Element(s): <**wink** /> – Root element of the Wanda ink XML.

<**empiric_displacement_error** >
<**absolute_error/**> – Absolute displacement error.
<**relative_error/**> – Relative displacement error.
</**empiric_displacement_error**>
Description: Due to the construction of the electronic pen (location LC-circuit) there are displaced activations of the electromagnetic field depending on the pen-tilt. This displacement can be empirically determined and later used for the automatic correction while assigning on- and offline data.

Parent Element(s): <**device_info** /> – Information relating to the tablet as precised by the manufacturer.

<**id**
value = CDATA – Tablet id value.

/>
Description: A unique id such as the serial number.
Parent Element(s): <**device_info** /> – Information relating to the tablet as precised by the manufacturer.

<**image_offset**
x = CDATA – X coordinate of the offset in unit "u".
y = CDATA – Y coordinate of the offset in unit "u".
/>
Description: Position of the wink canvas origin relative to the image region origin, in the data_block unit "u".
Parent Element(s): <**trace_context** /> – Information allowing to superimpose the trace on top of an image.

<**image_scale**
unit = "mmpu—dpu" – Unit of the image scale.
value = CDATA – Scaling factor value.
/>
Description: Scaling factor relating the image unit to the data_block unit "u". The original image coordinates are recovered by multiplying the channels x and y of the trace by the image_scale factor after adding image_offset.
Parent Element(s): <**trace_context** /> – Information allowing to superimpose the trace on top of an image.

<**manufacturer**
name = CDATA – Name of the tablet manufacturer.

/>
Description: Tablet manufacturer.
Parent Element(s): <**device_info** /> – Information relating to the tablet as precised by the manufacturer.

<**meta**
author = CDATA – Author of the annotations of this part of the document.
created = CDATA – Date created in the YYYY-MM-DD-mm-ss format.
email = CDATA – Contact email.
institution = CDATA – Author's affiliation.
modified = CDATA – Date last modified in the YYYY-MM-DD-mm-ss format.
version = CDATA – DTD version number.
/>
Description: Information about how the document annotations were generated.
Parent Element(s): <**wink** /> – Root element of the Wanda ink XML.

<**model**
code = CDATA – Manufacturer model code (e.g. stored in Wacom.dat).
label = CDATA – User defined label to name the tablet (e.g. consumer market name).
name = CDATA – Unique name used by the device drivers (e.g. found in the Windows registry).
type = CDATA – Manufacturer model type (e.g. stored in Wacom.dat).
/>
Description: Tablet model.
Parent Element(s): <**device_info** /> – Information relating to the tablet as precised by the manufacturer.

<**relative_error**
delta_angle = CDATA – Angle of relative displacement error in degrees.
unit = "mm—px—u—dot" – Relative error unit.
value = CDATA – Relative error value.
/>
Description: Relative displacement error.
Parent Element(s): <**empiric_displacement_error** /> – Empirical displacement error.

<**sample_frequency**
value = CDATA – Sampling rate value in points per seconds.

/>
Description: Sampling rate.
Parent Element(s): <**device_info** /> – Information relating to the tablet as precised by the manufacturer.

<**sampling_uniformity**
value = "true—false" – True is points regularly spaced (no skip), false otherwise.
/>
Description: Regularity of sampling.
Parent Element(s): <**device_info** /> – Information relating to the tablet as precised by the manufacturer.

<**sensitive_area**
height = CDATA – Sensitive area height.
width = CDATA – Sensitive area width.
/>
Description: Dimensions of the tablet sensitive area.
Parent Element(s): <**device_info** /> – Information relating to the tablet as precised by the manufacturer.

<**tablet_offset**
x = CDATA – X coordinate of the offset in unit "u".
y = CDATA – Y coordinate of the offset in unit "u".
/>
Description: Position of the wink canvas origin relative to the tablet origin, in the data_block unit "u".
Parent Element(s): <**trace_context** /> – Information allowing to superimpose the trace on top

of an image.

**<tablet_orientation**
value = "straight—upside_down—rotated_90_clockwise—rotated_90_counter_clockwise" – One of four orientation values.
**/>**
Description: Tablet orientation with respect to the manufacturer upright position.
Parent Element(s): **<trace_format />** – Format of a trace listing channels used and tablet orientation.

**<tablet_scale**
unit = "mmpu—dpu" – Unit of the tablet scale.
value = `CDATA` – Scaling factor value.
**/>**
Description: Scaling factor relating the tablet unit to the data_block unit "u". The original tablet coordinates are recovered by multiplying the channels x, y, and z of the trace by the tablet_scale factor after adding tablet_offset.
Parent Element(s): **<trace_context />** – Information allowing to superimpose the trace on top of an image.

**<trace**
type = "pen_up—pen_down" – Pen up / pen down information. By default, pen down is assumed.
**</trace>**
Description: The actual ink recorded as a sequence of coordinate points.
Parent Element(s): **<data_block />** – Block grouping several traces having same trace_context.

