Small-Scale Studies in the STELLAR Network of Excellence

Research Report

„ Identifying knowledge development - a context-aware approach for analyzing knowledge processes "

Authors

Know-Center

Stefanie N. Lindstaedt
Andreas S. Rath
Nicolas Weber
Didier Devaurs

Knowledge Media Research Center

Ulrike Cress
Johannes Moskaliuk
Joachim Kimmerle

Monday, December 21, 2009

Kompetenzzentrum für wissensbasierte Anwendungen und Systeme Forschungs- und Entwicklungs GmbH

Inffeldgasse 21a
8010 Graz / Austria
www.know-center.at

Knowledge Media Research Center

Konrad-Adenauer-Str. 40
72072 Tübingen / Germany
www.iwm-kmrc.de

Abstract

When people use wikis to work jointly on shared digital artifacts, this may lead to the collaborative creation of new knowledge. The consideration of the digital products and the insights into the construction process itself may lead to a better understanding of knowledge-building processes. In turn, this understanding of knowledge building may help to design environments that support the improvement of knowledge building. Therefore, the central research question for our paper is twofold: on the one hand it identifies indicators for knowledge maturity; on the other hand it provides some considerations on how knowledge maturing may be supported. In order to provide answers to these questions, we conducted a study in which participants had to work with a wikitext with the instruction to improve the text and add further arguments. On the basis of the users' behavior, we developed an ontology-based task detection approach that identified knowledge maturing processes with a rate of 79.12% (the findings of the tool were gauged with the ratings of two experts who evaluated people's actions with regard to knowledge maturing). The findings are discussed against the background of the question on how it can be ensured that knowledge workers contribute to the development of knowledge.

1 Introduction

Using wikis or other social software tools for knowledge work may lead to new forms of collaboration and learning (Raitman, Augar,& Zhou, 2005; Reinhold, 2006): Individuals can use wikis to work jointly on shared digital artifacts that were provided through the Internet or a local area network, say, of an organization. This will not only lead to accumulation of knowledge, by which the knowledge of many individuals is brought together and made available to others, but also to knowledge emergence, i.e the creation of new knowledge (Johnson, 2001; Moskaliuk & Kimmerle, 2009). Thus, writing wiki text in a collaborative way is not only a method to share information but also to construct new knowledge (Cress & Kimmerle, 2008). During the process of writing as a joint activity, users generate new ideas and innovation and discuss own arguments with others. They construct shared meaning and build a mutual understanding (Erkens, Jaspers, Prangsma, Kanselaar, 2005). This may lead to an evolution of knowledge. Emergent knowledge that was not part of the individual knowledge of a single user before may arise during this collaborative activity.

Thus, learning is always embedded in a social and cultural context (cf. Lave & Wenger, 1991; Vygotsky, et al., 1987). Shared digital artifacts like wikis or other social software tools are external representations of this social and cultural context and contain the knowledge of the corresponding community. External representations of knowledge reflect and stimulate individual learning processes within a community or an organization and lead to a development of knowledge over time. The concept of knowledge maturing (Schmidt, 2005) describes this phenomenon from a macroscopic perspective. It considers how external knowledge representations (such as digital artifacts) correspond, are derived from or inform internal knowledge representations. The concept of knowledge maturing describes how knowledge matures from the level of individuals to the level of communities, and, finally, to the level of organizations. It focuses on how knowledge matures from expressing individual ideas to formalizing knowledge on an organizational level.

The process of knowledge maturing is condensed in digital artifacts (e.g. in a wiki text). With the help of these digital artifacts it becomes possible to evaluate the amount of knowledge that is available in the community of participating users (e.g. a class or an organization). Additionally, by considering the writing process closely, the knowledge maturing process itself may be analyzed and new insights may be gained into how people construct new knowledge. This consideration of the digital products and the insights into the construction process itself may lead to a better understanding of knowledge maturing. In turn, this understanding of knowledge maturing may help to design environments that support the improvement of knowledge within organizations, e.g. by giving feedback on the current status of a text.

These considerations lead to the idea that the quality of a text and its maturing over time is not only an important goal of each knowledge organization, but may also be used as an indicator for evaluating successful knowledge building. Therefore, the central research question for our paper is twofold:

(1) Maturity indicators: How can the quality of a text be measured and used as an indicator for knowledge maturing processes within communities?

(2) Knowledge Maturing support: How can it be ensured that knowledge workers write good texts and contribute to the development of knowledge?

In order to provide empirical answers to these questions, we conducted a study to collect data as basis for an ontology-based task detection approach. In this study participants had to work with a wiki, improve the text and add further arguments. The participants obtained additional information that contradicts the content in the wiki. This should provoke knowledge maturing and lead to prototypical editing tasks we could use to train our task detection tool and cross-validate its results. The goal is to identify features for knowledge maturing, which will make it possible to measure behavior automatically. The results of this study are supposed to lead to a better understanding of knowledge maturing processes in a wiki and of the factors that influence them.

In the following we will first of all explain the theoretical background and introduce a framework model that explains the development of knowledge over time as co-evolution of cognitive and social systems (section 2). We will distinguish between accommodation and assimilation as two essential processes of knowledge building. Then we describe the research setting we designed and how we use it to develop and validate our task detection approach (section 3). In section 4 we describe our task detection approach in detail. We present the results in section 5. As a conclusion we discuss the results of the task-detection approach against the background of the question on how it can be ensured that knowledge workers contribute to the development of knowledge.

2 Theoretical Background

The co-evolution model of cognitive and social systems (Cress & Kimmerle, 2008) may be understood as a framework to describe knowledge processes that take place in a wiki. This model describes individual learning and collaborative knowledge building as a co-evolution between cognitive and social systems. The co-evolution model (Cress and Kimmerle, 2008) considers two relevant systems: The social system wiki and the cognitive system of a user. Both systems are independent from each other and build their border to the environment with help of a specific mode of operation. Here, the authors refer on Luhmann's systems theory (Luhmann, 1986, 1995, 2006) and transfer his ideas to the context of knowledge building with wikis. The social system means the communication within a (virtual) community that becomes manifest as written text in a digital artifact (e.g. the wiki). The cognitive system consists of psychological processes like learning, reasoning, problem-solving or perception.

According to this model, both the social and the cognitive system develop over time and become more and more complex. The information in a wiki (that represents the knowledge of a community) evolves in the course of time. At the same time, the knowledge in an individuals cognitive system increases. This mutual development as a co-evolution of social and cognitive systems can be understood as knowledge building or knowledge maturing. Cognitive conflicts (Piaget, 1977a, 1977b) that an individual perceives are considered as the key incitement factor of this co-evolution. In the sense of Piaget a cognitive conflict occurs if new information from the environment does not fit existing knowledge. These cognitive conflicts motivate individuals to contribute to the wiki or to change their own knowledge structure in order to establish equilibrium between own knowledge and new information.

