User Model Services for Adaptive Work-integrated Learning

Master's Thesis

at

Graz University of Technology

submitted by

Günter Beham

Knowledge Management Institute (KMI),
Graz University of Technology
A-8010 Graz, Austria

May 2011

Advisor: Univ.-Doz. Dr. Stefanie Lindstaedt

TU Graz

Abstract

A major challenge when designing software for work-integrated learning (WIL) is that the system provides real-time learning support adapted to a user's situation and background knowledge (adaptive system). In order to adapt to the user's situation and characteristics, a user model is needed that is continuously updated to the learning progress. While numerous adaptive systems have been designed for school and university settings, hardly any adaptive system exists for supporting WIL. The aim of my master's thesis was to design and develop a WIL User Model, WIL User Model Services, and a software architecture for a WIL system that provides learning support that is adapted to a user's task and background knowledge, that is able to re-use real knowledge resources, and that is integrated in the users' real computational environment. Requirements for the system were derived from WIL theory, and from existing use cases. The analysis of requirements indicated that three kinds of functionality seem crucial in order to support WIL: Unobtrusive user model maintenance, recommendation of learning content, and recommendation of knowledgeable people. Within this thesis, these functionalities were conceptualised using different kinds of user model services (logging, production, inference and control services). Together, these build the WIL User Model Services (WIL UMS). The WIL software architecture and WIL UMS were prototypically implemented in the adaptive WIL system APOSDLE. APOSDLE's user model is automatically maintained by tracking 'Knowledge Indicating Events'. Based on the information in the user model, APOSDLE recommends real resources and knowledgeable colleagues. APOSDLE and its WIL UMS have been instantiated for four real companies to provide intelligent adaptive WIL support. APOSDLE and its WIL UMS are integrated in the users' work environments.

Zusammenfassung

Die zentrale Herausforderung für die Entwicklung von Software für arbeitsintegriertes Lernen (work-integrated learning, WIL) ist es, Lerninhalte bereitzustellen, die an die situativen Gegebenheiten und das Vorwissen der NutzerInnen angepasst sind (adaptive Systeme). Um Adaptivität zu realisieren ist ein Benutzermodell (User Model) erforderlich, das kontinuierlich an den Lernfortschritt angepasst wird. Im Gegensatz zum Schul- und Universitätskontext existieren kaum adaptive Systeme zur Unterstützung von WIL. Ziel meiner Masterarbeit war es, ein WIL User Model, WIL User Model Services und eine Software-Architektur zur Unterstützung von WIL zu entwickeln. Das WIL System sollte sich an die Arbeitsaufgabe und das Vorwissen der BenutzerInnen anpassen, reale Arbeitsdokumente als Lerninhalte benützen und in die Arbeitsumgebung der Benutzer integriert sein. Anforderungen für das System wurden einerseits aus der Theorie zu WIL und andererseits aus existierenden Use Cases abgeleitet. Die Anforderungsanalyse ergab, dass drei Arten von Funktionalität zentral für die Unterstützung von WIL erscheinen: Non-invasive Wissensdiagnose, Empfehlungen von Inhalten und Empfehlungen von ExpertInnen. In meiner Masterarbeit wurden diese Funktionalitäten über verschiedene Arten von User Model Services konzeptualisiert (Logging, Production, Inference und Control Services), die gemeinsam die WIL User Model Services (WIL UMS) bilden. Die WIL UMS wurden prototypisch im adaptiven WIL System APOSDLE implementiert. APOSDLE's Benutzermodell wird über Log Daten ("Knowledge Indicating Events") automatisch aktualisiert. Ausgehend vom Benutzermodell empfiehlt APOSDLE reale Arbeitsdokumente und ExpertInnen. APOSDLE und die WIL UMS wurden als intelligente Lösung zur Unterstützung von WIL in vier Unternehmen installiert, und sind in die Arbeitsumgebung der BenutzerInnen integriert.

STATUTORY DECLARATION

I declare that I have authored this thesis independently, that I have not used other than the declared sources / resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

Place, Date

Signature

EIDESSTATTLICHE ERKLÄRUNG

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommene Stellen als solche kenntlich gemacht habe.

Ort, Datum

Unterschrift

Acknowledgements

Working on my master's thesis was definitely the longest project with a lot of exciting opportunities to get involved in. I was fascinated to be part of the EU research project APOSDLE with the ability to do research in a large interdisciplinary team, and to deploy research results in real work environments. After the great success of the APOSDLE project, I am very happy to finish my master's thesis.

Foremost, I would like to thank my supervisor Univ.-Doz. Dr. Stefanie Lindstaedt, manager of the division "Knowledge Services" at the Know-Center. She introduced me to the exciting field of technology-enhanced learning, and the work in large EU projects. With her help and backing, I was able to work on different topics and in challenging positions at the Know-Center.

I wish to thank Prof. Dr. Klaus Tochtermann, Professor at the Knowledge Management Institute at Graz University of Technology for his strong support in early phases of my master's thesis.

My sincere thanks go to the APOSDLE team in Graz with whom I closely worked together: Armin Ulbrich, Peter Scheir, Stefanie Lindstaedt, Barbara Kump, Viktoria Pammer, Hermann Stern, Peter Kraker, Tobias Ley, Daniel Resanovic, Christin Seifert, Philip Hofmair, Helmut Leitner, Robert Hoffmann, Franjo Bratic, Odo Benda, Nermin Katjazovic, Ilire Hasani-Mavriqi, and Victor Garcia-Barrios.

I wish to thank all my colleagues at the Know-Center for having a great time while working, developing, designing, and discussing.

I owe special thanks to my parents who supported me during my study not only financially but also with their trust and love. To them I dedicate my master's thesis.

Lastly, and most importantly, I would like to thank my partner Barbara for her love, for motivating me in difficult times, and for her tireless support during the process of finalising my master's thesis.

Guenter Beham
Graz, Austria, May 2011

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Current Situation

Today's learning is not anymore dedicated to school, university, or some formal trainings but is considered to happen continuously throughout the entire life. This tendency is referred to as lifelong learning (Fischer, 2000). Specifically, in our knowledge society, we are facing the necessity of continuously learning at work.

The topic of workplace learning has received quite some attention in the recent past. According to Eraut (2004), Eraut and Hirsh (2007), Billett (2001), Smith (2003), Lave and Wenger (1991) and many others, most learning at work happens informally, either through own experience or in interaction with colleagues. Moreover, workplace learning mainly takes place without clear intention to learn, and is still less based on dedicated training plans.

However, until more recently, these characteristics of workplace learning have not been reflected in current strategies of vocational training: Still, a vast amount of money is spent on formal trainings of the workforce. One of the main problems of formal training is caused by not considering requirements of real workplaces. Ley et al. (2005) argued that such formal trainings are not efficient as there is only little knowledge transfer from the training situation to the situation where the knowledge needs to be applied. Moreover, there is a cognitive disconnection between a work space, learning space, and knowledge space within an organisation, as well as a structural disconnection between these three spaces: Each of these spaces is implemented on different technical systems.

In order to overcome these disconnections that impair learning transfer from the situation where knowledge is acquired to the situation where knowledge is applied, Lindstaedt et al. (2009a) introduced the paradigm of Work-integrated Learning (WIL). The authors clearly state that working and learning cannot be separated but are integrated and happen at the same time. Moreover, they define learning as a dimension of knowledge work: Knowledge work thereby is seen as a continuum that may range from task-focused to learning-focused activities. As a consequence, instead of training workers 'offsite', learning must be supported 'on-site', closely related to the current work practices.

eLearning technologies are seen as powerful means to bring learning to the workplace. However, traditional eLearning systems do work well for past assumptions about how learning happens (at school, at universities) but get to their limits when trying to support learning during every day work. Lindstaedt et al. (2010) identified three major challenges for systems supporting WIL:

- **Real-Time Learning:** WIL-Systems should raise awareness for learning opportunities relevant a user's current work task, and need to be adapted to a user's work context and his or her experiences.

- **Real Knowledge Resources:** WIL-Systems should dynamically recommend knowledgeable people an resources within an organisation.

- **Real Computational Environment:** WIL-Systems should provide a variety of tools and services which are seamlessly integrated with the user's computational environment (e.g., desktop, laptop, mobile devices).

In the area of technology-enhanced learning (TEL) the first challenge - real-time learning - has been tackled by research into and development of adaptive (web-based) systems (Hutchison et al., 2006). Adaptive systems change their functionality in order to take into account the characteristics of its users. In the context of learning, the most important characteristic of users is a user's background knowledge; however, other characteristics have also been adapted to, such as a user's task or goal, interest, or location. Most of these systems origin from research into *Intelligent Tutoring Systems* (ITS) and different types of *Adaptive Hypermedia* (AH) systems. Adaptive systems contain models of the learning environment (domain model) and the users (user model) to then adapt the system's functionality to the users in different ways (Benyon and Murray, 1993). The user model is the most important component of an adaptive learning system: It represents the characteristics of the user to which the system shall adapt. Thus it must be continuously updated to the user's learning progress.

A large amount of work exists in the area of adaptive systems and user modeling (Brusilovsky and Millán, 2007), however most of the existing user-model based adaptive systems were designed and tested with focus on university settings, and little work has been done with regard to adaptive WIL. Still, most of the principles underlying user models and adaptive learning systems may also be very beneficial when thinking of how to support WIL.

To sum up, a lot of research has already been carried out to create learning systems that support users in 'learning what they need' in formal settings. However, the specific characteristics and requirements of supporting WIL have been neglected in traditional eLearning research.

1.2 Goal

It is the goal of my thesis to analyse the WIL situation and to derive a set of requirements for a user-adaptive WIL system that is based on a user model. Starting from these requirements, I will design and develop the user model, user models services and an architecture for a WIL system that supports learning at real time, learning from real content and learning within the user's 'real' computational environment.

1.3 Scope

The focus of my work will be on supporting learning at real time, i.e. adapting WIL support to the characteristics and needs of workers at their workplaces. As described above, supporting learning at real time means to adapt the presentation of learning content to the user's learning situation. According to Fischer (2001) or Kump (2010), a knowledge worker's learning situation is characterised (i) by the task at hand and (ii) the background knowledge of the user. Therefore, I will focus on these two aspects of adaptivity and not take into account other characteristics of the user (such as interest, motivation, etc.), or situation (e.g., time constraints). The scope of my master's thesis includes the review of the state of the art of adaptive learning systems in the area of technology enhanced learning. This review concentrates on design issues such as data collected from users, adaptation and recommendation techniques, and system architecture. Other issues such as authoring tools for content creation of learning systems are not covered within my master's thesis.

1.4 Overview

Theoretical work from various scholars can serve as a first starting point for designing an adaptive WIL system. However, in order to build a system that will be used in real work environments, it is necessary to also take into account the needs and requirements of its (future) users. The approach I have chosen to design a WIL-System is based on the idea to take best practices from different fields and combine them in a new way. Therefore, my work is based on three pillars

- Concepts of adaptive learning systems
- WIL theory
- Use cases of real users

In the following sections, I will derive design implications from these three sources. At several points, I will extract requirements for the user model and user model services of a WIL System architecture and present the conceptual design that is underlying my work.

My master's thesis is structured as follows:

> Chapter 2 provides an overview of adaptive learning systems by starting to introduce the most important terms and describe the basic components almost all adaptive systems are build of. In the following, adaptation techniques are presented which evolved over the last two decades. Based on these foundations, a number of research systems are investigated in detail to see how they designed and implemented the different aspects of an adaptive learning system.

Chapter 3 introduces challenges for work-integrated learning (WIL) systems and proposes and proposes conceptual solutions. It then investigates how this architecture can be realised.

Chapter 4 introduces the adaptive WIL System APOSDLE. It derives requirements for the implementation from a list of use cases collected by the APOSDLE project. These requirements and further technical constraints of the project build the bastris of the prototypical implementation.

Chapter 5 describes in detail how I prototypically implemented various WIL User Model Services for the APOSDLE system.

Chapter 6 revisits and discusses the results of the conceptual design and technical implementation.

Chapter 7 summaries the my work on user modelling services for work-integrated learning and presents topics to follow-up.

1.5 Related Publications

Parts of this work has been already published in

- Recommending Knowledgeable People in a Work-Integrated Learning System (Beham et al., 2010b)
- Providing Varying Degrees of Guidance for Work-Integrated Learning (Lindstaedt et al., 2010)
- MyExperiences: Visualizing Evidence in an Open Learner Model (Kump et al., 2010)
- iAPOSDLE – An Approach to Mobile Work-Integrated Learning (Beham et al., 2010a)
- Non-invasive User Modeling for Recommending Knowledgeable Persons in Work-integrated Learning (Beham et al., 2009)
- Getting to Know Your User - Unobtrusive User Model Maintenance within Work-Integrated Learning Environments (Lindstaedt et al., 2009b)
- Knowledge Services for Work-integrated Learning (Lindstaedt et al., 2008b)
- A Socio-Technical Approach towards Supporting Intra-Organizational Collaboration (Aehnelt et al., 2008)
- Software Architecture for 2nd (APOSDLE Consortium, 2007b) and 3rd (APOSDLE Consortium, 2008) Prototypes
- APOSDLE Conceptual Architecture (APOSDLE Consortium, 2010b)

Chapter 2

Adapting the Learning System to the Needs of its Users

This chapter summarises the most important characeristics of adaptive systems and their application in educational learning systems. Specific considerations for adaptive WIL support are given in each section.

Considering the challenge of providing real-time support for WIL systems presented in Section 1.1, the user's level of knowledge and her work task and work context seem to be the most important features for a WIL system to take into account. The level of knowledge describes experiences and skills a user may have or may have not about a concept of the domain. The work task or work context describes a user's current goal she has to accomplish. Together, the level of knowledge and current work task provide the information about what a user knows and what he or she wants to achieve.

The question of adapting a learning system to the situation of its users is by far not new but has a long tradition in research communities around intelligent tutoring systems (ITS), and adaptive hypermedia systems(AHS).

Intelligent Tutoring Systems provide personalised learning support to its users in various ways. Well known ITS are ELM-ART Weber and Brusilovsky (2001), or the ISIS-Tutor Brusilovsky and Pesin (1994), see Section 2.4.1.

Adaptive Hypermedia Systems are based on hypertext. The aim of these systems is to provide users with guidance for navigating through the hyperspace (i.e. linked hypertext) in order to avoid that they are 'lost in hyperspace'. On of the most prominent examples of AHS is AHA! (De Bra et al. (2002), see Section 2.4.3).

At their origins, ITS (Lester et al., 2004) and AHS (Brusilovsky, 2000) have been developed in parallel. Since the 1990ies, research in these areas has been aligned. Therefore, now the basic components of these two types of systems are very similar.

Both ITS and AHS have mainly been built for formal educational contexts such as courses at schools and universities, and thus do not take into account the specific situation of learning at work. Still, the basic principles of these systems can be re-used for the WIL case, as I will show in Chapter 4.

2.1 Definition of Terms

Before diving into the details of personalised systems and their various forms, I would like to clarify some important terms used throughout this chapter.

The term 'personalisation' is used very commonly in the field of adaptive systems and its various subtypes such as educational hypermedia systems, adaptive web systems systems. Kobsa et al. (2001) defines a personalised hypermedia application as systems supporting different forms of adaptations. Kobsa defines adaptation to be happening with respect to the content, to the structure, or the way how it is presented to users. They divide the process of personalisation into three tasks. First, the 'Acquisition' task collects available information about the user to build and maintain a user model. Collected information can range from user data to usage data and data about the environment the user is working in. Based on this initial user model, the second task 'Representation and Inference' takes this data and structures it into a more formal way and draw inferences. Drawing inferences on a user model means to interpret the structured data and make assumptions about the user (e.g. Experiences in a certain topic). The third task 'Production' is responsible for generating or producing the adaptation in terms of content, structure and presentation based on the inferences. Thus, personalisation can be seen as a general process through which a system adapts to the needs of a user. Two terms are important in the context of personalisation, namely the terms 'adaptable systems' and 'adaptive systems' - these describe different ways of how systems can be tailored to the needs of its users. For both terms I follow the definitions presented by Oppermann (1994). Both adaptable and adaptive systems can be seen as a concrete form of the personalisation process.

2.1.1 Adaptable Systems

Adaptable systems offer users ways to modify or change how the system behaves, looks like, or interacts with the them. The most important aspect of adaptability is that users are always in control of what is being adapted. Taking the example of an email application, users can change the layout of the application, modify whether emails should be sent in HTML format or plain text, and 'tell' the application to show a small window whenever a new message arrives in the inbox. The email client (system) provides users with the freedom to adapt it to their personal needs. Jameson (2003) points out a possible limitation of adaptable systems: often the large number of options to choose from makes it hard for users to select the best settings for the current situation.

2.1.2 Adaptive Systems

In contrast, adaptive systems take away control from the user and perform adaptations automatically. To automate adaptations, adaptive systems utilise available information of its users such as preferences, recent activities, or properties of the environment to make a decision which adaptations fit best for the user. Taking again the previous example, the email client could evaluate the inbox and rank messages based on the frequency the user has written messages to the sender in the past. Again, this approach is limited by the fact, that users cannot influence an application's behaviour.

2.1.3 Hybrid Systems

Adaptability and adaptivity are not mutually exclusive, but often are combined in the same system. To combine the best aspects of adaptable and adaptive systems, many systems go for a hybrid solution, where some features can be modified or turned on or off, while others will automatically adapted. Taking the example, the email client could offer a preference setting allowing users to manually enable or disable (adaptability) intelligent sorting of the inbox (adaptivity).

The decision whether to just go for one form or to combine both, depends on several factors, such as the kind of function which should be adapted, user expectations, usability issues, consequences of wrong adaptations, user control, etc. User control could serve as a kind of 'steering wheel' for adaptation. User may enable or disable adaptation at different levels of granularity and thus could decide in which cases adaptivity should be used by the system. Hybrid systems are a promising approach to contribute to the WIL challenge of supporting real work environments which can differ a lot (e.g., work environment of a secretary vs work environment of a software engineer).

2.2 Basic Components

During the past decade, many adaptive learning systems have been developed (e.g., Brusilovsky and Peylo, 2003; Conlan, 2003; Kobsa, 2001). One common property of such systems is to track user-related aspects and to use them to drive the adaptation process. Besides tracking of information, content, structure of concepts, and links between them are additional elements shared by most of the systems. Around 1990, researchers and developers started to design a reference model aiming at a more general description of an adaptive hypermedia system. Such reference models should provide a baseline for comparison and should also help to develop interoperable systems. The Dexter reference model presented by Halasz and Schwartz (1994) was the first model describing an adaptive learning system from a more general point of view. During the following years, further reference models were proposed to address some of the shortcomings found in Dexter (e.g., Hardman et al., 1994; Schwabe and Rossi, 1998). To specifically address the requirements of adaptive hypermedia systems, De Bra et al. (1999) designed the AHAM reference model, and Koch and Wirsing (2002) later on published the Munich reference model. Both models are based on the Dexter reference model. All reference models focus very much on engineering aspects for adaptive hypermedia systems, and do not take the step to provide a system independent view. Henze and Nejdl (2004) addresses this issue by introducing a formal description of system allowing to compare systems and components they consist of. Being able to compare systems on a logical basis allows to identify possible overlaps as well as missing functionalities for the case of WIL. The following section follows the approach of Henze and Nejdl (2004) and presents the four basic components of an adaptive system[1]:

- Document Space,
- User Model,
- Observations, and
- Adaptation Component.

2.2.1 Document Space

The Document Space is defined as the space of all documents and all associated information belonging to the adaptive system. With this definition, documents are not limited to textual information only but could also include multimedia files, HTML web pages, links to textual documents, etc. Documents can be attached with meta information describing document properties (e.g., author, creation date) and its relation to other documents in the space (e.g., part-of relation between documents). Documents and their associated information describe the knowledge residing in the Document Space. This formal description of knowledge is often stored in a separate model called domain model. A domain model contains the structured expert knowledge in the domain modelled as concepts. For example, in the domain of school mathematics, the domain model should contain basic concepts such as Algebra, Trigonometry, etc. In many cases, concepts are interconnected with each other through different types of relations. For example, to define a hierarchy of domain concepts, a is-a relation can be used. Documents are then associated with one ore more of these domain concepts.

2.2.1.1 WIL Specifics

For a WIL System, the Document Space is of great importance because it defines which organisational knowledge will be available, and thus can be provided to its users. Compared to most adaptive learning systems, a WIL system will have to address the issue of a constantly changing and evolving Document Space as new documents and topics are generated by an organisation.

The WIL situation poses a tough challenge for the domain model. While it is comparatively straightforward to model highly-structured domains such as school mathematics or programming, modelling ill-structured WIL domains is a nontrivial task. Consider, for example, the domain of 'Innovation Management' with concepts such as 'Creativity', 'Innovation Workshop' etc. It is a highly demanding task to

[1] Henze and Nejdl (2004) specify the four components of an Adaptive Educational Hypermedia System (AEHS), however, I argue that these can be generalised to other kinds of adaptive systems as well.

identify relevant concepts within the domain and structure them, e.g., according to a is-a-relation.

2.2.2 User Model

This section introduces the two most popular types of user models, scalar models and structural models, and discusses their relevance for the application in a WIL system.

In order to realise adaptivity, a user model (e.g., Brusilovsky and Millán, 2007) is needed. The user model is described as ”a representation of information about an individual user that is essential for an adaptive system to provide the adaptation effect, i.e., to behave differently for different user” (Brusilovsky and Millán, 2007). The user model constitutes the rationale for individualized learning opportunities of each user. Hence, the User Model is the core of an adaptive system. It represents users from different viewpoints in adaptive learning systems. According to Brusilovsky and Millán (2007), there are two main viewpoints: models representing the individual user and models representing the current (work) context. The most important features of individual user models are a user’s (level of) knowledge, interests, goals, background, and personal properties. Context modelling on the other hand aims at utilising a user’s surrounding environment as a source for maintaining a user model. Examples of such sources are the computational platform (e.g., desktop computer vs. laptop, and smartphones vs tablets). The User Model constitutes the basis for individualised learning opportunities, and is used throughout the whole adaptation process to store and retrieve user information. Typically, for every individual user, an instance of the user model is created which holds the user’s parameter-value for each of the topics in a domain. The user model is continuously updated throughout the interaction of the user with the system.

