

WANDA: A Generic Framework applied in Forensic Handwriting Analysis and Writer Identification

Katrin Franke¹, Lambert Schomaker², Christian Veenhuis¹, Christian Taubenheim¹,
Isabelle Guyon³, Louis Vuurpijl⁴, Merijn van Erp⁴, Geertje Zwarts²

¹ *Department of Security and Testing Technology,
Fraunhofer Institute for Production Systems and Design Technology, Germany.*

² *Artificial Intelligence Institute, Rijksuniversiteit Groningen, The Netherlands.*

³ *Clopinet, Berkeley, USA*

⁴ *Nijmegen Institute for Cognition and Information,
University of Nijmegen, The Netherlands.*

Email: franke@ipk.fhg.de, schomaker@ai.rug.nl

Abstract. This paper presents the WANDA Workbench, which is an open framework for electronic data processing. The framework provides generic interfaces for 'plug-in' applications for graphical user interfaces (client desktop with client plug-ins) and processing modules (server with server plug-ins). The applied plug-in concept allows for the functional extension of the workbench without changing the framework. Moreover, for data modeling, messaging and system configuration the eXtensible Markup Language (XML) is implemented, which supports the interoperability with existing applications and allows for continuous adaptations to the evolving state of technology.

The current application domain of the framework is writer identification and handwriting examination as frequently used in crime investigation and prosecution. Dedicated plug-ins using digital image processing and pattern recognition establish objective measurements and feature extraction to promote reproducible analysis results. Moreover, based on domain knowledge, which was collected from forensic document examiners, the data standard WANDAXML was designed that allows for the annotation, processing, journaling and storage of digitized questioned handwriting documents.

In the following, we detail the adopted design philosophy, provide pointers for employing the data model, and sketch the realized framework and current plug-ins.

The generic framework with appositive plug-in programmer handbooks as well as the data standard will be made available to the research community through the Internet <http://pentel.ipk.fhg.de/wanda>.

1 Introduction

The computer-based identification of a writer on the basis of a digitized piece of handwriting is a challenging task for pattern recognition. A number of systems have been in use in Europe, the United States, and Australia. However, most of these systems are becoming increasingly outdated. Latest research results in digital image processing and pattern recognition are not being considered there. Moreover, common data standards, procedures, and generic system environments are lacking.

In order to establish common grounds for international exchange, a standardized approach is proposed for the storage and 'plug in' application of analysis and evaluation procedures in forensic writer identification. The major objective is to create a usable and effective platform for forensic writer identification, in particular by means of:

1. *Standardization* of the data-format,
2. *Modularization* and *Extendibility* of the system concept,
3. *Objectification* in measurements and *Reliability* of analysis results.

A new writer identification system should at least be a viable solution for the next decade, and longer. It should allow continuous adaptations to follow the current state of technology and it should be extensible, keeping backward compatibility in mind. In order to ensure a long-term operation, two stable standards should be the basis of the proposed approach: the programming language JAVA and the data interchange language XML (eXtensible Markup Language). Particularly the usage of XML [2] supports the interoperability of existing and future applications. The greatest advantage of XML as a data interchange language is that it is text-based and human-readable. Therefore, it is independent from a particular platform, a particular programming language, and a particular middle-ware. XML tags can be freely defined and adapted to special needs, which makes XML so powerful and so flexible that it can be considered for a wide range of purposes as for example system configurations, data transfer protocols, and rich documents.

In the following we will sketch the application domain and the general usage scenario of a computer-based system for forensic handwriting analysis and writer identification (section 2). Next, we detail the concepts of the framework design (section 3), and provide technical aspects on the system architecture (section 4). Finally, we outline the current application of the presented framework and provide pointers to more general approaches (section 5). Envisaged further developments will conclude the work presented here (section 6).

2 Application Domain

In classical forensic handwriting examination, a human expert compares handwritings on the basis of well-defined sets of characteristics [9, 7]. As other related fields of forensic science, handwriting examination is primarily based on the knowledge and experience of the forensic expert. Due to the problem of non-objective measurements and non-reproducible decisions, it was attempted to support traditional methods, like visual inspection and expert rating, by computerized semi-automatic and interactive systems [1, 12, 4, 15]. Subject of computer-based forensic handwriting examination are human handwriting samples, which might be digitized by using an optical device such as a digital camera or scanner, handwriting samples that are digitized by employing an electronic writing tablet or an electronic pen, but also multi-modal digitized handwriting, and, signatures.

