Revista Romana de Interactiune Om-Calculator 8 (4) 2015, 281-303

Visualization of computer-supported argumentative writing processes using C-SAW

Kalliopi Benetos[1], Mireille Bétrancourt[2]

[1] TECFA, Faculty of Psychology and Educational Science - University of Geneva
Bd du Pont-d'Arve 42, 1205 Geneva, Switzerland
E-mail: kalliopi.benetos@unige.ch

[2] TECFA, Faculty of Psychology and Educational Science - University of Geneva
Bd du Pont-d'Arve 42, 1205 Geneva, Switzerland
E-mail: mireille.betrancourt@unige.ch

Abstract. The development of argumentative writing skills is necessary and expected in secondary and higher education to foster critical thinking, but they are difficult to master. A computer-supported argumentative writing application, C-SAW, was designed to support this process. In its development and testing, scripts were written to record the use of embedded scaffolding devices and textual changes made. For research purposes, a concept-mapping and visualization software (VUE) was used to manipulate and code recorded data to visualize writing process stages for each participant. Further analyses using transitional probability matrices revealed what types of aid were consulted during different stages of writing and which aids and actions were involved in effectively regulating writing processes and strategies. In addition to being useful in analyzing the usability and effectiveness of the technology-enhanced environment, this type of analysis also gave insight into individuals' writing processes and difficulties and could be used by instructors wishing to provide feedback or adapt their instructional designs.

Keywords: writing processes, argumentation, technology-enhanced learning, process visualization

1. Introduction

As a learning methodology, argumentation and argumentative writing can be a means to analyse and evaluate, modify, and justify one's beliefs or knowledge (Kuhn, 1991). "*Embedding and fostering argumentative activities in learning environments promotes productive ways of thinking, conceptual change, and problem solving,*" (Jonassen & Kim, 2010). However, engaging in self-aware reflective processes while generating text, reasoning to validate ideas and structuring them to attain a particular rhetorical goal calls upon meta-cognitive and self-regulatory capacities that are often underdeveloped. These are skills that learners need to acquire so that they can engage in the critical thinking and meaningful reflection necessary for learning through argumentation (Felton & Herko, 2004; Kuhn, 2001).

C-SAW (Computer-supported argumentative writer) is a browser-based authoring environment designed to support the acquisition of argumentative writing skills. In the design and development of C-SAW, a design-based research methodology—with several cycles of needs analysis, construction of a theoretical framework, development and implementation, and testing and reporting (Reeves, 2000; Herrington, 2012)—was undertaken to refine the embodiment of principles derived from previous cycles (Benetos, 2012), and to test the effectiveness of C-SAW in supporting the acquisition of argumentative writing skills. The principles embedded in its design aim to support the mechanisms that enable deeper reflection and learning by supporting argumentative writing on the process and product level.

In a second development cycle, a field study was conducted to ascertain the effects of C-SAW, its usability and its potential for acceptance by learners and instructors (Davis, 1989). This field study was important in order to evaluate C-SAW from an ergonomic and usability perspective, but was also an opportunity to observe patterns of use and possible emerging utilization schemes so as to evaluate how and why C-SAW could affect argumentative writing. This implied evaluating not only the product and learning outcomes, but also the processes that led to them, i.e.: when and how specific devices and contextual help were used to solve the task problem, as well as the outcomes of these uses.

This paper looks at current methods used to capture and analyse writing processes when writing with computer-supported learning environments. In the absence of suitable methods that capture the interactions between writing processes and the computer-supported learning environment being used, we will detail the method developed to capture and visualize the use of C-SAW so as to evaluate its effectiveness in supporting written argumentation. The opportunities these types of visualizations can offer researchers, instructors and with further development and research, perhaps even learners, will also be considered.

2. Evaluating computer-supported learning systems

In systems design and ergonomic testing of computer-based environments, think-aloud protocols, recording of computer log file and user-interface events, video and screen recordings, and eye-tracking are some of the methods used to gather data on human-computer interactions to determine a system's usability and effectiveness in achieving targeted tasks. Data collected from one or more of these methods can be further synchronized automatically or manually mapped to keystroke and log file recordings and analysed to reveal use patterns and potential usability problems (Ivory & Hearst, 2001).

In writing research, with the wide-spread use of computers for writing, keystroke logging software has been used to capture writing processes in real-time (Andriessen, Erkens, & Van De Laak, 2003; Baaijen, Galbraith, & de Glopper, 2012; Braaksma, Rijlaarsdam, & van den Bergh, 2009) and sometimes in conjunction with screen recordings (Latif, 2008). Keystroke logging allows for the time-stamped recording of keyboard and mouse events to give detailed information about writing processes on various levels of text production (Baaijen et al., 2012; Leijten & Van Waes, 2013) and can even provide data for creating Lindgren-Sullivan (LS) graphs to visualize writing processes (Lindgren & Sullivan, 2013). Combining methods, researchers can collect extremely rich data for quantitative and qualitative analysis. However, studies and methods looking at patterns of interactions between the system, writing processes and outcomes are lacking or focus mostly on comparisons between conditions (with or without systems and supports) on outcomes, without analysing specific human-computer interactions and the specific processes they evoke to understand which mechanisms they may be affecting or mediating.

3. Studying writing with C-SAW

3.1. Research questions

The field study conducted sought to answer certain research questions to inform further needs and considerations in the design of C-SAW, namely: *How do students appropriate the use of C-SAW? Which functions are used,* *how and to what effect? What are their perceptions of using C-SAW?* and *Can C-SAW be used to study writing processes and products?* Eight 1st-year university students enrolled in a mandatory writing course used C-SAW during two, one and a half hour sessions to write a draft of an argumentative text on a current affairs topic of their choice researched prior to the writing sessions. To gather further information on users' intentions and perceptions, think aloud protocols were used in early design testing but were deemed too intrusive for a writing task that would span several hours within a classroom setting. Instead, retrospective accounts during interviews that could be crossed with user interface event log file recordings and text products were used to corroborate and elaborate upon findings from recorded data.

To answer research questions regarding the kinds of uses participants made of C-SAW and their relationship to the writing process and product, we needed to be able to associate particular uses of the interface devices to particular effects on writing processes and products. It was important to trace how the tool was being used; what choices and sequences of actions were made or not made and what effects these may have had, so as to be able to compare them with the documented effects they may have had on participants' processes and written products. To test conjectures on the effects of design principles and their embodiment in the devices in C-SAW's interface, it was essential to capture the interaction between participants and the computer-based learning environment. With a small number of participants (N=8) we chose to focus on descriptive and qualitative analysis of the writing process using C-SAW.