2.2.2.1 Scalar Models

Scalar models follow a very basic approach in the sense that a user’s knowledge is represented by a single value which can range on a more or less fine-grained scale. Scales can be quantitative (e.g., between 0 and 100) or qualitative (e.g., from novice to expert) depending on the type of modelling approach. Scalar models focus on user’s knowledge only and are typically ramped up and maintained by self assessments and user tests. The main advantage of a scalar model is its simplicity. There is only one value to be calculated per user which is then used for adaptation. At the same point, this is also the main disadvantage: They are very unprecise, and users with different knowledge in single aspects (e.g., subtopics) of the learning domain will receive identical recommendations.

2.2.2.2 Structural Models

Structural models overcome the drawback of ‘one value for the whole domain’ by utilising domain models which can be separated into distinct parts. For each part, an individual value representing a user's knowledge can be calculated. Thus, structural models provide more fine-grained information about the user. The most popular form of structural models are overlay models (Brusilovsky and Millán, 2007). Overlay models create a layer on top of the domain model (the overlay) to store the user's knowledge for each part of the domain. The first (original) overlay models represented knowledge in the same ways as scalar models do: They used a binary value ('known or not known') to represent the user's knowledge about a certain concept. With overlay modelling, the domain model is considered to be the 'expert knowledge' and the overlay reflects the current knowledge of a user as compared to that expert knowledge. The granularity of knowledge which can be represented in an overlay model depends on the structure of the underlying domain model. For domain models providing relations between concepts, it would be possible to reason about a user's knowledge for concepts where no information is available yet (knowledge propagation).

Beyond the original overlay models a variety of extended models have been developed over time. Weighted overlay models allow knowledge to be stored as a range of values (e.g., on a scale from 0 to 1). Another way of representing the amount of knowledge is the use of bayesian models or uncertainty models (e.g., based on fuzzy sets). A different approach are bug models which are dedicated to also store 'incorrect' knowledge users might have about concepts. Another promising approach is to create richer models by adding additional overlays which could for example reflect a user's interest or knowledge from different sources. Stacking up several layers is often referred to as multilayer overlay model.

2.2.2.3 WIL Specifics

For WIL, the Document Space and domain model will most likely cover a different areas of an organisation. Thus, scalar models will lead to very rough estimations about a user's knowledge and may not be appropriate to support them on a task-based level. It is very unlikely that a user is e.g., an expert of a whole learning domain especially for large domains.

In order to address the challenge of real-time learning it is important to have several indicators of knowledge distributed over the whole domain. Having only a single knowledge indicator would lead to more or less the same knowledge resources for all concepts. Therefore, the scalar model can offer a very limited way of adaptation only, and a structural user model, or even an overlay model, should be given the preference.

As I will describe in Section 3.3.1, WIL domains are often 'enterprise models' that describe the domain from different perspectives (concepts, tasks, etc.). The structure of the enterprise model may serve as the basis for an overlay user model in an adaptive WIL system.

2.2.3 Observations

In order to provide personalisation at any point in time, the User Model needs to be continuously maintained. The observation component of an adaptive system describes the way and information which is tracked and collected from a systems' users. Compared to the other basic components, Observations are all about the runtime behaviour of an AHS. There are several ways to diagnose user skills and maintain a user model. Knowledge represented in the user profile can be elicited explicitly from the user but it can also be acquired implicitly from inferences made about the user (Benyon and Murray, 1993). In my view, as explained in Lindstaedt et al. (2009b), implicit acquisition means tracking naturally occurring actions (Jameson, 2003). Naturally occurring actions include all of the actions that the user performs with the system that do not have the express purpose of revealing information about the user. These actions may range from major actions like adding a resource to a collection to minor ones like scrolling down a page. Examples for explicit observations are questionnaires, quizzes to test the knowledge of certain subset of concepts of the domain model. Implicit observations on the other hand, record user actions such as opening a document, browsing through a web page, creating an annotation, etc.

2.2.3.1 WIL Specifics

Systems adapting to the work context should try to automatically identify the current work task to then reason about the resources which might help in accomplishing this task. Such systems can be considered as a special form of context-adaptive systems. Existing context-adaptive systems adapt to, for example, the computational capabilities or the current location of the user.

The question for WIL user models is how to keep track of the variation of knowledge over time. Because knowledge increases and decreases over time, levels of knowledge may also change and have to be taken care of by WIL user model. The following four questions summarise important aspects for a WIL user model:

- How can knowledge about concepts be identified?
- How can the current work context be identified?
- Which kind of user behaviour is relevant for learning and forgetting?
- How can changes in knowledge be recognised?

All questions are tightly related to the type of user model chosen for a WIL system.

2.2.4 Adaptation

The adaptation component contains the functionality for actually making a system adaptive. The characteristics of the adaptation component can range from offering documents which fit best for the user's knowledge state to changing the way how content is presented to users. One very powerful technique to maximise learning transfer is to suggest optimised learning paths, i.e. sequences of learning goals (Karampiperis and Sampson, 2005). More details on adaptation methods and techniques can be found in Section 2.3.

2.2.4.1 WIL Specifics

In order to support WIL in real-time, the system should adapt to the user's task at hand and to his or her background knowledge in any situation. Further, to maximise learning transfer, the system should suggest optimised learning paths through the company's document space. As learning at work happens to a large extent through social interaction (Eraut, 2004), the system should be able to suggest knowledgeable people who can provide help.

2.3 Adaptation Techniques

In the previous sections I described the basic components of an adaptive system with the user model building the basis for most kinds of adaptation. Now the main questions is "What can be adapted in an adaptive system?". This section investigates in detail how various kinds of adaptation can be realised. Brusilovsky (1996) distinguishes two general types of adaptation: *content-level adaptation* or *adaptive presentation*, and *link level adaptation* or *adaptive navigation support*. These types will be described hereinafter.

2.3.1 Adaptive Presentation

According to Bunt et al. (2007) Content-level adaptation or adaptive presentation comprises two steps namely (i) selecting the content (what), and (ii) presenting the content to the user (how). The first step focuses on finding the best possible content out of a repository, e.g. a company's content management system. A simple selection mechanism could select a single document from a directory containing an arbitrary list of documents. A more sophisticated algorithm could investigate the structure of a document to identify relevant paragraphs for a user. The second step, content presentation, aims at selecting the best way of presenting the content to the user. For example, an expert writing a company's financial statement might be presented with very detailed information about exceptions on taxation, while a novice user in the same situation would receive content providing her with an overview about creating financial statements in general.

2.3.1.1 Content Selection

The adaptation of content may vary in terms of the level of granularity which is used to select content. A quite simple way of selecting content is the so called *page-variant* approach (Kobsa et al., 2001) where each page is available in different versions (e.g., from a very general overview pages to very in-depth aspects). An example of page-variant systems is KBS Hyperbook (Nejdl et al., 1998) which selects page variants based on the actual goal and knowledge of a user. The adaptation mechanism selects one of these versions based on user characteristics and presents it to the user. This approach works very well for simple adaptation rules requiring only few versions of each page. For complex adaptations, this approach gets unmanageable because of the vast number of different pages needed.

To overcome the problem of the direct dependency between adaptations and the number of page versions, the approach of *fragment-variants*(Bunt et al., 2007) focuses on a more fine-grained level of detail. The idea is to combine fragments (e.g., paragraphs of text or an image) and present them to the user. In contrast to page-variants, there is no need to have several versions of a single page; instead, a pool of fragments is used which can be dynamically linked together to form a page. Two well-known methods to create such fragmented pages exist: pages build from optional fragments, or pages created from a given structure. The adaptation mechanism for optional fragments selects from a fixed number of fragments to fill up the slots in a page. In the second method, a page is defined by its structure. For each structural element, a number of fragments exist to choose from. This approach is used, for example, in the AHA system (De Bra et al., 2002) to create pages based on the user's current knowledge. The requirement to have the pool of fragments in advance, applies to all approaches. Beyond selecting page-variants or fragments, recent research has also investigated more complex content selection mechanisms (Bunt et al., 2007). The basis for all of these approaches is to utilise additional metadata (e.g., domain models, user preference models or Bayesian networks) to reason about the content.

2.3.1.2 Content Presentation

In the previous section I have described how content is selected in adaptive systems. Thus, the content has been selected which should be presented to the user. The second step of the adaptation process deals with the 'how'-aspect of adaptive presentation. When presenting content to users, two aspects are essential:

- Selected Media: The type of media in order to achieve *media adaptation*
- Amount of contextual information: The amount of additional information which is placed around the main content to provide the context the content is associated with

Media Adaptation. In media adaptation, the aim is to select the best type of presentation medium to convey the content to the user. Types of media can be text, images, videos, spoken language (podcasts), etc. Bunt et al. (2007) identified five factors to be considered:

- User-Specific Features: user preferences (prefers videos over text), abilities (lack of language skills), accessibility issues (e.g. visually impaired users should be presented with audio rather than text)
- Information Features: depending on the type of information, certain types of media are better suited to convey information (e.g. quantitative overviews as a chart rather than a table of numbers)
- Contextual Information: the media type is chosen depending on the environment the user is in
- Media Constraints: the combination of different media types should fit together (e.g., videos with subtitles)
- Limitations of Technical Resources: selected media types depend on the technical infrastructure (e.g. desktop computer vs. Smartphone, available bandwidth, surrounding noise, etc.)

The AVANTI system (Fink et al., 1998) is an example of a system which adapts media to user characteristics (e.g., audio is used to convey information to visually-impaired users) and technical limitations, by reducing the number of images in situations with low bandwidth.

Context Adaptation. While media adaptation focuses on the type media to choose, context adaptation takes into account additional context information to be presented along with the actual content. Adapting the presentation to the context of content is making a trade off between focusing on the actual content to convey, and additional information the content is embedded in (Tsandilas and Schraefel, 2004). The following techniques result in different types of context adaptation:

- **Conditional Text:** content is divided into several parts which can be individually displayed depending on the level of knowledge (or other characteristics of the user)
- **Strechtext:** contextual information can be uncollapsed by clicking on "hot words"; depending on e.g., the knowledge level, certain parts of the strechtext are uncollapsed by default. Strechtext is used in systems such as MetaDoc (Boyle and Encarnacion, 1994) or KN-AHS (Kobsa et al., 1994)

2.3.2 Adaptive Navigation Support

Adaptive navigation support aims at guiding users through the content of an adaptive system by linking content in a sequence which best possibly fits the user. Although the following techniques are not limited to web content only, their success is closely related to the concept of hypermedia. Links offer an ideal entry point to adapt the order in which content is presented.

The most popular navigation support techniques are the following:

- **Direct Guidance** is a simple technique which highlights the most useful **next** link in a list of links, or dynamically adapt the target of a link. Highlighting a link in a list of links offers more freedom to users because all other links remain visible as well.

- **Adaptive Ordering** takes a list of links and sorts them based on different properties of the user model. A simple example would be to put all visited links at the end of the list, and not visited links on the top. A clear limitation of this approach is the need for a list of links. It is hardly possible to sort links e.g., which are embedded in a paragraph. Brusilovsky (1996) proposed this technique for information retrieval applications. Additionally, it may not be clear to users why links have been re-sorted when they visit the same page at a different time.

- **Adaptive Hiding** is probably to most widely used technique to guide users through content. The idea is to hide links which contain information a user is not yet ready to understand. Hiding is based on data stored in the user model, and is not limited to lists of links. As for the previous approaches, it lacks to indicate that there is some more 'hidden' information available.

- **Adaptive Annotation** tries to annotate links with additional information which can be useful to the user. Annotations can be visualised by changing the color of a link, or by attaching small icons to them. The most popular form of annotation are colorings of visited links in most standard web browsers. In these web browser the user model is the history of visited links which is used to differentiate visited from not-visited links. Adaptive system can apply much more sophisticated heuristics based on the various data in the user model.

2.4 Example Systems

In this section I will introduce three of the most prominent adaptive systems which have been developed and used over the last decade. ISIS-Tutor, the first system is one of the first adaptive hypermedia systems merged with a intelligent tutoring system. The second system, NetCoach, is also based on a ITS but is already a web-based system. Finally, AHA! is an adaptive course system which has been widely used for teaching students at universities.

2.4.1 ISIS-Tutor

The ISIS-Tutor is an integrated learning environment based on the concepts of a intelligent tutoring system (ITS). It was developed by Brusilovsky and Pesin (1994) as tutoring system for users working with the information retrieval system CDS/ISIS/M (in short ISIS). ISIS provides a programming language which allows to access format records stored in the information retrieval system and to format the results. To support users in learning more than 50 commands and parameters, ISIS-Tutor provides a learning environment around the ISIS-System and its programming language. The core components of ISIS-Tutor are the model component storing a domain and a student model, a tutoring component offering personalised learning guidance, a hypertext component delivering the learning content, and a learning environment where users can select a sample record, apply a formatting string, and see the resulting output without affecting real data of the ISIS system. Users can execute all commands step by step to explore changes caused by a command.

2.4.1.1 Document Space

The document space of ISIS-Tutor consists of the domain model and the learning learning material linked to it. The domain model contains all concepts of the programming language to be taught. It is structured into a directed graph where edges denote prerequisite relations between concepts. Learning material is linked to concepts of the domain model and thus building the document space of ISIS-Tutor.

2.4.1.2 User Model

The user model (called student model in ISIS-Tutor) is an overlay model of the domain model described as part of the document space. At any point in time, the student model reflects which concepts of the domain model the user has mastered, was not able to master, or has not learned about at all. For each concept, an integer counter reflecting the level of mastery, is stored. The student model can be accessed by all components of ISIS-Tutor to adapt their behavior according to the current integer value available for a concept. The complexity of adaptation depends on the number of knowledge states a component can distinguish. A knowledge state is defined as a range of integer values the counter of a concept could be assigned. For example, a simple module could only differentiate between two states ‘known’ and ‘unknown’ with one threshold value. The ‘known’-state could be mapped to values greater then 5, whereas the ‘unknown’-state would range from 0 to 4. Brusilovsky and Pesin (1994) argue that this thresholding technique offers a flexible way of adapting to concepts of different levels of difficulty and different classes of users.

2.4.1.3 Observations

To understand how observations take place, we have to take a short look at the tutoring component first. New users users start with an empty student model. ISIS-Tutor tutor presents the user with a problem related to a concept. In short, mastering the problem means to master the concept. If there are no unsolved problems for a concept, the tutoring component presents a new concept to the user. Every time a teaching operation is completed, the student model is updated. Thus, the student model is automatically updated by different components while working with ISIS-Tutor. The hypermedia component, for example, tracks a users’ navigation through the learning learning material and updates the concepts in the student model.

2.4.1.4 Adaptation Component

In ISIS-Tutor, the tutoring component is responsible for generating the adaptation. The main adaptation parameter is the current knowledge state of a user which is stored in the student model. Depending on this knowledge state, the tutoring component marks links with different colors (see Section 2.3.2 and Brusilovsky and Pesin, 1998). These color-coded links inform users which content is e.g., ready for learning or has already been learnt.

2.4.1.5 System Architecture

ISIS-Tutor was designed as an environment for learning the print formatting language of the information retrieval system CDS/ISIS. Although ISIS-Tutor connects to the CDS/ISIS to retrieve data base records, it is designed as a standalone application. ISIS-Tutor was written in the programming language ISIS-Pascal.

2.4.2 Net Coach

NetCoach (Weber et al., 2001) is a web-based intelligent tutoring system which enables teachers to author adaptive learning courses without programming knowledge, and offers students a personalised way of learning. It is the successor of a long history of adaptive tutoring systems: ELM-ART II (Weber and Specht, 1997), ELM-ART (Brusilovsky et al., 1996), and ELM-PE (Weber and Möllenberg, 1994). Compared to printed textbooks or a lot of e-learning courses, NetCoach offers courses which are adapted to the user, adaptable by the user, interactive and communicative.

2.4.2.1 Document Space

The document space of NetCoach consists of four parts: concepts, documents, test items, and a training criterion. Concepts are connected through a prerequisite relation which defines the sequence in which concepts should be presented to a users. Documents and concepts are linked to each other with a one-to-one mapping, and are themselves structured in a section sub-section style (part-of relation). Prerequisite concepts can be specified by teachers when designing the course. If no prerequisite concepts are defined, the default sequence given by the document structure will be used. Test items are questions to assess the user's knowledge contributing to a concept. NetCoach uses test groups (a number of test items) to infer the knowledge of a concept. With a test item it is not only possible to infer the knowledge of a single concept, but also concepts related through the prerequisite relation. The last part of the document space are training criterions assigning a numerical value to each concept. The criterion specifies the number of test items need to be solved successfully solved to mark a concept as 'known'.

2.4.2.2 User Model

The user model of NetCoach is built on top of the concepts contained in the document space, and is implemented as an overlay model with four layers:

- **First Layer:** Stores for each concept whether the associated page has been visited.
- **Second Layer:** Stores which exercises and test items which have been done.
- **Third Layer:** Identifies concepts which have been inferred to be known by a user; inference is based on links between concept and data stored in the first layer.
- **Fourth Layer:** Identifies concepts which have been set by the user as known.

2.4.2.3 Observations

Observations in NetCoach are tightly bound to the four layers of the user model, and thus can be divided into four different sources of observation. When a user walks through pages of a course, all page accesses result in an update of the first layer. A user's work on exercises and test items is used to update the second layer of the user model. The third layer is updated by an inference mechanism which is indirectly steered by page accesses. The last source of observation is explicit input from a user about concepts which are already known.

2.4.2.4 Adaptation Component

NetCoach's adaptation component implements 'curriculum sequencing', a special type of adaptive ordering, to provide students with a course adapted to their needs. To guide students through this course, NetCoach makes use of adaptive annotation (see Section 2.3.2) to hide or show links depending on their current user model.

2.4.2.5 System Architecture

NetCoach is designed as a web-based client-server architecture. The server is based on CL-HTTP, which has been implemented with the programming language Common Lisp, and supports HTTP 1.1. The server maintains the content, user models, and NetCoach authoring tools to create adaptive courses. Users do only need a standard web browser to connect access courses hosted on a NetCoach server. NetCoach server runs on Linux environments, Mac OSX, and Windows systems. The authoring tools support the creation of courses from static HTML pages, Flash content, HTML with Javascript, and other plugins.

2.4.3 AHA!

AHA! (De Bra and Ruiter, 2001) is an adaptive learning system which originated from the field of teaching university courses. It aims to provide an alternative to the 'one course fits all students' by offering web-based courses adapting its content to the student's knowledge state. The development of the first version started in 1996, and is currently available in its third version (Bra et al., 2004). A course in AHA (also referred to as a AHA application) is defined as a set of concepts with certain requirements and links between them. A single concept could be represented as webpage but could also be more general, and may link to a number of other concepts. The author of a course can specify 'requirements' for each concepts, which AHA! uses for adaptation. AHA! is still applied for delivering adaptive courses such as a course on business English (Höver and Faltin, 2008). In parallel, basic concepts and lessons learned from work into AHA! have flown into the GRAPPLE-IP Project[2].

2.4.3.1 Document Space

The Document Space of AHA! v3.0 is built from two main elements: concepts and pages. Concepts are defined by a name, a description, and one or more attributes. Attributes can contain persistent (data from user model) and temporary information (only available during a teaching session). Concepts can be seen as metadata of pages. Attached to each concept is a *requirements expression* used by the adaptation component to decide which concept to present for a current user's knowledge state. The content of a course is contained in pages which themselves are linked to concepts. Pages are stored in HTML format providing a convenient way of displaying in web browsers, and allowing to add metadata (e.g. rules for inclusion of a fragment) as XML tags.

2.4.3.2 User Model

The user model of AHA! V3.0 is designed as an overlay model (see Section 2.2.2.2). The document space with concepts linked to pages builds the basis for the user model. The overlay itself consists of persistent and non-persistent attributes linked to concepts. The most important persistent attribute is the one storing knowledge levels. The knowledge level in AHA! can be increased and decreased by visiting a page. Additionally, AHA!'s propagation rules allow to update a set of concepts with a single page visit. Non-persistent attributes are an exception as they are temporarily (for a session) stored only. The following Section 2.4.3.3 will explain how non-persistant attributes are used for observing a user's activities in AHA!.

2.4.3.3 Observations

Observations in AHA! are based on the non-persistent *access* attribute in the user model. When a user logs into a course in AHA!, each *access* attribute of a concept

[2] http://www.grapple-project.org/

will be initialised with the value *false*. When a user walks through the course, all pages which have been visited are marked. That is, all *access* attributes of concepts belonging to a visited page will be set to *true*. At the end of a session (user logs out), any *access* attribute which was set to true will be be used to update the persistent *knowledge level* attribute in the user model.

2.4.3.4 Adaptation Component

The Adaptation component of AHA! v3.0 is driven by a user's selected page in the first place (navigation support), and by XML content in a page in the second place (content presentation). To accomplish these adaptations, AHA! utilises two adaptation techniques: Adaptive Navigation Support (see Section 2.3.2) and Adaptive Content Selection (see Section 2.3.1.1).

Adaptive Navigation Support: Before a page is presented to a user, all links contained in this page will be evaluated. A link will be shown under two conditions: the link has not been visited, and the *requirements* expression of the concept linked to the page is fulfilled.

Adaptive Content Selection: After having defined the links to other pages, the content of the page itself is selected. A page in AHA! can be constructed with static and dynamic content fragments which is surrounded by XML tags containing rules for inclusion/exclusion. The final page is constructed by taking the static content and including all dynamic fragments which fulfill the rules.