The development of a system for forensic handwriting analysis has to be grounded in established forensic investigation methods. Furthermore, potential users of an upcoming system should be involved during the specification and evaluation phase.

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 1 shows the stages in the work flow of handling forensic handwritten samples. Each stage is characterized by its typical requirements.

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
2.c	feature extraction	batch
3.a	normalization	batch
3.b	comparison	interactive
4.a	normalization	batch
4.b	pre-selection	interactive
4.c	identification matching	batch
4.d	hit-list inspection	interactive
5	report generation	interactive

Table 1: Stages in the forensic processing of handwritten samples. Note: stages 3 and 4 can be alternative performed. While stage 3 covers procedures for verifying a questioned handwriting sample against available reference material, stage 4 is representing the identification procedures for resembling material out of a pool.

3 Framework design

The concept behind the WANDA framework 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. Hence, within the WANDA framework XML shall be used for: (1) Data modeling and annotation, (2) journaling of data processing, (3) data transfer protocol, and (4) system configuration.

The second insight is that it is desirable to frequently reuse/share used mechanisms in each new application. A set of software routines that cover fundamental mechanisms like accepting input from a user and responding by performing an action is also called an application framework [11]. For solving a specific problem a framework needs to be extended according to one or multiple domain models describing the structure of the data and how it is processed.

There are a number of open source and commercial software products that provide such called *Plug-In Interfaces*. To give an example, “ImageJ” and “Adobe Photoshop” are software programs for digital image processing, which functionality can be extended by these so-called filter plug-ins.

This concept was a source of inspiration for the WANDA framework design. Beyond that, WANDA should allow for generic data processing comprising data filtering, feature extraction, and classification. Moreover, the application domain, processing flow, and data structures should not have to be predefined. The framework with its domain-specific plug-ins might be tailored to tasks at hand, not to overwhelm the user and the user’s budget. At the same time,

it should allow for several plug-in extensions in kind and in count.

During the design of the discussed framework we had forensic handwriting analysis in mind, but kept our view open to further application domains. In the following section 4, we will describe the technical aspects of the framework's architecture without strong references to a particular application domain. Afterwards, in section 5 we exemplify, for the domain of forensic writer identification, how domain-specific data structures and corresponding processing procedures are joint by means of the WANDXML language.

4 System Architecture

A systematic overview of the WANDA system architecture is given in figure 1. The guiding idea is a rather strict separation of data, graphical user interfaces and processing modules. In order to support long-term interoperability between the diverse system components, data and process instructions are described in XML documents. System configurations and data transfer protocols are also XML -based. In the following, we will briefly describe the major components, being: (a) clients with client plug-ins, which implement the graphical user interface, (b) servers with server plug-ins, which implement the processing modules, (c) data repositories, and working sets, and, (d) web clients for remote data retrieval.

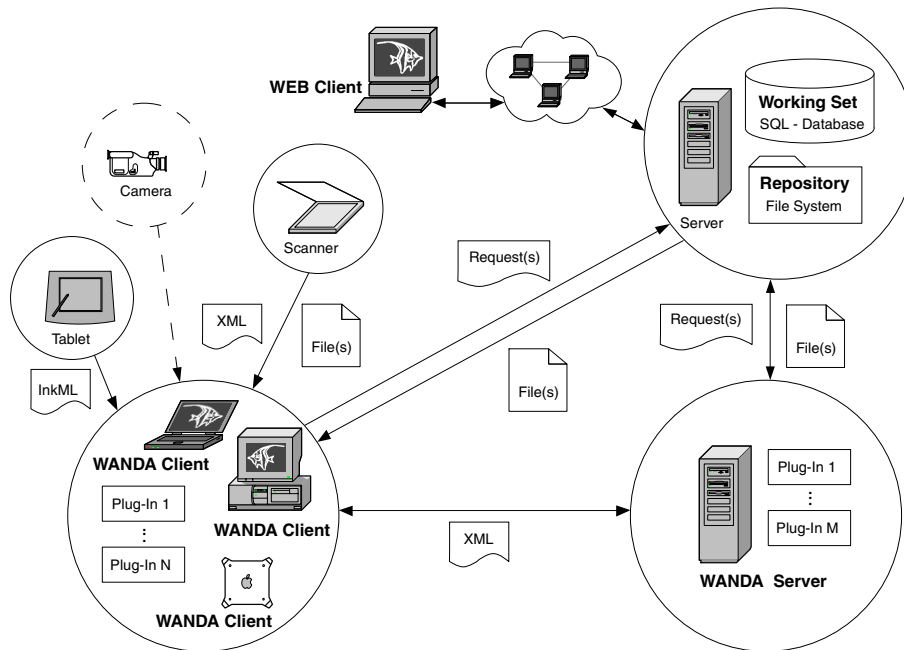


Figure 1: WANDA Workbench comprising: client(s), server(s), data repository(s), and web client(s).