3.2 The C-SAW writing environment

A literature review and early development and prototype testing cycles, resulted in design principles that were embodied in the devices of C-SAW (Figure 1 and Figure 2). These principles include providing: I – *self-regulatory facilitators* that give dynamic visual feedback to aid self-monitoring and task completion; II *global and local argument structural aid* to help with text cohesion; III *contextual cognitive aid* through prompts to scaffold reasoning and argument construction; and IV *multiple representations* to facilitate structuring and internalization of the argumentative writing schema (Benetos, 2012).

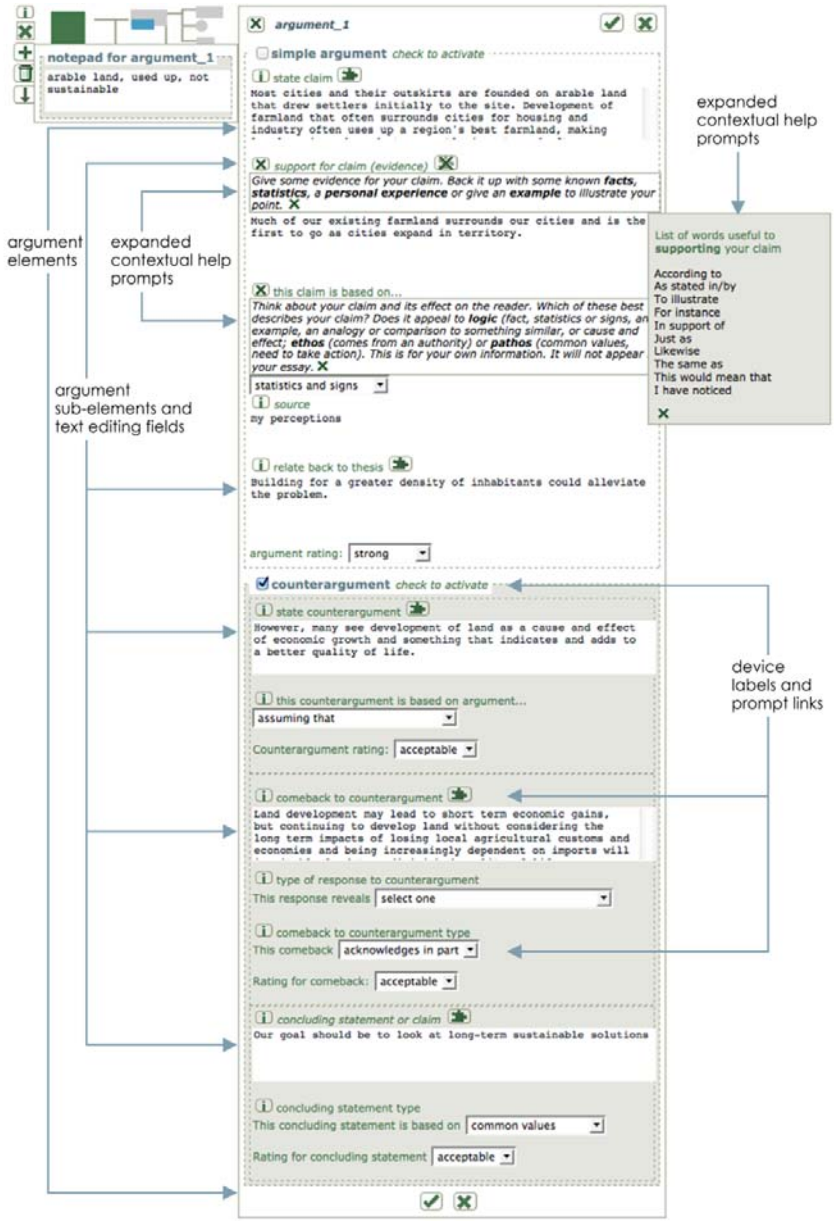


Figure 1: C-SAW interface with a view of an argument in edit mode with the simple argument content, notepad, contextual prompts and connectives list help in expanded view

These principles embodied in C-SAW's devices represented conjectures on scaffolding writing process mechanisms with particular expected outcomes forming the basis of research questions and hypotheses to be tested. 1

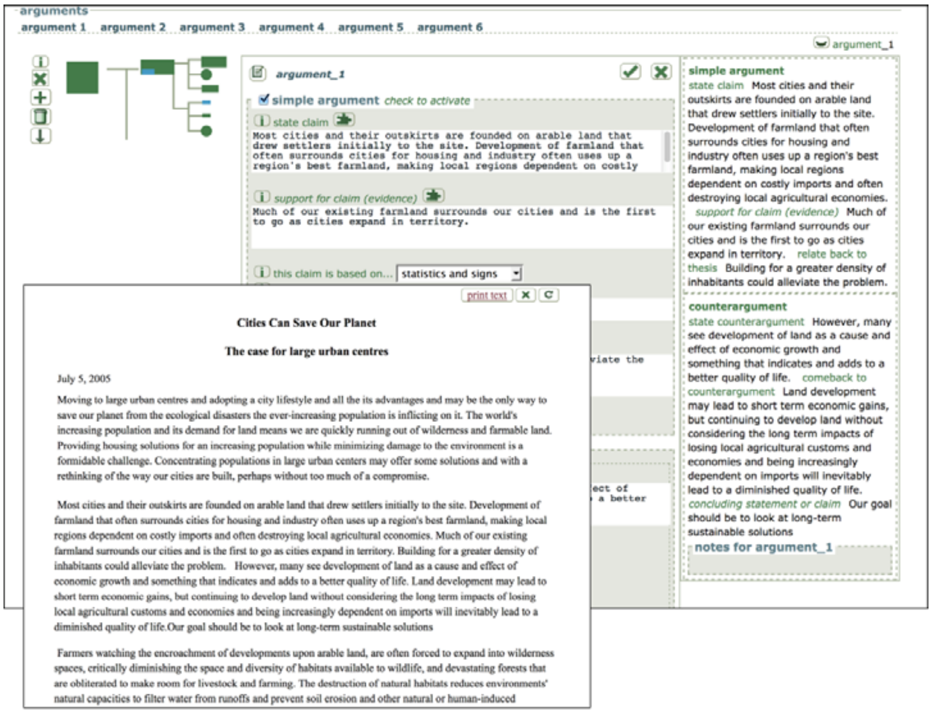


Figure 2: C-SAW interface with text view options – editing mode input fields and text with inline help (background) and full text print view (foreground). A graphic schema of each essay element showing task progression is always visible (top left)

C-SAW offers multiple representations of the structure of an argumentative essay to scaffold the writing process. Scaffolding devices come in three forms: *text input fields*, *prompts (markers and cues)*, and *reflection and self-evaluation drop down selection menus* (Figure 1). A graphic schema of each essay element showing task progression is always

[1] The details of this research and its findings, presented in a doctoral thesis (Benetos, 2015) are beyond the scope of this paper. They are currently being prepared for separate publication.

visible (Figures 1 and 2, top left). The text is edited through called up text input fields. The product can be viewed in text only, text and outline markers, or the dynamically generated graphic schema formats (Figure 2).

