

Writing and Speech Recognition

Observing Error Correction Strategies
of Professional Writers

Published by
LOT
Janskerkhof 13
3512 BL Utrecht
The Netherlands

phone: + 31 30 253 60 06
fax: + 31 30 253 60 00
e-mail: lot@let.uu.nl
<http://www.lotschool.nl>

Interior lay-out & type-setting: Mariëlle Leijten
Illustrations: Mariëlle Leijten & Yvonne Wanders

ISBN 978-90-78328-31-5
NUR 616

Copyright © 2007: Mariëlle Leijten. All rights reserved.

Writing and Speech Recognition

Observing Error Correction Strategies of Professional Writers

Schrijven en spraaktechnologie
correctiestrategieën van professionele schrijvers

(met een samenvatting in het Nederlands)

Proefschrift

ter verkrijging van de graad van doctor aan de Universiteit Utrecht op gezag van
rector magnificus prof. dr. W.H. Gispen, ingevolge het besluit van het college voor
promoties in het openbaar te verdediging op vrijdag 22 juni 2007
des middags te 4.15 uur

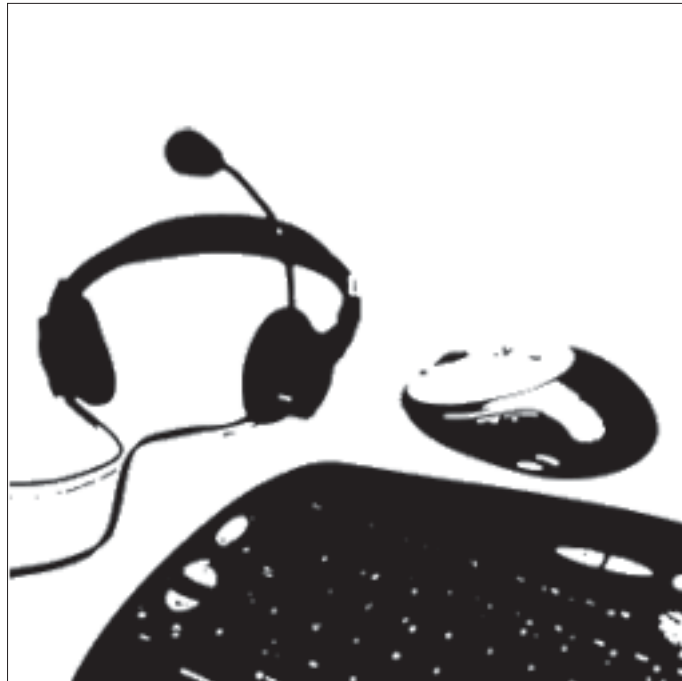
door

Maria Adriana Johanna Catharina Leijten

geboren op 24 mei 1975 te Raamsdonk

Section IV

Logging writing processes in a Windows environment



8

Inputlog

A research tool for observing and analyzing multimodal writing processes in a Windows environment

Abstract: Not only has the use of computers as writing instruments had a profound effect on the writing practice and the attitudes towards writing, it has also created new possibilities for research on writing. In the field of cognitive writing research especially, keystroke logging programs have become very popular. In this chapter we describe a logging program called Inputlog. Inputlog 2.0 Beta consists of four modules: (1) a data collection module that registers digital writing processes on a very detailed level; (2) a data analysis module that offers basic and more advanced statistical analyses (e.g. text and pause analysis); (3) an integration module that allows data merging between data files; (4) a playback module that enables researchers to review the writing session. In this chapter we describe the technical and functional characteristics of Inputlog 2.0 Beta and give advice on applying Inputlog as a research tool. We conclude the paper with a preview of the plans for further developments.

Keywords: Inputlog, keystroke logging, registration tool, on-line writing processes, pauses, writing modes, text analysis, pause analysis, research methods, writing observation, cognitive processes.

This chapter has been elaborated from Leijten & Van Waes (2006). Inputlog: New perspectives on the logging of on-line writing processes in a Windows environment. In K. Sullivan & E. Lindgren (Vol. Eds.) & G. Rijlaarsdam (Series Ed.), Studies in Writing: Vol. 18. Computer Key-Stroke logging and Writing. Methods and Applications (pp. 73-94). Oxford: Elsevier; Van Waes, L., & Leijten, M. (2006). Logging writing processes with Inputlog. In L. Van Waes, M. Leijten, & C. Neuwirth (Vol. Eds.), & G. Rijlaarsdam (Series Ed.), Studies in Writing: Vol. 17. Writing and Digital Media (pp. 158-166). Oxford: Elsevier; Van Waes, L., & Leijten, M. (2006). Schrijfprocessen registreren met Inputlog. Een data-analyse van de interactie met de 'reeds geproduceerde tekst' [Observing Writing Processes with Inputlog: A Data Analysis of the Interaction with the Text Produced so Far]. Tijdschrift voor Taalbeheersing, 28(2), 198-219.

1 Introduction

As most writers nowadays produce nearly all of their texts on a word processor, the computer has not only become the major writing instrument, its use as a research tool has also increased. The computer enables researchers to collect detailed information about the writing process that were hardly accessible before. Or, as Spelman, Miller and Sullivan (2006) state:

As an observational tool, keystroke logging offers the opportunity to capture details of the activity of writing, not only for the purposes of the linguistic, textual and cognitive study of writing, but also for the broader applications concerning the development of language learning, literacy, and language pedagogy. (p. 1)

In this day and age, researchers make frequent use of keystroke logging tools to describe online writing processes in detail. These logging programs enable researchers to exactly register and accurately reconstruct the writing processes of writers who compose texts at the computer. The basic concept of the different logging tools that have been developed is more or less comparable. First, the keystroke logging tools register all keystrokes and mouse movements. During the writing process these basic data are stored for later processing. This continuous data storage does not interfere with the normal use of the computer, creating an ecologically valid research context. At a later stage, the logged data can be made available for further analysis, either within the program environment itself or as exported data in statistical programs such as SPSS or SAS. Depending on the research question, researchers can choose to analyze different aspects of the writing process and the writing behavior by combining, for instance, temporal data (e.g. time stamps or pauses) with process data (keystrokes or mouse movements). Computer-based data collection (and processing) is much faster and more accurate than manual data collection.

Depending on the research question, researchers can study different aspects of the writing process or the writing behavior by combining temporal data (e.g. absolute time and pauses) and writing process data (keystrokes and mouse movements) for analyses. In this chapter we describe Inputlog, a logging tool for writing process research developed for Windows environments. First, we present the most important characteristics of Inputlog 2.0 Beta. Then we give a more detailed description of the technical and functional characteristics of Inputlog. Next, we elaborate on the applications of Inputlog. To conclude we preview the further developments of the program.

The chapter describes Inputlog in great detail. Readers who are mainly interested in the general characteristics of Inputlog should read section 2, 4 and 5. Readers who are interested in the technical aspects of Inputlog should also read section 3. Readers who already are familiar with Inputlog, and are interested in specific techniques of data analyses find more information on the topic in section 6.

2 Characteristics of Inputlog

For the development of Inputlog we were able to fall back on the functionality of two existing programs: JEdit and Trace-it on the one hand (Kollberg, 1998; Severinson Eklundh, 1994; Severinson Eklundh & Kollberg, 1992, 1996, 2003; Spelman Miller & Sullivan, 2006), and ScriptLog on the other hand (Strömquist & Karlsson, 2001; Strömquist, Holmqvist, Johansson, Karlsson, & Wengelin, 2006). Both have serious limitations. JEdit and Trace-it are designed for Macintosh personal computers. JEdit only logs data in an in-house developed, limited word processor. ScriptLog also mainly logs in a limited word processor that was developed for research purposes (i.c. mainly writing experiments with young children). Trace-it features an extended interactive revision module, while ScriptLog is the first program that combines logging data with recorded eye-tracking data.

Most logging-tools are either developed for a specific computing environment, or not adequately adapted to the current Windows environment¹. As such, they cannot be used for writing studies in which 'natural' writing and computer networks employ commercial word processors (e.g. Microsoft Word or WordPerfect). This discrepancy was one of the main reasons for deciding to start with the development of Inputlog in 2003.

Another impetus for the development of Inputlog has been the emergence of speech recognition as a new writing mode for word processors. In our first research study on the influence of speech recognition on the writing process we analyzed the writing process data manually (Leijten & Van Waes, 2003b; 2005b, see also chapters 2, 3 & 4). At that moment it was impossible to register keyboard input in combination with speech mode data with any of the existing logging tools. To collect the writing process data, we combined two digital observation instruments: a digital screen cam (i.c. Camtasia)² and a digital sound recorder (i.c. Quickrecord)³. The study showed that although the chosen observation instruments and analyzing methods did enable us to analyze and describe the specific speech recognition writing processes, the data analysis was very time-consuming. In follow-up experiments we registered and analyzed these multi-modal data with Inputlog (Leijten, Janssen, & Van Waes, in preparation; Leijten, Ransdell, & Van Waes, submitted; also chapters 5, 6 & 7).

In sum, Inputlog allows researchers to:

- record (keyboard, mouse and speech) data of a writing session in Microsoft Word, I-pad and other Windows based programs (cf. section 4.2);
- generate data files for statistical, text, pause and mode analyses (cf. section 4.3);
- integrate various types of data from other programs (cf. section 4.4);

¹ In June 2006 the company Noldus Information Technology released a keystroke logging program called Ulog that also registers in a Windows environment. The tool is an addition to The Observer software (www.noldus.com).

² Camtasia was renamed to Camtasia Studio. It is a commercial software package for recording, editing and sharing high-quality screen videos (more information: <http://www.techsmith.com>)

³ Quickrecord is a compact application for recording and playing sound on a Microsoft Windows PC (more information: <http://www.ptpart.co.uk/>)

- playback the recorded session at different speeds (cf. section 4.5).

The most distinguishing characteristics of Inputlog to date are its word processor independent functionality, the parsing technology, the standard XML structure of the output and the logging of speech recognition.

2.1 Word processor independency

Contrary to Trace-it and ScriptLog, Inputlog registers every keystroke and mouse movement independently of the word processor used. Inputlog is designed to log and analyze writing data produced in Microsoft Word. However, the program also logs keyboard and mouse actions in other Windows based programs⁴. In other words, not only writing processes as such can be observed with Inputlog, but also basic processes like consulting websites or programming in any Windows based language.

2.2 Parsing

Generally speaking, parsing refers to the process of analyzing an input sequence⁵. Inputlog logs the input of writing sessions and generates logging files that are used as input data for further analysis. After a writing session has been recorded, Inputlog generates different analyses (e.g. statistical, text, pause, and mode analyses) from the source logging file. The processing of input data depends on what has preceded, and even on what follows. As such, keeping track of these dependencies in order to verify how to process the data is called parsing. In order to facilitate the implementation of new functionalities and to better control program maintenance, we have opted to technically restructure version 2.0 of Inputlog by using a specific parsing technique: syntax directed parsing. Inputlog uses the tools Flex and Bison to implement the parsing (Aho, Sethi, & Ullman, 1986; Donnelly & Stallman, 1995; Levine, Mason, & Brown, 1992; Paxson, 1995). These parsing tools generate parsers in C++ code which are then integrated in the Inputlog program code.

The parsing techniques simplify the overall program by separating the input and processing components and by providing a natural, modular structure. Furthermore, by hiding the implementation details of the different analyses, one does not only get a more readable program structure, but one also obtains a framework in which it is possible to get the different analyses in just one or two passes. This makes the program considerably faster.

In *syntax directed parsing*, a standard algorithm automatically constructs the input part of a program from a high level description of the input data structure. Code to perform any required processing of the data is then attached to the description in a convenient way. This non-procedural description is generally easier to write and to

⁴ Researchers should keep in mind that for certain analyses (e.g. analyses on sentence or paragraph level) only the data logged in Microsoft Word can be used. For this kind of analyses the basic data are interpreted with a set of algorithms based on Microsoft Word.

⁵ We would like to thank Nico Verlinden for his effort in integrating Parsing in Inputlog.

modify than the equivalent program code, and much less likely to harbor bugs. It is also easier to read, and easier to maintain.

2.3 XML structure of output files

In the previous versions of Inputlog the output data were generated as Excel files. However, we have chosen to integrate the more universal XML structure in the current version of Inputlog. XML is the abbreviation of Extensible Markup Language, which allows users to define the tags (markup) that are needed to identify the data and text in XML documents.

```
<Event>
  <WritingMode>2</WritingMode>
  <Output>LeftButton</Output>
  <StartClock>0:00:16</StartClock>
  <StartTime>16672</StartTime>
  <EndClock>0:00:16</EndClock>
  <EndTime>16797</EndTime>
  <ActionTime>125</ActionTime>
  <PauseTime>16672</PauseTime>
  <X>252</X>
  <Y>55</Y>
</Event>
```

The advantage of XML is that researchers can easily adapt the data to their own research needs. Research data can be built in the same way as the Inputlog output and can then be easily integrated into one another. The example above shows how the XML structure of Inputlog is built. Each <tag> states the information between the tags: for example writing mode 2 is a mouse movement or click. In the future we are hoping to develop a standard for keystroke logging tools and exchange data XML structures that are compatible with other keystroke logging programs⁶. This would enable merging between XML structured data of various programs. We will give another example of these possibilities in more detail in the next section.

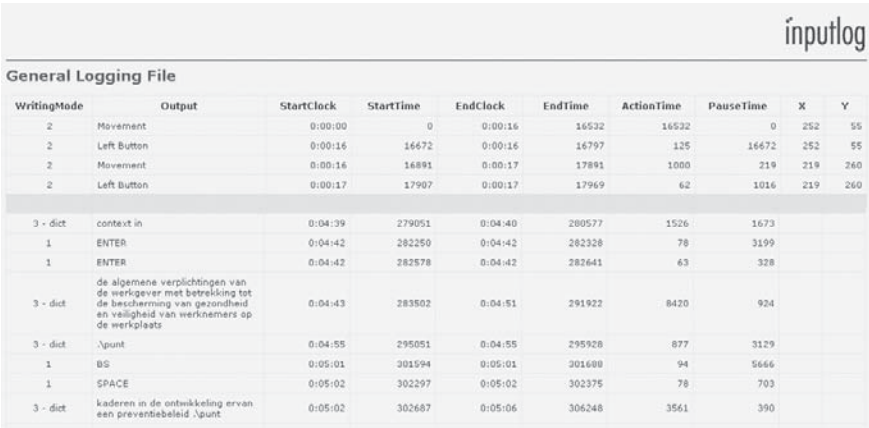
2.4 Speech recognition

As mentioned above, Inputlog is currently the only logging tool that can integrate the input of dictation devices using speech recognition software. Since the analyses of multi-modal speech recognition data turned out to be extremely laborious, computer assisted logging is very helpful in recording and analyzing writing sessions in which speech technology is used. For this reason we chose the most widely used speech recognition software, that is, Dragon Naturally Speaking. The specially designed logging add-on in the speech recognizer (in combination with a Python script) enabled us to integrate the dictated text with the data logged by Inputlog. Comparable to the general logging file generated by Inputlog, the logging file of Naturally Speaking relies on timestamps. Via a Python script that was developed for this purpose, we generated

⁶ Data collected with ScriptLog then becomes exchangeable with Inputlog.

an XML file of a recorded speech session that is basically structured in the same way as an Inputlog data file. By also converting the output files of Inputlog 2.0 Beta into XML, we were able to combine both logging files and integrate them into one general logging file based on the convergence of timestamps. The result is a single file that can be used for further analysis of multimodal writing sessions in which speech input is combined with keyboard & mouse.

Figure 1 shows a short XML excerpt of a logged writing process in Microsoft Word in which three writing modes were used (column 1: writing mode): keyboard (1), mouse (2), and speech (3 – dict).



