Wasting "Waste" is a Waste: Gleaning Deleted Text Fragments for Use in Future Knowledge Creation

Hiroaki Ikuta
School of Knowledge Science
Japan Advanced Institute of Science and Technology
Ishikawa, Japan
email: ikuta@jaist.ac.jp

Kazushi Nishimoto
Graduate School of Advanced Science and Technology
Japan Advanced Institute of Science and Technology
Ishikawa, Japan
email: knishi@jaist.ac.jp

***Abstract*— While creating a document, an author externalizes his/her thought or idea to text fragments and then organizes and revises them iteratively to polish up the document. In this process, the author may partially delete the contents (i.e., text fragments) from the document because he/she determines that the fragments are not proper as parts of the document. However, these deleted text fragments may be valuable in other future attempts at knowledge creation. This paper proposes Text ComposTer, which is a document-writing support system equipped with a function of collecting the deleted text fragments. We first conduct a pilot study to assess what kind of deleted text fragments are reusable and implement Text ComposTer based on the assessment results. We conduct user studies and confirm that Text ComposTer can efficiently collect reusable deleted text fragments.**

Keywords-intangible waste; knowledge utilization; writing support system; deleted text fragments.

I. INTRODUCTION

Is it true that objects determined as waste have no value? In conventional waste management, tangible waste is found to have value to utilize for other purposes. However, intangible waste such as ignored ideas, facts, and knowledge has never been (re)utilized so far. We believe that this intangible waste is also worth utilizing for other purposes. Behind this belief, there are our experiences: a text fragment written in a draft of a document but eventually deleted from the document is sometimes worth utilizing for other purposes. For instance, the second author of this paper found that piano players often press the keys too forcefully when there is insignificant delay between key pressing and sound emission. He described this finding in a draft version of [1]. However, he determined that this finding was inconsistent with the entire context of [1]. As a result, he deleted descriptions about this finding and completed the paper as [1]. In later years, he read the draft version of [1] again when he carried out research about supporting drumstick control, and he found a way to apply the finding to this research. Finally, he completed the research and published [2]. Thus, the second author found value in utilizing the unused finding described in the deleted text fragment for other purposes. There may be a lot of other possibilities like the above example.

There are several models of the document-writing process [3][4][5]. Each proposed model suggests that writing involves a number of distinct activities that interact in complex, interconnected ways [5]. All of these models also contain an activity corresponding to editing of the contents of a document. In addition, an author of documents usually cannot write the completed version perfectly from the beginning; he/she gradually progresses to the completed version. In this process, a lot of knowledge (i.e., text fragments, parts of the document) is merely discarded as intangible waste in the writing process although it might be valuable.

Thus, intangible waste in the writing process is worth saving. Nevertheless, there are few studies on utilizing it. To the best of our knowledge, there have been no attempts to collect deleted text fragments (DTFs) to utilize them for other document writing tasks. Some existing document writing applications are equipped with functions to keep DTFs, such as "Track Changes" function in Microsoft Word and "Snapshot" function in Scrivener. However, these applications keep DTFs to reuse them in the same document for version management; not for utilizing them in the different document composition. Microsoft Word is also equipped with a function to collect text fragments as "building blocks" to reuse them in the different document composition. However, this function requires the users to intentionally save the text fragments to reuse them afterward; they are not DTFs. Therefore, there are no applications that collect DTFs to use them in future knowledge creation.

In this paper, we propose a novel document-writing support system named Text ComposTer. Text ComposTer provides functions to compose a document (i.e., Text ComposTer is a "composer") and, in addition, functions to collect DTFs to utilize them for other purposes afterward (i.e., Text ComposTer is a "composter" of text fragments as intangible waste).

The rest of this paper is organized as follows. Section II presents an overview of several related works. Section III describes a pilot study for assessing what kinds of DTFs are worth utilizing for other purposes. Section IV illustrates Text ComposTer. We show two different user studies in Section V and discuss the usefulness of Text ComposTer based on user studies in Section VI. Section VII concludes this paper.