C-SAW's devices scaffold the writing process and the graphic schema gives dynamic feedback for comparison between the current and desired states, depending on areas completed and the writer's self-evaluation while writing. But for research purposes—to understand how and to what effect C-SAW was being used—data was collected while participants wrote, and the traces were then analysed through the use of the separate visualizations discussed here that were not available to participants.

3.3 Data collection for analysing writing processes

Keystroke logging can reveal much about pauses, text production bursts or deletions and revisions and with more advanced tools like InputLog, even mouse events in applications other than the word processor can be recorded (Leijten & Van Waes, 2013). For logging interactions within C-SAW's dynamic and adaptable interface, keystroke logging had some limitations. Testing the study design revealed that keystroke logging resulted in writing data whose granularity was too small, without providing the interaction information needed. For example, while mouse events are recorded, user-interface events such as the choices made among a list of menu items, are not. Additionally, even when coordinates of mouse clicks are recorded, if the user scrolls the page, resizes the window or alters the layout by activating certain options, these coordinates have to be remapped to the changing placement of the areas of interest. Synchronizing interactions between the system and the writing process recorded with keystroke logging becomes extremely difficult and time-consuming.

Screen recordings and eye-tracking can reveal much about how interfaces are used in real-time, but are also similarly limited in interfaces where the interface display can be resized or changes dynamically due to the users' interactions, or where scrolling displaces the area of interest. Analysis of screen recordings would require very time-consuming manual coding of videos without necessarily giving precise information on menu selections and text changes made, which, as with eye-tracking, would have to be mapped to keystroke recordings. These methods were abandoned in early testing as the data provided was too detailed and its qualitative analysis proved be arduous and prone to human error without giving the level of information necessary to answer research questions regarding the patterns of use of particular devices and subsequent responses.

3.4 Data visualization preparation

Recording user-interface events provided the information needed on users' interactions with C-SAW. However, here too, the data provided can be overwhelming, requiring further automated scripting to extract pertinent information (Hilbert & Redmiles, 2000).

Customized scripts needed to be written to capture relevant events at the level of granularity appropriate to the research questions for each user. PHP script was written to record every function activated and every selection and input made by a user within the C-SAW interface browser window. Each interface event is uniquely identified according to its essay schema location and type of event (text input, menu selection, on-click event). These are recorded and time-stamped, capturing data on what functionalities are accessed, when, and what text changes are made throughout the writing process (Figure 3). For example, activating the edit mode of a particular element such as the introduction or an argument, clicking on a link to access additional help or open one of the text views, making a menu selection, entering text into a field, etc. Every text entry saved or interface event selected is also recorded and labelled according to its position in the essay schema so that screen coordinates need not be considered.

The resulting log file renders data for each user, in tables, that can be viewed through the browser and saved as CSV (comma separated values) files. The script also produces a quick frequency count of each recorded edit or clicked function. The information recorded in these log files allowed for the collection of specific interactions targeted for analysis that would be have been arduous and difficult to discern from keystroke analysis or screen recordings.

To exploit the data collected so as to reveal uses and patterns, it was necessary to find ways to visualize and describe the process of each participant in order to compare, analyse and interpret what was happening on a global and local level and to cross this data with data collected from interviews and the written product.

Data extracted from C-SAW's customized log files were imported into Excel. A macro was used to index all actions and determine the previous value or state of each interaction. This was important to be able to determine what type of text edit was made (spelling correction, addition or removal of text) or what menu input value changed (increase in rating, change in claim type definition) as this would reveal the type of cognitive writing process in which a participant was engaged during the interaction. In Figure 3, the highlighted row shows that the participant spent almost two minutes editing the text of the first counterargument. Comparing the previous text to the current text shows that the semantic content of the text was significantly revised.

1	Process action	Event ID	Date/Time	Time elapsed	Time flow	Value	Previous value
123	rewrite	arg1-c-state-value	12/2/13 9:10:50	00:00:48	01:34:06	Other people would suggest that our president is not the source of the embarrassment but rather the artist, because of his blatant disrespect for the president and for publicly portr	Other people would suggest that our president is not the source of the embarrassment but rather the artist, because of his blatant disrespect for the president and for publicly portraying him in a negative light.
124		arg1-c-comeback-valu	12/2/13 9:11:00	00:00:10	01:34:16		0
125	remove text	arg2-s-relate-value	12/2/13 9:11:10	00:00:10	01:34:26		Therefore it should be considered vital that the artists rights are preserved as much as possible, because it could have a long term damaging effect on the country.
126	save	arg2-save	12/2/13 9:11:10	00:00:00	01:34:26		0
127	review	intro-edit	12/2/13 9:11:13	00:00:03	01:34:29		0
128	edit	arg1-edit	12/2/13 9:11:25	00:00:12	01:34:41		0
129	revise	arg1-s-relate-value	12/2/13 9:13:02	00:01:37	01:36:18	Therefore Zuma's actions are not justified, because they place our freedom of speech in jeopardy because Zuma was angry with an unflattering portrayal of himself.	Therefore Zuma's actions are not justified because he completely overreacted to the painting, as as a result, he has placed the country's basic human rights at risk.
130	revise	arg1-c-state-value	12/2/13 9:14:48	00:01:46	01:38:04	Others would suggest that Zuma was fully within his rights to protest the painting, as it not only placed himself in negative light in the eyes of his family, but also in the eyes of the rest of the country and the world.	Other people would suggest that our president is not the source of the embarrassment but rather the artist, because of his blatant disrespect for the president and for publicly portr
131	help	arg1-c-comeback-help	12/2/13 9:14:48	00:00:00	01:38:04		0
132	generate	arg1-c-comeback-valu	12/2/13 9:16:20	00:01:32	01:39:36	It is understandable that Zuma was not happy about this painting, however, the fact that he called for boycotts and that he sued the artist and the gallery suggests that he is corrupt. His actions are similar to an authoritarian ruler, and it is not fitting in a democratic society.	0
133	revise	arg1-c-concluClaim-val	12/2/13 9:17:06	00:00:46	01:40:22	In conclusion, Zuma's actions are not justified on the grounds that he is going against the country's democratic values.	In conclusion, Brett Murray's painting is justified, and Zuma's reaction is not.
134	save	arg1-save	12/2/13 9:17:10	00:00:04	01:40:26		0
135	edit	arg1-edit	12/2/13 9:17:14	00:00:04	01:40:30		0
136	validate	arg1-s-type-value	12/2/13 9:17:25	00:00:11	01:40:41	fact	statistics and signs
137	rate increase	arg1-s-rating	12/2/13 9:17:30	00:00:05	01:40:46	strong	acceptable
138	validate	arg1-c-type-value	12/2/13 9:17:35	00:00:05	01:40:51	being not logical	being untrue
139	rate increase	arg1-c-rating	12/2/13 9:17:37	00:00:02	01:40:53	strong	acceptable