WritingMode	Output	StartClock	StartTime	EndClock	EndTime	ActionTime	PauseTime	X	Y
2	Movement	0:00:00	0	0:00:16	16522	16522	0	252	55
2	Left Button	0:00:16	16672	0:00:16	16797	125	16672	252	55
2	Movement	0:00:16	16891	0:00:17	17891	1000	219	219	260
2	Left Button	0:00:17	17907	0:00:17	17969	62	1016	219	260
3 - dict	context in	0:04:39	279051	0:04:40	280577	1526	1673		
1	ENTER	0:04:42	282250	0:04:42	282328	78	3199		
1	ENTER	0:04:42	282578	0:04:42	282641	63	328		
3 - dict	de algemene verplichtingen van de werkgever met betrekking tot de bescherming van gezondheid en veiligheid van werknemers op de werkplaats	0:04:43	283502	0:04:51	291922	8420	924		
3 - dict	\punt	0:04:55	295051	0:04:55	295928	877	3129		
1	BS	0:05:01	301594	0:05:01	301688	94	5666		
1	SPACE	0:05:02	302297	0:05:02	302375	78	703		
3 - dict	kaderen in de ontwikkeling ervan een preventiebeleid.\punt	0:05:02	302687	0:05:06	306248	3561	390		

Figure 1. Example of a writing session with keyboard, mouse and speech recognition.

In this excerpt the writer dictates the next segments following each other closely (translated from Dutch):

04.39	<context>
04.43	<The general obligations of the employer related to the protection of the employees health at the working place.>
04.55	<full stop>
05.02	<fit in with the development of the prevention policy><full stop>

The words that are dictated with the DNS software are represented as segments. Each continuous dictation is considered a segment. For each segment a time stamp for the start and end times are given in a dual coding: in hours:minutes,seconds and in milliseconds. The key presses, mouse movements and mouse clicks are represented per row as their actual output or as a readable code for each action (e.g. BS refers to backspace). Next to this, the timestamps are shown for the start and end of each action: for example key press in and key press out, beginning and end of mouse movement and start and end of a dictated text segment. For mouse movements and clicks the x-and-y-values are represented.

These multimodal logging data enable us to study the hybrid character of this kind of ‘writing’ in which dictated segments alternate with keyboard based word processing. Inputlog thus analyzes mode switches between speech and keyboard, error correction in various writing modes or rates of productivity in both speech and keyboard writing. The implementation of speech recognition in Inputlog will stimulate research on the effect of this new technology on the writing process (of both professional writers and writers with learning disabilities). Moreover, we would also like to explore the logging possibilities of speech recognition to simultaneously transcribe thinking-aloud protocols and/or retrospective interviews (see section 5.3 further applications).

We have developed an interface in which researchers can select the analyses they need for a specific source file. This user-friendly interface allows researchers to adapt Inputlog to their research needs. This basic functionality is described in section 4. In section 3 we first give some more information on the technical background of Inputlog.

3 Technical description

In this section we describe Inputlog from a more technical perspective. Technical terminology will be further explained in the glossary (see Appendix 1). First, we illustrate the flow of the program. Then we describe the program structure. Finally we formulate several ‘best practices’.

Because of the various types of output files that can be generated by the program, Inputlog is able to log and analyze writing processes from different perspectives. When logging writing processes, Inputlog captures input data at a level before they are converted to screen information, namely:

- scan codes of keystrokes (e.g. scancode ‘12’ refers to the letter ‘e’);
- mouse activities (clicks, movements, location);
- time stamps of all input ‘events’ or ‘actions’.

These data are stored in so-called IDF-files (Inputlog Data Files) that are converted to different output files afterwards, preparing the rough data for qualitative and quantitative analyses (cf. functional description in section 4). Figure 2 visualizes the flow of the most recent version of the program: Inputlog 2.0 Beta.

In step 1 the process data of the online writing session is logged (cf. *supra*). In step 2 the logging data are saved in the IDF-file. In this source file each ‘event’ or action of the writing process is stored separately as binary data. The source file data can be used either as input for the analyses, or as input for the playback module (cf. *infra*). In step 3 the binary data of the source file are converted to text data.

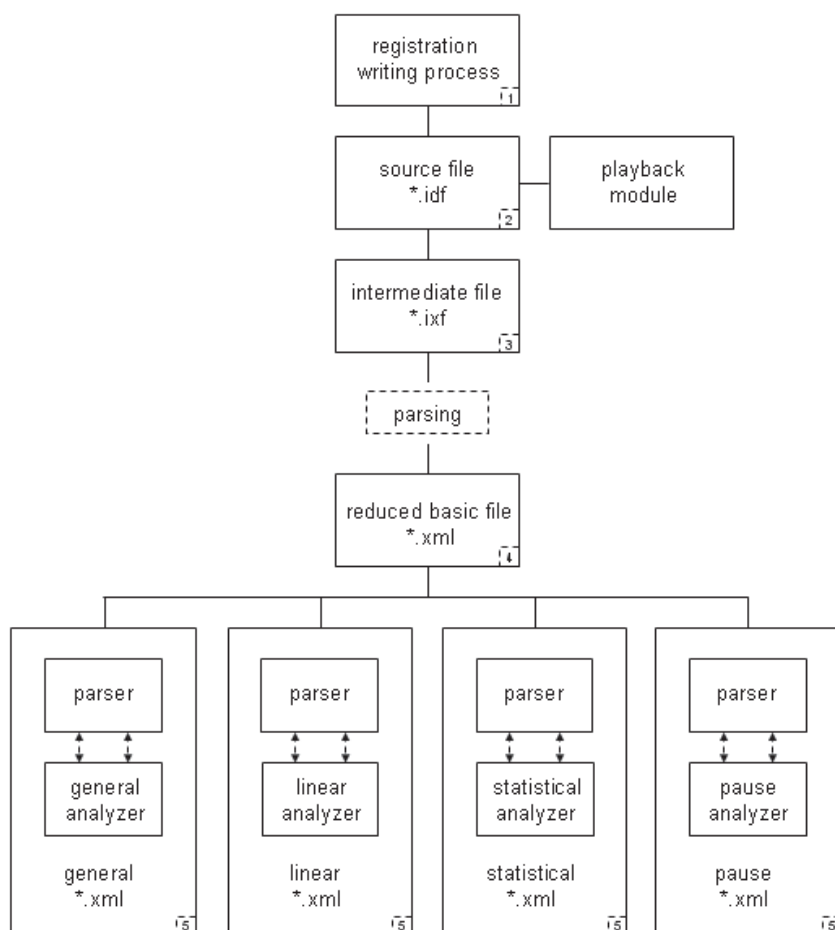


Figure 2. Basic flow of the logging and analysis conversion in Inputlog (without merging of speech).

This conversion is necessary because of the syntax directed parsing technique that we use (cf. supra). The logging data are parsed between step 3 and 4: the text data are parsed to a readable XML-format by combining and reducing data, for example mouse movements are reduced to a starting point and an end point. The reduced XML file is the starting point that enables researchers to analyze the modus data, the pause data and the text data on a more aggregated level in step 5. In this step the data are parsed by performing more specific rules on the data, for example via a set of rules which locate every pause on a specific text level (e.g. pause within words).

The source file (IDF) is also used as input for the play back module⁷. This enables researchers to replay the writing session exactly as it was registered, or speed up to the researcher's preference, for instance by reducing long pauses.

3.1 Programming language

Inputlog is mainly programmed in Visual C++.NET and the interface is programmed in Visual Basic.NET. We opted for C++ as a programming language because of its object orientedness and processing speed. Visual Basic.NET is the standard for interface development.

It is very important that the resident logging program does not interfere with the use of the word processor, and consequently, that it does not hinder the writer during his or her writing process in any way. Therefore logging and analyzing have been separated. Each part of Inputlog is programmed in different classes (see Figure 3). This allows us to easily update and debug the current analyses, and to further extend the functionality of the program. Inputlog runs on a PC with only one CPU and its use of system resources is quite limited, contrary to, for example, Java.

For the logging of the writing process Inputlog uses two Windows Hooks: Journalrecord and Journalplayback. Journalrecord registers every keystroke and mouse operation (movement and click) together with the corresponding time stamps. The Journalplayback supports the playback module. For the parsing of the analyses we used the so-called Flex (Paxson, 1995) and Bison (Donnelly & Stallman, 1995) tools. Merging with Dragon Naturally Speaking was conducted with a Python script ("Python", 2007).

3.2 Structure of Inputlog

Figure 3 shows the structure of Inputlog, the so-called 'system topology'. Inputlog consists of four subsystems: central, playback, analysis and record. The central system consists of the classes session manager and data manager. The session manager manages a single writing session. The data manager maintains the logging data of a writing session.

The central system has three subordinated systems: playback, record and analysis. These three subsystems are all dependent on the central system:

- The playback system needs logged data to replay a writing session.
- The record system builds the logging data.
- The analysis system uses the logging data for further analysis.

The session manager can run any type of analysis by initializing the right class (e.g. PauseAnalyzer initializes the pause analysis). Per analysis the analyzer and the parser interact with each other.

⁷ Take into account that replaying requires a controlled test environment: the screen settings must be exactly the same between logging and replaying (cf. section 3.4.2).

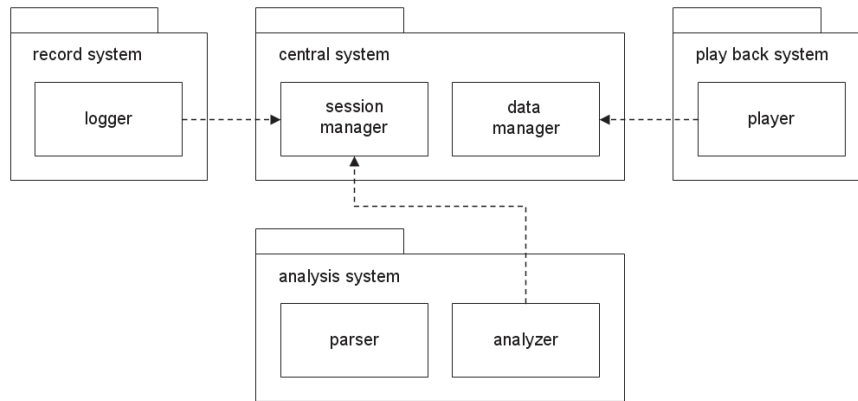


Figure 3. Inputlog system topology.

The classes' player and logger are so-called 'singletons', because only one of them can be active in the system. Consequently, Inputlog always needs to perform a consecutive analysis to verify that only one of these two subsystems is active: the logger cannot log while playing and vice versa. The three subsystems function independently. The GUI interacts with the central system to log, analyze or play a logging session.

3.3 Structure of IDF-files

Bytes	Meaning												
4	starting time of the log in milliseconds												
14 * 4	14 digits that indicate the size (viz. number of signs) of the session variables (6 values for the defined variables, 4 for the user defined variables and 4 for the undefined values)												
14 * #	14 session variables * number of the size that has already been read												
# * 20	serial number of keyboard and/or mouse event (type Eventmsg)												
	Each event that is logged by Inputlog creates a log event of 20 bytes. An Eventmsg is:												
	<table border="1"> <thead> <tr> <th>bytes</th> <th>meaning</th> </tr> </thead> <tbody> <tr> <td>4</td> <td>message type (key in, key out, mousemovement, left mouse button in, left mouse button out, etc.)</td> </tr> <tr> <td>4</td> <td>paramL for keystrokes: virtual keycode for mouse events: x-value of location of the mouse</td> </tr> <tr> <td>4</td> <td>paramH for keystrokes: scancode for mouse events: y-value of location of the mouse</td> </tr> <tr> <td>4</td> <td>timestamp in milliseconds starting from the beginning of the log (time event – starting time of computer) * starting time of the log</td> </tr> <tr> <td>4</td> <td>handle to active window</td> </tr> </tbody> </table>	bytes	meaning	4	message type (key in, key out, mousemovement, left mouse button in, left mouse button out, etc.)	4	paramL for keystrokes: virtual keycode for mouse events: x-value of location of the mouse	4	paramH for keystrokes: scancode for mouse events: y-value of location of the mouse	4	timestamp in milliseconds starting from the beginning of the log (time event – starting time of computer) * starting time of the log	4	handle to active window
bytes	meaning												
4	message type (key in, key out, mousemovement, left mouse button in, left mouse button out, etc.)												
4	paramL for keystrokes: virtual keycode for mouse events: x-value of location of the mouse												
4	paramH for keystrokes: scancode for mouse events: y-value of location of the mouse												
4	timestamp in milliseconds starting from the beginning of the log (time event – starting time of computer) * starting time of the log												
4	handle to active window												

Figure 4. Structure of the IDF-file.

Each IDF-file contains the data of one logging session. In this file the logging data are combined with the session identification data (see section 4.2). The structure of the IDF-file is shown in Figure 4.

3.4 Program settings

Inputlog is dependent on some basic settings that are particular to the computer configuration that is used to log the writing session. We shortly discuss three configuration elements that are relevant when using Inputlog: system requirements, screen settings and keyboard lay-out.

3.4.1 System requirements

The computer must meet the following minimum requirements: Microsoft Word XP, Pentium III, 400 Mhz, 128mb RAM and 50MB additional disk space. In addition, Microsoft.Net Framework 2.0 must be installed, because the interface was developed in Visual Basic.Net⁸.

3.4.2 Screen settings

As stated before, Inputlog replicates the writing session exactly as it has been logged. In other words, it is not a video recording, but an actual playback of the writing session⁹. Therefore, it is important that the computer settings - both screen and program settings (e.g. toolbars, language settings, personal dictionaries etc.) - are exactly the same when replaying a writing session as when the writing session was recorded.

A short example is given to illustrate this issue. A writing session is recorded with only the standard toolbar active. However, between the logging session and the moment the observation session is replayed, the formatting and reviewing toolbar are added to the working environment, resulting in a different positioning of certain (graphic) elements on the screen in comparison to the previous screen outline. Consequently, because the replay uses the graphic xy-position of mouse clicks, the cursor might select a different icon or item than it did at the time the session was recorded.

Figure 5 shows, for instance, that during a replay with different screen settings, instead of changing the font into size 12, the paste-icon was selected. The result is that the continuity of the writing process reconstruction is severely disturbed. That is why we recommend researchers to keep the settings of the computer screen exactly the same between logging and replaying.

Note that this particular point of interest should only be taken into account when using the replay function of Inputlog. The generation of the different analyses is not disturbed by different settings.

⁸ These requirements are related to Microsoft Word. Researchers can download an additional tool from the Inputlog website to check if the research computer can run Inputlog.

⁹ We advice researchers who would like to use this replay as a resource for e.g. retrospective interviews to combine Inputlog with an on-line video registration tool (e.g. Camtasia by Techsmith). A combination of these tools does not lead to any conflict. Keep in mind that the system requirements of on-line video registration tools are quite high.

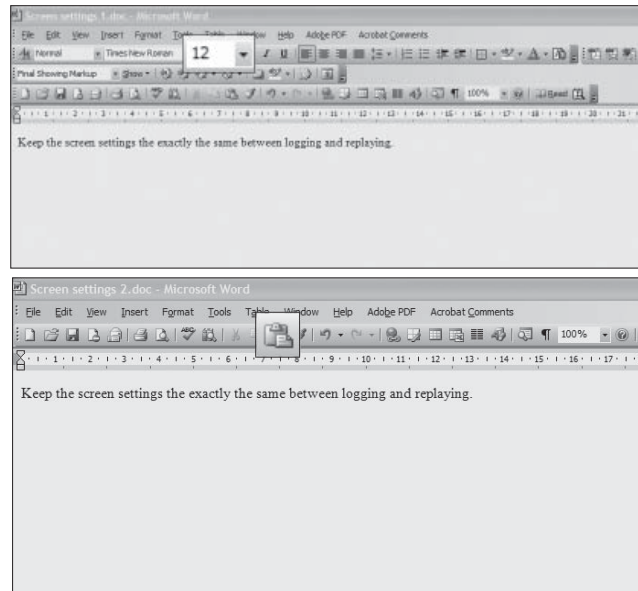


Figure 5. Illustration of playback with different screen settings.