II. RELATED WORKS

Utilizing knowledge via technologies has been the focus of attention for a long time. One of the main attempts at utilizing knowledge is an expert system in the artificial intelligence area. An expert system stores expert knowledge in a knowledge base, and users call upon the system for specific advice as needed. Liao [6] showed various kinds of

 ISBN: 978-1-61208-538-8

expert systems. Many kinds of knowledge utilization patterns have been proposed. In addition, Mizoguchi et al. [7] pointed out problems of the knowledge base for knowledge utilization (sharing and reuse in this context), and they proposed a methodology using ontology for enhancing knowledge utilization. These studies attempted to properly formalize useful knowledge in order to utilize it.

Some interactive systems have been proposed to utilize knowledge [8][9][10]. Shibata and Hori [8] proposed a system to support long-term creative thinking in daily life. They developed a system that stored personal awareness or interests in daily life and utilized them for idea generation. Sharmin et al. [9] proposed a system to support reusing presentation slides for making slightly different materials such as a more detailed version of a base material by changing audiences. Simbelis et al. [10] proposed a system named Delete by Haiku to utilize existing text messages in a mobile phone. Delete by Haiku transforms a set of text messages that the user has selected into a *haiku*, a traditional form of Japanese poetry featuring a simple constructive form with a limited number of syllables [10].

These studies [6][7][8][9] aimed to utilize knowledge that has been determined to have potential for (re)utilizing. Simbelis et al. [10] aimed to utilize knowledge whose usefulness has not yet been determined. In contrast, we aim to utilize knowledge that has once been determined to be unuseful.

Therefore, our main research contribution in this paper is to explore a new paradigm of knowledge utilization by finding value in deleted creations in the creative process. Particularly, we start to investigate ways to utilize DTFs generated in the document-writing process.

III. ASSESMENT OF DELETED TEXT FRAGMENTS

We conducted a pilot study to assess what kinds of DTFs are worth utilizing for other purposes.

A. Experimental System

We developed a special text editor for the pilot study that is equipped with a function to collect DTFs as well as functions of a usual text editor, such as copy, cut and paste, and find and replace. This editor automatically collects and stores DTFs when it detects the following three types of user manipulations:

- Hitting elimination keys (e.g., “Delete key” and “Backspace key”),
- Inputting some characters while a string is selected, and
- Executing the replace function.

When the user hits the elimination keys, the text editor collects the eliminated string as a deleted text fragment. Similarly, when the user inputs characters while a string is selected, the text editor collects the selected string as a deleted text fragment. Also, the text editor collects the replaced string as a deleted text fragment when the user executes the replace function.

TABLE I. RESULTS OF THE PILOT STUDY

	Subject 1	Subject 2	Subject 3	Subject 4
# of characters	4726	2468	535	418
# of sentences	72	43	12	10
# of DTFs	551	124	16	99

B. Experimental Setting

We asked four Japanese people (including the first author) to perform a document-writing task using the special text editor. Each subject wrote a part of a conference paper in Japanese as the writing task.

C. Results

The results of the writing task are shown in Table I. We analyzed the results and found the following three possible factors of storing DTFs in accordance with user actions.

- Factor 1: Correcting mistypes.
- Factor 2: Revising expression.
- Factor 3: Eliminating sentences that are inconsistent with the context.

Factor 1 and Factor 2 relate to mere editorial matters. In contrast, Factor 3 relates to the content of the document. Therefore, only the DTFs obtained by Factor 3 would be useful for other purposes.

We found two issues in collecting DTFs using this editor. One of them is that collected DTFs are very messy. The reason for this is that the editor did not distinguish the deleted DTFs based on the three factors. The other issue is that the number of DTFs generated by Factor 3 was quite small. One reason for this issue is that the text fragments that are inconsistent with the context but that are suitable to utilize for other purposes are not created very often. Another reason is that the editor was not suitable for the upstream process of document composition where trial-and-error frequently occurs. Traditional text editors (including our special editor) are suitable for making clean copy rather than for the upstream process. In other words, traditional text editors cannot save the author’s thoughts that are yielded in the middle of the document-writing process but that are not finally adopted in the completed version of the document.

