UNIVERSITY OF JOENSUU

DEPARTMENT OF COMPUTER SCIENCE AND STATISTICS

DISSERTATIONS 25

Jussi Nuutinen

Nucleus Model for Designing Social Mindtools: Woven Stories

Academic dissertation

To be presented, with the permission of the Faculty of Science of the University of Joensuu, for public criticism in the Louhela Auditorium of the Science Park, Länsikatu 15, Joensuu, on September 18th, 2009, at 12 noon.

UNIVERSITY OF JOENSUU

2009

Supervisors	Professor Erkki Sutinen Department of Computer Science and Statistics University of Joensuu Joensuu, FINLAND
	Associate Professor Piet Kommers Deparment of Behavioral Sciences University of Twente Twente, The NETHERLANDS
Reviewers	Professor Paul De Bra Department of Computer Science Eindhoven University of Technology Eindhoven, The NETHERLANDS
	Professor Nian-Shing Chen Information Management Department National Sun Yat-Sen University Kaohsiung, TAIWAN
Opponent	Associate Professor Mike Joy Deparment of Computer Science University of Warwick Coventry, UK

ISBN 978-952-219-266-0 (printed)
ISBN 978-952-219-267-7 (PDF)
ISSN 1796-8100 (printed)
ISSN 1796-8119 (PDF)
Computing Reviews (1998) Classification: H.4.m, H.5.3, H.5.4, K.3.1, J.m

Joensuun yliopistopaino
Joensuu 2009

Nucleus Model for Designing Social Mindtools: Woven Stories

Jussi Nuutinen

Department of Computer Science and Statistics
University of Joensuu
P.O.Box 111, FIN-80101 Joensuu, FINLAND
jussi.nuutinen@cs.joensuu.fi

University of Joensuu, Department of Computer Science and Statistics, Dissertations 25
Joensuu, 2009, 180 pages

Abstract

Due to the vast amount of new technology, the development of educational technology seems to concentrate on relatively complex tools and technologies. At the same time, there is still a need for simple, generalisable collaborative knowledge constructions tools; that is, social mindtools.

Woven Stories is a concept for a social mindtool. It uses a simple formalism to provide its users with an interesting and flexible approach to story, or any document, writing. It allows both synchronous and asynchronous collaboration and aims to work as a tool where all provided information is equally important.

Social mindtools are a subset of mindtools. These tools supply a learning community with the possibility to collaboratively construct and present knowledge. The main differences to mindtools are that these tools must also support knowledge presentation to certain degree. Furthermore, in order to support collaboration, these tools must provide users with awareness related information and must have features to support communication. Thus the requirements of social mindtools can be divided to three distinct layers: concept, awareness and communication.

Based on the concept of Woven Stories, a social mindtool called Loom was developed during 2003–2008. The evaluation of Loom is presented in six different case studies. The results show that Loom is best suited to learning tasks where the knowledge to be processed has strong sequential relationships. Furthermore, knowledge that includes time series, such as is contained in stories and narratives, is most valuable for the learning tasks. The findings suggest that the usage of Loom boosts users' imagination and creativity. It can be used for several different application domains, for example in debating, adventure game scripting and concept mapping.

Based on the results of a retrospective analysis of the design, implementation and evaluation of Loom, the Nucleus Model for designing social mindtools is introduced. This model is founded on a layered model of concept, awareness and communication. The Nucleus Model is a potential approach for designing social mindtools in an efficient way. It makes it possible to integrate research to development fluently, and provides guidelines for development on such detail that it is easy to follow.

Keywords: mindtools; social mindtools; collaborative writing; Woven Stories; collaborative learning; computer supported collaborative learning; collaborative work; computer supported collaborative work

Acknowledgements

I wish to express my gratitude to my supervisors Professor Erkki Sutinen and Professor Piet Kommers. Their valuable guidance helped me through the difficult moments. I thank the reviewers of this thesis, Professor Nian-Shing Chen and Professor Paul De Bra for their insightful comments. I also thank Dr. Antony Harfield, who edited the language of this thesis.

The support from colleagues with whom I have had the honour and pleasure to write and publish has been important. I am happy to list them all here: Roman Bednarik, Adele Botha, Renald Buter, Teemu Laine, Kimmo Liinamaa, Niko Myller, Ed Noyons and Hannu Vanharanta. Furthermore, I am grateful to Matti Tedre and Justus Randolph for their comments and support. I also want to mention Mikko Taivainen, whose programming skills were essential during the implementation of Loom.

Thanks to everyone who have been linked to this thesis on a way or another: my parents Erkki and Marja Nuutinen, my sister Outi Nuutinen, Marcus Duveskog, Petri Gerdt, Elina Hartikainen, Ilja Jetsu, Ilkka Jormanainen, Tuomo Kakkonen, Esko Kähkönen, Javier López, Andrés Moreno, Maxim Mozgovoy, Phyllis Ngai, Ulla Pötsönen, Timo Rui, Irakli Tskhvediani, Carolina Islas Sedano, Jarkko Suhonen, Mikko Vesisenaho, Marjo Virnes, Scifest Africa, administrative personnel of the department and all the students and kids who participated the workshops and classes. Special thanks to everyone I forgot.

Perhaps the most influential person for finally finalising this thesis is, however, my daughter Vanamo (born 4th June, 2009). Her expected delivery date was on May 27th and this thesis was submitted to review on May 25th. An observant reader might see a dependency here. I also wish to thank my wife Anu. Her silent[1] support has been important throughout the whole process.

This work was funded by the Department of Computer Science and Statistics at the University of Joensuu, Finland, the Finnish Funding Agency for Technology and Innovation (TEKES) and the East Finland Graduate School in Computer Science and Engineering (ECSE). During my studies I have been a student of IMPDET (International Multidisciplinary PhD Studies in Educational Technology).

I dedicate this thesis to my grandmother Bertta Elisabet Salonen.

Joensuu, August 27th, 2009

Jussi Nuutinen

[1]She saw my pressure, and never asked when my thesis will be ready.

List of Terms and Abbreviations

Activity Theory	The theory that maps subjects activities to object, and community.
AJAX	Asynchronous JavaScript and XML.
Awareness	Features that allow users to know what other users have done, what they are doing etc.
CSCL	Computer Supported Collaborative Learning
CSCW	Computer Supported Collaborative Work
CSS	Cascading StyleSheets
Design	A process in which an application or a feature is planned.
Development	A process that aims at creating a finished product. Includes design and implementation.
Edge	A link that connects the individual sections in a woven story to form storypaths.
Episode	A part of a woven story, consisting of several sections, which describes a logical part of the story.
HSQLDB	HyperSQL DataBase
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
Implementation	A process where planned features or application are created.
JSON	JavaScript Object Notation

Knowledge building	A process that aims at creating new information.
LAN	Local Area Network
Loom	An application that implements the concept of Woven Stories.
Meaningful learning	The process in which a learner can relate learnt concepts to his/her existing knowledge structures.
Mindtool	A tool for processing information and building knowledge.
Nucleus Model	A model for designing and implementing social mindtools.
Section	An individual part; a building block, of a woven story.
Social Mindtool	A collaborative mindtool.
Storypath	A path in a woven story from a selected beginning to any reachable end.
Woven Stories	A concept for collaborative writing.
Woven Stories application	see *Loom*.
Woven story	A product of using a Woven Stories tool.
WS2	Woven Stories 2. An earlier prototype implementation of Woven Stories.
WWW	World Wide Web
WYSIWIS	What You See Is What I See
WYSIWYG	What You See Is What You Get
XML	Extensible Markup Language

Contents

Chapter 1

Introduction

Woven Stories (WS) is a concept for collaborative writing. It is based on a simple idea to produce the text in small pieces called *sections* which are linked together with arrows, called *edges*, to form storylines. Thus a woven story is a graph of sections where each individual path forms a story of its own. During the research reported in this thesis, a computer application based on the concept of Woven Stories, Loom, was implemented and analysed.

This thesis has two main results. Firstly, it demonstrates that applications based on the concept of Woven Stories are *social mindtools*. These social mindtools are simple and generalisable tools for collaborative knowledge building and processing. Secondly, the thesis shows that the design and implementation process of social mindtools should proceed in certain stages. The previous claim is supported with evaluation of Loom in several case studies. The latter one is based on the lessons learnt during the design and implementation of Loom. Furthermore, this thesis emphasises the importance of keeping educational applications simple, yet extendable for maximised flexibility.

Because of the continuous flow of new technologies, research in the field of educational technology tends to concentrate on the use of mobile technologies, virtual worlds and other relatively complex tools and technologies. Given the popularity of technology-driven designs of this kind in research, it is easy to overlook the fact that learning is inextricably bound up with thinking and that simple solutions can often be crucial in the evolution of educational research.

Tools are devices, appliances and methods that ease people's tasks. Tools enable people to extend their physical and cognitive capabilities. According to Jonassen, while most tools are very specific and meant for certain purposes, there are more generalisable tools that can facilitate cognitive processing [67]. Cognitive capabilities or processes that tools are able to facilitate are attention, perception, learning, memory, language, problem solving, reasoning, and thinking [47]. Tools that are capable of supporting and facilitating these largely individual processes are called *cognitive tools* [67] or *mindtools* [68] as they are referred to in this thesis.

Jonassen [67] notes: "Rather than developing more powerful teaching software, we should be teaching learners how to think more efficiently." In order to accomplish what Jonassen recommends, there is a need to develop a range of simple and generic thinking tools that can be taught to learners. The development of simple tools of this kind is predicated on the realisation that the primary purpose of learning is not how to use the tool but how to think efficiently.

While designing educational tools and software, it is helpful to bear in mind that contemporary scholarly communities exist through a web of international communication and interaction, and that it is important to be able to support collaboration among these communities. In modern-day educational institutions, for example, on-line teaching and learning are being used ever more widely. This ubiquity of on-line teaching and learning in education presupposes a need for effective methods of collaborative knowledge processing. Internet facilities, such as email, discussion forums and Usenet news, have become so widely used for collaboration and information delivery that they seem to have become indispensable [83], even though they are not equally effective in all situations.

The development of Internet-based tools, from the traditional web to Web 2.0 [132], and from information delivery to interpersonal collaboration and group-based knowledge construction and creation, indicates the shared direction in which the majority of social mindtools are heading.

The development of social mindtools is demanding. It can be a long journey from conception to implementation, in order to take the original idea to the working version of the application. Currently, there are neither methods nor models for the development of these generalisable applications. Traditional software development processes are often not applicable, since these require that it is known beforehand what the application is supposed to do and how.

It is unclear what the best methods are for designing social mindtools that contain the necessary features for "teaching to think efficiently". This open question is one that this thesis aims to address. While creators of educational media and educators themselves can use intuitive or ad-hoc approaches to design, or could embrace some classical software engineering approach, the outcome of such a process would be uncertain, considering the complexity of the domain. In order to make the design process more efficient and focused, specialised models are needed.

This thesis is a result of a long and challenging experimental work with the Woven Stories. Woven Stories originated as a means for writing stories, it subsequently proved its usefulness as a mindtool [68] in several other application areas such as progress reporting, collaborative learning of programming and corporate strategy planning ([94], [89], [99], [110]). Woven Stories provides an example of a useful tool that is characterised by simplicity, generalisability and collaborative dimensions. It also seems to overcome the fact that it is hard and complicated to support collaborative writing [74, 126]. There are tools for creating hypertext documents and there are applications that facilitate collaboration, but the approach used in Woven Stories concept is a novel combination of existing tools.

Based on literature analysis and the analysis of the design and implementation process of Loom, a set of commonly required features for social mindtools are presented. Furthermore, based on these requirements and a retrospective analysis of the design, implementation and evaluation of Loom, a layered model for the development and research of the social mindtools is proposed. This layered model is called the Nucleus Model.

Chapter 2

Questions and Methods

2.1 Research Questions

This thesis concentrates on the design and implementation of social mindtools. There were three goals: to design and implement a functioning social mindtool based on the concept of Woven Stories, to evaluate the developed social mindtool, and to formulate a generalisable approach for designing and implementing social mindtools. The latter is done in order to provide researchers and programmers with guidelines and a framework to be used in the demanding process of developing social mindtools.

In order to achieve the goals of this study, five research questions have been set. These questions, with references to Chapters where each question is answered, are presented in Table 2.1.

Below, I explain the research questions in detail.

What features characterise a social mindtool within the set of mindtools?

Social mindtools are a subset of mindtools. They are mindtools that are used collaboratively. Due to their collaborative nature, the design of these tools is not straightforward. Various aspects of both collaboration and mindtools need to be considered. There is neither a generally accepted framework nor a model for the development of mindtools, and therefore in order to answer this question a framework for these tools needs to be created. This question is considered and answered in Chapter 4.

How are the characteristics of a social mindtool present in the architecture of Woven Stories?

Table 2.1: The research questions of this study.

	Question	Chapter
Q1	What features characterise a social mindtool within the set of mindtools?	4
Q2	How are the characteristics of a social mindtool present in the architecture of Woven Stories?	5
Q3	How do the characteristics of a social mindtool influence the technical implementation of Woven Stories application?	6
Q4	What kinds of learning tasks does Woven Stories support?	7
Q5	How does the Nucleus Model accommodate the design processes for Woven Stories and other social mindtools?	8

In order to implement a social mindtool based on the concept of Woven Stories the architecture of the application needs to be designed. In order for the application to meet the definition and requirements of a social mindtool, the guidelines for these tools (i.e. the answers to Q1) are to be followed. Each individual social mindtool has a different set of requirements and features that are to be implemented and by answering this question I provide the basic architecture for Woven Stories based applications. This question is answered in Chapter 5.

> *How do the characteristics of a social mindtool influence the technical implementation of Woven Stories application?*

This question sets the starting point for the implementation of the Woven Stories application, Loom, and can be considered as the main contribution of the thesis. The question is answered in Chapter 6 by introducing the current implementation of Loom.

> *What kinds of learning tasks does Woven Stories support?*

This question is explored by analysing first the experiences from six case studies undertaken between 2004 and 2008. The question is answered in Chapter 7 by presenting an analysis of the case studies. Furthermore, a comparison between Woven Stories and Wikis, another collaborative writing tool, is presented.

During the evaluation process of Loom it became evident that mindtools should be implemented in stages. By implementing the application gradually, it becomes easier to evaluate the tool and to concentrate on certain issues in the evaluation. The order is based on a framework for social mindtools and gradually builds features starting from the very concept of a social mindtool. In order to analyse this more carefully, the following research question was introduced.

How does the Nucleus Model accommodate the design processes for Woven Stories and other social mindtools?

This question is answered in Chapter 8. The problems in the evaluation led to consider that there has to be specific ways to link the development and research of social mindtools in a meaningful way. This means that development and research should support each other and be as fluent as possible. Furthermore, it should be possible to carry out the work in close collaboration between computer scientists and educationalists.

The interrelationships between the questions presented in this section are shown in Figure 2.1.

2.2 Research Methods

In order to answer the questions presented in Section 2.1 several methods have been used. These methods are shown in Table 2.2 in contrast to the questions where they have been applied.

Table 2.2: The research methods used in this study.

Question	**Method**
Q1	Literature analysis
Q2	Design, Technical
Q3	Software engineering, Technical
Q4	Evaluation, Analysis
Q5	Retrospective Analysis, Synthesis

The method used to answer research question Q1 was literature analysis. Relevant literature is derived from the journals and conferences listed below. Furthermore, for specific needs articles has been searched by utilising the the databases of ACM[1] and IEEE.[2] The main journals have been:

- Computer Supported Cooperative Work
- International Journal of Computer-Supported Collaborative Learning
- Communications of the ACM

Important conferences include:

[1]`http://portal.acm.org/`
[2]`http://ieeexplore.ieee.org/`

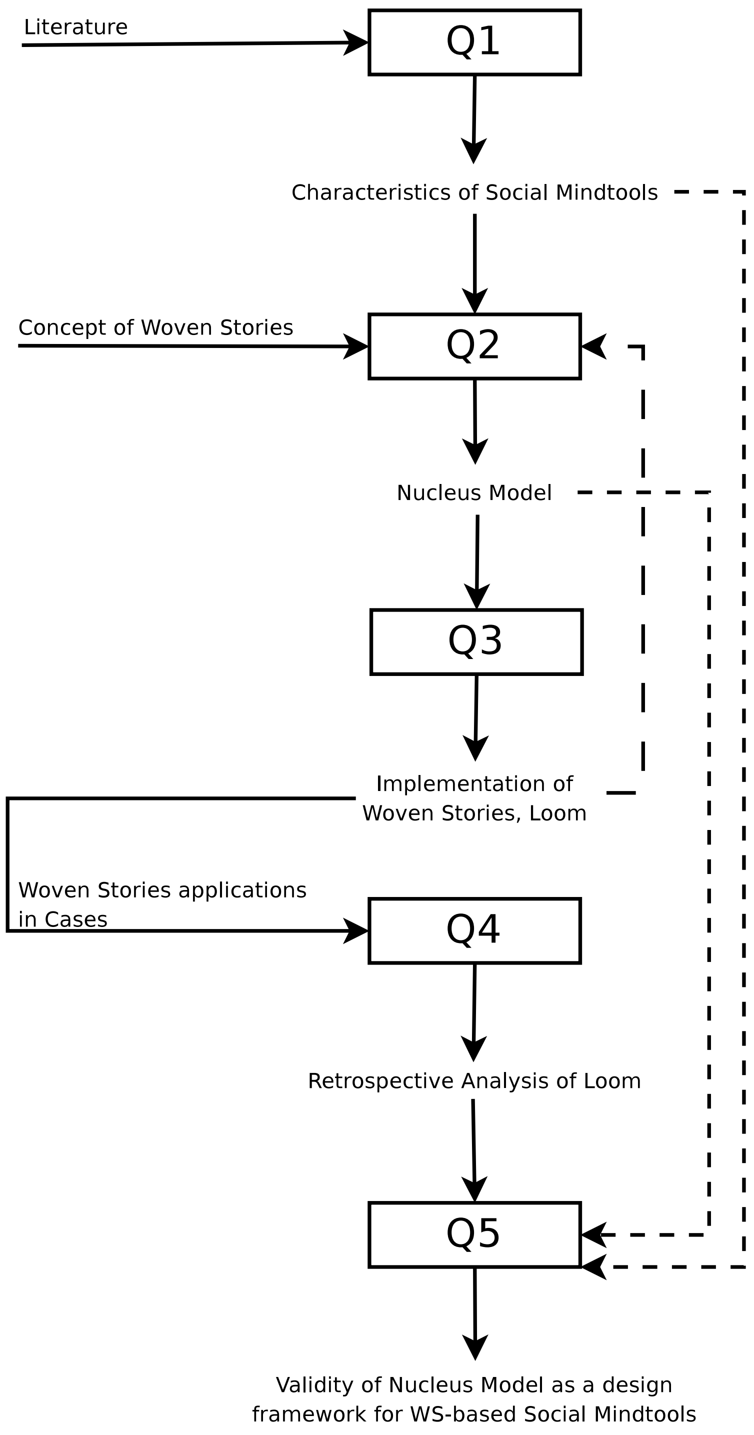

Literature
Q1
Characteristics of Social Mindtools
Concept of Woven Stories
Q2
Nucleus Model
Q3
Implementation of Woven Stories, Loom
Woven Stories applications in Cases
Q4
Retrospective Analysis of Loom
Q5
Validity of Nucleus Model as a design framework for WS-based Social Mindtools

Figure 2.1: Interrelationships between research questions.

- SIGCHI Conference on Human Factors in Computing Systems
- ACM Conference on Computer Supported Cooperative Work
- IEEE International Conference on Advanced Learning Technologies
- ACM Conference on Hypertext and Hypermedia

The literature analysis proceeded in cycles. First the concept of mindtools was defined, which then formed a need to seek information about learning and knowledge building. This led to literature that considers knowledge in general. During the development process literature about collaborative applications and their requirements was sought. Figure 2.2 illustrates this cycle.

The answers to research questions Q2 and Q3 were obtained by analysing the concept of Woven Stories and the answer of Q1. The concept of Woven Stories was divided into construction blocks that are needed to transform the manual Woven Stories (see Section 3.1.3) to computer based application. Then, it was considered which features were needed to meet the requirements of a social mindtool. Based on these, an architecture for a social mindtool based on Woven Stories is given and an implementation of Loom is presented.

Question Q4 was answered by utilising the various use cases in which Loom has been used. A selection of analysis methods has been used in each case. These methods included:

- observation,
- interviews,
- questionnaires, and
- analysis of the WS artifacts of the test persons.

The actual use of these methods is described in Section 7 under each case where the method was used.

Up to this point in the research questions, the study has been following the *Development Research Approach* [107]. This is an iterative approach, where an application or a concept is developed based on existing theories and evaluations. This idea, and the structure of the research for answering research questions Q1, Q2, Q3 and Q4 is represented in Figure 2.2.

For research question Q5 the main method has been *retrospective analysis.* The retrospective analysis is based on the experiences of the previous work, in contrast to what was known at the time when analysis was done. The answers and the process to achieve the answers of research questions Q2, Q3 and Q4 were analysed

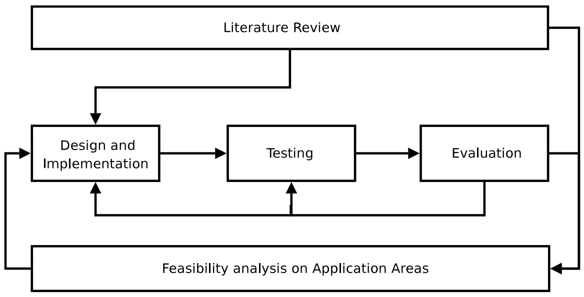

Literature Review
Design and Implementation
Testing
Evaluation
Feasibility analysis on Application Areas

Figure 2.2: Research followed this pattern before the retrospective analysis was done.

thoroughly. All this was done after answering questions Q2, Q3 and Q4, thus the name retrospective analysis. The idea of this method was to analyse the design and implementation process in contrast to the experiences of the case studies. Based on the difficulties met during the analysis of Q4, it was possible to determine important phases from the design process. The results of this analysis, and answers to the other research questions, made it possible to build a generic model for the development of social mindtools.

The thesis is strongly based on lessons learnt during the design, implementation and evaluation process of the Woven Stories application.

2.3 Main Results and Contributions

As part of this study, a computer application, Loom, was developed based on the Woven Stories concept. This is one of the main contributions of this work. Loom was designed and implemented by the author, with Mikko Taivainen and Teemu Laine providing important help during the implementation.

Eight articles related to Woven Stories have been published as results of the work in this thesis :

P1 Nuutinen, J., Liinamaa, K., Sutinen, E., and Vanharanta, H. Strategist's Learning Space. In *Web-Based Education* (2004), Acta Press, pp. 544–548. [98].

In this article I was the main author and wrote about the woven stories application and the educational theories behind it. Kimmo Liinamaa wrote about the strategy process. This paper was the first to introduce the extended version of the Woven Stories application called *Woven Strategies.*

P2 Nuutinen, J., Liinamaa, K., Sutinen, E., and Vanharanta, H. Woven

Stories as a Tool for Corporate Strategy Planning. In *Web Based Communities 2004* (Lisbon, 2004), IADIS Press, pp. 438–441. [99].

The Woven Strategies platform was introduced and analysed in this article from the community point of view. I was the main author and wrote about the application. This paper took a new viewpoint to the Woven Strategies platform, as a tool that would support an organisation's objectives to transform into a web-based community or learning organisation.

P3 Nuutinen, J., Laine, T., Sutinen, E., Buter, R., and Noyons, E. Problem and Content Development to Support Evaluation of Science. In *Proceedings of the E-Learn 2004*. [97].

This paper described how to apply Woven Stories application to bibliometrics. It showed that the concept can be utilised in various fields of science. In this paper Woven Stories was used to gather the data from the experts for the bibliometrics application, in order to define a field of science.

P4 Liinamaa, K., Nuutinen, J., Sutinen, E., and Vanharanta, H. Collaborative Strategic Planning On-line. *Psychnology 2*, 2 (2004). [78].

The Woven Strategies concept was elaborated in this paper. Kimmo Liinamaa was the main author and this paper was mostly based on his Master's thesis. However, it should be noted that this paper would not exist without Woven Stories application.

P5 Myller, N., and Nuutinen, J. JeCo: Combining Program Visualization and Story Weaving. *Informatics in Education 5*, 2 (2006), 255–264. [89].

This article describes a prototype of an application called JeCo. JeCo is an application based on the Loom which allows collaborative learning of programming by using the Jeliot [85] application to visualise the Java programming code. Niko Myller wrote the parts of the paper that considered Jeliot and I wrote parts that consider the Woven Stories application.

P6 Nuutinen, J., Bednarik, R., and Sutinen, E. A layered approach to the development process of social mindtools. In *Proceedings of EdMedia 2008*, pp. 2109–2118. [95].

The Nucleus Model as a development approach for social mindtools was presented in this paper for the first time. This article was written by me with support from Roman Bednarik.

P7 Nuutinen, J., Botha, A., Sutinen, E., and Kommers, P. From mindtools to social mindtools: Collaborative writing with woven stories. *British Journal of Educational Technology*. (in press). [96].

This article analyses the concept of Woven Stories, the concept of the social mindtools and the debating case which is also presented in this thesis in Section 7.1.6. Furthermore, it includes a comparison between the concept of Woven Stories to that of Wikis. I was the main author in this paper. Adele Botha provided me with help on Activity Theory.

P8 Nuutinen, J. and Sutinen, E. Information Retrieval Techniques for Collaborative Text Searches. *Proceedings of ICALT 2009*, pp. 390–392. [101].

This paper describes the need to provide the users of Loom with search functionality that enables them to find relevant contents from a large collection of woven story documents. In order to find proper algorithms for this purpose, several algorithms were selected and analysed. I conducted the analysis and was the main author of this article. The results of this analysis are presented in Section 6.6.

In addition to the articles published about *Woven Strategies*, my research played an important role in Markku Salo's PhD thesis *Woven Strategies* [110].

Table 2.3 represents the relationship between the published articles and the research questions of this thesis.

Table 2.3: The research questions of this study in relation to published articles and Chapters of this thesis.

Question	**Paper(s)**	**Chapter(s)**
Q1	P1, P7	4
Q2	P1	5
Q3	P1	6
Q4	P1, P2, P3, P4, P5, P6, P7, P8	7
Q5	P6	8

An important contribution of this thesis is the application based on the concept of Woven Stories; Loom (see Chapter 6). Based on the evaluations and experiences it is useful in several different application areas and can be used in real settings as reported in Section 7.1. Furthermore, based on the experiences and results of the design and implementation of Loom and as a result of the retrospective analysis, a proof of concept, web-based version of Woven Stories, WS@Web was implemented.

2.4 Structure of the Thesis

In order to make the distinction between the concept, application, and the product of Woven Stories, I use *concept of Woven Stories* for the concept, *Loom* or *the Woven Stories application* for the application and *woven story* or *document* for a product of using the Woven Stories application.

In Chapter 3, I introduce the concept of Woven Stories and a method to employ it without computers. In this same chapter the previous prototypes of the concept are introduced.

Chapter 4 outlines the relevant theoretical framework in order to position the Woven Stories. This chapter answers research question Q1 and defines the concept of social mindtools and presents their requirements as a framework.

Chapters 5 and 6 introduce the current architecture and implementation of the Woven Stories application, Loom. These chapters give an overview of the application as well as its capabilities and also provide answer for research question Q2 and Q3.

In Chapter 7 I analyse six different cases where Loom has been utilised. These cases are all different from each other and thus provide examples from the various application areas where Woven Stories can be applied. Furthermore, lessons learnt from these studies and a comparison between Woven Stories and Wikis is given.

The Nucleus Model and the answer to research question Q5 is presented in Chapter 8. This chapter covers the idea of the model and provides suggestions on the research and development process of social mindtools.

Finally, I conclude my findings and present conclusions with potential future challenges.

Chapter 3

The Evolution of Woven Stories

The concept of Woven Stories was first introduced in 1999 [60]. Since that time the applications implementing the Woven Stories concept in computer systems have advanced considerably. At the same time the concept of Woven Stories has evolved. By no means were any of these prototypes finished tools but each of them was an improved version of its predecessors. The concept of Woven Stories, because it is so simple and potent, is worth investigating for purposes of research. The prototypes and their use have shown that the concept has several interesting application areas.

This chapter covers the concept of Woven Stories and briefly introduces existing and current prototypes.

3.1 Concept of Woven Stories

The name of the concept, *Woven Stories*, deserves a review before going more deeply into the concept itself. The Oxford English Dictionary [103] gives four explanations to "woven", of which the following two are the most relevant:

1. That has undergone the process of weaving; formed or fabricated by weaving.
2. Formed by interlacing or intertwining after the manner of weaving.

The same dictionary defines weaving as: *The action of the v. WEAVE; esp. the operation of forming cloth or other stuff by the interlacing of yarn or other filaments in a loom.*

Based on those dictionary definitions, if something is *woven*, it has been made of several fibres or fabrics and has undergone a long process. The product is durable and strong due to its structure because all the small parts support each other. Imagine a knitted sweater. Something as fragile as woollen string has been transformed into a warm and durable piece of clothing. But why *Woven Stories*? What does it mean when we *weave* or *intertwine* stories?

3.1.1 Structure of Stories

Stories or *narratives* that are printed in books, told to children by grandparents, or spread on the streets are normally sequential. Those stories have a beginning, and an ending, and passages that bind these two together. Hence, most stories can be divided into separate parts.

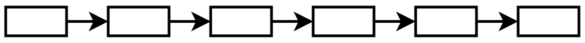


Figure 3.1: Visualisation of a traditional story with six sequential parts. The leftmost part is the start of the story and the rightmost part is the ending.

The visualisation of a traditional story, as defined above, could be represented as it is in Figure 3.1. The parts of the story are visualised separately as boxes and the progress of the story is visualised with the arrows. If the boxes were filled in with text, there would actually be a real story which could be read from the visualisation. Thus the story would form a simple graph that contains just one path.

The stories are in a constant state of change. For example, urban legends are notorious for constantly changing. The stories carry the same message (e.g. that alligators live in the sewers of New York[1]), but they keep changing, sometimes even rapidly. One story might state that an alligator ate a man who was taking a shower and another story might state that it ate a maintenance man who was cleaning a blockage from the sewer. What this actually means is that these stories have so much in common that they actually are two altered versions of the same story. They might have a different beginning or ending, but they still have much in common, so much that they could share common parts.

How would it be possible to visualise the dynamic evolution of these urban legends or rumours, as they are often called? There are certain common parts and some parts that differ from each other. It is obvious that the visualisation presented in Figure 3.1 is not an option. It should be somehow made possible to visualise the parts that are not common, but still maintain the common plot. One option is presented in Figure 3.2, which visualises two stories that share the same skeleton.

The visualisation presented in Figure 3.2 actually works rather well. If the boxes were filled in with text, the reader should still be able to follow the two, or actually

[1]see e.g. `http://en.wikipedia.org/wiki/Sewer_alligator`

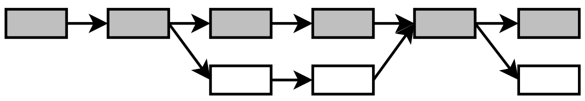


Figure 3.2: Visualisation of two urban legends sharing the same skeleton. The parts of the original story are grey and the parts that have been modified from the original story are white.

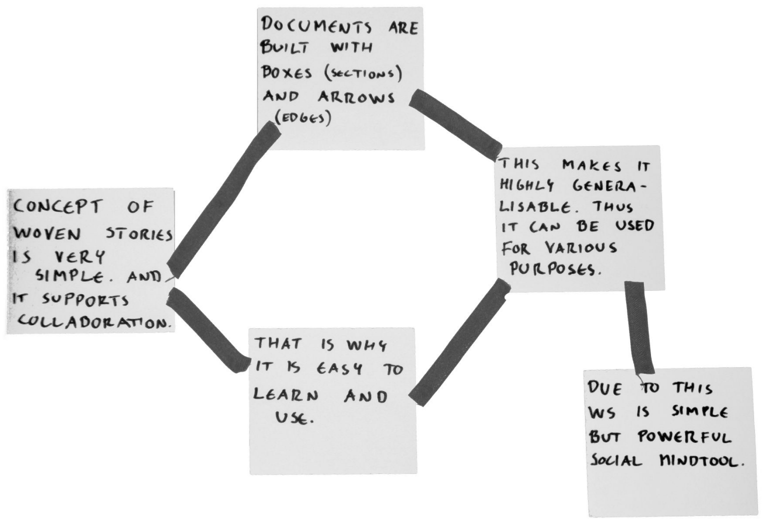

CONCEPT OF WOVEN STORIES IS VERY SIMPLE. AND IT SUPPORTS COLLADORATION.
DOCUMENTS ARE BUILT WITH BOXES (SECTIONS) AND ARROWS (EDGES)
THAT IS WHY IT IS EASY TO LEARN AND USE.
THIS MAKES IT HIGHLY GENERA-LISABLE. THUS IT CAN BE USED FOR VARIOUS PURPOSES.
DUE TO THIS WS IS SIMPLE BUT POWERFUL SOCIAL MINDTOOL.

Figure 3.3: An example of a woven story that has been created with common office tools. In this case, the story space has been a office meeting table. This woven story has five sections, five links and two storylines (the story should be read from left to right).

four different stories. Thus, there actually is one story that has many alternative storylines, like urban legends and rumours usually have.

3.1.2 The Concept

The concept of Woven Stories is a mixture of *concept mapping* [93], *flow charts* [32], *collaborative writing* [79], *graphs* [19, pp. 119–164] and *finite automata* [63, pp. 37–81]. Table 3.1 presents what features each of these concepts have contributed to the concept of the Woven Stories. Furthermore, the concept has similarities to hypertext (see section 4.4) and it builds especially on the visual representations of hypertext used in various hypertext systems. The document, a woven story, is visualised as a graph that contains nodes and edges just like a graph or a concept map. Due to its nature, a woven story is a directed graph that may (but most likely

Table 3.1: Features from different concepts to Woven Stories.

Concept	**Features to WS**	**Difference with WS**
Concept Mapping	Semantics of the links	Amount of text in nodes
Flow Charts	Flow of the stories	Amount of text in nodes
Collaborative Writing	Way to write the stories	Way to represent the stories
Graphs	Graphical representation	Contents of the nodes
Finite Automata	Concept of start and end states as well as the transitions	Contents of the nodes

Figure 3.4: The tools needed to create a woven story are indeed simple.

does not) contain cycles. Each of the nodes of the graph contains a piece of text. The visualisation of a woven story is similar to the visualisation of a concept map. Since the graph represents a story, each of the paths in the graph forms a separate story. While reading the story, the reader must follow the flow of the story, which can also represent a process, as is case with flow charts.

The actual woven story, the document, is a result of the collaboration between several authors in a shared *story space*. This story space allows users to add nodes, called *sections*, and to add edges to the document. In the story space the users can construct the story as if it were a directed graph of pieces of text. The texts of the story are in sections. Each section contains text that the original author can edit.

A woven story can be created without a computer (see Fig. 3.3). What is needed is an empty wall or table, a stack of Post-It notes, pens, scissors, tape or Blu-Tack, and string (see Fig. 3.4). The story space is a wall or a table, the sections are

Post-It notes and the edges are pieces of string. The woven story begins by one person writing a piece of text onto a Post-It note and then placing it on the wall. The others can then read what that person wrote. Another person can continue the story by writing a continuation to that piece of text onto a new Post-It note or write a completely new story. That Post-It note is placed on the wall and a piece of string connects the two Post-It notes. The string represents the flow of the story. See Section 3.1.3 and Figure 3.6 for an illustrative example.

Since it is impossible to determine the direction of the links with the string, the group has to agree on the direction in which the stories are to be read (for example from left to right). Other limitations include e.g. that it is hard to relocate an existing section and that the contents of a section are limited to the size of the Post-It notes used. It is also possible that someone can remove or alter the sections provided by other members of the group.

Figure 3.5 shows what a woven story might look like when created in a computer environment. Figure 3.5 is a woven story about a day at a zoo, written by three authors Jussi, Ville and Kalle. Each of the authors had quite a similar day at the zoo, though there had been individual differences too.

Let us first concentrate on Ville's day at the zoo. He first saw the giraffes, then he had a hamburger, and finally he went to see the tigers. Jussi also went to see the giraffes but instead of getting a hamburger after seeing the giraffes, he went to see the lions first and then had an ice cream. Like Ville, Jussi went to see the tigers.

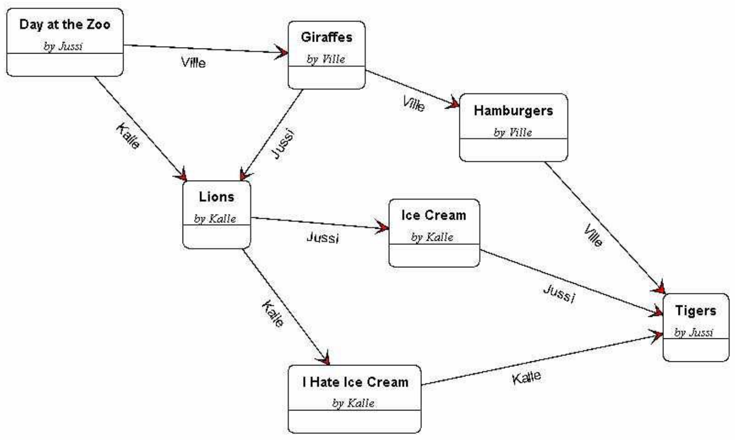


Figure 3.5: An example of a woven story created with a computer environment. Note that only the titles of sections are shown. Contents are shown on a separate pop-up window.