3.4.3 Keyboard layout

A second configuration aspect that can influence the logging of a writing session is the keyboard layouts of the computer used during the logging session. As a result of the vast amount of different keyboard layouts, Inputlog has to detect the correct layout of the keyboard used. For instance, if a writing session is logged with QWERTY-settings and the actual layout of the keyboard is AZERTY, the following sentence will be represented incorrectly in the logging output:

<p>Typed sentence: This is a short writing session in Inputlog. Logged sentence: This is q short zriting session in Inputlog.</p>
--

To avoid this problem Inputlog is programmed to detect the correct hard-coded keyboard layouts for the lowercase characters and reads the connected Windows settings for the uppercase characters. At this moment, 33 different keyboard layouts are predefined in Inputlog 2.0 Beta (see the program's help-file for a detailed list).

4 Functional description

The previous section described the technical background of Inputlog. In this section we shortly explain the basic functionality of the program and its interface. The interface of Inputlog consists of an entry screen and 4 different tab pages: record, generate, integrate and play (see Figure 6). Inputlog also provides a help file. This is a

standard html-file that can also be consulted without starting Inputlog itself. For more detailed information about the use of the program we refer to this file.

4.1 Entry screen

Inputlog starts with an entry screen that provides the user with a short overview of the program. The user can opt to (a) start a new logging session, (b) generate files from an existing logging file, (c) integrate various data, or (d) replay a logged session. The different tabs give the user direct access to the different functions. By using the main menu items at the top of the screen - File, Tools and Help - basic file operations can be activated, program settings can be changed and the help file can be accessed.

4.2 Record

By selecting the record tab, users can start a new logging session in the Microsoft Word environment, in I-pad or without any predefined program. The user can open Windows based software like the Internet, another Word processor such as Works or WordPerfect or programming software¹⁰.

Microsoft Word is the most common word processor at the moment and is therefore very familiar to most writers. I-pad on the other hand looks quite similar to Microsoft Notepad, but is more restricted. This 'in-house' developed limited word processor also provides x/y positions for every character and allows for more fine grained revision analyses in Inputlog. This analysis is still under development.

Next to the keyboard based word processing information from Microsoft Word or I-pad, researchers can integrate speech data from Dragon Naturally Speaking 8.1. To do so, the user has to select the required profile.

Before a new session is started the researcher can first specify the logging file and the identification data for a specific session. It can be identified by a maximum of ten variables (6 predefined and 4 user defined variables). These variables should enable the researcher to identify a writing session in detail. This information is included in the headers of the generated analytical files based on the session source file. In the file information, users can indicate where they want to save the logging session on their computer hard or network drive¹¹, and they can enter the unique filename for the source file, for example FirstnameLastname1 (= Firstname participant + Lastname participant + Number of session). The (source) file that will be generated in the recording session has the extension *.IDF, which is added automatically. This file will be used as input for the generate, integrate and play functions (cf. infra).

¹⁰ Researchers should keep in mind that for certain analyses (e.g. analyses on sentence or paragraph level) only the data logged in Microsoft Word can be used. For this kind of analysis, the basic data are interpreted with a set of algorithms based on Microsoft Word.

¹¹ It is recommended to create a unique folder per participant and per session. This will facilitate the administration of research projects.

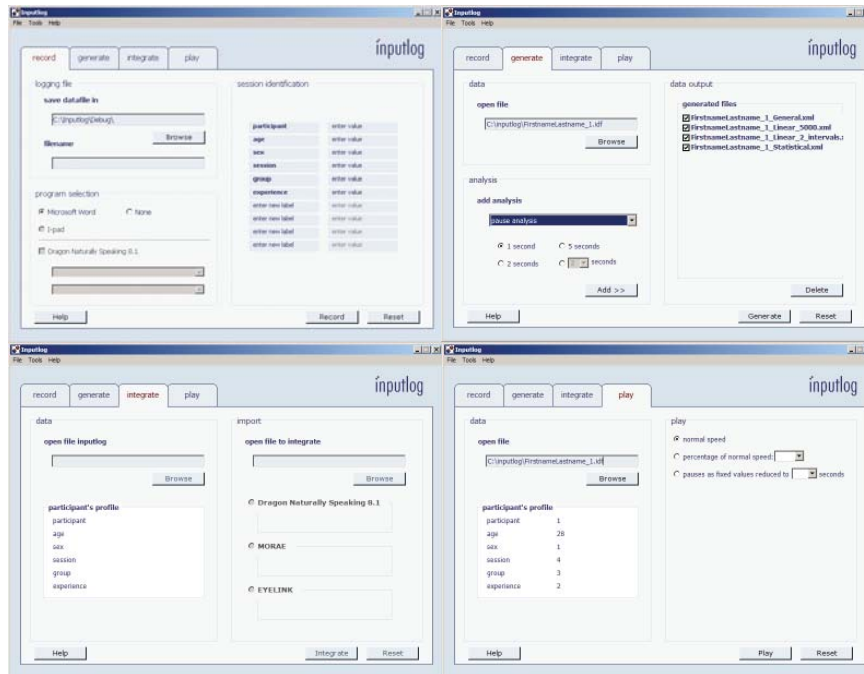


Figure 6. The four main tab pages of the Inputlog interface.

4.3 Generate

The generate tab opens a window showing the different analysis options. In this part of the program, analysis files can be generated on the basis of a source file that was recorded in a previous logging session. In other words, any IDF file can be opened at any time to generate data output files for specific analyses. The generate window consists of three sections. In the 'data section', users can specify the IDF file from which they would like to generate the analyses. In the analysis section the researcher can specify which output needs to be generated, by adding various types of analyses (e.g. pause analyses with 1, 2 and 5 seconds). In the section 'data output', a list of all the files to be generated is displayed. At this stage the researcher start generating or can delete an individual analysis or reset the whole generating procedure.

Inputlog 2.0 Beta offers 4 different data analyses:

1. General logging file: an XML file with the basic logging information of the writing session in which every line represents an input action (keyboard, mouse click or movement and - if present - speech); for every input action the session information is stored together with an identification of the input, the time stamp, the pause time that followed it, and – for a mouse operation - the xy-value of the

screen position (for an overview of the outputs of the analyses see Appendix 2)¹². The XML file can be converted to an Excel file or can be exported to SPSS for further analyses.

2. Linear text analysis: a plain linear text in XML-format with the complete linear production of the text (keyboard and speech) including mouse movements and pauses. The linear analysis is divided into two options: on the one hand researchers can generate a linear output in which the writing activities are divided into periods (fixed time durations of x seconds, free to choose) or intervals (fixed number of equal timeslots in which the writing process is to be divided) of their choice. In both options the threshold for the pause length can be adapted to meet the requirements for a particular study.
3. Statistical analysis: an XML file with basic statistical information of the writing session on a more aggregated level. Several process characteristics are shown, such as the number, mean and standard deviation of characters, words, sentences and paragraphs produced, pause times (based on the threshold entered in the interface) and the use of the different writing modes.
4. Pause analysis: an XML file with analyses of every non-scribal period. The threshold for the pauses can be set to 1, 2 or 5 seconds as a standard or to any user defined level larger than 1 millisecond. Pause data are generated on a more general level: number of pauses, mean and standard deviation of pause length, and on a more specific interval level in which the writing session is divided into 10 equal timeslots. Finally, pauses are summarized per word, sentence and paragraph location.

One other analysis is under construction at this moment:

5. Revision analysis: an XML file with a basic analysis of the number, the level and the kind of revision that has taken place during the writing session (see section 7 further developments).

Inputlog takes some time to generate the requested files; the progress is shown via the MS DOS screens that automatically appear. The different files are all placed in the same folder as the source file of the selected writing session (cf. 4.2 Record). This allows users to keep track of the recordings and analyses per writing session. The XML files that are generated can be read as such or can be imported into various statistical programs. As shown in Figure 1 the data structure of output files has a column structure. The general file can be used as a search file and a back-up file. The linear text files are represented in a more readable format. The researcher can easily switch between these two files (cf. section 4.3.2 Linking) to search for information. The statistical and pause analyses are structured in one column. This enables researchers to

¹² In this chapter we have opted to append examples of the various output files. For more detailed information on these outputs we refer to the Inputlog website and the program's help file.

transpose the data to a row structure and subsequently combine the data from various participants to a large data set. See Appendix 2 for further information on the XML files that are generated.

As stated before, researchers should take into account that keystroke logging provides a lot of data that need to be carefully analyzed (Spelman Miller & Sullivan, 2006). Researcher can adapt output files to provide the correct information to their research question, for instance, pauses that are recorded in keystroke logging serve different purposes, and therefore, describe different writing processes (Schilperoord, 2002). Also definitions on for example pause locations should be carefully kept in mind. The definitions that researchers have in mind might not be the same as the one the program uses.

In the sections below we provide more detailed information about how researchers can influence and use the data structure to get the best result for their analyses. The output files have some special features that we would like to elaborate on: selection of specific information by filtering the data output, linking between output files, error correction and generation via command line.

4.3.1 Filter data output

The XML structure of the data output enables researchers to adapt the data file to their needs by selecting the most important parts of a writing session, or in other words to filter out the parts that they do not need. This feature can be used to delete, for instance, the start of a writing session. In the beginning of a session participants might have some additional questions about a task while Inputlog is already recording, or the initial reading and planning time is informative as one measure, but not in the whole analysis. It can also be used to filter out data during a writing session, for instance, mouse movements that are related to search actions on the internet during a writing session. Again, the searching behavior of participants might be interesting in one measure, but not in the analyses as a whole. In such cases researchers can opt to filter the data output at a later stage. Consequently, the files will become more accurate if certain data are deleted.

The reduced basic XML file that is generated can easily be filtered via MS Notepad. Researchers only need to remove the irrelevant segments to be left with the adequate data for answering a research question (see the program's help file for more detailed information).

4.3.2 Linking

Inputlog 2.0 Beta offers the possibility to switch between output files (viz. general and linear, cf. *infra*) by linking time stamps (StartClock, cf. Figure 1). When using keystroke logging as a research tool, interpreting your data is an important step. Detailed case analysis may help to illustrate certain findings. If we take for example a writing study on press releases in which the researcher is interested in preformulation, he or she might be interested in a particular word, name or phrase that should be integrated

in the preformulation. It is easy to search for these kinds of segments in the linear text representation. The links on the time stamps related to these text segments make it possible to specifically locate the episodes in the general logging file, providing more detailed and fine-grained information about for example pause times preceding the preformulation. A reverse example might be that the researcher is interested in pauses that fall within a certain range. These pauses can easily be selected in an Excel representation of the general file (viz. by conditional formatting) and can then be seen in their writing context in the linear file, making a contextual interpretation easier.

4.3.3 Error correction

At present, more than a hundred researchers from all over the world are using Inputlog as a tool for writing research. As mentioned before, an important configuration aspect that can influence the logging of a writing session are the keyboard layouts of the computer used during the logging session. Although we have predefined different keyboard layouts some characters or symbols are not known by Inputlog. This may hinder the logging process. In that case researchers can search via MS Notepad in the reduced basic XML file and replace the unknown character by a known character (see help for more detailed information).

4.3.4 Generation via command line

Professional users can also use the command line of MS DOS to generate various analyses. This command line is related to the interface. An advantage of this command line is that researchers can generate an enormous amount of various relations in one command and which they can save in a Batch file. The commands that need to be used to create the command line can be found in the help file.

4.4 Integrate

The third tab of Inputlog is Integrate. This module has been developed on a conceptual level and the actual integration is being performed manually at the moment. This functionality will be available in the summer of 2007. This module will allow researchers to merge different XML output files of other logging and observation programs. At this point there are three main programs that we are planning to integrate with Inputlog: Dragon Naturally Speaking 8.1, Morae and Eyalink.

4.4.1 Dragon Naturally Speaking 8.1

As mentioned before Inputlog data can be combined with speech recognition output from Dragon Naturally Speaking (cf. Figure 1). On the record tab researchers can indicate which user of Dragon Naturally Speaking needs to be selected and on the Integrate tab researchers can merge both logging files into one file. To guarantee a successful merger, both logging files need to count with the same time measure. More specifically, Inputlog and Dragon Naturally Speaking need be based on an absolute

epoch time¹³, resulting in a new XML file that will be used as starting point for the analysis.

4.4.2 Morae

Inputlog can be seen as a research instrument that provides data on the micro-analytic level of writing processes. To combine these very detailed data with macro-analytic data provides new perspectives. Morae is a macro oriented observation tool developed by Techsmith (www.techsmith.com) for usability research purposes. It also enables researchers to code a process on various levels while observing it online. It is our intention to complement the Inputlog data with Morae logging data. This program captures, for instances, changes between programs on a higher level and registers, for instance, the url-addresses of websites that are accessed during a writing session. In addition, researchers can flag important segments of writing sessions and code these segments leveling great detail. Just like Inputlog, Morae also logs very detailed timestamps which should enable us to integrate the additional data registered by Morae into the output of Inputlog. For the observation of writing processes during which the participants combine Microsoft Word with other programs especially, this integration opens new perspectives for further analyses. In the research project described in chapter 7 we combine Inputlog with data from Morae (and Dragon Naturally Speaking).

Figure 7 shows an example of a merger between the output files of Inputlog, Dragon Naturally Speaking and Morae. As seen in Figure 1 Dragon Naturally Speaking data are shown per dictated segment and the Inputlog data per event.

In this example four extra columns are added (cf. Figure 1: for the English translation of this excerpt see page 232). These extra columns are the clock time of Morae, the merge time (in milliseconds) between Morae and Inputlog calculated manually in MS Excel and SPSS, Morae's marker and corresponding code.

In this excerpt Morae's marker refers to a technical error caused by the speech recognition software. The code 'O' is chosen by the researcher and stands for 'technical error, immediately solved'. The participant initially dictated the following two segments as one segment (4:43) 'The general obligations of the employer related to the protection of the employees' health at the working place' and (5:02) 'fit in with the development of the prevention policy'. In the pause related to the full stop (3.1 seconds) the participant is reading the text produced so far and notices that the final segment does not appear on the screen. After a pause of 5.6 seconds he opts to switch into writing mode to delete the full stop with his keyboard and subsequently types an interspace. He proceeds with speech and dictates the final segment once again. It is striking though that he does not correct the other smaller errors in the text (title of section), for example 'context' instead of 'context in'. A subsequent segment of the

¹³ The start time of the epoch is January 1, 1970, 00:00 h GMT.

observation revealed that he prefers to continue with text production and to finish his paragraph.

Morae	Excel & SPSS	Morae	Morae	Inputlog	Inputlog & Dragon Naturally Speaking	Inputlog	Inputlog
ClockTime	Mergetime	Marker	Code	WritingMode	Output	StartClock	PauseTime
	214000			3 - dict	context in	0:04:39	1673
	218000			1	ENTER	0:04:42	3199
	218000			1	ENTER	0:04:42	328
	219000			3 - dict	de algemene verplichtingen van de werkgever met betrekking tot de bescherming van gezondheid en veiligheid van werknemers op de werkplaats	0:04:43	924
	230000			3 - dict	\punt	0:04:55	3129
	237000			1	BS	0:05:01	5666
	238000			1	SPACE	0:05:02	703
	238000			3 - dict	kaderen in de ontwikkeling ervan een preventiebeleid \punt	0:05:02	390
0,05:42.77	244000	Marker	○				.