IV. TEXT COMPOSTER

A. Approach to resolving the issues

To resolve the above two issues, we developed a novel text composition support system named “Text ComposTer,” which supports all activities in document writing from the upstream process to the downstream one (e.g., generating, organizing, composing and revising [5]). Inspired by the Art #001 system [11], which supports the entire document writing process, we designed a user interface for Text ComposTer to make it possible to separately collect two kinds of DTFs (namely, factors 1 and 2 related fragments and factor 3 related fragments) in accordance with its usage.

 ISBN: 978-1-61208-538-8

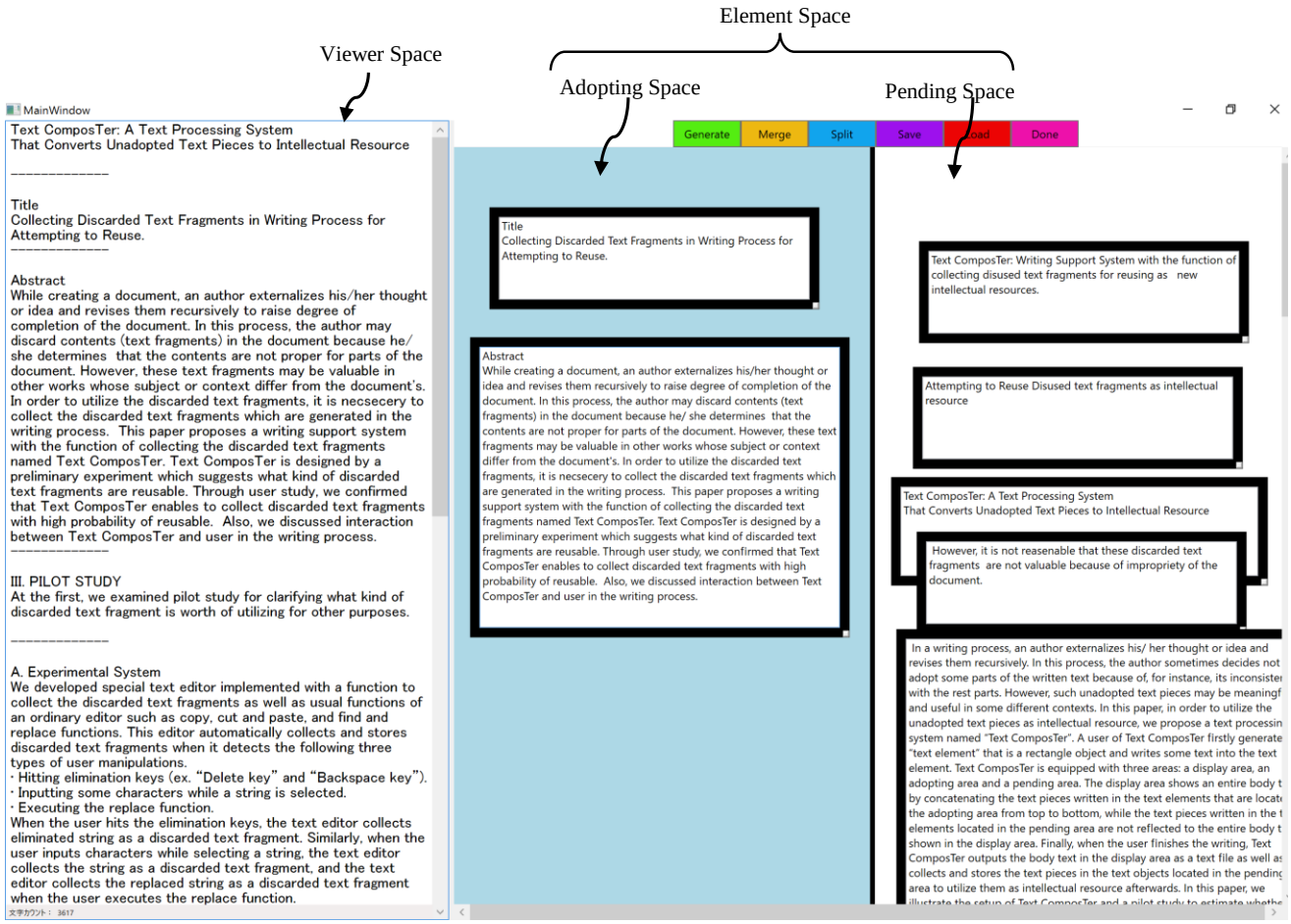

Figure 1. User interface of Text ComposTer.

B. System Overview

Text ComposTer is a native application on Windows OS™ and implemented in C# programming language. Figure 1 shows the user interface of Text ComposTer. It is roughly divided into two spaces: the viewer space and the element space. The element space is also divided into two spaces: one is the adopting space and the other is the pending space. Using Text ComposTer, a user composes a document in any language by writing sentences in a rectangle object named "element" in the element space and sequentially arranging the elements along with the flow of the narrative of the document. The viewer space shows the entire document by concatenating the sentences written in the elements that are located in the adopting space from top to bottom, while the sentences written in the elements located in the pending space are not reflected in the entire document shown in the viewer space. The user interface is equipped with six buttons (i.e., Generate, Merge, Split, Save, Load, and Done buttons) on the upper side of the element space. Each button is a trigger of following functions.

- Generate Function: The user can generate a new element in the element space by pushing the generate button.
- Merge Function: The user can merge multiple elements located in the adopting space. Text fragments written in the selected elements are merged from top to bottom and stored in an element.
- Split Function: The user can split an existing element into two elements by pushing the split button. The sentences in the original element are divided into the two new elements.
- Save Function: The user can save the current work environment of Text ComposTer as an XML-format file by pushing the save button.