4.1 WANDA Client and plug-in concept for graphical user interfaces

The WANDA client is an integrated desktop environment for all plug-ins that come with a graphical user interface. This integrated desktop environment provides the user with a toolbar representing the available tools and a multiple document interface for displaying and interacting with these tools. Moreover, the client desktop acts as a manager for tools and provides them with: (Ia) Intercommunication (message exchange between the tools), (Ib) Access to

commonly used functions such as network, printing etc. via centralized functions, (Ic) Window management (order/tiling/arrangement), and (Id) Extended drag-on-drop functionality.

The guiding design principles were (IIa) easy to extend with new tools, (IIb) simple usage for tool developers, and, (IIc) a user-friendly graphical interface. Various tools should be able to be integrated into the client desktop by following a plug-in concept. This means there's no limitation on how many or what kind of plug-ins will be developed for the client desktop, as long as they adhere to some simple requirements. A developer who is programming tools for the framework, should only have to make minor changes (use a common base class) to fit a former stand-alone JAVA application into the client desktop. For intercommunication with other tools or windows, a simple Application Programming Interface (API) is provided.

The plug-in concept of the client desktop is realized by XML -based configuration files. A self-explaining example is given in table 2. Based on the `<client_plugin>` description, JAVA software packages as well as required resource bundles will be automatically loaded during the start up of the client desktop and these will extent the desktop with the desired functionality.

The organization of the client desktop and its plug-ins (see figure 2a) reveals similarities with communication architectures for pattern recognition agents that the authors proposed in earlier work [20]. However, the current version of the WANDA workbench does yet not implement a pure autonomous agent concept. Particularly, the communication between the plug-ins needs to be controlled by the user, either by employing predefined communication protocols or by human user interaction.

The client desktop is entirely written in JAVA . Thus, it is platform-independent and can operate on every platform which has installed the JAVA 1.4 runtime environment or higher. So far, we used it successfully on Linux RedHat, Windows 2000, and MacOS X. Detailed instructions for the implementation of WANDA client plug-ins are given in [17].

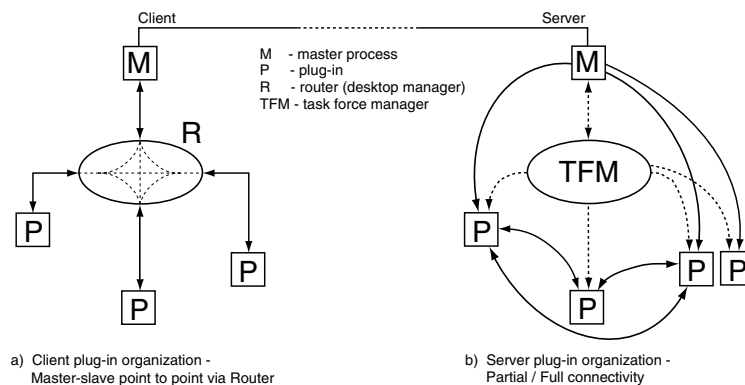


Figure 2: Current WANDA client (a) and server (b) plug-in organization inspired by multi-agent communication architectures [20].