**<trace_context >**
**<image_scale/>** – Scaling factor relating the image unit to the data_block unit "u".
**<image_offset/>** – Position of the wink canvas origin relative to the image region origin.
**<tablet_scale/>** – Scaling factor relating the tablet unit to the data_block unit "u".
**<tablet_offset/>** – Position of the wink canvas origin relative to the tablet origin.
**<bounding_box/>** – The bounding box of the wink canvas.
**<brush/>** – Brush parameters.
**</trace_context>**
Description: Information allowing to superimpose the trace on top of an image.
Parent Element(s): **<data_block />** – Block grouping several traces having same trace_context.

**<trace_format >**
**<tablet_orientation/>** – Tablet orientation with respect to the manufacturer upright position.
**<channels/>** – Container of channel entries that are currently used to record traces. [+]
**</trace_format>**
Description: The trace_format block groups channel and tablet orientation information. Compared to InkXML, the trace_format is simplified. The sampling rate is assumed to be the maximum sampling rate sample_frequency provided in the device_info block. We do not provide the resolution

as a channel attribute for x, y, and z channels. It is provided in the device_info block.
Parent Element(s): <**wink** /> – Root element of the Wanda ink XML.

<**wanda_link**
href = `CDATA` – Link reference (URL or file name).

/>
Description: Element that will be replaced by the file it points to (e.g. a description file).
Parent Element(s): <**wink** /> – Root element of the Wanda ink XML.

<**wink** >
<**device_info/**> – Information relating to the tablet as precised by the manufacturer. [?]
<**trace_format/**> – Format of a trace listing channels used and tablet orientation. [+]
<**data_block/**> – Block grouping several traces having same trace_context. [?]
<**wanda_link/**> – Element that will be replaced by the file it points to. [*]
<**meta/**> – Information about how the document annotations were generated. [?]
</**wink**>
Description: The wink root element that opens a Wanda ink block.
Parent Element(s): None

## References


[1] Wissenschaftliche Anforderungen an aussgepsychologische Begutachtungen (Glaubhaftigkeitsgutachten). BGH, Urt. v. 30.7. 1999 -StR 618/98 (LG Ansbach). *Strafverteidiger*, 9, 473-478, (in German).

[2] W.C. de Jong, L.N. Kroon van der Kooij, and D.Ph. Schmidt. Computer aided analysis of handwriting, the NIFO-TNO approach. In *Proc. 4th European Handwriting Conference for Police and Government Handwriting Experts*, 1994.

[3] XML eXtensible Markup Language. *<http://www.w3.org/XML/>*.

[4] H. Gieschen, P. Heyne, and M. Philipp. FISH - Forensisches Informationssystem Handschriften: Abschlußbericht. Technical report, Bundeskriminalamt Wiesbaden, Thaerstraße 11, 65193 Wiesbaden, Germany, 1997. (in German).

[5] M.R. Hecker. *Forensische Handschriftenuntersuchung.* Kriminalistik Verlag, 1993. (in German).

[6] N. Köller, K. Nissen, M. Riß, and E. Sadorf. Probabilistische Schlußfolgerungen in Schriftgutachten - Zur Vereinheitlichung der Sprachregelung im Bereich der verbalen Wahrscheinlichkeidsgrade. Technical report, PROJEKTGRUPPE "Vereinheitlichung der Sprachregelung im Bereich der verbalen Wahrscheinlichkeidsgrade" der Kommision "Kriminalwissenschaft und -technik/Erkennungsdienst", 2002.

[7] W. Kuckuck and M. Philipp. FISH-Das forensische Informations-Sysetm Handschriften. In W. Conrad and B. Stier, editors, *Grundlagen, Methoden und Ergebnisse der forensischen Schriftenuntersuchung*, pages "159–188". Schmidt-Römhild, Lübeck, Germany, 1989. (in German).

[8] L. Michel. *Gerichtliche Schriftvergleichung.* De Gruyter, 1982. (in German).

[9] M. Philipp. Expected future developments in the Forensic Information System Handwriting (FISH). In *4th European Conerence for Police and Government Handwriting Experts*, London, UK, 1994.

[10] M. Philipp. Fakten zu FISH, Das Forensische Informations-System Handschriften des Bundeskriminalamtes - Eine Analyse nach über 5 Jahren Wirkbetrieb. Technical report, Kriminaltechnisches Institut 53, Bundeskriminalamt, Thaerstraße 11, 65173 Wiesbaden, Germany, 1996. (in German).

[11] L.R.B. Schomaker and L.G. Vuurpijl. Forensic writer identification: A benchmark data set and a comparison of two systems. Technical report, Nijmegen Institute for Cognition and Information (NICI), University of Nijmegen, The Netherlands, 2000.

[12] W3C - World Wide Web Consortium. *<http://www.w3.org/>*, 2003.

# Index