Figure 7. Example of merged output file with data of Inputlog, DNS and Morae.

This example shows the interaction between the macro-analytic approach of extra process coding in Morae and the micro-analytic approach of Inputlog. It enables researchers for example to search through the information in a more structured manner and to combine more global analyses with very detailed analyses.

4.4.3 Eyelink

Finally, we are exploring the possibilities to integrate eye tracking data with Inputlog data. At the moment ScriptLog is the only logging tool that combines eye tracking and keystroke logging¹⁴ (Andersson & et al., 2006). Eyetracking is ideally suited for research on reading and writing processes. In chapters 5, 6 and 7, we study the effect of the text produced so far in more detail. In these studies pause time was one of the measures that we used to describe the cognitive load that the text produced so far imposes on a writer. Pause times may be indicative of monitoring the text produced so far. However, there is no direct indication of what writers are looking at and whether they, for example detect an error in the text produced so far. The recurring comment made in those chapters is the possibility to obtain more fine-grained data about the writers' monitoring activities during text production¹⁵. Eyetracking enables researchers to see what writers are looking at during a writing task.

Eyelink is one of the head mounted eye tracking systems of SR Research¹⁶. The eye tracker also logs very detailed timestamps that could form the basis for the integration of both data sets. Of course, the amount of data will increase enormously when combining these two data sets. However, in this situation the Inputlog data can provide more insight into the eye tracking data.

¹⁴ ScriptLog uses the eye tracking system iView X HED + HT (www.smi.de)

¹⁵ The study described in chapter 5 and 6 was replicated with the addition of Eyelink (Thomas Quinlan was the coordinator of this project).

¹⁶ More information on Eyelink can be found on www.eyelinkinfo.com

4.5 Play

The final tab of the Inputlog interface is the play function. A recorded writing session can be replayed using Inputlog. Again, the IDF file is used as a source file for the replay. To verify the participants' profile of the file that is selected for a play back session, all defined variables of the session identification appear in the dialog box on the left side of the screen. The writing session can be replayed at different speeds. It can be played back exactly as it was recorded (in real time): this is an exact reproduction of the recording of the session. Another option is that users select a percentage of the real time speed. However, we would like to restrict this option to a maximum of 120% of the original speed. Otherwise the program may have difficulties in processing and performing certain actions that require extensive memory access. The final option 'pauses at a fixed value' enables researchers to assign a fixed value, for example 0.1 seconds, to every pause (non-scribal activity). This allows users to view a writing session without long interruptions or pauses.

4.6 Help file

So far, this chapter has described the possibilities of Inputlog, the most important technologies and main functionalities. For more information on certain features we refer to the help file. The help file contains a structured manual of the program. The main part of the help file consists of four chapters, each corresponding to the four basic functions of the software tool: record, generate, integrate and play. The help file on each of these functions is also directly available via the help button on the associated tabs. Furthermore, we integrated rules and definitions. Researchers should be able to have a clear idea of what they are measuring when using Inputlog as a research tool. This explains why we detail, for example, all the rules that are used to define pause locations and why we provide definitions of standard deviation.

5 Applications

To illustrate the research potential of Inputlog, we describe three possible applications in this section. In the first two applications, we briefly discuss writing processes taken from experiments in which we have used Inputlog as a research tool. The third application shows Inputlog as a research instrument in a broader sense.

Keystroke logging can be a suitable research instrument in a number of contexts (for an overview see Sullivan & Lindgren (2006)). Research areas that Inputlog can provide data for include: studies on cognitive writing processes in general, description of writing strategies in professional writing or creative writing, the writing development of children, of those with and without writing difficulties, first and second language writing, the writing of expert and novice writers in professional contexts and in specialist skill areas such as translation and subtitling. Next to strict writing research it can also be integrated in educational domains: second language learn-

ing, programming skills, typing skills and again subtitling¹⁷. For example, Lindgren (2003) used keystroke logging to facilitate reflection through peer-based intervention by replaying the writing session (in an educational context). Next, Sullivan and Lindgren (2006) discuss in their keystroke logging book how keystroke logging can also be used to investigate theories that do not primarily concern writing, but for which writing provides a window on the theory. Case in point is Galbraith's (1999) dual-process model. They state that:

To use key-stroke logging, coding criteria would need to be developed that would permit the researcher to decide when a new idea had been developed in the evolving text and to decide when an editing action can be defined as a revision as defined within the dual-process model.

Theory-building through the use of keystroke logging holds tremendous potential.

In this section we show two possible research designs - a case study versus an experimental study - in which keystroke logging can be used. The first example is taken from a case study of a writer producing a bad news letter; the second experiment features a more technical writing experiment focusing on the working memory requirements necessary for error detection in the 'text produced so far' (see also chapters 5 & 6).

5.1 Case study: pausing and revision behavior in writing bad news letters

In Figure 8 we show the linear text representation of a writer producing a bad news letter in which (s)he declines an offer to deliver a keynote address for an international conference. We were especially interested in the process characteristics of the writing episode that concerned the wording of the bad news itself. As we know from the literature on this issue, the strategic considerations to make the bad news as acceptable as possible for the reader are crucial in the perception of the message (and may also determine the future interpersonal relation with the reader).

The writer needed about ten minutes to finish the letter of about 130 words. The replay function of Inputlog was used as a stimulus for a retrospective think-aloud protocol. Figure 8 shows the sequential linear output (periods of 30 seconds) of a fragment that illustrates the writer's strategic considerations at the beginning of the second paragraph.

0:02:30	{8440} Presenting a {1330} paper to {2360} this group {3660} [CTRL+LEFT] {1020} quality {180} [END] {1030} deserves a {1530} thor
0:03:00	ough {1390} and {2420}time {1190} [BS 4] {1660} comprehensive effort {2130}. {3610} Of course, [BS 12]. {1020} Obviously {2900}
0:03:30	, such an effort {2030} requires {5550} a lot of [BS 8] {1050} considerable [BS 12] time. {4840}

¹⁷ In January 2007 we started a research project entitled 'Live subtitling using speech recognition: procedures for quality improvement' (in cooperation with the Flemish Radio and Television Broadcast station).

0:04:00	However, {1340} my schedule {2660} [CTRL+LEFT 4] {1280} [DEL 10] M [END] {2380} is fully comm{1770}itted to a
0:04:30	writing project {2300} {7170} [CTRL+LEFT 7] {2160}

Figure 8. Linear output of two fragments of a writing process in which a bad news letter was produced.

The fragment shows a very fragmented and staccato writing process that starts off with a long pause of more than 8 seconds (8440 milliseconds) in the beginning of the second paragraph (announcement of the refusal). In the production of about 30 words, there were 28 pauses longer than one second, which is substantially more than in the previous period when the introductory context of the letter was written. About half of the time in this fragment is used for pausing, showing the time attributed to careful and strategic formulation. An interesting example of such a strategic consideration is the revision that takes place after four minutes (0:04:00). The participant starts the sentence with 'However', but two words later he rereads the beginning of the sentence and realizes that this contrastive connective announces the bad news in a too early stage of the paragraph. Therefore, he decides to delete the connective to neutralize the context of this argumentative sentence.

Again, this example shows that process logging enables researchers to analyze writing processes from different perspectives enriching possible interpretation based on text analysis. Other observation methods, like recording (retrospective) thinking aloud protocols, might complement the acquired data.

5.2 Experimental study: the text produced so far and the use of working memory

An experiment was set up to assess the memory load while interacting with the 'text produced so far' during the text production phase. The design included the most frequently occurring error types found in a case study of professional writers that were using speech recognition for the first time to write business texts (Leijten, Ransdell, & Van Waes, submitted, also chapters 5 & 6). In the example at hand, we selected two text fragments in which a sentence with large speech recognition errors is corrected. The errors that occurred in the text produced so far were considered major errors because the number of characters that differed from the intended text was more than two. Besides, the selected errors could only occur in speech recognition, because they were misrecognitions that resulted in phonologically similar words. The task in the experiment consisted of a set of sentences that were presented to the participants as contextual information. In Figure 9 an example of a correct and an incorrect sentence is given (see sentence 2: translated from Dutch). After every context sentence (1) the participants had to click the 'ok' button, to indicate that they had finished reading the sentence. A subclause of the previous sentence was then presented as text produced so far (TPSF) in a subordinate causal structure (2), and the participants were prompted to complete the sentence (3) on the basis of the context provided earlier (1).

<p><i>Correct</i></p> <ol style="list-style-type: none"> 1. Because the height was not indicated, the pick-up truck drove by the underpass. 2. The pick-up truck drove by the underpass, 3. because the height was not indicated. <p><i>Incorrect</i></p> <ol style="list-style-type: none"> 1. Because the height was not indicated, the pick-up truck drove by the underpass. 2. The picot trug drove by the underparts. 3. because the height was not indicated. <p>Dutch “De heeft week reed onder de brief door want de hoogte was niet aangegeven.”</p>

Figure 9. Examples of correct and incorrect sentences in the TPSF experiment.

The participants could either choose to correct the errors first and then complete the sentence, or they could complete the sentence first and then correct the error. In casu, for this sentence 83% of the participants preferred to complete the sentence first and correct the error afterwards. Figure 10 shows the linear output of the writing session of two participants that used different writing strategies. Pauses longer than 1 second are included in the output; texts are in Dutch.

Writer 1	Writer 2
sentence completion > error correction (83%)	error correction > sentence completion (17%)
[MwC.910,701-391,348] [McLeft] de hoogte was niet aangegeven {2190} [MwC.491,352-145,346] [McLeft] [BS 9] eftruck {1270} [RIGHT 17] [DEL 4] ug <1170> [MwC.165,335-933,699] [McLeft]	{2660} [McLeft] [MwC.910,711-149,347] [McLeft] [MwC.149,347-375,355] [BS 9] ftruck {1600} [MwC.372,357- 269,340] [Mselect] [MwC.270,340-342,341] [BS 4] ug <1.81> [RIGHT 15] de hoogte stond niet aangegeven <1250> [MwC.348,338-884,695] [McLeft]

Figure 10. Example of linear output of a text fragment generated by Inputlog.
 Remark: MwC: mousemovement without click; McLeft: mouse click left;
 Mselect: selection of text by mouse; BS: backspace; right 17: arrow right 17.

In these examples of two short writing fragments, different writing strategies can be distinguished. The first writer prefers to continue completing the text first, before correcting the mistake in the TPSF. He positions his cursor after the first segment (TPSF; cf. xy-value of left mouse click) – without a significant previous pause – and then completes the sentence. After the sentence is completed, he pauses for 2.19 seconds. He then positions the mouse behind the incorrect word and deletes it by using the backspace key. Next, he navigates through the text by pressing the right arrows key and deletes the second error. Finally, he pauses briefly before moving on to the next sentence.

The log of the second participant's writing session reveals a different pattern. This writer pauses before he starts writing and then positions the cursor behind the error to

correct this first. After correcting the second error he completes the sentence. In the analysis of the research data presented here, we were mainly interested in a description of the interaction with the text produced so far. Differences in writing strategies can be related to both individual differences and to error characteristics in the TPSF. This short example illustrates how the detailed process information that is generated by Inputlog provides a basis for analyzing writing strategies that are no longer visible in the final written product¹⁸.

5.3 Applications on a broader level

Inputlog is a research tool that logs writing processes. In addition, Inputlog can also be used as a research instrument to facilitate research methods on a broader level. For instance, Inputlog can be integrated as an additional research instrument to register complimentary research methods such as thinking aloud and retrospective interviews. In the research project described in chapter 7, we recorded the retrospective interviews of the participants via Inputlog and Dragon Naturally Speaking. As a result, the retrospective interview was automatically transcribed.

This too opens up avenues for further research on thinking aloud and retrospective interviews as research methods. In addition to pauses in the writing process, pauses in a thinking aloud protocol can be an interesting measure for gaining insight in the cognitive load during production of protocols.

6 Data analysis

As mentioned earlier, keystroke logging generates enormous amounts of data. Hence, we would like to describe three methods for analyzing Inputlog data. The first perspective is the use of factor analyses on the aggregated data of Inputlog. The second and third perspectives are related to developing text visualizations.

6.1 Factor analysis

Keystroke logging programs have the advantage that they collect data on a very low level. However, this advantage also has its drawbacks. It is sometimes very hard to interpret the large amount of data. Moreover, the main objective of logging these data is to indirectly measure some aspects of cognitive writing processes. Unfortunately, deciding which of the reported measures are driven by the same underlying variable can be difficult.

A statistical technique that might be helpful in overcoming these problems is *factor analysis*. In this section we shortly introduce the main procedure for using factor

¹⁸ Moreover, writing processes can be hard to register on-line, since pauses are related to various writing subprocesses (Schilperoord, 2002). A combination with thinking-aloud or retrospective interviews may help in these cases.

analysis as a way to explore data collected and analyzed with Inputlog¹⁹. In his introduction to factor analysis, Field (2005, p. 619) explains that this technique has three main uses:

1. to understand the structure of a set of variables;
2. to construct a questionnaire to measure an underlying or latent variable;
3. to reduce a data set to a more manageable size while retaining as much of the original information as possible.

In the perspective of writing research with logging programs, the third function is of great value. Factor analysis²⁰ can help to reduce a set of interrelated variables into a smaller set of so-called factors (or latent variables). In other words, the objective of this technique is to explain the maximum amount of variance by using the smallest number of explanatory concepts. Factor analysis reduces large amounts of measures taken to their underlying dimensions and cluster variables in a meaningful way. This is achieved by looking for variables that correlate highly with a group of variables, but which do not correlate with variables outside of that group.

Exploratory factor analyses on Inputlog data can be conducted by following this procedure:

1. Select variables: All the variables in the statistical and pause analysis of Inputlog can be selected (e.g. variables related to text length, duration of the writing session and pausing behavior). These general measures can be complemented with calculated ratios (e.g. words produced during the writing process divided by the number of words in final text, as a possible indication of the recursivity of the writing process).
2. Conduct the factor analysis: Run the factor analysis by selecting the variables you want to include in the exploratory analysis (e.g. by using SPSS).
3. Check the variables in a correlation matrix: The correlation matrix (R-matrix) gives an overview of the significance value of each correlation. Because factor analysis has to include variables that correlate fairly well, you can use this matrix to evaluate the pattern of relationships. It is recommended to exclude strongly correlating variables (e.g. a correlation coefficient higher than .9) and variables that hardly or do not correlate. If necessary, rerun the factor analysis (step 2)²¹.

¹⁹ We would like to thank Gert Rijlaarsdam (University of Amsterdam) who presented this method during a workshop in Amsterdam (November 2006).

²⁰ There are many different techniques or methods for conducting a factor analysis (for example, principal axis factoring, maximum likelihood, unweighted least squares, generalized least squares) and also many different types of rotations that can be done after the initial extraction of factors. Because this is only a short introduction to factor analysis, we will mainly base our description on a simple principle component analysis in combination with a varimax rotation.

²¹ SPSS also offers the possibility to report a determinant value (which should be different from 0), the Kaiser-Meyer-Olkin Measure (minimum value > .6) and the Bartlett's test of sphericity (to test whether the correlation is an identity matrix). In sum, these tests provide a minimum standard for conducting factor analyses.