While reading a woven story, one should not concentrate only on the facts that an individual has written. It is as important to concentrate on which sections the person has linked together. It is of no use to rewrite parts that someone else has already written. Instead it is best to reuse the part as a part of your own story, like writers of the story in Figure 3.5 had done. By linking sections together, an individual author can compose new stories by reusing existing story sections.

The result of using the Woven Stories concept is not an unambiguous story. It is a versatile combination of stories within one document. A woven story is a combination of sections and edges in which the sections include the contents of the story. The edges link the sections together and thus represent the flow of the story. With edges and sections, a complex graph can be produced. All the paths that can be followed should form sensible stories.

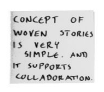
Due to its graphlike nature, a woven story can be understood as a hypertext document. Each section of text is followed by zero or more links (edges) that guide the reader to new sections. A collection of hypertext documents on the Internet is visualised in a browser as a text document, but in Woven Stories it is visualised as a graph. Understanding the structure of the website can be difficult without a proper *sitemap*. Woven Stories solves this problem by using the graph as an interface for the stories, thus making it easy to understand the structure of even complicated woven stories. The semantics of the links in Woven Stories are different. While in hypertext the links are embedded in the text, in Woven Stories the links have similar semantics as graphs - the link can only be drawn from one section to another section.

One of the strengths of the Woven Stories concept is its simplicity. Even though the basic concept does not have many rules and functions, it can still be used for several purposes. Due to its flexibility and collaborative nature, Woven Stories can be seen as a social mindtool (see Section 4.7). The concept offers a few basic functions to handle the data and the users create the rest. Since the concept is flexible and has only few restrictions, it enables the user to employ it freely.

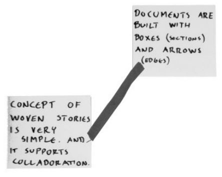
3.1.3 Creating a Woven Story

Even though Woven Stories was originally meant to be a collaboration tool, it can also be used by a single user. Chapter 7 describes the variety of possible uses for Woven Stories in more detail. This section gives a brief example of a basic Woven Stories session, with common tools (as shown in Figure 3.4).

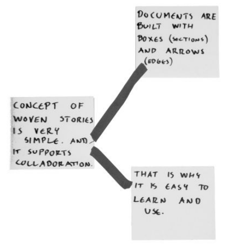
The story is initiated by one member of the group, who writes a piece of text on a Post-It note and places it on the wall, as shown in Figure 3.6(a). This text does not necessarily need to be the start of the story, it can be any part, even the end. The first section is rather important for the process, since it will most likely guide


CONCEPT OF WOVEN STORIES IS VERY SIMPLE. AND IT SUPPORTS COLLABORATION.

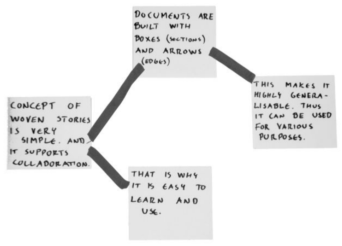
(a) The first section has been added.


DOCUMENTS ARE BUILT WITH BOXES (SECTIONS) AND ARROWS (EDGES)
CONCEPT OF WOVEN STORIES IS VERY SIMPLE. AND IT SUPPORTS COLLABORATION.

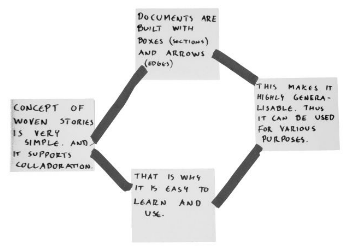
(b) The second section has been added and a link to it from the previous section.


DOCUMENTS ARE BUILT WITH BOXES (SECTIONS) AND ARROWS (EDGES)
CONCEPT OF WOVEN STORIES IS VERY SIMPLE. AND IT SUPPORTS COLLABORATION.
THAT IS WHY IT IS EASY TO LEARN AND USE.

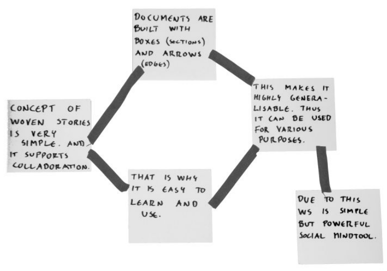
(c) One more section added.


DOCUMENTS ARE BUILT WITH BOXES (SECTIONS) AND ARROWS (EDGES)
THIS MAKES IT HIGHLY GENERALISABLE. THUS IT CAN BE USED FOR VARIOUS PURPOSES.
CONCEPT OF WOVEN STORIES IS VERY SIMPLE. AND IT SUPPORTS COLLABORATION.
THAT IS WHY IT IS EASY TO LEARN AND USE.

(d) The fourth section added.


DOCUMENTS ARE BUILT WITH BOXES (SECTIONS) AND ARROWS (EDGES)
THIS MAKES IT HIGHLY GENERALISABLE. THUS IT CAN BE USED FOR VARIOUS PURPOSES.
CONCEPT OF WOVEN STORIES IS VERY SIMPLE. AND IT SUPPORTS COLLABORATION.
THAT IS WHY IT IS EASY TO LEARN AND USE.

(e) This time only a new link has been introduced.


DOCUMENTS ARE BUILT WITH BOXES (SECTIONS) AND ARROWS (EDGES)
THIS MAKES IT HIGHLY GENERALISABLE. THUS IT CAN BE USED FOR VARIOUS PURPOSES.
CONCEPT OF WOVEN STORIES IS VERY SIMPLE. AND IT SUPPORTS COLLABORATION.
THAT IS WHY IT IS EASY TO LEARN AND USE.
DUE TO THIS WS IS SIMPLE BUT POWERFUL SOCIAL MINDTOOL.

(f) And again a new section was added.

Figure 3.6: Creation of a simple woven story. Note that in order to create new storylines or paths it is only necessary to add links like in picture e.

the thoughts of the other members of the group.

After the first section has been placed on the wall, the other members of the group can read it. Then the other members can start to write sections. They can continue the story, embellish the story with a new longer piece, or start a totally new episode. The relationships between the sections are represented with pieces of string that weave the story together.

The group continues the story creation as long as they want. The result of the work can be a rather complex graph with several storypaths. It is important to note that it might not be necessary to write text to create a new story. It could be enough just to add a new edge, like shown in Figure 3.6(e).

Modifications to the existing sections are prohibited unless they are made by the original author. This way the members of the group are forced to contribute their own text if they disagree with existing sections. The aim is that after the work has been finished, the final woven story represents the knowledge and viewpoints of all the members of the group.

3.2 Prototypes

The first implementation of Woven Stories was a web-based tool developed with Java, PL/SQL, HTML, and an Oracle 7 database. It used a WWW (World Wide Web) browser to facilitate co-authoring. The prototype was limited and it only allowed for the creation of tree-like structures out of story sections.

The next prototype was called Woven Stories 2 (WS2) and it took a different approach to the concept of Woven Stories than the first prototype. WS2 introduced a new structure display, on which the users could draw sections wherever they wanted. Also, the limitation to tree-like structures was removed, and graphs became possible structures. WS2 also had a chat tool that helped users to communicate during collaboration.

The implementation approach of WS2 differed from that of the first prototype. While the first prototype was a web-based application, WS2 was based on a client-server architecture, where both client and server were stand-alone applications. The whole package was written with the Java programming language and it used an open source HyperSQL database engine (HSQLDB).

During the following years, Woven Stories application went through a total transformation. The latest Java-based version is called Loom. This version has several new features compared to the earlier versions. The biggest difference is that the communication between the client and the server has been totally changed. Also it is easier to implement new features into the software due to new communication methods. Loom is presented in detail in Chapter 6.

A tool called Woven Strategies was developed concurrently with Loom. Woven Strategies is an extended version of Loom and was created in a project where the aim was to create an environment to support corporate strategy planning. More about this project and the tool is found in Section 7.1.4.

The final version of the Woven Stories application is a recently implemented web-based version of the Woven Stories concept. This version is called WS@Web. The client is written with JavaScript and the server uses PHP. The application uses the AJAX (Asynchronous JavaScript and XML) approach to transmit data between the client and the server. Due to the selected development methods, this version can be run on any modern web-browsers. Thus, the users do not need to install any specific software on their computer to start using the application. This application is light for the administrator, since its installation is straightforward. WS@Web is presented in Section 6.5.

Table 3.2 briefly summarises the properties of the different prototype implementations.

Table 3.2: Prototypes

Prototype	*Year*	*Developer(s)*	*Design tools*	*Example Applications*	*Environment*
WS1	1999	Hassinen & Harviainen	Java, PL/SQL, HTML, Oracle 7	Writing	Internet
WS2	2001	Gerdt	Java, HSQLDB	Writing	Any
Loom	2003 2008	Nuutinen, Taivainen & Laine	Java, HSQLDB	Writing, planning, problem solving	Any Internet
Woven Strategies	2004 2005	Nuutinen, Taivainen & Laine	Java, Oracle 9	Planning corporate strategy	Any Internet
WS@Web	2008 2009	Nuutinen	PHP JavaScript MySQL	Writing, planning, problem solving	Web-based

Chapter 4

Towards Social Mindtools

"If you cannot build a model of what you are studying, then you do not understand what you are studying."
– David H. Jonassen [69]

This chapter provides the definition of Social Mindtools and discusses the related theories from fields of education and computer science. The issues are covered in such a detail that it becomes clear what social mindtools are, how they help in the learning process and what is important in the development process of these tools.

Figure 4.1 presents the relations between learning, mindtools, and computer supported collaborative work (CSCW) in relation to Social Mindtools (grey area). While Woven Stories and social mindtools are in general primarily located in the grey area, they also make contributions to the surrounding areas. Due to this it is important to review these areas in order to be able to design and implement a good application based on the concept of the Woven Stories.

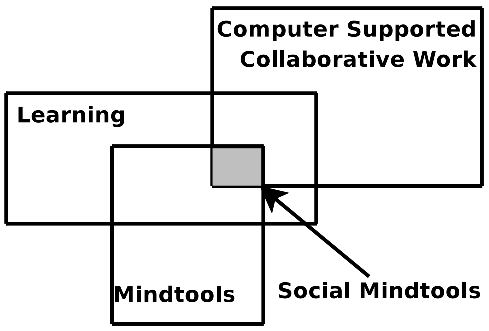


Figure 4.1: Position of Social Mindtools in the research areas.

The main references cited in this chapter are written by recognised authors. For mindtools the main source has been work of Jonassen. For learning theories the work of Ausubel and Bereiter has been studied. In the context of collaboration, the extended activity theory of Engeström has been used. From the field of computer supported collaborative work authors such as Grudin and Gutwin are cited. Some of the references used might seem rather old to the reader, but it has to be kept in mind that even though the technologies develop quickly, the base still remains the same. And, it seems that most of the research cites these same articles. The environments have changed, e.g. to Web 2.0, but the fundamental issues behind collaboration are still the same.

The relationships between different parts of the chapter are represented in Figure 4.2. First section introduces and defines mindtools. Since mindtools are used for learning, and especially for knowledge building, these are covered next with the introduction to learning and writing, which is important in the context of Woven Stories. After, knowledge related literature is covered, before continuing towards collaboration and to computer supported collaborative work. Finally, by utilising all the above, social mindtools are defined and a framework for these tools is given.

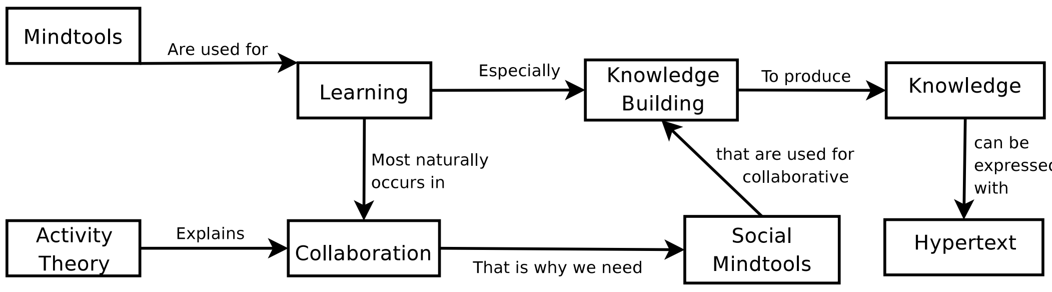


Figure 4.2: Relation of the Sections of this Chapter to each other.

4.1 Mindtools

Mindtools are tools that can be used for different purposes and in several domains. The main idea is that these tools facilitate the processes of constructing knowledge by learners. They are "knowledge representation tools that function as intellectual partners of learners" [69]. They are knowledge construction tools—tools that extend the mind [67] and force the user to think. In other words, a mindtool is simply a device, or technique, for focusing the learner's analytical processes [82].

Based on statements of Jonassen [67] and Mayes [82], mindtools can be defined as "generalisable knowledge construction devices or techniques that help learners to focus their analytical processes". This definition is strongly influenced by the fact that mindtools are mostly used for learning. However, it should be noted that these tools can be used for other purposes as well and are not tightly bound to education. This definition of mindtools is problematic though. It is hard to define what mindtools actually are and what kinds of features these tools should have. Jonassen [68, pp. 18] presented a list of features that a tool or application should have in order for it to qualify as a mindtool. These features are presented in Table 4.1 in the column *necessary features of a mindtool.*

While working with mindtools the learner must be able to construct new knowledge in several domains. Examples of mindtools include concept maps (see e.g. [93, p. 15]), spreadsheets and hypermedia (i.e. constructing a web site) [69]. A tool that enables the learner to construct new mathematical knowledge does not meet the requirement of generalisation and cannot thus be considered to be a mindtool. The importance of critical thinking cannot be underestimated. Learning requires processing and critical thinking is processing at its best. The ease of usage of mindtools is also important. Since schools are often under-equipped with computers, it is important that students are able to use the tools after a short introduction. At the same time, students should be able to get as much benefit as possible out of the mindtool.

While mindtools are mostly meant for processing information and thus learning the relevant parts of the content processed, they may also be used for the evaluation and assessment of students. Even though the teachers might find this approach useful, it should be noted that these tools support processing information and knowledge. This means that the actual outcome might hide the most important aspects of what students have learnt. Furthermore, during the use of these tools, the learners will most likely learn new things. Therefore these tools should be developed and constructed in ways that support the process rather than just emphasise the outcome.

The features of mindtools presented in this section have been compiled in Table 4.1 in the column labelled Characteristic. Jonassen's elaboration of these characteristics is contained in the column labelled Necessary features of a mindtool according to Jonassen [68]. A short description is provided in the column labelled Remark. It should be noted that Jonassen's input [68, p. 18] implies that all mindtools are computer applications.

Table 4.1 suggests the following four characteristics of mindtools: accessibility, engagement, multi-purpose utility, and usability. Accessibility means the extent to which the tool is available to its users and the cost of using the tool. Usage should ideally be free. If it is not free, fees for usage should be as low as possible. When mindtools are distributed as freeware, they are often able to benefit far more educational institutions than expensively priced software. Engagement means that

Table 4.1: Characteristics of a mindtool

Characteristic	**Necessary features of a mindtool**	**Remark**
Accessibility	• Application is available • Application is affordable	Funds limit what educational institutions ca do. It is therefore important that mindtools are easily available and affordable.
Engagement	• Intended for knowledge construction • Supports critical thinking	These tools assist learners to think and construct knowledge on the basis of their previous knowledge and experience.
Multi-purpose utility	• Generalisable • Transferable to other forms of learning	When students master just one good tool, they can use it for all of their subjects.
Usability	• Based on a simple, powerful formalism • Easy to learn	When students find it easy to master a mindtool, the spin-off is that they will gain at least some acquaintance with technology in education. A good mindtool helps users to focus on the subject rather than on the tool itself.

a tool should be designed in such a way that it serves the purposes of knowledge construction and supports critical thinking. Multi-purpose utility signifies that an efficient mindtool needs to be generalisable. This means that it can be applied in several application areas and a number of subject domains. Tools of this kind should also facilitate the transferability of whatever skills have been learnt. Usability intends that it is easy to master the use of a particular tool and to apply it in practise. This criterion takes into account not only the formalism or the concept of a tool, but also its technical quality. The interfaces of the best mindtools have been carefully and thoughtfully conceptualised, designed and developed.

Mindtools are useful because the skills that are used in their application are easily transferable from one subject domain to another and from formal to informal learning and vice versa. The use of mindtools promotes the integration of technology and education when little time is available to master the navigation of a technology and to apply its benefits to new subject areas.

4.2 Mediation of Learning

Learning is effective when it is meaningful. By providing learners with meaningful, motivating tasks better outcomes can be achieved. Thus, the motivation is the driving force of the learning process.

Learning is mediated by thinking [67]. According to Jonassen [67], "thinking is activated by learning activities and learning activities are mediated by instructional interventions". This means that in order to learn something, a person must think and understand the information being dealt with. Thinking processes the information we possess into knowledge.

Finally, in the context of Woven Stories writing has an important role. Writing can be a valuable learning tool, since it forces learners to process the knowledge they possess into explicit form.

The following Subsections cover these concepts.

4.2.1 Knowledge Building

Knowledge building is, according to Bereiter, doing something to a conceptual artefact [14, p. 255]. Bereiter [14, p. 58] states that "conceptual artefacts are human constructions like other artefacts, except that they are immaterial, and instead of serving purposes such as cutting, lifting, and inscribing, they serve purposes such as explaining and predicting". Thus the actual purpose of knowledge building is to build theories or explanations instead of presentations or videos. Similarly, in model building, the learner must find out what elements fit together [69] and make certain choices, in which, according to Jonassen [69], learning process lies. Working with Woven Stories is an example of such a process since the aim is to create a meaningful story instead of a presentation. Learners should be able to create something that is usable for them. As long as learners understand that the artefact they are creating is going to be useful, it keeps them internally motivated.

It is important that the product of knowledge building is indeed a conceptual artefact. Otherwise the actual process might focus too much on the physical features of the artefact, not on the actual process (see e.g. [102]). Thus the artefact produced should not be a poster or a movie [14, p. 294]. Still, the produced conceptual artefacts should be authentic to the extent that they are things the learners can actually use [14, p. 294]. Hence the conceptual artefact can be used to create presentations or posters afterwards. During the creation of the artefact it is important that the learners concentrate only on contents, relationships between the content items and the concept with which they are dealing with. This is the idea of the Woven Stories. First the structure and content of the document are created, and after document is finished, it can be utilised elsewhere.

Bereiter emphasised that knowledge building should be distinguished from learning; these are two separate concepts. Learning is doing something to alter the state of one's mind to achieve a gain in personal knowledge or competence [14, p. 255]. Learning happens trough processes of *internalisation* and *externalisation* [37]. In context of Woven Stories externalisation is the process, where individual transforms the knowledge possessed in to form that can be written to a section or presented as a storyline. Internalisation is opposite to externalisation. During a story weaving process a learner reads others' contributions. This new information is then processed and integrated into individual knowledge. Trough internalisation individuals develop new knowledge [37]. Collaborative knowledge building then represents collective advancement of knowledge [56].

Through internalisation it is also possible to reach *meaningful learning.* Ausubel [7, p. 27] defined meaningful learning as follows:

> meaningful learning takes place if the learning task can be related in nonarbitrary, substantive fashion to what the learner already knows, and if the learner adopts a corresponding learning set to do so.

In other words: if the learner can relate the new knowledge with something that is already known, then meaningful learning occurs. *Rote learning* is the opposite of meaningful learning; the learner tries to internalise something that cannot be related to any existing knowledge. One of the most important things in Ausubel's theory is that what is to be learnt should be based on the facts that are already known. Better results can be achieved by making learning meaningful. One way of achieving this is by utilising discovery learning [25]. In discovery learning, learners follow the same procedure as scientists [134]. They generate hypotheses, they set up experiments and tests, and they interpret data. These activities are traditionally associated with empirical research. Discovery learning encourages learners to discover concepts for themselves rather than to have them presented [7, p.24].

The distinction between learning activities and knowledge building activities is not clear. One may start reading with a learning purpose in mind, then notice something significant that causes to shift into a knowledge building mode [14, p. 256].

According to Bereiter [14, p. 274]:

- People learn from what they process; and

- The skills most likely to be learnt are the minimal ones necessary to accomplish the range of tasks presented.

Working with Woven Stories forces the learners to process the data. The approach where the stories are presented in a visual form guides the learners to divide knowledge into meaningful parts. Knowledge building is also an indirect learning activity [14, p. 277]. Thus learning happens without even noticing it. For example, in context of Woven Stories a user might learn to divide the text written into meaningful parts. Furthermore, Cress et Kimmerle [37] state that contributing to an article in a wiki can lead to individual learning processes in the contributors. The mental effort needed to externalise the knowledge requires deep processing and can thus lead to extended knowledge [37].

The knowledge building process can be summarised in these two items.

1. Knowledge building aims at creating a conceptual artefact - for instance an explanation, a design, a historical account, or an interpretation of a literary work.
2. A conceptual artefact is not something in the minds of the students; neither material nor visible but nevertheless real and something students can use.

Figure 4.3 presents a perspective of knowledge building. Learner creates from a certain topic an artefact with mental processing. The artefact is a new representation of the subject matter, something that the learner can use in the future.

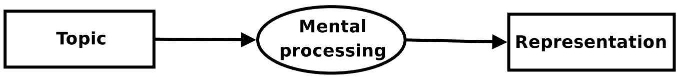


Figure 4.3: Knowledge building process is aimed at creating a representation. This representation is a product of mental processing.

This theory of knowledge-building led to the computer application called CSILE (see [112]). Nowadays CSILE is known as Knowledge-Forum [4].

4.2.2 Learning with Writing

In the context of Woven Stories, writing is an essential skill. Before discussing computerised writing applications, a distinction between writing with and without computers must be made. With a computer it is easier to correct mistakes and to write coherent text. Even though it sometimes seems that a computer is a more efficient tool to produce text than paper and pencil, there are also drawbacks to using computers for writing. Davies [38] did an experiment with third and fourth year pupils where they were supposed to write text with computers in pairs. The drawback with using computers was that the students did not do any pre-writing activities, that is, they did not plan their work, but just started typing; this might be explained by the fact that at the time of the experiment computers were rather rare.

In many cases writing can be seen as an efficient learning tool when certain conditions are fulfilled. For instance, the writing task should promote active knowledge building and the task should make use of writers' previous knowledge and existing concepts about the topic [131], thus combining meaningful learning and knowledge building. Furthermore, writing forces individuals to externalise their knowledge (see Section 4.2.1) and provides a source of information for others to internalise.

According to Hartley and Tynjälä [59], writing is typically divided into three overlapping stages.

1. *Planning and collecting* - thinking about the content of the text, its organisation and what materials are needed;

2. *Initial drafting leading to more final writing* - putting down ones thoughts on paper or on screen; and

3. *Revising and editing* - rethinking and re-planning the content, as well as correcting spelling errors, checking the page numbers of publications, and similar revision activities.

These stages can also be called pre-writing, writing, and rewriting [117]. As mentioned, these stages overlap and can even be done in any order. For example, a writer can iterate between planning and writing numerous times changing parts already written and introducing new parts. Despite the order of the stages, the process requires concentration and, above all, constant knowledge building and processing. Bereiter and Scardamalia call this advanced way of writing *knowledge transforming* [15]. Knowledge transforming is a writing process that includes much rethinking and restating and that creates fully developed thoughts. Another process of writing is called knowledge telling [15], which is used by children and less educated writers. In knowledge telling, the writer explains the facts that are known without any processing. Since people learn what they process, it can be concluded that writing can indeed be a powerful learning tool.

Woven Stories supports the process of writing at all the described stages. Planning can be done by introducing sections and their relations. This way it is possible to organise what is to be written; initial contents can also be added to sections. Revising and editing is also easy, since new sections can be introduced to include new contents or, the relations between the existing sections can be reorganised to revise the flow of the document. The ability to change the contents of the document without deleting previous content is important, because it prevents loss of possibly important knowledge.

Hartley and Tynjälä [59] discuss collaborative writing. They have listed some aspects that are typical to collaborative writing. They state that collaborative writing can be:

- *more efficient* - because different aspects of the task can be shared;
- *of better quality* - because different individuals can contribute different ideas and can contribute different types of expertise;
- *better thought out* - because each individual has to take into account the others' points of view;
- *faster* - because the less-able contributor is helped by the more-able; or
- *slower* - because the less-able contributor holds back the more-able ones.

Even though collaboration is never un-problematic, as can be seen even from the list above, according to Hartley and Tynjälä [59] it generally provides a good context for learning. In some occasions, collaborative writing can cause the result to be fragmented texts that can not be twined together. The use of computer supported technology offers interesting possibilities when assisting writing. Computer support may facilitate, or provide a vehicle for better writing and learning [59].

Woven Stories takes into account the aspects from the list above. It provides users with the possibility to share their workload and to contribute to their ideas and knowledge. It forces collaborators to process the text others have contributed before they are able to create links between the sections. Furthermore, users need to analyse the text they write and divide it into meaningful parts that others can use. The approach used to visualise the stories helps the less able contributors to see the relationships between different parts of the document, but at the same time, provides more able users with the possibility to continue their work without need to wait for less able users.

4.3 Knowledge

Section 4.2.1 discusses the knowledge building process in education. If there is a need to store, retrieve and manage knowledge in digital format, it should be known what knowledge includes. Wiig [137] defined knowledge as – "the insights, understandings, and practical know-how that we all possess" – the fundamental resource that allows us to function intelligently.

According to Sunassee and Sewry [125], knowledge is divided into two different categories; tacit and explicit. Sunassee and Sewry [125] define tacit knowledge as:

> Tacit knowledge is the form of knowledge that is subconsciously understood and applied, difficult to articulate, developed from direct experience and action and usually shared through highly interactive conversation, storytelling and shared experience.

Thus tacit knowledge is something that is difficult to share, something we are not aware of, or something we are not able to speak about. According to Sunassee and Sewry [125], "explicit knowledge is easy to articulate, capture and distribute in different formats, since it is formal and systematic". Thus, from the technical point of view, explicit knowledge is much more easily obtained. On the other hand, tacit knowledge might be something that is more beneficial.

Knowledge, whether practical or theoretical, is always a goal in an intentional learning process. *Intentional learning process* means the process whereby a learner is intentionally trying to learn, thus trying to achieve new knowledge. Another important thing to keep in mind is Bereiter's statement presented in Section 4.2.1 that people learn what they process. In Figure 4.3 I have referred to the artefact that is the product of knowledge building as representation. This representation is a new, learner-generated presentation of the topic that is to be learnt.

The representation described above is an explanation of what the learner has created. It does not even need to be in a physical format, it can just be in learners' mind. It is important that the learner has processed it into a form that can be reused if needed and that the learner can also create physical representation of this knowledge. These physical representations can include concept maps, mind maps [29] or other methods that are found useful by that person. If the learner has been able to create a representation of the topic it can be said that new knowledge has emerged in the learners' mind.

When the knowledge a person obtains is supposed to be shared with other people, the task is complicated. In order to create a representation that is understandable for others, there should be some agreed upon methods to represent that knowledge. One common, and maybe even the most often used, way to represent knowledge is through language. People are able to explain or write what they know. Another popular method is to visualise (see e.g. [122]) the knowledge possessed. The process of visualisation also requires common standards for representing the data or knowledge possessed. According to Spence [122], any method for representing knowledge is a cognitive activity.

Modern methods to represent knowledge often include computers. The knowledge a person possesses is often transformed into a digital form. The advantages of this approach are that digitally stored data are easy to share with others, easy to modify and easy to store. The only problem is finding or designing suitable software. For the developers of knowledge-storing software the problem is how to transform human knowledge into a form that computers can store and manage. This area of research is called knowledge representation [121].

Knowledge has three aspects that should be taken into consideration. The first aspect is *knowledge building* as described in section 4.2.1. The second aspect is *knowledge representation* and the third one is *knowledge management*. Figure 4.4 represents these stages of knowledge and their relations to each other.

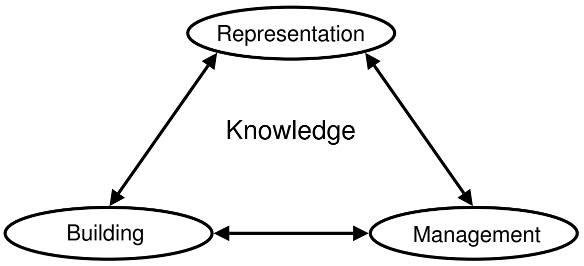


Figure 4.4: Processing knowledge has three major parts; knowledge *building*, *representation* and *management*. Similarly these areas are related to the sciences of education, cognitive psychology and technology.

The aspects of knowledge presented in Figure 4.4 can be correspondingly mapped to three fields of study: knowledge building is studied within the educational sciences; representations within the cognitive sciences and psychology; and knowledge management is studied within business and the technological sciences.

Knowledge management can be defined as *the process with which the knowledge possessed is stored, handled, distributed, and organised*. Given this definition, knowledge management is something that happens at all times. In many sources, knowledge management is defined for organisations (see e.g. [125]). Berztiss [20] states that one aim of knowledge management is to transform items of personal knowledge into institutionalised knowledge. Thus, transforming the knowledge possessed to the format that also others are able to use.

A system that supports knowledge management should be able to store, distribute, organise, and let users handle the stored knowledge. The primary aim of computerisation is to facilitate access to the knowledge that would help individuals in performing their duties [20]. In systems meant for education it could be said that learning is the learner's duty. Knowledge management process must contain an educational component [20] and vice versa, educational systems should contain a knowledge management component.

Figure 4.5 illustrates the roles of knowledge building, knowledge management, and knowledge representation from the viewpoint of a learning community. In order

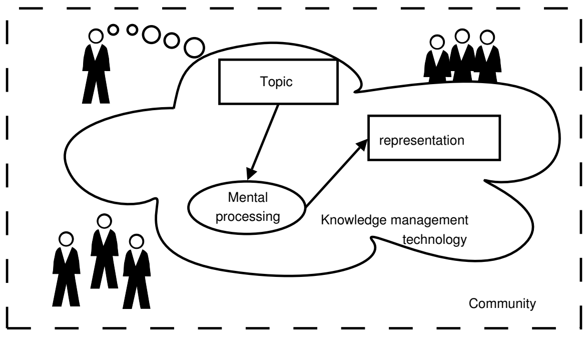


Figure 4.5: Knowledge management is an important task of a learning community; especially when the learning environment is computer-based and distributed over time and space. The knowledge should be well managed in order to be easily shared.

for the learners to employ, analyse and use the knowledge others possess, the knowledge should be well managed. When knowledge is properly managed, it is easily used by others and can be applied for several purposes. When all the members of the community can share their knowledge, these contributions can lead to situation, where other community members can also process and internalise that knowledge. These kinds of systems can promote the acquisition of tacit knowledge, which can be difficult in regular settings.

4.4 Hypertext

The original idea of hypertext was presented by Vannevar Bush [28, reprint] in 1945. He described a system where an individual stores all his books, records and communications [28] and where all this information could then be retrieved with exceeding speed and linked together, forming trails that could be later followed. Words *hypertext* and *hypermedia* were coined by Ted Nelson [91] in 1965. With hypertext, Nelson meant a body of a written or pictorial material interconnected in such a complex way that it could not conveniently be presented or represented on paper [91]. The first implementation that included hypertext linking features was On-Line System (NLS) [44].

The success and familiarity of hypertext and hypermedia started to gain general popularity after the introduction of World Wide Web (WWW) [17, 16]. WWW made hypertext easily accessible for everyone. However, even before WWW hypertext was a popular target of research. In an article published in September 1987, Conklin [34] surveyed 18 different hypertext systems. By that time hypertext systems were versatile software for easy authoring, annotating and interlinking information. These systems were mostly meant for individual users to organise and manage their information.

On top of what nowadays is understood as a hypertext link, an HTML A-tag which links the current document to another document, the first hypertext systems had a richer typology of links. *Bi-directional links* offered users with ways to know which other documents refer to the current document. Most hypertext systems with bi-directional links offered a local map, showing nodes and connecting links [136]. *Typed links* provided the author of hypertext with a possibility to describe the relationship between the source and destination [136]. For example Sepia [123] shows the link type as a label of the link in the local map. Links with *multiple endpoints* were also used in many hypertext systems. These links do not connect only two, but a set of related nodes [136]. When a user followed a multiple endpoint link, the system presented a list where the final destination was selected. A more recent addition is *adaptive linking* [30]. An adaptive linking system modifies the link structure of the document depending on users' knowledge state [30].

Already in the late 1980s researchers had identified problems with hypertext. Conklin [34] introduced the "disorientation problem" which was caused by the degree of freedom hypertext offered: the user could easily get "lost in (hyper)space". This problem was addressed by tools such as NoteCards [58], Storyspace [23], WE [119] and gIBIS [36, 35] by providing the user with a visual representation of structure of a hypertext. Even this approach, however, has its limitations; when the graph represented by the map enlarges, it can be difficult to the users to find relevant information from it.

The emerging of World Wide Web and HTML changed the research on hypertext. Researchers, although sometimes reluctantly (see e.g. [51, 118]), started to see WWW and HTML as object, and as a tool, of research. Whilst previous hypertext and hypermedia systems often included visualisation of the hypertext structures, the limitations of the HTML made this impossible. According to Schraefel et al. [113] the Web left behind many of interaction rich features of hypertext. Furthermore, the research on hypertext engaged in software efforts to engineer the Web into something more hypertext-like [113]. Other possibilities to guide users in the complex hypermedia structures were introduced, such as adaptive hypermedia (see e.g. [26], [39] and [40]) and spatial hypermedia (see e.g. [81]). Spatial hypermedia, however, still relied largely in the graphical representation of the relations.

4.4.1 Examples of Hypertext Implementations

The following four paragraphs briefly outline five different tools that are interesting from the perspective of Woven Stories. These tools share features with Woven Stories, but are generally based on more complex formalism. Three of these tools, NoteCards [58], Storyspace [23] and WE [119] were meant for individual users, whilst gIBIS [35] and Sepia [123] were collaborative tools.

NoteCards [58] provided its users with electronic notecards that could be interconnected by typed links. Users created virtual notecards that could include text, graphics, images and other editable data [58]. By introducing the links between the cards the users were able to connect the cards and thus form the graph. NoteCards also included a browser where users were able to traverse the graph of cards in a structured way, similar to that of Woven Stories. NoteCards was designed to support people to work with their ideas [58] and was based on the notion that creative intellectual work is a hand-craft, a uniquely human skill that cannot easily be automated. Thus, NoteCards is a mindtool. The main limitation of NoteCards was the lack of support for collaborative work [58].

Bolter and Joyce [23] introduced a system called Storyspace. Storyspace was a tool for writing interactive fiction. The approach used to represent and construct the stories in Storyspace was similar to that of Woven Stories. Each part of the story is represented as a node of a graph and the connections between the parts are represented with links. Storyspace does not provide any collaborative features. For reading the stories it had another user interface. Storyspace has been developed since, and is still being sold by Eastgate System Inc. Storyspace allows links to be inside the boxes as well, thus hiding some of the structure from users [70].

WE [119] is based on three phases of authoring; exploration, organisation and writing [119]. For each phase, WE has a special mode. For exploration, WE offers a network view, where user adds and edits ideas and notes as nodes of a graph. This mode is meant for "retrieving potential concepts from long-term memory and/or external sources, representing these concepts in tangible form, clustering them into related groups, defining special relations or associations between pairs of concepts, and constructing small hierarchical structures“ [119]. This mode is also meant to be used when writing hypertext documents. This idea is similar to that of Woven Stories. WE offers also the tree mode, which is meant for supporting the organisation of the text. In the tree mode, the user is constrained to the tree data structure. Thus, the user cannot move the nodes of the tree, nor create such links that would transform the tree into a graph. The idea is that the user organises the concepts to *a single, integrated hierarchical structure* [119]. Smith et al. [119] state that organisation is a process of conscious, deliberate construction. Actual writing is supported with editor mode, which allows the user to add contents to the added concepts that is, fill the nodes of the graph or a tree with text. Finally, text mode can be used to represent the document constructed in the tree mode as a linear document. Text mode traverses the tree top-down, left-to-right and produces a continuous document.

Conklin and Begeman introduced gIBIS [35, 36], a discussion and argumentation tool based on the Issues Based Information Systems (IBIS) method. In gIBIS the users can post nodes of three types (issues, positions and arguments), and connect these nodes with nine types of links, thus creating complex issues based graphs. This graph is visualised in the browser, and users are able to place the nodes according to their needs. This approach, according to Conklin and Begeman [36], has several advantages, including that the effort of coming up with a layout reveals aspects of their problem that are not obvious beforehand [36]. In general, Conklin and Begeman [36] report that the structure of gIBIS on discussions was very useful. The gIBIS system was used for various tasks and discussions. As a collaborative discussion tool, it can be said that gIBIS was among the first social mindtools (see Section 4.7).

In 1992 Streitz et al. introduced Sepia [123], a cooperative hypermedia authoring environment. Based on their analysis of cognitive processes they characterised writing as a design activity [123]. Based on this characterisation, they used the following three main features of every design process as their development guidelines [123]:

- design is a complex problem solving process,
- design is the construction of an artifact, and
- design is usually a social process.