The model by Cress and Kimmerle (2008) specifies the co-evolution process and describes two different processes on the basis of the ideas of Piaget: assimilation and accommodation. Assimilation means active shaping of the environment by interpreting and explaining current experiences, giving them a place in existing schemata. Accommodation means adaptation to the environment in the form of qualitatively changing one’s own cognitive schemata. In Piagets understanding assimiliation and accommodation are processes in the cognitive system; the co-evolution model, however, expands Piaget’s point of view by describing accommodation and assimilation not only from the perspective of an individual’s cognitive system, but also from that of a social system. Users assimilate information from the artifact into their own cognitive schemata, and they accommodate by modifying their schemata induced by information from the wiki. Analogous processes of learning and knowledge building may take place in the social system: in the case of assimilation users add pieces of information from their own knowledge. This, however, will not change the basic message and structure of the wiki, but only add supplementary aspects. Accommodation is also possible in a wiki if users contribute their knowledge in such a way that the message is changed and, sometimes, new structures are being created. Accommodation tends to result in some qualitative modification of the artifact, whereas assimilation has primarily to do with quantity, introducing additional arguments or new examples but no fundamental innovation.

To sum up, the co-evolution model is considered as useful framework to describe how individuals use digital artifacts like wikis to jointly construct knowledge. The constructed knowledge manifests in digital artifacts and is therefore accessible for deeper analysis. In order to identify features for assimilation and accommodation actions we have to consider two relevant dimensions: The artifact dimension and the usage dimension (Mentzas, 2007)

The first dimension, content artifacts, provides a static picture of the world and is probably the best managed type of knowledge entity. It can take the form of notes, contributions, threads, protocols, lessons learned, learning objects, courses, pictures, videos, podcasts, etc. In many organizations textual contents are the most prevalent contents. For content maturing support, we rely on knowledge discovery algorithms mainly using statistical methods and some shallow natural language processing for text mining and text analysis. These methods extract features in form of feature vectors from a textual information object, from which textual similarity measures can be derived. These in turn are then used for text classification, clustering and for other kinds of operations.

The analysis of external knowledge representations would not be sufficient considering the analysis and support of organizational knowledge maturing processes. Hence, a further issue that has to be taken into account is the way in which these external knowledge representations are being used. In the usage dimension we seek to learn from the behavior of people interacting with digital artifacts. The interaction traces can tell something about the person (e.g. the role, interests, knowledge or skills the person is likely to have), about the knowledge asset the person is dealing with (e.g. the context or task a knowledge asset was created, used, changed or shared), and about the activities performed within an organization (e.g. process and task executions).

Looking at the artifact dimension and the usage dimension we are able to gain new insights how individual use texts to build knowledge and how the knowledge matures over time. Integrating the theory of Cress and Kimmerle (2008) with the knowledge maturing idea we suggest that the process of accommodation is a central maturity indicator. This is true for both, the cognitive system and the social system. Assimilation means a quantitative development of knowledge where new information only completes existing knowledge as accumulation of knowledge. Accommodation leads to better understanding of complex information, produces integrative mental models, new ideas and innovation, or emergent knowledge. This can be described as qualitative development of knowledge. The quality of text depends on accommodation processes. We propose that emergent effects usually occur through processes of accommodation in artifacts. This will lead to a higher complexity of the wiki and, accordingly, to knowledge processes in other people’s cognitive systems.

In addition to psychological approaches for measuring individual knowledge in the cognitive system (e.g. the experiment by Moskaliuk, Kimmerle, & Cress, 2009) it is interesting to take into account the processes in a digital artifact as indicators for processes in the social system. If we measure the quality of text and its development over time we can conclude underlying knowledge-building processes within communities (maturity indicators) and can use this results as feedback for users to support knowledge building (knowledge maturing support).

The main goal of the current study is to examine how accommodation (as opposed to assimilation) happens in detail within a wiki system and how knowledge maturing may be analyzed in an automatic way. We started with a qualitative assessment of user interactions on knowledge artifacts based on expert observations. In order to identify features for accommodation actions we designed a study where participants had to work with a wiki and add new information. Then, we developed an approach for the selection of (quantitative) interaction-based features in order to detect accommodating activities as maturity indicator.

3 Research Setting

We designed a research setting in which participants had to work with a wiki and add new information. The wiki and the additional information were presented at people's computers. The additional information was presented in the form of short arguments that participants could browse. In order to induce cognitive conflicts, the existing information in the wiki and the new information contradicted each other. This was supposed to provoke accommodation processes and is therefore considered an ideal condition to train the task detection tool.

All runs took place in a silent work environment in a secluded room during daytime. Participants had a total of 50 minutes to edit the wiki, the complete study lasts about 90 minutes. Participants were instructed to use the provided laptop computers and that it should solely be operated via the attached mouse and the keyboard and not by using the touchpad. Participants then started by viewing a welcoming page after which they had to click through several information and instruction pages regarding further details about the study and the difference between qualitative (accommodation-like) and quantitative (assimilation-like) changes they would make. Participants were instructed to save their changes after each set of alterations belonging together (editing task). Participants' task was to complete a given text of a wiki (initial state text), so that a scientifically balanced article about violent computer games and their possible danger for users and the society would result. Two windows were available for the participants during the study: a wiki page and a page with the additional information. Those two windows were accessible through two tabulators at the top of the page. The wiki page (wiki-tab) presented a text that, initially, was biased toward a contra violent computer games position. Here, only the risks and dangers of violent computer games were presented. The page with the additional information (info-tab) contained ten different arguments that were biased towards a pro violent computer games position. These arguments invalidated the arguments in the wiki text or explained why the contra points were one-sided or wrong. Participants could copy, paste, delete and edit the text freely or type in new text.

Participants are instructed to click on save each time they finished alterations belonging together. We expect that this would lead to single editing tasks. Each editing task could be rated on the dimension of assimilation and accommodation by the participants themselves (user-ratings), by experts on the same dimensions (expert-ratings), and could be measured using readings-scores.

User-Ratings: Whenever a participant clicked on “save”, a box popped up. In this box, one had to make two rating of the changes between the last and the present saving point (one single transformation), on a five-point Likert scale. The first rating is about “qualitative text-changes”, scale ranging from “no qualitative changes” to “many qualitative changes” (accommodation-like transformation), the second is about “quantitative text-changes”, scale ranging from “no qualitative changes” to “many qualitative changes” (assimilation-like transformation).

Expert-Ratings: Each transformation was rated by two independent experts on the same dimension like the user-ratings (qualitative text-changes; quantitative text-changes) on a five-point Likert scale. The overall correlation between the ratings of the two experts was significant. The correlation for the two accommodation ratings was $r_{acc}(156)=.797$, $p<.01$; for the assimilation rating, the correlation was $r(156)_{ass}=.741$, $p<.01$. For further analysis we build an average of the two rating for each transformation.

Reading-Scores: The objective of computing reading scores is to analyze content to facilitate the assessment of the maturity of a document. Readings scores are calculated from quantitative metrics like sentence length, number of syllables or number of words. We applied the *Flesch Reading Ease test* (Si & Callan, 2001), the *Gunning fog index* (Gunning, 2004), and the *Flesch–Kincaid readability test* (Flesch, 1948) to each transformation. More specifically, we computed the reading scores on the state of the document before the transformation and after the transformation. For each reading score we subtracted the respective resulting values, which we used as one of the variables for the correlation. The second variable for the correlation computation was the average over the accommodation ratings of the two experts. Since the used variables showed no normal distributions we used the rank order correlations. The results showed no significant correlations.

The goal of this research setting was to collect data on the basis of which we could develop an approach for the selection of interaction-based features in order to detect accommodating activities as maturity indicator. We used a context-aware system for observing users in performing changes to the wiki page. This observation is on a fine granular level that takes people's interactions with the wiki into account. For discriminating different levels of accommodation for the observation wiki transformations, we apply our ontology-based task detection approach. We describe this approach in the following section.