2.4.3.5 System Architecture

AHA! v3.0 is designed as a client-server system which uses http-requests to exchange information. Clients are web browsers installed on computers run by users, and a Apache Tomcat web server. The server runs a number of Java servlets containing the actual functionality of AHA!. Data about users and concepts is stored in a MySQL database. The execution flow is as follows:

- A user opens a HTML page of AHA! in her web browser (A http-request is sent to a servlet)
- The request is processed by a servlet running the adaptation engine
- The web server sends back the adapted web site

AHA! is licensed under GPL v2.0.

2.5 Summary and Outlook to Next Chapter

Being able to compare systems on a logical basis allows to identify possible overlaps as well as missing functionalities for the case of WIL. In this chapter, three different aspects have been covered. First, I have described the four basic components of adaptive systems (Document Space, User Model, Observations, and Adaptation

Component). Clearly, all these components are involved when designing user model services for adaptive WIL support. For each of the four basic components, I have identified and discussed specific requirements for the case of WIL. Second, I have presented different techniques of how adaptation can be realised. Third, I reviewed three of the most prominent adaptive systems and described in detail how these systems implemented the four basic components. Comparing the implementation of these systems with the discussion of 'WIL Specifics' in Section 2.2 it becomes obvious that these adaptive systems support only a subset of functionality to run in a WIL environment (see Table 2.1).

Table 2.1: WIL Specifics compared to the functionality of example adaptive systems: ISIS Tutor, NetCoach, and AHA!

	Example System		
WIL Requirement	ISIS Tutor	Net Coach	AHA!
Document Space			
Automatic Update of Document Space	no	no	no
Different document formats	no	no	no
Real work documents	no	no	no
Observations			
Support of current work task	no	no	no
Automatic user model maintenance	no	no	no
Adaptation			
Recommendation of knowledgeable people	no	no	no
Support of learning path	yes	yes	yes
User Model			
Structural User Model available	yes	yes	yes
System Architecture			
Configuration of different domains	no	yes	yes
Client/Server Architecture supported	no	yes	yes

To conclude, this chapter dealt with existing adaptive systems and a discussion of what could be re-used and what is missing in order to support WIL with regard to user modeling and user model services. In a next step, now the question shall be answered what else is needed to enable adaptive work-integrated learning. What are the concrete requirements for the WIL user model and user model services that can be derived from WIL theory? This question shall be looked at in the next chapter, and conceptual design solutions shall be proposed.

Chapter 3

Adapting Work-integrated Learning

In this chapter, theoretical considerations of WIL are being discussed, and requirements are derived for designing a WIL User Model and WIL User Model Services. Part of this work has been published in the following papers and reports co-authored by me: APOSDLE Consortium (2010b); Lindstaedt et al. (2008b, 2009b).

Having described the basic components of a learning system needed to provide adaptive support, the question remains how a WIL system must actually look like in order to be most useful. How much and what kind of learning guidance do people engaged with workplace learning desire, need, and actually use?

3.1 Theoretical Considerations for WIL

A number of design and usability challenges have to be considered in order to actually bring the benefits of adaptation to the individual user in a WIL setting. Jameson (2003) has identified predictability & transparency, controllability, unobtrusiveness, privacy, and breadth of experience as critical challenges for adaptive systems. In line with Jameson (2003), I will use the term obtrusiveness to refer to the extent to which a system raises the user's attention. For WIL environment design, unobtrusiveness and privacy constitute the hardest challenges. I will elaborate these two aspects in Section 3.1.1 and in Section 3.1.2. In addition, I will highlight the role of learning from knowledgeable colleagues in the context of WIL in Section 3.1.3.

3.1.1 Unobtrusive User Model Maintenance

As described in Section 2.2.2, the user model constitutes the rationale for individualized learning opportunities of each worker. Many adaptive systems build user models in an intrusive manner: in order to maintain valid user models, they require explicit feedback from the user and often at a significant level of user involvement (Adomavicius and Tuzhilin, 2005). In systems that support learning it is often natural to administer tests of knowledge or skill. The main advantages of testing are that it can be used in many domains and it is easy to implement. However, testing is highly obtrusive and cannot be applied to WIL for many reasons including the absence of the one correct solution for most work tasks. Thus, ideally, each user's knowledge and skills should be determined based on his or her normal work activities. Various approaches have been suggested for automatically maintaining user competency profiles in organizational databases, e.g. based on document authorship or name-concept co-occurrences (for overviews see Maybury, 2006; Yimam-Seid and Kobsa, 2003). A theory-based approach has been proposed, for instance, by Ley (2003) who suggest inferring employee competencies from past task performance.

The problem with automatically diagnosing competences is that deriving information about user expertise from user interaction with the system (e.g., from document authorship, communications, e-mails) may lead to an inaccurate user model. That is, inaccurate information may be represented in the users' profiles and as a consequence, the adaptation is 'wrong'. Thus it is important that users have the possibility to access and edit their user models. Such functionality has been presented in the context of research into open learner models (e.g., Bull and Kay, 2007). A further advantage of open learner models is that they enable reflection, the planning of individual learning, or monitoring one's own learning progress.

Another challenge for user model maintenance is that users learn from diverse sources (emails, document repositories, portals) and that there is no central learning system. Therefore, user activities have to be collected from interactions with these different sources. A way to address this challenge has been suggested e.g., with the CUMULATE server by Brusilovsky (2004).

Summing up these considerations, user models for supporting work-integrated learning should (i) be as unobtrusive as possible, that is, the efforts for users for maintaining their profiles should be minimized, (ii) and they should be accessible to its users in order to improve accuracy, and to support reflection and learning, and (iii) they should be able to collect data from various sources and integrate them.

3.1.2 Privacy

Another requirement for WIL Systems is the aspect of privacy. Privacy is highly important when deploying systems in real work environments, and may be a knock-out criteria if not addressed throughly. To enforce user privacy in a WIL system which maintains a user model, appropriate organisational and technical measures have to be applied. Enhancing privacy in adaptive systems is a quite complex task as it depends on the organisational environment, data collected, privacy regulations etc. Additionally, there is no standard approach to enhance privacy in adaptive systems (Wang and Kobsa, 2007) but rather of a lot of different approaches developed in various fields of research (e.g., Spiekermann and Cranor, 2009). For my thesis I want to highlight the importance of user privacy, but do not investigate it in more detail.

3.1.3 Finding Knowledgeable People

In line with Billett (1993), I understand learning at work as the acquisition of knowledge and skills as a function of participation in authentic tasks, with support and guidance from other more skilled users. What becomes obvious in this definition is that (work-integrated) learning is social in nature. It is widely acknowledged that knowledgeable colleagues are one of the most important sources of knowledge for workers.

For instance, Kooken et al. (2007) in their workplace learning study observed four “solution categories” of persons seeking help at their workplaces: interpersonal help seeking, seeking help from paper based written material, seeking help from digital written material and practical application (‘trial and error’). According to the authors, of these strategies, interpersonal help seeking is the one that is applied most frequently (in 70% of the cases). Finding knowledgeable people, however, can often be difficult for several reasons. First, the number of workers within an organization may be too large to know the fields of expertise of everyone. Second, even if workers work together in one and the same office, they are often not aware of what their colleagues are working on. Third, competency databases, if available, are often outdated because manually maintaining these databases is costly. In addition, such competency databases often comprise rather coarse-grained competencies (e.g. ‘programming skills’). This may result in the situation that most of the questions that occur are posed to a relatively small group of ‘experts’, even though other persons also might have been able to provide support. Therefore, in order to support learning at work, people recommendation is a powerful means.

3.2 Requirements from Theoretical Considerations

In the introduction of my master’s thesis (Chapter 1) the following challenges were mentioned: Real-Time Learning, Real Knowledge Resources, and Real Computational Environment. Starting from these three challenges, three high level requirements can be formulated:

- **Theory-Requirement 1 (TR1):** The WIL System should support the user in an adaptive manner during his or her everyday work
- **Theory-Requirement 2 (TR2):** The WIL System should re-use resources within the organisational repository as learning content
- **Theory-Requirement 3 (TR3):** The WIL System should be embedded in and interact with a user’s normal work environment

TR1 can be further refined based on the theoretical considerations from Section 3.1.

- **Theory-Requirement 1.1 (TR1.1):** The WIL System should adapt learning to the task at hand and the background knowledge of its users
- **Theory-Requirement 1.2 (TR1.2):** The user model of a WIL system should be maintained in an unobtrusive manner.

- **Theory-Requirement 1.3 (TR1.3):** A WIL System should take care of privacy regulations of the environment being placed in.

TR2 includes both document resources and 'human resources' and can be further refined based on the theoretical considerations from Section 3.1 as follows

- **Theory-Requirement 2.1 (TR2.1):** The WIL System should be able to identify and recommend relevant documents within the organisation.
- **Theory-Requirement 2.2 (TR2.2):** The WIL System should be able to identify and recommend knowledgeable people within the organisation.

Theory-Requirement TR3 can be refined as follows:

- **Theory-Requirement 3.1 (TR3.1):** The WIL System shall be able to interact with standard software that is used at the workplace

3.3 Conceptual Design of a WIL UMS

This section will present a conceptual architecture for designing an adaptive system for the use in working environments which integrates working and learning aspects in one place. Foundations of such an architecture have been discussed in Chapter 2 and Section 6.1. In the following, I will describe:

- WIL user model structure and maintenance
- Recommendations supporting WIL
- Different types of WIL user model services

3.3.1 WIL User Model Structure and Maintenance

As explained in Section 2.2.2, the user model is crucial for realising adaptivity. Various types of user models exist (structural models, feature-based models, etc.). The question arises how the user model of an adaptive WIL system should look like. Research into organizational structures has identified that many companies create and maintain different types of formal models, so called enterprise models of their work domain (Fox and Gruninger, 1998). According to Ghidini et al. (2009), the three most popular models are *work domain models* (typically represented as an ontology), *process or task models* (typically represented as a workflow or process model), and *competency (or skill) structures* (typically represented as a simple list or matrix). Such models provide a comprehensive representation of the whole domain, i.e. they capture the entire knowledge of the workplace. Based on these insights, it seems obvious to structure the WIL user model as an *overlay* (see Section 2.2.2 of existing enterprise models of the workplace domain to be supported with the adaptive WIL system.

In order to provide adaptive functionality throughout the interaction of the user with the system, it is necessary that the user model is continuously maintained. As described in Lindstaedt et al. (2009b), a number of interesting approaches to user model maintenance have been suggested in other adaptive systems. For instance, researchers interested in adaptive hypertext navigation support have developed a variety of ways of analyzing the user's navigation actions to infer his or her interests or to propose navigation shortcuts (e.g., Goecks and Shavlik, 2000). Schwab and Kobsa (2002) came up with an unobtrusive approach for user learning interest profiles implicitly from user observations only. The problem is that such approaches for diagnosing user interest cannot easily re-used for diagnosing user knowledge and skill (which is necessary for maintaining WIL user models).

In Lindstaedt et al. (2009b), a paper of the research group in which my thesis was written, we have suggested to tackle the challenge of user model maintenance by observing naturally occurring actions of the user which are then interpreted as Knowledge Indicating Events (KIE). KIE denote user activities which indicate that the user has knowledge about a certain topic. Examples for KIE include executions of tasks which involve that topic, communication with other users about that topic, and the creation of documents which deal with that topic. All types of user interaction with the system may serve as KIE. KIE thus are based on usage data. This approach goes into a similar direction as the work of Wolpers (2008), who suggest using attention metadata for knowledge management and learning management approaches. In order to interpret usage data (KIE), each of the KIE must be related to one or several concepts in the user model. This allows relating user actions to knowledge and skills and drawing conclusions about the user's knowledge level.

3.3.2 Recommendation

From use cases and requirements, two types of recommendations can be derived that seem to be specifically relevant for supporting WIL, namely

- Learning Goal and Content Recommendation, and
- People Recommendation.

3.3.2.1 Learning Goal and Content Recommendation

One of the main goals of a WIL system is to provide a user with learning content in a highly adaptive manner. For the selection of learning content, the WIL system should take into account the actual learning need of a knowledge worker. As has been explained in detail elsewhere (e.g., Ley et al., 2010a), performing a task is the main objective of a knowledge worker. This means the learning need of the user is typically associated with the task he or she wants to perform. The actual learning need of a user in a concrete situation shall be determined by requirements of a task at hand, and by a knowledge worker's existing knowledge and skills, which has originated from previous learning and working experiences.

According to APOSDLE Consortium (2010b), three steps have to be performed in order to provide adaptive learning support to a user: (1) building a task-based learning history, (2) performing learning need analysis, and (3) computing learning paths. In the following these will be described in more detail.

With regard to building a task-based learning history, it was suggested to approximate a user's knowledge state based on tasks the user was engaged with in the past. The learning history for each user would consist of a collection of learning goals related to these tasks which approximates the most likely knowledge state of the user (see for more details APOSDLE Consortium, 2010b). This step can be seen as a special case of the KIE approach, here, the only KIE is the task a user has performed.

In order to provide the best possible support for a worker who tackles a certain task, the worker's learning need has to be specified. The second step, performing learning need analysis, can also be applied if multiple KIEs exist: If there is a direct link of tasks and learning goals in the enterprise model (competence model), there is an opportunity to compare the requirements of a task (in terms of learning goals) with the knowledge state of a worker.

In the third step, a learning path is computed for all the learning goals. A learning path is a ranked list of learning goals to be acquired by the worker in order to be able to perform the task. One of the main assumptions in technology enhanced learning derived from pedagogical principles and theory, is that the learning process can be improved through guidance (Schmidt, 2005). If the enterprise model depicts a prerequisite relation between learning goals, this information can be exploited in the learning goal ranking. If no such prerequisite relation exists, other algorithms need to be designed.

Starting from a raked list of learning goals, the organisational data base can be queried in order to find relevant resources that may support the user in performing his or her task. 'Scruffy methods' for finding relevant content using associative retrieval have been described in Lindstaedt et al. (2008a).

3.3.2.2 People Recommendation

In work-integrated learning, recommending knowledgeable persons means finding people within the organization who have expertise related to the current (learning) goal of a user. Similar functionality has been realized by expert finding systems. For instance, the MII Expert Finder (Maybury et al., 2002) analyses documents and resumes authored by a company's employees to build up a name-topic mapping for recommending experts. Extracting information from outgoing emails, stored chats and profiles and user details from directories are utilized in the SmallBlue system (Lin et al., 2008) to provide a ranked list of persons based on a search query. Both the MII Expert Finder and SmallBlue are designed as centralized systems which collect all information in a central repository to apply algorithms for recommendation. A decentralized approach was presented by Vivacqua and Lieberman (2000) who developed expert-finding agents running on a developer's computer and investigated java code written by this developer. To find an expert, an agent asks other agents on a network about the expertise they can provide and applies a similarity model to compile a list of knowledgeable users. For a comprehensive overview of other approaches to non-commercial expert finders see, e.g., Yimam-Seid and Kobsa (2003).

Maybury (2006) reports on the challenges of expert finding systems and gives an overview of how these challenges were addressed in existing commercial products. All systems described in the review provide a simple keyword-based search interface to find experts. Recommended experts can be filtered and/or ranked according to different properties such as years of experience or number of publications. Most of the systems in the report utilize a variety of sources (databases, email servers, document management systems) to extract indicators of expertise from text. Additionally, most of the tools provide ways to specify personal expertise profiles or keywords manually. Only a few of them include some kind of behavioral processing for instance, by analyzing search queries, access to documents, or contributions to portals. In all the described systems, users do not have the possibility to access and edit information about their own skills and competencies automatically extracted by the expert finder tools.

Yimam-Seid and Kobsa (2003) provide an in-depth analysis of the expert finding problem, and give an overview over existing noncommercial systems in this domain. The authors present what they call an Intuitive Domain Model of Expert Finding systems. The model is a faceted classification scheme to describe expert finders. The following facets are distinguished: expertness deduction operation (e.g., document authorship), expertise indicator source (e.g., authored documents), expertise indicator extraction operation (e.g., domain knowledge driven), expertise model (e.g., query- time generated), query mechanisms (e.g., explicit query for expert), matching operations (e.g., exact/overlap matching) and output presentation (e.g., ranked list of names).

In line with the *Intuitive Domain Model*, the following considerations have to be made when designing expert recommendation for WIL:

1. What are the criteria based on which user expertise is inferred by the system?
2. What are the user interactions with the system (i.e. operations) from which the expertise of the users can be derived?
3. How is the expertise derived from the user interaction with the system?
4. What is the underlying model of expertise?
5. How can users seek for experts within the system?
6. How does the system respond to a request for an expert?
7. How is the result presented to the users?

With regard to (1), in WIL, expertise can manifest itself in various ways, such as the authorship of documents or publications, or the acquisition of projects. These criteria of who is an expert may also vary in different organizations. Typically, workers at their workplaces are subjected to time pressure. Thus, in order to derive information of who is an expert, the challenge in WIL related to (2) and (3) is that the system can identify expertness as unobtrusively as possible in order to not disturb the users instead of helping them. To define who is an expert in a domain, an underlying model of expertise is needed. Often, organizations have models about concepts of the learning domain and their relationships. Such centralized organizational models can then serve as the basis for identifying the level of expertise of individual employees in each of the concepts in the enterprise model (4). Two different scenarios are conceivable in WIL how users could want to seek for experts within the system (5): First, a user might want to learn about a specific topic. Then, he or she could access the expert finding system and search for a knowledgeable colleague who can provide help in this topic. The second scenario is that a user does not know what he or she is looking for. Instead of active search, the expert finder could recommend a knowledgeable person who addresses the knowledge need of a user by taking into account the users' context. In such a system, no active search of the user is necessary, the system knows what information a user needs in a learning situation. It is especially this second scenario that seems relevant for WIL. How the system responds to a request for an expert, (6) and how the result is presented to the users, (7) are design decisions that are not specific to WIL systems.

3.3.3 WIL User Model Services

At this point, the question arises how to exploit the information in the WIL user model and to provide adaptive functionality described in Section 3.3.2.

As has been explained in a paper of the working group in which this thesis was written (Lindstaedt et al., 2009b), integrating learning support into work practices does not only mean running a WIL system and applications already deployed in organizations side by side, but also the possibility to extend and enrich existing applications. In order to meet this requirement, a service oriented architecture (SOA) approach to WIL user model design and maintenance based on the OASIS reference model[1] is proposed. With the term WIL user model services, I refer to all kinds of WIL functionality that maintains and utilizes the data stored within the WIL user model.

The SOA approach has at least four advantages for WIL: First, the paradigm of SOAs allows to split adaptive functionality into different subgroups (services) that can be used independently from each other. Second, services can easily be integrated in existing applications which make them especially attractive for the WIL situation. Third, services are formally described and thus it is easier to have an overview of service functionality, protocols, etc. Fourth, existing services can be used for implementing new services (service mashups).

[1] http://www.oasis-open.org/committees/soa-rm

In the following, four types of WIL services are presented that are related to the user model of an adaptive WIL system and to the WIL system's adaptivity. These shall be termed WIL User Model Services (WIL UMS):

- Logging services
- Production services
- Inference services
- Control services

3.3.3.1 Logging Services

Logging services are responsible for updating the WIL user model with observed KIEs, and thus provide the basis for all other services. Sensors connected to may different applications within the work environment send detected user activities (such as task executions, collaboration events, titles of opened documents) to logging services to be added to the user model. Preprocessing of incoming user activities are handled here. This could involve the transformation of user activities into a format required by the user model, or enriching incoming data with timestamps and other system related information.

3.3.3.2 Production Services

Production services make the stored KIE available to other services within the WIL system. Service consumers could either be other server-side services, or clients which e.g., visualise KIEs to support predictability and transparency of the adaptation. Based on the specific requirements of clients or other consumers, production services could filter or aggregate KIEs to provide specialised views on the data. For example, one service could produce a list of all tasks executed by a single user. A client consuming this service could then provide a timeline-based visualisation of task executions over time. Different views on the data also offer a way to retrieve usage data associated with a specific enterprise model. Besides providing predefined views to filter usage data, production services could also allow to query the user model with individual parameters.

3.3.3.3 Inference Services

Inference services process and interpret KIEs to draw conclusions about different aspects of users, such as levels of knowledge or learning opportunities. Inferences may then be utilised to adapt the functionality of the service itself, or by providing the outcome to other services. A WIL user model should allow to generate inferences in different ways: For example, heuristics could be directly applied on KIEs to generate aggregated information about users. Exploitation of KIEs with regard to enterprise models, or a hybrid approach by combining heuristics with enterprise models, could also lead to inferences.

3.3.3.4 Control Services

Control services provide ways to control KIEs stored in the WIL user model. Controlling usage data is important for handling privacy issues and imprecise KIEs collected in the user model. Privacy issues could be addressed by applying certain privacy policies of organisations on logging and production services. An example would be a policy about data retention, demanding the deletion of KIEs after a certain period of time. The aspect of imprecise data can be addressed by presenting users with an overview of KIEs associated with them. Based on this overview users could then use a control service to manually delete or modify the collected data.

3.4 Summary and Outlook to Next Chapter

In this chapter, I have derived requirements from WIL theory. A first focus was on the question of how to structure and maintain the WIL user model. The structure of the WIL user model is crucial because it determines what can be stored in a user model, the maintenance is important for having up-to-date information. Then, recommendations in WIL have been identified to be important for finding people, and finding learning goals and learning materials. To sum up, three kinds of functionality seem to be extremely relevant for supporting WIL: Unobtrusive user model maintenance, recommendation of learning goals and content, and recommendation of knowledgeable people. Integrating learning support into work practices does not only mean running a WIL system and applications already deployed in organizations side by side. Instead, it is important that there is the possibility to extend and enrich existing applications. Hence, the approach suggested in my master's thesis is to build on some general principles of the SOA paradigm. Following this paradigm, four types of user model services were proposed (Logging Services, Production Services, Inference Services and Control Services) at the end of this chapter that are needed in order to implement the desired functionality.

In a next step, these general conceptual design principles for WIL User Model Services will be applied to a real adaptive system (APOSDLE) to be used in real work environments thereby integrating working and learning aspects in one place. For this system, existing use case descriptions will be analysed, and further requirements for WIL User Model Services will be derived. The main contribution of the next chapter will be the conceptual architecture of WIL User Model Services of a concrete adaptive WIL system.