4.2 WANDA Server and plug-in concept for processing modules

The WANDA server provides the functionality of processing modules (plug-ins) over a network. By connecting to the server, clients can call upon these plug-ins via TCP/IP network sockets. Beside its functionality, the server plug-in's meta-data as for example name, description or version numbers, can also be obtained. The server is multi-threaded. Thus, it can

Sample-Description for WANDA Client Plug-in

```
<client_plugin
  name="wandaWriter">
  <description>
    Writer Annotation
  </description/>
  <platform name="all"/>
  <module
    exec="writer.Writer"
    type="client"/>
  <lib value="writer.jar"/>
  <icon
    value="writer/writer.png"/>
  <menu
    category="annotation"
    toolbar="true"/>
  <style
    resizable="false"
    maximizable="false"
    iconifiable="true"/>
</client_plugin>
```

Sample-Description for WANDA Server Plug-in

```
<server_plugin
  name="wandaClean"
  numInputs="2"
  numOutputs="1">
  <description>
    Performs background removal and
    imprint cleaning
  </description>
  <platform name="windows"/>
  <call>
    <location>
      wandaClean/wandaClean.exe
    </location>
    <invocation>
      <use_input number="0"/>
      <use_output number="0"/>
      <use_input number="1"/>
    </invocation>
  </call>
  <protocol>
    <to_server>
      <transfer_input
        number="0"
        type="image"/>
      <transfer_input
        number="1"
        type="xml"/>
    </to_server>
    <call_plugin/>
    <to_client>
      <transfer_output
        number="0"
        type="image"/>
    </to_client>
  </protocol>
  <errors>
    <error
      code="-1"
      msg="Invalid arguments"/>
    <error
      code="-2"
      msg="Not enough memory"/>
  </errors>
</server_plugin>
```

Table 2: Samples of description files for WANDA Client and Server Plug-ins. Those descriptions, have to be provided for each software module intending to expand workbench's functionality. Particularly, they are considered to perform system configuration and plug-in invocation.

process any number of client requests concurrently. Each client gets its own corresponding thread on the server side and access to all installed plug-ins. The server organization is also inspired by the communication architectures between pattern recognition agents mentioned earlier [20]. As depicted in figure 2b, a *master process* can invoke one or multiple plug-ins and plug-ins can be invoked recursively. Moreover, there is a so-called *task force manager* that maintains an overview of available plug-ins and their location on the Internet.

A plug-in might provide a single algorithm, complete methods or other auxiliary functions in the envisaged application domain. The plug-in can be *any executable binary or script*, which can be written in any programming or script language. To manage this, an XML -based description for loading plug-ins within the server was developed. A self-explaining example is given in table 2. This `<server_plugin>` description defines the invocation of server plug-ins, including the order and type of arguments as well as the sequence of processing steps defined as individually needed protocols to describe the data flow between a WANDA client and a server plug-in. While initializing, the server scans a specific directory to get all installed plug-ins. Each plug-in consists not only of its binary files but also of the mentioned xml-based description.

The server is implemented in the programming language JAVA and has been successfully used on Linux RedHat, Windows 2000 and MacOS X. Detailed instructions for the implementation of WANDA server plug-ins are given in [19].

4.3 XML-based communication language

The communication between a client and a server as well as the transfer of data between a client and an appropriate plug-in are realized by XML documents (figure 1). Additionally, the input and output data files for example, digital images can be exchanged between client and server. The guiding idea for implementing the communication `<protocol>` (compare table 2) was the development of a W3C standard for sending messages between applications, which is called Simple Object Access Protocol (SOAP) [16]. However, SOAP is still under development, which bears risks for ad-hoc application. So, we decided to prepare the communication infrastructure, and to take advantage of SOAP's platform-independent messaging protocol in the near future.

4.4 Repository and Working Set

The *Repository* is a restricted file system where all data files are stored. The data files, as for example a digital image, are stored together with its XML -based description files, be it for annotation, features or processing journals. The file system area is intended to be exclusively accessed via a dedicated PHP Hypertext Preprocessor (PHP) script [13] that allows for administrating the restricted file system (create/delete/inspect directories) but also for up- and downloading files via the Hyper Text Transfer Protocol (HTTP).

During the development, it became apparent that the requirements for a fast and effective data entry selection cannot be met by the current XML software, i.e., at the scale which is required by the actual application domain ($\gg 20.000$ forensic 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 handling and SQL-based logical subset selection. Please, keep in mind that the usage of

traditional databases is optional and not necessarily required for small and medium-sized applications.

The proposed solution is to assume that the repository (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 still be able to recognize the structure of the XML. The data format, which is in our case the Target Image File 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, which is performed by the user via a client desktop.
2. Batch commitment of a complete XML directory tree or sub tree to the working-set database. This action should be performed by an administrator via a dedicated client and/or server tool.

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. The implemented tool *xml2ws* was written for PostgreSQL [14] as well as MySQL [10] databases.

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.