Table 1. Example of an SPSS output showing the Total Variance Explained in a factor analysis

Component	Initial Eigenvalues			Extraction Sums of Squared Loadings			Rotation Sums of Squared Loadings		
	Total	% of Variance	Cumulative %	Total	% of Variance	Cumulative %	Total	% of Variance	Cumulative %
1	4,949	32,992	32,992	4,949	32,992	32,992	4,201	28,007	28,007
2	3,018	20,120	53,112	3,018	20,120	53,112	3,018	20,121	48,128
3	1,727	11,510	64,622	1,727	11,510	64,622	2,430	16,200	64,328
4	1,136	7,576	72,198	1,136	7,576	72,198	1,181	7,871	72,198
5	,977	6,515	78,713						
6	,854	5,693	84,406						
7	,605	4,033	88,438						
8	,514	3,427	91,866						
9	,426	2,838	94,704						
10	,371	2,474	97,178						
11	,185	1,232	98,410						
12	,093	,619	99,029						
13	,080	,533	99,562						
14	,038	,250	99,812						
15	,028	,188	100,000						

Extraction Method: Principal Component Analysis.

- Determine the factors: In the principal component analysis the components are ordered on the basis of the initial eigenvalues as well as a cumulative percentage of the total variance. By default, SPSS uses a cut-off point of the eigenvalue that is larger than 1 and uses all the factors above this threshold to optimize the factor structure in a Rotation Sums of Squared Loadings. In the example we present in Table 1 fifteen variables were used in the factor analysis. Using the above mentioned criterion (eigenvalue > 1) we can account for roughly 72 % of the total variance by using only four components or factors, which is a radical reduction of the total number of variables.
- Interpret the factors: To make the factors more accessible you can use a Rotated Component Matrix to see which variables correlate with which factor. Because factor analysis is an exploratory technique it is now up to the researcher to make sense of the factors. However, the loading values reported in the matrix are very helpful in making sense of the contribution of each of the variables to a specific factor. Also, comparing individual cases on the basis of the selected factors can be useful for interpreting the factors. In our example, for instance, the first factor was mainly characterized by a low number of produced words, a large pause time with especially fairly long pauses in the first half of the writing process and a high ratio of number of words produced vs. words in the final text (little revision). The second factor on the other hand was characterized by a much lower ratio, a high number of (short) pauses and a relatively long total production time. As this illustration shows, these factor descriptions create an interesting basis for relating them to writing profiles.

6. Conduct a multi and/or univariate analysis: In the last step of the procedure, the factors can be used as variables, for instance, to test the effect of experimental conditions.

In sum, factor analysis is a possible way for the researcher to reduce of the set of measures Inputlog datafiles produce. The short description presented above details a possible procedure for exploring the data by using this statistical technique and exploring the underlying dimensions of clusters of variables in a meaningful way.

6.2 Progression analysis and GIS

The development of the text is currently represented in Inputlog in the linear text analysis. To visualize this textual development we would like to extend Inputlog with two graphical representations of the text progression, a basic and a more extensive one. In the basic progression analysis we would like to visually represent the number of characters that are produced at each moment during the writing process taking into account the characters that are deleted at that stage. This basic progression analysis is based on Perrin's (2003) writing strategy research .

Because this basic analysis is a static reproduction of the writing process, we would also like to develop a more interactive representation inspired by Lindgren (2002; 2007). She uses a Geographical Information System (GIS) to visualize and summarize the writing process. GIS enables researchers to analyze different subprocesses of the writing process by selecting representative variables. The graphical representations are not static. Instead they allow a researcher to interact with the data at different levels and to move back and forth between the data and their representation. Consequently, as well as being a tool for visualization and data mining, this technique can support a dynamic analysis of the cognitive processes during writing due to the interactive nature of the data mining approach on which GIS is based. Figure 11 shows an example of a GIS graph that is based on a manually adapted dataset of Inputlog²². To generate these kinds of progression analyses automatically, we first need to further optimize the revision analysis because these analyses components are based on the revision output.

The x-axis represents the time (in seconds) while the y-axis indicates the number of characters that are produced *cq.* realized effectively in the text produced so far. The top line indicates the total character production including deleted characters at each point in time; the bottom line indicates the characters retained after deletions at each point in time. The dotted line shows all the points in time at which the writer is working on the text, representing both pauses and deletions. The size of the circles refers to the length of the pause. When the line drops, a number of characters are deleted.

²² We would like to thank Eva Lindgren, Umeå University (Sweden) who generated this graph for us with ArcGIS (ESRI).

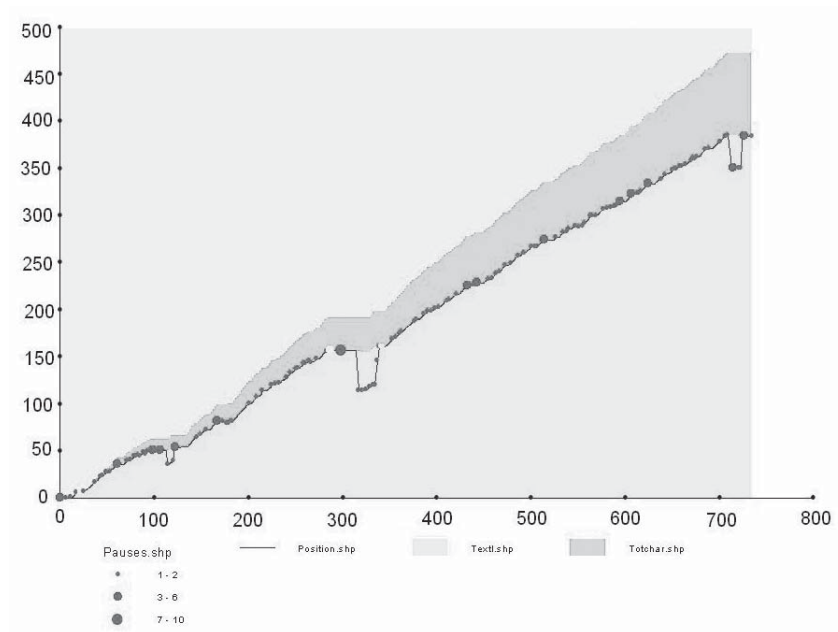


Figure 11. Text progression graphic as logged in Inputlog and visualised in a GIS application (source: Lindgren).

7 Further development

We have identified four important niches that may increase the applicability of Inputlog, especially in the domain of writing process research. In the near future we would like to further develop the following (in order of priority).

7.1 Revision analysis

Work on the revision analysis component for Inputlog has already been started, but because of its complexity it is very time consuming to further stabilize and extend the analysis module. In the revision analysis we would like to produce an output analysis in which different characteristics of in-process revisions are described, for example the number of revisions, type of revisions, level of revisions, number of words and characters involved in the revision operation, as well as the location of the revisions in relation to the point of utterance. To define revisions we have developed an algorithm and a set of rules. The revision analysis first of all defines critical events in the writing process that can be linked to a revision and then evaluates these instances by comparing the operations in the isolated writing episode to the revision rules in the algorithm. Inputlog successively analyses the beginning of the revision, the selection of the text to revise or the positioning of the cursor, the (possible) deletion of the

text and the end of the revision. In Figure 12 we describe two (technical) revision operations to change the last word of the sentence ‘Questions of science, science and progress.’ into ‘evolution’.

Questions of science, science and [progress.]¹ {evolution.}

The first operation is a very basic one: the writer simply uses the backspace key at the point of utterance to delete the full stop and the word ‘progression’ and then types the new word ‘evolution’ (see rule 1). This is a rather minimal operation, because the writer does not have to move or position the cursor in the text produced so far. However, the writer could also opt for another sequence to realize this substitution: he can move the mouse to the left, position the cursor by left-clicking the mouse, use the delete key to delete the word, change the text and move to the point of utterance by using arrow keys to the right (see rule 2).

	begin (movement)	selection/positioning	deletion	end (movement)
1	-	-	backspace	-
2	mouse movement left	left click	delete	arrow right
<i>n</i>

Figure 12. Examples of rules defining revisions.

At the moment we have predefined about 50 sets of rules to test the algorithm for deletions and substitutions. However, after the testing phase, the rules will have to be extended and further tested to cover a more complete range of revisions.

7.2 Subtitling via speech recognition

Speech recognition software is now commonly used during simultaneous subtitling of (live) television broadcasts via ‘respeaking’. Running commentary and live interviews during the live broadcasts are dictated with speech recognition software to a subtitling program. The speed of text production in speech recognition increases the possibility to subtitle more and more programs. Broadcasting stations that use this technique include: the BBC in England, the VRT in Belgium and the NOS in the Netherlands.

Inputlog can be used to describe the writing process during subtitling in order to contribute to the improvement of this ‘writing’ method. Live subtitling always takes place under high time pressure. The translation of spoken text to written text makes it an extra complicated task. From a methodological point of view, these writing processes are very interesting since they allow us to observe human-computer-interaction in a detailed manner. In this research area we would like to focus on three perspectives: speech recognition (how can the quality of speech recognition be improved?), subtitling (what is the best method - block versus scrolling - to use for live subtitling?) and cognitive writing processes (how can cognitive difficulties be

prevented?). The main goal of this research project is to increase the speed and accuracy of subtitling via a procedure for quality improvement.

7.3 Typing tests

The influence of a writing instrument on the writing process probably interacts with the degree of automation of the writing process at hand (Caporossi, Alamargot, & Chesnet, 2004). When writers start using the keyboard to write their texts, we see that it is influenced by motor skill and by linguistic elements (cf. developmental writing). Therefore, researchers might also be interested in measurements that evaluate and measure writers' typing skills and use it as an assessment and coaching tool in typing courses.

An interesting measure on typing skill that Inputlog can provide is the 'inter key intervals' (Nottbusch, Weingarten, & Sahel, 2007; Weingarten, Nottbusch, & Will, 2004) or the transition time (Grabowski, 1996; Wengelin, 2006). When elaborating parsing rules in Inputlog, a possible solution is a self-regulatory add-on that recognizes typing errors in copying tasks and shows statistics on a typing test (e.g. speed, accuracy, type of revisions). To this end, the revision analysis of Inputlog needs further improvement. The results of such a typing test could be used to define writer specific pausing thresholds. Typing performance can also be used as a covariate in analyses in which, for instance, process time is important.

Furthermore, detailed results of typing tests and key transitions also enable teachers to gain insight in the difficulties of a writer acquiring typing skills (e.g. left hand-right hand coordination during typing) and to teach adequate and very specifically diagnosed strategies. A related topic that we would like to address is the analysis of writing processes of dyslectic writers.

7.4 Searchable output files

The linear representation of writing sessions logged with Inputlog is helpful when searching for important information. In section 4.3.2 we described the possibility to switch between the general output and the linear text file to facilitate searching in the files. However, we would like to develop this feature in a broader context. At present, researchers can search 'manually' in the files and they can search for correct representations of words, for example a preformulation in a press release must contain the word 'bank stock'. This method is not flawless, since writers make typing errors that disrupt the linear representation of words. Parsing rules should make it possible to find strings with typing errors. In the interface researchers should also be able to define the view of the linear representation, for example each 'searched word' is represented on a separate row with the corresponding action and pause times.

This opens several new research perspectives for example for studies on text coherence. Researchers can easily filter causal relations that are linguistically marked and analyze the production and pause times.

In addition to the developments in these particular areas, we will pay special attention to the further development and optimization of the existing modules. In addition, the flexibility of adapting the output files of Inputlog to the specific needs of researchers is high on the agenda. Furthermore, the compatibility with different versions of the Windows operating systems and the Office environment will require constant attention.

8 Conclusion

Inputlog is a registration tool that enables researchers to register computer based writing processes in Windows environments and subsequently analyze and interpret the writing behavior of writers. In addition to other keystroke logging programs Inputlog also logs texts dictated by the speech recognition software Dragon Naturally Speaking 8.1.

Keystroke logging is an unobtrusive way to observe writing processes (indirect), and as such, it does not influence the writing processes of the writers. They might be aware that their writing process is logged, but other than that they are not confronted with them being a research object. This differs from thinking-aloud protocols (direct observation of cognitive activities) in which writers are very aware of the research situation (Janssen, Van Waes, & Van den Bergh, 1996; Van Someren, Barnard, & Sandberg, 1994).

We see various possibilities to integrate Inputlog as a research tool in writing process research. On the one hand as a research tool that can be used autonomously, to answer research questions on linguistic, textual and cognitive studies of writing, but also for broader applications concerning the development of language learning, literacy and language pedagogy (Spelman Miller & Sullivan, 2006, p. 1). Secondly, it can be used in combination with other research methods, for instance eyetracking, usability testing and thinking aloud protocols (to create triangulation, as described by Van der Geest, Leijten & Van Waes (2006)). Finally, it can be called in as an instrument to analyze other research methods, for instance, to measure pause durations within thinking aloud protocols that are automatically transcribed via speech recognition.

Notes

To facilitate a broad usage of Inputlog, the program is put at the disposal of the research community free of charge (www.inputlog.net), provided that reference is made to: Leijten, M., & Van Waes, L. (2006). *Inputlog: New perspectives on the logging of online writing processes in a windows environment*. In K. Sullivan & E. Lindgren (Vol. Eds.) & G. Rijlaarsdam (Series Ed.), *Studies in Writing: Vol. 18. Computer Key-Stroke logging and Writing. Methods and Applications* (pp. 73-94). Oxford: Elsevier.

User feedback is very important for the evaluation and further development of Inputlog. For any and all questions or feedback, please feel free to contact

marielle.leijten@ua.ac.be or luuk.vanwaes@ua.ac.be.

Acknowledgements

We would like to thank Wesley Cabus, Ahmed Essahli, Wim Claessens, Mathia Van de Poel and Bart Van de Velde for their excellent work in programming Inputlog (BOF 2005-2007). We would also like to thank the KdG-Polytechnic Antwerp for providing internships and a collective project on Inputlog (PWO 2005-2008; dr. Nico Verlinden). We would also like to thank Stijn van Even, Guido Gallopyn, Hans Geuns and Neil Grant of Nuance (previous Scansoft) for all their efforts in making the logging facility at Dragon Naturally Speaking available to us. Finally, we would like to thank Tom Van Hout for proofreading this chapter.

Appendix 1

Glossary

ArrayList	Implements the IList interface using an array whose size is dynamically increased as required.
BAT	A batch file is a text file containing a series of commands intended to be executed by the command interpreter. When a batch file is run, the shell program (cmd) reads the file and executes its commands, normally line-by-line. DOS batch files have the filename extension .BAT.
CPU	Central Processing Unit Reads software instructions and tells your computer what to do.
DLL	Dynamic Link Library A Windows library that can be shared by multiple applications
event	Every action on the computer
EVENTMSG	The EVENTMSG structure contains information about a hardware message sent to the system message queue. (This structure is used to store message information for the JournalPlaybackProc callback function.)
GIS	A Geographic Information System is a system for creating, storing, analyzing and managing spatial data and associated attributes. GIS is a tool that allows users to create interactive queries, analyze, and edit data.
GUI	Graphical User Interface
IDF	Inputlog Data File
IXF	Inputlog XML file (intermediate)
journalplayback	Public static final Hook.Descriptor JOURNALPLAYBACK Posts messages previously recorded by a JOURNALRECORD hook procedure.
journalrecord	Public static final Hook.Descriptor JOURNALRECORD Records input messages posted to the system message queue. (http://www.jniwrapper.com/docs/javadoc/winpack/com/jniwrapper/win32/hook/Hook.html)
scancodes	The data from a keyboard comes mainly in the form of scancodes, produced by key presses or used in the protocol with the computer. The PC keyboard interface is designed so the system software has maximum flexibility in defining certain keyboard operations. This is accomplished by having the keyboard return scancodes rather than ASCII codes. Each key generates a 'make' scancode when pressed and a 'break' scancode when released. The computer system interprets the scancodes to determine what operation it is to perform. (http://www.barcodeman.com/altek/mule/scandoc.php)
Virtual Keycode	Unique number that identifies one key
Windows Hook	A hook is a point in the system message-handling mechanism where an application can install a subroutine to monitor the message traffic in the system and process certain types of messages before they reach the target window procedure.
XML	Extensible Markup Language: which allows you to define the tags (markup) that you need to identify the data and text in XML documents.
XSL	XSL is a way of applying transformations and formatting to XML documents.

