HandSpy – a system to manage experiments on cognitive processes in writing

Carlos Monteiro[1] and José Paulo Leal[2]

1 **CRACS & INESC-Porto LA, Faculty of Sciences, University of Porto**
Porto, Portugal
carlosmonteiro@dcc.fc.up.pt

2 **CRACS & INESC-Porto LA, Faculty of Sciences, University of Porto**
Porto, Portugal
zp@dcc.fc.up.pt

Abstract

Experiments on cognitive processes require a detailed analysis of the contribution of many participants. In the case of cognitive processes in writing these experiments require special software tools to collect gestures performed with a pen or a stylus, and recorded with special hardware. These tools produce different kinds of data files in binary and proprietary formats that need to be managed on a workstation file system for further processing with generic tools, such as spreadsheets and statistical analysis software. The lack of common formats and open repositories hinders the possibility of distributing the work load among researchers within the research group, of re-processing the collected data with software developed by other research groups, and of sharing results with the rest of the cognitive process research community.

This paper presents HandSpy, a collaborative environment for managing experiments in cognitive processes in writing. This environment was designed to cover all the stages of the experiment, from the definition of tasks to be performed by participants, to the synthesis of results. Collaboration in HandSpy is enabled by a rich web interface developed with the Google Web Toolkit. To decouple the environment from existing hardware devices for collecting written production, namely digitizing tablets and smart pens, HandSpy is based on the InkML standard, an XML data format for representing digital ink. This design choice shaped many of the features in HandSpy, such as the use of an XML database for managing application data and the use of XML transformations. XML transformations convert between persistent data representations used for storage and transient data representations required by the widgets on the user interface. This paper presents also an ongoing use case of HandSpy where this environment is being used to manage an experiment involving hundreds of primary schools participants that performed different tasks.

1998 ACM Subject Classification J.4 Social and Behavioral Sciences: Psychology

Keywords and phrases InkML, collaborative environment, XML data processing

Digital Object Identifier 10.4230/OASIcs.SLATE.2012.123

1 Introduction

Writing is a basic tool for a successful personal and academic growth. Given the importance of this subject social scientist are actively researching the cognitive processes involved in writing. As the goal of this research is in general to determine the factors that influence a development of writing skills, the participants are normally school children. The object of these research studies are writing productions on different tasks such as narratives, copies, dictations and alphabet transcriptions.

1st Symposium on Languages, Applications and Technologies (SLATE'12).
Editors: Alberto Simões, Ricardo Queirós, Daniela da Cruz; pp. 123–132

OpenAccess Series in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

The tools used for researching cognitive processes in writing are usually composed of two components: an hardware component capable of recording writing gestures and a piece of software to analyze the data. These tools use proprietary data formats for the collected data that hinders the possibility of sharing collected data to be analyzed in other systems or for other purposes. Moreover, the data collected with these type of hardware typically generates several files for each participant, on each task. Due to the large quantity of participants the amount of generated data files is considerable, hence managing data files is a major problem in this type of experiments.

This paper describes HandSpy, a new web based system offering typical features of cognitive processes research tools, but innovating on the experiment management. By providing a repository , researchers can set up an experiment for storing all the entities involved, such as tasks definitions, trait information on participants involved in the experiment and the generated data. The system follows a paradigm of collaborative experiments where various researchers can work on the same experiment simultaneously. HandSpy uses a standard XML[5] format for data files which enables users to collect data from various hardware devices as long as the generated files are kept in this format.

The analysis process differs also from the other existing systems. Instead of inputting the analysis results of each writing production on an external statistical analysis suite to generate results, HandSpy uses a selection engine to define a pool of productions bounded by a set of characteristics, the selected productions are subjected to an analysis process and the merged results of the selection can be exported.

The paper is organized as follows. Section 2 describes technologies used to develop HandSpy and outlines the main characteristics of two tools designed to study handwriting processes. Section 3 is a main description of HandSpy design and implementation methods. Section 4 is a description of a case study conducted with HandSpy and describes a data collection platform used to record data. Section 5 concludes this paper and sets the next steps for HandSpy improvement.