4 Task Detection Approach

Context-aware (or sentient) systems are systems that can adapt their operations or behavior to their current context of use, without explicit user intervention. Context-awareness thus enables to increase the usability and effectiveness of a system by taking into account environmental elements (such as time or location), individual and organizational elements (such as user’s identity or position), as well as elements relative to the interactions of the user with the system (such as pressed button or entered character). The first context-aware systems, developed in the 1990s, were mainly focused on providing functionalities specific to the user’s location (Baldauf et al., 2007). Today’s context-aware systems are much more sophisticated, and integrate complex mechanisms for the acquisition and storage of context, the abstraction and understanding of context, and the adaptation of the system behavior based on the recognized context.

Context information may be gathered in a variety of ways, such as applying (physical or virtual) sensors, recording network information and device status, or browsing user profiles and organizational databases. Then, a context model is needed for storing the recorded user context data in a machine processable form. Various context model approaches have been proposed, such as key-value models, markup scheme models, graphical models, object oriented models, logic-based models, or ontology-based models (Strang & Linnhoff-Popien, 2004). However, the ontology-based approach has been advocated as being the most promising one (Strang & Linnhoff-Popien, 2004; Baldauf et al., 2007) mainly because of its dynamicity, expressiveness and extensibility.

An important aspect of the user’s context is the current task she is performing. Detecting the user’s task enables to provide her with personalized and relevant support (Dey et al., 2001; Coutaz et al., 2005). By task detection we mean task class detection also referred to as task classification, as opposed to task switch detection. Task switch detection involves predicting when the user switches from one task to another (Shen et al., 2009). Task classification deals with the challenge of classifying usage data from user task execution into task classes or task types. Automatic task detection is classically modeled as a machine learning problem, and more precisely a classification problem. This method is used to recognize Web based tasks (Gutschmidt et al., 2008), tasks within emails (Shen et al, 2006) or tasks from the complete user’s computer desktop (Shen et al., 2006, 2009; Lokaiczyk et al., 2007; Rath et al., 2009). This relates to the present study in the sense that we consider accommodation and assimilation processes as being refined editing tasks that we would like to detect.

4.1 Automatically Detecting Accommodation

Solving our editing task classification problem is done based on the following steps (illustrated by Figure 1): (i) The user context data is captured by system and application sensors. (ii) Features, i.e. parts of this data, are chosen to build classification training instances, which is done at the task level. (iii) To obtain valid inputs for machine learning algorithms, these features are first transformed into attributes. This transformation may include data preprocessing operations, such as removing stopwords and application specific terms, or constructing word vectors. (iv) *Attribute selection* (optional step) is performed to select the best discriminative attributes. (v) Finally, classification/learning algorithms are trained and tested.

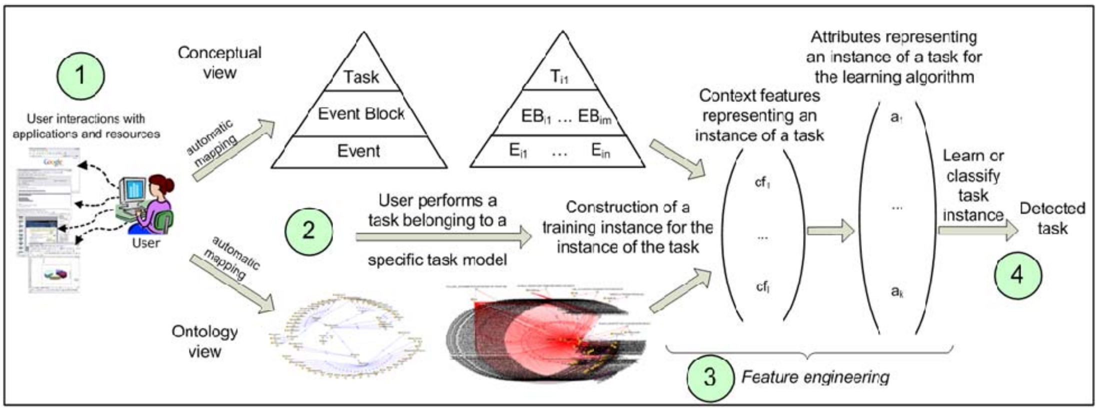

Figure 1: This figure visualizes the complete *user interaction context ontology task detection pipeline* (UICO pipeline) starting from (1) the automatic unobtrusive user interaction observation mechanisms to (4) detecting the user's task. The automatic population of the user interaction context model (2) is displayed in two ways (i) the instantiation of entities in the conceptual model (*conceptual view*) and (ii) in the ontology model (*ontology view*). In (3) the feature engineering process for transforming a task instance into a training instance is shown. The training instance is further fed to attribute selection and learning algorithms for (4) detecting the task.

4.2 User Interaction Context

The first step of the detection process consists in capturing the "user context". Our view of the "*user context*" goes along with Dey's definition that context is "*any information that can be used to characterize the situation of entities that are considered relevant to the interaction between a user and an application, including the user and the application themselves*" (Dey et al., 2001). We refine Dey's perspective by focusing on the *user interaction context* that we define as "*all interactions of the user with resources, applications and the operating system on the computer desktop.* Resources are digital artifacts on the computer desktop, e.g., documents, web pages, e-mails, persons, appointments and notes." (Rath et al., 2009). In the case of the wiki environment applied here both tabs (wiki-tab and info-tab) were perceived as an unique resource. Context observation mechanisms are used to capture the behavior of the user while working on her/his computer desktop. Low-level operating system and application events initiated by the user while interacting with the desktop are recorded by context observers, also referred to as context sensors. We distinguish between *system* and *application* sensors, based on the origin of the data they deliver. For capturing the user interaction context in the wiki environment of the present study we employed context sensors that have already been utilized in our previous user interaction context observation (Rath et al. 2008) and task detection experiments (Rath et al., 2009). We developed sensors for Macromedia Flash which was the base technology for the wiki editor and the page with the arguments (info-tab). Additional fine-granular user interaction context information included (i) switch to info-tab, (ii) switch to wiki-tab, (iii) text formatting (BOLD, UNBOLD, ITALICS, UNITALICS, UNDERLINE, UNUNDERLINE, COLORCHANGE, ALIGN_LEFT, ALIGN_RIGHT, ALIGN_CENTER, ALIGN_BLOCK, FONT_INCREASE, FONT_DECREASE, FONT_NAME_SWITCH), (iv) text editing, (v) text selection and (vi) selection of a specific argument on the info-tab. The content of the wiki page as well as the text around the cursor were also recorded for each single user interaction.

The conceptual representation that we propose for the user interaction context is a *semantic pyramid*. At the bottom of the pyramid are events that result from the user's interactions with the computer desktop. Above events are event-blocks, which are sequences of events that belong logically together, each event-block connecting the user's actions associated with a specific resource acted upon. At the top are tasks that are grouping of event-blocks representing well-defined steps of a process that cannot be divided into sub-tasks, and in which only one person is involved. The layers of the semantic pyramid represent the different aggregation levels of the user's actions. The semantic pyramid is illuminated in Figure 2.