In terms of cognitive modelling [123], Sepia was close to WE. However, whilst the final product of WE was a linear document, Sepia products are real hyperdocuments that is, documents that have hyperlinks. Sepia introduced a new working model, where authors could sometimes work individually and then, if needed, shift to close collaboration with other users. The user interface relied on the graphic layout of graphs, and it included features for providing users with awareness related information. Based on the facts that Sepia could boost collaborative cognitive processing, it indeed was a social mindtool.

4.4.2 Hypertext and Woven Stories

Before the WWW the hypertext systems were complex applications, for hypertext production, annotation and knowledge organisation. These tools offered several ways to interlink documents and organise information. These tools typically had several types of links, and some even had methods to type the hypertext nodes. A common feature was a graphical browser, in which the structure of the hypertext nodes and links was visualised as a graph. WWW made the hypertext more accessible, but changed it from a read-write system to mainly a read-only system [113]; only the author of the document can change the links and contents of the document. Instead of being seen as a way to organise and author information, hypertext is seen as an effective way to publish information. Currently, the WWW and hypertext are perhaps shifting back to what was seen as hypertext in the 1980s. There is more interaction, possibilities to author and change contents of web pages and even annotate them. The problem is, however, that all these are achieved with separate tools; none of them are standard features of WWW or HTML.

Even with its limitations, WWW with HTML and other implementation methods is an excellent infrastructure as Smith stated in 1997 [118]. It provides a publishing medium for documents and texts, but also acts as a platform for various applications. It provides easy access to information and applications, almost wherever, without the need to install specific applications to users' computers. What started as a perhaps shorthanded implementation of a hypertext system has evolved to an important infrastructure of the modern information society.

Woven Stories is a concept that would not exist without WWW or the previous hypertext research. It has similarities to several earlier hypertext tools and its communication strongly relies on the HTTP protocol, the base of WWW. The relation to some earlier hypertext applications and WWW to Woven Stories is presented in Figure 4.6. In the context of Woven Stories the definition of hypertext is simple; sections of text are interlinked with directed links. This is close to Halasz's [57] narrow definition of hypermedia as information representation and management system that organises information into networks of multimedia nodes interconnected by links. In the context of stories and narratives, the use of directed links between nodes is justified. According to Bolter et al. [23] in its simplest form, interactive fiction requires only two types of elements: episodes and decision points (links) between episodes. A similar approach is also used by Storyspace [18]. This approach makes it easy for the reader to follow the plot of the story and, on the other hand, for the writer to plan the plot. However, the possibility to freely name the links in the Woven Stories can, in a loose sense, be interpreted as a possibility to type the links if needed.

Among the users of Storyspace, the storyspace map has been the preferred interface for most users [18]. Whilst Storyspace allows a writer to produce graphs, another similar implementation, Fluid Writer [138], relies on trees. However, the graph approach allows the authors to produce more flexible documents, and this is the reason why it has been selected as the basis of Woven Stories as well. Due to the simplicity of the selected linking typology, the structure of the document is easy to visualise. Furthermore, easy linking typology and visualisation of the structure helps keeping the user interface light and maintains the cognitive load of both the writers and the readers as low as possible.

The concept of Woven Stories has similarities to several earlier hypertext tools. What makes it different from many other tools is the possibility to collaborate asynchronously and synchronously within the same document. Furthermore, the possibility to integrate information retrieval methods (see Section 6.6) can help to

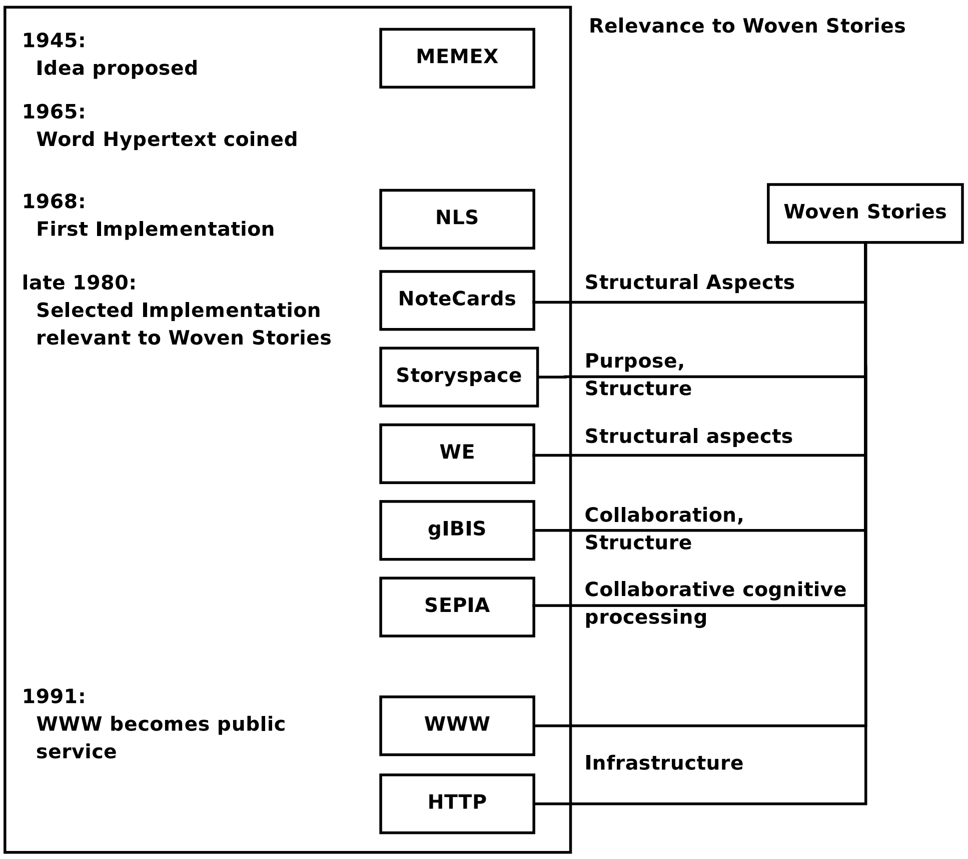


Figure 4.6: Relation between several earlier hypertext applications, WWW and Woven Stories. Selected events from hypertext history are presented left and their relevance to Woven Stories are represented with connecting links.

overcome the problem of the authors getting lost in the structure. The fact that Woven Stories relies on a simple formalism makes it applicable for several areas. The latter has been also the case with Storyspace; Bernstein [18] reports unexpected applications of Storyspace.

4.5 Towards Collaboration

The internet has been a catalyst for communities that are often called virtual communities [83]. In particular, the Web 2.0 (see e.g. [108]) phenomenon has inspired vast numbers of new, active communities. These communities consist of people who seldom or maybe never see each other in real life. Their only contact might be based on conversations in a web-based forum, a chatroom or a newsgroup. A virtual community commonly emerges around people with a common hobby or a common interest. It seems that if the participants join the community on a voluntary basis, the community is much more active than if the participants were forced to be a part of the community. However, even such communities have problems getting members to contribute [13].

In several cases these communities act as learning communities, even though the members might not even realise it. According to Jonassen [65], learning most naturally occurs in teams of people working together. For example, an aquarium enthusiast community shares information about fishes, aquarium techniques, and other aquarium related topics. An experienced hobbyist engages in vivid conversations and others learn from what the experienced hobbyist writes. Due to these kinds of activities, many on-line communities can be referred to as learning communities.

Since mindtools, and social mindtools, can be used for other purposes than learning, I refer to these types of communities as *Mind Communities.*

4.5.1 Mind Communities

According to Hoadley and Kilner [62] knowledge is generated and shared when there is purposeful conversation around content in context. They introduced a model, C4P, consisting of five elements: content, conversation, connections, context, and purpose. Hoadley and Kilner argued that the greater these elements are presented in any community, the more likely and effective the knowledge generation and transfer will be. They also claimed that an increase in any of these elements will result in an increase in all of elements.

The C4P model includes aspects for a successful knowledge building community. It emphasises purpose, motivation, and includes the aspects of good collaboration. However, it leaves out a vital part of such communities — knowledge building. Collaborative knowledge building cannot exist without conversation or communication. The collaborators need to share information. Conversation can be a good knowledge building tool, but more specific tools are needed to foster efficient knowledge building.

In order to extend and support the knowledge building activities of communities, access to specific tools should be organised. Communities that have access to collaborative tools, communication tools, mindtools, and social mindtools–and thus are able to build, represent, and manage knowledge–could be called *Mind Communities.* These communities are mainly meant for creating and processing knowledge in order to benefit its members. I define mind communities as follows: *Mind communities are groups of people whose common aim and interest is to create and process common knowledge with appropriate tools.* A community based on a web-based forum is not a mind community since the community does not have access to any mindtools even though they are using a collaborative tool (the forum). A group of learners that has a similar forum and access to a collaborative concept mapping tool is a mind community, since the group is also able to work with a mindtool.

Meaningful conversation is fostered by quality content, a clear purpose, and personal connections [62]. Here it should be noted that purpose is a key to collaboration. The group needs to share a common goal—the motivation to collaborate. In mind communities, the content is introduced, created, and polished by the community. In order to do this, communication, processing, and representations are used. Figure 4.7 presents a model of mind communities. Purpose brings the people together and acts as a motivating factor for the members of the group. Context sets the guidelines for the work and provides metadata for the provided content. The community is developed and maintained through communication, processing, and representations.

A mind community's members' goal is to get or share new knowledge. In order to achieve this goal, the members of the community must have the motivation to build collective knowledge and to contribute to existing knowledge. With such members the community can achieve a common level of knowledge and gradually expand the knowledge of its members. With tools that support knowledge building (collaborative tools, mindtools) and knowledge representation (mindtools), these goals can be achieved more easily. Gradually the members of the community also achieve the skill to represent the knowledge they possess in a way that other members can also understand and grasp.

4.5.2 Activity Theory

In order to enhance mindtools to support collaborative knowledge construction, it is important to analyse the factors that affect the users, or learners, while they

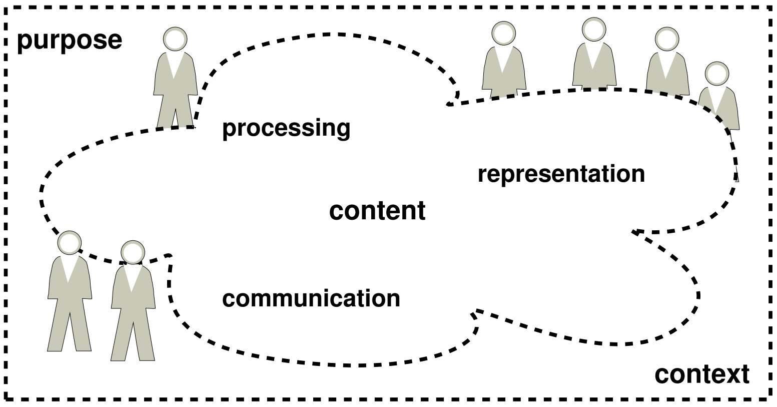


Figure 4.7: Mind communities are knowledge building communities equipped with collaborative mindtools.

are using the tools. In order to explain the factors, activity theory is used as a conceptual framework. Activity theory has been selected since it quite elegantly illustrates important aspects of collaborative activities and since it is applicable for software design as reported e.g. in [33].

Several researchers [22, 73, 77, 90] have proposed the use of the activity theory as a conceptual framework for understanding mediated work practise. Activity theory has its foundations in Soviet psychology [45]. Activity theory states that human mind develops, exists and can only be understood in the context of meaningful, goal-oriented interactions between the human beings and the environment in which these actions take place [72]. Kaptelinin and Nardi [73] observe that in the activity theory people act with tools and that tools are both designed and used in the context of intentional actions. People act as subjects, constructing and instantiating their intentions and desires as objects. In the activity theory this relationship between people and tools is seen as one of mediation. The tools are used to build objects, which do not need to be physical. Thus, the activity theory explains the process of knowledge building (see Section 4.2.1). In Engeström's extended activity theory model [45] he conceived the activity system as presented in Figure 4.8.

In Engeström's model the relationship between the object and the community is mediated by the division of labour and the relationship between the subject and the community is mediated by rules. The tool is used in the transformation process while the rules are the explicit and implicit norms, rules, conventions and social relations within a community. The division of labour is the explicit and implicit organisation of a community [111].

Activity theory posits that conscious learning emerges from action [66]. Activity theory is apparently not the only model to frame the Woven Stories and collabo-

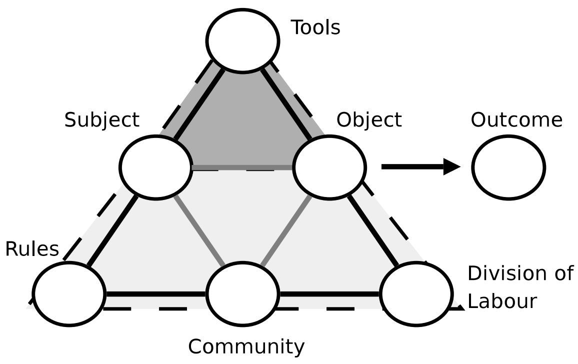


Figure 4.8: The extended activity theory model [45].

ration in general, but it clearly posits all the factors that affect an individual and community in any activity. Learning using a computer application and knowledge building activities can be explained from the view of activity theory. Thus it is vital to design and implement such software that can support the activities of learners. Mindtools are one such possibility since these are meant for supporting the thinking activities of the learners.

In the context of activity theory mindtools cover the upper part (darker grey) of the triangle in Figure 4.8 by providing a means for an individual learner to process information in order to achieve an objective. However, as a conceptual framework the activity theory points out the importance of the surrounding community. This is especially true in learning since, as Jonassen states [65], learning most naturally occurs within teams of people working together. When individuals engage in learning activities together, they form a learning community. In order to support the learning communities with knowledge building tools there is a need to extend mindtools with features to enable, support and engage learners in collaboration.

4.6 Computer Supported Collaborative Work

According to Jonathan Grudin [53], the concept of Computer Supported Collaborative Work (CSCW[1]) was first introduced in 1984, when Paul Cashman and Iren Greif organised a workshop devoted to CSCW. Since then CSCW has interested researchers and new areas of research has been formed around CSCW (see e.g. [76]).

What is computer supported collaborative work? Or which applications can be classified under the common term groupware? Grudin states that no formulation of defining groupware will satisfy everyone engaged in CSCW research and devel-

[1]Also referred as *Computer Supported Cooperative Work.*

opment [53]. Ellis et al. [43] have defined groupware to be computer-based systems that support groups of people engaged in a common task or goal and that provide an interface to a shared environment.

The biggest difference between individual and collaborative work is communication; effective collaboration demands that people share information [43]. In terms of activity theory, communication is used to determine division of labour and rules of the community. Communication has different levels that depend on the availability of the collaborators. Based on spatial location and time it is possible to create a time-space taxonomy of computer supported collaborative systems. The time-space taxonomy presented in Figure 4.9 was originally created by Ellis et al. [43].

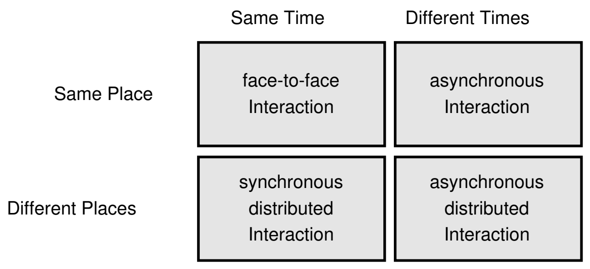


Figure 4.9: The time-space taxonomy for collaboration by Ellis et al. [43].

The taxonomy presented in Figure 4.9 can be used to classify different levels of communication, and it unquestionably points out that communication is an important part of collaboration. It is obvious that a comprehensive groupware system might best serve the needs of all of the quadrants [43], thus serving individuals and groups alike, as well as synchronous and asynchronous collaboration.

Synchronous collaboration means that the collaborators act and are present in the system at the same time. *Asynchronous collaboration* means that collaborators do not act nor are necessarily present in the system at the same time. An example of a synchronous system is a chatroom where two or more persons have a conversation together. When used by one person alone, the chatroom is less useful for collaboration. An example of an asynchronous system is a file repository, where all the users add their work. The users are not able to edit the files at the same time, but they can see afterwards what other members of the group have done. In the context of collaborative writing, synchronous collaboration is essential; especially during brainstorming and outlining when asynchronous collaboration is vital to writing, editing and reviewing [9].

Although synchronous and asynchronous aspects are important in collaborative environments, there is also a third aspect in the taxonomy. This class is called *semi-synchronous*. The semi-synchronous system combines the aspects of synchronous and asynchronous collaboration. Users can collaborate in real-time or collaborate with a delay, thus actually combining all the cells of Figure 4.9.

A semi-synchronous system can present current information on synchronously co-present collaborators, and can present information about the past activities of other collaborators who were not simultaneously present [42]. This means that regardless of the way users access information, they always should be aware what others are doing or what they have done.

In order to work successfully, computer supported collaborative work has a number of special requirements. These requirements, collected by Bannon and Schmidt [11] are presented in Table 4.2 and are compared to the activity theory and characteristics of mindtools

Table 4.2: Requirements of CSCW positioned with activity theory and characteristics of mindtools.

Requirement	**Activity Theory**	**Characteristic**
Articulating cooperative work	Community	Engagement
Sharing an information space	Tools	Usability
Adapting the technology to the organisation, and vice versa	Rules Division of labour Community	Access Engagement Multi-purpose Usability

The requirements presented in Table 4.2 support the contribution of the activity theory (see Figure 4.8) and the characteristics of mindtools (see Table 4.1). They provide additional requirements for each aspect of activity theory.

The time-space taxonomy of Ellis et al. presented in Figure 4.9 hints at the concept of computer supported collaborative work, but it is not clear how the actions that users are going to take relate to this taxonomy. Thus, there is a need to integrate the functionality of application to the time-space taxonomy. In their article [120], Sohlenkamp and Chwelos integrated the two dimensions of synchronous and asynchronous work and produced two tables. The combination of these two tables is shown in Table 4.3.

The integration by Sohlenkamp and Chwelos provides a good viewpoint on the requirements for computer supported collaborative work. It gives an overview of what features a collaboration tool should have. Furthermore, it addresses both synchronous and asynchronous modes, thus defining the requirements of a semi-synchronous system.

Table 4.3: The two dimensions of computer supported collaborative work integrated. The table includes the two dimensions and the design goals for each of the classes. Combined table adapted from Sohlenkamp & Chwelos [120].

	Synchronous	**Asynchronous**
Communication	*Communicate in real time:* • make and break verbal and visual contact with one or more other people • form and dissolve private conversations while in larger conference • quickly suspend all audio • control access to communication spaces	Leave messages for others: • leave notes for others wherever they are needed
Cooperation	*Simultaneous work using groupware tools.* • manipulate (create,edit,etc.) shared artefacts • create, join or leave synchronous cooperation sessions • select the appropriate degree of coupling for synchronous sessions	*Turn-taking work* • coordinate turn-taking work • determine what changes were made to shared artefact by others • merge divergent work on shared artefacts
Awareness	*What are others doing now?* • obtain some idea of what co-workers are doing • ascertain a co-worker's availability for contact • control their own level of availability • control the information about themselves which is broadcast to others • know when shared documents are in use by others • know exactly what others are doing during a shared editing session	*What have others done recently?* • determine when shared artefacts have been changed by others • determine how those artefacts have changed • determine when and where others have left messages for them
Others	• quickly join others in a shared work context (communication and collaboration activities) • suspend and later resume a complete work context • control access to shared artefacts • dynamically configure work groups to adapt to changing needs • customise their view of the shared environment	

While working in computerised environments many things are complicated, especially when it comes to collaboration. In a face-to-face meeting, people can see and use non-verbal communication and can see where others are looking and what they are doing. This understanding of other people's work is called *workspace awareness* [54] and is defined as *the affordances of physical workspaces that allow people to maintain awareness of others' locations, activities, and intentions relative to the task and to the space-awareness that enables them to work together more efficiently.* Briefly, it is the collection of up-to-the-minute knowledge a person uses to capture others interaction with the workspace [55].

In face-to-face situations this kind of information is directly available [75], but it is more difficult to maintain it in computer-mediated communication and groupware. In groupware systems, people only see a fraction of the workspace and they might not see other members of the group at all.

Typically the awareness related information can be analysed with lists of questions (see e.g. [55], [31]), such as "who are present?". By providing answers to these questions, the application takes care of the awareness related issues. In terms of activity theory (see Section 4.5.2) awareness provides the users information about the other members of the community. Awareness is useful when planning the division of labour [55] since these decisions depend in part on elements of workspace awareness [55].

Gutwin and Greenberg created a framework for workspace awareness [54]. The framework is shown in Table 4.4. By using the questions presented in Table 4.4, the developers of collaborative systems can check whether they have taken into consideration the relevant aspects of workspace awareness.

The framework presented in Table 4.4 has been extended by Gutwin et Greenberg and is published in [55]. However, in order to keep the social mindtools simple, the older framework is presented in this study.

4.6.1 Computer Supported Collaborative Writing

An important application of computer supported collaborative work is computer supported collaborative writing. Collaborative writing is highly popular. Some older research shows that a significant number of documents are produced in collaborative ways. Fish et al. [48] claimed in 1988 that 65% of documents in science are written collaboratively. Baecker et al. [8] claimed in 1994 that 85% of all documents written in business of science were written collaboratively. Although the cited research above is old, it is still reasonable to assume that the numbers are even higher nowadays, especially because of the increased utilisation of the World Wide Web and its applications.

Conceptually the process of collaborative writing seems simple. A group of

Table 4.4: Elements of workspace awareness according to Gutwin and Greenberg [54].

Element	**Relevant Questions**
Presence	Who is participating in the activity?
Location	Where are they working?
Activity Level	How active are they in the workspace?
Actions	What are they doing? What are their current activities and tasks?
Intentions	What will they do next? Where will they be?
Changes	What changes are they making, and where?
Objects	What objects are they using?
Extents	What can they see? How far can they reach?
Abilities	What can they do?
Sphere of Influence	Where can they make changes?
Expectations	What do they need me to do next?

people, sharing one common goal, produce a document together. When the group works in the same room, the process can, but not always will, be easy. According to Baecker et al. [8] physical proximity is often an important factor for successful cooperative writing.

In order to support collaborative writing with computers, there should be enough knowledge about the practises of collaborative writing. Kim et al. [74] carried out a study where 11 academics were interviewed about their collaborative writing practises. The most interesting results of this study included:

- Document management is centralised,
- The writing usually proceeds asynchronously,
- Small groups are common (usually only two persons),
- Commenting is problematic since comments from different authors might conflict,
- Annotating on a paper copy is common, and
- Email is the main communication and comment-sharing tool.

While the findings listed above might seem obvious, they offer much insight for a person implementing a computer–supported collaborative writing system. At

Table 4.5: Requirements of computer supported collaborative writing positioned with activity theory and characteristics of mindtools.

Requirement	**Activity Theory**	**Characteristic**
Shared document support	Tools	Usability
Conversation space support	Communication	Engagement
Coordination support	Rules Division of labour	Access Engagement Usability
Conference management support	Rules Division of labour	Access Engagement Multi-purpose Usability

least they show that even though there are already systems to support collaborative writing, these are not used; conventional methods are used instead. It seems that the tools used are basically digital versions of the traditional paper, pen and mail. Most likely the writing proceeds asynchronously because the tools function asynchronously. If the tools encouraged the users to do synchronous work, this might happen more often.

There are aspects one should consider when transforming collaborative writing to computer environment. Hofte [127] gives a list of service features that are especially needed in computer supported collaborative writing. His list is more developed than Bannon and Schmidt's list on computer supported collaborative work presented in Table 4.2. Hofte's list takes a deeper view, especially concerning collaborative writing. These needed features are presented in Table 4.5.

Hofte's list is a collection of different services needed in a computer–supported collaborative writing tool. *Shared document support* concerns users' interaction with a shared document. *Conversation space support* covers tools for direct communication. *Coordination support* concerns users' rights and responsibilities and *conference management support* includes users' ability to manage the conferences they are attending to (e.g. coming together to write a document is a conference action).

Gerdt [50] did an extensive survey on computer assisted collaborative writing in his master's thesis. Based on Hofte's [127] and Sohlenkamps & Chwelos' [120] work he constructed a four-item list for the requirements of collaborative writing.

1. Shared document,
2. Coordination,
3. Communication, and
4. Awareness.

Even though computers have proved to increase efficiency and ease communication, working with or through computers is not the same as working in face-to-face situations. Take the example of a person reading a long scientific article. It is most probable that the person will print it out first and then read it while making annotations on the paper with a pen rather than using a piece of software. It is easier to take a glance at the paper and outline the article than to do the same on the computer. However, tools such as Kindle [12] are improving this situation.

4.6.2 Idea Generation

Shaw et al. [115] stated that "Generating divergent ideas, not just routine notions is an important part of the problem solving process, perhaps the most important, since it is so difficult and unpredictable." While idea generation is crucial for problem solving, it is important in other domains, such as writing. Coming up with new ideas, and especially contributing them to the group, is profitable. Discussion about ideas assists in developing them further and may lead to new ideas by other group members. Without new ideas significant benefits from collaboration can be lost.

Prante et al. [105] defined three requirements for CSCW tools that support idea generation and structuring:

1. prevention of turn-taking,
2. structuring the idea space, and
3. lack of process constraints.

In terms of idea generation, one important factor is time. The sooner one can contribute ideas the better. If the system is based on turns, ideas cannot be contributed immediately after they are conceived. On the other hand, there will not be much time to process the idea when writing it down. Prante et al. [105] found a dramatic decrease in performance when working in turns. Due to this fact, they suggest that the first requirement should be to allow synchronous work on a shared idea space that allows parallel input. Synchronous work would allow users to work at their own pace, thus easing idea generation.

Although brainstorming frequently occurs in face-to-face meetings, face-to-face brainstorming has some disadvantages. Consider a person contributing a new idea. That person usually is forced to constantly generate new ideas, thus bypassing the idea recently generated. This might lead to a situation where an individual misses the reason behind the idea that was contributed. Why did he contribute it? What was so interesting about it? While working in a computer–supported system, where individuals can take their own time to process their ideas deeper, it is possible to overcome these problems.

Structuring the idea space means that users can locate the ideas in the idea space as they see fit. This is similar to placing Post-It notes on the wall, for example; it is easy to change the spatial positions as wanted. Prante et al. [105] noticed that this feature seems to foster the group's creative performance.

Idea generation is never a very structured process. It does not have a strict policy how to proceed and it is seldom limited by time. Based on their observations, Prante et al. [105] suggested that an idea generation tool should not have any constraints when considering the actual idea generation process.

4.7 Social Mindtools

Currently, there are mindtools that serve individual processing needs, and collaborative systems that serve needs such as communication and data sharing. Even though effective collaboration demands that people share information [43] and communicate, there is still more that should and can be achieved. In order to facilitate a community with better capabilities for processing data and perform thinking activities and knowledge processing, there is a need for mindtools that can be used collaboratively. *Social mindtools* are such mindtools that enable the community to work efficiently, to process data and to undertake activities together. They are mindtools that are intended for collaborative use. Thus social mindtools can be defined as *generalisable and collaborative knowledge construction devices or techniques that help learners to focus their analytical processes in order to achieve a common objective.* This definition maintains the requirements of mindtools, but adds a new requirement for collaboration. Thus it can be said that the social mindtools are extended mindtools and thus a subset of the set of mindtools.

In a learning community the individual activity transforms to dialogue and collaboration. Community shares a common objective and it uses tools to mediate the work and to achieve the object. By using the tools the community develops common knowledge through activity. All members of the community are able to contribute and thus participate in common knowledge building. By gathering viewpoints and opinions from different individuals, social mindtools extends the benefits of mindtools that are meant for individual work. With a social mindtool the learner is able to explain and give reasons for the data submitted to the community, thus forcing the learner to think and process the knowledge possessed by the community. The transformation from mindtools to social mindtools is actually similar to the transformation from traditional web to Web 2.0 (see e.g. [108]). Instead of information delivery and individual work, the focus is now put on collaborative information processing. These tools represent great potential for the facilitation of collaborative learning and situated cognition [64]. There are tools that can be classified as social mindtools; a common example would be wikis (see e.g. [37]) that allow anyone to contribute their knowledge for common use. Generally, the aim of the Web 2.0 trend seems to be, more or less, to offer tools that provide users with the possibility to share knowledge and collaboratively construct new knowledge. These tools encourage participation and they are social and open [132]. However, many of these tools are meant for one specific purpose and thus cannot be utilised outside their original application area.

How the social mindtools support collaborative work is tool-specific, but there are requirements that can be drawn from the viewpoint of activity theory. Naturally the requirements of mindtools are important in the case of social mindtools as well, but on top of these they need to support collaboration. Furthermore, as discussed in Section 4.1 mindtools are meant primarily for information processing and creation. Whilst this is still important in context of social mindtools, these tools should also emphasise the presentation of information in such a way that collaborators can easily understand and internalise it. Table 4.6 concludes the supplementary requirements of social mindtools. Table 4.6 maintains the same structure as Table 4.1 and thus these two can be compared.

The mind community has to be able to construct a commonly available knowledge structure and to communicate. By introducing features for communication, the community is able to set the rules and organise the division of labour. However, in order to engage the community deeper into the activities, proper features supporting *awareness* (see Table 4.4) have to be introduced. This information is used to create a feeling of community for the community members. These awareness features have been found out to be important in our evaluations, for example in the debate case that is presented in Section 7.1.6.

By using the social mindtools it is possible to cover all the factors affecting individual work as part of a community in terms of activity theory. While mindtools cover the upper part of the extended activity theory model (see Fig. 4.8) social mindtools cover the whole model.

Table 4.6: Added characteristics of social mindtools.

Charasteristic	**Requirement**	**Comment**	**Activity Theory**
Access	Preferably web-based	Web-based applications are easy to access and the load of maintaining the software can be minimised for the end user.	Community
Engagement	Provides awareness information Provides means for communication	In order to communicate, the learners must first be aware of each other. Awareness also provides learners with information about what other learners have done. Communication is a natural requirement of collaboration.	Community and Rules
Multi-purpose		*No added features*	
Usability	Allows editing a common object	In order to efficiently collaborate, the learners must be able to edit a common object.	Community and Division of Labour.

4.8 Framework for Social Mindtools

Everything covered in this chapter has been related to social mindtools. In order to codify all the issues covered, this section describes a framework for such tools.

Social mindtools have to meet the requirements for being mindtools. Second, mindtools are computer-supported collaborative applications. In order to combine these features, a comprehensive analysis must be carried out. It is important to analyse whether the system should be synchronous, asynchronous, or semi-synchronous.

First I will consider the features required from mindtools. Table 4.1 combines the basic characteristics and necessary features of a mindtool. It should be noted that due to the generalisable nature of the mindtools, it is not possible to create a complete model. Table 4.7 adapts Table 4.1 into a form of questions that can be used for determining whether a tool is a mindtool or not.

After the application has been classified as a mindtool, the next step is to find out if it fulfils the requirements for being a computer-supported collaborative tool, that is, a social mindtool. There are several possibilities for the tool to meet these requirements. Both synchronous and asynchronous tools are collaborative tools, although they are meant to be used in rather different ways. In an ideal case the tool is usable in both of these scenarios, thus the tool is semi-synchronous. In order to determine which class the application belongs to, the time-space taxonomy by Ellis et al. [43], shown in Figure 4.9, can be applied.

While the time-space taxonomy can be used to determine whether the application is asynchronous or synchronous, it does not determine what functions and properties such tools should have. These features, however, are rather well defined in Table 4.3, by Sohlenkamp et Chwelos [120]. Table 4.3 takes into consideration all three major aspects of computer–supported collaborative work and their essential features. However, in order to make this list more useful for developers and evaluators, the contents of the table should be transferred into form of questions.

A metaphor for an asynchronous system could be a meeting room of a workgroup. On the room there is a table, where all the work happens. Members of the group can enter the room whenever they want but only one user is able to modify the contents of the table. Users can leave notes to other users on the table. If simultaneous users are present, they do not see each other, they just see if the contents of the table are currently being edited (that is, they are not able to edit it since someone else is doing it at the same time). On the table there is a logbook that explains what modifications have been done and by whom. Table 4.8 presents the questions that can be used in order to evaluate and design the features needed in asynchronous collaborative applications. Since users cannot edit the same artefacts at the same time, these kinds of systems are rather easy to develop. It should be noted that the features related to the concept of the mindtool are included in the aspect of

Table 4.7: Questions to determine features of a mindtool.

Characteristic	**Question**	**Remarks**
Access	Is application available?	Mindtools should be easily accessed and, preferably, should have several possible uses.
	Is application affordable?	Since money is an issue for most schools and institutions, it is important that they can afford the tools.
Engagement	Can it be used for knowledge construction?	Mindtools are knowledge construction tools. The tools should be able to construct or represent content or personal knowledge.
	Does it support critical thinking?	These tools should promote critical thinking. Critical thinking requires analytical processing and high-level comprehension.
Multi-purpose utility	Is it generalisable?	A tool designed just for one subject domain is not a mindtool. Mindtools should be usable in almost any domain.
	Can it promote transferable learning?	Using a mindtool should foster generalisable skills for the users (skills that can be used in various fields). Thus, these tools teach the user to "think better."
Usability	Does it use simple, yet powerful formalism?	The formalism embedded into mindtools should be a simple yet powerful way of thinking. This also could be a part of the cognitive features, but it has also much to do with usability.
	Is it easy to learn?	In order to be efficiently employed at schools and other institutions, mindtools should be easy to learn. The tools should have a very gentle learning curve.

collaboration.

Truly synchronous applications are more difficult to develop than asynchronous applications. There are several technical issues to solve: for example, how to cope with situations where two or more users want to edit the same part of the artefact at the same time. When compared to the metaphor of asynchronous work, the workroom, as a metaphor for synchronous work, would be open for all members of the group at all times. Users could enter the room, modify the artefact at the same time, and have conversations. If the dimensions of Sohlenkamp et al. [120] are strictly followed, users would not be able to leave messages to each other. The ability to leave messages to others should not be needed, since all the members would be simultaneously present. Table 4.9 presents the questions that should be used while developing or evaluating a synchronous collaborative application.

While Tables 4.8 and 4.9 can act as an aid while developing or assessing collaborative systems, it has to be kept in mind that most collaborative tools are actually semi-synchronous. This means that these tools share features from both of these tables.

The three Tables (4.7, 4.8 and 4.9) presented in this Section, in addition to the time-space taxonomy presented in Figure 4.9, can be used to determine the requirements for developing a social mindtool. This model can also be used in order to determine if an application meets the requirements of a social mindtool. An example of usage of this framework is given in Section 6.7 where the current implementation of the Loom is compared to this framework. Furthermore, this framework is used as a base for development of Loom presented in Chapters 5 and 6 and the Nucleus Model, the development and research approach for social mindtools, presented in Chapter 8.

4.9 Summary

The first research question of this thesis (see Q1, Table 2.1) asks what features characterise a social mindtool within the set of mindtools. Social mindtools are defined as *generalisable and collaborative knowledge construction devices or techniques that help learners to focus their analytical processes in order to achieve a common objective.* Thus, social mindtools are a subset of mindtools that support collaborative knowledge building. Due to this, the social mindtools set several specific requirements (see Table 4.6) compared to those of mindtools (see Table 4.1). The biggest difference is the built-in collaborational aspect. For example, there are extra requirements that relate to the need to support awareness (see Table 4.4) and communication.

Fundamentally mindtools are knowledge creation tools and do not concentrate

Table 4.8: While developing an asynchronous collaborative tool, these questions should be considered in order to properly support communication, cooperation and awareness. The questions presented in this table characterise the asynchronous applications.

Aspect	Question	Remark
Collaboration	1. Do the users have a shared working area?	In order to do efficient collaboration, the users should have a common working area. It is recommended that they would have a similar view of this area.
	2. Can users coordinate the turn taking work?	If the work is based on turns, the users need to be able to decide the order, to see whose turn it is and who is next.
	3. Can users identify the changes made to the shared artefact?	When users return to the work, it is important to know what has happened since the last session. It is crucial to know what tasks have been done and what should be done next. This is strongly related to awareness.
	4. Can users merge divergent work?	If users have done divergent work, or have divergent ideas, are they able to merge these? In most cases the objective of the work is to produce one result that can be agreed upon by others.
Awareness	1. Can users see when the shared artefact has been changed?	When working simultaneously it is important for users to know what is happening while they work. The ability to determine what changes were made to an artefact increases the awareness.
	2. Can users determine who has done the changes?	In collaboration it important to see who does what. This supports the feeling of a community and also makes it easier for the members of the group to divide the work.
	3. Can users determine how the shared artefact was changed?	It is not enough to just see that something has changed. It is more important to see how something was changed.
	4. Can users see when and where others have left messages for them?	This is strongly related to communication (see communication, question 1). Users should be aware of the fact that they have received messages.
Communication	1. Can users leave messages to others?	Users must be able to communicate with each other, even though they might not be simultaneously using the system.

Table 4.9: In synchronous collaboration tools the users use the system simultaneously. This makes working rather different from asynchronous applications. These questions can be used to evaluate if the requirements of synchronous applications are met.