Appendix 2

Examples of outputs

label	action
.	interspace
BS	backspace (* number of keystrokes)
DEL	delete
SHIFT	shift on/off: this is visible in the output
Caps Lock	caps lock on/off: this is visible in the output
CTRL	control + key
ALT	alt + key
UP 5	up * number of lines up
DOWN 3	down * number of lines
LEFT 8	left * number of characters
RIGHT 8	right * number of characters
TAB	tab * number
ENTER	enter * number
HOME	home
END	end
PU	page up * number
PD	page down * number
INS	insert + character
Left Button	mouse click left
Movement	mouse movement (+ the start and end value of the xy-axis)
MouseWheel	scroll with the mouse scroll wheel & the Windows scrollbar (+ the start and end value of the xy-axis)
Right Button	mouse click right
Writing Mode	Input mode (1=keyboard, 2=Mouse, 3= Speech)
Output	Text that appears on the screen
Start Time	Time of key in: in milliseconds
End Time	Time of key up: in milliseconds
Action Time	Time between key in and key up: in milliseconds
Pause Time	Time between key in and key in: in milliseconds
Pause Location	Time of key up: hours:minutes.seconds,100th of a second
x value	Location of the mouse on x-axis
y value	Location of the mouse on y-axis

* The last two columns contain the values of the mouse. The location of the keystrokes is not detailed.

General Logging File

Session Identification	
Filename	FirstnameLastname_1.idf
Recording Time	28/01/2007 14:39:26
Participant	1
Session	4
Pause Treshold	500ms

WritingMode	Output	StartTime	EndTime	ActionTime	PauseTime	Pause Location	X	Y
2	Movement	15	32078	32063	15		426	313
2	Left Button	32172	32312	140	94		426	313
1	T	33687	33812	125	1515	7		
1	h	34093	34218	125	406			
1	e	34281	34422	141	188			
1	SPACE	34468	34593	125	187			
1	s	34843	35000	157	375			
1	c	35234	35468	234	391			
1	i	35422	35515	93	188			
1	e	35828	36031	203	406			
1	n	35984	36109	125	156			
1	t	36312	36468	156	328			
1	i	36500	36593	93	188			
1	s	36687	36843	156	187			
1	t	36906	37062	156	219			
1	ENTER	38015	38140	125	1109	3		
1	C	39218	39375	157	1203	3		
1	o	39531	39625	94	313			
1	l	39797	39890	93	266			
1	d	39968	40125	157	171			
1	p	40687	40797	110	719	1		
1	l	40922	41140	218	235			
1	a	41187	41328	141	265			
1	y	41281	41406	125	94			
1	SPACE	41672	41812	140	391			
1	(42312	42406	94	640	2		
1	NUM 2	42828	42953	125	516	2		
1	NUM 0	43125	43218	93	297			

WritingMode	Output	StartTime	EndTime	ActionTime	PauseTime	Pause Location	X	Y
1	NUM 0	43281	43390	109	156			
1	NUM 2	43500	43593	93	219			
1)	44531	44625	94	1031	2		
1	SPACE	45890	46031	141	1359	2		
1	LEFT	46750	46843	93	860	2		
1	UP	47953	48093	140	1203	7		
1	LEFT	48500	48609	109	547	7		
1	LEFT	48687	48781	94	187			
1	LEFT	48875	48953	78	188			
1	LEFT	49015	49109	94	140			
1	LEFT	49172	49297	125	157			
1	LEFT	49343	49422	79	171			
1	LEFT	49500	49593	93	157			
1	LEFT	49687	49765	78	187			
1	LEFT	49828	49937	109	141			
1	LEFT	50000	50109	109	172			
1	LEFT	50172	50297	125	172			
1	LEFT	50343	50422	79	171			
1	LEFT	50500	50609	109	157			
1	CTRL + B	52031	52140	109	1531	7		
1	RIGHT	53031	53172	141	1000	7		
1	RIGHT	53750	53843	93	719	7		
1	ENTER	54593	54703	110	843	4		
1	ENTER	55172	55281	109	579	4		
2	Movement	58047	63390	5343	2875	4	418	396
1	T	65265	65406	141	1875	4		
1	e	66156	66297	141	891	1		
1	l	67125	67218	93	969	1		
1	l	67312	67422	110	187			
1	SPACE	67547	67672	125	235			
1	m	67828	68000	172	281			
1	e	68047	68250	203	219			
1	SPACE	68187	68359	172	140			
1	y	68797	68906	109	610	2		
1	o	68968	69140	172	171			
1	u	69078	69218	140	110			
1	r	69812	69984	172	734	1		
1	SPACE	70093	70250	157	281			

WritingMode	Output	StartTime	EndTime	ActionTime	PauseTime	Pause Location	X	Y
1	s	70593	70718	125	500	2		
1	e	70859	71062	203	266			
1	c	71203	71359	156	344			
1	r	71468	71672	204	265			
1	e	71625	71812	187	157			
1	t	72062	72250	188	437			
1	s	72468	72625	157	406			
1	SPACE	72922	73062	140	454			
1	a	73250	73390	140	328			
1	n	73437	73625	188	187			
1	d	73578	73734	156	141			
1	SPACE	73687	73797	110	109			
1	a	75406	75562	156	1719	2		
1	s	75672	75843	171	266			
1	k	75797	75953	156	125			
1	SPACE	76718	76875	157	921	2		
1	m	77047	77234	187	329			
1	e	77172	77359	187	125			
1	SPACE	77312	77437	125	140			
1	y	77562	77672	110	250			
1	o	77734	77890	156	172			
1	u	77828	77968	140	94			
1	r	78281	78468	187	453			
1	SPACE	78422	78578	156	141			
1	q	79156	79281	125	734	2		
1	u	79406	79531	125	250			
1	e	79578	79656	78	172			
1	s	79718	79906	188	140			
1	t	79859	80031	172	141			
1	i	79984	80156	172	125			
1	o	80093	80265	172	109			
1	n	80203	80343	140	110			
1	s	81109	81265	156	906	1		
1	.	82047	82156	109	938	3		
1	SPACE	82312	82422	110	265			
1	O	82703	82828	125	391			
1	h	83172	83297	125	469			
1	SPACE	83390	83515	125	218			

Linear Logging File | periods of 60 seconds

Session Identification	
Filename	FirstnameLastname_1.idf
Recording Time	28/01/2007 14:39:26
Participant	1
Session	4
Pause Threshold	2000ms

Interval	Output
0:00:00	[Movement][LeftButton]The scientist[ENTER]Coldplay ·{NUM2}[NUM0][NUM0][NUM2]) {LEFT}[UP][LEFT13][CTRL+B][RIGHT2][ENTER2]{2875}[Movement]
0:01:00	Tell me your secrets and ask me your questions. Oh lets go back to the start. Running in circles, coming up tails. Heads on a silence apart. {Movement}
0:02:00	[LeftButton][ENTER2][UP2]{2187}Come up to meet you, tell you I'm sorry. ·You don't know how lovely you are. {3110}I had to find you, tell you I need you. {3860}Tell you I set you apart
0:03:00	. [BS]{2234}! {11219}[Movement][LeftButton][Movement][LeftButton][Movement][LeftButton][Movement]{4906}[LeftButton]{5906}
0:04:00	[LeftButton][Movement][ENTER2]{2329}Noboda[BS]y said it was easy, it4s [BS3]'s such a sho[BS]ame for us to part. {2890}Nobody said it was easy, no one ever said it would be this hard{4125}. {5016}Oh take me back to the start
0:05:00	. {4344}[ENTER2]{2719}[Movement][LeftButton][Movement][LeftButton][Movement]{10734} [LeftButton]{8578}[LeftButton][Movement][LeftButton][Movement][LeftButton]
0:06:00	[Movement][LeftButton]! {4594}was just gu{5515}essing at numbers and gi[BS2]figures. {5813}Pullu[BS]ing your puzzles apart. Questions of science, science and progra[BS]ess. {13578}
0:07:00	[Movement][LeftButton]{2828}[DEL]{3235}[LEFT4][DEL] Pulling your puzzles apart. ·Qestion[BS2]ions of science, sciences[BS] and progra[BS]ess.{11547} Do not p[BS]speak as I
0:08:00	oud as my heart{10797};[BS].{6281}[ENTER2]{18781}Tell me you love I[BS]me, come back and haunt me, Oh and I rush to the start
0:09:00	. Running in circles, chasing our tails. Coming back as we are. {14313}[ENTER2]{2000} Nobody said it was a[BS]eays, ·Of[BS]h it's such a shame for us to part{2093}. {2297}
0:10:00	Noby[BS]ody said it was easy, no ov[BS]ne ever said it would be w[BS]so hard. {4375}[Movement][ENTER2]I'm going back to the start.{3297}[Movement][RightButton][Movement][LeftButton]

Statistical Logging File

Session Identification	
Filename	FirstnameLastname_1.idf
Recording Time	28/01/2007 14:39:26
Participant	1
Session	4
Pause Treshold	500ms
Age	28
Sex	1
Experience	2
Group	3
task	2

Process Information	
Words	
Total Characters	761
Total Words	195
Average Word Length (WL)	3.815
Standard Deviation (WL)	1.844
Sentences	
Total Sentences	23
Average Characters/Sentence (C/S)	33.087
Standard Deviation (C/S)	17.056
Average Words/Sentence (W/S)	8.478
Standard Deviation (W/S)	5.133
Paragraphs	
Total Paragraphs	6
Average Characters/Paragraph (C/P)	126.833
Standard Deviation (C/P)	115.390
Average Words/Paragraph (W/P)	32.500
Standard Deviation (W/P)	27.970
Average Sentences/Paragraph (S/P)	3.833
Standard Deviation (S/P)	3.371

Process Time	
General	0:10:56
Number of Segments	193
Total Process Time (in seconds)	656.703
Average Process Time	3.403
Standard Deviation (Process Time)	5.453
Total Pause Time	0:05:32
Number of Pauses	192

Total Pause Time (in seconds)	332.582
Average Pause Time	1.732
Standard Deviation (Pause Time)	2.559
Active Writing Time	0:05:24
Total Writing Time (in seconds)	324.121
Average Writing Time (WT)	1.679
Standard Deviation (WT)	4.278

Writing Mode	
Keyboard	0:07:18
Total Time Keyboard (in seconds)	438.343
Number of Clusters Keyboard (CK)	7
Average Time (CK)	62.620
Standard Deviation (CK)	58.180
Number of Segments Keyboard (SK)	180
Average Time (SK)	2.435
Standard Deviation (SK)	2.488
Switches Keyboard to Mouse	7
Mouse	0:03:38
Total Time Mouse (in seconds)	218.360
Number of Clusters Mouse (CM)	8
Average Time (CM)	27.295
Standard Deviation (CM)	20.694
Number of Segments Mouse (SM)	13
Average Time (SM)	16.797
Standard Deviation (SM)	13.227
Switches Mouse to Keyboard	7

Pause Logging File

Session Identification	
Filename	FirstnameLastname_1.idf
Recording Time	28/01/2007 14:39:26
Participant	1
Session	4
Pause Treshold	1000ms

General Information	
Total Process Time	0:10:56
Interval Length	0:01:05
Interval Length (in seconds)	65.671
Total Number Of Pauses	84
Total Pause Time	0:04:17
Total Pause Time (in seconds)	257.625
Total Mean Time (MT)	3.067
Standard deviation (MT)	3.441

Pause Location	
Within Words (1)	
Number of Pauses (WW)	6
Mean Pause Time (WW)	1.945
Standard Deviation (WW)	1.766
Between Words (2)	
Number of Pauses (BW)	19
Mean Pause Time (BW)	1.961
Standard Deviation (BW)	2.282
Between Sentences (3)	
Number of Pauses (BS)	31
Mean Pause Time (BS)	3.001
Standard Deviation (BS)	3.280
Between Paragraphs (4)	
Number of Pauses (BP)	14
Mean Pause Time (BP)	4.679
Standard Deviation (BP)	5.302
Initial Pauses (5)	
Number of Pauses (IP)	0

Mean Pause Time (IP)	
Standard Deviation (IP)	
End Pauses (6)	
Number of Pauses (EP)	1
Mean Pause Time (EP)	3.297
Standard Deviation (EP)	0.000
Undefined Pauses (7)	
Number of Pauses (UP)	13
Mean Pause Time (UP)	3.604
Standard Deviation (UP)	3.089

Summary per Interval	
Interval 1	
Number of Pauses (I1)	10
Mean Pause Time (I1)	1.470
Standard Deviation (I1)	0.562
Interval 2	
Number of Pauses (I2)	7
Mean Pause Time (I2)	1.449
Standard Deviation (I2)	0.404
Interval 3	
Number of Pauses (I3)	12
Mean Pause Time (I3)	2.490
Standard Deviation (I3)	2.898
Interval 4	
Number of Pauses (I4)	4
Mean Pause Time (I4)	3.621
Standard Deviation (I4)	2.139
Interval 5	
Number of Pauses (I5)	8
Mean Pause Time (I5)	4.090
Standard Deviation (I5)	2.994
Interval 6	
Number of Pauses (I6)	6
Mean Pause Time (I6)	4.490
Standard Deviation (I6)	2.864
Interval 7	
Number of Pauses (I7)	12

Mean Pause Time (I7)	2.745
Standard Deviation (I7)	3.474
Interval 8	
Number of Pauses (I8)	7
Mean Pause Time (I8)	7.279
Standard Deviation (I8)	6.775
Interval 9	
Number of Pauses (I9)	11
Mean Pause Time (I9)	2.642
Standard Deviation (I9)	3.884
Interval 10	
Number of Pauses (I10)	7
Mean Pause Time (I10)	2.259
Standard Deviation (I10)	1.200

References

- Adams, J. W., Hitch, G., & Hutton, U. (1997). Working memory and children's mental addition. *Journal of Experimental Child Psychology*, 67, 21-38.
- Aho, A., Sethi, R., & Ullman, J. (1986). *Compilers: Principles, Techniques and Tools*. Reading, Massachusetts: Addison-Wesley
- Alamargot, D., Dansac, C., Ros, C., & Chuy, M. (2005). Rédiger un texte procédural à partir de sources: Relations entre l'empan de production écrite et l'activité oculaire du scripteur. In D. Alamargot, P. Terrier & J. M. Cellier (Eds.), *Production, compréhension et usage des écrits techniques au travail* (pp. 51-68). Toulouse: Octarès.
- Andersson, B., & et al. (2006). Combining keystroke logging with eye-tracking. In L. Van Waes, M. Leijten & C. Neuwirth (Eds.), *Writing and Digital Media* (Vol. 17, pp. 166-172). Oxford: Elsevier.
- Baddeley, A. D. (1986). *Working memory*. Oxford: Oxford University Press.
- Baddeley, A. D., & Hitch, G. (1974). Working memory. In G. A. Bower (Ed.), *Recent advances in learning and motivation* (Vol. 8, pp. 47-90). New York: Academic Press.
- Bangert-Drowns, R. L. (1993). The word processor as an instructional tool: A meta-analysis of word processing in writing instruction. *Research in the Teaching of English*, 63(1), 69-93.
- Bereiter, C., & Scardamalia, M. (1987). *The psychology of written composition*. Hillsdale, NJ: Lawrence Erlbaum Associates.
- Bernstein, L. (1990). *Developing an adequately specified model of state level student achievement with multilevel data*. Paper presented at the American Educational Association
- Blau, S. (1983). Invisible writing: Investigating cognitive processes in writing. *College, Composition and Communication*, 34, 297-312.
- Bourdin, B., & Fayol, M. (1994). Is written language production more difficult than oral language production? A working memory approach. *International Journal of Psychology*, 29, 591-620.
- Bridwell, L. S., & Duin, A. H. (1985). Looking in-depth at writers: Computers as writing medium and research tool. In J. L. Collins & E. A. Sommers (Eds.), *Writing On-Line* (pp. 115-121). Upper Montclair, NJ: Boynton/Cook.
- Caporossi, G., Alamargot, D., & Chesnet, D. (2004). Using the computer to study the dynamics of handwriting processes. *Lecture Notes in Computer Science*, 3245(242-254).
- Chenoweth, N. A., & Hayes, J. R. (2003). The Inner Voice in Writing. *Written communication*, 20(1), 99-118.
- Crossman, W. (2004). *VIVO. [Voice-In/Voice-Out] The coming age of talking computers*. Oakland: Regent Press.
- Daiute, C. A. (1986). Physical and cognitive factors in revising: Insight from studies with computers. *Research in the Teaching of English*, 20, 141-159.
- Daneman, M., & Carpenter, P. A. (1980). Individual differences in working memory and reading. *Journal of Verbal Learning and Verbal Behaviour*, 19, 450-466.
- De Maeyer, S., & Rymenans, R. (2004). *Onderzoek naar kenmerken van effectieve scholen: Kritische factoren in een onderzoek naar schooleffectiviteit in het technisch en beroepssecundair onderwijs in Vlaanderen. [Study of the characteristics of effective schools: Critical factors in school effectivity of secondary education in Flanders]* University of Antwerp, Antwerp.