5 Framework Applications

5.1 WANDA Workbench 2003

The realized prototype allows for fundamental processing of handwriting samples in the context of forensic handwriting examination and writer identification, particularly: Data acquisition (scanning, tablet input, combined); Preprocessing offline data; Annotation of handwritten documents (in forensic context); Measurement of selected handwriting characteristics, and Search in the data pool. Hence, the WANDAXML language [5] is decomposed into the following subsets:

<code><wandoc></code>	describing a document with pages, regions of interest as well as filters (operations applied to the documents), and annotation encapsulation;
<code><scan></code>	filter subset encoding scanner operations;
<code><proper></code>	filter subset encoding image preprocessing [6];
<code><wink></code>	filter subset encoding electronic ink, particularly an application-specific extension of the Ink Markup Language (InkML) [8];
<code><writer></code>	annotation subset describing writer;
<code><script></code>	annotation subset describing handwriting script;
<code><material></code>	annotation subset describing "material" (e.g. paper, pen, ink);
<code><content></code>	annotation subset describing document content;
<code><nicifeat></code>	filter subset encoding measurements [18];
<code><profiler></code>	filter subset encoding image color management.

All subsets are independent and implemented by dedicated plug-ins. The only exception is `<wandoc>`, which acts as a "wrapper", and, represents the WANDA Framework applied in forensic handwriting analysis and writer identification (compare table 3). The WANDAXML design revolves around a small number of key concepts: annotations, filters, and regions (see table 3 and table 4).

- *Annotations:* Within `<wandoc>` we have defined a generic tag `<annotation>`, which wraps around annotations of different types. 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. 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 or to regions. Examples of regions include paragraphs, lines, words, and characters. The region types can be defined by the users.

5.2 Generic Approaches

Investigation report: According to current forensic practice and the required quality assurance, such called feature protocols have to be provided for each handwriting investiga-

Skeleton for WANDA Documents	Skeleton for WANDA Regions
<pre> <wandoc> <filters/ > [?] <annotations / > [?] <wanda.link / > [*] <pages / > [?] <page / > [+] <filters/ > [?] <annotations/ > [?] <regions/ > [?] <wanda.link/ > [*] <meta/ > [?] </page> </pages> <meta/ > [?] </wandoc> </pre>	<pre> <region> [+] <points> [+] <point/ > </points> <filters> [?] <filter/ > </filters> <annotations> [?] <annotation/ > </annotations> <regions> [?] <region/ > </regions> <wanda.link/ > [*] </region> </pre>

Table 3: Adopted skeletons for wandoc and region

tion/comparison. Those protocols are being supplied for the questioned handwriting sample as well as for the reference handwritings. Performing these procedures manually is 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. On users demand the annotated data might be transferred into the case management system. Also, a text document could be generated, e.g. by means of XSL Transformations (XSLT) [3, 21], which can be further completed by using a standard text processor. In this way it will be possible to speed up the documentation of forensic evidences as well as to harmonize those documentations.

Moreover, 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 proposed format will ensure that data can be rapidly imported into another computer system that does not need to be provided by the same manufacturer. So, WANDA promotes interoperability, and, with the anchors for further extensions, long-term usability.

Research and Development: Besides representing existing practices in forensic data analysis in a standard way, WANDA can facilitate experimenting with new ideas. The framework and data 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

Skeleton for WANDA Filters	Sample for WANDA Annotations
<pre> <filter> [+] <inputs> [+] <input/> [+] </inputs> <module/> [+] <outputs> [+] <output/> [+] </outputs> </filter> </pre>	<pre> <annotation label="WANDA Writer" xmlns="../writer.dtd"> <writer> ... </writer> </annotation> </pre>

Table 4: Adopted skeletons for filter and annotation

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.

6 Future Work

There are many ideas for the further extension and application of the WANDA Workbench. Primary we are envisaging its usage in forensic case work. We expect a huge potential for research and development in the field of forensic handwriting analysis and writer identification. Particularly, the distributed client/server architecture allows for joining and sharing collected handwriting data, domain-specific knowledge, and implemented software routines.

In the near future, we foresee adaptations and extensions of the data model and its corresponding modeling language WANDAXML according to reviews by forensic handwriting experts as well as by new demands arising with upcoming investigation and analysis objectives. It is understood that those objectives will also result in the integration of further client and server plug-ins.