Aspect	Question	Remark
Collaboration	1. Are users able to manipulate the shared artefact's?	Working simultaneously with common artefact's is a characteristic of synchronous collaboration. Users should be able to edit common document, for example, at the same time.
	2. Can users leave, create or join collaborative sessions?	Users need to be able to administrate these sessions.
Awareness	1. Can users see what others are doing?	In simultaneous collaboration, it is important to be aware of what others are doing. This helps the members of the group direct their work. Also in conversations, it is important to know what the other person is doing. This also includes users seeing who is also on-line. The information given this way should be as exact as possible, especially if the users are working with the same artefact.
	2. Can users determine whether the other person is available for contact?	When a user has withdrawn from a conversations, other users should be aware about the withdrawal.
	3. Can users control the information about themselves that is broadcast to others?	This includes their status in the system, their personal information, and if they are available for conversation.
	4. Can users determine when a shared artefact is being used or changed by others?	Users should be aware which artefacts are being used by others. This helps them to control their work and to see where interesting parts of the work might be.
Communication	1. Can users have instant verbal or visual contact?	User need to be able to talk or write about what they are doing. Common solutions to this are, for example, chatrooms.
	2. Can users have private conversations?	This is sometimes useful, especially if there are groups within the collaborating group. However, this is not always needed.
	3. Can users control their access to communication spaces?	Users need to have the ability to also dismiss the conversation if it is bothering them. In parts of the work conversation can be more disturbing than useful. This is especially true if the conversations are audio-based.

on presenting knowledge and information. However, in collaborative context the ability to present knowledge in understandable ways is also important. Thus, the users of the social mindtools need to be able to externalise their knowledge to a form that others can then internalise. Thus, social mindtools are a step towards representing and creating collaborative knowledge.

Chapter 5

Architecture of the Woven Stories Application

In this study an application based on the concept of Woven Stories was implemented. Implementation of the application was overlapped with the definition of social mindtools, presented in Chapter 4. This Chapter introduces architecture for the application. The architecture follows the concept of Woven Stories and the framework for social mindtools presented in Section 4.8. The architecture includes features from Woven Stories, but also introduces features that can only be implemented on a computer application.

5.1 Defining the Essence: the Core

The elements of material that make up the universe are called atoms. At the heart of the atom is the core. The core is built up from small particles called nuclei. In some ways, computer programs are built in a similar manner, especially when considering the object oriented programming paradigm. Small but important parts are combined together and a more complex application is composed.

When a particular process is implemented as a computer application, it is not always useful to implement the process in the same way as it would be performed without computers. The computerisation of the process often provides new features or functionality to its users, and therefore it gives a good reason to use the application instead of using the non-computer version. These new features can be small additions, such as ability to store or share the work easily. Even though the implemented software and the process that it implements could be different, they still share a common idea. This similarity is the *core* of the program.

During the development of the Woven Stories software, one of the main principles was to develop a system that could be used in several application domains with minor changes or extensions. In order to realise this principle, there was a need to define and implement the core of the Woven Stories application. This section describes the three nuclei of the Woven Stories core, which are essential in making the software work. These three nuclei are based on the framework for social mindtools presented in Tables 4.7, 4.8 and 4.9.

5.1.1 Nucleus I: the Concept

It is obvious that the most important part of the core is the concept of the Woven Stories. As a minimum, the application needs to provide exactly the same functions that can be achieved using manual woven stories process. The concept nucleus is based on Table 4.7 with the aspects of collaboration from Table 4.8 and Table 4.9.

As described in Section 3.1, there are two kinds of objects in the concept of Woven Stories that the application needs to manage. These types of objects are *sections* and *edges*. In order to work with these objects fully, users should be able to create, edit and view sections and edges. When creating a woven story with a computer there needs to be replacements for each of the materials necessary to create a woven story manually.

According to the concept of Woven Stories, the Woven Stories application needs to enable users to do the tasks shown in Table 5.1.

As can be seen from Table 5.1, the concept nucleus is strongly based on the manual version of Woven Stories presented in Section 3.1.3. This approach assures that all important features of the Woven Stories are present in the architecture. The features that make the computerised version different from the manual version are part of the other Nuclei and extensions.

Other computer applications have the computerised counterparts described in Table 5.1. For example, a simple drawing tool like Microsoft Paint has all the features that are needed to create a woven story. Users can draw the sections and links and are able to write the text into sections. It is important to note, however, that Microsoft Paint is asynchronous.

Synchronicity is a requirement for the Woven Stories concept to work. The group members need to be able to read, write or place the notes and strings to the wall whenever they want. However, if a woven story were created with Microsoft Paint, the authors would have to take turns on the same computer. This would be a major drawback when compared to the manual process, since it would mean that the users would have to wait for their turn. This can cause dramatic decrease in performance, as discussed in Section 4.6.2.

Table 5.1: The features of the Woven Stories concept and their counterparts in the computer application at a conceptual level.

Nucleus I: the Concept	
Manual Feature	**Computerised Feature**
Wall	*Shared Storyspace* • A display on which the objects can be placed • All group members must have access to it
Post-It Note	*Section* • Can contain user specified text • All users should be able to see it
String	*Edge* • A line connecting two sections • All users should be able to see it
Pen	*Text Editor* • A tool with which the contents of Sections can be edited by the author
Tape	*Flexibility* Users should be able to: • place the sections and edges in the shared storyspace wherever they want • move the sections and edges as they wish • remove sections and edges they have created by themselves

5.1.2 Nucleus II: Awareness

When a group of people are working in the same room, they are aware of what is happening around them. When work is performed on a computer application, the situation is not the same; people are not aware of what is happening around them as discussed on Section 4.6. By using Table 4.4 it is possible to identify the main features of an application that fosters awareness.

A manual woven story is created within one room. The time needed to create a woven story might be rather long, say over the course of several days, which means that probably not all the group members would be present at the same time all the time. However, it is likely that there are times when multiple members of the group are present simultaneously. Thus, while creating a manual woven story the work can be both asynchronous and synchronous.

When a creator of a manual woven story enters the workroom, the first thing observed is *who the other participants are* and *if they are present.* After that the person checks *what modifications have been done*, *what is new*, and *what the others have done.* If there are other persons in the room, it would be interesting to know *what they are doing* and *whether they are available for conversation.* One important question, often neglected though, is *what have I done*? This question is often asked, when a person has returned to the workroom after a period of absence. Also in situations where the story consists of a large number of sections, it might be difficult for a person to identify their own constructions.

By taking into consideration the questions presented above, Table 4.4 and the awareness related issues noted in Table 4.8 and Table 4.9, it is possible to determine what features are needed in Nucleus II, the awareness nucleus. The awareness related features of Woven Stories application are presented in Table 5.2.

5.1.3 Nucleus III: Communication

If a group creates a manual woven story, the woven story needs to be created in one room. They are not necessarily there at the same time, but it is likely that they know each other and will meet other members. When two people meet, they often have a conversation. The conversation might deal with the work they are doing, or might be small talk. Either way, the ability to communicate enables them to share their ideas.

When the concept of Woven Stories is implemented as a computer application, the possibility for communication between group members is decreased compared to manual woven story. Group members using a computer version of Woven Stories might still share a common workroom or classroom and could still talk to each other. The group members could also use other tools for communication, such as

Table 5.2: The second nucleus should provide answers to the questions presented in this table. The answers enable the group members to be aware of what is happening around them and how the work is proceeding. This table combines features from Sohlenkamp et Chwelos [120] and Gutwin et Greenberg [54].

Nucleus II: Awareness	
Element	**Question**
Presence and Availability	• Who is present? • Who is available for communication?
Actions	• What have they done? (past) • What are they doing? (present) • What have I done? (past)
Woven Story	• Who is using my contributions?

Skype [116].

One part of normal communication that is hard to implement in a computer-based system is the use of *conversational props* [24]. Conversational props are artefacts to which persons can refer to as they communicate. For example, in a manual woven story session a person could point to a section and ask why it was created. Thus the group members can immediately see which section that person was referring to.

Even though there are external ways to manage communication during the process of creating a woven story, the Woven Stories application core should offer users some basic communication tools. The minimum requirement is that users are able to have a text-based conversation in which all members can participate. Since the work of Woven Stories is strongly affected by visual representations of a story, users should also be able to refer to these representations while communicating with others. The requirements of the communication nucleus, as presented in Table 5.3, are based on the communication aspects of Table 4.8 and Table 4.9.

Even though this nucleus does not seem to have many requirements, it may be the most important one for the users, especially when the work is collaborative. By default, collaboration is seen as a process where group members communicate with each other in order to share ideas and knowledge. The importance of communication can be seen in any collaborative computer application; they all have some features that allow people to communicate. However, the communication nucleus is also

Table 5.3: Communication is an important part of the Woven Stories process. The minimum communication requirement is to provide a text-based chatroom for users.

Nucleus III: Communication	
Concept	**Computer**
Talking	Users should be able to talk with other members of the group
Referring to objects	Users should be able to refer to objects while communicating

the one that suffers the most when implemented in a computer application. Face-to-face communication cannot yet be replicated in a computer application in such detail that it is as authentic as the real thing. It is difficult to communicate body language, gestures, and other subtle nuances of non-verbal communication via an electronic device. However, several instant messaging programs already support the use of video based communication.

5.1.4 Fusion of the Core

The previous three sections have described the nuclei of the Woven Stories application's core. The nuclei specify the requirements needed for a successful implementation of the Woven Stories application. The structure of the core is shown in Figure 5.1.

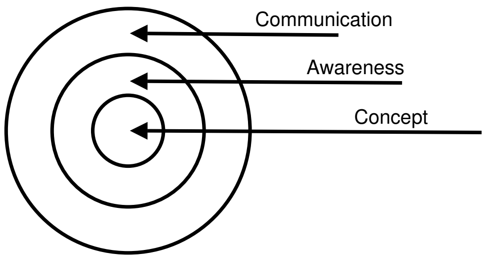


Figure 5.1: The core of the Woven Stories application consists of three layered nuclei. The most essential, the inner nucleus, is the concept itself. The second nucleus concerns awareness and the outermost concerns communication. These three nuclei are the core of the Woven Stories application.

At first it might seem strange that awareness is considered to be more fundamental than communication. While communication is important, in real-life setting the awareness aspect comes first. For example when entering a room, a person shall see who is in the room. In order to communicate, a person needs to be aware of those available. Awareness is a prerequisite for communication.

The features that these nuclei are concerned with are important for a computer application based on the concept of Woven Stories. From the above discussion, it has been shown that only the concept nucleus features are compulsory for an application. However, if the application were implemented with only the concept nucleus features, it could not fulfil the requirements for efficient collaboration. Due to this, all three nuclei are equally important prerequisites for a successful implementation of a Woven Stories computer application.

5.2 Extending the Core

Usually people do not want to use technology to recreate education just as it is [112]. The same goes for other application domains. Why use technology to do something that can be done even without it? There should be added value in order to motivate the creation of new software. This should also hold true for Woven Stories.

One of the key issues in the design of the Woven Stories application was versatility. It is intentionally planned that the application could act as a base, or a platform, for different applications. The concept of Woven Stories itself can be used for many different purposes, but with a few extensions the number of possible uses can be multiplied. Some of the extensions might be features that ease the use of the concept of Woven Stories, and thus can even be seen as an important part of the concept itself. While others might add a completely new feature to the application. This section briefly covers the basic extensions that make the concept more efficient through the use of a computer application, as well as showing that there are some features that are almost impossible with the manual version of the Woven Stories.

5.2.1 Remote Working

One advantage of using computers is the potential to communicate through the use computer networks. Networks enable the transfer of information from one point to another within a short amount of time. Applications that use networks, such as the world-wide web and e-mail, have made it possible to communicate over long distances quickly and efficiently.

Networks have also made network-based collaboration possible. Collaborators can be located spatially far away from each other, but still are able to share files and exchange e-mail rapidly. Several applications allow people to work within the same virtual workspace at the same time.

The original concept of Woven Stories requires that all the group members have access to a certain physical space where the story is located (e.g. to a wall of a certain room). This means that the group collaborating on the story must be located in the same physical place. In order to enable a spatially distributed group to work with the Woven Stories application, the story space should be accessible via a network.

In order to achieve this, the Woven Stories application should be based on network technologies. One of the requirements of social mindtools, presented in Table 4.6 is that the application is easily available, preferably web-based. For example, it could be a client-server based application, where all the data is stored at the server and is accessible at all times. If the system were implemented like this, the users would be able to use it anywhere and anytime.

5.2.2 Managing, Representing and Retrieving the Stories

Computers are good at storing data. Due to this the advantage of a computer applications is that the data that are produced can be easily stored, edited and shared, even by more than one user at a time.

It would not be worthwhile to create a system that could only handle one story at a time. Due to this reason, story management is an important feature of the Woven Stories application. Story management should enable users to create and edit new stories (also called documents), access them whenever they wish, and to delete them when they are not needed any more. The concept of story management is close to that of knowledge management (see Section 4.3). These features are covered in Table 5.4.

Table 5.4: The Woven Stories application should be able to manage several stories simultaneously.

Issue	**Feature**
Editing	User should be able to • create new documents • edit the document • delete a document
Storing	The system should be able to • store several documents at the same time • provide access to the documents for the users

5.2.3 User Management

Network-based applications and services are usually restricted to certain, predefined user groups. This prevents unwanted users from accessing private or confidential information. Besides adding security, user management also provides additional information that the application can use. For example, the application might automatically log user activity and provide user awareness related information (e.g. when a user is online or busy).

In order for a woven story to be successfully created, users have to know what others, and themselves, have already written. It is important to know who the author of a certain section is. Persons who note that they share common interests or similar thoughts with another user might want to pay more attention to that person's work. Without proper user management this could not be done.

What is more, the issues of ownership of sections and edges are easy to solve with proper user management. When a user creates a section, the system can mark it with that person's identification number. Afterwards, only this person can edit the contents of the section or delete the section. This raises a rather interesting issue related to the ownership of woven story sections. If a person has contributed a section to the woven story, can that person alone decide whether the section should be deleted? Should the decision be done by all the users of the system? Or would it be enough to ask the persons that have linked to that section?

The user management features that extend the core of Woven Stories are presented in Table 5.5.

Table 5.5: In network applications the user management is for security reasons. While it can serve for security, it can also be used to collect important information; for example, about ownership and editing.

Issue	**Feature**
Restrict	• Only authorised users should be allowed to access the software
Serve	With Sections, Edges and Documents (objects): • Who has created the object and when? • Who has edited the object and when?

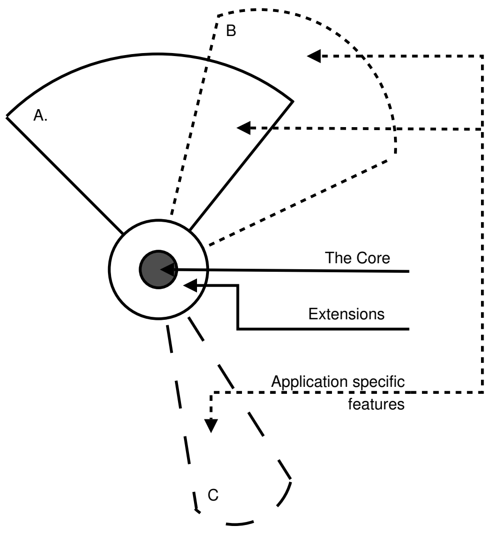


Figure 5.2: Different application domains need different features but they still share the Woven Stories Core. The features of the domain specific extensions can overlap, like in this case, applications A and B, or be totally discrete, as in application C.

5.2.4 Putting It All Together

One of the interests and aims in this study was to design the Woven Stories application to be a tool that could be used in several application domains. It is obvious that some tasks need specific features. Examples of these specific features are the *merge sections* and *split edge* operations that were implemented for the Woven Strategies tool (see Section 7.1.4).

Although different versions of Woven Stories have different features, they still need to employ the Woven Stories core in order to be called Woven Stories applications. Figure 5.2 represents the relationships between different domain-specific versions. It should be noted that the features can overlap or be totally discrete.

The implementation of the Woven Stories as a computer application opens new doors for the utilisation of the Woven Stories concept. First of all, collaborators do not need to be located in the same place in order to collaborate. The stories can now be easily saved and stored, which is not necessarily the case with manual stories. Most importantly, there are the possibilities for information retrieval, which facilitates the creation of new stories and the production of knowledge. With the possibility to add extensions to the Woven Stories application, the concept can also be applicable in several different application domains.

5.3 Summary

Based on the framework for social mindtools presented in Section 4.8, it is possible to derive three distinct nuclei that a social mindtool should implement. These nuclei are concept, awareness and communication. The nuclei are dependant on each other. Without concept, there is no need for awareness, and without awareness no communication related features are needed. By utilising each of the nuclei, the architecture of the Woven Stories application supports all of the characteristics of a social mindtool.

The core concept of Woven Stories is simple: there are sections containing text and edges that link these sections together in order to provide the flow of the story. Thus, the concept of a Woven Stories application is strongly dependent on these objects. The concept related requirements are presented in Table 5.1.

In collaborative applications awareness and communication play an important role. However, in some applications and some contexts these requirements can vary. Woven Stories is meant for both synchronous and asynchronous activities, and therefore certain features are required in a Woven Stories application. The minimal requirements in terms of awareness related features were presented in Table 5.2 and the minimal requirements in terms of communication related features were presented in Table 5.3.

By implementing the concept as a computer application, it is possible to introduce new features that support the collaboration and other Woven Story related activities. By extending the three nuclei with proper features, it is possible to meet all the requirements of mindtools and social mindtools. New features that are unique to Woven Stories applications, such as remote working, managing/representing/retrieving stories and user management, may become relevant to the concept nucleus.

Chapter 6

Current Implementation: Loom, WS-Server and WS@Web

The previous chapter described the requirements and the architecture of the Woven Stories application. This chapter introduces the implementation of the application. In order to distinguish the Woven Stories application from the concept of Woven Stories, the client is called Loom. The server is called WS-Server.

The aim was not to create a final version of the Woven Stories application. The aim was to build a tool that could be used in real settings in order to evaluate the concept of Woven Stories, its suitability for computer-supported collaborative work and whether it meets the requirements of a social mindtool. The priority for the implementation was that the application should implement the concept of Woven Stories as closely as possible. This means that all the features of manual woven story process should have been included in the application.

The first section briefly introduces the technical architecture and justifies its selection. Next, the functions of Loom are covered in terms of the three nuclei presented in Chapter 5. Some issues and problems that arose during the development are presented in the following section as well as the tools that were used to develop the application. An analysis of the information retrieval algorithms to be used with Loom is presented, which is followed by a short introduction to the next generation of Woven Stories application, WS@Web. Finally, I analyse Loom as a social mindtool and provide concluding remarks.

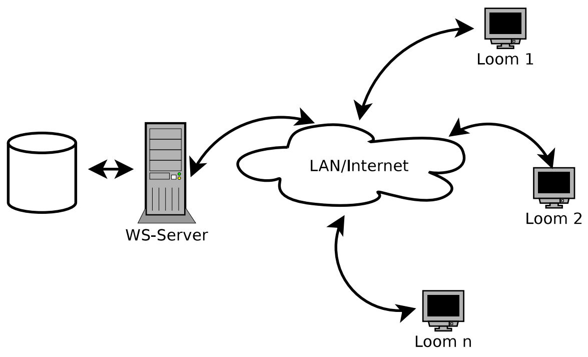


Figure 6.1: The Woven Stories prototype is based on a client-server architecture. All information between the client(s) and the server are transmitted through a network. The clients are not able to communicate without a server.

6.1 Technical Architecture

The requirement concerning the use of the Woven Stories tool for both synchronous and asynchronous work affected the choice of the basic architecture. In the concept of Woven Stories, simultaneous users should be able to edit the same document and preferably should have *relaxed WYSIWIS* (What You See is What I See) displays (see e.g. [52]). These relaxed WYSIWIS displays share the same data at all times, but every user can have a different view on the data at all times. Hence, a client-server based architecture seemed to be the only choice. A peer-to-peer approach could have been a solution, but it was not considered due to its complexity and because the data should preferably be stored in one place. Also, the decision was influenced by the fact that the previous prototype of the Woven Stories (WS2) was based on a client-server architecture.

One of the main ideas behind the development of Loom was that the software should be easy to use. This means that the installation and execution of both the client and the server should be simple. Due to this, the WS-Server includes an HTTP-server (Hypertext Transfer Protocol), so no installation of an external HTTP-server is needed.

The server and the client can be run on any computer running Java 1.4, or later. The server needs one free port for communication between the clients and itself. The server can be run on the same computer as the client, thus allowing Woven Stories to be used as a standalone program. The basic client-server architecture is presented in Figure 6.1.

Since Woven Stories uses the relaxed WYSIWIS approach, all users can see the same data at the same time. This does not mean that they share the view for the data, but that the data is displayed similarly in all clients. In order to accomplish

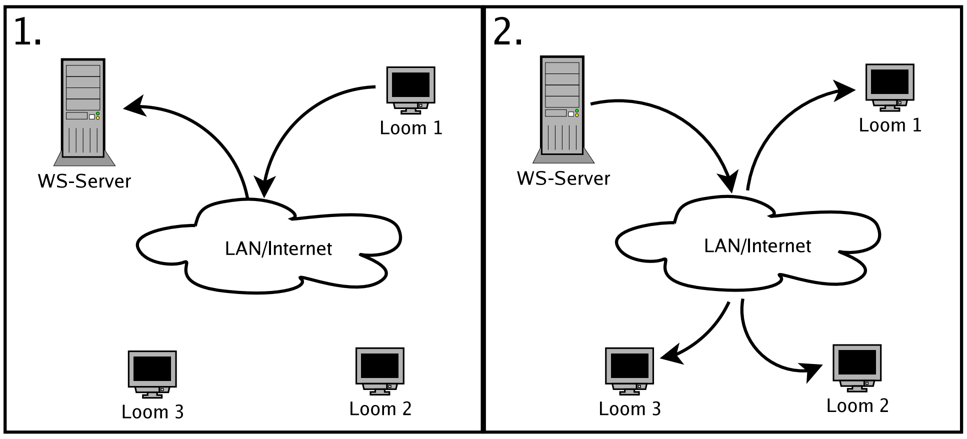


Figure 6.2: When a user edits or adds something in the client (Loom 1), the data is first sent to the WS-Server (1). After this, the server executes the required operations and sends the data back to all Looms (2). This ensures that all the clients share the same data at all times.

this, the communication between the clients and the server was designed in a specific way: when a user edits or adds data in the client, the data is first sent to the server. The server executes the required manipulation of the data, stores the data in the database if needed, and then sends the data back to all the clients. The clients receive the data and determine if they are interested in it. This approach is shown in Figure 6.2.

In Figure 6.2, the user of Loom 1 has done an operation that requires that data be sent to the server and other clients. Assume that the user has created a new object. The object would be first sent to the WS-Server (see Figure 6.2, part 1). The server would then store the data in the database and add more information to it; for example it might add a timestamp and a user identification number. After storing the data, the server would then create an event and send that event to all the clients (see Figure 6.2, part 2). The clients would then receive the event and decide whether they need the received data or not. If other users were currently using a view where the received data is important, they could immediately see this new data added by the user using Loom 1.

6.2 Implementing the Three Nuclei

As argued in Section 5.1, the core of the Woven Stories application should consist of three parts that are called nuclei. The requirements specified in Chapter 5 were used as guidelines for developing Loom. However, some of these requirements, such as awareness (see Section 5.1.2), have emerged and proved to be important during the design and evaluation and are thus not fully implemented in Loom. This is a result of the cyclic nature of this research (see Figure 2.2).

This section describes the current implementation of Loom (the applications client), based on the nuclei presented in Chapter 5. The order of the three nuclei is the same as in Section 5.1.

6.2.1 Nucleus I: the Concept

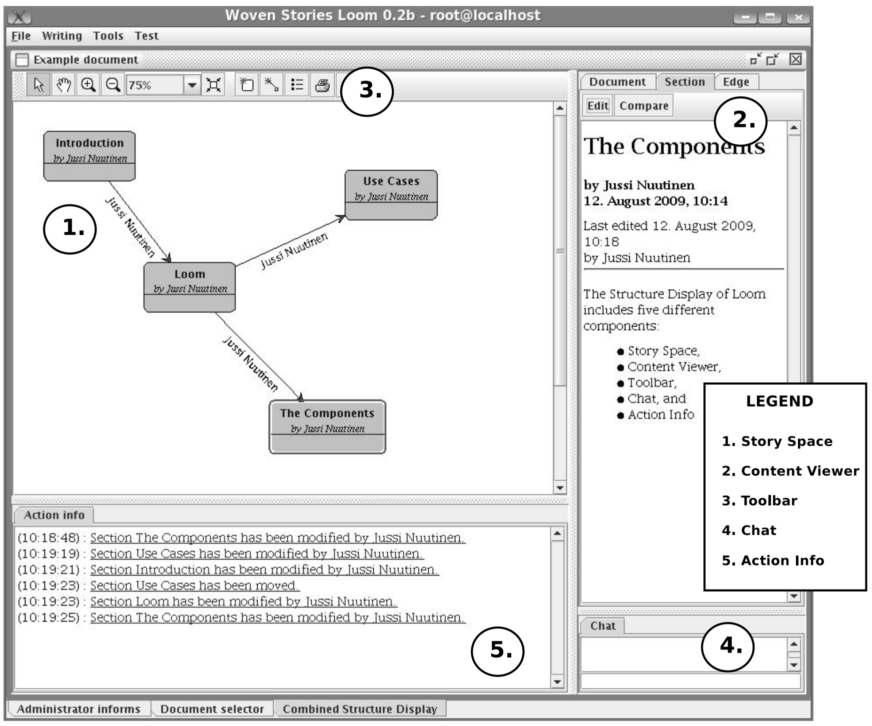
The concept of Woven Stories is the most important component of the application. In order to stay true to the concept and to the characteristics of social mindtools, the application should be easy to use, easy to learn and be an efficient and powerful way to create, process and represent the knowledge possessed by users. These aspects are difficult to measure, but it can be done by observing the users of the application. Nucleus I, the concept, described in Section 5.1.1 includes all the functions that an application should have in order to be called a Woven Stories tool.

In order to emulate the manual version of Woven Stories, all features of Nucleus I, the concept, were implemented in one window. This window is called the *Structure Display*; its most visible task is to illustrate the woven story in a graph like format. The Structure Display, shown in Figure 6.3, provides features to edit the graphical representation of the document. The Structure Display also includes features for Nuclei II, awareness, and III, communication.

The Structure Display is divided into five different areas:

- The *Story Space*, which shows the document as a graph (See 1 in Figure 6.3).
- The *Content Viewer*, which displays the textual contents of the document, sections, and edges (See 2 in Figure 6.3).
- A toolbar (See 3 in Figure 6.3), which is used to select tools to edit the visual representation of the document.
- *Chat*, which allows users' to communicate (4 in Figure 6.3).
- *Action Info*, (See 5 in Figure 6.3), which tells users what has happened in the document lately.

Editing the structure as well as adding and deleting sections and edges is performed within the Story Space. This is the area that takes up most of the space in the Structure Display. If a user wants to add a new section or edge, it should be added to the story space first. After this, its title and contents can be edited. Whenever a new item is added to the story space, it is immediately sent to all the



The Components

by Jussi Nuutinen
12. August 2009, 10:14

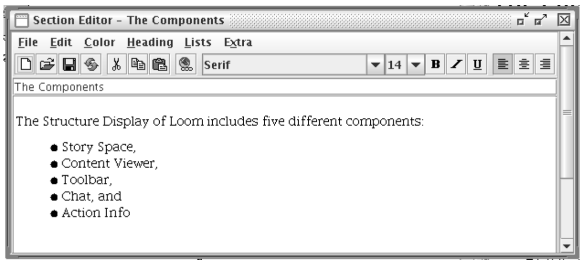
Last edited 12. August 2009, 10:18
by Jussi Nuutinen

The Structure Display of Loom includes five different components:

- Story Space,
- Content Viewer,
- Toolbar,
- Chat, and
- Action Info

```
(10:18:48) : Section The Components has been modified by Jussi Nuutinen.
(10:19:19) : Section Use Cases has been modified by Jussi Nuutinen.
(10:19:21) : Section Introduction has been modified by Jussi Nuutinen.
(10:19:23) : Section Use Cases has been moved.
(10:19:23) : Section Loom has been modified by Jussi Nuutinen.
(10:19:25) : Section The Components has been modified by Jussi Nuutinen.
```

Figure 6.3: The functions of Nucleus I, the concept, are included in a window called the Structure Display. This window also includes features from Nuclei II, awareness, and III, communication.


Section Editor - The Components
File Edit Color Heading Lists Extra
Serif
14
B
U

The Components

The Structure Display of Loom includes five different components:

- Story Space,
- Content Viewer,
- Toolbar,
- Chat, and
- Action Info

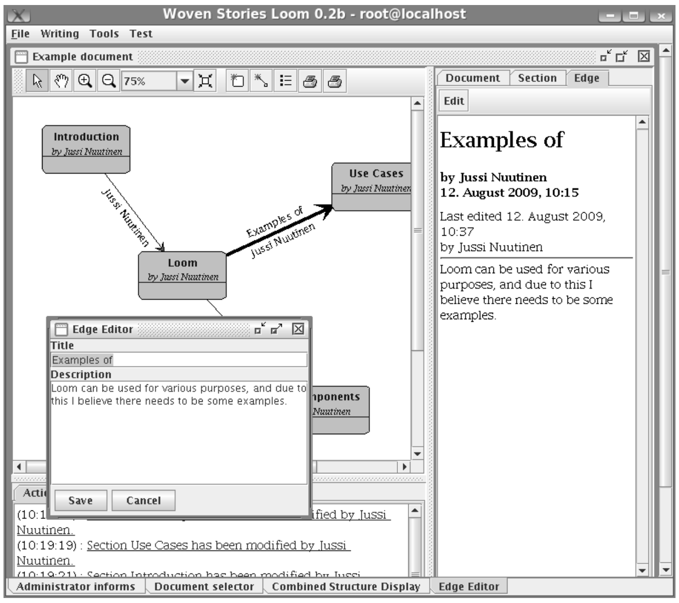
Figure 6.4: The section editor is like any simple word processing tool.

clients; even the user that added the item cannot see it before it has been sent back from the server. This way, users can see what others are doing at the time and, thus, are aware of their actions; this is also related to Nucleus II, awareness. After the user has given the section a title and written in text, the section is again sent back to other clients. At any time, all users browsing the same document share and use exactly the same data.

If a user wants to read the contents of sections of the document, the user needs to first select that section by clicking on it in the story space. Whenever a section, edge, or document is clicked, its content, title, and meta information, such as the creator's name and the time of creation, are shown in the Content Viewer (See 2 in Figure 6.3). The tabbed panel of content viewer changes according to the type of the item the user has selected. In case the user wants to edit the contents of the selected item, the edit button on the toolbar of the Content Viewer brings up the editing view.

In order to edit the contents of a section, a separate editor window is provided. This editor, shown in Figure 6.4, has the look and feel of a simple text editing tool. The editor produces HTML formatted text. Because it formats the text while the user types it, it is a WYSIWYG (What You See Is What You Get) interface. Since much of the work in Woven Stories is related to writing, this editor is simple and easy to use, yet offering enough formatting capabilities to write well formed pieces of text.

While the editor for the sections is a versatile text processing tool, the editor for the contents of edges is very basic. The editor for edges is a simple window (Figure 6.5) that makes it possible for users to explain why an edge was created and give a visible title for the edge. It is not desirable for users to write long descriptions or important data about the edges, since the edges are not natural sources for information. Still, the descriptions for the edges were included, since the relationships between sections might not be that obvious and thus can need more clarification. Due to this, data given to edges can be seen as a sort of metadata. The possibility to add descriptions to edges enables new applications such as game



Title
Examples of
Description
Loom can be used for various purposes, and due to this I believe there needs to be some examples.

Examples of

by Jussi Nuutinen
12. August 2009, 10:15

Last edited 12. August 2009, 10:37
by Jussi Nuutinen

Loom can be used for various purposes, and due to this I believe there needs to be some examples.

(10:1 … ified by Jussi Nuutinen.
(10:19:19) : Section Use Cases has been modified by Jussi Nuutinen.
(10:19:21) : Section Introduction has been modified by Jussi

Figure 6.5: Edge editor and how its data are shown in the Structure Display.

scripting as presented in Section 7.1.5.

In Table 6.1 I compare the requirements presented in Section 5.1.1 with what Loom actually implements. As can be seen from Table 6.1, all the requirements of the concept nucleus have been implemented in Loom.

6.2.2 Nucleus II: Awareness

Although an important prerequisite for collaboration, awareness is not properly supported in Loom. There are just a few features that make it possible for users to follow what actions others have taken in the system. Most of these features support the synchronous side of collaboration; thus, the asynchronous side is neglected. The weakness of asynchronous part is mostly due to the fact that these kinds of features are not straightforward to implement. In Table 6.2 I compare the requirements of awareness to current implementation.

The first feature that supports awareness is the panel on the bottom left of the Structure Display window. This awareness tool is called *Action Info* (See 5 in Figure 6.3). The panel shows all the important actions by the users of the current document. For example, when a user adds a new section, the display informs all the other users about it. Since Loom is based on relaxed WYSIWIS, users have the same content, but might not see the same part of the document. This panel helps users to keep track of the contents and their modifications. Action Info shows all

Table 6.1: The features of Loom compared to the requirements of Nucleus I (Table 5.1).

Nucleus I: the Concept	
Feature	**Implementation**
Shared Storyspace	
• A display on which the objects can be placed	*Story Space* implements this.
• All group members must have access to story space	All users who are in the same session have equal access to Story Space.
Section	
• Can contain user specified text	Sections are represented as boxes in the Story Space. Users can add any text to the sections.
• All users should be able to see	Sections are shown in the Story Space. All users browsing the Story Space can see the same sections at all times.
Edge	
• A line connecting two Sections	Implemented as lines between the sections. An arrow represents the direction of the Edge.
• All users should be able to see	Edges are shown in the Story Space. All users browsing the Story Space can see the same edges at all times.
Text Editor	
• A tool with which the contents of Sections can be edited	Edge Editor implements this.
Flexibility	
Users should be able to	
• Place the sections and edges in the shared storyspace wherever they want	This is possible in the Story Space.
• Move the sections and edges as they wish	All objects of the story can be moved freely within the Story Space.
• Remove sections and edges they have created by themselves	Users can remove any objects they have created by themselves. Objects created by other people cannot be removed.

add, edit, and delete actions.

The second feature that supports awareness helps users to identify the authors of the sections. For teachers, and other users, it might be useful to link the authors with their contributions. If the document has only a small amount of sections, it is rather easy to read the author names from the sections. However, in documents with a large number of sections, the identification of authors is difficult. In order to make identification easier, it is possible to colour-code the sections based on the author.

This author identification feature is simple to use. A user selects the tool from the toolbar and then selects the authors and desired colour for author's sections. The list of authors only includes the authors that have contributed to the current document.

The author identification feature is useful in several circumstances. For example, user A might have noticed that user B writes in a similar way to user A and that they share a common interest. In order to locate all the sections that user B has written, user A could colour-code all the sections written by user B. For a teacher doing assessment this feature is also beneficial, since the teacher can easily locate the sections written by the student currently being assessed.

The latest addition to the awareness related features is the possibility to track the sections which the current user has already read. Each section is, by default, marked with text "unread". When user has read a section, it can be marked with status "read". This makes it easier for a user to keep track on the sections that have already read. This feature was added based on the feedback obtained from the evaluation.

6.2.3 Nucleus III: Communication

The last and smallest nucleus of the Woven Stories core is communication. For asynchronous collaboration there are no communication tools in Loom, but the *Chat* feature facilitates synchronous communication (See 4 in Figure 6.3). The Chat tool is located in the bottom right corner of the structure display. Chat works like any other chat, echoing written text to all the clients that are currently browsing the same document. In Table 6.3 I compare the requirements to the current implementation.

Even though the chat was designed to fulfil the need for communication, it apparently is not enough. There is still a need for tools with which the users are able to communicate more precisely about the contents of the document. Instead of saying "the section called beginning" users should be able to point that section out from the document somehow.

Table 6.2: The features of Loom compared to the requirements of the Nucleus II (Table 5.2).

Nucleus II: Awareness		
Feature	**Exists**	**Implementation**
Presence		
• Who is present?	X	Partially implemented. From the *Chat* (see Section 6.2.3) users can see who logs in as they are in the system, but they cannot determine who is currently on-line.
Actions		
• What have they done? (past)	X	Partially implemented. Colouring of sections enables users to see who has created what.
• What are they doing? (present)	X	Partially implemented. All edit, add and delete actions can be seen from the Action Info panel and from the Story Space.
• What have I done? (past)	X	Partially implemented. Colouring of sections enables users to see what they have created and users can keep track on sections they have read.
Communication		
• Who is available for communication?		Not implemented.
Woven Story		
• Who is using my contributions?		No specific tool for this. Users need to browse the document to find the answer.

Table 6.3: The features of Loom compared to the requirements of Nucleus III (Table 5.3).

Nucleus III: Communication		
Feature	**Exists**	**Implementation**
Talking		
Users should be able to...		
• talk with other members of the group	X	Chat implements this. However it is not possible to establish private conversations.
• refer to objects while communicating		Not implemented.