Figure 3: Example of a participant's C-SAW generated log files. The names of the actions (Event ID column) are the unique identifiers used to call up functions as users interact with C-SAW: e.g. arg1-c-state-value indicates the user entered a text value in the counterargument field of argument 1 with the text in the *value* column being the value of the text when the user performed the next action. Similarly, arg1-c-rating and the value in the *value* column indicate the user evaluated the counterargument of the first argument as being 'strong'. The *previous value* column shows the state of the previous values (for comparison). The time of the interaction (*Date/time* column) is recorded. The time spent on each action (*Time flow*), and the *time elapsed* since the start of the writing activity are also calculated.

4. Visualizing writing processes using C-SAW

4.1. Mapping users' actions

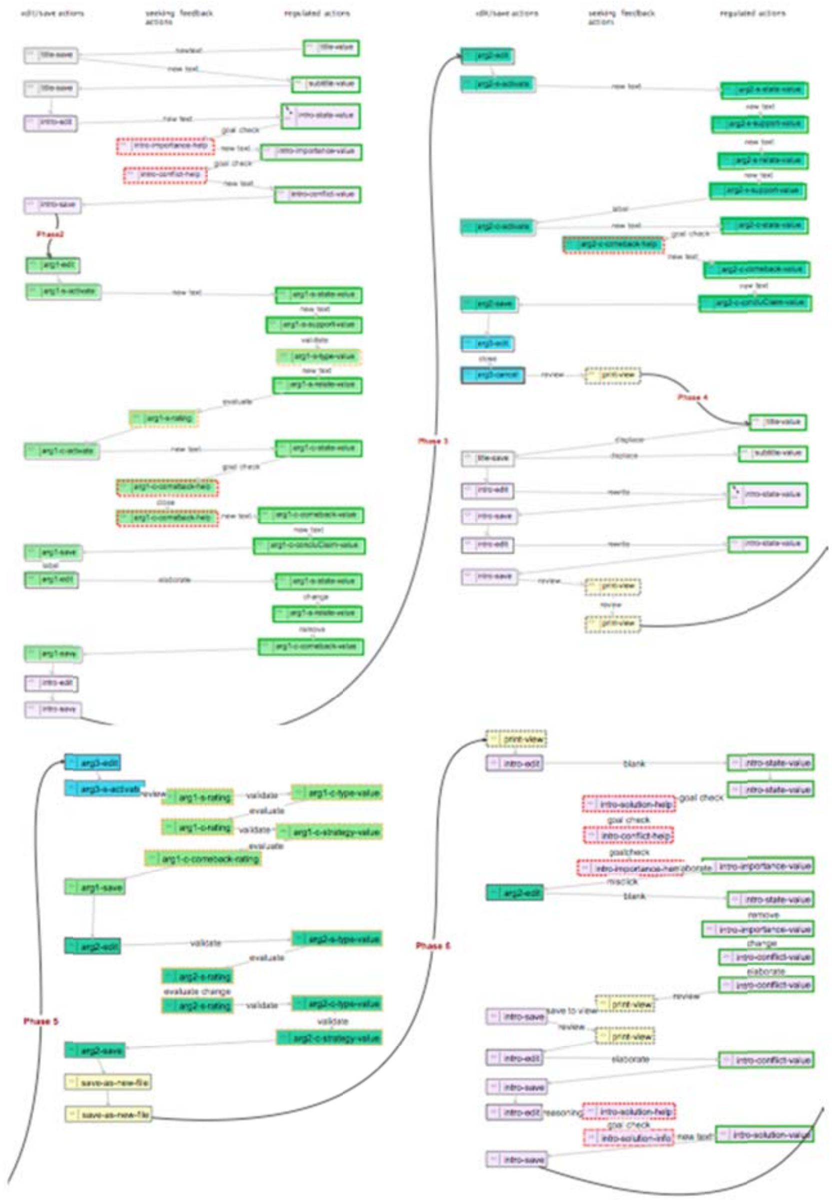
Extracted CSV files were used to generate chronological maps of users' actions using VUE[2] (Visual Understanding Environment), a concept mapping and data visualization software. Upon import, VUE generated nodes for each user-interaction event and its corresponding table row information that could then be automatically sorted chronologically (Figure 4 and inset Figure 5). Using simple string searches, specific nodes labeled according to the interaction represented by the input type unique identifiers, were colour-coded according to the element of the argumentative essay (introduction, arguments, conclusion) and arranged according to three broad categories representing interaction types. In Figures 4 and 5, nodes are organized into 3-column groupings of phases moving from left to right. The left column of each group shows *edit/save actions* to manipulate the interface, the centre column shows *feedback seeking actions* to access help and views, and the right column shows *regulated actions* to add, cut, paste text or make other menu selections, such as organizing text or self-evaluation, etc. Different colours are used for each area and each argument (pink for the introduction, green hues for arguments and pale blue for the conclusion. Continuous line borders represent user input, dotted lines borders represent the activation of prompts or views. Phase changes (lines between 3-column sections) represent the writer moving to a new cycle of editing marked by a focus on a different area or idea in the text. Links were added between each node in chronological order and labeled with the codes representing the writing process involved in moving from one action to the other. This step was essential because while the recordings of the events activated let us know where and what a participant did at a particular point, they revealed little about the process that lead to this, nor what effect it had on subsequent actions. This could only be inferred by looking at the previous and subsequent actions of each node. For example, a participant could have activated the edit mode of an argument to modify the text, to reread it, or to copy text to paste elsewhere. Or it could be that the

---

[2] Visual Understanding Environment (VUE) is an Open Source project based at Tufts University (http://vue.tufts.edu/).

participant wished to attribute a strategy type to a claim or simply rate it. The purpose and thus the process in which the participant is engaged can only be inferred by following the sequence of actions undertaken. Similarly, the nature of the text modifications made can only be defined in relation to the previously existing text. These *process actions* were the inferred purpose of the writing activity sequence. Codes were developed with reference to Breetvelt, Bergh, & Rijlaarsdam's (1994) proposition of observed cognitive activities while writing to define categories of actions related to the planning, translating and revision phases and their monitoring as presented in Flower & Hayes' (1981) cognitive model of writing. These were revised and refined iteratively in the course of the coding of the first few process maps to include system interactions (Table 1).