Chapter 4

Conceptual Design of the APOSDLE WIL-UMS

In this chapter, the adaptive WIL system APOSDLE is introduced, and the conceptual architecture of the APOSDLE User Model, and User Model Services is being described. Part of this work has been published in the following papers and reports co-authored by me: Aehnelt et al. (2008); APOSDLE Consortium (2010b); Beham et al. (2009, 2010b); Lindstaedt et al. (2009b).

The conceptual foundations presented in the previous chapters will now be applied to the case of a real WIL system. My work on the conceptual design and prototypical implementation was done in close collaboration with the EU-Project APOSDLE[1]. The main outcome of the APOSDLE project is the APOSDLE system (Advanced Process-oriented Self-directed Learning Environment) aiming to support knowledge workers during their daily work. The APOSDLE system includes intelligent services that automatically detect the work context of a user, services that recommend resources and learning episodes. Because of its many benefits described in Section 3.3.3, the paradigm of a service-oriented architecture was used for APOSDLE.

Before I will describe in detail the implementation work of my master's thesis, it is necessary to understand the functionality of the APOSDLE system.

4.1 Goals of APOSDLE

One of the core goals of APOSDLE is to improve the productivity of knowledge workers during their every day work by supporting them with relevant resources directly at their workplaces. Compared to other intelligent tutoring systems or adaptive educational systems, APOSDLE puts a strong focus on incorporating as much as possible existing resources available in organisations, rather than creating new learning resources. The APOSDLE approach (Lindstaedt et al., 2006) is to support learning and teaching episodes tightly integrated into the work processes by taking into account the work context, such as the task at hand, and the prior knowledge of the knowledge worker. Workers are provided with resources relevant to their work context, thus raising their own awareness of learning situations, content, and people that may be useful for learning at that point in time. This context-aware knowledge delivery takes place within the usual work environment of the users. Thereby, APOSDLE takes into account the three challenges of WIL introduced in Chapter 1: Real-time learning, real resources and real computational environment.

[1] http://aposdle.org

4.2 APOSDLE Scenario

The following scenario shall illustrate the core functionality that the APOSDLE system should provide:

Georg recently joined a mid-sized consultant agency in the field of innovation management. His main job is to consult clients to find new innovations on top of their products. Although he already has some background knowledge on consulting, innovation strategies and creativity techniques, he is considered as a beginner in his company. To support employees in doing their daily work, his company introduced the APOSDLE work-integrated learning system. APOSDLE runs in the background of his laptop computer and collects data about his interactions (mouse moves, opened documents, written emails, etc.) From time to time APOSDLE informs Georg about documents available in the company's document store which could help him do his current task. Let's see how this works in a concrete work situation.

Georg was told to prepare a workshop with Carview - a company who is a client in the automotive business building different types of rear mirrors for cars. Carview asked Georg's agency to help them finding new ideas and strategies to introduce new, innovative products. Georg received some initial information about Carview and how to prepare such workshop from the project leader Susan. He starts to investigate the workshop template to see which items should be on the agenda. Additionally, he skims through the current product portfolio of Carview. While reading a specification of a rear mirror, Georg receives a notification telling him that APOSDLE 'thinks' he is working on preparing an innovation strategy, and might need some additional information available within the company. Georg decides to follow APOSDLE's offerings afterwards. In his Windows task bar, an icon indicates that a task was detected and that information is available. He clicks on the icon which opens the APOSDLE client presenting him resources relevant for the task "preparing an innovation strategy". Georg finds not only documents and videos but also receives recommendations of colleagues of other departments who have recently prepared similar workshops. At first, he decides to open a document offering methods to collect strengths and weaknesses of his client and their products. APOSDLE opens an extended PDF reader capable of highlighting important sections (called snippets) inside large documents. Georg is directly guided to a snippet introducing the SWOT analysis. A second snippet introduces another method to find out how current products a positioned in different markets. Having this information, Georg is now able to prepare the action point 'Analysis the current situation' proposed in the template.

4.3 Use Cases for WIL Support

Use cases of real users are essential to develop systems that fit the requirements of real environments. For my work, I could profit from a large-scale study applying a multitude of participatory design methods Jones and Lindstaedt (2008) at the beginning of the APOSDLE project. The outcome of the study was an extensive set of use cases that is documented in APOSDLE Consortium (2007a).

In the following, I will present these seven use cases which I have identified to be the most relevant for knowledge workers interacting with an adaptive WIL system. The scope of each use case is described by a short problem statement, a summary of the use case, all actors involved, and preconditions which are mandatory for successfully carrying out a use case scenario. Triggers specify events which have to be met to enter a certain use case. The row 'Success Scenario' lists all steps of actors leading to a successful end state of a use case as outlined in the summary. Possible variations of the Success Scenario are listed as "Extended Scenario". Figure 4.1 presents an overview of general use cases relevant for the

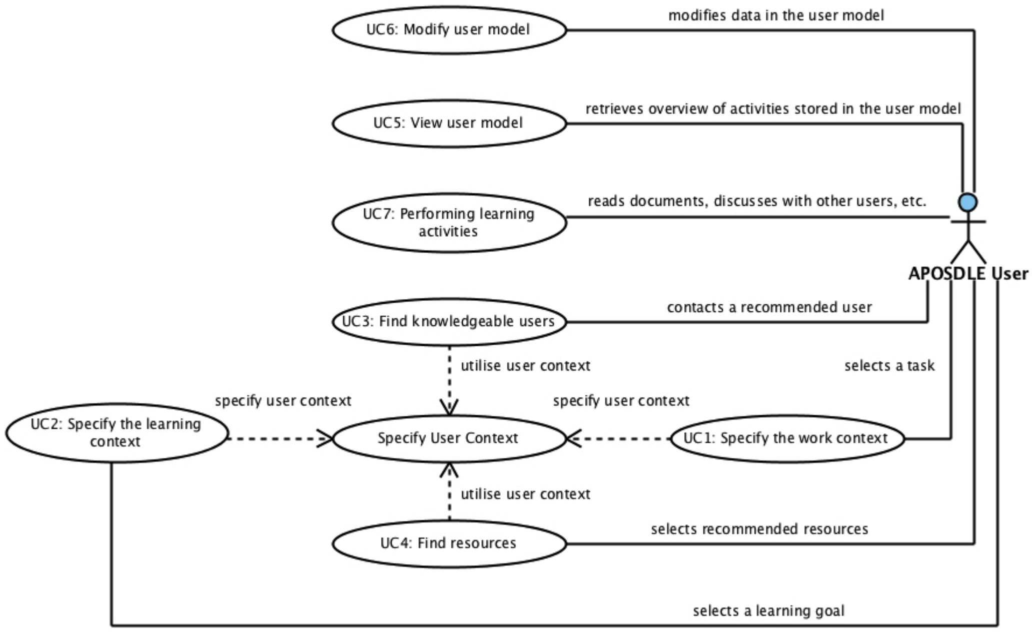

Figure 4.1: Use cases related to the functionality of APOSDLE's user model. Relations between use cases show how they depend on each other.

design and development of a user model. The use case 'Specify user context' which contains UC1 and UC2 is the entry point for users into the system. Based on a users' context, a WIL system provides relevant resources (UC4) and knowledgeable users (UC3). To investigate the data a WIL system collects, users are able to view (UC5) and modify (UC6) certain aspects of their user model.

4.3.1 UC1: Specify the work context

Table 4.1 defines how users manually set their current task in the WIL Client, or how a the WIL Client automatically detects the current task.

4.3.2 UC2: Specify the learning context

Table 4.2 describes how users select a learning goal for which the WIL Backend generates a list of learning resources.

4.3.3 UC3: Find knowledgeable users based on current user context

Table 4.3 describes how the WIL Backend infers a list of knowledgeable users which fit to a user's current work task.

Table 4.1: Use Case 1: Specify the working context

Use Case Item	Item Description
Actors	User, WIL Client, WIL Backend
Problem Statement	A user needs additional information to accomplish her task at hand and wants to query organisational memories for related information.
Summary	A user states her current task at hand either manually by selecting a task, or her task has been automatically detected. This context information will be used to find related documents and people by the WIL Backend (see UC3 and UC4)
Triggers	1.) The WIL Client automatically detects the current task at hand of a user. 2.) A user manually selects a task in the WIL Client.
Preconditions	A model describing all tasks of the domain is available in both the WIL Client and the WIL Backend. The WIL Backend can store the current tasks for further processing.
Success Scenario	1.) The user browses through the task in the WIL Client. 2.) The user selects a single task in the WIL Client. 3.) The WIL Client sends the task information to the WIL Backend. 4.) The WIL Backend stores the task for the user as current context.
Extended Scenario	1.) The WIL Client detects a task and recommends during the normal course of a user's work activities. 2.) The WIL Client recommends this task as a new context to the user. 3.) The user accepts the recommendation. Step 3 and 4 are the same as for the "manual selection"
Assumptions	-

Table 4.2: Use Case 2: Specify the learning context

Use Case Item	Item Description
Actors	User, WIL Client, WIL Backend
Problem Statement	A user needs to find resources which provide her with the missing information for accomplishing the task at hand.
Summary	Based on a selected task, the WIL Client presents a list of recommended learning goals to the user. The user can browse through this list and selects one of the learning goals. The selected learning will be transferred to the WIL Backend to define the current learning context.
Triggers	1.) A user selects a new task in the WIL Client. 2.) A user changes her current learning goal.
Preconditions	A model describing the dependency between tasks and learning goals of the domain is available in the WIL Backend.
Success Scenario	1.) The user specifies her current work context (see UC1) 2.) The WIL Client requests a list of learning goals for the current work context. 3.) The WIL Backend generates a list of recommended learning goals and sends them to the client. 4.) The WIL Client presents the list to the user requesting her to select a learning goal. 5.) The user browses through the list and selects a learning goal. 6.) The WIL Client sends the learning goal to the WIL Backend.
Extended Scenario	The user wants to change an already selected a learning goal. 1.) The Success Scenario was executed once. 2.) The user browses through the existing list of learning goals and selects a new one. 3.) The WIL Client sends the learning goal to the WIL Backend to update the learning context.
Assumptions	-

Table 4.3: Use Case 3: Find knowledgeable users

Use Case Item	Item Description
Actors	User, WIL Client, WIL Backend
Problem Statement	Information is not only stored in organisational memories but is captured in the minds of knowledge workers (experts) within the organisation. Users might not know all these experts for a certain topic in an organisation.
Summary	Based on the current user context of a user the WIL Backend generates a ranked list of knowledge workers who are knowledgeable about the topic of the current context. A user can browse this list and choose a knowledge worker to get into contact with.
Triggers	1.) A user has selected a new learning goal in the WIL Client. 2.) The WIL Client updates the list of users automatically after a fixed time span.
Preconditions	1.) The WIL Backend maintains the current state of knowledge for all users. 2.) The WIL Backend knows the current work context of the user requesting the list of knowledgeable users
Success Scenario	1.) The user selects her user context (see UC1 and UC2). 2.) The WIL Client retrieves a list of ranked knowledgeable users from the WIL Backend and presents it to the user. 3.) The user selects a knowledgeable person 4.) The WIL Client starts the process of establishing a collaboration.
Extended Scenario	The user has previously set her user context. Step 1.) from the Success Scenario is skipped. 2.) The WIL Client updates the ranked list of users after a fixed time span. Steps 3.) and 4.) remain the same.
Assumptions	The ranking takes into account the user model, the current work context, the availability of users, and optional factors like organisational distance or social distance.

4.3.4 UC4: Find resources based on current user context

Table 4.4 describes how the WIL Backend uses the current work task to find related resources in organisational memories.

Table 4.4: Use Case 4: Find resources

Actors	User, WIL Client, WIL Backend
Problem Statement	Knowledge workers need to find resources relevant for their current work at hand.
Summary	The WIL Client presents users with a list of resources related to their current work context. Resources can be textual as well as audio-visual content.
Triggers	A user has selected a new learning goal in the WIL Client.
Preconditions	1.) The WIL Backend knows the current user context of the user requesting the list of users. 2.) The WIL Backend can utilize the current user context to find related resources.
Success Scenario	1.) The user selects her learning context (see Table 4.2). 2.) The WIL Backend searches for resources related to the current working and learning context. 3.) The WIL Client receives the list of related resources and presents them to the users.
Extended Scenario	In case no resources have been found an empty list will be shown to the user.
Assumptions	-

4.3.5 UC5: View user model

Table 4.5 describes how users can investigate information maintained in the user model located in the WIL Backend.

4.3.6 UC6: Modify user model

Table 4.6 describes how users can modify certain information in the user model.

4.3.7 UC7: Performing learning activities

Table 4.7 describes how the WIL Backend supports users with different learning activities which may be available in different media types.

Table 4.5: Use Case 5: View user model

Use Case Item	Item Description
Actors	User, WIL Client, WIL Backend
Problem Statement	The user wants to get an overview of her user model. To update the current user model a user needs a way to access the user model.
Summary	A user model may contain personal information such as name, email address, job title, etc., and usage data which has been recorded during the interaction with the WIL Client. To provide users access to their model, the WIL Client offers a user modelling tool (UMT) to view and change certain aspects of the user model. The tool also enables users to get some details about other users (e.g., when they are recommended as knowledgeable for a topic).
Triggers	1.) A user has opened the UMT from the WIL Client. 2.) A user wants to get detailed information about a knowledgeable person who has been recommended.
Preconditions	The WIL Backend maintains an up-to-date user model.
Success Scenario	1.) The user opens her own user model from the WIL Backend. 2.) The WIL Client starts the UMT. 3.) The UMT loads the model from the WIL Backend. 4.) The UMT displays the different types of information.
Assumptions	-

Table 4.6: Use Case 6: Modify user model

Use Case Item	Item Description
Actors	User, WIL Client, WIL Backend
Problem Statement	The user wants to change her user model.
Summary	The WIL system allows users to modify parts of the data stored in their user models. The UMT offers an user interface to perform modifications which are updated in the WIL Backend.
Triggers	A user has selected a data entry in the UMT for modification.

Table 4.7: Use Case 7: Performing learning activities

Actors	User, WIL Client, WIL Backend
Problem Statement	The user extracts information from resources
Summary	The WIL Backend provides users with different types of resources to interact with.
Triggers	1.) A user has opened a learning event. 2.) A user has opened a resource. 3.) A user has contacted a knowledgeable person.
Preconditions	The WIL Backend is able to link interactions with resources (i.e. learning activities) with a user's current learning context.
Success Scenario	1.) The user performs either UC3, UC4, or requests a list of learning events. 2.) Depending on the type of resource, usage data will be sent to the WIL Backend. 3.) The WIL Backend is updated with the new usage data.
Assumptions	Data contained in the WIL Backend is categorised into private and public data. Only public data is shown to other users than the owner of the model.

4.4 Requirements from Use Cases

The following requirements can be derived from UC1:

- **Use-Case Requirement 1.1 (UCR1.1):** A task is an information for the WIL System to determine a user's current working context.
- **Use-Case Requirement 1.2 (UCR1.2):** The WIL System shall be able to receive a task from two different sources, an automatic detection mechanism or a manual selection by a user.

The following requirements can be derived from UC2:

- **Use-Case Requirement 2.1 (UCR2.1):** The WIL System shall provide a list of learning goals for a users' current learning context.
- **Use-Case Requirement 2.2 (UCR2.2):** The WIL System shall save a users' selected learning goal as the current learning context.

The following requirements can be derived from UC3:

- **Use-Case Requirement 3.1 (UCR3.1):** The WIL System should deliver a ranked list of knowledgeable users for the current context of a user.
- **Use-Case Requirement 3.2 (UCR3.2):** The calculation of the ranking of knowledgeable people shall be triggered by setting a learning goal.
- **Use-Case Requirement 3.3 (UCR3.3):** The WIL System should maintain up-to-date information about users' knowledge about all available learning goals.
- **Use-Case Requirement 3.4 (UCR3.4):** The WIL System should store the availability of users.
- **Use-Case Requirement 3.5 (UCR3.5):** The calculation of the ranked list of knowledgeable users shall include previous activities stored in the user profile, the current working context (task and selected learning goals), and the availability of users.
- **Use-Case Requirement 3.6 (UCR3.6):** The calculation of knowledgeable users should be flexible to take into account factors like organisational or social distance.

The following requirement can be derived from UC4:

- **Use-Case Requirement 4 (UCR4):** The current working and learning context is made available to other components of the WIL System to find relevant resources based on this context.

The following requirement can be derived from UC5:

- **Use-Case Requirement 5 (UCR5):** The WIL System shall be able to assemble stored usage data and provide it to WIL Clients.

The following requirement can be derived from UC6:

- **Use-Case Requirement 6 (UCR6):** The WIL System should provide a way to modify data contained in the user model.

The following requirement can be derived from UC7:

- **Use-Case Requirement 7.1 (UCR7.1):** The user model of the WIL System shall be able to store usage data related to learning activities.
- **Use-Case Requirement 7.2 (UCR7.2):** The user model of the WIL System shall be able to link learning activities to the users' current learning context.

4.5 Enterprise Models

The APOSDLE system has been designed as a framework system that can be customised to any learning domain by creating three different types of *enterprise models*: the domain model, the task model and the learning goal model. These enterprise models must be created by domain experts and knowledge engineers before APOSDLE can be used.

4.5.1 Domain Model

The purpose of the domain model is to provide a semantic and logic description of the work domain which also constitutes the learning domain of an APOSDLE deployment environment. The domain is described in terms of concepts, relations, and objects that are relevant for this domain. Technically speaking the Domain Model is an ontology that defines a set of meaningful terms which are relevant for the domain and, which are used to classify and retrieve resources and knowledgeable people.

4.5.2 Task Model

The objective of the task model is to provide a formal description of the tasks the knowledge worker can perform in a particular domain. The YAWL[2] workflow system is used as conceptual basis for the task modelling. This formal description is used in various ways within APOSDLE. One aspect is the task detection, which needs a set of predefined tasks. Another important aspect is the dependent task-competence mapping forming the learning goal model.

4.5.3 Learning Goal Model

The learning goal model within APOSDLE establishes a relation between the domain model and the task model. It maps tasks of the task model to concepts of the domain model. A learning goal describes knowledge and skills needed to perform a task, with respect to a certain topic in the domain model. In other words, each learning goal refers to one topic in the domain model. This relationship is necessary for a number of functionalities provided by the APOSDLE user model services. For example, it enables the determination of user skills from past task executions and other knowledge indicating events (People Recommendation, see Section 3.3.2.2), or the identification of a user's learning need with respect to a certain task (Learning Goal Recommendation, see Section 4.8.1).

4.6 APOSDLE Context Detection

In APOSDLE, learning is integrated into the work environment and thus way is needed of observing what users are doing in order to identify their current task or topic they

[2] http://www.yawlfoundation.org

are working on. In APOSDLE, the task and topic detection is realised by a specialised agent (Lokaiczyk et al., 2007) part of the APOSDLE Client. This agent observes the user interactions (e.g., key strokes, mouse movements, applications specific actions) with typical MS Office and internet applications and compares them to previously learned interaction patterns of the organization. APOSDLE monitors a user's daily work activities.

4.7 APOSDLE User Model

The APOSDLE User Model is an overlay of the topics in the Domain Model. The APOSDLE user model is automatically maintained applying the approach of knowledge indicating events (KIE, see Section 3.3.1). In a nutshell, different types of naturally occurring actions of users are observed and inferences are made on the user's underlying knowledge level in a certain topic. To give and example from Georg, the innovation consultant from the scenario in Section 4.2, the repeated execution of a task ‚preparing a creativity workshop' can be seen as a KIE for topics such as ‚creativity technique' and ‚workshop moderation'. Another KIE for the topic 'creativity technique' could be that a person has been contacted repeatedly about this topic. KIE can be interactions with the APOSDLE Desktop Client, or interactions with other programs such as web browsers, office applications etc.

Whenever a user executes a task, or carries out another KIE within the APOSDLE environment the counter of the topic related to that KIE within his or her user model is incremented. If the KIE is a task, the relationship between tasks and competences in the learning goal model can be exploited: one can infer that the user has knowledge about all the topics related to that task. Therefore, by means of an inference service (see Section 3.3.3.3), information is propagated along the relationships defined by the learning goal model, and the counter of all topics related to the task is also incremented. The same procedure is applied to a number of other KIEs.

APOSDLE distinguishes three different levels of expertise: Learner, Worker and Supporter. Each of the KIE is assigned to one of these levels. For instance, 'carrying out a task' is a KIE for the 'Worker' level, whereas 'being contacted by another person' would indicate a 'Supporter' level. Thus, whenever a KIE occurs within the APOSDLE environment, the levels for all topics related to the KIE are updated. This means, at any point in time, an algorithm in the APOSDLE user model decides in which level of expertise a user is with respect to every topic in a domain.

4.8 APOSDLE User Model Services

For the APOSDLE system, I have designed service implementations for all types of WIL user model services proposed in Section 3.3.3. Figure 4.2 presents an overview of APOSDLE user model services and how data is exchanged with the user model and corresponding APOSDLE Client applications.

The APOSDLE system implements two different logging services. The Work Context Logging Service is dedicated to collect executions of tasks corresponding to the task model (delivered from a task detection agent Lokaiczyk et al., 2007). Logging information consists of a user identifier, a task identifier and an optional timestamp (depends on privacy settings). The second logging service, Resource Activity Logging Service collects all activities related to resources presented to users. Such actions are reading documents, engaging in learning events, or contacting another user.