6.3 Making It All Work - Adding Extensions

Although the features described in Section 6.2 are the core of Loom, they are not enough to make the application work properly. There is a need to manage the users and their documents as described in Section 5.2. Sometimes users want to export parts of the data they have created. In this section I go through the extensions that are implemented in Loom. As Section 6.1 has already introduced the client-server architecture that enables distance access and distance work, it is not covered in this section.

6.3.1 Story Management

A story, conversation, concept map, or any artefact that users elaborate with Loom is located in a *document*. A document is one story space where a woven story is created, thus it could be seen as one file. Usually a document contains only one woven story, but it is also possible that one document contains several woven stories, or storypaths that are not connected. Figure 6.6 represents this kind of situation where there are multiple stories in one document. Since many users and user groups can share the same server, there is a need to have an efficient document management system.

In the previous versions of Loom, the documents were in one list, ordered by the time they had been created, as shown in Figure 6.7. While testing the software among developers, this approach was soon rejected, since it was difficult to find the desired document when the amount of documents increased.

In order to make searching and managing documents easier, a tree hierarchy was selected. The current document management system is similar to a common file

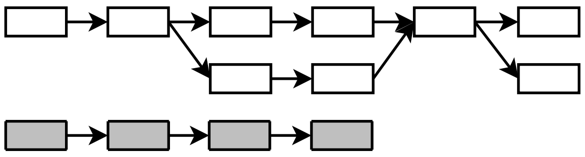
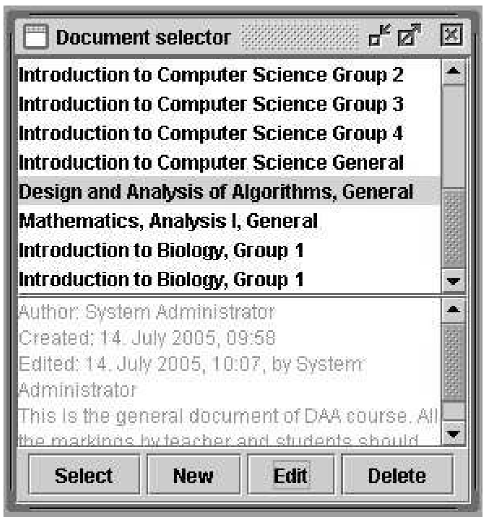


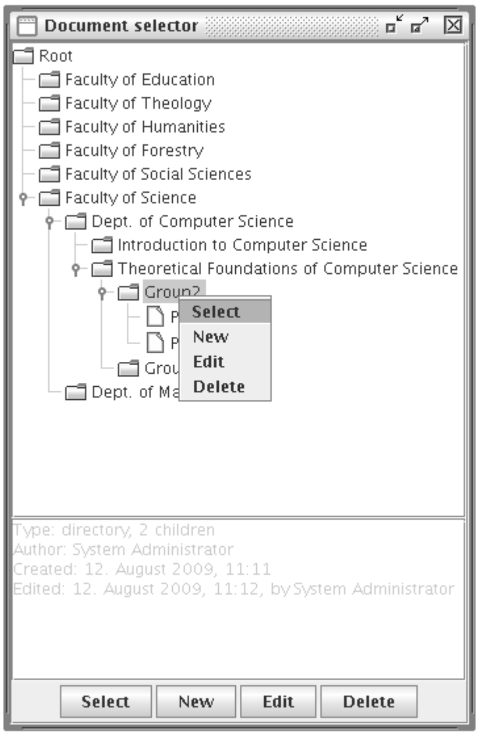
Figure 6.6: One document can contain several woven stories as in this example (grey and white).



Introduction to Computer Science Group 2
Introduction to Computer Science Group 3
Introduction to Computer Science Group 4
Introduction to Computer Science General
Design and Analysis of Algorithms, General
Mathematics, Analysis I, General
Introduction to Biology, Group 1
Introduction to Biology, Group 1

Author: System Administrator
Created: 14. July 2005, 09:58
Edited: 14. July 2005, 10:07, by System Administrator
This is the general document of DAA course. All the markings by teacher and students should

Figure 6.7: The older versions of Loom prototype had a document selector window that showed the documents as a list. This approach was found too difficult to use when the amount of documents increased.



Root
Faculty of Education
Faculty of Theology
Faculty of Humanities
Faculty of Forestry
Faculty of Social Sciences
Faculty of Science
Dept. of Computer Science
Introduction to Computer Science
Theoretical Foundations of Computer Science
Group2
P
P
Grou
Dept. of Ma

Type: directory, 2 children
Author: System Administrator
Created: 12. August 2009, 11:11
Edited: 12. August 2009, 11:12, by System Administrator

Figure 6.8: The new document selector is based on a tree hierarchy. It is easier and faster to use.

system browser on an operating system; the documents are stored in folders and the structure is visualised as a tree (see Figure 6.8). New documents or folders can be created inside a folder.

The hierarchical visualisation complicates the process of creating a document. First, the user needs to locate the correct place for the document, then create a folder if needed, and then create the actual document. In the old version of the document management system, the user just created a new document. The old approach can still be used by just neglecting the folder creation feature.

Data for the document selection window is queried from the database only when needed. For example, if a folder contains children then it is shown with slightly different icons to indicate that there are children within that folder and the children are not actually loaded until they are requested. This solution prevents unnecessary data being fetched from the database, thus saving bandwidth.

The document selection window also offers all the operations needed in order to create, edit, and add folders and documents. All the operations are offered from a button panel, located at the bottom of the window, or via a pop-up menu,

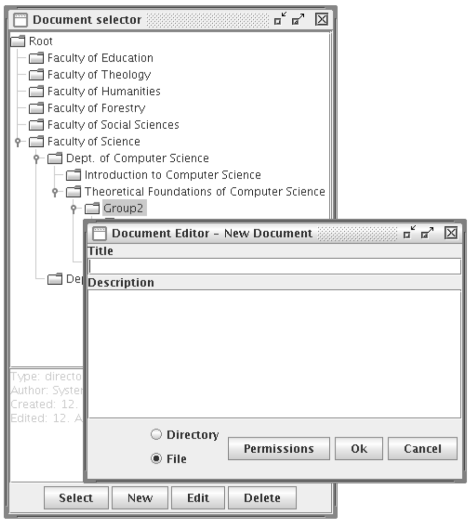


Figure 6.9: The window for document and directory editing. If a user wants to create a directory instead of document (file), the radiobutton on the bottom of the window should be set correspondingly.

as presented in Figure 6.8. The document can be selected by double clicking the document from the tree or by selecting it and then clicking the select button or choosing select from the pop-up menu.

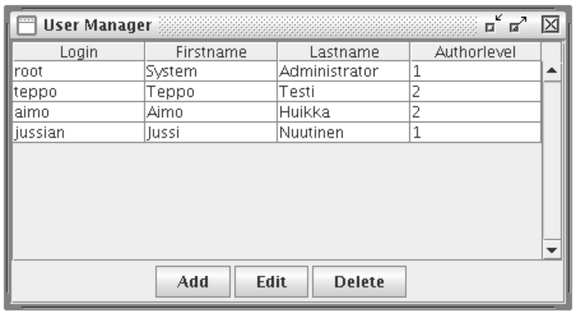
For editing existing documents and folders a simple window is used as shown in Figure 6.9. This window lets a user create new documents or folders and modify existing ones. A title, shown in the tree, and a description are given for any object created within this window. The title is important because it is used to identify the various documents and folders in the hierarchical tree. The description is metadata that specifies what the document or folder was created for. This metadata could be used in future implementations of Loom for searching the documents.

In addition to the capability for adding and editing documents and folders they can also be deleted. If a user wants to delete a specific item, that item needs to be selected first. After the selection, the document or folder can be deleted by clicking the delete button from the bottom button panel or by selecting delete from the pop-up menu. The software then asks the user to confirm the delete operation. In case the folder contains other folders or documents the program does not allow that folder to be deleted.

The story management features implemented in Loom meet the requirements introduced in Section 5.2.2. Table 6.4 compares the requirements of the story management to the features implemented in Loom. The requirements were taken from Table 5.4.

Table 6.4: The features of Loom compared to the requirements of Story Management (Table 5.4)

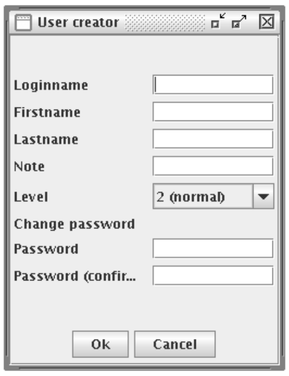
Story Management		
Feature		**Implementation**
Actions		
A user should be able to...		
• create new documents	X	All users can currently add new documents and folders to the system.
• edit the document	X	All users can edit the description and the title of the document and they can partake in the document creation process.
• delete a document	X	A user that has created the document can also delete it.
Storing		
The system should be able to...		
• store several documents at the same time	X	The system can store virtually any number of stories (documents) at the same time. The server stores these in a database.
• provide access to the documents for the users	X	When a user requests a story (document) the server loads it from the database and sends it to the Loom (client) that the user is running. Simultaneous users can access the same story.



User Manager

Login	Firstname	Lastname	Authorlevel
root	System	Administrator	1
teppo	Teppo	Testi	2
aimo	Aimo	Huikka	2
jussian	Jussi	Nuutinen	1

(a) The user management window allows access to add, modify and delete user accounts. Authorlevel 1 is the administrator and 2 is normal user.



User creator

Loginname	
Firstname	
Lastname	
Note	
Level	2 (normal)
Change password	
Password	
Password (confir...	

(b) User creator window.

Figure 6.10: The windows used for user management of Loom are very simple.

6.3.2 User Management

In order to distinguish one Loom user from another there needs to be user management functionality. In this section I present the features that are related to user management that are implemented in Loom.

A user first encounters a user management related feature when Loom starts up. The first screen the user will observe is a traditional login screen that asks for a login and password. The user must get these from the administrator of the WS-Server in order to be able to log into the system.

Besides logging into the system, a user does not directly need to use any user management related tool. However, the user is able to see information based on user management related data. For example, when a user reads the contents of a section, the user can determine who has created that section and when. Users do not need to add this information by themselves, but it is automatically created. The information from user management is also used when the Action Info panel announces a new event or when a person says something in the chat.

Although a normal user does not usually need to utilise the user management related features of Loom, administrators need to manage user accounts. Figure 6.10 shows both of the windows that the administrator is able to use for user management. The window on left is the *User Manager* (see Figure 6.10(a)). It lists all the users in the system. If the administrator wants to add, edit or delete users, this can be done by selecting the user from the list and pressing the appropriate button. If the administrator wants to add a new user, the window presented in Figure 6.10(b) is displayed. Loom supports only two levels of users (administrators and normal users).

Table 6.5: The features of Loom compared to the requirements of User Management (see Table 5.5)

User Management		
Feature		**Implementation**
Restrict		
• Only legitimate users should be allowed to access the software	X	Only users with valid login and password can access the system.
Serve		
• Who has created the object and when?	X	This information is shown for all the objects in the system (sections, edges, and documents).
• Who has edited the object and when?	X	This information is shown for all the objects in the system (sections, edges and documents).

Loom also provides basic user groups and permissions. A group is a collection of users. With these the administrators can control which users can view or edit certain documents. The owner of the document always has full control over the document. Read and write (edit) permissions can be set to group and others.

As can be seen from Table 6.5, all the requirements concerning user management have been implemented in Loom.

6.3.3 Exporting the Stories

Although Loom can be used for several purposes, in a loose sense it still is a production tool (i.e., a tool with which users produce texts). While the text in the system is useful for other members of the group, the user needs to be able to export the text at some point.

In order to export the stories a storypath (see Definition A.3) selection tool[1] was developed. With this tool the user first selects an appropriate storypath. The tool lets the user select only correct paths (i.e. paths where the order is correct). After the correct path has been selected, it can be exported. In order to export the data in a format other than pure ASCII, a template needs to be created. With these templates different output formats can be created. These templates are implemented with HTML mark-up, and contain specific tags to tell the exporter where to put each part of a section.

A whole document, instead of just one storypath, can also be exported using

[1]It also works with episodes (see A.2).

the same templating method. If a user wants to export the whole document, the user selects whether Loom should try to find the start sections. If the woven story contained in the document is cyclic, Loom cannot find the roots and the user needs to select them manually.

An example of an exported section is shown in Figure 7.4.

6.4 Implementation Details

6.4.1 Communication and Encapsulation

The second prototype of Woven Stories had a client-server architecture written in Java. Although it had many good features, it was rather difficult to implement new features or modify existing ones that required client-server communication. This was caused by the communication system that was implemented in that version. The communication required that all the data to be converted to strings before being sent, and then converted to the object again by the receiver. With this approach, simple things like adding a new variable to some class would cause much work to be done. Due to this fact, a new implementation was designed.

The communication and all the necessary features in order to accomplish the client-server functionality are implemented in a package called *ws_core* (which should not be confused with the Core of the Woven Stories application). The ws_core package includes classes that implement the HTTP-server as well as all the classes needed to create the client. Much effort has been put into improving the ease of use of the ws_core package. Even though this package was created in order to be utilised while implementing the Woven Stories tool, it could easily be used in any project that requires client-server communication with the HTTP-protocol. The ws_core package is not Woven Stories specific.

The encapsulation of the ws_core package, that is the communication features, also eases the programming of new features. The ws_core is implemented in a way that enables a developer to introduce new services to the server easily. The developer needs to extend one certain class from the ws_core, add the desired features, and after this the service can be used from the client. An example of this procedure follows later in this chapter in Section 6.4.2.

The ws_core package as well as all the packages of Woven Stories are divided into three subpackages: the client package, the common package, and the server package. This division is rather obvious, based on the needs of the client and server sides. The division is presented in Figure 6.11. The basis for the whole system is the core package. All the Woven Stories specific features have been added on the top of the core.

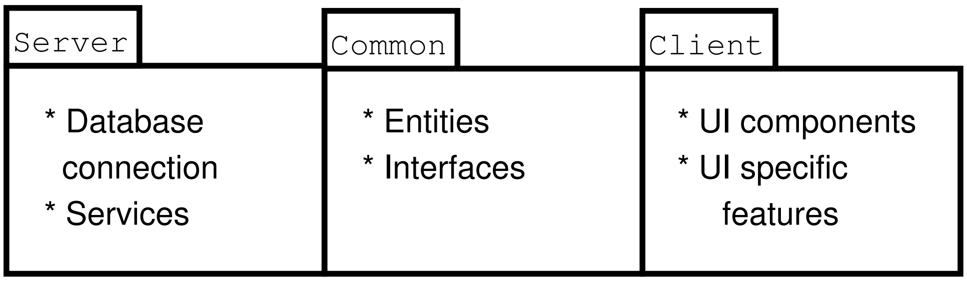


Figure 6.11: The architecture is based on three packages: the client package, the common package, and the server package. The basis for the system is the package called ws_core.

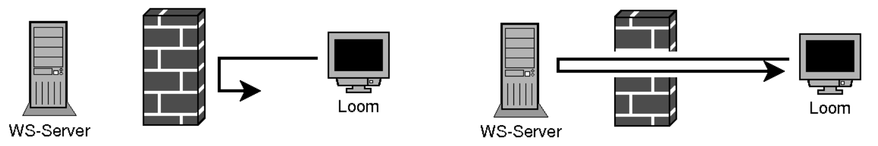
6.4.2 Implementing Services to Server

When a developer wants to add a new feature to the server, the first thing to be decided on is the kind of objects that will be transmitted between the client and the server. If the class for these objects has not yet been implemented, it needs to be created. First the developer needs to create an interface for the object. While designing the interface it should be kept in mind that this interface will primarily be used in the client-side of the program; thus, only the operations needed in the client should be introduced.

After the interface is ready, the actual class implementing the previously created interface should be created. This implemented class should extend a class from the ws_core called *AbstractEntityObject* and implement the previously created interface. To distinguish between these classes, the interface might be named, for example *Document* and the implementing class might be named *DocumentEntity*. Both of these classes should be stored in the common package of the software.

In order to create the actual service for the server, a developer should first introduce an interface for the service in the common package of the software. This interface should extend the *ServerObject* class from ws_core. Again, the service interface is primarily used on the client-side, thus it should implement only the features needed by the client. The service can transport all kinds of objects and primitive data types, as long as all the objects are serialisable. This interface should be named so that it can be easily understood by naming it is a service that handles certain objects. For example it could be named *DocumentManager* or *DocumentService*.

After the service interface has been implemented, the actual class should be introduced in the server package. If the interface were called, for example *DocumentManager*, the implementing class should be called *DocumentManagerImpl*. The implementing class does not need to implement or extend any other classes other than the service interface previously introduced. All kinds of operations can



(a) A firewall may block the communication.

(b) An open port solves the problem.

Figure 6.12: The firewall can cause problems within the client server communication.

be done in the service and all kinds of data can be returned to the client. If the data from the service should be transmitted to all the clients using the server at the moment, firing an event is the best choice.

When the service is ready, it can be used from the client-side as if it were a normal class. The only difference is when the service is introduced at the client. A developer needs to use a class called *ClientDispatcher* in order to use the service. The ClientDispatcher has a method called $getStubObject(Classintf)$ with which the service can be then used as needed.

The approach presented above might seem to be a bit difficult at the first glance, but when a developer becomes familiar with this approach, it makes introducing new services very easy. Since the developer does not need to know anything about the HTTP communication protocol, the coding is straightforward. All the data can be transmitted as is, without any modifications or parsing; the ws_core does everything necessary.

6.4.3 Loom and the Firewalls

Although the firewalls make using the Internet safe for common users, they certainly can make the life of a developer difficult. This is also the case with Loom and WS-server.

The application that has been created is based on a client-server approach. This means that in most cases the server is located on a computer other than the computer the clients are running on. This is not a problem when these computers are in the same LAN (local area network) and are not separated by firewalls. The problem occurs when the computers are not in the same LAN. The problem is visualised in Figure 6.12(a).

In order to make the server of the Woven Stories visible to the client outside the LAN in which the server resides, an open port for the server needs to provided so that that computers from the outside world can access the server via this port. This is visualised in Figure 6.12(b). In most cases this solves the problem, especially when the clients are run on home computers, where the user is able to setup the firewall alone. But when the computer running the client is behind, for example, a corporate firewall, this becomes more difficult.

In order for the client to contact the server via a port (e.g. port 4040), the same port should be open also from a firewall that separates the computer running the client from the internet. Considering the suspicious minds of the network administrators, it is not an easy task to get an open port. Actually in many cases the inability to have an open port might even prevent the use of the application, since the contact cannot be established.

One option to make the server visible to any computer on the Internet would be to introduce one's own webserver for the Woven Stories server. In this case, port 80 could be used for the Woven Stories server. This would make it visible to all the computers on the web, since this port is traditionally used by web-services: because the web-pages that can be seen come through port 80, no firewall blocks the information that comes through port 80.

This issue is one of the biggest reasons why the development of WS@Web was started. WS@Web is introduced in Section 6.5.

6.4.4 Development Tools

The first prototype of Woven Stories was implemented with Java and PL/SQL programming languages, HTML, and an Oracle 7 database. The tools that were used in the second prototype (WS2) were open source. WS2 was implemented purely with Java and it employed the Hypersonic SQL (HSQLDB) database engine. Before the development of Loom began, the selection of tools was a key issue.

When the tools for the current prototype were being considered, there were a couple of major aspects that influenced the selection. These included:

- price,
- availability,
- platform independence, and
- familiarity.

When considering the issues listed above, and when taking into account the fact that Java and HSQLDB were used in WS2, the selection for the most recent implementation was rather easy. Loom is implemented completely with Java and employs the HSQLDB database engine. It was decided that these tools were to be used because they are free and are well known among the students and researchers at universities. Java has particularly become the language of choice for most students and potential co-workers. Also, the ease of developing client-server-architecture-based software with Java and the platform independence Java offers were considered to be advantageous.

HyperSQL DataBase [128] (HSQLDB) is a free database engine written in Java. It is small in size and can be run embedded or as a server. The main reasons that HSQLDB were selected for the current prototype were that it is:

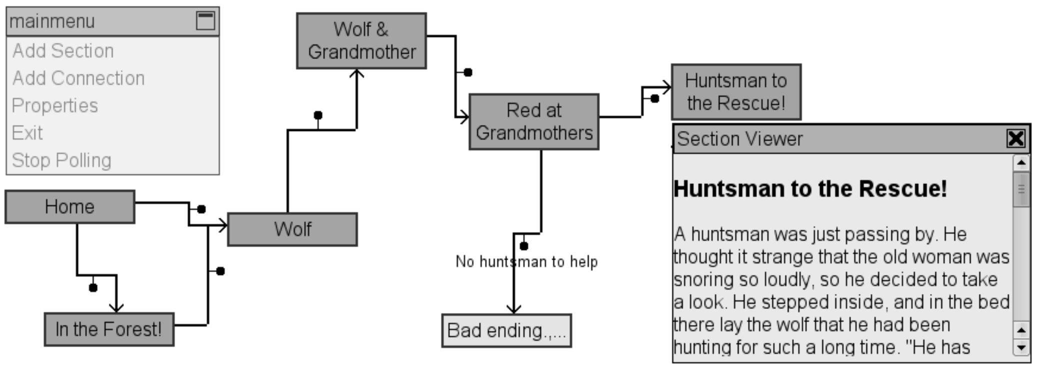
- free of charge,
- easy to install,
- embeddable, and
- platform independent.

In the spin-off version of the current Woven Stories prototype, called Woven Strategies (which is discussed further in section 7.1.4), an Oracle database was used. This was due to the fact that the environment for which the version was built had already been implemented using an Oracle database. As the work progressed we became more certain that the selection to use HSQLDB was correct because since, for example, it made it possible to run Loom as standalone application. Still, using Oracle led to one major feature to the current implementation of Woven Stories database manager; instead of using auto incremental columns, we use sequences to get the primary keys for tables since auto incremental columns were not supported in Oracle.

6.5 WS@Web

WS@Web is a JavaScript, CSS, AJAX and PHP based implementation of the concept of Woven Stories. The implementation was started based on the results of the retrospective analysis of Loom and its use cases (see Section 7.2). WS@Web is a proof of concept application for testing if and how it is possible to implement a Woven Stories application that can be run in a web-browser without any additional software.

WS@Web has a very similar architecture to that of Loom and it is also based on the architecture presented in Chapter 5. It has a client, written with HTML, JavaScript and CSS, which can be run in any modern web-browser. There is also a server, which is implemented with PHP and which manipulates a MySQL database [124]. Data between client (that is, in fact, the web-browser) and the server (a PHP page) is transmitted in JSON format [3] by utilising the *XMLHttpRequest* object (see e.g. [104, pp. 179-181]). In order to ease the implementation, JavaScript libraries Prototype [104, 106] and Script.aculo.us [104, 49] were used.



Huntsman to the Rescue!

A huntsman was just passing by. He thought it strange that the old woman was snoring so loudly, so he decided to take a look. He stepped inside, and in the bed there lay the wolf that he had been hunting for such a long time. "He has

Figure 6.13: User interface of WS@Web. Section Viewer has been moved with graphics editing tool under the corresponding section. Normally it opens at the exact same location as where section is located.

The client of WS@Web, presented in Figure 6.13, is one web page that implements all the features from the concept nucleus as presented in Table 5.1. Thus, there are boxes that act as sections, and arrows that act as edges. As in Loom, a user can drag the sections around the page. Edges are used to connect the sections in order to present the flow of the story, again in similar way to Loom. The biggest difference is perhaps the rectangular edges, which are due to certain limitations of HTML and CSS. Drawing straight edges could require even more DIV elements than the length of the edge would be. With the approach shown in Figure 6.13 three DIV elements is enough.

Every section can have a title and contents. The title is shown in the box that represents a section. Also the edges can have titles and descriptions.

All actions in the user interface can be activated either from the main menu or with a context menu (popup menu) that can be opened by pressing the right mouse button. For example, in order to add a section the user can right click the page. This opens a context menu where the only option is "Add Section". When this item is clicked in the context menu, a section is placed at the same location where context menu was opened.

When a user executes any operation in the user interface that affects the global state of the document, an event is sent to the server. The server is actually a PHP script which processes the event, does modifications to database if needed, stores events to database and gives a response back to the client as a JSON formatted string. The client then decodes the JSON string into an object, and processes the response if needed.

All events, such as additions of new sections and edits to contents, are delivered to the client by utilising another PHP script. A client knows which event it has received last time, and based on this information it calls this polling script from the server. If new events have been fired after the last poll, these events are retrieved from database and transmitted to client, again as a JSON formatted string. This approach is similar to that of Loom and WS-Server presented in Figure 6.2.

There is also simple functionality to manage users and documents.

The value of WS@Web lies in two important aspects. First, since it uses a web-browser as the client, all data is transmitted through standard HTTP port 80 and therefore firewalls are not a problem for WS@Web. Secondly, it is easy to install and above all, users do not need to install any other additional software to use the application. Based on the initial experiences of using WS@Web, it seems obvious that this is the way how Woven Stories should be implemented in the future. The current version of the WS@Web is available at `http://cs.joensuu.fi/wovenstories`.

6.6 Section retrieval methods

During the experiments with Loom (see Section 7.1), it was found that in order to efficiently use and construct new data, the users need tools to browse the existing data efficiently. It would have been possible and relatively easy to implement a "normal" keyword-based search. However, since the contents of the sections are full text, and often there is a need to find sections that are similar to each other, a keyword-based search is not enough. Due to this, a comparison between different *information retrieval* (IR) algorithms was done.

Methods of *Information Retrieval* (IR) [10] are used when there is a need to provide the user with easy access to the information they are interested in. Since these methods often involve natural language, such as written text, the retrieved objects might be inaccurate and the result set can even contain undesired objects [10]. Due to this, the methods used are measured with *Recall* and *Precision* that gives a measure of how well a particular method performed [10].

Recall is calculated by dividing the amount of retrieved relevant documents $|Ra|$ by the total number of relevant documents $|R|$ in the collection. Relevant documents are those documents that should be included in the result.

$$Recall = \frac{|Ra|}{|R|}$$

Thus, Recall tells how well the actual result set covers the expected result set. Using Recall alone omits the fact that the result set can contain undesired hits. Precision is used to measure this. Precision is calculated by simply dividing $|Ra|$ with the size of the retrieved document set $|A|$;

$$Precision = \frac{|Ra|}{|A|}$$

Precision makes the correctness of the used method explicit. There is typically a trade-off between Recall and Precision. This means that when Recall is high Precision is low and vice versa. Due to this, the developers need to evaluate which they appreciate more.

The following subsections cover the algorithms and present an analysis of each of the algorithms.

6.6.1 Frequency of Words

The first set of algorithms tested is based on wordsets that are generated from all the texts that need to be compared. Each text is preprocessed by removing all stopwords and delimiters such as commas and periods. Stopwords are words that are very common in a given languages, such as "and". After this a *wordset* for each of the texts is created. A wordset is a collection of words in the text in a relation with their frequency within that text. These sets are produced similarly to vectors W_1 and W_2 in Figure 6.14.

```
input  : Wordsets $W_1$ and $W_2$
output: Float $result$
common = 0 total = 0  forall word $w$ in $W_1$ do
    if $w \in W_2$ then
        common += MIN(amount of $w$ in $W_1$,amount of $w$ in $W_2$);
    end
    total += amount of $w$ in $W_1$;
end
return common / total;
```

Algorithm 1: FoW_1 algorithm is based on wordsets.

Based on these sets two different ways to calculate the similarity of the texts was used. The first approach uses the following algorithm 1 to calculate the similarity. Thus, the algorithm calculates the number of shared words in W_1 and W_2 and this value is then divided by the total number of words in W_1. This method is called FoW_1. It should be noted that this algorithm is not symmetric. Assuming the W_1 and W_2 from Figure 6.14 the return value for this algorithm for W_1 would be $\frac{2}{7}$ and for W_2 $\frac{2}{3}$.

The second approach uses the same wordsets but instead of comparing all the words, it includes only those words that have the highest frequencies. This is controlled with value *classes*. If, for example, $classes = 3$, the three most common words are taken into account. However, if some of the most common words share a frequency, these are calculated as one class, and thus the resulting set can have more than 3 words. Otherwise the calculation is similar to the approach used above. This second method is called FoW_2.

6.6.2 Cosine Similarity

There are three different algorithms tested during this process that are used based on this method. The first algorithm, Cos_1, does not use weights for the words, the second uses the term frequency-inverse document frequency, $tf - idf$, weights and the last one, *alpha*, is strictly an experimental algorithm that uses only letters of the compared strings.

First a global *word vector* V_g is created based upon all the texts compared. A word vector is a vector that maps a word to its frequency in a given text. Stop words are removed in the process. Based on the global word vector, a vector that includes the frequencies of the words of the given text is created. The process of creating these vectors is shown in Figure 6.14.

D_1 = "Grass is green and peas are green", D_2 = "green is colour"

$V_g = \{grass \Rightarrow 1, is \Rightarrow 2, green \Rightarrow 3, colour \Rightarrow 1, and \Rightarrow 1, peas \Rightarrow 1, are \Rightarrow 1\}$
$V_1 = \{grass \Rightarrow 1, is \Rightarrow 1, green \Rightarrow 2, and \Rightarrow 1, peas \Rightarrow 1\}$
$V_2 = \{green \Rightarrow 1, is \Rightarrow 1, colour \Rightarrow 1\}$

Figure 6.14: Example of creating global word vector V_g and local word vectors V_i.

The cosine similarity (and the desired result) of vectors A and B is calculated with the following formula:

$$similarity = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|},$$

where A and B are vectors of $\mathbb{R}^k$. The *dot product* $A \cdot B$ is calculated as

$$A \cdot B = \sum_{i=1}^{n} a_i b_i,$$

and the *magnitude* $\|A\|$ is calculated as

$$\|A\| = \sqrt{x_1^2 + ... + x_k^2}.$$

Based on these formulae, the production of the algorithm is trivial and is omitted here.

The calculation in the weighted version of this algorithm is the same, but instead of using only the number of occurrences of a given word in the text, it uses *term frequency-inverse document frequency* ($tf - idf$) as a weight for the words.

$tf - idf$ is based on the values tf and idf. Term frequency (tf_{ij}) for a given term (word) t_i in a given text T_j is calculated as follows:

$$tf_{ij} = \frac{n_{ij}}{\Sigma_k n_{kj}}$$

Furthermore inverse document frequency (idf) is calculated as

$$idf_i = \log \frac{|D|}{|\{d_j : t_i \in d_j\}|},$$

where D is the total number of texts and $|\{d_j : t_i \in d_j\}|$ is the number of texts that contain the given term t_i.

Now, after the $td - idf$ values have been calculated, vectors A and B are created based on these. The weight of terms in these vector is the multiplication of tf_{ij} and idf_i. Otherwise the calculation is similar.

The last algorithm uses the $tf - idf$ approach, but instead of using words as terms, it uses the letters of the texts as terms. Thus, the vectors are created from all the letters present in the given text. Then, the calculation is performed similarly to the weighted version of this algorithm. The stop words are not removed in this algorithm because we wanted this simple algorithm to be as plain as possible. This algorithm is called *alpha*.

6.6.3 Suffix Array-based Greedy String Matching Algorithm

Suffix Array-based Greedy String Matching Algorithm (SABGSMA) is presented by Mozgovoy et al. [86]. The reason for this algorithm being selected is that it seems to perform well when compared to other plagiarism detection algorithms and it is relatively fast. However, this algorithm is originally meant for detecting plagiarism of source code, and thus needed a small modification. The original version of the algorithm tokenises the source code stored in files before comparing them, but this version omits the tokenisation.

The algorithm is based on an index structure built over all the files (documents) to be compared [86]. By using this index structure the algorithm compares two given files Q and F_i by taking substrings of length γ from Q and searching these from the index. The γ value can be adjusted by the user. In this study the value of γ was 2.

Finally, the similarity between files Q and F_i is calculated as

$$Similarity(Q, F_i) = MatchedSubstrings(F_i)/q,$$

where q is the total number of substrings of length γ in Q.

A more in-depth explanation of this algorithm is available in [86].

6.6.4 Analysis and Implementation

The data used for the analysis was five different versions of the well known fairytale *Little Red Ridinghood.* A fairytale was selected as Woven Stories was originally intended for writing stories. All five stories were transferred to WS@Web and divided into 5 to 7 parts depending on the version of the story. The aim was to divide all the stories to similar parts; for example, each story had a section called *Wolf*, where Ridinghood first meets the Wolf in the forest. In total these five stories include 30 sections.

Based on the assumption that each individual part where Ridinghood leaves home was similar to other parts where Ridinghood leaves home, a matrix of true data was constructed. Altogether 142 matches were present in that matrix, 112, if the diagonal is eliminated.

After the data had been constructed, a testbed was implemented for the algorithms. The testbed compared all the different parts of the Ridinghood stories against one another producing a $30 * 30$ matrix for each method where each matrix cell $[i, j]$ included the result of the current algorithm when comparing parts p_i and p_j. Several different types of images and graphs were drawn to analyse the data. However, the main method to analyse the efficiency of algorithms was the recall-precision curve.

The recall-precision curve was constructed by calculating the values for precision for all the values of recall. This, in practise, was done as follows:

- all return values $x_1 ... x_n$ were sorted in such a way that $x_1 \geq x_2 ... \geq x_n$
- for each return value x_i it was calculated how many correct and false positive were got with return values $x \geq x_i$
- based on the values obtained in previous item, precision and recall values were calculated.

Then, all of these relations were plotted to Figure 6.15. Recall is on the X axis, thus explaining the rate of the relevant findings and precision is on Y axis explaining how many of the found articles were correct.

Based on Figure 6.15 it is clear that the *Suffix Array-based Greedy String Matching Algorithm* outperforms the other methods. However, there are several methods that challenge the SABGSMA method on low recall rates, the difference becomes evident with this data at recall rates larger than 0.2. Furthermore, it is evident that

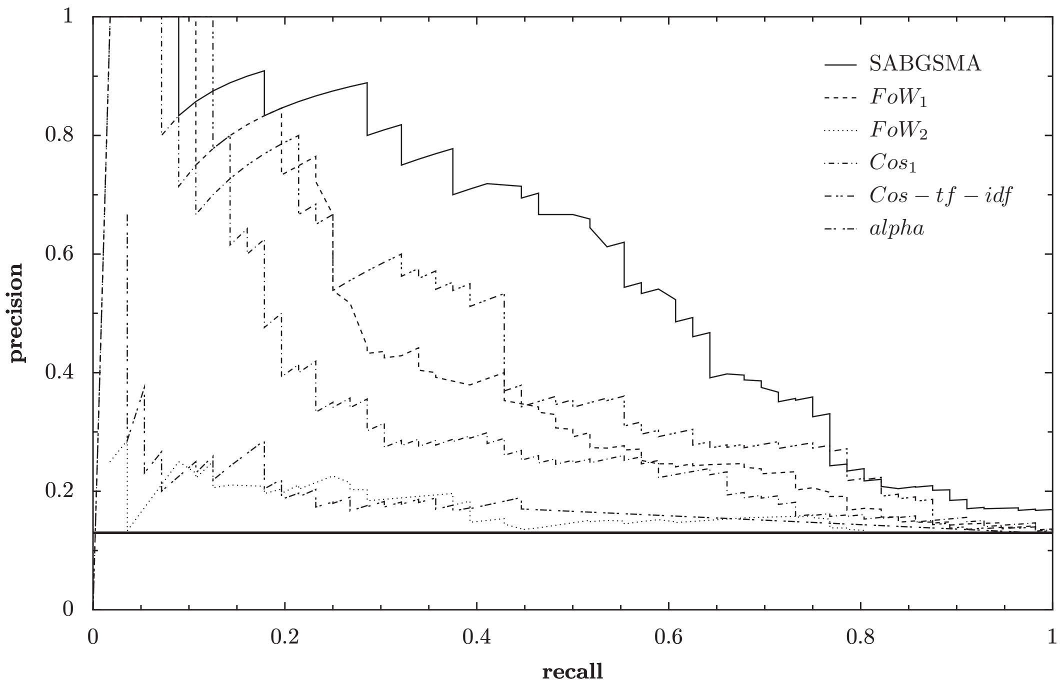


Figure 6.15: Recall-Precision curve of all the algorithms tested.

even the most naive methods are better than just randomly guessing (thick black line on Figure 6.15).

The success of SABGSMA is rather interesting, considering its background in source code plagiarism detection. The other interesting point is that this is actually the only algorithm with the *alpha* for which the stop words were not eliminated before execution. Surprisingly, if we eliminate the stop words for this method, the efficiency reduces significantly.

Based on this analysis the SABGSMA algorithm was selected as a base for the text searches. The implementation of the algorithm is integrated into WS@Web. However, since it is implemented with Java, it would be relative easy to integrate it also into Loom.

6.7 Loom as a Social Mindtool

Section 4.8 discussed the features that social mindtool should have. To evaluate what features of such an application Loom supports and which it does not support, in this section I compare the features of Loom to the features discussed in Section 4.8. First I compare and contrast the ideal features that high-quality mindtools should have

Table 6.6: Elements of mindtools compared to features of Loom.