- Load Function: The user can load the work environment saved previously by pushing the load button, and then select one of the saved XML files.
- Done Function: The user can output the entire document shown in the viewer space in a text-format file, as well as save the entire document by pushing the done button.

C. Collecting DTFs

Text ComposTer collects DTFs in two different grain sizes: rough-grain DTFs (R-DTFs) and fine-grain DTFs (F-DTFs). When the user pushes the done button, the text fragment in each element located in the pending space is

regarded as an R-DTF, and each R-DTF is saved in XML format. An F-DTF is a text fragment deleted by one series of delete operations in an element. Namely, the F-DTF is the same as the deleted text fragment collected in Section III.

A user of Text ComposTer generates an element, then writes text in it and moves it in the element space iteratively. Finally, when the user finishes writing, Text ComposTer outputs a series of text fragments in the viewer space as a text file and also collects and stores the text fragments in the elements located in the pending space to utilize them as intellectual resources afterwards.

V. USER STUDIES

We conducted two types of user studies to investigate characteristics of Text ComposTer.

A. User Study 1: Comparing R-DTF and F-DTF

We asked four Japanese subjects to perform a writing task using Text ComposTer. Each subject was a master's student and wrote a research proposal in Japanese. In this user study, each subject used Text ComposTer to complete the first or second draft. We instructed each subject in the functions of Text ComposTer. We especially emphasized that, once generated, elements cannot be removed; to delete a text fragment in an element from the final document, it should be moved to the pending space. After the writing task, we asked each subject to evaluate the collected R-DTFs and F-DTFs. Particularly, we asked them to evaluate whether each R-DTF or F-DTF was "useful" or "useless" or "neither" regardless of the context of the research proposal.

Table II presents the number of collected DTFs, average character numbers of the DTFs, and standard deviation of the character numbers for grain sizes and for subjects. In user study 1, it was found that all of the R-DTFs generated by each subject were fewer, longer and had larger deviation of character numbers than correspondent values of F-DTFs.