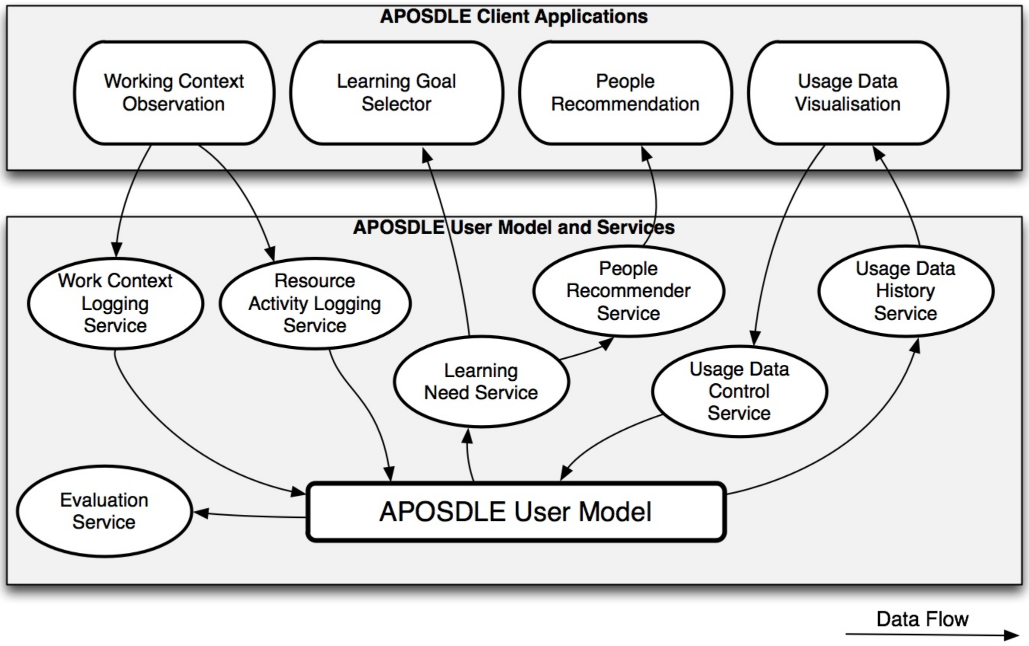

Figure 4.2: APOSDLE User Model Services

In order to allow users to examine the information WIL services have gathered, APOSDLE offers two production services. The Usage Data History Service delivers a history of task executions and all resource-based actions. The output of this service is basically a history of all KIEs. Another feature is that relations between events are also preserved. This provides a way to analyse which steps users have taken when doing a certain task. It features also the links to outputs generated by Inference Services (for example a ranked list of learning goals inferred by the Learning Need Service (see also the Learning Goal Recommendation Section 4.8.1).The Evaluation Service is another kind of production service. It is specially designed to export different aspects of usage data for evaluation outside the APOSDLE system. In APOSDLE this service generates files containing detailed information about task executions, system usage, and information from inference services.

From a conceptual perspective, the most important user model services within APOSDLE are the two Inference Services, the Learning Need Service and the People Recommender Service. These will be introduced here; a more detailed description will then be provided in the following sections (Section 4.8.1 and Section 4.8.2). The Learning Need Service allows to compute a learning need for a user. Its design is driven by the goal to support knowledge workers based on their knowledge level. The People Recommender Service aims at finding people within the organization which have expertise related to the current learning goal of the user.

APOSDLE implements two control services. The Usage Data Control Service allows users to modify and delete any usage data. APOSDLE Clients present users with a task history provided by Usage Data History Service, and invoke the Usage Data Control Service to delete task executions selected by users. A dedicated privacy component (part of the APOSDLE platform) also accesses this service to enforce certain privacy policies on usage data.

4.8.1 APOSDLE Learning Goal Recommendation

As explained in Section 3.3.2.1, one of the main goals of a WIL system is to provide a user with learning content in a highly adaptive manner. Within APOSDLE, a user's learning need is inferred by the *Learning Need Service* in three steps:

Starting from the user's current task, the user model is queried to retrieve the required learning goal vector *lgv* for this task. The vector *lgv* represents all learning goals for the current task. The user model is again queried with the required learning goal vector *lgv* as parameter to retrieve the current knowledge levels vector *klv* for the user. For each topic in the domain, the vector *klv* contains the current knowledge level. The second step generates the list of recommended learning goals. The lower the knowledge level (learner < worker < supporter) of a topic in the learning goal vector *lgv*, the higher the rank of the learning goal. The 'most required' learning goal is therefore listed on the top of the learning need list. If there are multiple topics with the same knowledge level, these topics which are assigned to more tasks in the knowledge base are ranked higher. This simple rule ensures that the prerequisite relation that is assumed between topics of the knowledge base is taken into account (see Ley et al., 2010a). The learning need is used by APOSDLE in two ways. An application running in the working environment of the user visualizes the result as a ranked list. The first learning goal is automatically pre-selected, which invokes an Associative Retrieval Service (see Scheir et al., 2008) to find resources relevant for the learning need.

For instance, remember Georg, our innovation consultant from the scenario (Section 4.2). Imagine that Georg interacts with various desktop and web applications on this computer. From this interaction, APOSDLE recognizes that Georg is trying to prepare a creativity workshop. APOSDLE compares the requirements of this task as modeled in the underlying enterprise models (Section 4.5) with the skills that Georg has available as represented in the user model. APOSDLE detects a learning need comprising of the 'ability to apply creativity techniques', and the 'knowledge about how to prepare an agenda for a creativity workshop'.

4.8.2 APOSDLE People Recommendation

As mentioned in Section 4.8, the People Recommender Service is an important inference service within APOSDLE. It aims at finding people within the organization who have expertise related to the current learning goal of the user determined by a task or topic he or she is working on. That is, recommendations of colleagues are always provided depending on a user's current learning need that can be one or several topics from the domain model (see Section 4.8.1). For each of the topics in a user's learning need, APOSDLE recommends learning materials and knowledgeable colleagues. Similarly, if APOSDLE detects that a user deals with a certain topic, the system suggests learning materials and knowledgeable colleagues directly for this topic. Let us go back to the scenario of Georg: Consider that APOSDLE has detected a learning need comprising of the 'ability to apply creativity techniques', and the 'knowledge about how to prepare an agenda for a creativity workshop'. In order to support Georg, APOSDLE suggests a variety of learning materials, and also a list of knowledgeable colleagues who can provide help. On the top of the list of knowledgeable colleagues is Susan, who has both these skills and thus can help Georg in his task.

But how did APOSDLE 'know' that Susan is a knowledgeable colleague who can help? The recommendation workflow is shown in Figure 4.3. Users specialised in certain topics have high knowledge levels for these topics in their user models. To infer knowledgeable users, the People Recommender Service uses theuser model to access user-related information. To process user-related information, the People Recommender Service can be configured with different algorithmic components executing the recommendation task.

One of the algorithms retrieves the history of all KIEs related to a topic from one of the APOSDLE Production Services and generates a user-knowledge level matrix. The user-knowledge level matrix is calculated by summing up all KIEs mapped to a knowledge level. A row of the matrix specifies whether a user is currently categorized either as learner, worker, or supporter. In the next step, the algorithm removes all users with lower knowledge levels compared to the user receiving the recommendation. The remaining users are then ranked according to their knowledge levels. The most knowledgeable user will be ranked highest.

Moving back to the scenario, the People Recommender Service recommends Susan to Georg as a knowledgeable person because Susan has a higher knowledge level in these two learning goals. Knowledgeable persons are found by comparing the current knowledge levels of all users with the knowledge level of the user who will receive the recommendation. Another algorithm applies a weighting function boosting, lessen, or even masking out certain KIEs when creating the user-knowledge level matrix. Additionally, APOSDLE contains a basic version of a forgetting function which applies a window on the history of KIEs. With this window it is possible to take into account a certain timeframe only.

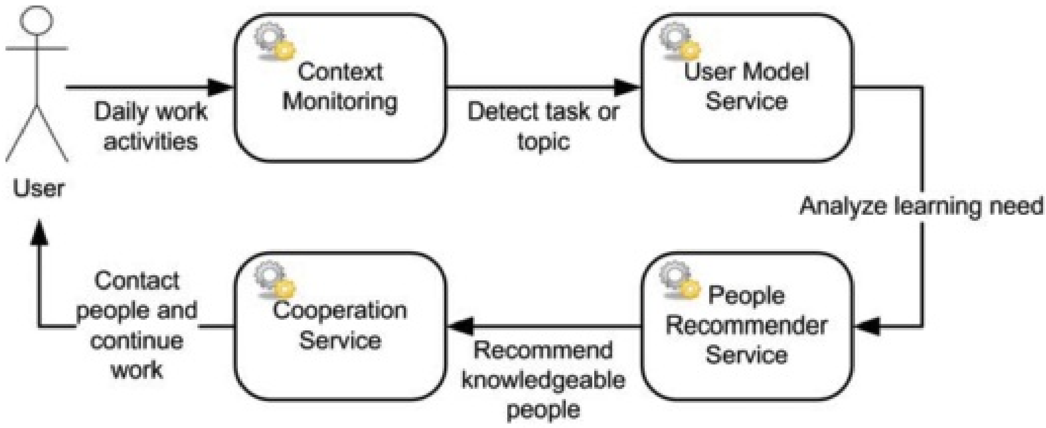

Figure 4.3: Recommendation in the APOSDLE System

Clicking on the name of a knowledgeable colleague in the list of recommended people in APOSDLE, the user can inspect organizational (department, contact details, etc.) and personal information (name, picture, etc.) about the colleague. When a user clicks on the contact' icon below the name of a person in APOSDLE Suggests, APOSDLE displays a ranked list of communication tools (e.g., Skype, email, phone, etc.) to choose from. The People Recommendation Service can be configured to also use the availability status of users as ranking criteria. This setting allows recommending only users currently available.

The People Recommender Service provides similar functionality as the expert finding systems described in (Maybury, 2006). Users specialised in certain topics are represented in the User Model with high knowledgelevels for these topics. Other users can now individually be provided with colleagues having equal or higher experience. Compared to the MetaDoc system (Boyle and Encarnacion, 1994) the APOSDLE service uses a more dynamic way of identifying experts: Knowledgeable users are always identified compared to the knowledge of the user who will receive the recommendation. To infer knowledgeable users, the APOSDLE People Recommender Service utilises the Learning Need Service to retrieve knowledge levels for all users.

To compare APOSDLE's People Recommender Service with other expert finding systems, in the following it is classified according to the Intuitive Domain Model for Expert Finding system presented by Yimam-Seid and Kobsa (2003), see also Section 3.3.2.2. In order to deduce the expertness, an APOSDLE Logging Service tracks specific user activities associated with topics in the underlying enterprise models and stores these activities together with the respective topics in the user model (expertness deduction operation). These user interactions are interpreted as KIE for three levels of expertise: learner, worker, and supporter (Expertise Indicator Source). The expertise indicator extraction operation is domain knowledge independent; a topic-name-association is derived from user behavior. APOSDLE can be customized to any organizational learning domain without changing the expertise indicator operation of the People Recommender Service. The domain model underlying the APOSDLE user model is a centralized enterprise model that comprises tasks, topics and learning goals of the application domain.

Table 4.8: Classification of the APOSDLE People Recommender Service according to the Intuitive Domain Model of Expert Finding systems

Domain Factors	APOSDLE People Recommender Service
Expertness Deduction Operation	User activities associated with topics in the enterprise model; explicit feedback from user about deduced expertise
Expertise Indicator Source	User interactions with the APOSDLE Desktop Client (e.g., viewing and annotating documents, contacting people) and other programs (e.g. searching for a topic in a web browser, creating a document with an office application)
Expertise Indicator Extraction Operation	Domain knowledge independent; Topic-name-association derived from user interactions
Expertise Model	Associated to user model which is an overlay of the centralised enterprise model
Query Mechanism	Explicit query for a task/topic of interest; Induction of user's needs with context monitoring agent
Matching Operations	Exact match of expertise vectors for a single topic, or for a range of topics
Output Presentation	Ranked list of knowledgeable users, personal information, tag cloud of topics most worked on

4.9 Summary and Outlook to Next Chapter

In this chapter, the APOSDLE system, an adaptive WIL system was introduced. Then, seven use cases dedicated to the APOSDLE User Model were shown, and requirements for the user model, user model services and software architecture were derived. The three different types of enterprise models underlying APOSDLE were presented (domain model, task model, and learning goal model). These build the basis for the APOSDLE User Model which is designed as overlay of the enterprise models. The unobtrusive maintenance of APOSDLE's user model by means of KIE was explained. Finally, a functional description of the different APOSDLE User Model Services was provided. Because they constitute the most important user model services within APOSDLE, the Learning Need Service and the People Recommender Service were described in more detail.

In the next chapter, I will describe extensively how I implemented the APOSDLE User Model and the APOSDLE User Model Services. Then, I will briefly sketch the four work domains for which APOSDLE was instantiated.

Chapter 5

Technical Implementation and Application of the APOSDLE UPS

This chapter presents the implementation of the APOSDLE User Model and APOSDLE User Model Services of the adaptive work-integrated learning system APOSDLE. Moreover, four work domains are being presented for which APOSDLE was instantiated. Part of this work has been published in the following reports co-authored by me: Project Deliverable on Software Architecture for 2nd (APOSDLE Consortium, 2007b) and 3rd (APOSDLE Consortium, 2008) Prototypes.

The core of my master's thesis is the prototypical implementation of a number of services related to the user model of the adaptive work-integrated learning system APOSDLE. Because in the context of the APOSDLE project, the user model was termed 'user profile', in the following, this collection of services will be termed APOSDLE User Profile Services (APOSDLE UPS).

The APOSDLE UPS are based on the conceptual foundations presented in the previous chapters, and on the requirements derived from theory (see Section 6.1) and from the APOSDLE use cases (see Section 6.2).

In the following sections, I will describe in detail the software architecture of the APOSDLE UPS and directly related services.

5.1 Scope

The software component presented in this chapter is the APOSDLE UPS, an adaptive Service-oriented User modelling component which was designed and developed as part of my master's thesis. It aims at implementing a set of user model services as have been presented in Chapter 4.

First, the methodology will introduce the steps which have been undertaken in developing this prototype. An architectural description points out the prototype from different viewpoints, and provides detailed insights from top-level service interfaces down to the database backend. The chapter ends with a description of four real application cases in which the APOSDLE system (including the APOSDLE UPS) has been deployed.

What is out of scope but important: The APOSDLE system as a whole is a work-integrated system meaning that it will deployed in real working environments. Real working environments require additional measures than lab experiments when it comes to using systems during every day work. This work puts its focus on the development on the core system itself leaving some issues for the project to solve before deploying it in real environments.

Compliance to learning standards issued by different committees (IEE LTSC[1], IMS[2], ADL[3]) is one of the topic which indeed is important when it comes to connecting different learning systems together. For APOSDLE UPS, compliance with standards was kept in mind for the overall architecture, but was not implemented in the APOSDLE UPS. Exporting data according to a specific standard can be implemented as e.g., an additional service.

Security and privacy considerations are another important topic and very much depend on the environment APOSDLE is used in. Therefore, the project investigated legal and technical aspects to collect a list of requirements. Based on this list, a security and privacy component was designed to safeguard APOSDLE.

User interfaces are the only part of a learning system users actually see and interact with. For APOSDLE UPS, a user interface is equally important but mostly dependents on the environment it will be used in. Therefore, the prototypical implementation offers a well-defined interface for developers implementing user interfaces on top of APOSDLE UPS.

Dependencies to other software components are described to clarify their functionality, and how the APOSDLE UPS with them.

5.2 Software Documentation Specification

For documenting the APOSDLE UPS, the standard IEEE 1471-2000 (IEEE Computer Society, 2000) was chosen as a guideline to present the software architecture. The use of IEEE 1471-2000 was required by the APOSDLE project as the standard for documenting the software system. IEEE 1471-2000 recommends a common practice how a software-intensive should be documented but leaves it up to the project how detailed each of the parts are documented. It is organised into one or more views addressing concerns of the projects' stake holders. For documenting views the *4+1 Views* model by Kruchten (1995) was chosen. Besides other architectural view models such as Siemens Four Views (Hofmeister et al., 2000) or the SEI view model (Clements et al., 2002), the *4+1 Views* model fits very well for iterative development processes, addresses concerns important for the APOSDLE UPS (e.g., client-server concerns, implementation, or process concerns), and incorporates different stake holder roles. The first part of this architectural description will define all stakeholders, concerns, and the structure four viewpoints. A viewpoint defines a set of concerns, methods and techniques which should be used. A view is a viewpoint applied to a concrete project. The second part describes the four views of the APOSDLE UPS.

5.2.1 Stakeholders

Stakeholders have different interests in the software architecture of APOSDLE UPS depending on their role. Stakeholders may have different roles and concerns about the software which have to be addressed by this description. The following paragraphs describe each of the roles and stakeholders.

[1] The IEEE Learning Technology Standards Committee, http://www.ieeeltsc.org
[2] IMS Global Learning Consortium, http://www.imsglobal.org
[3] Advanced Disitributed Learning, http://www.adlnet.gov

5.2.1.1 Project Manager

Two of the most important tasks of the project manager is to plan and align a project's tasks, and to monitor the project's execution. Within APOSDLE, this role is assigned to Joanneum Research (JRS) and Know-Center (KC). JRS is responsible for overall management activities and KC is being concerned to coordinate the software development of the APOSDlE system.

5.2.1.2 Users

Users are the group of stakeholders who are consuming features of the APOSDLE UPS as a software component providing user modeling capabilities and user-adaptive services. They could, for example, be system integrators connecting the UPS to other systems, or developers who may want extend features. Another way would be to extend an existing application with adaptive features provided by the APOSDLE UPS. The APOSDLE UPS does not provide an interface for end-users who want to interact with it in an application (like for example a web browser or a spreadsheet application) style manner. Therefore, end-users are only considered as indirect users as they need an application processing and visualising data delivered by APOSDLE UPS services.

5.2.1.3 Acquirers

Acquirers are considered as stakeholders who are buying, licensing, or owing APOSDLE UPS. Acquirers are to some extend the same persons and institutions as the *Users* described before. KC is owning the UPS as well as using and enhancing it. Indirect acquirers or customers could be organisations licensing or buying the whole APOSDLE system.

5.2.1.4 Developers

Developers of APOSDLE UPS are involved in the transformation of use case descriptions into requirements for the software systems. Based on these initial requirements and further constraints given by Users and Acquirers, their main task is to design, implement, test, and evaluate APOSDLE UPS as a software system.

5.2.1.5 Maintainers

Maintainers are running APOSDLE UPS as a software component or standalone server. Therefore they need detailed information about how to setup and maintain APOSDLE UPS during runtime. In APOSDLE maintainers are on the one hand TUG who prepare the setup and configurations of the overall APOSDLE system for APOSDLE's application partners. On the other hand IT departments at application partner sites' are maintaining APOSDLE installations. APOSDLE's application partners are ISN (Austria), CCI (Germany), CNM (Germany), and EADS (France). Additionally, the APOSDLE system has been deployed at the FernUniversität Hagen (Germany).

5.2.2 Concerns

The following list of concerns which have been considered when designing the architecture of APOSDLE UPS. The first four concerns have been compiled based on the specification laid out in IEEE 1471-2000. They mostly cover questions which may arise from different roles of stakeholders. The latter two are based on the concerns presented in the APOSDLE Software Architecture document (APOSDLE Consortium, 2008).

5.2.2.1 Appropriateness of the system for fulfilling its missions

Appropriateness of the systems addresses the implementation of initially specified requirements and constraints. Changes of requirements during the process of development are also covered by this concerns. Besides fulfilling requirements applicability of the delivered system is a topic of interest.

5.2.2.2 Feasibility of constructing the system

This concern raises the question whether the APOSDLE UPS can be realised in terms of given resources, scheduled time frames and the technology used to develop the UPS.

5.2.2.3 Risks of system development and operation

This concern comprises all questions about risks during the process of development, and later installation and usage of APOSDLE UPS: Technical risks are for example problems occurring during the use of third-party libraries and frameworks, complexity of implementing designed algorithms, etc. Risks which may affect users and acquirers could be licensing schemes of third party software used by APOSDLE UPS, organisational policies for collecting and storing user-related information, etc.

5.2.2.4 Maintainability, deployability, and evolvability of the system

Maintainability and deployment are important for APOSDLE UPS as it is designed to run in working environments where proper function must be ensured. Questions brought up by this concern are "Which maintenance tasks have to be executed?" or "Which requirements have to be met by a target platform running APOSDLE UPS?". Evolvability addresses the ways how it can be improved and extended with new functionality.

5.2.2.5 Structure of system into functional entities connecting interfaces

This concern specifically deals with the internal structure of APOSDLE UPS in terms separation of functionality into closed entities, interfaces connecting theses entities, modularity of the structure, and interfacing of third party software.

5.2.2.6 Performance of the system

The last concern covers issues about the performance of ASUMS in real world situations. For APOSDLE UPS it is important to deliver its functionality in a stable and well-performing way in a clearly defined environment. This concern further deals with possible effects of APOSDLE UPS on other systems depending on different environments it may be embedded in.

5.2.3 Development Process

As the APOSDLE UPS prototype development ran in parallel to the overall APOSDLE development process, synchronisation of development activities and requirements were needed to ensure successful integration. First, a development process had to be chosen which fits the following requirements

- Shall be appropriate for the small prototype development project
- Allows to iteratively develop the prototype
- Supports changes of requirements

A review of existing software development processes and best practices lead to the conclusion to use an iterative and incremental development process. The well known waterfall model would fit for this small-scale project but lacks to adapt to changes in early stages of the project, and does not support iterative development. A V-model approach also lacks support for iterations and is more dedicated to development of large scale systems. The rather generic iterative and incremental process of development recommends four subsequent steps which will be walked through in every iteration. At the beginning of an iteration the new development unit is planned, new or existing requirements are reviewed, and integrated in the subsequent steps of design, implementation, deployment, and evaluation. For the development of this prototype the generic approach was slightly adapted to allow for a test-driven development. Based on the requirements a set of features are derived to create the design. According to the '4+1 Views' approach the design is split into a Logical, Process, Development, and Physical View. Implementation and testing phase are designed as an additional cycle to ensure continuos testing of implementation units. The last phase of an iteration is to evaluate the outcomes against the requirements to identify possible problems and deviations. Results of the evaluation phase will be used to plan the next iteration.