The process actions derived (Figures 4 and 5) were coded into 3 categories of writing processes using C-SAW: 1) *text product actions (green outlines with continuous line border):* actions that involved manipulating the text production, 2) *self-regulating actions (orange outlines with dotted line border):* actions aimed at getting guidance or feedback from the system to regulate their writing process, and 3) *interface actions (grey with continuous line border):* all actions aimed at manipulating the interface. This allowed for an easy viewing of the types of processes each participant engaged in as he or she progressed through the task.


Phase 5
arg3-edit
arg3-s-activate
review
arg1-s-rating
validate
arg1-c-type-value
evaluate
arg1-c-rating
arg1-c-strategy-value
arg1-c-comeback-rating
arg1-save
arg2-edit
arg2-s-type-value
arg2-s-rating
evaluate change
arg2-c-type-value
arg2-c-strategy-value
arg2-save
save-as-new-file
Phase 6
print-view
intro-edit
blank
intro-state-value
intro-solution-help
goal check
intro-conflict-help
intro-importance-value
misclick
remove
change
elaborate
intro-conflict-value
intro-save
save to view
reasoning
intro-solution-info
new text
intro-solution-value

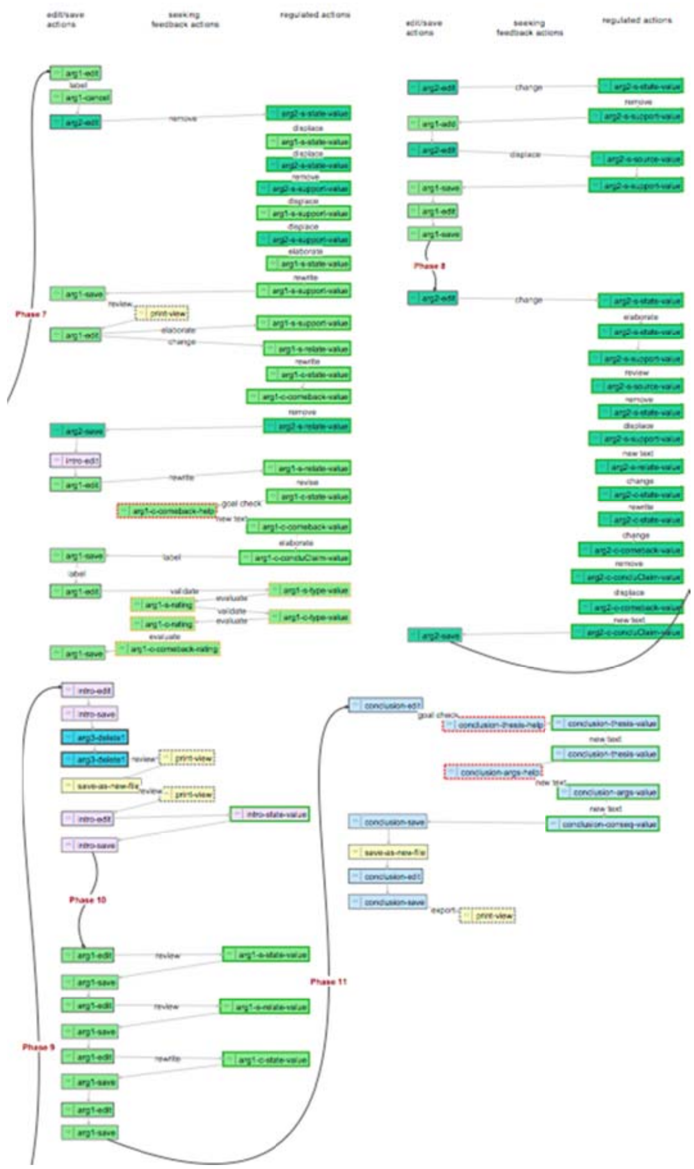


Figure 4: Example of one full coded process map presented in 4 sections (top to bottom).

A closer view of the map in Figure 5 shows a dialog box containing all the information of the selected node, including this node's previous value and the time elapsed between this and the previous action. The connecting lines are labelled according to the type of writing process in which the user was engaged between the two actions. This is the same portion of the sequence of actions shown in the CSV file visible in Figure 3.

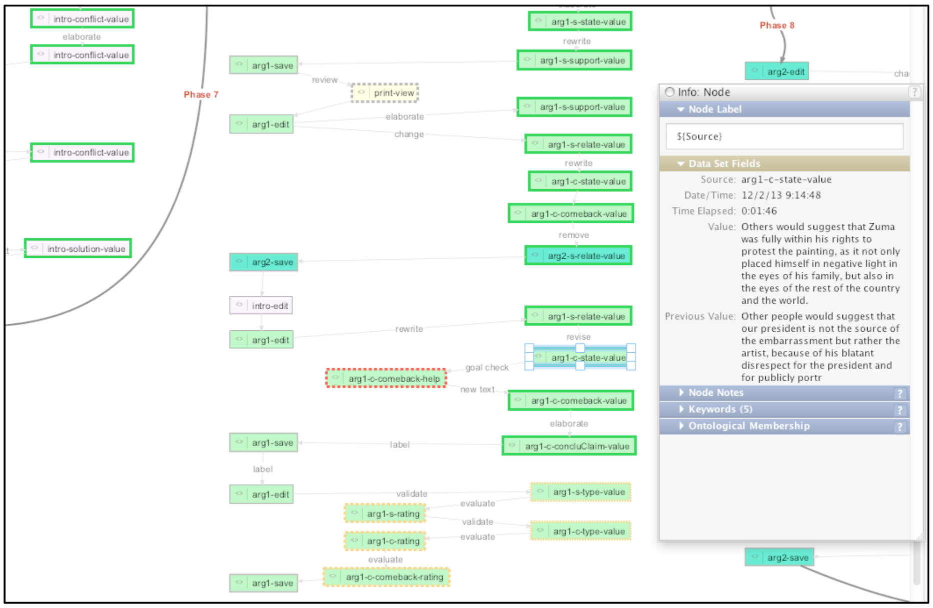


Figure 5: A portion of the same participant's process map with links manually coded in VUE. The dialogue box shows node's previous values and time stamp. Also shown in table format in Figure 3.