- Donnelly, C., & Stallman, R. (1995). *Bison: The YACC-compatible parser generator. Version 1.25*.
- Ericsson, K. A., & Kintsch, W. (1995). Long-Term Working Memory. *Psychological Review*, 102(2), 211-245.
- Ericsson, K. A., & Simon, H. A. (1980). Verbal reports as data. *Psychological Review*, 87, 215-251.
- Faigley, L., & Witte, S. (1981). Analyzing revision. *College Composition and Communication*, 32, 400-414.
- Feng, J., Karat, C., & Sears, A. (2005). How productivity improves in hands-free continuous dictation tasks: Lessons learned from a longitudinal study. *Interacting with computers*, 17, 265-289.
- Field, A. (2005). *Discovering statistics using SPSS*. London/Thousand Oaks/New Dehli Sage Publications.
- Fisk, A. D., Derrick, W. L., & Schneider, W. (1986-87). A methodological assessment and evaluation of dual-task paradigms. *Current Psychology Research & Reviews*, 5, 315-327.
- Flower, L., & Hayes, J. R. (1980). The dynamics of composing: Making plans and juggling constraints. In L. Gregg & E. Steinberg (Eds.), *Cognitive processes in writing* (pp. 31-50). Hillsdale, NJ: L. Erlbaum.
- Flower, L., & Hayes, J. R. (1981). A cognitive process theory of writing. *College Composition and Communication*, 32, 365-387.
- Flower, L., & Hayes, J. R. (1985). 'Talking about protocols'. *College Composition and Communication*, 37, 16-54.
- Flower, L., Hayes, J. R., Carey, L., Schriver, K., & Stratman, J. (1986). Detection, diagnosis and the strategies of revision. *College, Composition and Communication*, 37, 16-55.
- Galbraith, D. (1996). Self-monitoring, discovery through writing and individual differences in drafting strategy. In G. Rijlaarsdam, H. Van den Bergh & M. Couzijn (Eds.), *Theories, Models and Methodology in Writing Research* (Vol. 1, pp. 121-141). Amsterdam: Amsterdam University Press.
- Galbraith, D. (1999). Writing as a knowledge constituting process. In M. Torrance & D. Galbraith (Eds.), *Knowing what to Write: Conceptual Processes in text Production* (pp. 139-160). Amsterdam: Amsterdam University Press.
- Galbraith, D., & Torrance, M. (2004). Revision in the context of different drafting strategies. In L. Allal, L. Chanquoy & P. Largy (Eds.), *Revision: Cognitive and Instructional Processes* (pp. 63-86). Dordrecht: Kluwer Academic Publishers.
- Galbraith, D., Torrance, M., & Hallam, J. (in preparation). Effects of writing on conceptual coherence. University of Staffordshire.
- Goldberg, A., Russell, M., & Cook, A. (2003). The effect of computers on student writing: A meta-analysis of studies from 1992 to 2002. *Journal of Technology, Learning, and Assessment*, 2(1), 1-24.
- Goldstein, H. (1995). *Multilevel statistical analysis*. London: Edward Arnold.
- Gould, J. D. (1978). How experts dictate. *Journal of Experimental Psychology: Human Perception and Performance*, 4(4), 648-661.
- Gould, J. D. (1981). Composing letters with computer-based text editors. *Human Factors*, 23(5), 593-606.
- Gould, J. D., & Alfaro, L. (1984). Revising documents with text editors, hand-writing recognition systems and speech-recognition systems. *Human Factors*, 26(4), 91-406.
- Grabowski, J. (1996). Writing and speaking: Common grounds and differences towards a regulation theory of written language production. In C. M. Levy & S. E. Ransdell (Eds.), *The science of writing: Theories, methods, individual differences, and applications* (pp. 73-92). Mahwah, NJ: Lawrence Erlbaum Associates.
- Greene, S., & Higgins, L. (1994). 'Once upon a time': the use of retrospective accounts in building theory in composition. In P. Smagorinsky (Ed.), *Speaking about Writing* (pp. 115-140). London: Sage.
- Haas, C. (1989a). Does the medium make the difference? Two studies of writing with pen and paper and with computers. *Human-Computer Interaction*, 10, 149-169.
- Haas, C. (1989b). How the writing medium shapes the writing process: Effects of word processing on planning. *Research in the Teaching of English*, 23, 181-207.
- Haas, C. (1989c). 'Seeing it on the screen isn't really seeing it': Computer writers' reading problems. In G. E. Hawisher & C. L. Selfe (Eds.), *Critical perspectives on computers* (pp. 16-29). New York: Teachers College Press.
- Haas, C. (1996). *Writing Technology: Studies on the materiality of literacy*. Mahwah, NJ: Lawrence Erlbaum.
- Hacker, D. J. (1994). Comprehension monitoring as a writing process. *Advances in Cognition and Educational Practice*, 6, 143-172.
- Hacker, D. J. (1997). Comprehension monitoring of written discourse across early-to-middle adolescence. *Reading and Writing*, 9(3), 207-240.
- Hacker, D. J., Plumb, C. S., Butterfield, E. C., Quathamer, D., & Heineken, E. (1994). Text revision: Detection and correction of errors. *Journal of Educational Psychology*, 86(1), 65-78.
- Halverson, A., Horn, D. B., Karat, C., & Karat, J. (1999). *The beauty of errors: Patterns of error correction in desktop speech systems*. Paper presented at the Human-Computer Interaction — INTERACT '99,

- Edinburgh.
- Hartley, J. (2007). Longitudinal studies of the effects of new technologies on writing: Two case-studies. In M. Torrance, L. Van Waes & D. Galbraith (Eds.), *Writing and Cognition: Methods and Applications* (Vol. 20, pp. 293-306). Oxford: Elsevier.
- Hartley, J., Howe, M., & McKeachie, M. (2001). Writing through time: Longitudinal studies of the effects of new technology on writing. *British Journal of Educational Technology*, 32(2), 141-151.
- Hartley, J., Sotto, E., & Pennebaker, J. (2003). Speaking versus typing: A case-study of the effects of using voice-recognition software on academic correspondence. *British Journal of Educational Technology*, 34(1), 5-16.
- Hayes, J. R. (1996). A new framework for understanding cognition and affect in writing. In C. M. Levy & S. E. Ransdell (Eds.), *The science of writing: Theories, methods, individual differences, and applications* (pp. 1-27). Mahwah: New Jersey: Lawrence Erlbaum Associates.
- Hayes, J. R., & Chenoweth, N. (2006). Is Working Memory Involved in the Transcribing and Editing of Texts? *Written Communication*, 23(2), 135.
- Hayes, J. R., & Flower, L. (1986). Writing research and the writer. *American Psychologist*, 41, 1106-1113.
- Hayes, J. R., Flower, L., Schriver, K., Statman, J., & Carey, L. (1987). Cognitive processes in revision. In S. Rosenberg (Ed.), *Reading, writing, and language possessing* (Vol. 2, pp. 176-240). Cambridge: Cambridge University Press.
- Hayes, J. R., & Hayes, L. S. (1980). Identifying the organization of writing processes. In L. W. Gregg & E. R. Steinberg (Eds.), *Cognitive Processes in writing* (pp. 3-30). Mahwah, New Jersey: Lawrence Erlbaum Associates.
- Holmqvist, K., Johansson, V., Strömquist, S., & Wengelin, Å. (2002). Studying reading and writing online. In S. Strömquist (Ed.), *The diversity of languages and language learning* (pp. 103-123). Lund: Centre for Languages and Literature, Lund University.
- Honeycutt, L. (2003). Researching the use of voice recognition writing software. *Computers and Composition*, 20, 77-95.
- Honeycutt, L. (2004). Literacy and the writing voice: The intersection of culture and technology in dictation. *Journal of Business and Technical Communication*, 18(3), 294-327.
- Hoskyn, M., & Swanson, H. (2003). The relationship between working memory and writing in younger and older adults. *Reading and Writing*, 16, 759-784.
- Hox, J. (2002). *Multilevel analysis: Techniques and applications*. Mahwah, New Jersey/London: Lawrence Erlbaum Associates Publishers.
- Janssen, D., Van Waes, L., & Van den Bergh, H. (1996). Effects of thinking aloud on writing processes. In C. M. Levy & S. Ransdell (Eds.), *The science of writing: Theories, individual differences, and applications* (pp. 233-250). Mahwah, NJ: Lawrence Erlbaum Associates.
- Johnson-Laird, P. N. (1983). *Mental models: Towards a cognitive science of language, inference, and consciousness*. Cambridge, MA: Harvard University Press.
- Joram, E., Woodruff, E., Lindsay, P., & Bryson, M. (1990). Students' editing skills and attitudes toward word processing. *Computers and Composition*, 7, 55-72.
- Just, M. A., & Carpenter, P. A. (1992). A capacity theory of comprehension: Individual differences in Working Memory. *Psychological Review*, 99(1), 122-149.
- Karat, C., Halverson, C., Horn, D., & Karat, J. (1999). *Patterns of entry and correction in large vocabulary continuous speech recognition systems*. Paper presented at the CHI 99, Pittsburgh.
- Karat, J., Horn, D. B., Halverson, C. A., & Karat, C. (2000). *Overcoming Unusability. Developing strategies in speech recognition systems*. Paper presented at the Conference on Human Factors in Computing Systems, CHI 2000, Den Haag.
- Kellogg, R. T. (1988). Attentional overload and writing performance: Effects of rough draft and outline strategies. *Journal of Experimental Psychology: Learning, memory and cognition*, 14(2), 355-365.
- Kellogg, R. T. (1994). *The Psychology of Writing*. New York: Oxford University Press.
- Kellogg, R. T. (1996). A model of working memory in writing. In C. M. Levy & S. E. Ransdell (Eds.), *The Science of Writing: Theories, methods, individual differences and applications* (pp. 57-71). Hillsdale, NJ: Lawrence Erlbaum.
- Kellogg, R. T. (1999). Components of Working Memory in Text Production. In M. Torrance & G. Jeffery (Eds.), *The cognitive demands of writing: processing capacity and Working Memory effects in text production* (Vol. 3, pp. 43-61). Amsterdam: Amsterdam University Press.
- Kellogg, R. T. (2001). Competition for working memory among writing processes. *American Journal of Psychology*, 114, 175-191.
- Kellogg, R. T. (2004). Working memory components in written sentence generation. *American Journal of Psychology*, 117, 341-361.
- Kieft, M., Rijlaarsdam, G., Galbraith, D., & Van den Bergh, H. (in preparation). The effects of students individual characteristics and writing instruction on writing performance. University of Amsterdam.
- Kieft, M., Rijlaarsdam, G., Galbraith, D., & Van den Bergh, H. (to appear). The effects of adapting a

- writing course to students' writing strategies. *British Journal of Educational Psychology*.
- Kolb, D. A. (1984). *Experiential Learning*. Englewood Cliffs, New Jersey: Prentice Hall Inc.
- Kollberg, P. (1998). *S-Notation - a computer based method for studying and representing text composition*. Unpublished Lic. Thesis, Stockholm University, Stockholm.
- Larigauderie, P., Gaonac'h, D., & Lacroix, N. (1998). Working memory and error detection in texts: What are the roles of the central executive and the phonological loop? *Applied Cognitive Psychology*, 12, 505-527.
- Lee, Y. J. (2002). A comparison of composing processes and written products in timed-essay tests across paper-and-pencil and computer modes. *Assessing Writing*, 8(2), 135-157.
- Leijten, M. (2007). How do writers adapt to speech recognition software? The influence of learning styles on writing processes in speech technology environments. In M. Torrance, L. Van Waes & D. Galbraith (Eds.), *Writing and Cognition: Research and Applications* (Vol. 20, pp. 279-292). Oxford: Elsevier.
- Leijten, M., De Maeyer, S., Ransdell, S., & Van Waes, L. (in preparation). Isolating the effects of writing mode from error span, input type, and lexicality when correcting text production errors: a multilevel analysis. University of Antwerp.
- Leijten, M., Janssen, D., & Van Waes, L. (in preparation). The text produced so far in business texts: error correction strategies by professional speech recognition users. University of Antwerp.
- Leijten, M., Ransdell, S., & Van Waes, L. (submitted). Isolating the effects of writing mode from error span, input type, and lexicality when correcting text production errors. University of Antwerp.
- Leijten, M., & Van Waes, L. (2003a). Schrijven zoals je spreekt, spreken zoals je schrijft: De invloed van spraakherkenning op het schrijfproces van dicteerders met verschillende leerstijlen [Writing as you talk, talking as you write: The influence of speech recognition on the writing process of dictators with different learning styles]. *Tijdschrift voor Taalbeheersing*, 25(4), 325-341.
- Leijten, M., & Van Waes, L. (2003b). *The writing processes and learning strategies of initial users of speech recognition: a case study on the adaption process of two professional writers* (Research Report No. 2003022). Antwerp: University of Antwerp.
- Leijten, M., & Van Waes, L. (2005a). *Inputlog: A logging tool for the research of writing processes* (Research Papers, Faculty of Applied Economics). Antwerp: University of Antwerp.
- Leijten, M., & Van Waes, L. (2005b). Writing with speech recognition: The adaptation process of professional writers with and without dictating experience. *Interacting with Computers*, 17(6), 736-772.
- Leijten, M., & Van Waes, L. (2006a). Inputlog: New perspectives on the logging of on-line writing processes in a Windows environment. In K. P. H. Sullivan & E. Lindgren (Eds.), *Computer Key-Stroke Logging and Writing: Methods and Applications*. (Vol. 18, pp. 73-94). Oxford: Elsevier.
- Leijten, M., & Van Waes, L. (2006b). Repair strategies in writing with speech recognition: The effect of experience with classical dictating. In L. Van Waes, M. Leijten & C. Neuwirth (Eds.), *Writing and Digital Media* (Vol. 17, pp. 31-46). Oxford: Elsevier.
- Levitt, W. J. M. (1983). Monitoring and self-repair in speech. *Cognition*, 14(1), 41-104.
- Levine, J. R., Mason, T., & Brown, D. (1992). *Lex & Yacc* (2nd ed.).
- Levy, C. M., & Marek, P. (1999). Testing components of Kellogg's multicomponent models of Working Memory in writing: The role of the phonological loop. In M. Torrance & G. Jeffery (Eds.), *The cognitive demands of writing. Processing capacity and Working Memory effects in text production*. (Vol. 3, pp. 25-41). Amsterdam: Amsterdam University Press.
- Levy, C. M., & Ransdell, S. E. (1996). Writing signatures. In C. M. Levy & S. E. Ransdell (Eds.), *The Science of Writing: Theories, Methods, Individual Differences, and Applications* (pp. 149-162). Mahwah, NJ: Lawrence Erlbaum.
- Levy, C. M., & Ransdell, S. E. (2001). Writing with concurrent memory loads. In T. Olive & C. M. Levy (Eds.), *Contemporary Tools and Techniques for Studying Writing* (pp. 9-29). Dordrecht: Kluwer Academic Publishers.
- Lindgren, E. (2002). The LS graph: A methodology for visualising writing revision. *Language Learning*, 52(3), 565-595.
- Lindgren, E. (2005). *Writing and Revising: Didactic and Methodological Implications of Keystroke Logging*. (Skrifter från moderna språk, No. 18). Umeå, Sweden: Umeå University, Department of Modern Languages.
- Lindgren, E. (2007). GIS for writing: Applying geographic information system techniques to data-mine writing's cognitive processes. In M. Torrance, L. Van Waes & D. Galbraith (Eds.), *Writing and Cognition: Methods and Applications* (Vol. 20, pp. 83-96). Oxford: Elsevier.
- Lindgren, E., & Sullivan, K. P. H. (2003). Stimulated recall as a trigger for increasing noticing and language awareness in the L2 writing classroom: A case study of two young female writers. *Language Awareness*, 12, 172-186.
- Lindgren, E., & Sullivan, K. P. H. (2006a). Analysing on-line revision. In K. P. H. Sullivan & E. Lindgren (Eds.), *Computer Keystroke Logging: Methods and Applications* (Vol. 18, pp. 157-188). Oxford: Elsevier.