2 Related Work

This section covers background topics related to the development of a collaborative environment for managing experiments on cognitive processes in writing. The proposed environment is based on the use of the InkML [6] data format thus the first subsection is devoted to this standard. To the best of the authors knowledge, no collaborative environment with goals similar to those of HandSpy was ever described in the literature. However, there are some tools for collecting and processing gesture data for experiments in writing. The second subsection describes two tools to which HandSpy owes credit. The final subsection provides an overview on two types of components used in the development of HandSpy, namely the XML database and web interface toolkit.

2.1 InkML

The recent trend of sketching or writing on digital devices capable of recording hand gestures created the need for a standard to describe this kind of data. InkML is a W3C recommendation for storing and exchanging what is commonly called *digital ink*. It is an XML data format to describe a set of strokes digitally representing handwriting and other ink input gestures.

The ink in InkML is defined by characteristics associated with the act of creating a trace such as the width and color of the trace, the pen orientation while writing, the pen distance

Listing 1 InkML "hello" example.

```
<ink xmlns="http://www.w3.org/2003/InkML">
 <trace>
    10   0,  9  14,  8  28,  7  42,  6  56,  6  70,  8  84,  8  98, 8 112,
     9 126, 10 140, 13 154, 14 168, 17 182, 18 188, 23 174, 30 160, 38 147,
    49 135, 58 124, 72 121, 77 135, 80 149, 82 163, 84 177, 87 191, 93 205
 </trace>
 <trace>
   130 155, 144 159, 158 160, 170 154, 179 143, 179 129, 166 125, 152 128,
   140 136, 131 149, 126 163, 124 177, 128 190, 137 200, 150 208, 163 210,
   178 208, 192 201, 205 192, 214 180
  </trace>
  <trace>
   227  50, 226 64, 225 78, 227 92, 228 106, 228 120, 229 134, 230 148,
   234 162, 235 176, 238 190, 241 204
  </trace>
  <trace>
   282  45, 281 59, 284 73, 285 87, 287 101, 288 115, 290 129, 291 143,
   294 157, 294 171, 294 185, 296 199
  </trace>
   <trace>
   366 130, 359 143, 354 157, 349 171, 352 185,  359 197, 371 204,
   385 205, 398 202, 408 191, 413 177, 413 163, 405 150, 392 143, 378 141
   </trace>
</ink>
```

to the surface (whether the trace was made with the pen down or hovering the writing surface), among others.

A set of traces can be grouped in a context defining optional features such as starting time, writing surface dimensions and characteristics of the trace. The Listing 1 is an example of the "Hello" word described in a basic InkML file. The word has five letters represented with five trace elements, the values are separated by commas and represent the coordinates (X,Y) of every point to draw each letter.

2.2 Hand Gesture Collecting Tools

The two main tools currently available to research on cognitive processes in writing are Eye And Pen[3] and Ductus[8]. The use of these tools to conduct experiments involves two main steps: data recording and data processing. Despite some differences on the data recording step, both software solutions provide an interface to predefine settings to direct calculations. The information generated by these software tools is focused on the complementary concepts of burst and pause. A *burst* is a time span in which text was produced without interruptions. A *pause* is non-writing time span between two writing bursts.

Both software tools generate information about the pauses such as position in text and duration time. Document overall kinematic information on the written data such as duration of the experiment, velocity and writing distance is also displayed. The calculations can be exported in a spread sheet format.

The two systems differ in the way they collect gesture data. The Ductus software uses a digitizer to record the pen position and pressure. The Eye and Pen software uses a digitizing tablet and an eye tracker to synchronously record the pen position and the eye point of regard on the tablet, correlating the eye and pen movement during bursts and pauses. The user interface provides a player that displays the written text and eye gaze point recorded during the text production time. The user interface displays also bounding boxes surrounding each possible word. These bounding boxes are computed from the distance between traces using

a threshold that can be customized for each case. Each identified word can be associated with an ASCII string and extra information about it can be added.

2.3 XML Databases

With the popularity of XML use as a transactional data format, there was a natural increase in the use of XML databases. There are two types of XML databases, the XML enabled and the native XML databases.

The XML enabled databases follow a structure of a relational database. The structure of the XML file can be used to transform it into a common relational table set. Systems like Oracle object-relational DBMS can store XML files in XMLType which is a specific type to store XML files with built-in function that allow the management of the content in the files.