The process maps produced from C-SAW's log file data allowed for a visualization of the extent to which individuals wrote linearly or recursively, the level and type of engagement, and emerging patterns of use. In Figure 4, for example, we can see that the use of the contextual help for the arguments (green hue nodes) fades out by the time the participant is writing the third argument, potentially showing a fading out of the need for scaffolding. We can also see that he or she modified the introduction (pink nodes) at several points, adjusting his or her rhetorical goal as the argumentation developed. This inference was confirmed through a closer look at the text changes and during interviews.

Table 1: Process action codes in 3 categories and 5 groups and 8 subgroups

Category	Process level actions	Action (activity)	Interaction description
Text product			
Text production	grammatical	rewrite	correct or rephrase for grammatical errors
	semantic	generate	introduce new idea
		develop	continue idea introduced
		elaborate	add related idea to existing text to expand idea
		remove text	text is permanently removed
		revise	replace written text with new idea
		plan	text entered as outline or note form
Text organization: displace text from one section or argument to another			
	organize	cut text	remove text that will be pasted elsewhere
		paste text	cut text is pasted
		order	re-order arguments
Self-regulation: actions to regulate writing process			
Self-evaluation	validate	analyze	select an type to qualify a component
		source	add a source cited in justifications
	evaluate	rate	add a rating
		rate increase/ decrease	increase/decrease rating
Goal-setting			
	help-seeking	help	solicit help prompts
		connect	solicit connecting words prompts
	review	review	no text changes made to recorded field
		review local	solicit local text view
		review global	solicit global text view
Interface manipulation			
	interface	activate	de/activate argument parts
		deactivate	
		misclick	click with no related subsequent action
		export	export text produced
		save as/load	save as a new or loading a saved essay
		edit	call up edit mode of an area
		save	save changes in an area

The process actions coded in the process map links were simultaneously added to source data spreadsheets in chronological order. A coding scheme was developed to associate user interactions in C-SAW with activities involved in writing processes. *Text product* actions were split into two groups: *text production*: all activities related to text generation, and *text organization*: moving text from one area to another. Text production activities were further divided into two sub-groups: *grammatical*: edits made to correct or rephrase text without changing the meaning, and *semantic*: edits that introduced new ideas or modified the meaning of existing text.

Self-regulation actions included all interactions aimed at seeking guidance or feedback on the cognitive or metacognitive level to either evaluate one's product (*self-evaluation*) or set further goals (*goal-setting*). *Self-evaluation* refers to actions made to *validate* their arguments (selecting an argument type or adding a justification source) and to the use of the rating devices to *evaluate* the elements of one's arguments. *Goal-setting* includes *help-seeking* through accessing prompts and connectives leading to writing but also text views for *reviewing* and the selection of appropriate strategies in response to it.

4.2 Transition visualizations

The process action data gathered for each participant were then further analysed using Markov's transition probability matrix to find which process actions were more likely to precede or follow other actions. The result allowed for a case-by-case coding, visualization and comparison of individual participants' processes and the group as a whole, leading to a better understanding of similarities and differences between participants in approaches to writing using C-SAW and revealing patterns of use sequences and potential usability problems.

The resulting frequencies of transitions between the eight category process level actions (Table 1), were visualized in a table with conditional formatting to show the highest values of occurrence for each transition per participant and the group mean, so as to show which actions were most likely to precede or follow each action type and reveal information about each participant and the group as a whole (Figure 6). The series of tables below can be read by row to give the probability of actions following a particular action type or by column to give the probability of actions preceding a particular action.

For example, *grammatical* changes (spelling and grammar) were most likely to precede *semantic* text changes (text changes that changed the meaning or content of the text) with a 51% probability, as would be expected in a text production phase, moving from local revision to new text production. This is the case with most participants (P2, P5, P6, P8). Participant 1, however, seems to have mostly viewed his text in one of the text previews after making grammatical changes (75% probability) indicating a need to view the entire text. Grammatical text changes were also most likely to be preceded by grammatical changes, showing participants engaged in grammar editing phases. These high-occurrence transitions are not really transitions. They represent, rather, process action *phases* and appear in other actions as well, e.g. *semantic*, *help-seeking*, *review*, etc.

As a further example, *semantic* text changes, beyond leading to more semantic text changes indicating a text production phase, were most likely to precede *review* actions (viewing text produced within the area next to text input fields or the entire text in a separate pop-up window) or *help-seeking* (accessing prompts). Help-seeking would be expected when writers move from one text element field to the next, looking for guidance as to what is expected of them so as to set appropriate goals before they write. In the case of participants with higher proportions of *semantic* to *help-seeking*, this could be a sign of being ‘stuck’ in some way in their writing, either because they do not understand what is expected or they cannot generate adequate ideas to fulfil the writing goal at this point.

The fact that the transition probabilities of certain actions can most readily be explained by the order in which they are presented in the interface, should not be used solely to discount the high occurrence of these transitions. The order of the prompts and devices was intentional and conceived so as to interfere as little as possible with the writing process while offering writers the option to engage in immediate reflection on their preceding or subsequent propositions. That expected transitions were in fact happening is interpreted as a good sign. When evaluating the system’s usability, it is the unexpected transitions that are informative as they could be indicators of design problems. Unexpected uses could also reveal information of emerging instrumentation and instrumentalization (Rabardel, 1995).

1	grammatical -> grammatical	grammatical -> semantic	grammatical -> organize	grammatical -> validate	grammatical -> evaluate	grammatical -> help-seeking	grammatical -> review
tester01	0.05	0.15	0.00	0.00	0.00	0.05	0.75
tester02	0.13	0.50	0.13	0.00	0.00	0.25	0.00
tester03	0.50	0.36	0.00	0.00	0.00	0.07	0.07
tester05	0.00	1.00	0.00	0.00	0.00	0.00	0.00
tester06	0.00	1.00	0.00	0.00	0.00	0.00	0.00
tester08	0.00	1.00	0.00	0.00	0.00	0.00	0.00
tester09	0.00	0.00	0.00	0.00	0.00	0.00	0.00
tester10	0.00	0.00	0.20	0.60	0.00	0.20	0.00
average	**8.4%**	**50.1%**	**4.1%**	**7.5%**	**0.0%**	**7.1%**	**10.3%**

2	semantic -> grammatical	semantic -> semantic	semantic -> organize	semantic -> validate	semantic -> evaluate	semantic -> help-seeking	semantic -> review
tester01	0.04	0.20	0.02	0.09	0.09	0.27	0.29
tester02	0.08	0.50	0.04	0.04	0.02	0.14	0.18
tester03	0.08	0.56	0.02	0.06	0.00	0.15	0.13
tester05	0.03	0.35	0.00	0.27	0.08	0.16	0.11
tester06	0.03	0.45	0.00	0.15	0.08	0.15	0.15
tester08	0.00	0.18	0.18	0.09	0.09	0.00	0.45
tester09	0.00	0.89	0.00	0.00	0.00	0.05	0.05
tester10	0.07	0.32	0.07	0.11	0.07	0.29	0.07
average	**4.1%**	**43.3%**	**4.2%**	**10.1%**	**5.3%**	**15.0%**	**17.9%**