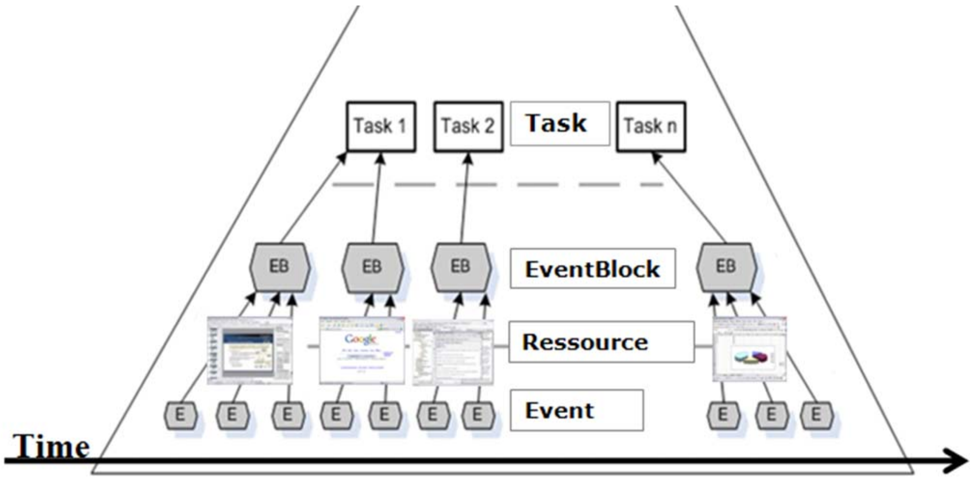

Figure 2:The semantic pyramid (Rath et al., 2008) comprises the event, the event block and the task layer.

4.3 User Interaction Context Ontology (UICO)

A context model is needed for storing the user context data in a machine processable form. We have defined a user interaction context ontology (UICO) (Rath et al., 2009a) which is illuminated in Figure 3. The UICO contains 88 concepts and 272 properties. It is modeled in OWL-DL, by using the Protégé ontology modeling tool. The ontology web language (OWL) is a W3C standard for modeling ontologies widely accepted in the Semantic Web community. From the 272 properties there are 215 datatype properties and 57 objecttype properties. From a high-level perspective, the concepts of our ontology can be grouped into five different dimensions: the action dimension, the resource dimension, the user dimension, the information need dimension and the application dimension. Figure X2 illuminates the user interaction context ontology.

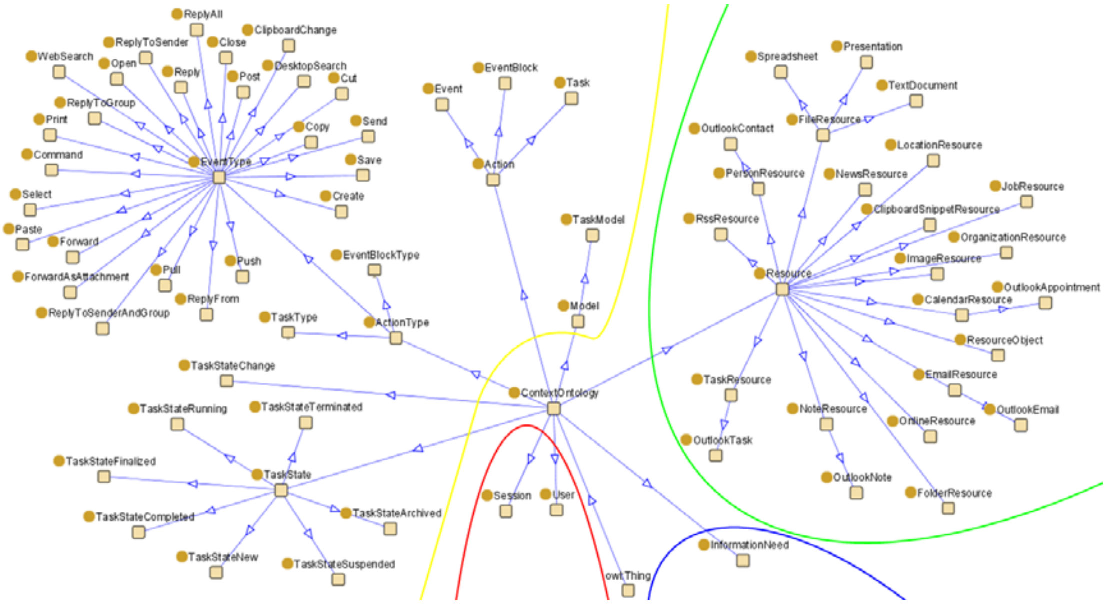

Figure 3: The concepts of the user interaction context ontology (UICO) visualized in the Protégé tool. In the left area this figure shows the *action dimension*, in the right area the *resource dimension*, in the bottom left area the *user dimension* and the *information need dimension* on the bottom right area. The *application dimension* is not represented as concepts and hence not visible here.

The **action dimension** consists of concepts representing user actions, task states and models. The *Action* concept is refined by the sub-concepts Event, EventBlock and Task. Examples of user Event sub-concepts are *Print*, *Close*, *Save*, *Copy*, *Paste*, *Cut*, *WebSearch*, *Post*, *Reply* and *Forward*. An *Event* concept stores information about the application in which it happens, through the datatype properties *hasWindowTitle*, *hasApplicationName* and *hasProcessId*. The different types of task states are borrowed from the Nepomuk Task Management Model (Grebner et al., 2007). In this model, a task can be *New*, *Running*, *Suspended*, *Completed*, *Terminated*, *Finalized* or *Archived*. The only model available at the moment is the *TaskModel* which is used to categorize a task.

The **resource dimension** contains concepts for representing resources on the computer desktop. Examples of possible *Resource* are *File*, *TextDocument*, *Presentation*, *Spreadsheet*, *E-Mail*, *Folder*, *Person*, *Location*, *Organization and OnlineResource*. Relations can be defined between concepts of the resource dimension and of the action dimension for modeling on which resources what kind of user actions are executed, via the objecttype property isActionOn.

The **user dimension** contains only the *User* and *Session* concepts. It is related to the action dimension in the sense that each *Action* is associated with a *User* via the objecttype relation *hasUser*. The *Session* concept represents session and user login information.

The **information need dimension** represents the context-aware pro-active information delivery aspect of the UICO. An information need is detected by a set of fixed rules based on the available context information. The *InformationNeed* concept has properties to define the accuracy of the detection and the importance to fulfill the information need in a certain time-frame. For details about information need detection it is referred to (Rath et al., 2007). This dimension is not used during our wiki study.

The **application dimension** is a "hidden" dimension because it is not modeled as concepts in the UICO. This dimension is present is such a way that each user interaction happens within the focus of a certain application, e.g., the user's desktop, Microsoft Word or the Microsoft Windows Explorer. The Event concept holds the information about the user interaction with the application by the datatype properties *hasApplicationName* and *hasProcessId*. Standard applications that run on the Microsoft Windows desktop normally consist of graphical user interface (GUI) elements. Also console applications have GUI elements such as the window itself, scroll bar(s) and buttons for minimizing, maximizing and closing the application. Most of the GUI elements have an associated accessibility object[1] which can be accessed by context sensors. Datatype properties of the *Event* concept hold the data about the interactions with GUI elements. A resource is normally accessed and manipulated by the user within an application hence there is a relation between the resource dimension and the application dimension. This relation is indirectly captured by the relation between the resource dimension and the action dimension, i.e., by the datatype property *hasApplicationName* of the *Event* concept. For a user it is not convenient to manually enter the data about her context on such a fine-granular level. Hence semi-automatic and automatic mechanisms are required to ease the process of 'populating' the user interaction context ontology.