Table III presents statistics (comprising number and ratio) of evaluation of usefulness for grain sizes and for subjects. The ratio of evaluation as "Useful" to all evaluations in R-DTFs is much greater than the corresponding ratio in F-DTFs. In addition, any R-DTFs and F-DTFs evaluated as "Useful" tend to be long strings and include technical terms.

TABLE II. STATISTICS OF DTFS IN USER STUDY 1

		Number	Avg. length	SD
R-DTF	Sub. 1	5	240.6667	233.2600
	Sub. 2	8	86.6250	51.8361
	Sub. 3	0	–	–
	Sub. 4	6	97.6667	69.8538
	Total	19	106.4737	84.7984
F-DTF	Sub. 1	196	5.3163	14.4821
	Sub. 2	309	3.9029	5.7124
	Sub. 3	38	10.1842	14.4495
	Sub. 4	84	6.5176	28.7859
	Total	627	4.6571	10.0065

TABLE III. RESULTS OF EVALUATION OF DTFS IN USER STUDY 1

		Useful	Useless	Neither
R-DTF	Sub. 1	3 (50%)	3 (50%)	0 (0%)
	Sub. 2	5 (62.5%)	3 (37.5%)	0 (0%)
	Sub. 3	0 (-%)	0 (-%)	0 (-%)
	Sub. 4	3 (50%)	0 (0%)	3 (50%)
	Total	11 (55%)	6 (30%)	3 (15%)
F-DTF	Sub. 1	1 (0.5%)	195 (99.5%)	0 (0%)
	Sub. 2	5 (1.6%)	303 (98.1%)	1 (0.3%)
	Sub. 3	5 (13.2%)	12 (31.6%)	21 (55.3%)
	Sub. 4	0 (0%)	84 (100%)	0 (0%)
	Total	11 (1.8%)	594 (94.7%)	22 (3.5%)

B. User Study 2: Comparing Text ComposTer and the Special Editor in Section III

We asked eight Japanese master's students to perform two writing tasks using Text ComposTer and the special editor presented in Section III: Text ComposTer for one of the tasks and the special editor for the other task. The topics of the writing tasks are as follows.

T_1: Please predict form and function of mobile phones 10 years in the future.

T_2: Please devise a way to make JAIST (Japan Advanced Institute of Science and Technology, to which the subjects belong) widely known to the public. At a minimum, describe a specific way and assess the merits and demerits of its execution.

Each writing task was limited to 30 minutes, and the number of characters of each document was restricted in the range of 100 to 400. We allowed each subject to take a break for about five minutes in each writing task. For counterbalancing, each subject was allocated to different combinations of systems used (Text ComposTer or the special editor), topics (T_1 or T_2) and order (first writing task, second task). After the writing tasks, we carried out semi-structured interviews in which we mainly asked about usability of each system and how to use each system in performing the writing task.

Table IV presents the average number of collected DTFs, average number of characters of the collected DTFs, and standard deviation of the character number for systems used and for topics. As shown in Table IV, the average number of characters of DTF collected by the special editor and F-DTF are almost the same, but their standard deviations are different.

We found that usage of Text ComposTer and the special editor differed depending on the subjects' writing styles.

TABLE IV. STATISTICS OF DTFS IN USER STUDY 2

		T_1			T_2		
		Avg. number	Avg. length	SD	Avg. number	Avg. length	SD
Text ComposTer	R-DTF	5.75	40.2727	50.1789	2.25	63.3333	47.8774
	F-DTF	74.75	4.7715	9.2677	42.25	4.5444	6.8937
Special Editor in Sec. III		41.5	6.0663	19.1707	97	6.4897	13.2141

Subject 1 started to write a document after he had finished composing the narrative of the document in his mind. In other words, he only used both systems to make a clean copy; thus the number of DTFs from subject 1 using both Text ComposTer and the special editor was quite small.

The writing styles of Subjects 2, 3, 4, 5, and 8 were as follows. First, they enumerated keywords and ideas for a document, then selected some of these keywords and ideas, and wrote the document based on the selected ones. When using Text ComposTer, they generated an element and described a keyword set or an idea in the element. Then, they arranged the elements in the adopting space and the pending space and completed the document. Finally, some elements remaining in the pending space were collected as R-DTFs. When using the special editor, on the other hand, they first wrote down a list of ideas and keywords in the special editor; then they wrote the body of the document while referring to the list. They finally deleted the list and completed the document.