3	organize -> grammatical	organize -> semantic	organize -> organize	organize -> validate	organize -> evaluate	organize -> help-seeking	organize -> review
tester01	0.00	0.17	0.33	0.33	0.00	0.00	0.17
tester02	0.00	0.18	0.73	0.00	0.00	0.00	0.09
tester03	0.00	0.50	0.50	0.00	0.00	0.00	0.00
tester05	0.00	0.00	0.00	0.00	0.00	0.00	0.00
tester06	0.00	0.00	0.67	0.00	0.00	0.00	0.33
tester08	0.00	0.10	0.70	0.10	0.00	0.00	0.10
tester09	0.00	0.00	0.00	0.00	0.00	0.00	0.00
tester10	0.00	0.29	0.43	0.14	0.00	0.14	0.00
average	**0.0%**	**15.4%**	**41.9%**	**7.2%**	**0.0%**	**1.8%**	**8.6%**

4	validate -> grammatical	validate -> semantic	validate -> organize	validate -> validate	validate -> evaluate	validate -> help-seeking	validate -> review
tester01	0.00	0.20	0.00	0.10	0.30	0.40	0.00
tester02	0.00	0.13	0.00	0.13	0.63	0.00	0.13
tester03	0.25	0.00	0.00	0.00	0.00	0.25	0.50
tester05	0.00	0.40	0.00	0.07	0.40	0.00	0.13
tester06	0.00	0.20	0.00	0.10	0.40	0.00	0.30
tester08	0.00	0.00	0.00	0.33	0.67	0.00	0.00
tester09	0.00	0.00	0.00	0.00	0.00	0.00	0.00
tester10	0.07	0.00	0.00	0.27	0.33	0.20	0.13
average	**4.0%**	**11.6%**	**0.0%**	**12.4%**	**34.1%**	**10.6%**	**14.9%**

5	evaluate -> grammatical	evaluate -> semantic	evaluate -> organize	evaluate -> validate	evaluate -> evaluate	evaluate -> help-seeking	evaluate -> review
tester01	0.00	0.23	0.00	0.08	0.31	0.15	0.23
tester02	0.00	0.22	0.00	0.56	0.22	0.00	0.00
tester03	0.00	0.00	0.00	0.00	0.00	0.00	0.00
tester05	0.00	0.45	0.00	0.18	0.18	0.09	0.09
tester06	0.00	0.71	0.00	0.00	0.00	0.14	0.14
tester08	0.00	0.40	0.00	0.00	0.40	0.00	0.20
tester09	0.00	0.00	0.00	0.00	0.00	0.00	0.00
tester10	0.14	0.29	0.00	0.00	0.00	0.29	0.29
average	**1.8%**	**28.8%**	**0.0%**	**10.2%**	**13.9%**	**8.4%**	**11.9%**

6	help-seeking -> grammatical	help-seeking -> semantic	help-seeking -> organize	help-seeking -> validate	help-seeking -> evaluate	help-seeking -> help-seeking	help-seeking -> review
tester01	0.00	0.63	0.00	0.07	0.04	0.19	0.07
tester02	0.00	0.69	0.00	0.00	0.00	0.31	0.00
tester03	0.04	0.26	0.00	0.04	0.00	0.48	0.17
tester05	0.08	0.46	0.00	0.08	0.00	0.23	0.15
tester06	0.00	0.55	0.00	0.18	0.00	0.27	0.00
tester08	0.00	1.00	0.00	0.00	0.00	0.00	0.00
tester09	0.00	0.33	0.00	0.00	0.00	0.67	0.00
tester10	0.05	0.38	0.05	0.14	0.00	0.14	0.24
average	**2.1%**	**53.8%**	**0.6%**	**6.5%**	**0.5%**	**28.6%**	**8.0%**

7	review -> grammatical	review -> semantic	review -> organize	review -> validate	review -> evaluate	review -> help-seeking	review -> review
tester01	0.31	0.18	0.05	0.00	0.02	0.04	0.40
tester02	0.17	0.33	0.00	0.00	0.06	0.00	0.44
tester03	0.05	0.47	0.00	0.00	0.00	0.11	0.37
tester05	0.06	0.17	0.00	0.06	0.00	0.17	0.56
tester06	0.00	0.35	0.04	0.04	0.00	0.00	0.57
tester08	0.11	0.44	0.11	0.00	0.00	0.00	0.33
tester09	0.00	0.00	0.00	0.00	0.00	1.00	0.00
tester10	0.00	0.24	0.00	0.04	0.00	0.16	0.56
average	**8.7%**	**27.3%**	**2.6%**	**1.7%**	**0.9%**	**18.4%**	**40.3%**

Figure 6: Transitional probability matrix for eight process actions from coded process maps created from log files of participants. For example the transition probability matrix of process actions following help-seeking actions for all participants shows that participants were most likely to engage in semantic text modifications following a help-seeking action

For example, P1 and P8 made high frequency transitions between semantic changes and reviewing (viewing text product in one of the views). From the interview, it was revealed that this was due to ergonomic problems of a different nature. One found the insertion of devices between fields did not allow him to review his text as a whole while writing; the other found the font in the fields too small. Similarly, intense organization phases, revealed that participants were copying and pasting text to reorganize their arguments rather than using the ‘move’ option which log files showed was not used at all.

5. Visualization results

By crossing and comparing the visualizations of user-interaction events indicating participants’ uses of C-SAW’s self-regulatory devices, the retrospective interviews and the evaluation of text quality scores, particular patterns of uses that appeared to have been beneficial to argumentative text writing for participants in this study emerged. Participants who used C-SAW’s self-regulatory devices with moderation also appeared to have written higher quality texts (n=2), while low use appeared to be associated with low text quality (n=2) and high self-regulatory uses with medium or low quality texts (n=2), indicating potential problems in use or understanding (Benetos, 2015). These gave crucial information that could then be looked at more closely through the qualitative analysis of retrospective interviews, to understand what difficulties participants encountered and what further scaffolding could be offered through C-SAW or through the instructional design. Here it was found that low device use was also associated with a low perceived usefulness (n=2) due to not understanding the terminology used, not having seen or understood the use of certain devices, or feeling constrained by the schema.