4.4 User Interaction Context Ontology Population

The contextual information sent by the context sensors is used as a basis for populating the context ontology, i.e. instantiating its concepts. The *Event* concept can be directly instantiated by the sensor data. In order to instantiate the *EventBlock* concept, events have first to be aggregated, using application-specific as well as generic static rules and heuristics. As an event-block represents a sequence of events associated with the same resource, this aggregation process heavily relies on the resource discovery process.

We use three different techniques for discovering resources. (i) The *regular expression* approach identifies resources in the sensor data based on specific character sequences predefined as regular expressions. This is used to identify files, folders, web links and email addresses for example. (ii) The *information extraction* approach extracts person, location and organization entities in text-based elements of the sensor data, using the *KnowMiner framework (*Granitzer 2008). (iii) The *direct resource identification* approach finds data about a potential resource directly in the sensor data, and build the resource by directly mapping certain fields of the sensor data to properties of the Resource concept. These three techniques produce what we call *used resources*, in the sense that the user has interacted with them. We are also interested in unveiling relations among these *used resources*, or between these resources and other resources. We say that a resource is an included resource if its content is part of the content of another resource. A resource is *referenced resource* if it is mentioned and identified in the content of another resource (e.g. names of persons, locations and organizations, paths of folders and files, URLs of web pages and email addresses). For the wiki environment the third type of resource discovery, the direct resource identification, plays a key role. The special wiki context sensor as described in the previous section provided the information on which elements of the wiki edit page or on which argument the user was working on.

[1] Microsoft Active Accessibility, http://msdn.microsoft.com/en-us/accessibility/

The information about these elements sent by the context sensor was used to directly construct resources in the ontology with a unique URI.

A rule-based aggregation of user actions into tasks might be a reasonable approach for well-structured tasks, such as administrative or routine tasks. But is obviously not appropriate for tasks that involve a certain freedom and creativity in their execution which is the case for text editing and text manipulation. To handle such tasks the idea is to automatically extract tasks from the information available in the using context ontology by means of machine learning techniques. Once detected, these tasks will also populate the ontology.

4.5 Feature Engineering

50 features were engineered based on the concepts and relations of the user interaction context ontology (UICO). We have defined 50 features that can be grouped in six categories: (i) ontology structure, (ii) content, (iii) application (iv) resource, (v) action and (vi) switching sequences. The *ontology structure category* contains features representing the number of instances of concepts and the number of datatype and objecttype relations used per task. The *content category* consists of the content of task-related resources, the content in focus and the text input of the user. The *application category* contains the classical window title feature (Oliver et al. 2006; Shen et al., 2006; Lokaiczyk et al., 2007; Granitzer et al., 2008) the application name feature (Granitzer et al., 2008) and graphical user interface elements (*accessibility objects*[2]) features. The *resource category* includes the complete contents and URIs (URLs) (Shen et al., 2006) of the used, referenced and included resources, as well as a feature that combines all the metadata about the used resources in a 'bag of words'. The *action category* represents the user interactions and contains features about the interactions with applications (Granitzer et al., 2008), resources types, resources, key input types (navigational keys, letters, numbers), the number of events and event blocks, the duration of the event blocks, and the time intervals between event blocks. The *switching sequences category* comprises features about switches between applications, resources as well as event and resource types.

We use the machine learning toolkit Weka (Witten & Frank, 2005) for parts of the feature engineering and classification processes. The following steps are performed to preprocess the content of text-based features (in this sequence): (i) remove end of line characters, (ii) remove markups, e.g. \&lg and ![CDATA, (iii) remove all characters but letters, (iv) remove German and English stopwords, (v) remove words shorter than three characters. We transform text-based features into vectors of words with the *StringToWordVector* function of Weka. For numeric features, we apply the Weka *PKIDiscretize* filter to replace discrete values by intervals. A complete listing of all features with a short description is given in Table 1.

[2] Microsoft Active Accessibility: http://msdn.microsoft.com/en-us/accessibility/

Id	Feature	Id	Feature	Id	Feature
	Action Category	20	semantic type	37	UICO objecttype relation
1	EB duration	21	action type of E		**Resource Category**
2	resource types interaction		**Application Category**	38	used resource content
3	control input keys	22	accessibility object name	39	resource content
4	number of Es/EBs	23	accessibility object description	40	used resource metadata
5	included resource interaction	24	accessibility object role	41	referenced resources
6	referenced resource interaction	25	accessibility object role description	42	used resources
7	resource interaction	26	accessibility object value	43	included resource content
8	letter input keys	27	accessibility object help	44	included resources
9	task duration	28	accessibility object help topic	45	referenced resource content
10	applications interaction	29	application name		**Switching Sequences Category**
11	navigation input keys	30	window title	46	application switch sequence 2
12	EB resource interaction	31	raw event source	47	E type switch sequence 2
13	mean EB duration		**Content Category**	48	E level resource switch sequence 2
14	mean time between EBs	32	content of EB	49	E & EB resource switch sequence 2
15	used resource interaction	33	content in focus	50	E & EB resource type switch sequence 2
16	action element of E	34	user input		
17	median time between EBs		**Ontology Structure Category**		
18	number input keys	35	UICO concept		
19	median EB duration	36	UICO datatype relation		

Table 1: This table shows all the 50 features classified into 5 feature categories. All the features were extracted based on the user interaction context ontology (UICO). E and EB stand for event and event block respectively.

4.6 Task Detection

The fourth step of the ontology-based task detection pipeline is to detect the task based on the features engineered from the populated UICO. In case of this study, for each transformation of a study participant a new training/class instance is built. The training instances are given to machine learning algorithms to train/build a classification model and then to decide to which class another training instance belongs to.

5 Results

The participants were 10 graduate students from Germany their mean age was 25.30 (*SD*=0.51). 4 of these were women, 6 were men. During the wiki study we recorded a dataset of 158 editing tasks. Each participant saved on avarage 15.8 (*SD*=7.29, min=7, max=26) editing tasks.

5.1 Expert Ratings & User Ratings

As reported above the correlations between the two independent experts for each transformation were on a high level. We understand this as evidence for the validity of the expert ratings and as a hint that it is possible to estimate one single transformation on the basis of the concept of accommodation and assimilation. However, there was also a high correlation between assimilation and accommodation for the expert ratings: $r(156)=.826$, $p<.01$). This is basically in line with the results from Moskaliuk et. al (2009) who reported correlations between assimilation and accommodation on a medium level. On basis of this result we assume that assimilation and accommodation processes are tightly connected. One can only integrate new information in the existing wiki on the wiki-tab if as a first step new information is added from the info-tab to the wiki-tab. In our setting assimilation is a precondition for accommodation and its therefore not possible to measure this two processes independently.

We expected a high correlation between expert and user ratings. Contrary to our expectations the correlations between user ratings and the (average) expert-ratings for assimilation are non-significant: $r(156)=.112$, $p<.01$; However, looking at the correlation between expert and user rating for single user we found a high variance from $r_{user3}(12)=-.611$ $p<0.05$ to $r_{user2}(17)=.783$, $p<0.01$.

The correlations for accommodation between user ratings and the (average) expert ratings are significant but only on a medium level: $r(156)=.436$, $p<.01$. The results regarding the correlation for single user show also a high variance from $r_{user9}(6)=-.250$, n.s to $r_{user10}(5)=.810$, $p>.05$ but are mostly on a medium positive level.