- Lindgren, E., & Sullivan, K. P. H. (2006b). Analyzing on-line revision. In K. P. H. Sullivan & E. Lindgren (Eds.), *Computer Key-Stroke Logging: Methods and Applications* (Vol. 18, pp. 157-188). Oxford: Elsevier.
- MacArthur, C. A. (2006). Assistive technology for writing: Tools for struggling writers. In L. Van Waes, M. Leijten & C. Neuwirth (Eds.), *Writing and Digital Media* (Vol. 17, pp. 11-20). Oxford: Elsevier.
- Matsuhashi, A. (1982). Explorations in real-time production of written discourse. In M. Nystrand (Ed.), *What writers know. The language, process, and structure of written discourse* (pp. 269-290). New York: Academic Press.
- Matsuhashi, A. (1987). Revising the plan and altering the text. In A. Matsuhashi (Ed.), *Writing in real time: Modelling production processes* (pp. 197-223). New York: Academic Press.
- Mazeland, H. (2003). *Inleiding in de conversatieanalyse [Introduction to conversation analysis]*. Bussum: Coutinho.
- McCutchen, D. (1996). A capacity theory of writing: Working memory in composition. *Educational Psychology Review*, 8(3), 299-325.
- Nottbusch, G., Weingarten, R., & Sahel, S. (2007). From written word to written sentence production. In M. Torrance, L. Van Waes & D. Galbraith (Eds.), *Writing and Cognition: Research and applications* (Vol. 20, pp. 31-54). Amsterdam: Elsevier.
- Olive, T. (2004). Memory in writing: Empirical evidences from the dual-task technique working. *European Psychologist*, 9(1), 32-42
- Olive, T., & Kellogg, R. T. (2002). Concurrent activation of high- and low-level production processes in written composition. *Memory and Cognition*, 30, 594-600.
- Olive, T., & Piolat, A. (2002). Suppressing visual feedback in written composition : Effects on processing demands and coordination of the writing processes. *International Journal of Psychology*, 37(4), 209-218.
- Paxson, V. (1995). *Flex: A fast scanner generator, edition 2.5*.
- Perrin, D. (2003). Progression analysis: Investigating writing strategies at the workplace. *Journal of Pragmatics*, 35(6), 907-921.
- Pilotti, M., Chodorow, M., & Thornton, K. C. (2004). Error detection in text: Do feedback and familiarity help? *The Journal of General Psychology*, 131(4), 242-266.
- Piolat, A., Roussey, J. Y., Olive, T., & Amada, M. (2004). Processing time and cognitive effort in revision: effects of error type and of working memory capacity. In L. Allal, L. Chanquoy, P. Largy & Y. Rouiller (Eds.), *Revision: Cognitive and Instructional Processes* (pp. 21-38). Dordrecht: Kluwer Academic Publishers.
- Python, (2007). Retrieved February 12, from <http://www.python.org/>
- Quené, H., & Van den Bergh, H. (2004). On Multi-Level Modeling of data from repeated measures designs: A tutorial. *Speech Communication*, 43(1-2), 103-121.
- Quinlan, T. (2002). *Speech recognition technology and the writing processes of students with writing difficulties*. Paper presented at the Earli Sig Writing 2002, Stafford, England.
- Quinlan, T. (2004). Speech recognition technology and students with writing difficulties: Improving fluency. *Journal of Educational Psychology*, 96, 337-346.
- Quinlan, T. (2006). Young Writers and Digital Scribes. In L. Van Waes, M. Leijten & C. Neuwirth (Eds.), *Writing and Digital Media* (Vol. 17, pp. 21-29). Oxford: Elsevier.
- Quinlan, T., Loncke, M., Leijten, M., & Van Waes, L. (in preparation). Writers juggling error-correcting with sentence generation. Educational Testing Service, University of Ghent, University of Antwerp.
- Rabbitt, P. (1978). Detection of errors by skilled typists. *Ergonomics*, 21, 945-958.
- Rabbitt, P., Cummings, P., & Vyas, S. (1978). Some errors of perceptual analysis in visual search can be detected and corrected. *Quarterly Journal of Experimental Psychology*, 30, 417-427.
- Ransdell, S. E., & Hecht, S. A. (2003). Time and resource limits on working memory: Cross-age consistency in counting span performance. *Journal of Experimental Child Psychology*, 86, 303-313.
- Ransdell, S. E., & Levy, C. M. (1996). Working memory constraints on writing quality and fluency. In C. M. Levy & S. E. Ransdell (Eds.), *The science of writing: Theories, Methods, Individual Differences, and Applications* (pp. 93-105). Mahwah, NJ: Lawrence Erlbaum Associates.
- Ransdell, S. E., & Levy, C. M. (1999). Writing reading and speaking memory spans and the importance of resource flexibility. In M. Torrance & G. Jeffery (Eds.), *The cognitive demands of writing: processing capacity and working memory effects in text production*. Amsterdam: Amsterdam University Press.
- Ransdell, S. E., Levy, C. M., & Kellogg, R. T. (2002). The structure of writing processes as revealed by secondary task demand. *L1-Educational Studies in Language and Literature* 2(2), 141-163.
- Rashbash, J., Steele, F., Browne, W., & Prosser, B. (2004). A user's guide to MLwiN Version 2.0. Retrieved February 6 2007, 2007
- Reece, J., & Cumming, G. (1996). Evaluating speech-based composition methods: Planning, dictation, and the listening word processor. In C. M. Levy & S. E. Ransdell (Eds.), *The science of writing: Theories, Methods, Individual Differences, and Applications* (pp. 361-380). Mahwah, New Jersey: Lawrence Erlbaum Associates.

- Rodman, R. D. (1999). *Computer speech technology*. London: Artech House Publishers.
- Sadowski, M., Kealy, W. A., Goetz, E. T., & Paivio, A. (1997). Concreteness and imagery effects in the written composition of definitions. *Journal of Experimental Psychology*, *89*, 518-526.
- Schegloff, E., Jefferson, G., & Sacks, H. (1977). The Preference for Self-Correction in the Organization of Repair in Conversation. *Language*, *53*, 361-382.
- Schilperoord, J. (1996). *It's about time: Temporal aspects of cognitive processes in text production*. Amsterdam/Atlanta: Rodopi.
- Schilperoord, J. (2002). On the cognitive status of pauses in discourse production. In T. Olive & C. M. Levy (Eds.), *Contemporary tools and techniques for studying writing* (Vol. 10, pp. 61-88). Dordrecht/Boston/London: Kluwer Academic Publishers.
- Selzer, J. (1983). The composing process of an engineer. *College Composition and Communication*, *35*, 178-188.
- Selzer, J. (1984). Exploring options in composing. *College Composition and Communication*, *35*, 276-284.
- Severinson Eklundh, K. S. (1994). Linear and Non-linear strategies in computer-based writing. *Computers and Composition*, *11*, 203-216.
- Severinson Eklundh, K. S., & Kollberg, P. (1992). *Translating keystroke records into a general notation for the writing process* (IPLab-59). Stockholm: Department of Numerical Analysis and Computing Science, Royal Institute of Technology.
- Severinson Eklundh, K. S., & Kollberg, P. (1996a). Computer tools for tracing the writing process: From keystroke records to S-notation. In G. Rijlaarsdam, H. Van den Bergh & M. Couzijn (Eds.), *Models and Methodology in writing research* (pp. 526-541). Amsterdam: Amsterdam University Press.
- Severinson Eklundh, K. S., & Kollberg, P. (1996b). Computer tools for tracing the writing process: From keystroke records to S-notation. In G. Rijlaarsdam, H. Van den Bergh & M. Couzijn (Eds.), *Theories, Models and Methodology in writing research* (pp. 526-541). Amsterdam: University Press.
- Severinson Eklundh, K. S., & Kollberg, P. (2003). Emerging discourse structure: Computer-assisted episode analysis as a window to global revision in university students' writing. *Journal of Pragmatics*, *35*, 869-891.
- Shah, P., & Miyake, A. (1996). The separability of working memory resources for spatial thinking and language processing: An individual differences approach. *Journal of Experimental Psychology: General*, *125*, 4-27.
- Sharples, M. (1999). *How We Write: Writing as Creative Design*. London: Routledge.
- Sharples, M., & Pemberton, L. (1992). Representing writing: External representations and the writing process. In P. O'Brien Holt & N. Williams (Eds.), *Computers and Writing: State of the art* (pp. 319-336). Dordrecht: Kluwer Academic Publishers.
- Smagorinsky, P. (1989). The reliability and validity of protocol analysis. *Written communication*, *6*, 463-479.
- Snijders, T. A. B., & Bosker, R. J. (1999). *Multilevel analysis: An introduction to basic and advances multilevel modeling*. London/Thousand Oaks/New Dehli: Sage Publications.
- Snyder, M. (1974). Self-monitoring of expressive behavior. *Journal of Personality and Social Psychology*, *30*(4), 526-537.
- Spelman Miller, K., & Sullivan, K. P. H. (2006). Keystroke logging – an introduction. In K. P. H. Sullivan & E. Lindgren (Eds.), *Computer Keystroke Logging and Writing: Methods and Applications*. Amsterdam: Elsevier.
- Sternberg, S. (1969). The discovery of processing stages: Extensions of Donders's method. *Acta Psychologica*, *30*, 276-235.
- Stifelman, L. J. (1993). *User repairs of speech recognition errors: An intonational analysis*. Massachusetts: MIT Media Laboratory Technical Report.
- Strömquist, S., Holmqvist, K., Johansson, V., Karlsson, H., & Wengelin, Å. (2006). What key-logging can reveal about writing. In K. P. H. Sullivan & E. Lindgren (Eds.), *Computer Key-stroke Logging and Writing: Methods and Applications*. Amsterdam: Elsevier.
- Strömquist, S., & Karlsson, H. (2001). *ScriptLog for Windows: Users manual*. Lund: University of Lund.
- Strömquist, S., & Malmsten, L. (1997). *ScriptLog Pro User's manual*. Göteborg: Göteborg University, Dept of Linguistics.
- Sullivan, K. P. H., & Lindgren, E. (2006). *Computer Key-Stroke Logging and Writing*. Oxford: Elsevier Science.
- Terrel, S. R. (2002). The effect of learning style on doctoral course completion in a Web-based learning environment. *Internet and Higher Education*, *5*, 345-352.
- Torrance, M., & Galbraith, D. (2006). The processing demands of writing. In C. A. MacArthur, S. Graham & J. Fitzgerald (Eds.), *Handbook of writing research* (pp. 468). New York: Guilford Publications.
- Towse, J., Hitch, G., & Hutton, U. (1998). A reevaluation of working memory capacity in children. *Journal of Memory and Language*, *39*, 195-217.
- Van den Bergh, H., & Rijlaarsdam, G. (1996). The dynamics of composing: Modelling writing process

- data. In C. M. Levy & S. E. Ransdell (Eds.), *The science of writing: Theories, methods, individual differences, and applications* (pp. 207-232). Mahwah, NJ: Lawrence Erlbaum Associates.
- Van den Haak, M. J., De Jong, M. D. T., & Schellens, P. J. (2003). Retrospective versus concurrent think-aloud protocols: Testing the usability of an online library catalogue. *Behaviour and Information Technology*, 22(5), 339-351.
- Van den Haak, M. J., De Jong, M. D. T., & Schellens, P. J. (2004). Employing think-aloud protocols and constructive interaction to test the usability of online library catalogues: A methodological comparison. *Interacting with Computers*, 16, 1153-1170.
- Van den Haak, M. J., De Jong, M. D. T., & Schellens, P. J. (2006). Hardopdenkprotocollen en gebruikersonderzoek: Volledigheid en reactiviteit van de synchrone hardopdenkmethode. *Tijdschrift voor Taalbeheersing*, 28(3), 185-197.
- Van der Geest, T., Leijten, M., & Van Waes, L. (2006). Taalproductie en -verwerking onderzoeken met de computer [Researching language production and processing with the computer]. *Tijdschrift voor Taalbeheersing*, 28(3), 181-185.
- Van Someren, M. W., Barnard, Y. F., & Sandberg, J. A. C. (1994). *The think aloud method: A practical guide to modelling cognitive processes*. London: Academic Press.
- Van Waes, L. (1991). *De computer en het schrijfproces: De invloed van de tekstverwerker op het pauze- en revisiegedrag van schrijvers [The computer and the writing process: The influence of the word processor on the pausing and revision behavior of writers]*. Enschede: VMW (PhD thesis).
- Van Waes, L., & Leijten, M. (2006). Logging writing processes with Inputlog. In L. Van Waes, M. Leijten & C. Neuwirth (Eds.), *Writing and Digital Media* (Vol. 17, pp. 158-166). Oxford: Elsevier.
- Van Waes, L., & Schellens, P. J. (2003). Writing profiles: The effect of the writing mode on pausing and revision patterns of experienced writers. *Journal of Pragmatics*, 35(6), 829-853.
- Weingarten, R., Nottbusch, G., & Will, U. (2004). Morphemes, syllables and graphemes in written word production. In T. Pechmann & C. Habel (Eds.), *Multidisciplinary approaches to speech production* (pp. 529-572). Berlin: Mouton de Gruyter.
- Wengelin, A. (2006). Examining pauses in writing: Theories, methods and empirical data. In K. P. H. Sullivan & E. Lindgren (Eds.), *Computer Key-Stroke logging and Writing: Methods and Applications* (Vol. 18, pp. 107-130). Oxford: Elsevier.
- Williams, N., Hartley, P., & Pittard, V. (2004). Talking to write: Investigating the practical impact and theoretical implications of speech recognition (SR) software on real writing tasks. In L. Van Waes, D. Galbraith & M. Torrance (Eds.), *Recent developments in writing process research* (Vol. 2): Kluwer.
- Wood, E., Willoughby, T., Specht, J., & Porter, L. (2002). An examination of how a cross-section of academics use computer technology when writing academic papers. *Computers & Education*, 38, 287-301.