Furthermore, the process of the writing task with Text ComposTer differed between subjects 2, 3, 8 and subjects 4, 5. Subjects 2, 3, 8 first generated the elements in which ideas and keyword sets are described, and then moved all these elements to the pending space. They subsequently generated an element for a clean copy of the document, and described the body of the document in the element referring to the elements located in the pending space. Subjects 4 and 5, on the other hand, first generated the elements of keyword sets and ideas in the same manner as subjects 2, 3, 8. Then, they selected some elements, put them in the adopting space, added text in these elements, and merged them into one element by using the merge function to complete the document.

In contrast, the writing styles of subjects 6 and 7 changed depending on the system. They used Text ComposTer in the same manner as subjects 2~5 and 8, while they used the special editor in the same manner as subject 1.

From these observation results, we can assume that most people who have peculiar writing styles (like subjects 1~5 and 8) tend to adhere to their own writing styles regardless of the text-writing system.

We got replies in the interview about comparison between Text ComposTer and the special editor. As an affirmative response about Text ComposTer, subjects 3, 4, 6, 7 reported that they could create many more ideas with Text ComposTer than with the special editor. They could also write the document systematically because they could organize their thoughts better using Text ComposTer. Contrarily, as a negative response about using Text ComposTer, subjects 3, 8 reported that they felt uncomfortable writing down the body of the document in the element.

VI. DISCUSSION

This section discusses the usefulness of Text ComposTer based on the results of the user studies. At first, we discuss usability of Text ComposTer, and then we discuss whether Text ComposTer can efficiently collect R-DTFs, mainly focusing on the two issues shown in Section III.

A. Usability of Text ComposTer as a document-writing support system

There are two significant differences between Text ComposTer and usual text editors, including the special editor: 1) similar to the Art#001 system [11], the user writes a part of the document (text fragments) in an element and composes the entire document by sorting the elements based on the storyline; and 2) if the user judges that certain text fragments in an element are not necessary, he/she moves (not deletes) the element to the pending space for elimination from the final document. These differences may cause incorrect usability as a document-writing support system.

Evaluation on the usability varied depending on the writing styles of the subjects and on whether the subjects could change their styles or not. Most of the existing text editors are based on the WYSIWYG concept: the users are always viewing the final image of the document while writing it. They are familiar with this style and they often unintentionally attempt to make the final document from the beginning. However, Text ComposTer requires users to use a totally different writing style from such ordinary text editors: it requires them to make parts at first, then to assemble them to compose the final document. Therefore, the users are required to change their document-writing style when using Text ComposTer.

The subjects who responded affirmatively in the interview would be able to change their styles, or their writing style was originally similar to that of Text ComposTer. Subjects 6 and 7 actually changed their styles depending on the tool. Subjects 3 and 4 first enumerated keywords and ideas even when they used the special editor. This style is potentially similar to that of the Text

 ISBN: 978-1-61208-538-8

ComposTer. Therefore, they could quickly change the style when using Text ComposTer. Although subjects 2, 5, and 8 did not respond affirmatively, they also enumerated keywords and ideas first. This means that their writing style is similar to that of subjects 3 and 4, although they did not recognize this as an advantage of Text ComposTer. In contrast, subject 1 used both tools in the WYSIWYG editor manner. For such users, it should be necessary to give them some instructions about the writing styles beforehand.

Consequently, as for the users who are potentially familiar with the writing style of Text ComposTer, its usability is acceptable. However, as for users who adhere to the WYSIWYG writing style, its usability is not good and some instructions to change their style are necessary.

B. Can Text ComposTer efficiently collect R-DTFs?

In this subsection, we discuss whether Text ComposTer can efficiently collect meaningful R-DTFs. Furthermore, we would like to inspect whether the following two issues revealed in the pilot study were solved: 1) collected DTFs are very messy using the special editor and 2) the number of DTFs is quite small.