The low correlations between user and expert ratings for assimilation and accommodation leads to the assumption that the participants are, against our assumption, not able to rate their own transformation as assimilation or accommodation. One possible explanation is the different granularity of the participants' transformations (the variance of transformation per participants reaches from 7 to 26), which made it hard to rate it in an appropriate way. This argumentation is supported by the overall correlation between assimilation and accommodation for the user ratings: $r(156)=-.194$; $p<0.05$. The range of the correlations for single participants shows also a high and non-systematic variance.

As a consequence of these results we decided to build the classes for the machine learning pipeline based on the average (accommodation) ratings of the two experts. This is described in detail in the following sections.

5.2 Automatic Detection of Accommodation Levels

Fore each editing tasks of a study participant a new training/class instance for the machine learning algorithm is built based on the recorded usage data. During the study we recorded a dataset of 158 transformations. For 19 transformations the log files show now difference between the two versions and were excluded form further analysis. Based on the average accommodation ratings of the two experts we divided the 139 transformations in equal thirds using the terzils. The leads to three classes; we computed the boundaries of the classes with low<=1.5; medium<=2.5; high>2.5 (based on the average accommodation ratings of the two experts). This leads to the following 3 classes representing different levels of accommodation: high accommodation (54 class instances), medium accommodation (55 class instances), and low accommodation (30 class instances).

5.2.1 Evaluation of the Machine Learning Pipeline

In order to evaluate influencing factors to this classification problem we varied the following parameters: (i) the learning algorithm, (ii) the set of used features and (iii) the number of attributes generated from the features. Furthermore, the set of used features is varied by including (i) each feature individually, (ii) each feature category individually, (iii) all feature categories or (iv) the *top k* best performing single features, with $k \in \{2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 20\}$.

We studied the Naive Bayes (NB), Linear Support Vector Machine (SVM) with cost parameter $c \in \{2^{-5}, 2^{-3}, 2^{-1}, 2^{0}, 2^{1}, 2^{3}, 2^{5}, 2^{8}, 2^{10}\}$[3], J48 decision tree (J48) and k-Nearest Neighbor (KNN-k) with $k \in \{1,5,10,35\}$ algorithms. For each

[3] The interval was chosen according to the libSVM guide at: http://www.csie.ntu.edu.tw/~cjlin/libsvm/

classifier/learning algorithm l ϵ L, for each feature category and each feature *f* ϵ *F* we selected the *g* attributes having the highest *Information Gain (IG)* value to obtain our dataset. As values for *g* we used 50 different measure points. Half of them were equally distributed over the available number of attributes with an upper bound of 5000 attributes. The other half was defined by *G* = {3, 5, 10, 25, 50, 75, 100, 125, 150, 175, 200, 250, 300, 500, 750, 1000, 1500, 2000, 2500, 3000, 3500, 4000, 5000, 7500, 10000}. Which attributes are finally used by the classifiers depends on the *attribute selection* algorithm (*IG* in our case). We measured the accuracy (*a*) of the used algorithms (*l*), the number of attributes (*g*), the micro precision (*p*) and micro recall (*r*).

5.2.3 Results of the Classification

In Table 2 an overview of the best results about the about the performance of detecting transformations with low, medium and high accommodation by stratified 10-fold cross-validation for each feature category, for all feature categories combined, each single feature as well as the k top performing single features. The evaluation results show that a combination of four UICO features achieved an accuracy of **79.12%** with the Naive Bayes algorithm for detecting low, medium and high accommodation tasks. In comparison, the probability of randomly guessing whether a transformation belongs to the low, medium and high accomodation class is 39.57% on our dataset. This means that by applying the ontology-based task detection approach significantly improves the accuracy. A detailed discussion of the features is given in the following:

Feature Categories: The best performing feature category was the *content category* with an accuracy of 77.03% and the NB algorithm (g=175, p=0.86, r=0.77). The combination of all 50 features was closely behind with an accuracy of 74.12% with the same algorithm but required 1500 attributes (p=0.84, r=0.72). The *action category* achieved the third rank in the category ranking with an accuracy of 69.07% with KNN-10 algorithm and with 25 attributes (p=0.80, r=0.66). The difference between the best and the worst accuracy values was 23.68.

Single Features: The best performing feature was the content in focus, with an accuracy of 74.07% with the NB algorithm and with 75 attributes (p=0.84, r=0.72). The content in focus feature belongs to the content category which was the best performing category with 77.03%. The second best performing feature with an accuracy of 70.44% was the control input keys feature with only one attribute and the SVM algorithm (p=0.81, r=0.66). This attribute represents the number of times a control key, e.g. SHIFT, RETURN or INSERT, has been pressed by the user during the editing tasks. The distribution of values for the attribute showed that control keys were less used for low accommodation tasks than for high accommodation tasks. The values for the medium accommodation tasks are distributed in a balanced way. The user input feature ranked at third place among the best performing single features with an accuracy of 69.89% with the KNN-5 algorithm and with 10 attributes (p=0.80, r=0.66).

Top k Features: The best combination of the best performing single features was the top k=4 feature combinations with an accuracy of 79.12% with the NB algorithm and with 100 attributes (p=0.88, r=0.79). The third best combination was the top k=3 combination which only utilized the best three performing single features but also achieved a high accuracy value of 78.53% (l=NB, g=100, p=86, r=76). The top k=4 and the top k=3 performed almost as well in terms of accuracy with only 0.59% difference. All top k feature combination except the worst performing one outperformed all single features, all feature categories as well as the combination of all 50 features. This shows that not all features are helpful in the classification decision.

Set	R_S	f	l	g	a	p	r	R_G
	1	Content Cat.	NB	175	77.03	0.86	0.77	8
	2	All Categories	NB	1500	74.12	0.84	0.72	11
	3	Action Cat.	KNN-10	25	69.07	0.80	0.66	18
Feature	4	Resource Cat.	SVM-$C = 2^1$	3000	68.41	0.77	0.60	21
Categories	5	Ontology Str. Cat.	SVM-$C = 2^5$	5	64.12	0.74	0.57	38
	6	Switching Seq. Cat.	KNN-5	10	64.07	0.77	0.60	40
	7	Application Cat.	NB	10	53.35	0.65	0.50	48
	1	content in focus	NB	75	74.07	0.84	0.72	13
	2	control input keys	SVM-$C = 2^8$	1	70.44	0.81	0.66	15
	3	user input	KNN-5	10	69.89	0.80	0.66	16
	4	res. interact.	J48	175	69.78	0.80	0.67	17
	5	used res. interact.	J48	5	69.01	0.78	0.63	19
	6	resource content	SVM-$C = 2^{-2}$	75	69.01	0.78	0.61	20
	7	UICO concept	J48	104	68.35	0.79	0.64	22
	8	referenced res. interact.	SVM-$C = 2^1$	298	67.69	0.78	0.62	23
Single	9	E type switch seq. 2	J48	10	67.69	0.78	0.61	24
Features	10	res. types interact.	J48	14	67.58	0.78	0.62	25
	11	used res. metadata	SVM-$C = 2^{10}$	1000	66.98	0.76	0.59	26
	12	EB duration	J48	9	66.21	0.77	0.60	27
	13	applications interact.	SVM-$C = 2^{-1}$	3	66.15	0.75	0.57	28
	14	included res. interact.	SVM-$C = 2^{-1}$	75	66.04	0.75	0.58	29
	15	UICO objecttype rel.	J48	57	65.60	0.77	0.63	30
	16	nr. of E/EB	J48	10	65.49	0.76	0.61	31
	17	navigation input keys	NB	1	64.89	0.74	0.58	32
	18	included res. content	SVM-$C = 2^5$	50	64.78	0.77	0.63	33
	19	EB res. interact.	SVM-$C = 2^{-1}$	125	64.78	0.74	0.55	34
	20	referenced res. content	SVM-$C = 2^{-5}$	50	64.73	0.77	0.63	35
	1	Top $k = 4$	NB	100	79.12	0.88	0.79	1
	2	Top $k = 10$	NB	200	78.52	0.87	0.79	2
	3	Top $k = 3$	NB	100	78.52	0.86	0.76	3
	4	Top $k = 8$	NB	175	78.30	0.87	0.78	4
Top	5	Top $k = 6$	NB	150	77.69	0.86	0.77	5
k	6	Top $k = 7$	NB	150	77.64	0.86	0.77	6
Features	7	Top $k = 9$	NB	175	77.09	0.86	0.77	7
	8	Top $k = 5$	NB	125	76.21	0.84	0.75	9
	9	Top $k = 2$	NB	125	75.55	0.85	0.73	10
	10	Top $k = 20$	NB	750	74.07	0.84	0.73	12
	11	Top $k = 15$	NB	1000	73.46	0.84	0.73	14