On other hand native XML databases can store the actual XML files without following any schema. The XML structured files enable the database to perform indexing which is the key for efficient queries in these systems. Most native XML databases have a set of built in functionalities that enable XML files management and database access.

The following paragraph has a description of some technologies embedded in most native XML databases. XQuery[4] is a language used to query XML databases, the advantage of using XQuery is the possibility of creating result sets based on the contents of the XML file. The XQuery Java API (XQJ) permits to create and submit XQuery expressions to the database and use the results set as Java objects. The XML Database[1] (XML:DB) API allow the management of XML databases disregarding the database system, this permits to create an environment capable of working and manage several databases.

There are several XML database systems distributed under commercial and free software license. Under commercial license the most complete systems are Oracle XML DB, Documentum xDB and MarkLogic Server. All these systems offer basic tools for querying using XPath, XQuery and support for XSL transformations. On free systems the most complete are BaseX, eXist[2] and Sedna. These systems use the same basic tools as commercial products for querying but the most complete system is eXist offering support for XML:DB API and XQuery Java API.

2.4 Javascript Frameworks

Developing web pages using JavaScript frameworks became a common practice due to the large set of pre-written widgets and functions that are offered. These frameworks enable developers to focus on the creation of new cross browser interactive contents. There are several JavaScript frameworks such as Dojo, jQuery, SmartClient and Prototype, these frameworks apart from design differences offer almost the same obligatory features which are described in the following paragraph.

To simplify the access to methods of Documents Object Model (DOM) objects they are wrapped into new objects with simple access methods. Offering a large set of customizable widgets tied to data sources, simplifies data bindings with the server.

3 HandSpy

HandSpy is a web based application to manage distributed and collaborative experiments on cognitive processes in writing. This system has the following distinctive features:

- an experiment management philosophy encompassing all the steps of the research in cognitive processes in writing;

- a multiuser web interface fostering collaboration among researchers and enabling remote work on the experiments;
- a cloud-based data management system providing central storage for all data collected in the experiments;
- an analysis process of the collected data, inspired in the state-of-art systems described in Section 2;
- the ability to select and synthesize collections of data based on different criteria;
- the use of standard XML based data formats to promote interoperability and cooperation among researchers in this community.

HandSpy follows a 3-tier architectural model as depicted in Figure 1 where the presentation tier (a web interface) is represented by the left box, the logic tier (a web server) is represented by the central box and the data tier (an XML database) is represented by the right box. This diagram represents also in three rows the data flows between these tiers.

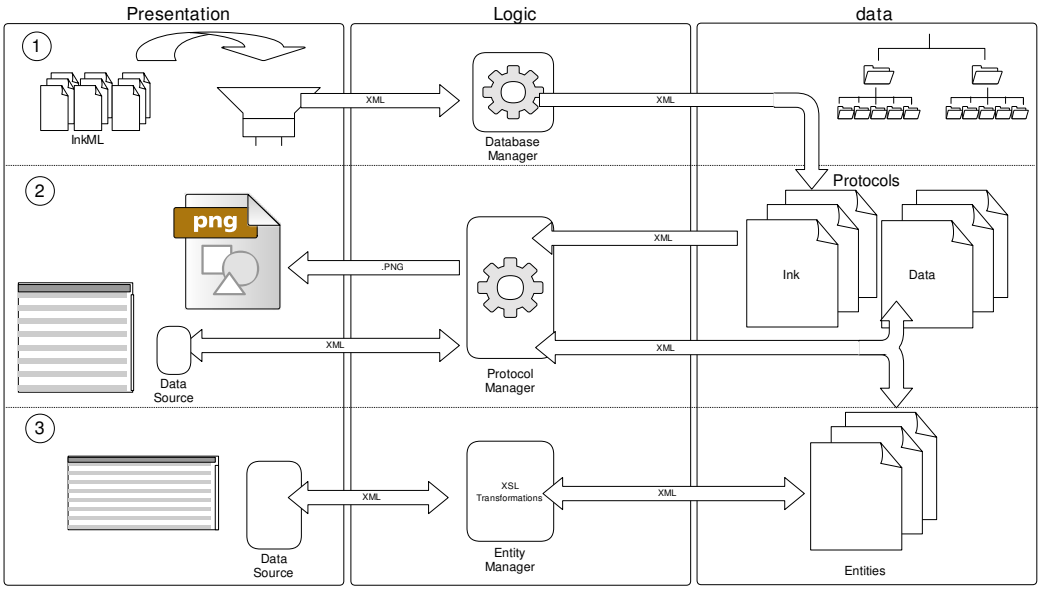

Figure 1 HandSpy application architecture.