5.2.4 Viewpoints

5.2.4.1 Logical Viewpoint

The Logical Viewpoint (LVP) is the transformation of functional requirements into a high-level system view. Stakeholders are provided with a high level view on the systems' functions and how they depend on each other. The granularity of the elements is on a package level in terms of the UML notation. The main goal is to present the major components and their responsibilities. Furthermore, this viewpoint addresses the functions the system should provide in terms of services to its Users (see Table 5.1).

5.2.4.2 Process Viewpoint

The Process Viewpoint (PrVP) covers dynamic and non-functional requirements of the APOSDLE UPS architecture. Non-functional requirements cover concurrency of processes and distribution of functionality over physical environments. Dynamic aspects describe how messages flow between elements in the LVP, and how processes are executed. Processes in PVP can be defined as a sequence of tasks creating a single path of execution. Examples are the process of configuring, starting, collecting usage data, or stopping the APOSDLE UPS. Different levels of abstraction help to differentiate between major processes (see Table 5.2).

5.2.4.3 Development Viewpoint

The Development Viewpoint (DVP) aims at identifying and structuring the software into subsystems and modules which serve as a basis for developers. A subsystems is denoted

Table 5.1: Logical Viewpoint: Stakeholders, Concerns, and Language

View Point Item	Item Description
Stakeholders	Project Manager, Acquirers, Maintainers
Concern 1	Is the APOSDLE UPS connected to all components necessary? The LVP provides a high-level overview of APOSDLE's overall system architecture which presents all components with connectors indicating how they are conceptually linked together.
Concern 2	How is the functionality of APOSDLE UPS distributed internally? The LVP shows how functions are structured into well-defined sub-components, and how they depend on each other. Reviewing the LVP could reveal scattered functionality and provides a good starting point for investigating its reusability.
Concern 3	Have all requirements been considered in the conceptual design? The LVP shows all components mapped to requirements derived from the Use Case Viewpoint. Thus, it allows to check whether all requirements have been addressed and integrated into the architecture of APOSDLE UPS.
Concern 4	Is it feasible to implement APOSDLE UPS with available resources? The LVP provides an overview which kind of components are needed to realise the UPS. Based on this information, a first feasibility check can be done to estimate implementation efforts.
Language	UML 2.0

Table 5.2: Process Viewpoint: Stakeholders, Concerns, and Language

View Point Item	Item Description
Stakeholders	Developers, Maintainers, Project Managers
Concern 1	Which information is exchanged between components? The PVP shows the flow of control between components of APOSDLE as function calls.
Concern 2	What are the steps to deliver results as defined in the APOSDLE use cases? The PVP presents all processes which have been derived from the use cases. This viewpoint allows to cross-check behavior and steps defined by uses cases with the actual flow of control in APOSDLE.
Concern 3	Which requests from users need to communicate with other backend systems? To answer this question, the PVP helps in identifying the processes targeting an external system.
Concern 4	Which requests from users can be handled in parallel? The PVP specifies in detail which of the processes could be handled in parallel, and how other components are affected by parallel requests.
Language	UML 2.0

as connecting a set of modules together. The viewpoint addresses internal requirements (for example derived form scenario views) and constraints imposed by programming languages and third party software. To address these issues, Kruchten (1995) recommends the introduction of layers which group modules and subsystems into domain-independent and domain-specific parts. This viewpoint also offers a level of abstraction to plan development timelines and estimate development costs (see Table 5.3).

Table 5.3: Development Viewpoint: Stakeholders, Concerns, and Language

View Point Item	Item Description
Stakeholders	Developers
Concern 1	Which functionality is provided and consumed by components?
Concern 2	How are components structured into classes?
Concern 3	How do third-party libraries connect to components?
Concern 4	Which parts of the systems' functionality can be covered by existing solutions?
Concern 5	Are concerns separated into functional units?
Concern 6	How domain-dependent are interfaces of components and classes?
Language	UML 2.0

5.2.4.4 Physical Viewpoint

The Physical Viewpoint (PhVP) maps software to hardware. Hardware could be networks, servers, or client computers. Elements to be mapped onto these hardware are subsystems and modules described in development views. Besides mapping, the PhVP addresses non-functional requirements such as availability, reliability, performance, and scalability of this software-hardware mapping (see Table 5.4). In APOSDLE, the PhVP specifically addressed the physical architecture of the client and server part, as well as backend and external systems (e.g., network drives and databases).

Table 5.4: Physical Viewpoint: Stakeholders, Concerns, and Language

View Point Item	Item Description
Stakeholders	Maintainers, Acquirers
Concern 1	What are the hardware requirements to run the system for a certain number of users?
Concern 2	Which settings have to be configured when deploying the system for a specific domain?
Concern 3	Which parts of APOSDLE UPS can be distributed to physically different systems?
Language	UML 2.0

5.2.4.5 Scenario Viewpoint

The Scenario Viewpoint (SVP) is the place where all functional and non-functional requirements from Users and Acquirers are brought together, and could be modelled as use cases (Bittner and Spence, 2002). Use case are are always user-centered and do not contain any details about technical requirements. Based on a set of use cases, requirements and constraints can be derived, and used as input for the Logical Viewpoint (LVP). The Scenario Viewpoint of the APOSDLE UPS is presented in Chapter 4.

5.3 APOSDLE-UPS Views

The following sections present the implementation of the viewpoints defined in Section 5.2.4 except for the Scenario Viewpoint which has been implemented in Chapter 4.

5.3.1 Logical View

The Logical View describes the software structure. It is used to identify the major design packages and classes and addresses the functional requirements of the system. The Logical View provides a top-level view onto the whole APOSDLE system.

5.3.1.1 APOSDLE Tools

Figure 5.1 shows the different types of tools APOSDLE provides to users. Modelling Tools are used by experts to create formal models of the knowledge worker's environment and work processes. These models are later on transformed to be used by the APOSDLE UPS

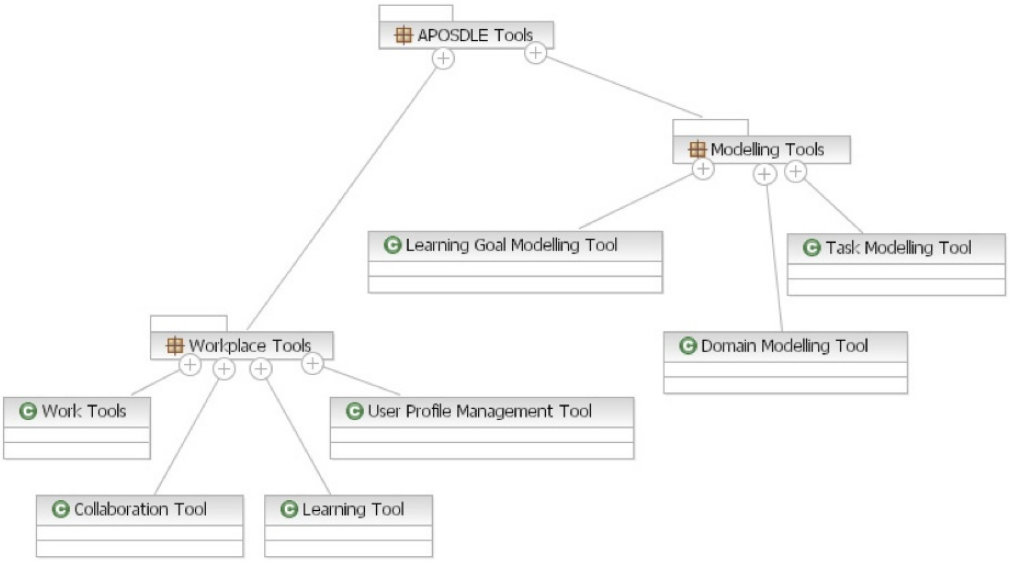

Figure 5.1: Logical View of all APOSDLE Tools

as part of its document space (see Section 2.2.1). Workplace Tools are used by knowledge workers to get support during their daily work.

Element catalogue

APOSDLE Tools: All tools provided by the APOSDLE System

Modelling Tools: The tools in the APOSDLE System used by experts for creating formal model of the knowledge worker's environment and work processes.

Learning Goal Modelling Tool: The tool used to model relations between Tasks and the Learning Goals associated with the Tasks.

Task Modelling Tool: The tool used to create models of the tasks of the knowledge worker.

Domain Modelling Tool: The tool used to create a model of the domain of the knowledge worker.

Workplace Tools: The tools used by the knowledge worker during their daily work.

Work Tools: The Work Tools comprises the Client Monitoring Daemon (detects the current user's task) and the APOSDLE Client.

Collaboration Tool: The Collaboration Tool allows users to chat and share information. It includes multiple collaboration tools (e.g., Skype).

Learning Tool: The Learning Tools is used to create and present learning events to users.

User Profile Management Tool: Used for presenting user-related information (name, address etc.) and usage-related information (tasks executed etc.) for a given user.

Relations

Modelling Tools and Workplace Tools are sub packages of APOSDLE Tools

Learning Goal Modelling Tool, Task Modelling Tool, Domain Modelling Tool are classes in the package Modelling Tools

Work Tools, Collaboration Tool, Learning Tool and the User Profile Management Tool are classes in the package Workplace Tools. Architecture background

Further details about the APOSDLE Tools are documented in APOSDLE Consortium (2007b) and APOSDLE Consortium (2008).

5.3.1.2 APOSDLE Platform

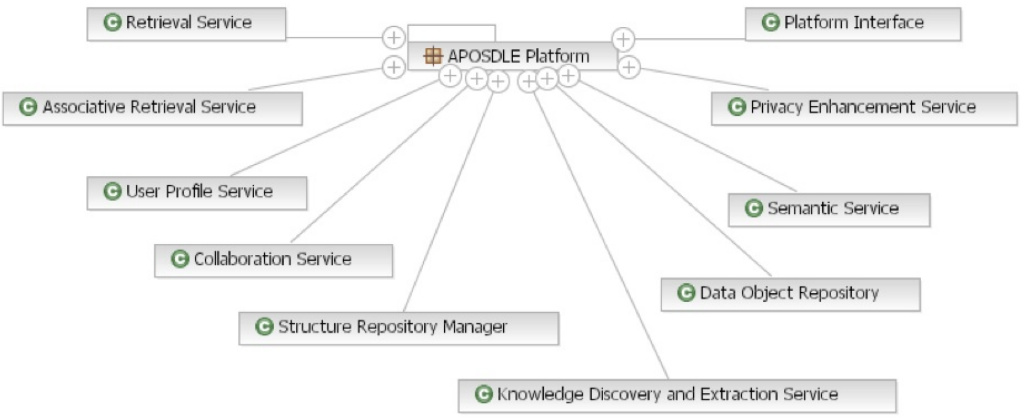

Figure 5.2: Logical view of the APOSDLE Platform

Figure 5.2 provides an overview of functionality provided by the APOSDLE Platform. The APOSDLE Platform fulfills two main tasks within the APOSDLE System: a) Provide foundation functionality to the whole APOSDLE System (i.e. the APOSDLE Tools and the APOSDLE Platform) and b) to provide a way of recommending resources for work-integrated learning, based on the context of the learner.

Element Catalogue

Associative Retrieval Service: Realises the search functionality for resources to be recommended to users

Retrieval Service: The general purpose retrieval service, queries the Associative Retrieval Service using context information from the APOSDLE UPS.

Knowledge Discovery and Extraction Service: Provides functionality to calculate the content-based similarity of resources. For more details see Scheir et al. (2008).

Semantic Service: Provides functionality to calculate the semantic similarity of elements in the APOSDLE models.

Data Object Repository: Realises the access to the backend system in a transparent way.

Platform Interface: Provides access to the APOSDLE Platform from outside (i.e. for the APOSDLE Tools).

Structure Repository Manager: Realises the functionality of managing models.

User Profile Service: Realises the functionality of managing information about the learner and the learner's context.

Collaboration Service: Realises the functionality needed for collaboration initialisation and management of collaboration tools.

Privacy Enhancement Service: Realises functionality to enhance the privacy of users of the APOSDLE System.

Further details about the APOSDLE Platform are documented in APOSDLE Consortium (2007b) and APOSDLE Consortium (2008).

5.3.1.3 APOSDLE Platform Interface

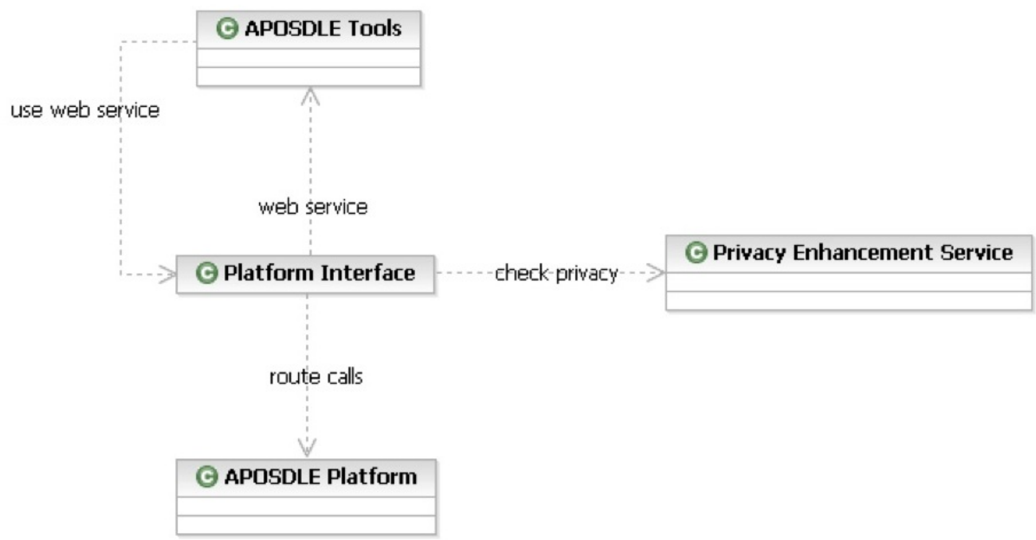

Figure 5.3: Logical view of Platform Interface

Figure 5.3 shows the main tasks of the Platform Interface: routing incoming calls form APOSDLE Tools to APOSDLE Platform components, and checking incoming and outgoing data according to privacy policies.

Element Catalogue

Platform Interface: The Platform Interface is part of the APOSDLE server which encapsulates the communication between APOSDLE Tools and other components inside the APOSDLE Platform. It handles all incoming and outgoing messages from and to the APOSDLE Platform.

APOSDLE Tools: The APOSDLE Tools provide an interface to the user of the APOSDLE System Modelling Tools: used by experts to create formal models of the knowledge worker's environment. Workplace Tools: used by the knowledge worker for work-integrated learning.

Privacy Enhancement Service: This service provides check privacy of all requests which come to Platform Interface. In dependence of responses of Privacy Enhancement Service, Platform Interface makes routing on certain component of APOSDLE Platform.

Further details about the APOSDLE Platform are documented in APOSDLE Consortium (2007b) and APOSDLE Consortium (2008).

5.3.1.4 APOSDLE UPS

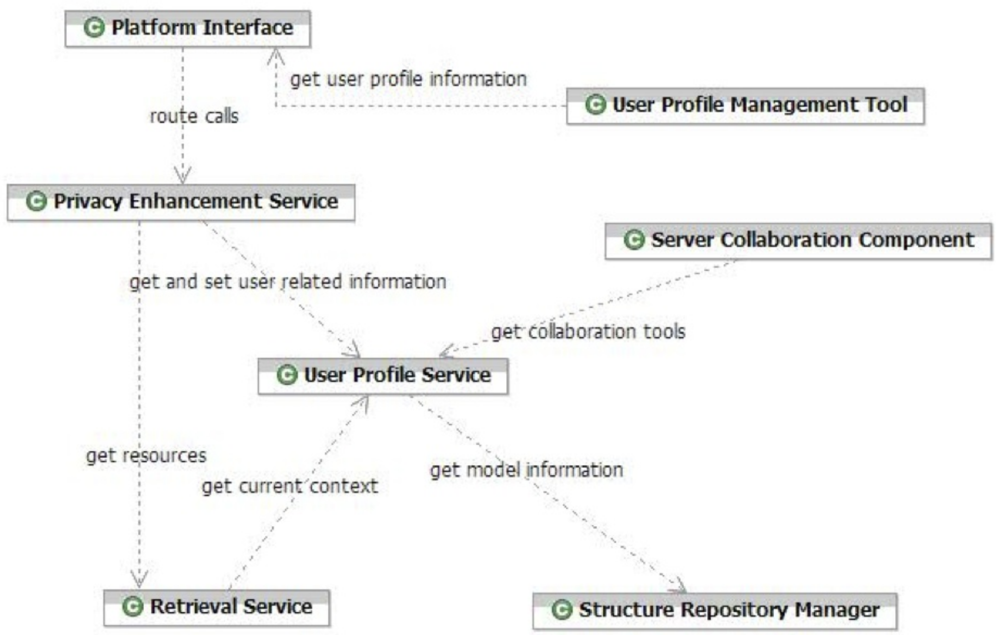

Figure 5.4: Logical view of the APOSDLE UPS with relations to other APOSDLE components

Figure 5.4 shows the APOSDLE UPS and how different information is exchanged between other components of the APOSDLE Platform.

Element Catalogue

User Profile Service (UPS): Basic service component for storing and maintaining user- related information. From this information pieces of information will be computed on demand, which serve best-possibly to identify the current users' needs.

Structure Repository Manager: The Structure Repository Manager provides the UPS with model information and mappings between model elements.

Retrieval Service: Requests a user's 'current context' from UPS. The current context consists of domain model elements, which have either been decided by the UPS to be highly relevant for the user's current work task and learning need, or which have actively been selected by the user.

Privacy Enhancement Service: Ensures privacy guidelines are met, when user-related information is stored and then later displayed to human users.

Server Collaboration Component: The APOSDLE UPS stores and also provides a list of favored collaboration tools for each user. The Server Collaboration Component requests this list from the APOSDLE UPS.

User Profile Management Tool: Tool for displaying user-related information to human users and allowing users maintaining data of their own. Note that there is no direct connection between the User Profile Management Tool and the UPS. Calls from the User Profile Management Tool for user-related data are routed through the Privacy Enhancement Service. The User Profile Management Tool covers the use case where users can investigate their user profile (see Section 4.3.5).

Platform Interface: Basic service interface, which the APOSDLE Platform exposes to the 'outside world'. Note that there is no direct connection between the Platform Interface and the UPS. Calls from the Platform Interface to the UPS are routed through the Privacy Enhancement Service. The Platform Interface has merely been added to the diagram for providing a broader overview.

Relations

UPS and Structure Repository Manager – get model information: The Structure Repository Manager stores (among others) information on how models are mapped onto each other. In particular the following mappings are requested from the Structure Repository Manager by the UPS: mapping between tasks and learning goals; mapping between learning goals and learning goal types; mapping between learning goals and domain model elements; prerequisite learning goals

Privacy Enhancement Service and UPS – get and set user related information: In principal each call from the Platform Interface to the UPS is routed through the Privacy Enhancement Service. The following user related information is exchanged between the UPS and the Privacy Enhancement Service: logged data which needs to be stored in the user profiles (availability, current task, invitation, selections made on the client side, relevance feedback etc.); recommended experts and potential collaboration partners; ranked list of a user's learning goals; Changes made to user-related data; privacy level, which a given user has chosen for herself; user profile data for presentation via the User Profile Management Tool ('business card', histories of tasks executed, collaborations participated in, Learning Goals acquired, etc.)

Retrieval Service and UPS – get current context: The current context is a list of domain model elements, which are sorted according to their assumed relevancy for the given user. The Retrieval Service requests this context from the UPS.

Server Collaboration Component and UPS – get collaboration tools: Each user can store her preferred collaboration tools within her own user profile. The Server Collaboration Component requests a list of collaboration tools from the UPS in order to recommend tools when a collaboration will be started.

5.3.1.5 APOSDLE UPS Management

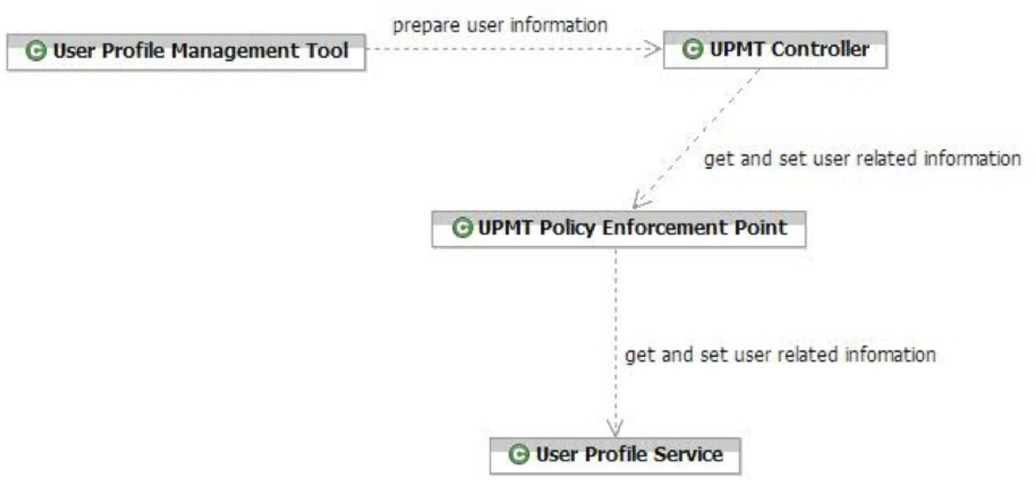

Figure 5.5: Logical view of the User Profile Management Tool with relations to the APOSDLE UPS

Figure 5.5 presents a logical view of the User Profile Management Service presenting its central components. The User Profile Management Tool (UPMT) is designed display and navigate through the content of the user profile. Additionally, it allows to modify certain data in the UPS. All data is retrieved from the User Profile Service and checked for privacy issues by the Privacy Enhancement Service. More details about the functionality of the UPMT has been presented by Garcia-Barrios et al. (2008). The Privacy Enhancement Service has been designed in close collaboration with the APOSDLE team at TUG. Implementation work was done by TUG.