Table 2: Overview of the best results about the performance of detecting tasks with low, medium and high accommodation by stratified 10-fold cross-validation for each feature category, for all feature categories combined each single feature as well as the k top performing single features. The learning algorithm (l), the number of attributes (g), the micro precision (p), the micro recall (r), the ranking in the corresponding section (RS) and across sections (RG) is also given.

5.2.4 Comparison with other Task Detection Approaches

The evaluation results show that we achieved an **accuracy of 79.12%** and a **precision of 0.86** for classifying transformations to low, medium and high accommodation levels. In comparison, by randomly guessing to which class a transformation belongs to someone would have a probability of 39.57% on our dataset. This is a good result in respect to the granularity of the task we try to detect: a transformation of a single wiki page. With granularity we mean here (i) the short duration of a transformation, (ii) user interactions on only one webpage and in one application, i.e., the browser.

Dataset: For us a "task" is a transformation of a single wiki page that can be categorized into a low, medium and high accommodation. The boundaries of the tasks were freely chosen by the participants of the study. Tasks used in other task detection experiments were higher level. Examples are "buying a book" task (Oliver et al., 2006), business tasks (Lokaiczyk et al., 2007) or tasks as in (Granitzer et al., 2008) comprised of email handling, paper writing, research, documentation or information collection.

Detection Performance: Existing task detection approaches focus on more higher level tasks and report similar performances: an accuracy of 76% with a precision of 0.49 (Oliver et al., 2006), 85% with a precision of (Lokaiczyk et al., 2007), an accuracy of 74.51% with a precision of 0.91 (5 classes) and an accuracy of 76.42% with a precision of 0.90 (4 classes) (Granitzer et al., 2008) and a precision of 0.8 (96 and 81 classes) (Shen et al., 2006).

Features: The most popular features identified for having a high discriminative power among tasks are the window title feature (Oliver et al. 2006; Shen et al., 2006; Lokaiczyk et al., 2007; Granitzer et al., 2008), the file path/web page url (Shen et al., 2006;), and the content in focus feature (Granitzer et al., 2008). The task detection results on our dataset show that the `window title` and the `file path/web page url` features do not work well. This is because of the fine-granularity of the tasks we were trying to detect. A whole transformation of a study participant only involved the Wiki simulation environment web page, i.e., a single flash application, and the browser.

Attributes: In terms of attributes used for training the machine learning algorithms an interval of 200-300 attributes is suggested to be sufficient by (Shen et al., 2006; Granitzer et al., 2008). Our results agree that only a small ratio of attributes is required to successfully detect a specific accommodation task.

Classifiers: In the task detection experiments reported in (Lokaiczyk et al., 2007) the SVM learning algorithm was mentioned as the one with the highest achieved accuracy. In (Granitzer et al., 2008) the good performance of the SVM learning algorithm was confirmed and the high accuracy achieved by the KNN learner highlighted. On our datasets the SVM and the KNN learner showed also good results but was beaten by the Naive Bayes algorithm.

6. Conclusion

In the research presented here we have developed a tool that is able to analyze how people interact with different artifacts in order to create a text. This context-analysis tool provided a method to measure processes of accommodation and assimilation in an automatic way. Our research provided also a more detailed differentiation of knowledge maturing processes. This is an important aspect in the larger context of social media and knowledge management, since the study provided some insights into the processes that lead to accommodation of knowledge. Consequently, our results may help to improve the understanding on informal learning and knowledge maturing.

The first goal of our study was to measure the quality of text as an indicator for knowledge building within communities and to identify 'accommodation patterns' (as opposed to 'assimilation patterns'). Based on the ratings of two experts we could distinguish among three classes of low, medium and high accommodation. The ontology-based task detection approach – that was applied in order to answer this identification question – finally yielded an identification rate of 79.12%. This was the first step for developing methods for the automatic detection of accommodation processes as indicators for knowledge building and knowledge maturing. The results of our study led to the following conclusions:

- It is possible to detect the accommodation level of a wiki text transformation with only a small ratio of features. This allows an efficient detection of text maturing and knowledge building on the basis of user behavior. The features achieving the highest accuracy values, however, were related to the content of the wiki article and the users' interactions with this content. Since content-related features are often domain dependent, it might be the case that they are not well generalizable with regard to other domains.
- The results led to a better understanding of accommodation processes. Expert ratings according assimilation and accommodation correlate on a high level. This means that the two constructs are not independent from each other. From our perspective the process of accommodation is central for text maturing and knowledge building. It seems to be a proper strategy for further studies to focus on accommodation as main concept.

The second research question aimed at answering the question how it can be ensured that knowledge workers write good texts and contribute to the development of knowledge. Here, the results of the user ratings are important. One could assume that users are not able to monitor their own work very well and can hardly decide, whether their edits will lead to a maturing of the text. Thus, it could be a good strategy to use the classification results (and the reading scores) as a feedback for users that guide them to become successful writers.

In a next step we would like to use the automatic way of analyzing accommodation processes for further studies that may identify possible support mechanisms to improve and/or encourage accommodation (towards supporting knowledge maturing). We also plan to implement our tool as a MediaWiki extension and to apply the task-detection method to real-world settings with real communities working on shared digital artifacts. In addition to the online detection of knowledge-building processes a post-hoc analysis of revision histories of collaboratively written text would be of great interest. This is would allow to analyze existing text corpora in the Internet, e.g. of the Wikipedia.

Altogether, we hope that our previous findings and our considerations concerning potential future studies will stimulate other researchers to initiate corresponding research on their own that might shed some light on the understanding of processes of knowledge building and knowledge maturing.

References

Baldauf, M., Dustdar, S., & Rosenberg, F. (2007). A survey on context-aware systems. *International Journal of Ad Hoc and Ubiquitous Computing, 2*(4), 263–277.

Coutaz, J., Crowley, J. L., Dobson, S., & Garlan, D. (2005). Context is key. *Communications of the ACM, 48*(3), 49–53.