In the top row marked with number one is represented the process of uploading data files in InkML format to the database through the web interface. On the server side the database manager module is responsible to organize the uploaded files to the respective collections based on the current user context.

The middle row represents the interaction with ink data. The two main components of HandSpy user interface are depicted in this row, an image viewer to display the written production of the protocol ink and a list grid to display calculations based on the protocol data. The server gets the ink of the selected protocol and generates an image file to feed the image viewer. The list grid is populated with a data source document containing the calculations results generated following the predefined settings on the interface. To optimize the system the main definitions on the HandSpy are classified and treated as entities. This generalization of data permits to manage it in the same way. All data showing objects are based on list grids which use XML data sources. In the third row of the model in

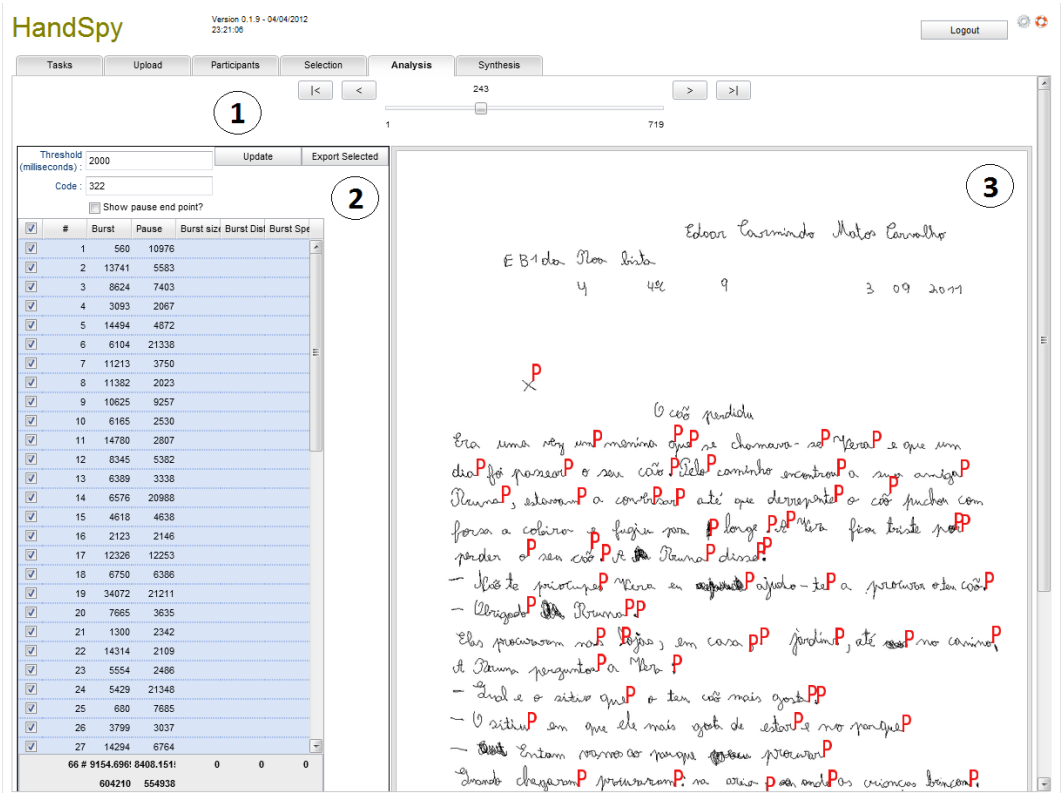

Figure 2 HandSpy interface: section 1 - slider to navigate through selection; section 2 - list with pauses duration, bursts duration, burst size (number of words in the burst), burst distance and burst average speed; section 3 - widget to display the protocol image with the selected pauses identified with a red P.

the Figure 1 is shown the Entity Manager that identifies the entity and uses the respective XSL transformer to transform the data stored in the data base into the client specific data source when the fetch operation is made. The operations of adding, updating and deleting entities entries use the correspondent XSL transformations to perform the operations and save the changes to the data base.