Characteristic	**Question**	**Answer**
Access	Is application available?	Loom will be distributed as an open source application.
	Is application affordable?	See above
Engagement	Can it be used for knowledge construction?	Woven Stories, as the concept behind Loom, supports the knowledge construction process.
	Does it support critical thinking?	Using Loom forces the users to separate the knowledge into small, logical blocks and create the semantics between the blocks. This process assists the users to think through processed data more carefully.
Multi-purpose utility	Is it generalisable?	Loom does not have any features that would restrict it to specific subject domain, so it can be utilised in other domains.
	Can it promote transferable learning?	The concept of the Woven Stories teaches the users to think logically and to create links between concepts and knowledge.
Usability	Does it use simple, yet powerful formalism?	The visualisation of the story as a graph is simple, yet effective way to present stories and sequential data in general.
	Is it easy to learn?	Evaluations have shown that Loom is easy to use and learn.

with the features that were ultimately included in this implementation of Loom. Then I compare and contrast the synchronous features of Loom and finally the asynchronous features. The order is based on the fact that most problems of Loom are related to the asynchronous collaboration.

6.7.1 Loom as a mindtool

In order to evaluate if Loom meets the requirements of mindtools, a comparison to the characteristics of mindtools is given. This comparison is shown on Table 6.6. It provides answers to the questions presented in Table 4.7. Table 6.6 maintains the same structure as Table 4.7 for easy comparison.

An ideal mindtool is one where the software should facilitate knowledge construction. The basic idea behind the concept of Woven Stories is to support story telling. This idea, in addition to the graphic way to visualise the stories, can indeed be used for knowledge creation. Above all, Loom can enable a group to build on common knowledge in a collaborative way. A mindtool should also be able to support critical thinking. While working with Loom, users are encouraged to divide the data they produce into small pieces and then link these pieces accordingly. Dividing the data into small pieces forces the users to think critically about the contents; linking forces the users to see the relationships between the parts created.

Loom is a generalisable application, since it does not contain any features that would restrict its use in any specific subject domain. Furthermore, a mindtool should ideally foster generalisable, transferable skills that can facilitate thinking in various fields [68, p. 18]. Without any deeper evaluation this cannot be confirmed, but it can be still argued that learning to think in a "Woven Stories way" can be useful, despite the subject in which this way to think has been learnt. For example, dividing an English essay into small pieces can help that person later to divide a difficult mathematical problem into smaller pieces.

The last category of features for an ideal mindtool is usability. Usability includes two features. The first is that the ideal mindtool should have a simple, yet powerful formalism. The formalism used in Loom is similar to the formalism in many other tools. The documents are visualised as graphs and should be rather easy to understand even for the beginners. Still, by using the formalism of Woven Stories even complex structures can be constructed. Thus, it can be stated that the formalism used in Loom is indeed simple yet powerful. Secondly, the mindtool should be easy to learn. Tests carried out with Loom (see Section 7.1) have demonstrated that the application is easy to use and learn.

This analysis and the experiences from the evaluation, presented in Section 7.1 propose that Loom (and the concept of the Woven Stories) is a mindtool.

6.7.2 Loom as a Collaborative Tool

Loom was originally developed in order to be able to test the concept of the Woven Stories in a computer environment. One of the main points behind the development was to create an application that could be used in collaborative way. The current implementation supports this, but it works best when all the users are present in the same room at the same time. In this section I compare and contrast the features of Loom with the ideal features of synchronous and asynchronous collaboration tools, which were presented in the framework for computer supported collaborative mindtool in Section 4.8.

In Table 6.7 I compare the asynchronous features of Loom to the features presented in the Table 4.8. I address the elements of communication, collaboration and awareness by presenting the questions that correspond with those elements and by remarking on how those elements have been implemented in Loom.

Table 6.7: The features of Loom when compared to features of an ideal asynchronous collaboration tool.

Aspect	Question	Remark
Collaboration	1. Do the users have a shared working area?	Yes
	2. Can users coordinate the turn taking work?	The work is not based on turns.
	3. Can users identify the changes made to the shared artefact?	Asynchronously this is not possible.
	4. Can users merge divergent work?	Yes and No. Users can express their divergent thoughts by introducing new sections. However, no merging is possible expect trough negotiations.
Awareness	1. Can users see when the shared artefact has been changed?	Asynchronously this is not possible.
	2. Can users determine who has done the changes?	The information about the last modifier is present.
	3. Can users determine how the shared artefact was changed?	No.
	4. Can users see when and where others have left messages for them?	No.
Communication	1. Can users leave messages to others?	No.

A detailed analysis of Table 6.7 reveals the fact that Loom does not support asynchronous collaboration very well. Out of the three categories, the category of collaboration seems to be the only one that is supported well enough. In order to utilise Loom more efficiently in asynchronous collaboration, the elements of communication and awareness must be developed further. Features such as annotations [92] or notes could be useful for communication. Also, features that involve informing users about recent changes and updates should be considered.

In Table 6.8 I compare the features of Loom to the features required in an ideal synchronous collaboration tool. In Table 6.8 I cover the three categories of collaboration, awareness and communication, the relevant questions, and my remarks about the implementation of Loom.

By evaluating Table 6.8 carefully, observations similar to the observations from Table 6.7 can be made. Loom supports collaboration related aspects of synchronous collaboration, but the aspects of communication and awareness are not that well supported. However, the features for synchronous communication and awareness are better supported than the asynchronous features.

When implementing future version of Woven Stories application, the emphasis should be put on developing the communication and awareness features. With these features the application would be more usable in cases of distributed collaboration. Currently Loom works best in cases where all the group members work in the same room and can easily communicate about the work they are doing.

6.8 Concluding Remarks

While the current implementation of Loom and the Ws-Server can be used in order to evaluate the concept of the Woven Stories, there are still important issues that need to be solved or improved. The current implementation of Loom fails to implement many of the requirements of the three nuclei of Woven Stories Core. In particular, the second nucleus, awareness, needs further development. Conversational props, or similar features, should also be added in order to improve the communication between users.

Annotations and other features that ease asynchronous collaboration should be considered as additional features. Currently there is no possibility to add comments or annotations to the sections other group members have written. While lack of commenting and annotation features forces users to contribute their own ideas as new sections, this feature could be useful in some application areas. For example, a group of researchers writing a joint paper might appreciate annotations or comments feature.

In the current setting user rights are implemented in a limited manner. Currently

Table 6.8: Comparing the features of Loom to the features of an ideal synchronous collaboration tool.

Aspect	Question	Remark
Collaboration	1. Are users able to manipulate the shared artefacts?	Yes.
	2. Can users leave, create or join collaborative sessions?	Yes.
Awareness	1. Can users see what others are doing?	They can see whenever a new section or edge has been added or when any of these elements have been modified.
	2. Can users determine whether the other person is available for contact?	No.
	3. Can users control the information about themselves that is broadcast to others?	No.
	4. Can users determine when a shared artefact is being used or changed by others?	Only changes can be determined.
Communication	1. Can users have instant verbal or visual contact?	Only text-based contact is possible.
	2. Can users have private conversations?	No.
	3. Can users control their access to communication spaces?	No. Users are online in the chat at all times when they are present in a document.

there are only two user types: *administrators* and *users*. Administrators can do whatever they want in the system; users can only create new documents, sections and edges and edit or delete their own documents, edges and sections. Although this approach forces users to contribute their ideas as new sections, the user privileges certainly need to be redefined.

Finally, the firewall issues should be addressed in future implementations. The approach used in implementing WS@Web seems to be the way to proceed in the design and implementation of Woven Stories based applications.

Even with its limitations, the current implementation of Loom works well enough in order to be used in small-scale settings. It has been possible to use the application in order to evaluate the concept of the Woven Stories in enough detail.

Chapter 7

Analysis of Loom

This chapter covers six different application areas where Loom has been tested. After the cases have been presented a comparison between Loom and a Wiki is provided. Finally a summary and an answer to research question Q4 is provided.

7.1 Use Cases

The current implementation of Loom has been used in several different case studies. These cases are completely different from each other with respect to their target population and use. This section covers analysis of six selected use cases of Loom. Table 7.1 presents the characteristics of these reported cases.

The first case centred around a brief Woven Stories session with the children participating in Kids' Club [46] at the University of Joensuu. The aim of the case study was to test Loom, to find out if new features should be introduced, and to see how the children in Kids' Club feel about using the system. A brief overview and the results of the Kids' Club case are described in section 7.1.1.

The following two case studies differed from the first one substantially. The participants in the case studies were university students and the purpose of the studies was not to run a Woven Stories process, but to employ Loom in another application domain other than writing. First of these case studies was carried out with Computer Science students creating concept maps and the second case study was carried out with Forestry students who were supposed to create and report on a simulation process. In subsection 7.1.2 I describe the study with Computer Science students and in section 7.1.3 I describe the study with the Forestry students.

The fourth case was entirely different from the three previously introduced cases.

Table 7.1: The six different cases where Loom has been utilised.

Case		Target group	No. users	Task
I	Kids' Club	Participants of Kids' Club	12 on nine computers	To write about their work during a year in the Kids' Club.
II	Concept Mapping	Computer Science students	9	To create a concept map out of the contents of *Theoretical Foundations of Computer Science* (Theory of Computability) course.
III	Forestry	Forestry students	5 divided into two groups	To report the simulated forest growth.
IV	Strategy	Business		To write a corporate strategy with Woven Strategies tool
V	Game Scripting	Participants of SciFest 2007	64 on four workshops on groups of four	To write a script for a text-based adventure game.
VI	Debating	University students	38	To participate in a debate on various topics.

The case employed Loom for corporate strategy planning. The case was carried out by Markku Salo as a part of his PhD thesis. In section 7.1.4 I describe this case in more detail.

The fifth case was run at SciFest 2007 held at Joensuu. SciFest is a festival for science and technology where the target audience is pupils of 11-16 years old and high school students [133]. A total of 64 school children participated in the workshops where the aim was to create a script for a text-based adventure game. This case is presented in section 7.1.5.

Finally, the sixth case included university students from University of Joensuu, Finland, Akaki Tsereteli University, Georgia, and University of Montana, Montana, USA. The students participated in a debate that was run with Loom. This case is presented in section 7.1.6.

7.1.1 Case I: Writing With Kids

This case was run with the Kids' Club (KC) participants. It included a 30-minute presentation of Loom, one hour of story weaving (using Loom), and post activity interviews. The 12 participants were 10 to 14 year old children out of which two were girls and the rest were boys.

The first meeting with the participants was not very encouraging. I gave a presentation of Loom and used a woven story of storylines of Donald Duck[1] comics as an example. I observed that the children were bored rather than enthusiastic. Few questions were asked and most of the time the children sat still.

The story-weaving session was carried out two weeks after the pre-presentation. The children were asked to create a woven story about what they had been doing at the Kids' Club that year. They had been modeling a copper mine with Lego robots during that year and were now supposed to report on all the activities they had done to construct the robots. I had created, together with the instructors of Kids' Club, a base story for the work and that base was supposed to be extended by the children. Since there was not enough computers for all the children, six children were asked to work in pairs, thus there were six children working individually and three groups of two children. In total, there were nine Looms running simultaneously and utilising one WS-Server.

Compared to the pre-presentation, the story-weaving session was a success. I observed that the children enjoyed using the system. It also seemed that they learnt to use the basic functions very rapidly. The children only asked a few questions about how to use the system. When the task was finished, one of the children asked me if he could continue working with Loom. He would have liked to create a new woven story about the Donald Duck storylines.

Children were rather productive during the process, especially when the amount of objects they created was considered. The median amount of sections was six, ranging from four to eleven, and for the edges the median was eight, ranging from five to eleven. The amount of sections was really impressive to me when considering the relatively short time the children had to write the stories. The contents of the sections were not that lengthy, usually a few sentences. The children mainly produced one storyline; only two groups created more than one storyline (they reused some sections). None of the children used sections that others had done. The full results are shown in Table 7.2.

During the process the children suggested new features that they thought would make the system easier to use. The first of these suggestions was to prevent user from moving sections others had done without the permission of the sections author. Some of the children got a bit frustrated when their sections were moved. Also, children suggested that the objects created should be coloured according to the author so that everyone could recognise their own sections by their colour. These same requests were repeated in the post-activity interviews.

It seemed to me that the system was rather easy to use and that children liked

[1]The Disney comic Donald Duck (Aku Ankka in Finnish) is a very popular comic series in Finland.

Table 7.2: The stories created by the Kids' Club participants were rather simple. They usually created just one storypath and did not use any of the sections other participants had written.

N in group	**Sections**	**Edges**	**Storypaths**	**Unlinked sections**	**Connections to other stories**
2	6	7	3	0	0
1	8	8	1	0	0
1	11	11	1	1	0
1	6	7	1	0	0
2	4	5	1	0	0
2	7	8	1	0	0
1	7	8	1	0	0
1	6	6	1	1	0
1	6	9	3	0	0

to work with it. Actually the instructors reported to me that the children had asked if they could continue working with Loom on the next session. The ease of use was also evident in the interviews. None of the children claimed that the software was difficult to use. However, they told me that they thought that the outcome of the process was rather tangled, but they also said that the story could still be read since the edges could be followed.

Altogether I deemed this trial to be a success, especially because it was the first time Loom was put to serious use. Besides gaining new ideas for the development of Loom, I had the opportunity to observe the tool being used by children for the first time. As predicted, reusing the sections others had written proved to be difficult for the children. This difficulty might have been caused by the rather short time that children had to create stories. Furthermore this can be also explained with the fact that it can be rather difficult to use sections that others have written to explain activities these children had done themselves. Also, this could have been explained with the difficulty to find these relevant sections.

Summary of this case is presented in Table 7.3.

7.1.2 Case II: Concept Mapping

The case with Computer Science students was carried out in the context of a course called Theoretical Foundations of Computer Science at the Department of Computer Science at the University of Joensuu. The students were asked to individually create a concept map about the contents of the course with Loom. The task was not compulsory. There were no interviews with the students after this task, but the

Table 7.3: Summary of the Kids' Club case.

Phase	**Notes**
Activity	Children create a woven story about what they had done in Kids' Club over the past year.
Methods	During the activity, I informally observed what went on. After the session children and tutors were interviewed in the same groups in which they worked.
Main Results	Loom was easy to use and quick to learn. It seems that it is difficult to reuse sections others have written.
Ideas	There should be tools to support the reuse of written sections.

students were asked to give written feedback. Only one student out of nine students who performed the task gave the feedback.

Even though there was not that much feedback from the students, this study pointed out to me issues that should be considered when Loom is developed further. I evaluated the task myself, since I was an assistant lecturer on that course. Although accessing the data for evaluation was easy, there could be still a feature in Loom to ease evaluation and assessment of the given task.

With the current implementation I thought that it was rather easy to get an overall view of each of the students' concept maps, but comparing their separate maps was rather difficult. It would have been possible to keep two windows open simultaneously, but that would not have been good approach to the comparing problem. I think it would have been useful to get numerical data from the maps, too. Although analysing the maps quantitatively might not have been the best way to go, it would have enabled me to compare the maps at least on some level.

Another feature that could have been useful would be the ability to create hidden annotations. By hidden annotations I mean comments that are made by a teacher that can only be seen by the teacher, or other teachers. This of course needs much development in terms of user management and permissions, but it would be worth the effort.

Since the main teacher of the course did not have access to the system, she was not able to see the maps. The possibility to print out the maps or save them as images would have enabled me to show the maps to her. I could have printed out the maps and then given them to the teacher. Another option would have been to give her access to the system, but I think it would not have been that beneficial because there was no method to discuss the evaluations. When developing Loom further I think the fact that there can be several teachers on one course should be taken into consideration. Their communication also needs to be organised.

Table 7.4: Summary of the concept mapping case.

Phase	**Notes**
Activity	Students of the course Theoretical Foundations of Computer Science created concept maps with Loom from the contents of the course.
Methods	The resulting concept maps were analysed.
Main Results	Tools for teachers are needed in the system, especially those supporting assessment.
Ideas	Some quantitative data about the stories could be useful for teachers. Also, in cases where several teachers assess the same documents, tools such as hidden annotations could be useful.

Although this study did not provide much information about the usage of Loom, it emphasised the fact that more tools, especially for teachers, are needed. This concept mapping case is summarised in Table 7.4.

7.1.3 Case III: Reporting Results

In this case study the aim was different from the aim of the two previously reported studies. In this case the users of Loom were Forestry students who were supposed to report on their project with Loom. Their project involved creating a forest-growth simulation with a separate tool called PuMe [135].

PuMe is a simulation tool with which the growth of a forest can be simulated. There is an impressive set of parameters that users can set and thus affect the growth of the simulated forest. After the simulation is finished, the software provides extensive data about the different products of the forest. For example, it can be determined how much timber the forest has produced or how much biomass the forest has.

The original type of task that was planned for the students was an open-ended problem. The students were supposed to use the simulation tool in order to find out what kinds of settings would produce optimal results (e.g. the most timber producing). The idea was that the students would have used Loom to store and report the outcomes of their simulations. However, the open-ended problem was changed to a closed problem where the students were supposed to run simulations with pre-determined parameters. This was done because the main teacher of the course suspected that the students would not learn everything that was required by solving an open-ended problem.

When the task changed, aborting the case study was considered. However, I decided to proceed. Instead of using Loom with all the students of the course, there were two groups working with Loom.

Because it first seemed to me that Loom would not be useful in this closed task, the groups were instructed to stop using the software if they found it not to be useful. They were given the instructions to use the software for reporting the results of the simulations.

The groups did not find the tool very useful in this closed task. While they drew trees about the simulations, they could not get any advantage out of them. The other group said that they first did all the simulations and after that they transferred all the data to Loom. The reason for the uselessness of Loom in this task was strongly affected by the nature of the task. Since the task was closed, the students were supposed to perform a predefined amount of simulations and thus could not benefit from the usage of Loom.

Even though the benefit the groups got from Loom in this task was rather poor, they thought it was easy to use and could be useful even in this kind of task. They said that it could be used as a planning tool to keep records of what should be done, what has been done, what are the results and what was the best path. In order to achieve this, Loom could include a feature to change the status of the sections. This would allow the users to keep track of which parts have been already covered and which still need processing. The interviews revealed that the use of Loom with the simulation tools like PuMe could be useful but it requires a more open problem and different planning for the task. The task should have included more processing than manual work, thus enabling better employment of the features of Loom.

This case is summarised in Table 7.5.

Table 7.5: Summary of the reporting case.

Phase	**Notes**
Context	Forestry students used Loom to report results of the PuMe simulations.
Methods	Students were interviewed after the work.
Main Results	Loom is not useful in cases where the task given to the students is too closed. Loom was found easy to use by the participants. Loom could be used for similar work if the task would be more open-ended.
Ideas	Features to track of the status of the sections could be useful in many situations.

7.1.4 Case IV: Woven Strategies

In collaboration with Tampere University of Technology, we carried out a project where a combination of tools for strategy planning was created. The main idea of this project was to create an environment that would support the strategy planning of companies. One of the main features of this environment was a tool called Woven Strategies [78, 98, 99, 109], a spin-off of the Woven Stories tool. The total application collection consisted of three major parts: a questionnaire tool developed at the Tampere University of Technology at Pori, the questionnaire analysis tool and the Woven Strategies tools.

Many theorists argue that computerised systems cannot support the intuitive state in the strategy development process [109]. This case provided evidence that a computer application can support this intuitive stage and that Woven Stories can play an important role in fostering tacit knowledge in strategies.

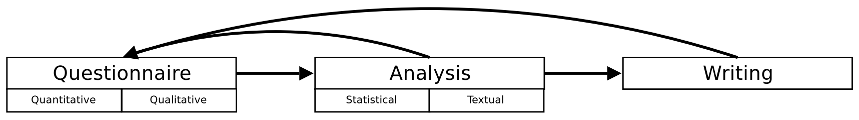
The environment for the strategy planning was based on a database of predefined questions. The idea was that when creating a strategy, the personnel of the company contribute their knowledge and ideas by answering a web questionnaire that the strategy group creates. The questionnaire is constructed from predefined questions from a database.

After the questions are answered, one member of the strategy group analyses the answers to the questions. This was done with a tool that was integrated into the Woven Stories tool as a separate view. The main purpose of the analysis is to create summaries about answers for the whole strategy group so that each member is not required to read all the answers. Each of the analysed questions can then be used as references for the actual Woven Stories document. In my opinion one of the most interesting parts in the analysis tool was the possibility to analyse text-based answers by creating concept maps out of them. The idea emerged from my previous work with concept maps [100].

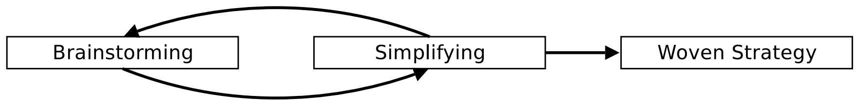
After the analysis, the group starts to write the document based on the ideas they got from the analysis results. The writing process starts with brainstorming and then continues towards a more in-depth processing of the material contributed during the brainstorming phase. As more questions emerge, the strategy group can create new questionnaires. The idea is that the strategy process is going to be more spiral than it has been. The process is presented in Figure 7.1.

Since the main goal of the strategy writing process is to create an unambiguous document, new features, or extensions, were created to the Woven Strategies application to be used in this project. First of all, a feature called *merge nodes* was added. Merging nodes means that two nodes and the links belonging to them are combined. Figure 7.2 visualises this approach.

The merge nodes feature was introduced, since there was a need to simplify the

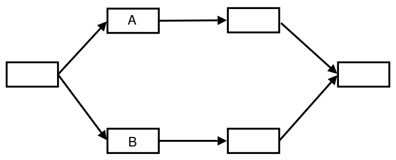


(a) Phases

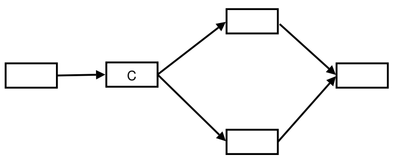


(b) Writing process

Figure 7.1: Phases of the Woven Strategies process and the strategy writing process.



(a) The original story with sections A and B that are very similar in content.



(b) The sections A and B have been combined to one new section C.

Figure 7.2: Merging is useful when a versatile story needs to be simplified. Two sections (A and B) can be merged together to produce one new section (C) containing the combined contents of old sections and the combined edges.

document after the brainstorming and idea presentation phase. Through the merge operation, users can put together two or more sections that share a common idea or topic. Since combining text automatically was too overwhelming a feature to be added, it was discarded as an idea and the combination of text was to be manually created by the person who merges the nodes. The old links are deleted and a new one is created automatically so that it corresponds to the situation as it was before the merge.

The main feature that was introduced in the Woven Strategies was the operation called *split edge*. The idea behind split edge was to ease the process when adding a node in between two sections. Edge splitting, however, is an activity that is inconsistent with the original concept of the Woven Stories, since splitting an edge can break down a path an individual has created. In the context of strategy writing it makes sense since there might be a need to create, for example, a new part of the strategy. Figure 7.3 illustrates the functionality of the split edge operation.

(a) There is the need to place a new section in between sections A and C.

(b) The edge can be split and replaced with two edges and a section

Figure 7.3: Splitting is useful when a new section needs to be created between two, already linked, sections.

Two smaller features that could be usable in the core of Woven Stories were added to the Woven Strategies application. One of these features was the ability to comment on sections and the other was the ability to add references to the sections. The comments are meant to ease the transforming of the sections in a direction desired by the group. Since the aim is to create a sequential document, this feature is needed. The commenting feature lets the group affect the evolution of sections that they have not written by themselves, still preserving the control at the person who originally wrote the section. The references on the other hand let the authors justify the opinions presented in the sections. Since questionnaires play an important role in collecting knowledge in the strategy writing process, the main reference types are the questions and their analysed answers. Also, text-based references are possible, thus allowing authors to introduce references to books, scientific articles and so on.

The main reason to use the Woven Stories as a part of a strategy environment was that it allows a group to contribute new ideas, even to a completed strategy. This way, the strategy process would not have to start from scratch every year, but could be updated every year. The ability to work over the Internet was one of the reasons, too.

The tool described within this section was used and evaluated as a part of Markku

Salo's PhD research. The questionnaire and questionnaire analysis tools were tested in several different cases. The actual Woven Strategy tool that employs the Woven Stories Core with the described extensions was used in two cases. Neither data about the amount of testers nor the names of the companies are available, but for this analysis it was not needed. Despite the fact that the tool was implemented in a rather short period of time and had minor usability problems, it proved to be useful. Salo [110, pp. 187] states that the "Woven Strategy tool showed its power; the strategy process was made much smoother than in previous cases and the time spent on it was shortened considerably". From another case Salo [110, pp. 190] reports that the "Woven Strategy tool was tested thoroughly, and the idea was regarded very good".

Salo's findings are encouraging. The domain where Woven Strategies tool was used is far from the original domain of the Woven Stories concept — story writing. However the concept of Woven Stories and the tool were found to be useful and they also appeared to improve the speed and efficiency of corporate strategy process. The positive effects attributed to Woven Strategies, which is an off-shoot of Woven Stories, is evidence that Woven Stories can be meaningfully applied in a variety of application areas.

This case is summarised in Table 7.6.

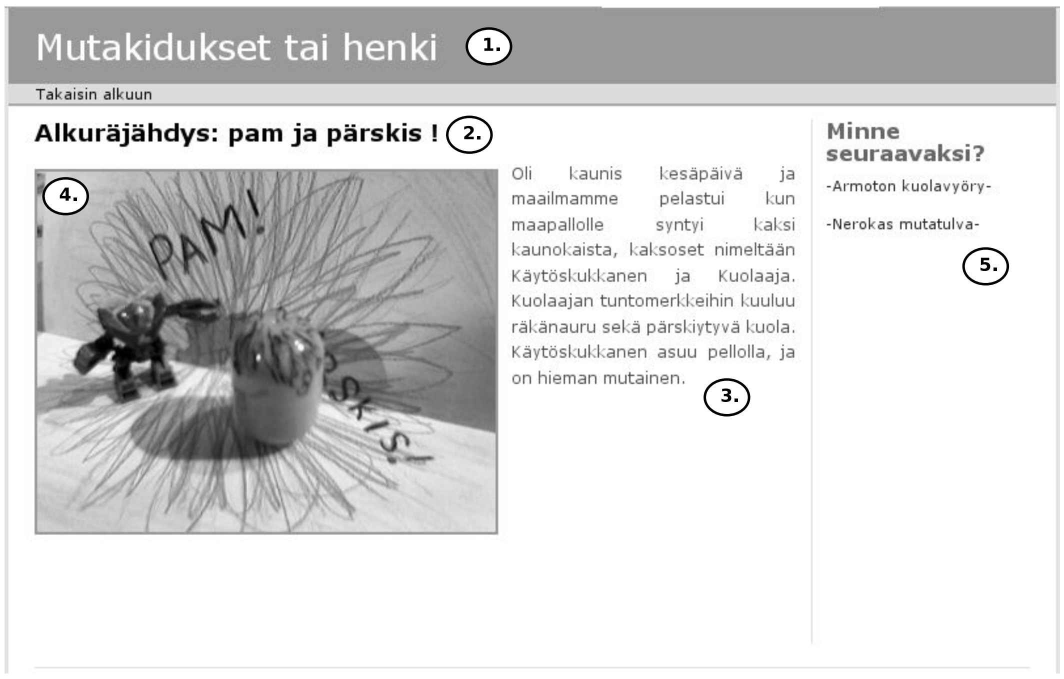
Table 7.6: Summary of the Woven Strategies case.

Phase	**Notes**
Context	Corporate strategy planning.
Methods	Based on PhD thesis of Markku Salo [110].
Main Results	Loom can be efficiently used for corporate strategy planning. It makes planning smoother and more efficient.
Ideas	Extending the application has to be easy.

7.1.5 Case V: Adventure Game Scripting

In several experiments users had problems creating stories that would have alternative story paths. In order to force the users to create these graph-like woven stories, a task that would require alternative story paths was designed. Scripting a text based adventure game seemed appropriate because when writing a game it is natural that the story must have multiple storylines.

This experiment has been carried out twice; at SciFest 2007 [133] at Joensuu and once at "Raatamo", a workshop for juveniles at the library of Joensuu. In each of these cases the task was to write a text-based adventure game within a given



Mutakidukset tai henki

Takaisin alkuun

Alkuräjähdys: pam ja pärskis !

Oli kaunis kesäpäivä ja maailmamme pelastui kun maapallolle syntyi kaksi kaunokaista, kaksoset nimeltään Käytöskukkanen ja Kuolaaja. Kuolaajan tuntomerkkeihin kuuluu räkänauru sekä pärskiytyvä kuola. Käytöskukkanen asuu pellolla, ja on hieman mutainen.

Minne seuraavaksi?

-Armoton kuolavyöry-

-Nerokas mutatulva-

Figure 7.4: An example of an exported section of a story “Mudgills or your life” produced at SciFest 2007.

timeframe. This Section is based on the evaluation of the workshops at SciFest.

For the SciFest workshop new features for Loom were added. Users were able to export the written woven story as a web page. The result of the export operation was a collection of PHP pages where each page contained contents of an individual section of the woven story. An example of an exported section is given in Figure 7.4. Each page contained the title of the story (1 in Fig. 7.4), the title of the current section (2 in Fig. 7.4), the text content of the section (3 in Fig. 7.4), potential media attached to the section (4 in Fig. 7.4) and links to the following sections (5 in Fig. 7.4).

In order to attach images, sound or video to the stories, a separate PHP script was written. This script allowed attaching media files to the exported story and hence to the final text-based adventure game. In order to produce these images each group of students was provided with a Nokia Communicator mobile phone. With the Communicator participants were able to take photos and record sound and video files to be attached to the exported game by using the mentioned PHP script.

The participants in the workshop were 11 to 16 years old visitors of SciFest. The workshop was run during the SciFest altogether four times, and the maximum amount of participants for each workshop was 16. Due to the vast amount of participants and the huge need for computers during SciFest, the participants worked in groups of four using a single computer. Thus, each group produced one game using only one Loom user account.

Each workshop lasted for three hours, during which a short introduction to the task and to the use of Loom was given, and at the end of the workshop all games were presented to the other participants. This allowed participants to work with their game for about two hours.

The introduction to the task was given with an example from the Disney comic book Aku Ankan Taskukirja.[2] In some of the issues of this book there are stories in which the reader can, after reading a certain amount of pages, decide what will happen next. The reader does the selection by jumping to a specified page based on options given. This idea is very close to Woven Stories.

Since the participants of SciFest were rather busy, participating in several workshops, the interviews or any other more time consuming methods would not have been possible. Due to this the selected method for this experiment was observation.

When compared to previous experiences, the participants in the workshop used the tools fluently from the beginning. It seemed that both the selected task and the given example made it easy for the participants to plan a story that finally would have a graph-like structure. The game as an aim also seemed to boost participants' imagination, out of which an example is the game "Mudgills or your life" whose opening section is presented in Figure 7.4. Furthermore, the ability to add media to the finished game seemed to be a particularly positive feature, since participants eagerly planned how to enliven the games with media.

A typical group had first a brief brainstorming about the topic and the title of their game. Then, one or two participants were assigned to construct and write the script of the game, whilst the rest of the group were planning and implementing the media to be attached to the game. Writing usually proceeded in such a way that the structure was created first, and then text was appended; thus resembling the typical writing process presented in section 4.2.2.

During all the workshops none of the participants reported problems with understanding the structure of the game that their group, or any other group, had written. Furthermore, the approach of using sections and edges for forming the stories was, again, natural for the participants. Even occasional visitors that popped in during the workshop had no problems to understand the flow and the structure of the scripts. I consider this to be solid proof that the concept, and especially the graphical visualisation of the stories, is indeed valuable for presenting and editing a complex document. The workshop ran at Raatamo confirmed these results.

[2]Donald Duck's pocket book

Table 7.7: Scripts in this case were a major improvement to previous cases.

Group																**Σ**
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	
Sections																
18	8	16	5	17	3	8	10	6	5	13	13	6	4	3	12	**147**
Edges																
18	6	17	4	25	2	9	15	6	4	16	16	6	3	2	17	**166**
Storypaths																
10	3	9	2	∞	1	4	∞	2	2	10	12	4	1	1	25	**86***
Selections																
8	2	7	1	5	0	3	5	1	1	8	7	2	0	0	7	**57**
Unused Sections																
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**

Altogether these sixteen groups produced 147 sections and 166 edges. All numbers are shown in Table 7.7. Note that the amount of storypaths does not hold, since two of the scripts written included loops. The loops are storypaths that eventually return to some part already used, and thus make calculating the exact amount of paths impossible.

As can be seen from Table 7.7 there are several stories with large amounts of optional storypaths. These are due to effective use of edges in the process. An example of how to achieve multiple storypaths is given in Figure 7.5. When compared to the number of storypaths in Case I, there is a big improvement in stories. Since the cases were rather similar in terms of participants, introduction to Loom and time for the work, the only variable that is left is the task given to the participants. Due to this, it seems to be important to form the tasks in such ways that naturally guide the users to think about optional storypaths.

The results of the workshops point out the importance of task formation. The task has to be meaningful and it should be formed in such a way that it naturally elicits structured thinking among the users. This seems to make it easier for the users to write documents with alternative storylines, and thus to utilise the full potential of Woven Stories.

The evaluation of the concept of the feasibility of the concept of Woven Stories was easy in these workshops. This seemed to be true for several reasons. Most important reason was that the students were using the software from only one computer. This approach removed all the bias and disturbance caused by the other non-conceptual related features. Furthermore, the fact that the participants were enthusiastic about the task made the evaluation smoother. The test proved that the concept of Woven Stories is really powerful when producing structured documents.

This case is summarised in Table 7.8.

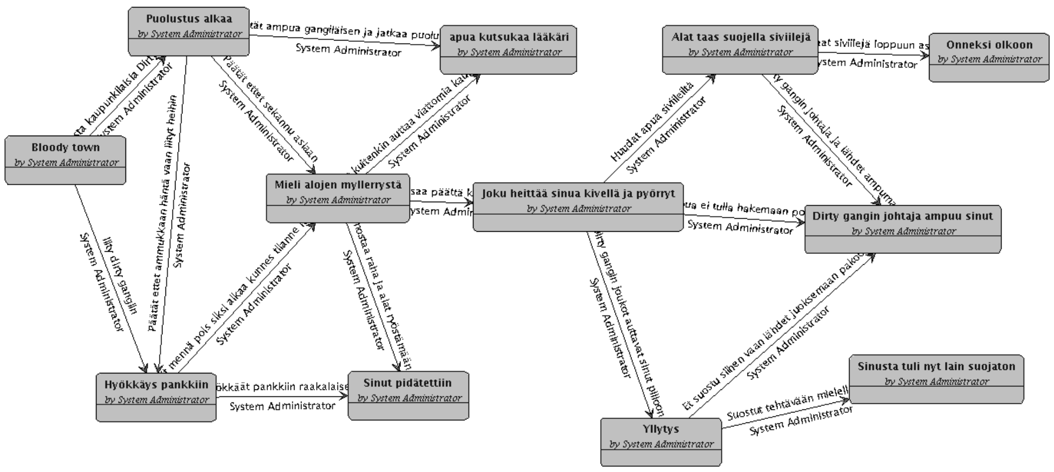


Figure 7.5: An example of a game script in Loom.

Table 7.8: Summary of the Adventure Game Scripting case.

Phase	**Notes**
Context	SciFest visitors writing and creating a script for a text-based adventure game.
Methods	Participants were observed during the work.
Main Results	Loom is useful in cases with open, meaningful tasks. Graphical visualisation of the stories is good approach for presenting and editing a complex document.
Ideas	Possibility to publish or export the document easier would be useful.

7.1.6 Case VI: Debates

In this case university students from the University of Joensuu (Finland), the University of Montana (United States) and the Akaki Tseretely State University (Georgia) participated in an online debate that was run with Loom. Altogether 38 students participated the debates. There were two rounds of debate and each round lasted for two weeks. Each student was required to post one argument supporting the debate statement and one rebuttal argument. In addition, each student was required to post an argument that commented another student's argument. After each round of debate an electronic questionnaire was sent to the students. This analysis is based on the first questionnaire, to which total of 19 students answered. Exact numbers of participating students and answers is shown on Table 7.9. The topic of the first debate was "Globalization through the use of information and communication technologies is creating cultural homogenisation along Western Lines". This case is presented in more detail on [96].

Table 7.9: Participants in the debate case.

University	**Major**	**No: of participants**	**No: of respondents**
University of Joensuu, Finland	Computer Science	25	14
University of Montana, United States	varying e.g.: Communications Political Studies	9	2
Akaki Tseretely University, Georgia	American Studies	4	3

The debate proceeded smoothly, although started rather slowly. However, by the end of the debate the document had grown to be rather big, as can be seen in Figure 7.6. Students seemed to enjoy using Loom for debating.

> Student C: *Simple, quick to learn, creates a good overall picture of a debate.*

The students who participated in the debate were not that experienced in the area of educational debating. Only two of the participants could be described experienced in the art of educational debating. Despite this, the debates went on well, and the process and the final result resembled a debate. Fifteen respondents said that Loom is useful tool for online debating.