Cress, U., & Kimmerle, J. (2008). A systemic and cognitive view on collaborative knowledge building with wikis. *International Journal of Computer-Supported Collaborative Learning, 3*(2), 105–122.

Dey, A. K., Abowd, G. D., & Salber, D. (2001). A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human-Computer Interaction, 16*(2), 97–166.

Dragunov, A. N., Dietterich, T. G., Johnsrude, K., McLaughlin, M., Li, L., & Herlocker, J. L. (2005). TaskTracer: a desktop environment to support multi-tasking knowledge workers (pp. 75–82). *IUI '05: Proceedings of the 10th International Conference on Intelligent User Interfaces.* New York, NY, USA: ACM.

Erkens, G., Jaspers, J., Prangsma, M., & Kanselaar, G. (2005). Coordination processes in computer supported collaborative writing. *Computers in Human Behavior, 21*(3), 463–486.

Flesch, R. (1948). A new readability yardstick. *Journal of Applied Psychology, 32*, 221–233.

Granitzer, M. (2008). *KnowMiner - Konzeption & Entwicklung eines generischen Wissenserschließungsframeworks.* Saarbrücken: VDM Verlag Dr. Müller.

Grebner, O., Ong, E., Riss, U., Brunzel, M., Bernardi, A., & Roth-Berghofer, T. (2007). NEPOMUK Deliverable D3.1: Task management model.

Gunning, R. (2004). Plain language at work newsletter. Retrieved from http://www.impact-information.com/impactinfo/newsletter/plwork08.htm

Gutschmidt, A., Cap, C. H., and Nerdinger, F. W. (2008). Paving the path to automatic user task identification. *Workshop on Common Sense Knowledge and Goal-Oriented Interfaces, IUI '08*.

Johnson, S. (2001). *Emergence: the connected lives of ants, brains. cities, and software*. New York, NY: Scribner.

Lave, J., & Wenger, E. (1991). Situated learning: legitimate peripheral participation. Cambridge: University Press.

Lindstaedt, S. N., Ley, T., Scheir, P., & Ulbrich, A. (2008). Applying scruffy methods to enable work-integrated learning. *Upgrade: The European Journal of the Informatics Professional, 9*(3), 44–50.

Lokaiczyk, R., Faatz, A., Beckhaus, A., & Goertz, M. (2007). Enhancing just-in-time e-learning through machine learning on desktop context sensors. CONTEXT '07: Proceedings of the 6th International and Interdisciplinary Conference on Modeling and Using Context. *Lecture Notes in Computer Science*: *Vol. 4635 (pp.* 330–341). Berlin, Heidelberg: Springer-Verlag.

Luhmann, N. (1986). The autopoiesis of social systems. In F. Geyer, & J. van der Zouwen (Eds.), *Sociocybernetic paradoxes: observation, control, and evolution of self-steering systems* (pp. 172–192). London: Sage Publications.

Luhmann, N. (1995). *Social Systems.* Stanford, CA: Stanford University Press.

Luhmann, N. (2006). System as difference. *Organization, 13*, 37–57.

Mentzas, G. , Kafentzis, K., & Georgolios, P. (2007). Knowledge services on the semantic web. (Vol. 50, pp. 53–58). New York, NY: ACM. Received from http://www.imu.iccs.gr/Papers/J54-CACM-Mentzas.pdf

Moskaliuk, J., & Kimmerle, J. (2009). Using wikis for organizational learning: functional and psycho-social principles. *Development and Learning in Organizations, 23*(4), 21–24.

Moskaliuk, J., Kimmerle, J., & Cress, U. (2009). Wiki-supported learning and knowledge building: effects of incongruity between knowledge and information. *Journal of Computer Assisted Learning, 25*(6), 549–561.

Piaget, J. (1977a). *The development of thought: equilibration of cognitive structures.* New York, NY: Viking.

Piaget, J. (1977b). Problems of equilibration. In M. H. Appel, & L. S. Goldberg (Eds.), *Topics in cognitive development* (Vol. 1, pp. 3–14). New York, NY: Plenum.

Raitman, R., Augar, N., & Zhou, W. (2005). Employing wikis for online collaboration in the e-learning environment: case study. *Proceedings – 3rd International Conference on Information Technology and Applications, ICITA 2005, Sydney, Australia* (pp. 142–146). New York, NY: ACM.

Rath, A. S., Devaurs, D., & Lindstaedt, S. N. (2009a). UICO: An ontology-based user interaction context model for automatic task detection on the computer desktop. *Proceedings of the 1st Workshop on Context, Information and Ontologies (CIAO 2009), workshop as part of the ESWC 2009, Heraklion, Greece.* New York, NY: ACM.

Rath, A. S., Devaurs, D. & Lindstaedt, S. N. (2009b). Detecting real user tasks by training on laboratory. *Proceedings of Exploitation User Attention Metadata (EUAM '09), Lecture Notes in Informatics.* Received from http://www.know-center.tugraz.at/download_extern/papers/arath_et_al_euam2009.pdf

Rath, A. S., Weber, N., Kroell, M., Granitzer, M., Dietzel, O. & Lindstaedt, S. N. (2008) Context-aware knowledge services. *Workshop on Personal Information Management (PIM2008) at the 26th Computer Human Interaction Conference*

(CHI2008), Florence, Italy. Recieved from http://www.know-center.tugraz.at/download_extern/papers/pim2008-rath-etal.pdf

Reinhold, S. (2006). Wikitrails: augmenting wiki structure for collaborative, interdisciplinary learning. *Proceedings of WikiSym '06 – International Symposium on Wikis 2006* (pp. 47–58). New York, NY: ACM.

Schmidt, A. (2005). Knowledge maturing and the continuity of context as a unifying concept for knowledge management and e-learning. *Proceedings of the Fifth International Conference on Knowledge Management (I-KNOW 05), Graz, Austria.*

Shen, J., Li, L., Dietterich, T. G., & Herlocker, J. L. (2006). A hybrid learning system for recognizing user tasks from desktop activities and email messages. *IUI '06: Proceedings of the 11th International Conference on Intelligent User Interfaces* (pp. 86–92). New York, NY: ACM.

Shen, J., Irvine, J., Bao, X., Goodman, M., Kolibaba, S., Tran, A., Carl, F., Kirschner, B., Stumpf, S., & Dietterich, T. G. (2009). Detecting and correcting user activity switches: algorithms and interfaces. *IUI '09: Proceedings of the 13th International Conference on Intelligent User Interfaces* (pp. 117–126). New York, NY: ACM.

Si, L., & Callan, A. J. (2001). Statistical model for scientific readability. *Proceedings of CIKM, Atlanta, Georgia, USA* (pp. 574–576). New York, NY: ACM.

Strang, T., & Linnhoff-Popien, C. (2004). A context modeling survey. *Workshop on Advanced Context Modelling, Reasoning and Management, UbiComp 2004*.

Vygotsky, L. S., Cole, M., John-Steiner, V., Scibner, S., & Souberman, E. (1978). Mind in society: The development of higher psychological processes. Cambridge, Mass.: Harvard University Press.

Witten, I. H., & Frank, E. (2005). *Data Mining: Practical machine learning tools and techniques* (2nd ed.). San Francisco, CA: Morgan Kaufmann.

Wolpers, M., Najjar, J., Verbert, K., & Duval, E. (2007). Tracking actual usage: the attention metadata approach. *Educational Technology & Society*, *10*(3), 106–121.