Element catalogue

User Profile Management Tool (UPMT): A web application to display user information and giving an interface to edit certain data (e.g. setting privacy level, deleting tasks, learning goals and collaborations).

UPMTController: Service implementation to prepare and provide user information, which is to be shown in the user interface of the UPMT.

User Profile Service: Main source of data for the UPMT. All requests from and to the User Profile Service are routed through the Privacy Enhancement Service.

UPMT Policy Enforcement Point: This service is the main entry point into the APOSDLE system and checks all information retrieved from the User Profile Service according to privacy policies.

Relations

UPMT Controller and UPMT Policy Enforcement Point get and set user-related information: The UPMT Policy Enforcement Point ensures privacy issues are considered sufficiently. It therefore filters out all data conflicting with a user's privacy level. After filtering the method calls are forwarded to the User Profile Service. The User Profile Service either stores user-related data (set information) or produces results and returns them (get information).

Further details about the UPMT are documented in APOSDLE Consortium (2007b) and APOSDLE Consortium (2008).

5.3.1.6 APOSDLE Privacy Enhancement Service

Figure 5.6: Logical view of the APOSDLE UPS and how it interacts with the Privacy Enhancement Service

Element Catalogue

Platform Interface: Main entry point into the APOSDLE Platform. It exposes different interfaces outwards and routes calls to the Privacy Enhancement Service for further processing inwards and thus it serves as the main employer of the Privacy Enhancement Service.

Privacy Enhancement Service (PES): Main authority taking care of privacy issues arising by storing and retrieving individual-related information. All requests received by the Platform Interface will directly be sent to the Privacy Enhancement Service to identify and react on all privacy issues applying to this request. After the request was privacy-cleared it will be routed to internal APOSDLE components.

User Profile Service (UPS): PES uses the User Profile Service to retrieve information about users (e.g. selected privacy level chosen by an user) to evaluate privacy policies and thus decide how to proceed with an incoming request. PES uses the UPS services only as an information point without storing any information in the UPS.

Data Object Repository (DOR): Besides dealing with privacy issues the PES is also responsible for initiating the check for access rights of all data objects sent by the APOSDLE Platform to any tools outside. DOR provides the PES with a service expecting a list of Data Objects and returning only those where access was granted.

The Privacy Enhancement Service has been designed in close collaboration with the APOSDLE team at TUG. Most implementation work was done by TUG. Further details about the Privacy Enhancement Service can be found in APOSDLE Consortium (2007b) and APOSDLE Consortium (2008).

5.3.2 Process View

The Process Viewpoint mainly deals with flow of control and information between components. It provides additional details to the higher-level Logical Viewpoint.

5.3.2.1 Recommendation of Resources

Element Catalogue

Platform Interface: Interface to the APOSDLE Platform.

Privacy Enhancement Service: Service addressing privacy issues.

Retrieval Service: General purpose service for retrieval of resources for work-integrated learning.

User Profile Service: Provides functionality to store, manage and access user related information.

Associative Retrieval Service: Service for retrieval of knowledge artefacts under consideration of semantic and text-based associations among them.

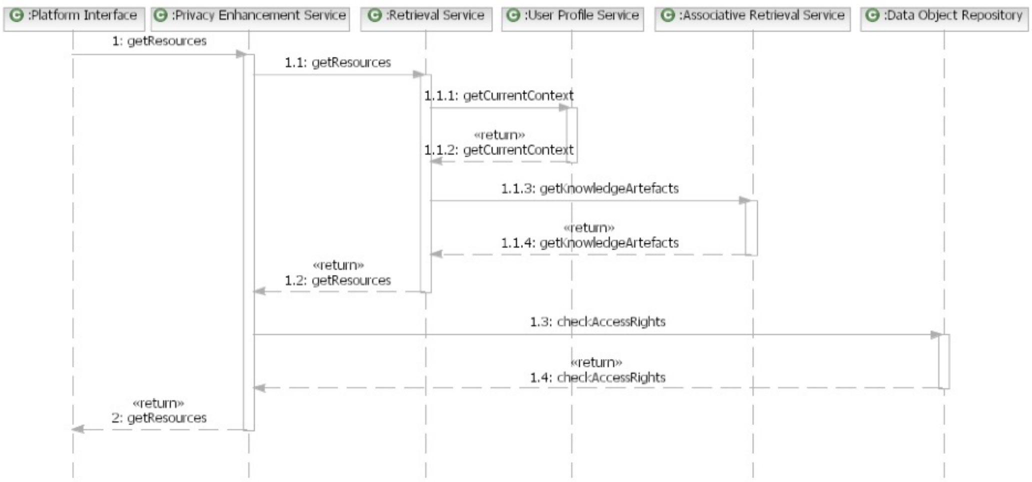

Figure 5.7: Recommendation of Resources based on the APOSDLE UPS.

Data Object Repository: transparent access layer to the backend systems.

The sequence diagram in Figure 5.7 describes the interaction of different components involved in the process of context-based retrieval of resources. The request for resources to the Platform Interface is routed through the Privacy Enhancement Service to the Retrieval Service. Depending on the need of the requesting APOSDLE tool, a query is issued to the Associative Retrieval Service. To formulate this query, the current context of the user is requested from the Use Profile Service. The Associative Retrieval Service provides search results to the query, which are returned to the PES. There, they are filtered according to the access rights of the current user and returned back to the Platform Interface.

5.3.2.2 Login Management

Figure 5.8 briefly describes how the User Profile Service is involved in the login and logout process of a user.

Element Catalogue

User Agent: An APOSDLE Tool which logs into the APOSDLE Platform

Platform Interface: Entry point for login and logout calls to the APOSDLE Platform. The Platform Interface routes login requests to previously configured components.

Privacy Enhancement Service (PES): PES establishes the privacy context for a user. Therefore, it requests some user details from the UPS.

User Profile Service (UPS): UPS authenticates the user through an external LDAP server and loads the user profile for the corresponding user.

Active Directory Service: A service interface which is connected to a external LDAP-based directory service for authenticating users.

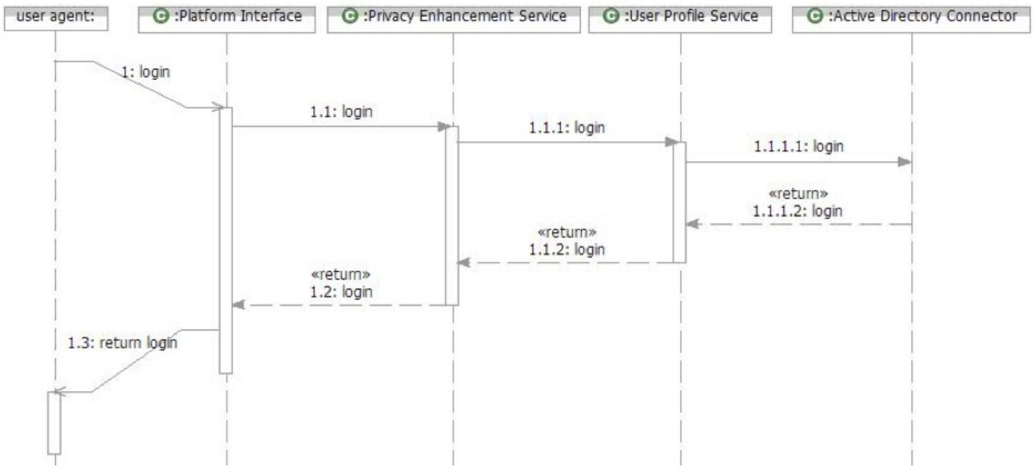

Figure 5.8: User profiling during login and logout of users.

All calls to the User Profile Service are routed through the Privacy Enhancement Service. The principal process of accessing the User Profile Service can therefore be inferred from this view packet.

5.3.3 Development View

APOSDLE is designed as client-server architecture, and has been implemented with different Java technologies and frameworks, e.g.:

- Spring Framework[4]
- Hibernate Framework[5]
- Apache Axis2[6]

Besides design patterns to solve specific problems, the principle of separation of concerns was the most important mantra throughout the whole development process. One piece of software (from package level to method level) is only responsible to do one specific thing. Among other criteria, adhering to this principle enables code to be maintainable and to be open for changes.

5.3.3.1 Technical Constraints

This section encompasses all technical constraints specified by the APOSDLE Consortium for the development of APOSDLE prototypes. To integrate a software component into the large APOSDLE software system following constraints must be met. Table 5.5 presents all core constraints which have to be met by the UPS.

The APOSDLE prototypes are developed as a client-server architecture whereas all core functionality is located on the server side. The client is designed as a thin client displaying

[4]http://www.springsource.org
[5]http://www.hibernate.org
[6]http://axis.apache.org/axis2/java/core/

Table 5.5: Development Requirements for UPS prototype

Type of Constraint	Description	Version
Programming Language	Java	1.6.10
Build Tool	Apache Maven	>= 2.0.8
Dependency Management of components	Spring Framework	2.5
Client-Server communication	Apache Axis Web Services	1.3
Deployment Format	Web application as WAR-file	-
Servlet Engine	Apache Tomcat	6.0.14
Deployment Platform	Windows	MS Win XP, MS Server 2003
Logging of system events	Apache log4j	1.2.15
Unit Test Framework	Junit	>= 3.8.1
Database Provider	MySQL	5.4.32

data and sending user actions (selection of tasks, opening of documents) to the server. The UPS prototype is therefore required to be developed as a server component providing its functionality through web services to APOSDLE clients.

The APOSDLE Platform is structured into software components each delivering a specific range of functionality. A component is defined by set of configuration files (build configuration, dependencies to third party libraries, etc.), a main source tree containing all Java code, a test tree for all unit tests, and additional directories containing resource files (e.g. model data files).

5.3.4 APOSDLE UPS Data Structure

Figure 5.9 gives a overview of the basic properties stored in the user profile.

Element Catalogue

User: The basic 'access point' to data stored within and inferred from user profiles.

UserBusinessCard: Business-card like representation of a user's user data.

OrganisationalUnit: Organisational unit to which the user belongs. Note that there is still discussion whether one particular user is allowed to belong to more than one organisational unit, or not.

ModelElement: Describes attributes, which most of the other elements of a user's profile must possess (i.e. description, label, URI, id).

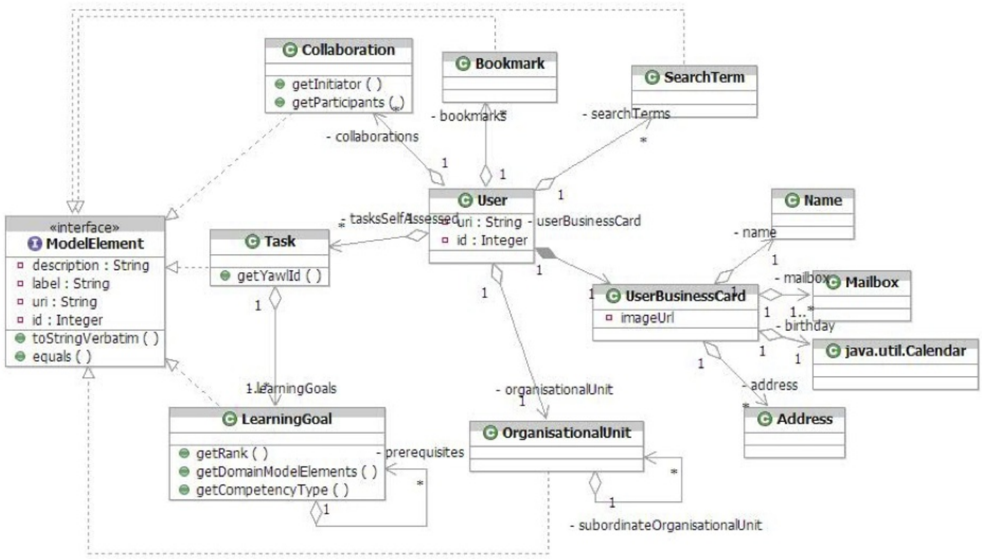

Figure 5.9: Data structure user for the APOSDLE representation of user profiles.

5.3.5 APOSDLE UPS Services

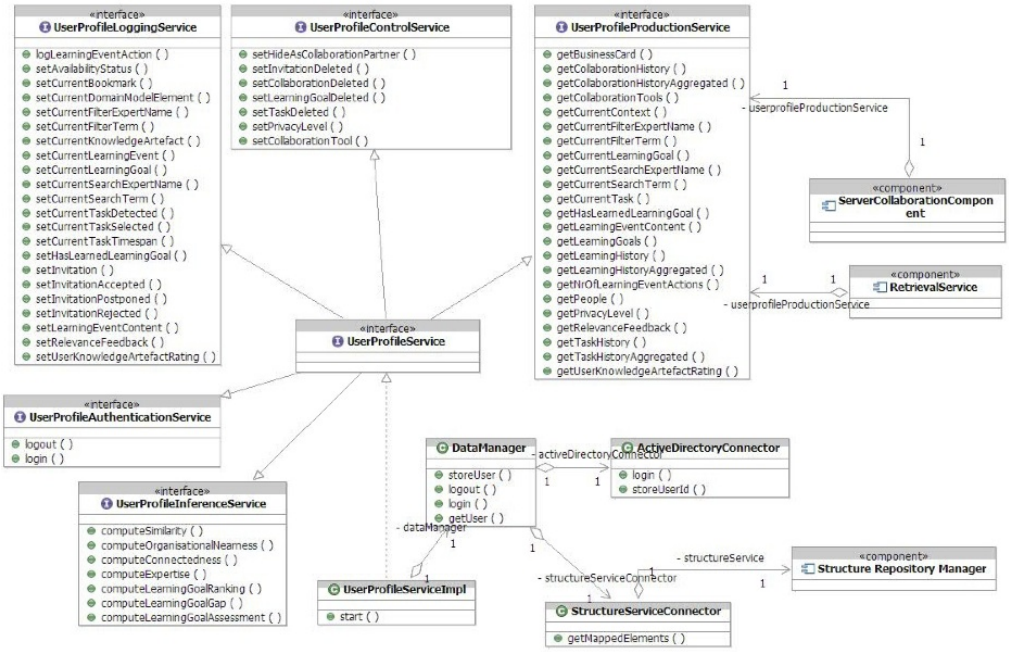

Figure 5.10: Functional decomposition of the User Profile Service into function-specific services

The User Profile Service (UPS) maintains user-related information, stores observations of usage data. It exposes its functionality via two sets of services: first there a set of services dedicated to special functionality of the UPS (i.e. authenticating users, logging data, controlling user-related data, producing e.g. recommendations and inferring further information). This set of services is shown in Figure 5.10. Second, there are application-specific services, which are dedicated towards special use cases (i.e. collaboration, user profile management, supporting learning and retrieving resources).

Application-specific services (see Figure 5.11) make use of the Decorator design pattern. Each call to them is delegated to one of the underlying function-specific services. Example: the CollaborationService uses the UserProfileLoggingService for storing information and the UserProfileProductionService for obtaining collaboration-related information. The Privacy Enhancement Service filters all user-related information and thus interacts closely with the services exposed by the UserProfileService. Therefore, there is a dependency relationship shown between the Privacy Enhancement Service and the interface ExposedService.

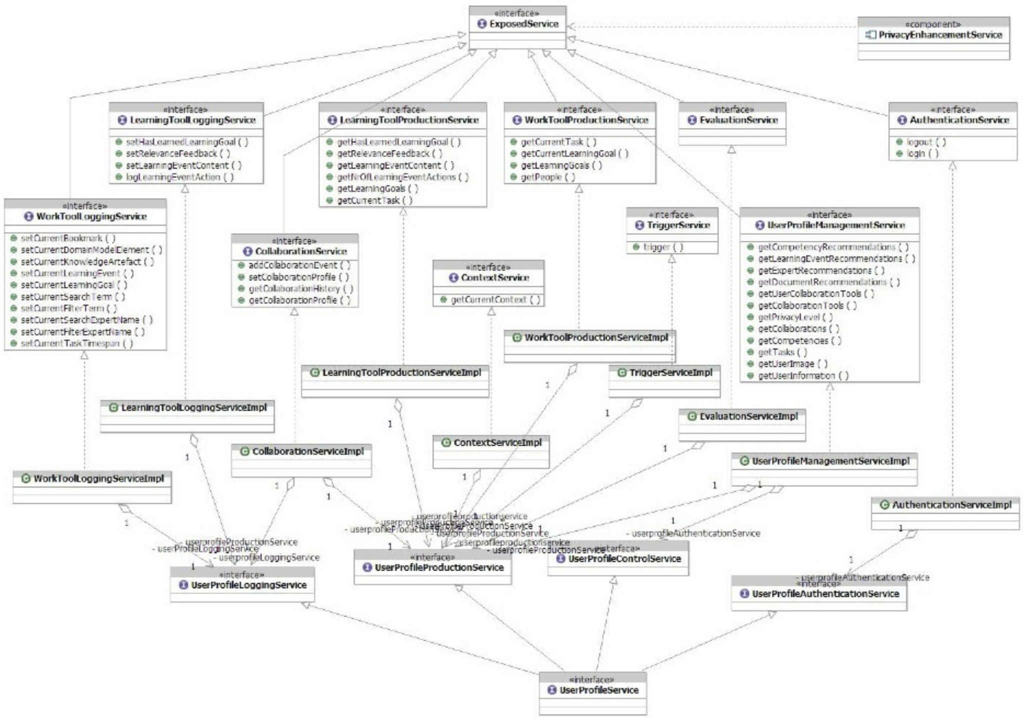

Figure 5.11: Interfaces of the User Profile Service used by other components of the APOSDLE Platform

5.3.6 Physical View

The Physical Viewpoint takes into account primarily the non-functional requirements of the system such as availability, performance and scalability. The software will execute on a network of computers, and various components will execute on different types of computers (servers, clients). Although several different configurations will be used at deployment time, the Physical Viewpoint gives a typical physical configuration.

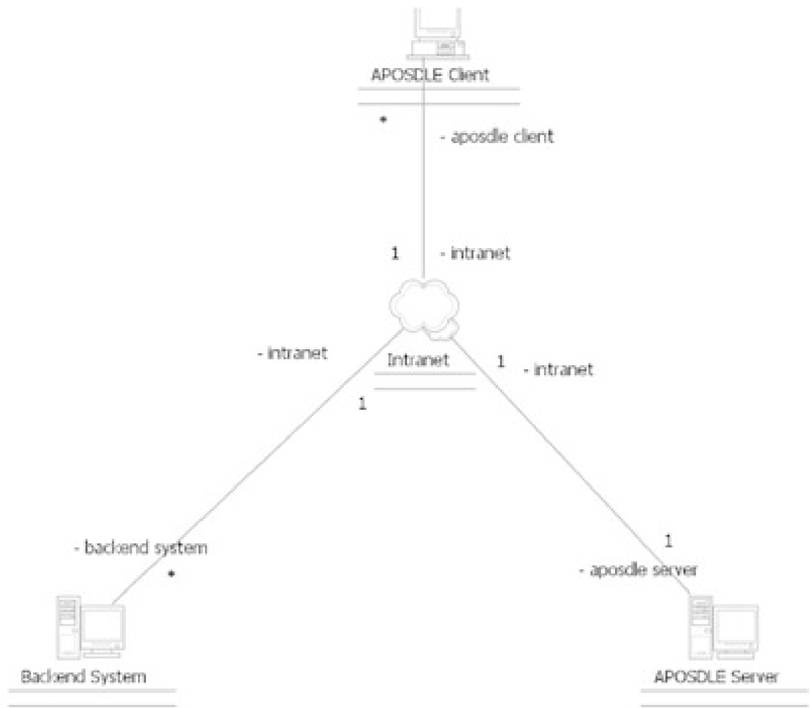

Figure 5.12: Communication of the APOSDLE System over the Intranet

Figure 5.12 shows the communication between APOSDLE Clients and the APOSDLE Platform.

Element Catalogue

APOSDLE Client: A computer of a knowledge worker on which the APOSDLE Workplace Tools are installed.

APOSDLE Server: A server machine on which the APOSDLE Platform is installed and running. The APOSDLE UPS is part of the APOSDLE Platform.

Backend Systems: A backend system used by the APOSDLE System and access via the Data Object Repository.

Intranet: The intranet of the company the APOSDLE System is connected to.

5.4 Application of the APOSDLE System

APOSDLE is designed as a system which can be configured to run in different WIL application domains. Te setup of APOSDLE for an application domain consists of the following steps:

- Creating enterprise models according to the Integrated Modelling Methodology developed by Ghidini et al. (2008)
- Installation of the APOSDLE Platform (on a company server) and the APOSDLE Client applications (on the users' computers)
- Import of enterprise models into the APOSDLE Platform
- Configuration of knowledge bases (e.g., network drives) to be connected to the APOSDLE Platform
- Annotation of documents: assigning topics from the domain model to parts of documents (snippets) to serve as a training set for later automatic annotation (Lindstaedt et al., 2008a).

APOSDLE instances were created for four real application domains: Innovation management at the company ISN, aircraft simulation in the company EADS, environmental consulting in the Chamber of Commerce and Industry (CCI) in Darmstadt, and library management in the FernUni Hagen (FUH). Although I was strongly involved in the exploitation and installation process (specifically for APOSDLE User Profile Services), the modelling and setup in the companies is not part of my master's thesis. Nonetheless, I will give a brief description of the four application cases to illustrate the four real-world settings in which the APOSDLE system including APOSDLE User Profile Services was actually up and running.

5.4.1 Innovation Management: ISN Domain

The company ISN[7] is a service and research company composed of a network of SMEs in Slovenia and Austria working in the field of innovation management. ISN is supplemented by more than 40 further partners from universities, competence centres and service companies acting as a pool of experts. Typical tasks in the innovation management domain are e.g., analysing trends in a certain branch, or identifying strategies of other customers in the branch. It is essential for the work of ISN consultants to have a good overview over past projects carried out by ISN and of resulting experiences. Therefore, the initial motivation for implementing APOSDLE at ISN is to create a common knowledge base which integrates resources from different repositories and backend systems. Another use case is to support novice workers by making them aware of the existing knowledge within the company and of people who could provide help with certain tasks or topics.