In Text ComposTer, only text fragments written in the elements that are finally located in the pending space are collected as R-DTFs; DTFs generated by correcting mistypes (Factor 1) and revising expressions (Factor 2) that often cause the messiness are excluded from R-DTFs. Therefore, it became able to automatically and selectively obtain meaningful R-DTFs that are text fragments generated only by Factor 3. In addition, from the results of user study 1 (see Table II), users of Text ComposTer said that R-DTFs are more useful than F-DTFs. Consequently, we can conclude that Text ComposTer can efficiently collect meaningful R-DTFs separated from F-DTFs and can also solve the first issue.

Although Text ComposTer also has the potential to solve the second issue, whether meaningful R-DTFs can be obtained depends on the usage of Text ComposTer. We designed Text ComposTer to support the entire text-writing process from the upstream to the downstream. Namely, in the beginning of the text-writing process, the author is required to create a lot of diverse ideas regardless of the context and then composes the document in a convergent manner while gradually establishing the context. In this process, each idea is evaluated and selected if it is necessary in the context. As a result, unused ideas are usually wasted, but Text ComposTer gleans them as seeds for future knowledge creations. Therefore, we expected that the users of Text ComposTer described each keyword set or idea in an element one-by-one in the beginning, then arranged the elements to compose the documents. Finally, several unused elements were moved to the pending space.

Thus, supporting and enhancing the initial divergent process of idea creation is important for increasing the number of R-DTFs and solving the second issue. Text ComposTer is equipped with this function. However, as shown in Table IV, the numbers of R-DTFs is still small and useful ones are similar to those obtained by the special editor (see Table III). Moreover, as with the usage of subjects in user study 2, Text ComposTer can be used in different ways of text-writing. To effectively increase the number of R-DTFs, some functions and/or restrictions should be added to let the users use Text ComposTer based on our expected way of text-writing.

Such additional functions and/or restrictions are necessary from the viewpoints of improving usability and of accurately gleaning R-DTFs. We need to consider improvement of the interface of elements because the negative responses from the interviews related to the elements; subjects 2, 3, 8 in user study 2 generated elements for making a clean copy of the document, which is an unexpected usage. One reason for this problem is that the element allows different usages. It allows the user to take memos of ideas and keywords, to write a part of the body of the document, and eventually to write a clean copy of the document. This feature causes the unexpected usage where the user writes the body of the document into the elements in the adopting space by referring to the ideas or keyword sets in the elements in the pending space. This causes the problem that useful knowledge and intangible waste are mixed and cannot be distinguished.

Furthermore, we need to consider eliminating the merge function from editing functions of Text ComposTer. In user study 2, subjects 4 and 5 completed the document by merging elements into one element. Text ComposTer enables users to support the text-writing process by sorting and arranging elements in the element space. However, if the elements are merged into one element once, users of Text ComposTer become unable to easily revise the document by sorting the elements. Besides, when users want to delete a part of a document (i.e., text fragment) in the merged element, Text ComposTer collects it as an F-DTF, not an R-DTF. To collect it as an R-DTF, the user needs to split the text fragment from the original element and move the split element to the pending space. Such a manipulation introduces a high cognitive load for collecting R-DTFs. Therefore, we conclude that the merge function should be eliminated from the viewpoints of both usability and of efficiently collecting R-DTFs.

One limitation of the user studies is that the number of subjects was too small to obtain statistically significant results. Additional experiments are necessary. However, the characteristics of R-DTFs and F-DTFs collected by Text ComposTer and DTFs collected by the special editor were evidently different. Therefore, by conducting the additional experiments, we would obtain statistically significant results that are almost similar to the results of the user studies shown in this paper.