3.1 Design

3.1.1 Application Interface

The graphical user interface of HandSpy relies on a web application. The workspace is divided in six tabs covering the usual flow of an experiment in on cognitive processes in writing.

Tasks Identification of tasks to be performed by the participants during the experiment. For instance, an experiment may include a task where participants must write as much letters of the alphabet as they can in a fixed amount of time.

Upload Upload of the InkML files collected with specialized hardware (smart pens or digitizing tablets) to the system. At this stage the InkML data is connected with a task and a participant. The interface displays a collections of thumbnail images of the

uploaded files and allows the selection of particular file that is displays in its real size for better identification.

Participants Manage the participants in the current experiment. Display the features and the tasks completed by each participant. Custom features describing the participants, such as handiness or mother language, can be added to the participants. The participants features are useful for selecting them to a particular study. The list of participants can be imported and exported as a CSV (Comma Separated Values) file.

Selection Selection of protocols based on tasks and on features of the participants such as age, handedness and gender. The selection is a collection of conditions on protocols to be analyzed and synthesized. Selections set by different researchers are independent from each other, enabling researchers to analyze different collections of protocols simultaneously.

Analysis Figure 2 is a screenshot of HandSpy interface with the Analysis tab selected. The area identified as 1 is a slider to browse the current of protocol selection (set in the Selection tab). Area 2 has a form to define the parameters to calculate the pauses which are listed in the table below. Each row has a pause duration, a burst duration, a burst size is a number of words present on the burst, burst distance and the burst average speed. The footer of the table presents statistics on some of its columns, such as the average and standard deviation of durations, and the count of words. Area 3 displays the written production with red Ps (for Pause) marking the place where the pauses selected in area 2 start. Pause selection allows worthless parts (for instance, a part where the participant erased a word) to be removed from the analysis.

Synthesis Displays global statistics on the data of processed on the Analysis tab and bounded by the selection criteria. The statistics presented in Analysis tab table footer for each protocol are computed on this tab aggregating all the selected protocols. These results can be exported to other systems, such as spreadsheets or statistical analysis packages.

3.1.2 Data Repository

HandSpy processes data uploaded in XML files and stores it in a native XML database. The database structure model is represented in the Figure 3. Every experiment has a set of entities that store data about Tasks, Participants, Selection and Configurations. The data files containing the text production of one experiment are stored in the collection Ink in InkML format and remain unchanged. For every task added to the project a task collection is created to store data files containing calculations and other related data.

3.2 Implementation

As depicted in Figure 1, HandSpy is a composed of a presentation, logic and data layers. The presentation layer was implemented on a SmartClient JavaScript framework. The logic layer was implemented on the Tomcat servlet container and the data layer on the eXist XML database. The remainder of this subsection presents implementation regarding the use of these components.

The Isomorphic SmartClient LGPL platform[1] provided the web toolkit for the user interface. SmartClient provides sophisticated table editing widgets connected to data sources in XML formats that are appropriated to the data handled in HandSpy. These widgets have many built-in functions, such as sorting and grouping on every column, search fields