Analyses using a probability transition matrix of the writing processes of participants (Figure 6) showed that participants mostly used C-SAW as intended and implied by the layout of the interface. However, there were no associations revealed between frequencies of specific transitions and particular outcomes, in either text quality or perceived usefulness. They wrote both linearly and recursively, working within one area at a time and working linearly through the devices associated with a sub-element. High transition frequencies within categories revealed participants engaged in clear intense phases of activities: text generating (43.3%), reviewing (40.3%), grammatical correction and text organization (41.9%) phases. Grammatical changes, preceded mostly by revision actions (8.7%), most often lead to semantic changes (50.1%). However, most transition from revision lead to semantic changes (27.3%) or help-seeking (18.4%). The highest frequency transitions were semantic changes were preceded by help-seeking actions (53.8%), showing participants were using the built-in help to guide their writing, and then subsequently reviewing their text (17.9%) or either seeking more help (15.0%) to monitor their progress when not engaging in further semantic text changes (43.3%). These combined transitions showed participants were using the C-SAW embedded scaffolding to regulate their writing process, showing design goals were being achieved.

6. Conclusion

First, recording specific user-interface events allowed for customized data collection that was more revealing than keystroke logging or screen recordings without the overload of information inherent in these. Additionally, C-SAW log files are easily accessible and readable through the web browser and can be consulted by instructors wishing to see the progression of the text production without requiring specific training.

The two-stage technique—process mapping and transitional probability matrices—for visualizing writing processes and interactions with C-SAW proved very beneficial in allowing an analysis at the level of granularity appropriate to answering targeted research questions. The process maps gave an overall, yet detailed view of how participants progressed through the writing task, their level of engagement with the environment and task, and potential difficulties encountered with both. The transition matrices complemented this information by allowing a quantitative comparison of targeted interactions and process sequences within and between participants. When crossed with the product and interviews, a detailed evaluation of the uses and potential effects of C-SAW on processes and outcomes were revealed.

However, further refinements are required to make them transferable to other writing research within technology-enhanced learning systems. In order to target particular events, scripts need to be developed specifically for each software or computer-supported environment. The visual encoding outlined here is currently somewhat arbitrary and should be improved to adhere more to information visualization design principles (Munzner, 2008) that would make reading of the process maps more intuitive and accessible. The technique, while being less labour intensive than manual coding of keystroke log files, eye tracking and video capture recordings for the level of data needed, and allowing for more easily readable visualizations than LS graphs (Lindgren-Sullivan graphs), is still far from being automated and is susceptible to similar human error. The two visualization techniques described can however be easily implemented with little technical training, making them accessible not only to researchers wishing to explore and analyse uses and effects of instructional technology on writing, but also to instructors wishing to analyse and assess learners' processes to provide appropriate feedback, making them valuable learning process analytics tools. From the results, future research and development should look at how to render the visualizations more accessible and meaningful to instructors and eventually examine if they can be used to guide writers. This would require looking more closely at which traces should be made available, when and in what form so as not to overburden writers. To do this more work is needed to clarify the relationship between processes, needs for guidance, and available and exploitable traces. This work presented here is intended as a first step.

References

Andriessen, J., Erkens, G., & Van De Laak, C. (2003). Argumentation as negotiation in electronic collaborative writing. *Collaborative Learning*, *1*, 1–25.

Baaijen, V. M., Galbraith, D., & de Glopper, K. (2012). Keystroke analysis: reflections on procedures and measures. *Written Communication*, *29*(3), 246–277.

Benetos, K. (2012). Design principles for supporting self-regulation during argumentative writing. In *EARLI 2012 - SIG 6 & 7 "Instructional Design" and "Learning and Instruction with Computers."* Bari.

Benetos, K. (2015). *C-SAW : Computer-supported scaffolding of argumentative writing* (Unpublished doctoral dissertation). University of Geneva. Geneva.

Braaksma, M., Rijlaarsdam, G., & van den Bergh, H. (2009). Hypertext writing: effects on writing processes and writing products. In *Proceedings of the 13th Biennial European*

Conference for Research on Learning and Instruction.

Breetvelt, I., Bergh, H. Van Den, & Rijlaarsdam, G. (1994). Relations between writing processes and text quality: When and how? *Cognition and Instruction*, *12*(2), 103–123.

Davis, F. D. (1989). Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Quarterly*, *13*(3), 319–340.

Felton, M. K., & Herko, S. (2004). From dialogue to two-sided argument; Scaffolding adolescents' persuasive writing. *Journal of Adolescent & Adult Literacy*, 672–683.

Flower, L., & Hayes, J. (1981). A cognitive process theory of writing. *College Composition and Communication*, *32*(4), 365–387.

Herrington, J. (2012). Design-based research: Implementation issues in emerging scholar research. In *EdMedia: World Conference on Educational Media and Technology* (Vol. 1, pp. 1–6). Chesapeake, VA: Association for the Advancement of Computing in Education (AACE).

Hilbert, D. M., & Redmiles, D. F. (2000). Extracting usability information from user interface events. *ACM Computing Surveys*, *32*(4), 384–421.

Ivory, M. Y., & Hearst, M. A. (2001). The state of the art in automating usability evaluation of user interfaces. *ACM Computing Surveys*, *33*(4), 470–516.

Jonassen, D. H., & Kim, B. (2010). Arguing to learn and learning to argue: Design justifications and guidelines. *Educational Technology Research and Development*, *58*(4), 439–457.

Kuhn, D. (1991). *The Skills of Argument*. New York: Cambridge University Press.

Kuhn, D. (2001). How do people know? *Psychological Science*, *12*(1), 1–8.

Latif, M. M. A. (2008). A state-of-the-art review of the real-time computer-aided study of the writing process. *International Journal of English Studies*, *8*(1), 29–50.

Leijten, M., & Van Waes, L. (2013). Keystroke logging in writing research: Using Inputlog to analyze and visualize writing processes. *Written Communication*, *30*(3), 358–392.

Lindgren, E., & Sullivan, K. P. H. (2013). The LS Graph: A methodology for visualizing writing revision. *Language Learning*, *52*(3), 565–595.

Munzner, T. (2008). Process and pitfalls in writing information visualization research papers. In A. Kerren et al. (Eds.), *Information visualization* (pp. 134–153). Springer, LNCS 4950.

Rabardel, P. (1995). *Les hommes et les technologies: approche cognitive des instruments contemporains*. Armand Colin.

Reeves, T. C. (2000). Enhancing the worth of instructional technology research through "design experiments" and other development research strategies. In *International perspectives on instructional technology research for the 21st century, New Orleans, LA, USA*. New Orleans, LA, USA.