VII. CONCLUSION AND FUTURE WORK

This paper described a novel writing support system called Text ComposTer that is equipped with functions to efficiently collect DTFs as a resource for supporting future knowledge creation. Text ComposTer was designed based on the findings of a pilot study that investigated what kinds of DTFs are proper to utilize. Text ComposTer has an advantage of collecting (re)utilizable DTFs because it can collect R-DTFs and F-DTFs separately. This advantage was

 ISBN: 978-1-61208-538-8

confirmed in user studies. Consequently, Text ComposTer is an effective system for collecting intangible waste that has the potential to be utilized.

Our contribution of this paper is to propose a document writing application that is equipped with an effective collection function of DTFs, in particular R-DTFs. Such applications have not ever existed because of lacking a perspective of utilizing intangible waste so far. On the other hand, this paper has not yet investigated whether the collected DTFs are actually useful or not. To totally claim the usefulness of Text ComposTer, we need to carry out investigation of utilization of collected DTFs.

In near future, we would like to improve the design of the elements to more clearly separate out useful knowledge. We would also like to create an environment for utilizing DTFs (mainly R-DTFs) for future knowledge creation, and to conduct user studies to confirm the usefulness of collected DTFs. Through these studies, we would like to create a future where people know that wasting "waste" is a waste.

ACKNOWLEDGMENT

The authors sincerely thank all of the research participants who willingly cooperated in our experiments. This work was supported by JSPS KAKENHI Grant Number JP15K12093.

REFERENCES

[1] C. Oshima, K. Nishimoto, and N. Hagita, "A Piano Duo Support System for Parents to Lead Children to Practice Musical Performances," ACM Transactions on Multimedia Computing, Communications and Applications (ACM TOMCCAP), vol. 2 issue 2, Article 9, pp. 1-21, 2007.

[2] A. Ikenoue, K. Nishimoto, and M. Unoki "iDAF-drum: Supporting Practice of Drumstick Control by Exploiting Insignificantly Delayed Auditory Feedback," Knowledge Information and Creativity Support Systems, AISC 416, Springer, pp. 483-498, 2016.

[3] C. Neuwirth, D. Kaufer, R. Chimera. and T. Gillespie, "The Notes program: A hypertext application for writing from source texts," Hypertext '87 Papers. Chapel Hill, NC. pp. 121-142, 1987.

[4] J. B. Smith, S. F. Weiss, and G. J. Ferguson, "A hypertext writing environment and its cognitive basis," Hypertext '87 Papers. Chapel Hill, NC. pp. 195-214, 1987.

[5] W. J. Hunter and J. Begoray, "A Framework for the Activities Involved in the Writing Process," The Writing Notebook, vol. 7, No. 3, pp. 40-42, 1990.

[6] S. Liao, "Expert system methodologies and applications—a decade review from 1995 to 2004," Expert Systems with Applications, vol 28, Issue 1, pp. 93–103, 2005.

[7] R. Mizoguchi, J. Vanwelkenhuysen, M. Ikeda, "Task ontology for reuse of problem solving knowledge," Towards Very Large Knowledge Bases: Knowledge Building & Knowledge Sharing, IOS, pp. 46-59, 1995.

[8] H. Shibata and K. Hori, "A system to support long-term creative thinking in daily life and its evaluation," In Proceedings of the 4th conference on Creativity & Cognition (C&C '02). ACM, New York, USA, pp. 142-149, 2002.

[9] M. Sharmin, L. Bergman, J. Lu, and R. Konuru, "On slide-based contextual cues for presentation reuse," In Proceedings of the 2012 ACM international conference on Intelligent User Interfaces (IUI '12), ACM, New York, USA, pp. 129-138, 2012.

[10] V. 'V. Simbelis, P. Ferreira, E. Vaara, J. Laaksolahti, and K. Höök, "Repurposing Bits and Pieces of the Digital," In Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16), ACM, New York, USA, pp. 840-851, 2016.

[11] K. Nakakoji, Y. Yamamoto, B. N. Reeves, and S. Takada, "Two-Dimensional Positioning as a Means for Reflection in Design," Proceedings of Design of Interactive Systems (DIS'2000), ACM , New York, USA, pp.145-154, 2000.

 ISBN: 978-1-61208-538-8