[1] `http://www.smartclient.com/`

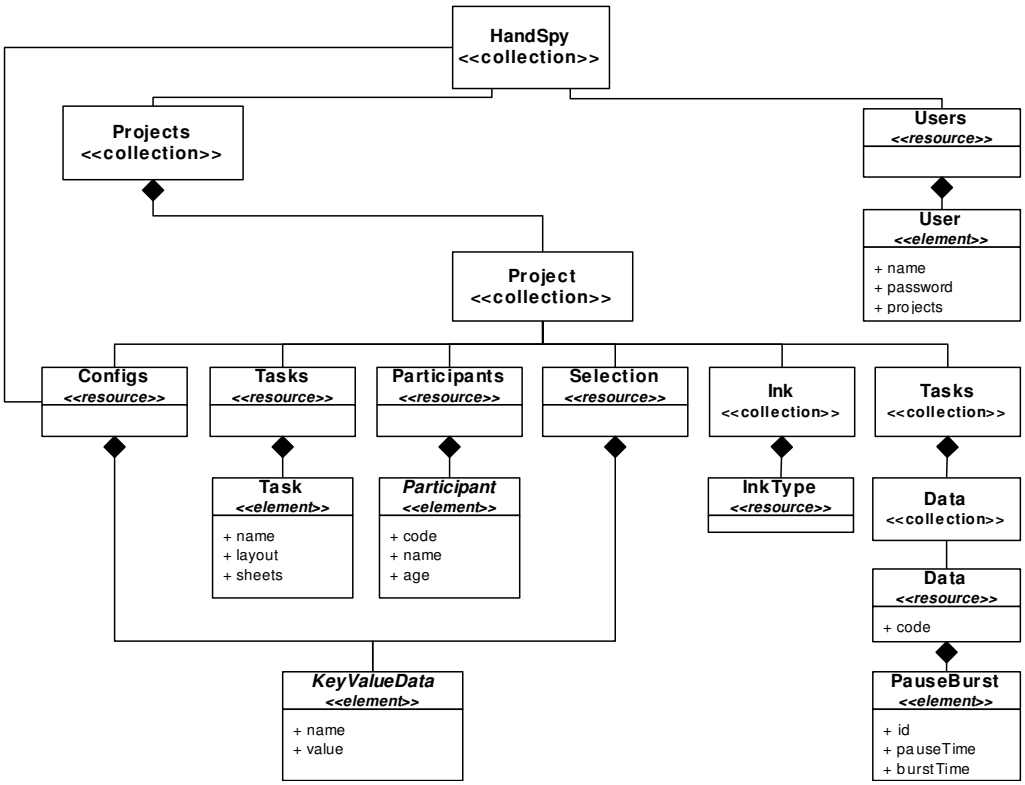

Figure 3 Database structure diagram.

and column customization. Data operations, such as fetching or querying, are built-in functions the data source object. Protocol images on different sizes are generated in the server is displayed in HTML canvas object and are embedded in a SmartClient HTML pane component. This enables the use of a native HTML control on the canvas for managing overlay image objects on the original image.

The server was developed on Tomcat[2] - a Java Servlet container to interact with the client interface and the data storage. For protocol image generation the InkML files are converted into Java objects to produce image files. For feeding SmartClient data sources the XML files in the database are converted into the XML format required by SmartClient data sources. The server runs a set of modules to manage the system some of the most important are listed below.

- `Process` is a Java Servlet, the only outside entry point in the system. Process the requests from the client and generate the responses.
- `DBconnection` is a singleton module responsible for making the connection with the data base using XML:DB API. The database system is implemented using eXist.
- `Selector` creates a XQuery expression from the user selection criteria to create a collection of protocols to be analyzed in the system.

[2] `https://tomcat.apache.org/`

- `EntitiesManager` uses XSL transformations to manage data transferences between the data base and the application client.

The database was implemented using eXist[3] database management system. The communication between the server and the database is made using XML:DB API. The implementation leverages on XML to create RESTful services that provide data sources to the web interface. The XML format required by data sources of SmartClient widgets is obtained by converting files stored in eXist using eXtensible Stylesheet Language Transformations (XSLT)[7]. Other uses of data stored in the database rely on the Java Architecture for XML Binding (JAXB)[9]. This API serializes Java Objects into XML files and vice versa, this enables total control for creating and editing XML files.

4 Case Study

HandSpy is being used to manage an experiment on cognitive processes in writing. This study is targeted to students in grades 2 to 7, totaling 560 children carrying out five different writing tasks. The data collection was made in the classroom with fifteen students at a time.

The device used to collect the data for this experiment was the Livescribe[4] smart pen. This smart pen is similar to a normal pen and has an infra-red camera capable of determining and recording its position over time on a special paper. This paper has a "patterns" of micro-dots that the camera is able to detect and use to determine its exact position.