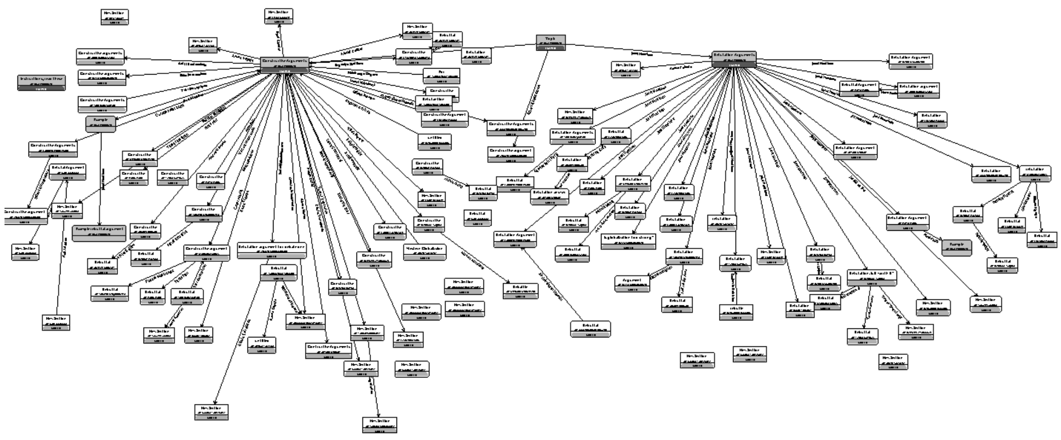


Figure 7.6: The debate document after first round of debates.

The visual representation of the story (or debate in this case) was thought to be "quite good". Fourteen students thought that it was easy to follow the debate in Loom. Even four of those five students who generally did not like Loom said that following the debate was easy. Generally, adding new arguments and linking the argument to each other were found to be easy.

Negative feedback related mostly to the navigation within the document. Several students complained that the document grew too big in order to find relevant sections from the document. This defect is observable on Figure 7.6. This could have been resolved either by limiting the amount of debaters, or by utilising information retrieval methods (see Section 6.6).

> Student B: *Its a nice way to see the everyone's debate. But I don't know. If the number of people was a little big larger, it's gonna be a mess.*
>
> Student C: *If there are a lot of arguments from which only few are interesting, it might be hard to find those good ones.*

Most of the students participated in the debate by only providing the comments that they were required to post. Only couple of students provided additional comments. Some students expressed the wish that there had been more activity:

> Student A: *It felt more like people just adding their 2 cents. Perhaps this may be because of the inexperience of the participants.*

As in the previous cases this result highlights the fact that the task was too closed. Students had certain goals to meet and after fulfilling them, there was no need to contribute to the debate anymore. Furthermore, this forced the students to generate arguments that might not have been natural to them. Due to these reasons, the resulting document had a tree-like structure as observable in Figure 7.6. It seems likely that if there would have been more time for the debates, the structure would have grown to become more like a graph than a tree.

One surprising finding was that some of the students were using also other tools than Loom whilst debating. Most likely this was due to the fact that most of the participating students were not native in English language, and thus they used word processing tools such as Word to write and spellcheck their arguments. This was reported by the students since there were problems with the copy and paste features of Loom.

Even though the communication features of Loom are not that sophisticated it seems that these were adequate enough to create a feeling of a community for the students. For example, few Computer Science students were worried about the ability of the other students to use Loom. However, several students mentioned that they would have preferred to be provided with more awareness information about the other participants.

This case was the only case, reported in this thesis, which included spatially distributed collaboration between participants. It proved that Loom, and Woven Stories in general, can be used as a debating tool. Furthermore, it validated that there is a need for a tool to provide awareness information. None of the students complained about the communication related features of Loom, and thus it seems that at least in this type of task these features are not that important.

The strength of the Woven Stories for debating lies in the visual representation of the document structure. Another advantage is that it is possible to link an argument to several previous arguments (and thus base a new argument on existing points). This is something that can not be achieved with traditional conversation and collaboration tools such as forums or wikis. Naturally the fact that Loom and Woven Stories can be used for many other kinds of tasks as well makes it a good selection for debating. This case is summarised in Table 7.10.

7.2 Lessons Learnt from the Case Studies

The evaluation of the concept of the Woven Stories, and the application based on it has not been easy. This has been due to mistakes done in the implementation process, which made it hard to concentrate on the issues I was willing to test and evaluate. This section covers all these problems and issues. This description is an important base for the Nucleus Model, a proposed design approach for social mindtools, presented in Chapter 8.

Table 7.10: Summary of the Debating case.

Phase	**Notes**
Context	University students engaging in a debate with Loom.
Methods	Resulting document was analysed. Participants were asked to fill in a questionnaire after debate. Answers of the questionnaire were analysed.
Main Results	Loom is useful for debating. The graphical visualisation of the document makes it easy to follow the flow of the debate. It is hard to find relevant content from big documents.
Ideas	Need to have a feature to find relevant content.

This section is based on retrospective analysis of the design and implementation process of Loom and evaluations presented in Section 7.1. The aim has been to identify how the decisions made during the implementation affected the results of the evaluation. Furthermore, the execution and results of the evaluation were analysed in order to see what could have been done better. All results of the analysis of the six cases are analysed as one case. All this was done in contrast to what was learnt from the cases, experiences from the design and implementation and most importantly, what was known at the time the analysis was conducted. For example, AJAX had gained popularity as an implementation approach and made it possible to implement WS@Web (see Section 6.5). Looking back to the cases, and knowing the results from all cases reveals a number of recurring issues. Furthermore, based on this analysis it is possible to identify issues that are important in the process of developing applications based on the concept of Woven Stories.

During the first case study I noticed that it was rather hard to concentrate on the exact issues of interest in the evaluation. The primary interest was on the concept of Woven Stories. However, the participants were distracted with the collaborational features of the applications, colours and other, from my viewpoint, rather minor issues. This has been evident in several cases. The reason behind this is most likely the fact that the application was implemented too far before it was taken to evaluation. This led to the situation where users had too much cognitive load during the work. Furthermore, since the application was already implemented quite far, it was hard to start designing and implementing new features.

Based on the above, it seems that the application should have been developed in very small pieces, starting from the very core of the application and then the other features could have been added step by step. This would have allowed an evaluation version of the application that included only those features that were critical from the viewpoint of the concept of the Woven Stories.

Luckily, it has been possible to evaluate the functionality of the concept in other case studies, namely the adventure game scripting case and the case with the forestry students. In these cases participants used the application from only one computer, and thus the collaborational aspects were not distracting them. Thus, it seems to be vital that during the first evaluations only very few users are included in the evaluation process and that these users are located in same physical space.

In the reporting results case and the debating case the task of the students was too closed. This forced the students to focus on the very topic of the task, but at the same time it made the actual process of using the Woven Stories rather straightforward and thus, perhaps unclear for the users why Woven Stories was used. Based on the experiences from cases I and V, Woven Stories seems to stimulate imagination and creation, and thus these more open tasks suit it better. Based on this, it seems that the power of Woven Stories can be fully utilised in open-ended, meaningful tasks.

Whilst the concept of the Woven Stories has been carefully evaluated, the layers of Awareness and Communication have proved to be difficult to evaluate. Due to these difficulties it is still not clear what kinds of features related to awareness a Woven Stories application should have. This has been due to two reasons. Either the application has been tested in cases where the users have been constructing the document from only one computer (forestry and adventure games) or the case has been carried out in a "real" setting, where each user has been using the application from their own computers (debating, concept mapping). This has made it difficult to analyse how the users actually refer to the sections and edges and how they really would like to be able to refer them whilst using the application. Furthermore, since most of the activities have been asynchronous, there have not been many genuine discussions while the users have been using the application. This could have been solved by evaluating the application in one classroom in such a manner that the users could not have seen each others screens, but could have emerged in discussion. Actually, this was the case in the Kids' Club case, but it was not understood at that point that it would have been important to analyse these discussions as well.

During the implementation process of Loom one of the aims was to keep the application simple and extendable. Even though it is relatively easy to extend the application to transfer different kinds of data between client and server, the actual implementation of the extending features is too difficult. In particular, the integration of the extensions to the client needs lots of manual work. There is no possibility to plug-and-play an extension if needed. Due to this, Woven Strategies (see Section 7.1.4) was implemented as a separate application. The implementation strategy should have been considered more carefully at the beginning of the design and implementation process. Again, this was one of the reasons to start experimenting with WS@Web.

The lessons learnt during the evaluation of Loom have played an important role in the formulation of the Nucleus Model. Without the mistakes found in the design, implementation and evaluation of Loom, it would have been difficult to define the approach for the development of the social mindtools. Due to this Chapter 8 is strongly influenced by these lessons learnt. Table 7.11 summarises the lessons learnt from the cases.

Table 7.11: Lessons learnt from the case studies.

Case	Notes
I	• In order to concentrate on the concept of the tool, no other features should be included in the tool. • First test should happen in a classroom where all users are located. • Needs for awareness related features can be identified by observing the users.
II	• Provide teachers with features that makes the tool meaningful for them. • Data has to be easily exported in various formats.
III	• The task has to be meaningful for the users.
IV	• Application must be easily extendable.
V	• Meaningful tasks motivate users. • Using the application in context where no awareness / communication related features are needed eases the evaluation of the concept. • Data has to be easily exported.
VI	• Awareness in important. • The task has to be meaningful for the users.

7.3 Comparison of Woven Stories and Wikis

Wikis provide an example of commonly used and openly available collaborative knowledge creation applications. Wikis and Woven Stories also share the fundamental idea to collaboratively construct and edit hypertext, but at the same time they are rather different in terms of representing knowledge. Wikis can be regarded as a media which supports learning due to their ability to facilitate collaboration [37] and thus have been set high expectations especially as a support mechanism for collaborative learning. However, several shortcomings in the usability of wikis have been identified [41, 27].

This comparison presented in this section shows how Woven Stories can overcome many of the limitations of wikis. Woven Stories and wikis are compared in terms of ten attributes represented in Table 7.12. These attributes have emerged from this analysis, but are strongly related to the characteristics of mindtools (see Table 4.7) and social mindtools (see Table 4.6), namely accessibility (A), engagement (E), multi-purpose utility (M) and usability (U). The characteristic related to each attribute is shown in parenthesis after the attribute name. Attributes are divided into five categories that compile closely related attributes together.

The first attribute, accessibility, comes directly from the requirements of social mindtools. Both Woven Stories and wikis are freely available. However, the drawback of Loom is that it is based on Java and thus needs an installed program for users. However, the new WS@Web application solves this problem. The next attribute, purpose, describes the expected function of the tools. In wikis this is to produce all kinds of documents, but the Woven Stories is meant for writing documents where the structure of the document is important. A typical product of Woven Stories is a collection of a student group's variants of a given folklore tale.

Two attributes are related to the usage of these tools. Representation describes how the information stored by the tool is shown for the users. In wikis the representation is a somewhat traditional webpage, while Woven Stories visualises the structure of the document as a graph in the user interface. Thus Woven Stories emphasises the meaning of the structure even in the user interface. Synchronicity refers to which extent the application allows simultaneous usage.

Three attributes are related to the structural aspects of the documents. Attribute structure refers to the structure of the document as well as to the linking between different parts of the document. Due to the plain linking system of Woven Stories, the structure of the documents remains simple. In wikis, however, the links can be placed anywhere in the text allowing extensive additional data and vast possibilities for traversing the document at the expense of comprehensible structure. Due to the simple structure of Woven Stories, it is possible to visualise the topology of the document for the users in the user interface. In wikis, the topology could be visualised with proper scripts based on the wiki links, but due to the complex nature of the structure, it is normally hidden from the user. Maintenance of topology then refers to how the topology and the structure of the documents are maintained in these systems. In Woven Stories the topology is maintained in the graphical user interface by adding boxes and links. Thus, the maintenance is a natural part of the

Table 7.12: Comparison between Woven Stories and Wikis.

Category	**Attribute**	**Woven Stories**	**Wiki**
Accessibility	Accessibility (A)	• available for free	• several implementations available for free
Purpose	Purpose (M)	• to write a document where the structure is important (eg stories)	• to write any document
Usage	Representation (U)	• the structure of the document is visualised as a graph	• no visualisation available • documents are hypertext pages
	Synchronicity (E,U)	• semisynchronous: allows both synchronous and asynchronous activities	• asynchronous
Structural Aspects	Structure (U)	• simple, sections form linear ”storypaths” • links represent the order of the storyline	• complex, links form nonlinear paths • links are independent of the order of the storyline
	Topology (U)	• visible in user interface	• hidden from the user
	Maintenance of Topology (U)	• easy to maintain • graphical maintenance	• typing errors and similar problems can make it difficult • sometimes need for transition diagrams (eg before writing stories)
Ownership and Access	Ownership (E,U)	• everyone can create new storylines (boxes with links) • only the creator can edit his/her storylines (individual boxes)	• everyone can create new content and links • everyone can edit all content
	Tolerance (E,U)	• low threshold to add new data • no need to change existing content	• high threshold to add new data, especially for novices • often need to edit existing content
	Conflicts (E,U)	• due to organised ownership no conflicts are possible	• possibility for edit collisions • possibility of edit wars

work flow. In wikis the topology is maintained by adding link tags to the document text, which can cause problems, for example due to typing errors [41]. Similarly in the study of Désilets et al. [41] the users drew a state transition diagram of their story before writing it to the wiki; with Woven Stories this would not have been necessary.

The last three attributes are related to the ownership and access of the documents. Ownership refers to the role of the creator versus user and browser; that is, who is in control of the contents stored in the application. In wikis everyone has a full control over all the data. Anyone can add new content and edit existing content. In Woven Stories, however, anyone can add new content, but only the original author can modify existing data. Ownership also relates to the tolerance of the system, by which we mean how easy it is for novices to add new data to the system. Research has reported that especially novice users of wikis are reluctant to make drastic changes when starting to use Wikipedia [27], and we do believe that this holds in all wiki use cases. Woven Stories requires the users to add new content instead of changing existing content. In this way the collaborating group can decide which pieces of alternative information are important. Furthermore the restricted access to content - the approach used in Woven Stories - reduces conflicts, such as edit collisions [41] and edit wars [27].

Table 7.12 summarises the differences between Woven Stories and wikis. Furthermore, Table 7.12 presents a classification which can be used to evaluate text-oriented social mindtools. It is not surprising that most of the attributes which emerged in this analysis are related to characteristics of engagement and usability, it is highly important that these types of tools can efficiently be used for collaborative knowledge construction, that they engage users for knowledge contributing and that they are easy to learn and use. Based on this analysis, we claim that Woven Stories can help to overcome the shortcomings of wikis especially in situations where the structure of the document produced and the thinking which it captures is important. Similarly, the approach used in Woven Stories can lower the threshold for novices to start contributing knowledge.

When compared to other text-oriented Web 2.0 tools, such as GoogleDocs or Blogs, Woven Stories offers a totally different approach to writing documents. GoogleDocs is just another collaborative text editor, while in contrast Woven Stories allows the users to create, see and edit a shared document from a totally different perspective, in a graphical way. Furthermore, due to the graphical representation of the structure, Woven Stories based applications are highly generalisable for various application areas.

7.4 Summary

The fourth research question of this thesis asks what kinds of learning tasks does Woven Stories support. In order to evaluate this, Loom was utilised in six different case studies.

In all the use cases the feedback from the users was positive. The strong focus on visual representation of the document structure was thought to be useful and important. The basic usage of Loom is easy to learn and thus easy to utilise for various tasks.

Based on the cases it can be summarised that Woven Stories gives the best results when used in cases where the knowledge generated and processed is based on strong relationships. The knowledge can have, for example, time-series based relationships as in the adventure game case (see section 7.1.5), or content based relationships like in the debating case (see section 7.1.6). These relationships enable the users to efficiently use the graphical presentation of the stories.

The tasks given to the users of Woven Stories should be meaningful and open ended. Tasks should be planned in ways that naturally can benefit from the structural representation of knowledge. Furthermore, the knowledge processed should be related to what the learners already know, in order to promote meaningful learning. By providing open ended tasks the processing can lead to knowledge building.

Based on the cases and comments received from participants, it seems that Woven Stories promotes creative thinking. Thus, it can be a valid tool for creative writing, of which the adventure game scripting case is an example. In contrast, writing stories based on real life can be problematic if there are no obvious connecting points between various stories.

Above all, the ecological validity of Woven Stories and Loom seems to be good. It can be used in real settings and for real tasks. Whether it improves the quality of learning, is an open question needed to be analysed by educationalists. However, students seem to enjoy using it, and this can be valuable source for motivation if the tasks are properly designed.

Chapter 8

Nucleus Model for Designing Social Mindtools

"Simplicity–the art of maximising the amount of work not done–is essential. "
– Agile Manifesto [1]

Literature on developing computer applications contains numerous methods and approaches for software design and implementation. In the context of social mindtools the problem with these methods is that these do not give any viewpoints about the order of development, but concentrate more on e.g. how to benefit from end users in the development stage or how to carry out the actual implementation process. Most, if not all, of these methods assume that the task of the application, the goal, is known before the development. This, however, is problematic in the context of social mindtools, since these can be used for various tasks and subject matters and due to this it is difficult to define what the finished application will look like.

In order to simplify the process of developing social mindtools, the framework presented in Section 4.8, the architecture of Woven Stories (see chapter 5) and the experiences from implementing and evaluating Loom have been used to derive a three stage model to be used as a basis for the design and implementation. This model is called the *Nucleus Model* and it is based on the idea to develop a minimal set of features from all the levels of the framework for social mindtools in order to produce a usable and meaningful version of the software early in the development process. Furthermore, following the model makes it easier to focus on the current issues in the development by giving the users a meaningful version of the application to be tested without distracting features.

8.1 Introduction

The Nucleus Model divides the development and research process to three stages; *Concept*, *Awareness* and *Communication.* In the concept stage the features related to the actual concept of the social mindtool, the main idea of the application, are to be implemented. After this the development is continued with awareness related features and finally, features related to communication are added to the application. The model is presented in Figure 8.1.

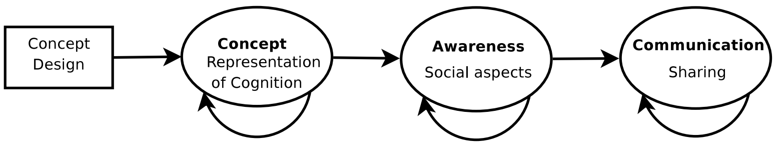


Figure 8.1: Flow of Nucleus Model.

Before the actual implementation process is started, the research group should focus on the *concept design.* In this process, the group defines an abstraction that outlines what people can do with a product and what concepts are needed to understand how to interact with it [114, p. 51]. Furthermore, the application area for the social mindtool will be analysed and defined by the concept in such detail that implementation can be started. If possible and applicable, it is worthwhile to test the concept manually (see example in Section 3.1.3) in order to see whether the concept is valuable enough to be implemented as a computer application. This can be done by utilising different prototypes [114, p. 530]. The criteria for the valuableness of the application are strongly dependent on the context of the application, but generally the requirements of mindtools (see Table 4.1) and social mindtools (see Table 4.6) can be used as the basis of the criteria.

After the concept has been designed, defined and tested in such detail that it satisfies the research group, the actual implementation can be started. Each of the stages of the Nucleus Model is implemented in the order presented in Figure 8.1. Furthermore, during each stage the research group shall carry out iterations of implementation and evaluation, in order to improve the features related to the current stage. Thus, the process is somewhat similar to *User Centred Approach* (see e.g. [114, pp. 425] and [80]) which is defined as *the active involvement of users for a clear understanding of user and task requirements, iterative design and evaluation, and a multi-disciplinary approach* [80].

In the context of the design and development of social mindtools the multi-disciplinary group should include professionals from both the fields of Computer Science and Education. This will make sure that both technical and educational aspects are considered efficiently and in enough detail.

Even though Figure 8.1 presents the Nucleus Model in such a way that it closely resembles the waterfall model [114, p. 449], the Nucleus Model is iterative by its nature. It is also possible, even encouraged, to step back to earlier stages if, for example, in the awareness stage it is noticed that some of the features of the concept need to be improved. The process has to be flexible and adjustable, still producing usable versions of the tool within short intervals, thus having similarities to Agile Development Methods [114, p. 454].

The benefits of following the Nucleus Model include:

- the model guides the developers to be devoted to the core idea of the application they are developing,
- it emphasises the importance of keeping the concept and application simple,
- it defines a clear order in which the features are to be implemented,
- a meaningful version of the application can be tested at an early stage of the development, and
- it defines a natural interface for multidisciplinary research, namely between Computer Science and Education.

This process is described in detail in the following sections.

8.2 Concept

The concept is the essence of a social mindtool. It is the fundamental part of the application and makes it different from other applications. Concept defines what the social mindtool does and what it is used for.

In order to define the concept of the application, the question that needs to be asked is:

what is the minimum set of features with which the tool can be used?

By answering this question the development team is able to define the features needed in the very core of the application. This core includes the features that are needed in order to be able to use the application in the way it is intended to be used. However, in the case of social mindtools, this task is not straightforward since these tools need to be generalisable and applicable on various tasks. Due to this, a well defined answer to this question is likely impossible. However, a rough estimate will do at this stage. An example of a minimal set of features for Woven Stories application is presented in Table 5.1.

The reason for this minimal approach is based on the case studies reported in Chapter 7. During the tests it was found to be difficult to concentrate on evaluating a specific set of features. If the only features to evaluate are the ones implemented focus is easier.

Unfortunately there is only one good way to evaluate what is a good concept for a computer application: testing it. A test can be executed in two ways, either the concept is table tested, or prototype application is implemented.

Since table testing (e.g. with paper prototypes [21]) is cheaper and faster to get started, it is generally a preferred way to test the concept. An example of such a “manual test” is given in Section 3.1.3. However, some concepts are impossible to table test and thus a prototype version of the concept, “a proof of concept” version of the application, needs to be implemented.

In the context of social mindtools, the minimum requirement is the ability to collaborate. Due to this, the application needs to have a backbone that allows transmission of data between several clients. It is a good idea to utilise some existing applications and implementations for this purpose. Loom was built from scratch and this took a lot of time. Since a social mindtool is preferably web based (see Table 4.6), an AJAX[1] or other such approach is recommended. By utilising JavaScript libraries (see e.g. [106, 49, 71]) and JSON [3] it is relatively easy to get a client running in a web-browser to communicate with a server implemented, for example, with PHP [130].

In order to clarify the idea of the concept an example is given. The development group is planning a collaborative drawing tool. The first requirement is naturally the ability to share data between clients. The second requirement would be the ability to draw something on a common surface. Thus, there is a need for a common drawing area and a tool with which user can draw a pixel to that area. Drawing one pixel should be enough at first step, since it can be easily used when implementing other features of the tool.

The concept of a new social mindtool is the stage of the Nucleus Model where few guidelines can be given. However, the best guidance is the definition and framework of social mindtools presented in Section 4.8.

After the concept has been tested and proved to be worth implementing, the first round of implementation is started. The first thing is to do an extensive background study in order to map potentially similar existing applications, and applications that could be used as a base for this new social mindtool. At this point all the potential implementation techniques are considered and analysed.

One of the first things is to design the basic user interface. The application can be based on the WYSIWIS approach or relaxed WYSIWIS. Furthermore, the

[1]Asynchronous JavaScript and XML

concept sets the requirements for the level of synchronicity. Also, the amount and type of data needed to transmit between the clients affects the design.

At this point following things should be defined:

- the minimum set of features for the concept,
- the implementation techniques, and
- the user interface design.

Based on these, the first version can be implemented. During implementation it is important to bear in mind that the application will most likely change considerably. Furthermore, it is likely that there will be several new features added in future. Due to this, the application needs to be designed in such a way that it can be easily be extended and modified in later stages of the development. Generally it is a good idea to provide the data transmitting and storing in a way that is as flexible as possible. Section 8.5 discusses extensions and implementing them in more detail.

During the implementation in the concept stage it is important to keep in mind that nothing except the concept features are to be implemented. This includes features that support collaboration (and work in general) as specified in Table 4.8 and Table 4.9. This will make it easier to analyse the features related to the concept when testing the application with users. Furthermore, the feedback from users will provide such information that will guide the design and implementation of these other features when it is time for them.

After the first version of the application at concept stage is ready, it is time to execute the first evaluation of the tool. It is important to point out that the first prototype should be available for testing immediately, but at the same time the application should be used for a meaningful task. The task should be organised in a classroom with enough computers for each participant. It is a good idea to locate the participants in a way that they can easily see each others' screens. While the participants work with the application, the researchers should concentrate on the following questions:

- (CQ1) *how does the concept of the application work?*,
- (CQ2) *what do they look from each others' screens?*, and
- (CQ3) *how do the participants communicate during work?*

Question CQ1 is related to the concept related features, (i.e. what has already been implemented?) and questions CQ2 and CQ3 are related to the features to be implemented in future. Naturally, in early concept stage it is important to concentrate on CQ1.

There are several characteristics in the concept stage. The evaluations performed in this stage should mostly concentrate on the concept itself. Since the application will be a social mindtool, the evaluation will require a group of users using the tool at the same time. This can easily be done in a computer classroom where users can work together. It is also important to track the details related to awareness and communication. For example, the researchers can study how the users refer to objects on the screen and what kinds of methods they use to communicate with each other.

At this point of the design it is also important to have a close collaboration with educationalists. The workload should be divided in a way that allows the computer scientist to concentrate on the application, its usability and implementation and the responsibility of the educationalist is to concentrate on the actual utility and improvement of the educational value of the application. It should be also emphasised that at this point, if it seems evident that the application does not work, it is wise to return to the drawing table and redesign the application, or even dump the concept and look for a better one.

Based on the results got from the testing, the development group proceeds with the implementation. The aim is to improve the concept related features and iterate the design, implementation and evaluation process until research group agrees that they cannot achieve more for the concept related features. During the last tests on the concept stage it makes sense to concentrate more on question CQ2 in order to get data for the planning of the next stage.

The following list summarises the main principles applied in this stage.

- Implement a minimal set of features to test the concept.
- Test and evaluate by following the development research approach [107].
- Iterate by adding features and reshaping existing features until the concept related features work.
- During this stage, evaluate/analyse features needed for awareness. For example, if implementing a synchronous tool and the test are carried out in classroom, pay attention to how users refer to objects on their screen, how to point them out and how they generally communicate.
- Based on the evaluation data, plan the features related to awareness.

The concept stage continues and iterates until the researchers are satisfied with the concept related features. This can take several iterations of design, implementation and evaluation. The acceptance criteria differ depending on the concept being develop. However, good indicators of moving to the next stage are, for example:

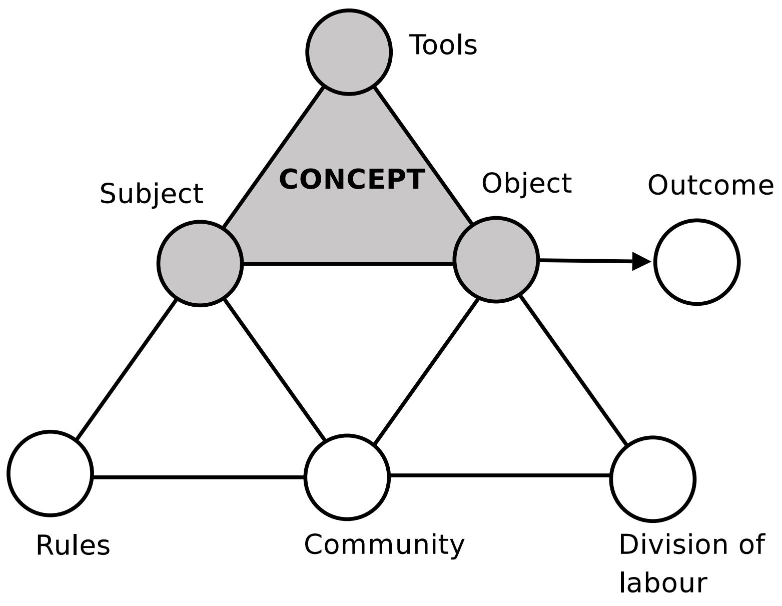


Figure 8.2: Relation between Activity Theory and concept stage of Nucleus Model.

- Users are able to use the software without asking help from developers
- Users learn the features of the tool quickly, and
- Users can easily produce what they want with the tool

Jonassen's list of mindtool features presented in Section 4.1 can be used when determining the acceptance criteria for this stage.

In terms of activity theory (see section 4.5.2), during the concept stage the focus is on subject-tool-object relationships. Since the features related to awareness and communication are still missing, the lower part of extended Activity Theory model is neglected at this point. This is presented in Figure 8.2. Even though the social mindtool at this stage implements collaborational aspects, it still hides the actual community. Due to this, it is easier to concentrate on evaluating subjects using the tool while working towards their object.

8.3 Awareness

Based on the literature and the case studies presented in this thesis, awareness plays an important role in social mindtools. Loom provides only a couple of features to support awareness. More should be provided. However, the problem with these kinds of features is that they are hard to evaluate in normal use cases. Furthermore, the concept of the application being developed sets certain requirements for these features that cannot be known before users have been observed using the application.

In order to support awareness in applications, the application should be able to answer “who, what, where, when, and how” questions [55]. The degree and quality of the required awareness information is provided depends strongly on the application being developed and its target domain. As an example of applications that maintain a reasonable level of awareness, there are instant messaging applications based on the MSN protocol such as Windows Live Messenger [84] and Kopete [129]. These tools provide information about user status, such as whether they are available for communication or not. Secondly, they provide awareness information also during communication, for example by notifying when a person at the other end is writing a message.

Before the implementation in the awareness stage, the concept stage should be completed in such a detail that it shall not disturb the implementation and evaluation in the awareness stage.

During this stage the implementation should concentrate on the features related to awareness. The requirements for these can be derived from the evaluation at the concept stage, by observing and analysing what users are pointing from each others screens and what kinds of information they request from each other about their work and so on. Furthermore, Table 4.4 provides a list of the elements of workspace awareness. From information in this table and the aspect of awareness from Table 4.8 and Table 4.9, the features needed for awareness can be designed.

At this stage of the Nucleus Model the evaluation can still be carried out conveniently in a classroom, within a setting where users can talk to each other but cannot see each others computer screens. A good indicator for the functionality of awareness is if the users do not need to go and see each others computers screens or do not need to ask from the other users what they are currently doing.

Even though collaboration needs communication, awareness is even more important. In order to communicate the users have to know if there is someone present with whom they can communicate. They need to know what others are doing and whether they are available for communication. In the following, I provide a list of main principles for this stage.

- Implement features related to awareness.
- Allow user to see what others are doing, what they have done and to which objects they refer while communicating.
- Evaluate by, for example, having the test in classroom in such way that users cannot see others displays. If awareness works, they should be able to work

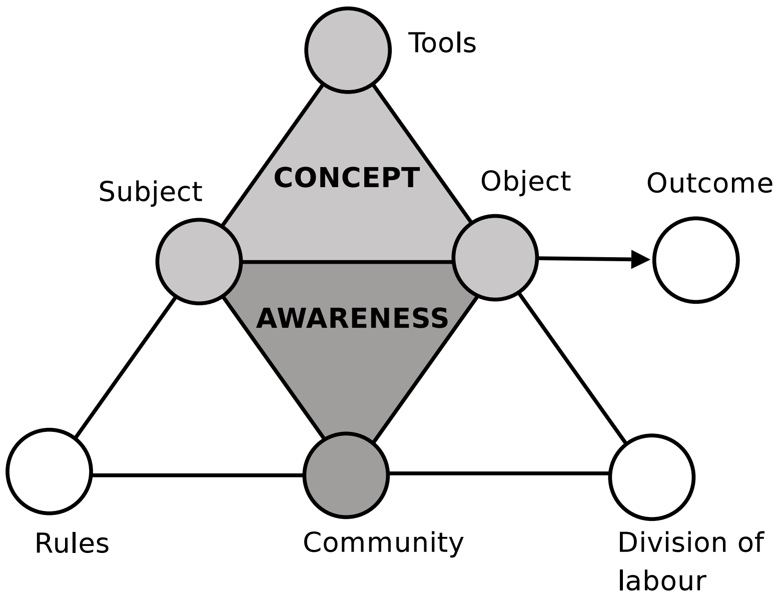


Figure 8.3: Relation between Activity Theory and awareness stage of Nucleus Model.

well without seeing others screens.

- During evaluation, analyse the needs for communication. This data can be used to design communication related features.
- Pay attention also to the concept. Does it still work? Is there need for modifications?

From the point of view of activity theory, the awareness stage brings in the community. The users of the application are not able to communicate with each other, but become aware of other users and their actions. This makes the usage of the collaborational tool more natural. The relationship between the extended Activity Theory model and the Nucleus Model is presented in Figure 8.3.

8.4 Communication

At this point it is in the interest of the development group to implement features that allow the users of the application to communicate with each other. Furthermore, in order to support communication better, there might be a need for features that allow the users to, for example, point to some objects on the screen and hence provide *conversational props* (see [24]).

In certain types of applications, it is possible that communication related features are not even needed. For example in the debating case, discussed in Section 7.1.6, the participants did not miss any features for communication. This was most likely due to the asynchronous nature of the task.

The requirements for the communication are strongly dependant on the context and concept of the application being developed. Due to this, no specific guidelines can be given. However, based on the evaluations and observations on previous stages the developers should already have a clear image of what kind of communication related features are needed in the application. Also, the level of synchronicity of the application affects the selection of the communication method. In semi-synchronous applications the users should be able to also use asynchronous communication methods.

Due to the vast amount of already existing, efficient communication software such as Skype, MSN Messenger and other IM and chat tools, it is possible that the communication layer is not implemented straight to the application, but an existing tool is utilised. This will decrease the workload of the developers but at the same time it makes the life of the user more difficult. In order to keep the application consistent and efficient, at least basic features for communication should be included. However, if it is already known that the application will be integrated into an existing framework, for example to a course management system such as Moodle [5], then the developers might consider utilising the communication methods of the framework.

Despite the way that communication related features are to be implemented, the aspects of communication in Table 4.8 and Table 4.9 can be used as guidelines.

At this point of the Nucleus Model the application should have already been tested in a real setting. This means that the users are not any more sitting in the same classroom, but are using the application as it is intended to be used. This means that the users do all the work and communication with the application itself. In order to make the analysis easier, it is a good idea to study some of the users more carefully, e.g. by observing when they are using the application.

During the evaluation the researchers should now evaluate the tool as a whole, whilst still focussing on the communication and collaboration in particular. Following questions can be used as guidelines:

- Can users communicate in a natural way? and
- Does the communication support the work?

This last stage finally covers all the items of the extended Activity Theory. Now the group utilising the social mindtool is able to engage in all relevant activities, and can use communication to discuss the rules of the group and division of labour. This relationship is presented in Figure 8.4.

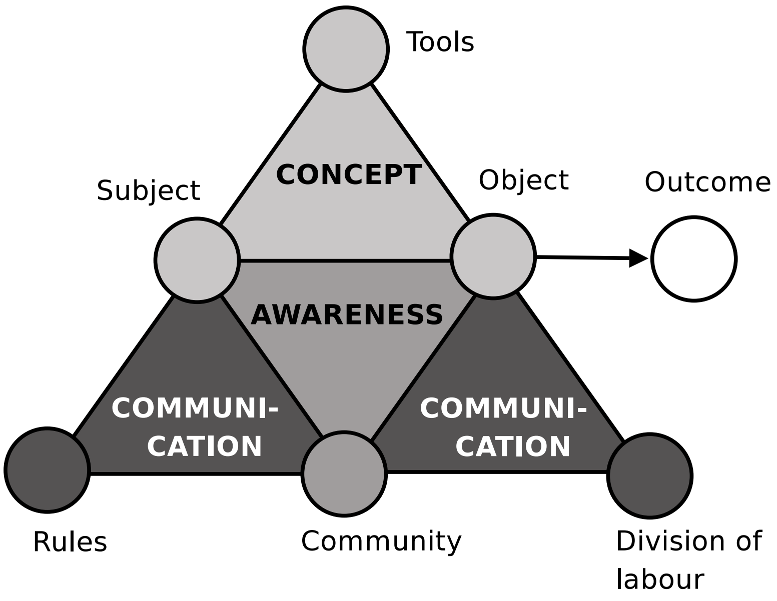


Figure 8.4: Relation between Activity Theory and communication stage of Nucleus Model.

8.5 Extensions

The agile manifesto [1] emphasises the amount of work not done. In the design and implementation process of social mindtools, this is important. The aim is to create a tool that can be used for various purposes and that is easy to extend.

As an example of a well designed and implemented tool, it is helpful to examine wikis. Wikis are fundamentally really simple applications, basically these store and display information, but with extensions it is easy to add more functionality to the application, making them suitable for various purposes. It would actually be possible to use a wiki as the basis for a Woven Stories application.

WikkaWiki [6] is a lightweight wiki implementation that is highly configurable by introducing new handlers, actions and formatters. Formatters and actions are responsible for output and handlers are responsible for the input of the data. All these can be implemented and added to the functionality of the WikkaWiki without changing the code of the core of the application. Implementation of all these is simple, and thus even a moderately skilled programmer is able to implement new features for WikkaWiki.

By allowing users to extend the application easily, the application can be easily adopted for various needs. Even though social mindtools should be generally usable for various tasks, extending their functionality easily is a valid asset. It is even better, if it is possible to include various extensions in the same installation of the application, as is the case in WikkaWiki.

Another common example of an extension of software is the Add-ons [87] for Mozilla Firefox [88]. These add-ons give the user extra features to extend the functionality of the browser. They can also be used to provide an alternative way to view a web page or provide more information from a site, such as the Foxtrick [2] add-on which supports the online football manager game Hattrick [61].