Beside the handwriting related research questions we are thinking of further improvements of the WANDA Workbench itself, as for example providing easy-to-use administration tools with graphical user interfaces and wizards. Another point is extending the workbench to customizable central views. It should be possible that developers can implement and integrate their own central views. This is the final step for allowing such called *domain themes* were the WANDA workbench can be set up to an application domain by using a dedicated data viewer e.g. video player with corresponding client and server plug-ins as for example for video processing.

7 Acknowledgments

Although many persons have directly or indirectly contributed to the framework design and implementation, special thank goes to those who were actively involved, particularly: Ottmar Bünnemeyer, Mario Köppen, Tomas Kühn, Altug Metin, Martin Penk, Steffen Rose, and Xiufen Liu (FhG-IPK), Johan Everts, Maarten Jacobs (RUG), Hubert Voogd, Martijn Beenders (NICI), Stefan Giesler (SWE), Werner Kuckuck, Axel Kerckhoff (BKA), Gerhard Grube, Reinhard Zschach (LKA Berlin).

This work was sponsored by the Bundeskriminalamt (the German Federal Police Bureau), section handwriting examination (B2.30-2212/02).

References

- [1] 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.
- [2] eXtensible Markup Language (XML). <<http://www.w3.org/XML/>>, 2003.
- [3] eXtensible Stylesheet Language (XSL) - W3C Recommendation. <<http://www.w3.org/TR/xsl/>>, 2001.
- [4] B.J. Found. *The Forensic Analysis of Behavioural Artefacts: Investigations of Theoretical and Analytical Approaches to Handwriting Identification*. PhD thesis, LaTrobe University Bundoora, 1997.
- [5] K. Franke, I. Guyon, L.R.B. Schomaker, and L.G. Vuurpijl. *WandaXML - A data standard for the annotation and storage of handwriting samples in the context of (computer-based) forensic handwriting analysis and writer identification*. International Unipen Foundation, <<http://unipen.nici.kun.nl/>>, 2003.
- [6] K. Franke and M. Köppen. A computer-based system to support forensic studies on handwritten documents. *International Journal on Document Analysis and Recognition*, 3(4):218–231, 2001.
- [7] M.R. Hecker. *Forensische Handschriftenuntersuchung*. Kriminalistik Verlag, 1993. in German.
- [8] Ink Markup Language (InkML) - W3C Working Draft. <<http://www.w3.org/TR/InkML/>>, 2003.
- [9] L. Michel. *Gerichtliche Schriftvergleichung*. De Gruyter, 1982. in German.
- [10] MySQL. <<http://www.mysql.com/>>, 2003.
- [11] ParcPlace-Digitalk, Inc. *VisualWorks - User's Guide, Rev. 2.0*. 999 East Arques Avenue, Sunnyvale, CA 94086-4593, 1995.
- [12] 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.
- [13] PHP Hypertext Protocol (PHP). <<http://www.php.net/>>, 2003.
- [14] PostgreSQL. <<http://www.postgresql.org/>>, 2003.
- [15] 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.
- [16] Simple Object Access Protocol (SOAP) - W3C Note. <<http://www.w3.org/TR/SOAP/>>, 2000.
- [17] C. Taubenheim and K. Franke. *Programmers Handbook WANDA Client*. Fraunhofer Institute for Production Systems and Design Technology (IPK), Section Security and Testing Technology, Pascalstr. 8-9, 10587 Berlin, Germany, 2003.
- [18] M. van Erp, L.G. Vuurpijl, K. Franke, and L.R.B. Schomaker. The WANDA Measurement Tool for forensic document examination. In *Proc. International Conference of the Graphonomic Society (IGS)*, Scottsdale, Arizona, USA, 2003.
- [19] C. Veenhuis and K. Franke. *Programmers Handbook WANDA Server*. Fraunhofer Institute for Production Systems and Design Technology (IPK), Section Security and Testing Technology, Pascalstr. 8-9, 10587 Berlin, Germany, 2003.
- [20] L.G. Vuurpijl and L.R.B. Schomaker. Multiple-agent architectures for the classification of handwritten text. In *Proc. 6th International Workshop on Frontiers in Handwriting Recognition (IWFHR)*, pages 335–346, Tadjon, Korea, 1998.
- [21] XSL Transformations (XSLT) - W3C Recommendation. <<http://www.w3.org/TR/xslt/>>, 1999.