There are several advantages of using smart pen to replace the digitizing tablets traditionally used for this kind of experiment. Firstly, the simplicity of setting up an experiment in a classroom, a place already familiar to the participants. Secondly, the fact of being a writing device similar to the pens normally used by your children in school. These features make the smart pen less intrusive that digitizing tablets. The cost of running the experiment with smart pens is also a relevant. The price of single digitizing tablet is equivalent to several pens, they are easy to carry, and a single researcher can supervise several participants at once.

The smart pen run a piece of software — a *penlet* – specially developed for this type of experiment. The penlet is an component developed using the Livescribe Java API which is based on the Java Micro Edition. The HandSpy penlet life cycle starts when the tip of the pen touches the HandSpy paper application and `initApp()` method is invoked. All the strokes events raised by the pen on this paper application are processed and additional information about the time of each point in the stroke is serialized to the penlet internal memory pool. The penlet life cycle ends when the pen is shut-down and destroy() method is invoked.

The sheet of paper with a micro-dots used by the pen to obtain its current position is called a *paper application*. The paper application is an Anoto Functionality Document (AFD) which has digital information about the physical paper such as active regions to raise events on the penlet. For this experiment were used three different page layouts to define each task using the same AFD. The main functionality of this paper application is added by an active region to define the start and end time of the narrative task.

Data stored in the smart pen is retrieved the HandSpy DataCapture application. This application was developed a C# using the Livescribe Desktop SDK. It is designed to browse

[3] `http://exist-db.org/`
[4] http://www.livescribe.com/

the Smart Pen AFD collection and use the additional data files, stored with the use of the HandSpy Penlet, in the memory of the Smart Pen to create an InkML file. DataCapture starts when the pen is attached to the USB powered connection dock. At this stage the collected data has already been uploaded to HandSpy and the association with the participants who produced the written data is done. The data is currently being analyzed.

5 Conclusions and Future Work

With the use of new devices capable of recording hand gestures, the use of digital handwriting as a transferable data is becoming more common. These new possibilities rise new ways of studying the cognitive processes involved in the handwriting process. In this paper is described the design and implementation of a new tool to support the study on cognitive processes in writing, HandSpy. This tool aims to extend the experience of conducting a study where large amounts of data need to be manage and where collaborative work speed up the analysis process of this data. Using a web platform, HandSpy is a powerful tool to be used as a cross platform environment and brings the cognitive study experience to a cloud environment.

HandSpy is currently being used to manage an experiment under a research project of the Faculty of Psychology and Educational Sciences of the University of Porto, successfully storing over a thousand written productions and related data. As future work we aim to improve the user experience by implementing missing features such as system management configurations, expand selection parameters, synthesis engine, written productions replay controls and other features that may arise with the current usage on the study case.

Acknowledgments This work is in part funded by the ERDF/COMPETE Programme and by FCT within project FCOMP-01-0124-FEDER-022701. The authors wish also to thank the reviewers for their helpful comments.

References

1 Application Programming Interface for XML Databases (XML:DB API). http://xmldb-org.sourceforge.net/.
2 eXist-db Open Source Native XML Database, 2011. http://exist-db.org.
3 D. Alamargot, D. Chesnet, C. Dansac, and C. Ros. Eye and pen: A new device for studying reading during writing. *Behavior Research Methods*, 2006.
4 S. Boag, D. Chamberlin, M. F. Fernandez, D. Florescu, J. Robie, and J. Simeon. XQuery 1.0: An XML Query Language, 2010. http://www.w3.org/TR/xquery/.
5 T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, and F. Yergeau. Extensible markup language(XML)., 2008. http://www.w3.org/TR/xml/.
6 Y. Chee, K. Franke, M. Froumentin, S. Madhvanath, J. Magana, G. Pakosz, G. Russell, M. Selvaraj, G. Seni, C. Tremblay, and L. Yaeger. Ink Markup Language (InkML), 2011. http://www.w3.org/TR/InkML/.
7 J Clark. XSL Transformations (XSLT), 1999. http://www.w3.org/TR/xslt.
8 E.Guinet and S. Kandel. Ductus: A software package for the study of handwriting production. *Behavior Research Methods*, 2010.
9 E. Ort and B. Mehta. Java Architecture for XML Binding (JAXB), 2003. http://www.oracle.com/technetwork/articles/javase/index-140168.html.