There can be several reasons to implement and introduce an extension to an application. For example, the application might lack:

- a general, often needed feature, or
- a specific feature needed for a certain application area or task, or
- a capability to store new kinds of data.

The above list only gives some examples of the needs. It is hard to give any general guidelines on extensions. The need for extensions can be determined at least with two different ways:

1. by creating a requirement specification for a known task, or
2. by using the application for a certain task and observing the need during the use.

The first way can be used in situations where it is already known for what kinds of tasks the application will be used. For example, there might be the need to provide spatial information while using the application and due to this the user must have possibility to add, edit and display spatial data in the application.

The second way is perhaps more likely to be used in context of social mindtools. The social mindtool is first tested for the purpose it is intended to be used, and during the testing it comes evident that certain features are missing. During the test the needs for the extensions are documented and after testing these can be implemented.

8.6 Discussion

Last research question of this thesis asks how does the Nucleus Model accommodate the design processes for Woven Stories and other social mindtools? The model consist of three stages, where the concept, awareness and communication related features are designed and implemented in certain order as presented in Figure 8.1.

Based on my experiences the combination of the Development Research Approach and Nucleus Model, they can provide educational technology researchers a way to proceed with the design of social mindtools or any collaborative educational media. The Nucleus Model divides the design and implementation of the software into the three individual main stages and thereby helps focusing the implementation and evaluation.

Unlike using an adhoc approach, it is possible to make even radical modifications to the application in any stage of the development, without causing incoherence and disturbing the whole process. However, adopting the model can also raise the threshold to go back in the development process.

The Nucleus Model aims to emphasise the importance of devotion to the original concept of the tool. Also, the importance of keeping the concept simple is pointed out. This model works as a focus lens for the implementation and evaluation of these tools. The main strength of the model is that it enables the possibility to evaluate the application, with a focus on the conceptual issues, in early stages of the development.

The Nucleus Model is the first step toward a manageable design and implementation process of social mindtools. It is based on the development of Loom and thus applicable for applications based on the concept of Woven Stories, but I believe that it can be utilised for social mindtools and collaborative educational media in general. I believe that applying this model in the application development process results in a more focused, correctly working and highly usable collaborative educational application. Furthermore, it is possible that the model could be used as a development approach for any collaborative application. The same principles apply to these tools; the concept is the core of the application, the application must support users with awareness information and, finally, the application should provide users with means for communication.

One of the disadvantages is the fact that it is not possible to state general acceptance criteria for these three stages. Thus, the researchers and developers are required to specify these criteria for the tool they are working with. On the other hand, the very same need of explicitly stating the requirements and acceptance beforehand can be seen as an advantage in that it imposes an organised development process of educational technology applications.

Chapter 9

Conclusions

During this study a social mindtool, Loom, based on the concept of Woven Stories was implemented. The process of implementing and evaluating the tool was demanding. Based on the analysis of the process and the challenges met, a new approach for the design of social mindtools, and collaborative tools in general, was introduced.

In Section 2.1 I introduced five separate research questions, to which I here give summarised answers.

Q1 *What features characterise a social mindtool within the set of mindtools?*

This question, covered in Chapter 4, was answered by utilising literature analysis. In order to formulate the answer, characteristics of mindtools, i.e., accessibility, engagement, multi-purpose utility and usability, are presented in Table 4.1. Based on this, specific requirements for social mindtools were derived and are presented in Table 4.6. The main result is a framework for social mindtools, which consists of a number of question sets to determine mindtool related features (see Table 4.7) and asynchronous (see Table 4.8) and synchronous (see Table 4.9) collaborative features, specific for social mindtools. This framework is presented in Section 4.8.

The second research question of this thesis, Q2, asks;

Q2 *How are the characteristics of a social mindtool present in the architecture of Woven Stories?*

Based on the framework for social mindtools, the architecture of the Woven Stories application is divided into three categories; concept, awareness and communication. These form a layered model of three nuclei. Chapter 5 analyses these nuclei in contrast to the concept of Woven Stories and presents the required features in Table 5.1, Table 5.2 and Table 5.3. In order to make the application usable, and provide features that only computer based applications allow, certain extensions, like user management and story management are needed. These extensions are introduced in Section 5.2.

Q3 *How do the characteristics of a social mindtool influence the technical implementation of Woven Stories application?*

This question is answered in Chapter 6 by introducing a Java-based implementation of Woven Stories called Loom. This application utilises the architecture presented in Chapter 5 as its base. The design and implementation process of Loom has been a long process, started already 2003. Furthermore, an analysis of Loom as a social mindtool, based on the framework of social mindtools, is given in Section 6.7. As a part of the answer to this research question, the newest Woven Stories application, WS@Web is introduced in Section 6.5.

Q4 *What kinds of learning tasks does Woven Stories support?*

In order to analyse the use and utilisation of Loom, it was applied to several use cases. Six selected cases are reported in Section 7.1. Based on these case studies, a compilation of lessons learnt is presented in Table 7.11. The main findings were that Loom can be used in several different application areas, as a social mindtool should be. Furthermore, the use of Loom (and Woven Stories in general) promotes creative thinking of its users. The most suitable tasks for using Loom are open ended and meaningful tasks.

This question is covered in Chapter 7, which includes also a comparison between Woven Stories and wikis. This comparison shows that, whilst these tools have similarities, wikis have certain limitations that Woven Stories can overcome.

Q5 *How does the Nucleus Model accommodate the design processes for Woven Stories and other social mindtools?*

Based on the framework of social mindtools, and the layered model presented in Section 5.1, it is possible to identify the feature layers of social mindtools. By combining these layers with the lessons learnt from the case studies presented in this thesis, a layered model for the design and implementation of social mindtools is presented. This model is called the Nucleus Model and is presented in Chapter 8. The Nucleus Model suggests that the design and implementation of social mindtools should follow the stages of concept, awareness and communication. The model provides the order in which these features are to be implemented, and gives guidelines how to evaluate the social mindtool during each stage. The Nucleus Model eases the design and implementation process of social mindtools and makes it easier to set the focus during the evaluation of the tool.

Given the progress of web-based techniques, the current implementation of Loom is already a bit outdated as far as technology is concerned. Java based applications might be good as desktop applications, but for communities and collaborative applications the web is the environment of choice. Based on the experiences and results of this thesis, a brand new version of Woven Stories application should be developed by following the Nucleus Model. WS@Web is a good starting point. Another interesting idea would be to implement Woven Stories on top of a wiki. As stated in Section 7.3, Woven Stories can overcome certain limitations of wikis and thus the combination of these two could be a powerful tool for various tasks. Furthermore, since wikis already provide a similar basic functionality needed in Woven Stories, this approach could speed up the design and implementation process as well.

The results of this thesis indicate the kinds of learning tasks that Woven Stories can support. The next step is to evaluate the educational effectiveness of Woven Stories and the novel approaches for learning that Woven Stories provides. This needs to be done in collaboration with educationalists.

The Nucleus Model provides researchers and developers of social mindtools with guidelines to work with. However, it has not been tested in real development yet. Due to this, it should be used in an appropriate design project in order to evaluate its effectiveness. Even though it is not possible to give or produce any general acceptance criterion for social mindtools, the Nucleus Model can provide developers with tools to make the design and implementation process of social mindtools fluent and functional.

The main contribution of this thesis is three-fold. It includes the framework for social mindtools, architecture and implementation of Loom and its evaluation, and the Nucleus Model. These results can be used for various tasks, like social mindtools. By utilising the framework, it is possible to evaluate and design social mindtools. With Loom several application areas can obtain the benefits of a structured writing tool. And finally, by following the Nucleus Model, the developers of social mindtools and educational media in general, can have guidelines to follow during the complex process of design and implementation of social mindtools. The model is based on the identified "right mistakes" done during the design and implementation process of Loom. By turning these mistakes into valuable lessons learnt, a promising model for designing social mindtools has emerged.

"If anybody wants to clap," said Eeyore when he had read this, "now is the time to do it." They all clapped. "Thank you," said Eeyore. "Unexpected and gratifying, if a little lacking in Smack."

– A.A.Milne, The House At Pooh Corner

References

[1] Agile Manifesto. http://agilemanifesto.org/. Accessed October 31, 2008.

[2] FoxTrick. http://www.ht-foxtrick.com/forum/portal.php. Accessed April 22, 2009.

[3] JSON. http://www.json.org/. Accessed April 20, 2009.

[4] Knowledge Forum. http://www.knowledgeforum.com. Accessed April 15, 2009.

[5] Moodle. http://www.moodle.org. Accessed April 16, 2009.

[6] WikkaWiki. http://www.wikkawiki.org. Accessed April 10, 2009.

[7] Ausubel, D. P., Novak, J. D., and Hanesian, H. *Educational Psychology: a Cognitive View.* Holt, Rinehart and Winston, New York, 1978.

[8] Baecker, R. M., Glass, G., Mitchell, A., and Posner, I. SASSE: the collaborative editor. In *Conference Companion on Human Factors in Computing Systems* (1994), ACM Press, pp. 459–462.

[9] Baecker, R. M., Nastos, D., Posner, I. R., and Mawby, K. L. The user-centered iterative design of collaborative writing software. In *CHI '93: Proceedings of the SIGCHI Conference on Human factors in Computing Systems* (New York, NY, USA, 1993), ACM Press, pp. 399–405.

[10] Baeza-Yates, R., and Ribeiro-Neto, B. *Modern Information Retrieval.* Addison Wesley, May 1999.

[11] Bannon, Liam, J., and Schmidt, K. CSCW: Four Characters in Search of a Context. In *Studies in Computer Supported Cooperative Work: Theory, Practise and Design*, J. Bowers and S. Benford, Eds. Amsterdam North-Holland, 1991, pp. 3–16.

[12] Bartholomew, D. A look at the kindle. *Linux J. 2008*, 176 (2008), 3.

[13] Beenen, G., Ling, K., Wang, X., Chang, K., Frankowski, D., Resnick, P., and Kraut, R. E. Using social psychology to motivate contributions to online communities. In *CSCW '04: Proceedings of the 2004 ACM Conference on Computer Supported Cooperative Work* (New York, NY, USA, 2004), ACM Press, pp. 212–221.

[14] Bereiter, C. *Education and Mind in the Knowledge Age.* Lawrence Erlbaum Associates, Mahwah, NJ, 2002.

[15] Bereiter, C., and Scardamalia, M. *The Psychology of Written Composition.* Lawrence Erlbaum Associates, Hillsdale, N.J., 1987.

[16] Berners-Lee, T., Cailliau, R., Luotonen, A., Nielsen, H. F., and Secret, A. The world-wide web. *Commununications of the ACM 37*, 8 (1994), 76–82.

[17] Berners-Lee, T. J. Information management: A proposal. Tech. rep., CERN, 1989. Accessed, July 23rd, 2009 at http://www.w3.org/History/1989/proposal.html.

[18] Bernstein, M. Storyspace 1. In *HYPERTEXT '02: Proceedings of the thirteenth ACM Conference on Hypertext and Hypermedia* (New York, NY, USA, 2002), ACM, pp. 172–181.

[19] Berztiss, A. T. *Data Structures Theory and Practise.* Academic Press Inc., New York, 1975.

[20] Berztiss, A. T. Dimensions of the knowledge management process. In *Database and Expert Systems Applications, 2001. Proceedings. 12th International Workshop on* (2001), pp. 437–441.

[21] Beyer, H., and Holtzblatt, K. *Contextual Design.* Morgan Kaufmann Publishers, San Francisco, CA, 1998.

[22] Bodker, S. A human activity approach to user interfaces. *Human-Computer Interaction 4*, 3 (1989), 171–195.

[23] Bolter, J. D., and Joyce, M. Hypertext and creative writing. In *HYPERTEXT '87: Proceedings of the ACM Conference on Hypertext* (New York, NY, USA, 1987), ACM, pp. 41–50.

[24] Brinck, T., and Gomez, L. M. A collaborative medium for the support of conversational props. In *CSCW '92: Proceedings of the 1992 ACM Conference on Computer-Supported Cooperative Work* (New York, NY, USA, 1992), ACM Press, pp. 171–178.

[25] Bruner, J. S. The act of discovery. *Harvard Educational Review 31* (1961), 21–32.

[26] Brusilovsky, P. Methods and techniques of adaptive hypermedia. *User Modeling and User-Adapted Interaction 6*, 2 (July 1996), 87–129.

[27] Bryant, S. L., Forte, A., and Bruckman, A. Becoming wikipedian: transformation of participation in a collaborative online encyclopedia. In *GROUP '05: Proceedings of the 2005 International ACM SIGGROUP Conference on Supporting Group Work* (New York, NY, USA, 2005), ACM, pp. 1–10.

[28] Bush, V. As we may think. *SIGPC Note. 1*, 4 (1979), 36–44.

[29] Buzan Centers. www.mind-map.com. Accessed May 24, 2009.

[30] Calvi, L., and De Bra, P. Improving the usability of hypertext courseware through adaptive linking. In *HYPERTEXT '97: Proceedings of the eighth ACM Conference on Hypertext* (New York, NY, USA, 1997), ACM, pp. 224–225.

[31] Carroll, J. M., Neale, D. C., Isenhour, P. L., Rosson, M. B., and McCrickard, D. S. Notification and awareness: synchronizing task-oriented collaborative activity. *International Journal of Human-Computer Studies 58*, 5 (2003), 605 – 632. Notification User Interfaces.

[32] Chapin, N. Flowcharting with the ansi standard: A tutorial. *ACM Computing Surveys 2*, 2 (1970), 119–146.

[33] Collins, P., Shukla, S., and Redmiles, D. Activity theory and system design: A view from the trenches. *Computer Supported Cooperative Work (CSCW) 11*, 1 (Mar. 2002), 55–80.

[34] Conklin, J. Hypertext: An introduction and survey. *Computer 20*, 9 (1987), 17–41.

[35] Conklin, J., and Begeman, M. L. gIBIS: a hypertext tool for team design deliberation. In *HYPERTEXT '87: Proceedings of the ACM Conference on Hypertext* (New York, NY, USA, 1987), ACM, pp. 247–251.

[36] Conklin, J., and Begeman, M. L. gIBIS: a hypertext tool for exploratory policy discussion. *ACM Transactions on Information Systems 6*, 4 (1988), 303–331.

[37] Cress, U., and Kimmerle, J. A systemic and cognitive view on collaborative knowledge building with wikis. *International Journal of Computer-Supported Collaborative Learning 3*, 2 (June 2008), 105–122.

[38] Davies, G. *Collaboration and Writing.* Open University Press, Milton Keynes, 1989, ch. Writing without a pencil, pp. 93–105.

[39] De Bra, P., Brusilovsky, P., and Houben, G.-J. Adaptive hypermedia: from systems to framework. *ACM Computing Surveys* (1999), 12.

[40] De Bra, P., Houben, G.-J., and Wu, H. Aham: a dexter-based reference model for adaptive hypermedia. In *HYPERTEXT '99: Proceedings of the tenth ACM Conference on Hypertext and Hypermedia : Returning to Our Diverse Roots* (New York, NY, USA, 1999), ACM, pp. 147–156.

[41] Désilets, A., Paquet, S., and Vinson, N. G. Are wikis usable? In *WikiSym '05: Proceedings of the 2005 International Symposium on Wikis* (New York, NY, USA, 2005), ACM, pp. 3–15.

[42] Dourish, P., and Bellotti, V. Awareness and coordination in shared workspaces. In *CSCW '92: Proceedings of the 1992 ACM conference on Computer-Supported Cooperative Work* (New York, NY, USA, 1992), ACM Press, pp. 107–114.

[43] Ellis, C. A., Gibbs, S. J., and Rein, G. Groupware: some issues and experiences. *Commununications of the ACM 34*, 1 (1991), 39–58.

[44] Engelbart, D. C., and English, W. K. A research center for augmenting human intellect. In *AFIPS '68 (Fall, part I): Proceedings of the December 9-11, 1968, Fall Joint Computer Conference, Part I* (New York, NY, USA, 1968), ACM, pp. 395–410.

[45] Engeström, Y. *Learning by Expanding: an Activity-Theoretical Approach to Developmental Research.* Orienta-Konsultit Oy, Helsinki, Finland, 1987.

[46] Eronen, P. J., Sutinen, E., Vesisenaho, M., and Virnes, M. Kids' club as an ICT-based learning laboratory. *Informatics in Education 1*, 1 (2002), 61–72.

[47] Eysenck, M. W., and Keane, M. T. *Cognitive Psychology : a Student's Handbook.* Taylor & Francis Inc, Hove, East Sussex, 2005.

[48] Fish, R. S., Kraut, R. E., and Leland, M. D. P. Quilt: a collaborative tool for cooperative writing. In *Conference Sponsored by ACM SIGOIS and IEEECS TC-OA on Office Information Systems* (1988), ACM Press, pp. 30–37.

[49] Fuchs, T. Scriptaculous. http://script.aculo.us/. Accessed April 20, 2009.

[50] Gerdt, P. Computer supported collaborative writing (tietokoneavusteinen yhteiskirjoittaminen). Master's thesis, University of Joensuu, 2001. In Finnish.

[51] Glushko, R., Dougherty, D., Kimber, E., Rizk, A., Russell, D., and Summers, K. Html (panel): poison or panacea? In *ECHT '94: Proceedings of the 1994 ACM European Conference on Hypermedia Technology* (New York, NY, USA, 1994), ACM, pp. 245–246.

[52] Greenberg, S. A fisheye text editor for relaxed-wysiwis groupware. In *CHI '96: Conference Companion on Human Factors in Computing Systems* (New York, NY, USA, 1996), ACM Press, pp. 212–213.

[53] Grudin, J. Computer-Supported Cooperative Work: History and Focus. *Computer 27*, 5 (May 1994), 19–26.

[54] Gutwin, C., and Greenberg, S. Workspace awareness for groupware. In *CHI '96: Conference Companion on Human Factors in Computing Systems* (New York, NY, USA, 1996), ACM Press, pp. 208–209.

[55] Gutwin, C., and Greenberg, S. A descriptive framework of workspace awareness for real-time groupware. *Computer Supported Cooperative Work (CSCW) 11*, 3-4 (2002), 411–446.

[56] Hakkarainen, K. A knowledge-practice perspective on technology-mediated learning. *International Journal of Computer-Supported Collaborative Learning 4*, 2 (June 2009), 213–231.

[57] Halasz, F. G. Reflections on notecards: seven issues for the next generation of hypermedia systems. In *HYPERTEXT '87: Proceedings of the ACM Conference on Hypertext* (New York, NY, USA, 1987), ACM, pp. 345–365.

[58] Halasz, F. G., Moran, T. P., and Trigg, R. H. Notecards in a nutshell. In *CHI '87: Proceedings of the SIGCHI/GI conference on Human Factors in Computing Systems and Graphics Interface* (New York, NY, USA, 1987), ACM, pp. 45–52.

[59] Hartley, J., and Tynjälä, P. *Writing as a learning tool: Integrating theory and practise.* Kluwer Academic Publishers, Dordrecth, The Netherlands, 2001,

ch. New Technology, Writing and Learning, pp. 161–182.

[60] Harviainen, T., Hassinen, M., Kommers, P., and Sutinen, E. Woven Stories: Co-authoring over the internet. In *Advanced Research in Computer and Communications in Education* (Chiba, Japan, 1999), IOS Press, pp. 285–292.

[61] Hattrick Ltd. Hattrick. http://www.hattrick.org. Accessed, April 21, 2009.

[62] Hoadley, C. M., and Kilner, P. G. Using technology to transform communities of practice into knowledge-building communities. *SIGGROUP Bulletin 25*, 1 (2005), 31–40.

[63] Hopcroft, J. E., Motwani, R., and Ullman, J. D. *Introduction to Automata Theory, Languages, and Computation.* Person Education International Inc., Upper Saddle River, N.J., 2003.

[64] Hung, D., Lim, K., Chen, D.-T., and Koh, T. Leveraging online communities in fostering adaptive schools. *International Journal of Computer-Supported Collaborative Learning 3*, 4 (Dec. 2008), 373–386.

[65] Jonassen, D. *Instructional-Design Theories and Models.* Lawrence Erlbaum Associates, Inc., Mahwah, NJ., 1999, ch. Designing Constructivist Learning Environments, pp. 215 –239.

[66] Jonassen, D., and Rohrer-Murphy, L. Activity theory as a framework for designing constructivist learning environments. *Educational Technology Research and Development 47*, 1 (Mar. 1999), 61–79.

[67] Jonassen, D. H. *Cognitive Tools for Learning*, vol. F 81 of *NATO ASI Series.* Springer Verlag, Berlin Heidelberg, 1992, ch. What are Cognitive Tools, pp. 1–6.

[68] Jonassen, D. H. *Computers as Mindtools for Schools*, 2nd ed. Merrill Prentice Hall, Upper Saddle River, N.J., 2000.

[69] Jonassen, D. H. *Modeling with Technology: Mindtools for Conceptual Change.* Pearson Education, Upper Saddle River, N.J., 2006.

[70] Joyce, M. Storyspace as a hypertext system for writers and readers of varying ability. In *HYPERTEXT '91: Proceedings of the third annual ACM Conference on Hypertext* (New York, NY, USA, 1991), ACM, pp. 381–387.

[71] jQuery Team. jQuery. http://jquery.com/. Accessed April 20, 2009.

[72] Kaptelinin, V., Kuutti, K., and Bannon, L. J. Activity theory: Basic concepts and applications. In *EWHCI* (1995), pp. 189–201.

[73] Kaptelinin, V., and Nardi, B. A. *Acting with Technology: Activity Theory and Interaction Design (Acting with Technology).* The MIT Press, October 2006.

[74] Kim, H.-C. E., and Eklundh, K. S. Reviewing practices in collaborative writing. *Computer Supported Cooperative Work 10*, 2 (2001), 247–259.

[75] Kimmerle, J., and Cress, U. Group awareness and self-presentation

in computer-supported information exchange. *International Journal of Computer-Supported Collaborative Learning 3*, 1 (Mar. 2008), 85–97.

[76] KOMMERS, P., ISAIAS, P., AND NUNES, M. B., Eds. *Proceedings of the Web Based Communities 2004* (2004), IADIS Press.

[77] KUUTTI, K. *Context and consciousness: activity theory and human-computer interaction*. Massachusetts Institute of Technology, Cambridge, MA, USA, 1995, ch. Activity theory as a potential framework for human-computer interaction research, pp. 17–44.

[78] LIINAMAA, K., NUUTINEN, J., SUTINEN, E., AND VANHARANTA, H. Collaborative strategic planning on-line. *Psychnology 2*, 2 (2004).

[79] LOWRY, P. B., CURTIS, A., AND LOWRY, M. R. Building a taxonomy and nomenclature of collaborative writing to improve interdisciplinary research and practice. *Journal of Business Communication 41*, 1 (2004), 66 – 99.

[80] MAO, J.-Y., VREDENBURG, K., SMITH, P. W., AND CAREY, T. The state of user-centered design practice. *Commununications of the ACM 48*, 3 (2005), 105–109.

[81] MARSHALL, C. C., AND SHIPMAN, III, F. M. Spatial hypertext: designing for change. *Commununications of the ACM 38*, 8 (1995), 88–97.

[82] MAYES, J. T. *Cognitive Tools for Learning*, vol. F 81 of *NATO ASI Series*. Springer Verlag, Berlin Heidelberg, 1992, ch. Cognitive Tools: A Suitable Case for Learning, pp. 7 – 18.

[83] MEMMI, D. The nature of virtual communities. *AI & Society* (Apr. 2006), 1–13.

[84] MICROSOFT. Messenger. http://download.live.com/?sku=messenger. Accessed April 21, 2009.

[85] MORENO, A., MYLLER, N., SUTINEN, E., AND BEN-ARI, M. Visualizing programs with jeliot 3. In *Proceedings of the Working Conference on Advanced Visual Interfaces, AVI '04* (New York, NY, USA, 2004), ACM Press, pp. 373–376.

[86] MOZGOVOY, M., FREDRIKSSON, K., WHITE, D., JOY, M., AND SUTINEN, E. Fast plagiarism detection system. In *Proceedings of the International Symposium on String Processing and Information Retrieval (SPIRE2005), Buenos Aires, Argentina, November 2005 (Lecture Notes in Computer Science)* (2005), Springer, pp. 267–270.

[87] MOZILLA. Firefox Add-ons. https://addons.mozilla.org. Accessed April 22, 2009.

[88] MOZILLA. Firefox web browser. http://www.mozilla.com/en-US/firefox/firefox.html. Accessed April 22, 2009.

[89] MYLLER, N., AND NUUTINEN, J. JeCo: Combining program visualization and story weaving. *Informatics in Education 5*, 2 (2006), 255 – 264.

[90] NARDI, B. A., Ed. *Context and consciousness: activity theory and human-*

computer interaction. Massachusetts Institute of Technology, Cambridge, MA, USA, 1995.

[91] Nelson, T. H. Complex information processing: a file structure for the complex, the changing and the indeterminate. In *Proceedings of the 1965 20th National Conference* (New York, NY, USA, 1965), ACM, pp. 84–100.

[92] Nokelainen, P., Kurhila, J., Miettinen, M., Floren, P., and Tirri, H. Evaluating the role of a shared document-based annotation tool in learner-centered collaborative learning. In *Proceedings of the 3rd IEEE International Conference on Advanced Learning Technologies* (2003), pp. 200–203.

[93] Novak, J. D., and Gowin, D. B. *Learning How to Learn.* Cambridge University Press, New York, 1984.

[94] Nuutinen, J. *Designing a Computer Supported Collaborative Mindtool: Woven Stories.* Licentiate thesis, University of Joensuu, 2006.

[95] Nuutinen, J., Bednarik, R., and Sutinen, E. A layered approach to the development process of social mindtools. In *EdMedia, 2008. Proceedings.* (2008), pp. 2109–2118.

[96] Nuutinen, J., Botha, A., Sutinen, E., and Kommers, P. From mindtools to social mindtools: Collaborative writing with woven stories. *British Journal of Educational Technology.* (in press).

[97] Nuutinen, J., Laine, T., Sutinen, E., Buter, R., and Noyons, E. Problem and content development to support evaluation of science. In *Proceedings of the E-Learn 2004* (2004).

[98] Nuutinen, J., Liinamaa, K., Sutinen, E., and Vanharanta, H. Strategist's learning space. In *Web-Based Education* (2004), Acta Press, pp. 544–548.

[99] Nuutinen, J., Liinamaa, K., Sutinen, E., and Vanharanta, H. Woven stories as a tool for corporate strategy planning. In *Web Based Communities 2004* (Lisbon, 2004), IADIS Press, pp. 438–441.

[100] Nuutinen, J., and Sutinen, E. Visualization of the learning process using concept mapping. In *Proceedings of the 3rd IEEE International Conference onAdvanced Learning Technologies* (2003), pp. 348–349.

[101] Nuutinen, J., and Sutinen, E. Information retrieval techniques for collaborative text searches. In *Proceedings of the Ninth IEEE International Conference on Advanced Learning Technologies* (2009), pp. 390–392.

[102] Orion, N., Dubowski, Y., and Dodick, J. The educational potential of multimedia authoring as a part of the earth science curriculum - a case study. *Journal of Research in Science Teaching 37*, 10 (2000), 1121–1153.

[103] Oxford University Press. The Oxford English Dictionary. http://dictionary.oed.com. Accessed April 20, 2009.

[104] Porteneuve, C. *Prototype and Script.Aculo.Us.* Pragmatic Bookshelf, Raleigh, North Carolina, USA, 2007.

[105] Prante, T., Magerkurth, C., and Streitz, N. Developing CSCW tools

for idea finding -: empirical results and implications for design. In *Proceedings of the 2002 ACM Conference on Computer Supported Cooperative Work* (2002), ACM Press, pp. 106–115.

[106] Prototype Core Team. Prototype. http://www.prototypejs.org/. Accessed April 20, 2009.

[107] Reeves, T. C. Enhancing the worth of instructional technology research through "design experiments" and other development research strategies. *International Perspectives on Instructional Technology Research for the 21st Century,* (2000).

[108] Rollett, H., Lux, M., Strohmaier, M., Dosinger, G., and Tochtermann, K. The web 2.0 way of learning with technologies. *International Journal of Learning Technology 3*, 1 (2007), 87–107.

[109] Salo, M. *Intuitive Strategy Development Process - a New Theoretical Approach - Possibilities to Use Predefined Questions.* Licenciate thesis, Tampere University of Technology, 2005.

[110] Salo, M. *Woven Strategies.* PhD thesis, Tampere University of Technology, 2006.

[111] Scanlon, E., and Issroff, K. Activity theory and higher education: evaluating learning technologies. *Journal of Computer Assisted Learning 21*, 6 (December 2005), 430–439.

[112] Scardamalia, M., and Bereiter, C. Computer support for knowledge-building communities. *The Journal of the Learning Sciences 3*, 3 (1994), 365–283.

[113] Schraefel, M., Carr, L., De Roure, D., and Hall, W. You've got hypertext. *Journal of Digital Information 5*, 1 (2004).

[114] Sharp, H., Rogers, Y., and Preece, J. *Interaction Design: Beyond Human Computer Interaction.* Wiley, March 2007.

[115] Shaw, T., Arnason, K., and Belardo, S. The effects of computer mediated interactivity on idea generation: an experimental investigation. *IEEE Transactions on Systems, Man and Cybernetics 23*, 3 (May-June 1993), 737–745.

[116] Skype Technologies S.A. Skype. http://www.skype.com. Accessed May 5, 2009.

[117] Smith, F. *Writing and the Writer.* Heinemann Educational Book Ltd, London, 1982.

[118] Smith, J. B. The king is dead; long live the king (keynote). In *HYPERTEXT '97: Proceedings of the Eighth ACM Conference on Hypertext* (New York, NY, USA, 1997), ACM, p. 240.

[119] Smith, J. B., Weiss, S. F., and Ferguson, G. J. A hypertext writing environment and its cognitive basis (panel session). In *HYPERTEXT '87: Proceedings of the ACM Conference on Hypertext* (New York, NY, USA, 1987),

ACM, pp. 195–214.

[120] Sohlenkamp, M., and Chwelos, G. Integrating communication, cooperation, and awareness: the DIVA virtual office environment. In *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work* (1994), ACM Press, pp. 331–343.

[121] Sowa, J. F. *Knowledge Representation.* Brooks/Cole, Pacific Grove (CA), 2000.

[122] Spence, R. *Information Visualization.* Pearson Education Limited, Harlow, England, 2001.

[123] Streitz, N., Haake, J., Hannemann, J., Lemke, A., Schuler, W., Schütt, H., and Thüring, M. Sepia: a cooperative hypermedia authoring environment. In *ECHT '92: Proceedings of the ACM Conference on Hypertext.* ACM, New York, NY, USA, 1992, pp. 11–22.

[124] Sun Microsystems, Inc. MySQL. http://www.mysql.com/. Accessed April 23, 2009.

[125] Sunassee, N. N., and Sewry, D. A. An investigation of knowledge management implementation strategies. In *SAICSIT '03: Proceedings of the 2003 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists on Enablement Through Technology* (Republic of South Africa, 2003), South African Institute for Computer Scientists and Information Technologists, pp. 24–36.

[126] Swarts, J. Cooperative writing: achieving coordination together and apart. In *SIGDOC '04: Proceedings of the 22nd annual International Conference on Design of Communication* (New York, NY, USA, 2004), ACM Press, pp. 83–89.

[127] ter Hofte, G. H. Generic Service Features of CSCW Applications. Tech. rep., Telematics Research Centre, 1996.

[128] The HSQLDB Development Group. Hypersonic DB homepage. http://www.hsqldb.org/. Accessed May 4., 2009.

[129] The Kopete development team. Kopete instant messenger. http://kopete.kde.org/. Accessed April 21, 2009.

[130] The PHP Group. PHP. http://www.php.net/. Accessed April 22, 2009.

[131] Tynjälä, P., Mason, L., and Lonka, K. *Writing as a learning tool: Integrating theory and practise.* Kluwer Academic Publishers, Dordrecth, The Netherlands, 2001, ch. Writing as a learning tool: an introduction, pp. 7–22.

[132] Ullrich, C., Borau, K., Luo, H., Tan, X., Shen, L., and Shen, R. Why web 2.0 is good for learning and for research: principles and prototypes. 705–714.

[133] University of Joensuu. SciFest. http://www.scifest.fi/home. Accessed August 20th 2009.

[134] van Joolingen, W. R. Designing for collaborative discovery learning. In *ITS*

'00: Proceedings of the 5th International Conference on Intelligent Tutoring Systems (London, UK, 2000), Springer-Verlag, pp. 202–211.

[135] Vanninen, P., Härkönen, S., Enkenberg, J., and Mäkelä, A. PuMe - Interactive learning environment employing the PipeQual model to forest growth and wood quality. *New Zealand Journal of Forest Science. 36*, 2–3 (2006), 280–292.

[136] Weinreich, H., Obendorf, H., and Lamersdorf, W. The look of the link - concepts for the user interface of extended hyperlinks. In *HYPERTEXT '01: Proceedings of the 12th ACM Conference on Hypertext and Hypermedia* (New York, NY, USA, 2001), ACM, pp. 19–28.

[137] Wiig, K. M. On the management of knowledge. http://www.km-forum.org/what_is.htm, 1996. Accessed May 24, 2009.

[138] Zellweger, P. T., Mangen, A., and Newman, P. Reading and writing fluid hypertext narratives. In *HYPERTEXT '02: Proceedings of the thirteenth ACM Conference on Hypertext and Hypermedia* (New York, NY, USA, 2002), ACM, pp. 45–54.

Appendix A

Formal Definition of Woven Stories

Let us now consider the formal definition of the concept of Woven Stories.

Definition A.1 *A* **woven story** *is a 6-tuple* $WS = \langle V, \delta, S, F, \Sigma, \tau \rangle$ *where*

- V *is the set of sections,*
- $\delta : V \rightarrow \mathcal{P}(V)$ *is the author defined transition function,*
- $S \subseteq V$ *is the set of start sections,*
- $F \subseteq V$ *is the set of end sections,*
- Σ *is the alphabet used, and*
- $\tau : V \rightarrow \Sigma^*$ *is a content function.*

In Definition A.1 the value of $\delta(v)$ is a set of sections that are followed by section v. It is also possible that $\delta(v)$ is empty. The set of start sections, S, is in most cases easy to define since each section $s \in V$ is included in S if $\forall v_x \in V, s \notin \delta(v_x)$. However if the story WS is cyclic, some start sections might be author specified. The problem with cyclicity also occurs with the set of end sections, F. In most cases, the end sections can be defined as sections for which $f \in V$ is included in F if $\delta(f) \bigcap \{f\} = \emptyset$. Again the cyclic stories might not have any such sections that would fulfil this requirement, so some of the end sections can be user-defined. Both of the sets, S and F, can also be empty sets if the nature of the story is cyclic. For example, a story written of the circulation of the oxygen in nature is a good example of a story with an empty set of start sections and end sections. Since users create the contents of the sections, the τ is a user-defined function.

As can be seen from Definition A.1, the 6-tuple WS gives enough specification to be able to create a graph. Set V defines the nodes and the function δ defines the edges of the graph. It should be noted that, by definition, a woven story is a directed graph. A woven story may contain cycles (see Definition A.4) and even loops, which means that there can be edges where $\delta(v_x) \ni v_x$. Thus, there can be edges that have the same start and end node (section). Loops, however, have not been introduced in any prototype implementing the concept of Woven Stories.

Let us now formally define a few other concepts that are related to the concept of Woven Stories.

Definition A.2 *An* **episode** *of length* $n+1$ *from* v_0 *to* v_n *is a sequence of sections* $v_0, v_1, ..., v_n$ *satisfying*
$v_i \in \delta(v_{i-1})$ *for* $i = 1, 2, ..., n$.

Episodes are small parts of the story. The length of the episode is determined by the amount of sections it includes, not by the amount of links, as is usually the case with graphs.

Definition A.3 *A* **storypath** *of length* $n+1$ *from* v_0 *to* v_n *is an episode satisfying*

1. $v_0 \in S$, *and*
2. $v_n \in F$

If the woven story is properly written, each of the storypaths should make a sensible story.

Definition A.4 *A woven story is cyclic if it includes an episode* $v_0, ..., v_i$ *where* $v_0 = v_i$.

Dissertations at the Department of Computer Science and Statistics

Julkaisija	Joensuun yliopisto Tietojenkäsittelytieteen ja tilastotieteen laitos
Publisher	University of Joensuu Department of Computer Science and Statistics
Sarjan toimittaja Series editor	Erkki Sutinen
Vaihdot	Joensuun yliopiston kirjasto/Vaihdot PL 107, 80101 Joensuu Puh. 013-251 2677, fax 013-251 2691 email: vaihdot@joensuu.fi
Exchanges	Joensuu University Library/Exchanges P.O. Box 107, FI-80101 Joensuu, FINLAND Tel. +358-13-251 2677, fax +358-13-251 2691 email: vaihdot@joensuu.fi
Myynti	Joensuun yliopiston kirjasto/Julkaisujen myynti PL 107, 80101 Joensuu Puh. 013-251 4509, fax 013-251 2691 email: joepub@joensuu.fi
Sales	Joensuu University Library/Sales of publications P.O. Box 107, FI-80101 Joensuu, FINLAND Tel. +358-13-251 4509, fax +358-13-251 2691 email: joepub@joensuu.fi