5.4.2 Aircraft Simulation: EADS Domain

The following description of the aircraft simulation domain at EADS IW[8] stems from (Kump, 2010) :

[7] Innovation Service Network, http://isn.at/index.php

[8] Innovation Works (IW) department of the European Aeronautic Defence and Space Company (EADS), Paris; http://www.eads.com

The application domain of EADS is about aircraft simulation. The aircraft simulation teams at EADS are designing and carrying out numerical simulations of electromagnetic problems, for example, simulations of electromagnetic attacks on aeronautical systems or subsystems for EADS-internal clients. The targeted groups of APOSDLE at EADS are engineers with a background in Physics or Mathematics and Computer Science who often work in geographically dispersed teams. The scope of the EADS domain model ranges from the first expression of the client's needs to the analysis and final presentation of the simulation results. EADS tasks are, for example, *Configure simulation data* or *Validate and test simulation*. Learning goals include an understanding of the *methode de resolution* (resolution method) and of the *contexte projet* (project context).

5.4.3 Industrial Property Rights Consulting: CCI Domain

The scope of the domain *Information and Consulting on Industrial Property Rights* at the Chamber of Commerce and Industry (CCI)[9] in Darmstadt is to inform clients about all legal and economic aspects of industrial property rights management in an introductory way and to enable them to do the next considerate and tactical steps. The main target group for information and consulting on industrial property rights are 'inventors' from small and medium enterprises.

The typical information and consulting process starts with an email or a phone call from the customer who then analyses the problem. In the next step and depending on the level of knowledge of the customer the consultant will explain and hand over standard information, send the client to a more specialised consulting institution or arrange a further consulting session. To be prepared for the individual consulting, the CCI expert will carry out literature research literature studies, researches in special databases and information portals as well as seek the advice of his CCI colleagues. The result is an individualised information package that will be explained to the client in a face-to-face or phone session or by email. Finally the consulting case is documented and archived.

Learning documents at CCI are mainly electronic and printed guidebooks on industrial property rights management, the documented consulting cases, specialised information portals and databases in the internet, CCI lists of institutions and specialists for advanced consulting, literature lists, lists of relevant events or seminars et cetera.

5.4.4 Library Management: FUH Domain

The FernUniversität Hagen (FUH)[10] offers supported distance learning for around 45000 students. The library of the FernUniversität Hagen decided to use APOSDLE in three of their areas, namely information, document delivery, and cataloguing. The area information includes the answering of customer requests (service requests) regarding the university library's range of services via e-mail, telephone, or in oral form. To answer these requests, besides knowledge that has to be conveyed to external persons, knowledge of internal workflows is necessary. In the area document delivery, the media (books, audiovisual media, etc.) are provided and made accessible to the users (borrowing, returning). Here, there are various interfaces to the area information. In the area cataloguing, processes and courses of business shall be documented, and instructions for work processes including example collections and course materials shall be collected.

[9] http://www.darmstadt.ihk.de/
[10] http://www.fernuni-hagen.de/

The aim why FernUniversität Hagen was interested in APOSDLE is that they wanted to represent distributed documents in one system, so that all workers could access them from a single source. All employees should be at the same knowledge state, and APOSDLE should serve as a research database in the sense of a system that is learning and saving time for modelling and annotation. The challenge in the library of the distance university was that everyone should be able to work in all areas. It was difficult to find knowledgeable persons for external requests. Special cases should be documented. While the knowledge for external requests was well-documented, the documentation of internal knowledge was lacking. The aim of the distance university was to see whether these bottlenecks could be overcome with APOSDLE.

5.5 Evaluations of APOSDLE: My Role

APOSDLE and its UPS has been evaluated in various settings. Even though the user-centered evaluation is not explicitly part of my master's thesis, I was actively involved in most evaluation activities. The different evaluations and my role in the process will be listed in the following:

- Simulation study of KIE approach: A simulation study was carried out to analyse and evaluate different algorithms to user model maintenance based on KIE. The main part of the simulation study has been carried out in collaboration with two master students, one in the field of Psychology (Pabst, 2011), and one in the field of Software Development (Resanovic, 2010) and is described in their master's theses.
- Evaluation of Learning Goal Ranking: The learning goal ranking based on the APOSDLE User Profile was investigated in a lab study by a master student in the field of Psychology (Gerdenitsch, 2009), see also (Ley et al., 2010b). Each participant in the study had to solve three tasks with the help of learning goals recommended by APOSDLE (three trials). For this study, I implemented three types of ranking algorithms for learning goals, and a sophisticated random-mechanism that decided which of the algorithms should take effect in a specific trial. This mechanism ensured a double-blind study setting (the experimenter did not know which algorithm was used when) while at the same time guaranteeing that all three algorithms were applied in the three trials. Moreover, it had to be ensured that all possible sequences occurred approximately equally frequently throughout the experiment with 24 participants.
- Evaluation of KIE in field: The validity of the information in the user model was investigated in two field studies, one in the FUH domain[11], and one in the ISN domain (Pabst, 2011), both by students of Psychology. For the evaluation, I aggregated the log data in the APOSDLE UPS (value of a user for each topic in the domain model) according to the students' specifications in a format that could be read by the statistical software package SPSS.
- Overall APOSDLE Evaluation: At the end of the APOSDLE Project, an extensive summative evaluation was carried out in all application domains (APOSDLE Consortium, 2010a). For this evaluation, I provided all log data (different than the data in the evaluation of KIE above) in a format that could be read by the statistical software package SPSS.

[11] http://eleed.campussource.de/archive/6/2625/

Chapter 6

Discussion and Self-Evaluation of Goals

This section presents a self-evaluation with respect to the goals stated in Section 1.2, the achievements outlined in Chapter 4, and Chapter 5.

Requirements were stated at two points in my master's thesis, in Section 6.1 and Section 6.2. In the following, I will revisit all these requirements and evaluate whether I was able to achieve them.

6.1 Requirements from Theoretical Considerations

Three high-level requirements were formulated in Section 3.2 based on the three main challenges for WIL:

Theory-Requirement 1 (TR1): The WIL System should support the user in an adaptive manner during his or her everyday work

> **Discussion:** My work, the APOSDLE UPS provides an important contribution to the system's adaptivity, as the user model is the core of an adaptive system. Thus, the first part of this requirement is clearly fulfilled by the APOSDLE system: it provides various adaptive functionality. Whether the system actually supports its users cannot be answered with my work; indicators in this direction have been found in the summative evaluation of APOSDLE, see (APOSDLE Consortium, 2010a).

Theory-Requirement 2 (TR2): The WIL System should re-use resources within the organisational repository as learning content

> **Discussion:** This requirement is not covered by my master's thesis but it is clearly fulfilled within the overall APOSDLE solution.

Theory-Requirement 3 (TR3): The WIL System should be embedded in and interact with a user's normal work environment

> **Discussion:** The service-oriented approach allows easy integration of WIL services into an existing computational environment. Clearly, all APOSDLE UPS adhere to this paradigm. This has the additional benefit that the APOSDLE UPS could also be integrated in other environments.

TR1 can be further refined based on the theoretical considerations from Section Section 3.1.

Theory-Requirement 1.1 (TR1.1): The WIL System should adapt learning to the task at hand and the background knowledge of its users

> **Discussion:** I have designed the APOSDLE UPS in a way that it fulfills exactly this requirement: The support is adapted to a user's task and the user's learning need (Section 4.8.1).

Theory-Requirement 1.2 (TR1.2): The user model of a WIL system should be maintained in an unobtrusive manner.

> **Discussion:** I have designed and implemented the APOSDLE UPS following the KIE approach to user model maintenance Section 3.3.1; this approach ensures that the user model can be maintained without explicit input from the user.

Theory-Requirement 1.3 (TR1.3): A WIL System should take care of privacy regulations of the environment being placed in.

> **Discussion:** This is clearly an important requirement, that has also been mentioned earlier in this work. Even though the issue of privacy is strongly related to my work, this has been out of scope for me but has been covered extensively by other partners in the APOSDLE project. For a comprehensive discussion of the privacy issue in APOSDLE see APOSDLE Consortium (2009), or Zinnen et al. (2008).

TR2 includes both document resources and 'human resources' and can be further refined based on the theoretical considerations from Section 3.1 as follows

Theory-Requirement 2.1 (TR2.1): The WIL System should be able to identify and recommend relevant documents within the organisation.

> **Discussion:** My work is related to this requirement as the Learning Need Service in APOSDLE triggers a search query and starts the Associative Retrieval (see Scheir et al., 2008). APOSDLE recommends documents given a learning need; whether they are relevant, however, cannot be answered with my thesis.

Theory-Requirement 2.2 (TR2.2): The WIL System should be able to identify and recommend knowledgeable people within the organisation.

> **Discussion:** Functionality that fulfills this requirement has been implemented in the People Recommender Service. The report of APOSDLE's summative evaluation (APOSDLE Consortium, 2010a) indicates that this functionality is useful, however it was found that expert recommendation requires that the system is used extensively; this was not the case in the APOSDLE evaluation. Clearly, research into KIE is still at an early stage and a lot of work needs to be done in this are to fine-tune the algorithms.

Theory-Requirement TR3 can be refined as follows:

Theory-Requirement 3.1 (TR3.1): The WIL System shall be able to interact with standard software that is used at the workplace

> **Discussion:** Due to the service-oriented approach (See TR3), the APOSDLE client is in principle able to interact with standard software. However, the APOSDLE UPS do not interact with standard software, as they are located in the server.

6.2 Requirements from Use Cases

The following requirements have been derived from use cases in Section 4.3. Requirements derived from Use Case 1 (UC1):

Use-Case Requirement 1.1 (UCR1.1): A task is an information for the WIL System to determine a user's current working context.

> **Discussion:** The APOSDLE UPS has been designed to use a user's task executions to infer the current working context (see also TR1.1). I want to point out that the APOSDLE UPS depends on the accuracy the APOSDLE Tools can detect a user's current task, and thus may vary in different work environments.

Use-Case Requirement 1.2 (UCR1.2): The WIL System shall be able to receive a task from two different sources, an automatic detection mechanism or a manual selection by a user.

> **Discussion:** The KIE approach to user modeling allows to specify events which can be mapped to different sources. With this approach, the APOSDLE UPS is adaptable to the requirements of the work environment it is used in.

Requirements derived from Use Case 2 (UC2):

Use-Case Requirement 2.1 (UCR2.1): The WIL System shall provide a list of learning goals for a users' current learning context.

> **Discussion:** The APOSDLE UPS can infer a learning context from a user's task executions and his or her work on topics contained in the domain model. Based on the inferred learning context, a ranked list of learning goals is calculated. This list is available from the Learning Need Service (see Section 4.8).

Use-Case Requirement 2.2 (UCR2.2): The WIL System shall save a users' selected learning goal as the current learning context.

> **Discussion:** Learning contexts are persisted in a database connected to the APOSDLE UPS. Thus, it is possible to retrieve information about a particular learning context occurred in the past. Although this requirement is fulfilled, more research is needed to tackle privacy issues like data retention.

The following requirements can be derived from UC3:

Use-Case Requirement 3.1 (UCR3.1): The WIL System should deliver a ranked list of knowledgeable users for the current context of a user.

> **Discussion:** This requirement has been fulfilled by implementing the People Recommender Service. As stated in (TR2.2), more research is needed into identifying knowledgable people by e.g., utilising more sources of information.

Use-Case Requirement 3.2 (UCR3.2): The calculation of the ranking of knowledgeable people shall be triggered by setting a learning goal.

> **Discussion:** The core of the APOSDLE UPS has been implemented to allow both: automatic calculation of knowledgeable people (as soon as the learning context is available), and manual calculation by calling a special function of the People Recommender Service. The current approach will have to be adapted for

large numbers of users because the time needed for calculating knowledgeable people may not be acceptable for users.

Use-Case Requirement 3.3 (UCR3.3): The WIL System shall maintain up-to-date information about users' knowledge about all available learning goals.

Discussion: The APOSDLE UPS constantly records user activities sent by APOSDLE Tools. User activities are translated into events and stored into a database at certain times. The APOSDLE UPS updates knowledge levels on demand only. I.e. knowledge levels are not updated automatically. Therefore, I classify this requirement as partly fulfilled.

Use-Case Requirement 3.4 (UCR3.4): The WIL System shall store the availability of users.

Discussion: In a first approach, availability information (whether a user is logged on, inactive, or offline) was maintained by the APOSDLE UPS. During the development of the third prototype of APOSDLE, this functionality was moved to the Cooperation Service which offers different communication channels.

Use-Case Requirement 3.5 (UCR3.5): The calculation of the ranked list of knowledgeable users shall include previous activities stored in the user profile, the current working context (task and selected learning goals), and the availability of users.

Discussion: The APOSDLE UPS stores a history of user activities for e.g., calculating ranked lists of knowledgeable users. The first approach for calculating knowledgeable people utilised the whole history available. An evaluation of the APOSDLE system showed several drawbacks of this first implementation and lead to an extension of the APOSDLE UPS (Resanovic, 2010).

Use-Case Requirement 3.6 (UCR3.6): The calculation of knowledgeable users should be flexible to take into account factors like organisational or social distance.

Discussion: Due to a lot of efforts to integrate the APOSDLE UPS with other components in APOSDLE, I did not address this requirement in my master's thesis.

The following requirement can be derived from UC4:

Use-Case Requirement 4 (UCR4): The current working and learning context is made available to other components of the WIL System to find relevant resources based on this context.

Discussion: The Usage Data History offers service functions the request the current and past working and learning contexts. Each context stored together with a time stamp in the database of the APOSDLE UPS.

The following requirement can be derived from UC5:

Use-Case Requirement 5 (UCR5): The UPS shall be able to assemble stored usage data and provide it to WIL Clients.

Discussion: This requirement is covered by the functionality of the Usage Data History Service (see Section 4.8).

The following requirement can be derived from UC6:

Use-Case Requirement 6 (UCR6): The WIL System should provide a way to modify data contained in the user model.

Discussion: This requirement can be addressed by the APOSDLE UPS only to some extend, as there is a user interface necessary allowing users to investigate and modify their user model. From APOSDLE UPS perspective, the *Usage Data Control Service* is in place which allows users to modify KIEs. The APOSDLE Client provides a visual user model tool called 'MyExperiences' (Kump et al., 2010) which connects to this service in order to change knowledge levels. The following requirement can be derived from UC7:

Use-Case Requirement 7.1 (UCR7.1): The user model of the WIL System shall be able to store usage data related to learning activities.

Discussion: APOSDLE UPS implements the *Resource Logging Service* to store incoming KIEs (e.g., a user asked for learning hints).

Use-Case Requirement 7.2 (UCR7.2): The user model of the WIL System shall be able to link learning activities to the users' current learning context.

Discussion: In addition to storing KIEs, the *Resource Logging Service* links KIEs with the current learning context of a user.

Chapter 7

Conclusion and Future Work

This master's thesis presented an approach which provides an architecture and a prototypical Implementation of a user modeling system and user-adaptive services for work-integrated learning. It employs existing techniques of adaptive (educational) systems to maintain user data and usage data about users to provide them as a basis for adapting different functions of an adaptive application. A number of conclusions can be drawn from my work. The KIE approach may have some limitations that are related to interface preferences, usability issues, or organisational constraints: for example, the tasks a user performs may be driven by organizational constraints or simply by task or job assignments and they may therefore only draw a partial picture of the knowledge and skills a user has available. In addition to the existing KIE, it may also be useful also plan to incorporate negative KIE, such as unsuccessful task executions. In doing so, instead of inferring the minimum competency state, i.e., competencies a worker has available at the minimum, the 'real' competency state of a worker could be approximated. Clearly, the usage of KIE brings up concerns with respect to user privacy. Logging user activities carries the risk that the data could potentially be abused for hidden productivity measurements (Hartman, 2001). Within APOSDLE, a dedicated Privacy Enhancement Service handles a variety of privacy measures such as filtering of service outputs or controlling access to certain usage data APOSDLE Consortium (2009), García-Barrios et al. (2009), García-Barrios (2009). Nonetheless, more in-depth research will be required to address remaining privacy issues. In order to overcome this problem, Kobsa (2007) suggested using client-side personalization. However, for the APOSDLE People Recommender Service, client-side personalization is not feasible as the underlying algorithms rely on the analysis of data from the whole user population. Besides technical measures to enhance user privacy, the following implications were derived by Kobsa (2007) to allay fears of users with regard to privacy: System developers must clearly communicate the benefits of their services; if users perceive value in the personalization, they are considerably more likely to use these systems and to supply the required personal information. Moreover, users should be given ample control over their data.

The work described within my master's thesis built the basis for two other master thesis projects in the Knowledge Management Institute (Software Development): One of them was dealing with adaptive reflection support (Radl, 2009), and in the other thesis, the KIE concept was refined and a simulation study was carried out (Resanovic, 2010). Moreover, two master theses in the field of Psychology were dealing with the evaluation of the validity of the WIL User Model based on KIE (Pabst, 2011), and the evaluation of the learning goal recommendation (Gerdenitsch, 2009), see also (Ley et al., 2010b).

My work suggests several avenues for follow-up. On one side, research must focus on the suitability of different KIE for diagnosing knowledge levels, which may also depend on the organizational setting. Moreover, as mentioned above, there may be a variety of additional KIE such as collaboration events and document creation. Another way to further develop the People Recommender Service in APOSDLE would be to exploit organizational structures in order to provide multidimensional rankings of knowledgeable colleagues Yimam-Seid and Kobsa (2003).

Bibliography

Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749, June 2005. ISSN 1041-4347. URL http://www.computer.org/portal/web/csdl/doi/10.1109/TKDE.2005.99. 23

Mario Aehnelt, Mirko Ebert, Guenter Beham, Stefanie N. Lindstaedt, and Alexander Paschen. A socio-technical approach towards supporting intra-organizational collaboration. *Times of Convergence*, 2008. URL http://www.springerlink.com/index/x334r771pr711048.pdf. 4, 33

APOSDLE Consortium. Deliverable D 6.3, Use Cases & Application Requirements 2, 2007a. 34

APOSDLE Consortium. Software Architecture for 2nd APOSDLE Prototypes, 2007b. 4, 51, 60, 61, 62, 65, 66

APOSDLE Consortium. Software Architecture for 3rd APOSDLE Prototypes, 2008. 4, 51, 54, 60, 61, 62, 65, 66

APOSDLE Consortium. Privacy in APOSDLE, 2009. 78, 83

APOSDLE Consortium. Summative Evaluation Report, 2010a. 76, 77, 78

APOSDLE Consortium. Conceptual Architecture including Component Evaluation, Deliverable D2.12, 2010b. 4, 23, 28, 33

Günter Beham, Barbara Kump, Daniel Resanovic, and Stefanie Lindstaedt. Non-invasive User Modelling for Recommending Knowledgeable Persons in Work-integrated Learning Systems. In Belgium Duval Erik K U Leuven and Switzerland Gillet Denis EPFL, editors, *Contextaware Recommendation for Learning*, 2009. URL http://www.cs.kuleuven.be/~{}katrien/arv/papers/beham.pdf. 4, 33

Günter Beham, Fleur Jeanquartier, and Stefanie Lindstaedt. iAPOSDLE – An Approach to Mobile Work-Integrated. *Scenario*, pages 608–613, 2010a. 4

Günter Beham, Barbara Kump, Tobias Ley, and Stefanie N. Lindstaedt. Recommending Knowledgeable People in a Work-Integrated Learning System. In *Knowledge Management*, volume 00. Procedia Computer Science, Elsevier, 2010b. 4, 33

David Benyon and Dianne Murray. Adaptive systems; from intelligent tutoring to autonomous agents. *Knowledge-Based Systems*, 6(4):197–219, 1993. 2, 11

Stephen Billett. Authenticity and a culture of practice. *Australian and New Zealand Journal of Vocational Education Research*, 2(1):1–29, 1993. 24

Stephen Billett. *Learning through work: Workplace affordances and individual engagement*, volume 13. 2001. 1

Kurt Bittner and Ian Spence. *Use Case Modelling*. Addison Wesley, 2002. ISBN 0201709139. URL http://www.amazon.co.uk/dp/0201709139. 58

Craig Boyle and Antonio O Encarnacion. Metadoc: An adaptive hypertext reading system. *User Modelling and UserAdapted Interaction*, 4(1):1–19, 1994. ISSN 09241868. URL http://www.springerlink.com/content/u372x25x59814674/?p=5136b088eb3741658fe0df05221153b7&pi=0. 14, 47

Paul De Bra, Natalia Stash, and David Smits. Creating Adaptive Applications with AHA! Tutorial for AHA! version 3.0, 2004. URL http://aha.win.tue.nl/. 19

Peter Brusilovsky. Methods and techniques of adaptive hypermedia - (Special issue on adaptive hypertext and hypermedia). *User Modeling and User Adapted Interaction*, 6 (2-3):87–129, 1996. 12, 15

Peter Brusilovsky. Adaptive Hypermedia: From Intelligent Tutoring Systems to Web-Based Education. *Lecture Notes in Computer Science*, 1839:1–7, 2000. URL http://www.springerlink.com/index/DNT2272VMW6WJFAR.pdf. 5

Peter Brusilovsky. KnowledgeTree: a distributed architecture for adaptive e-learning. In *International World Wide Web Conference*, pages 104–113, 2004. URL http://portal.acm.org/citation.cfm?id=1013386. 24

Peter Brusilovsky and Eva Millán. *User Models for Adaptive Hypermedia and Adaptive Educational Systems*, volume 4321, pages 3–53. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007. ISBN 978-3-540-72078-2. URL citeulike-article-id:1668793http://dx.doi.org/10.1007/978-3-540-72079-9_1. 2, 9, 10

Peter Brusilovsky and Leonid Pesin. ISIS-Tutor: An Intelligent Learning Environment for CDS/ISIS Users. In *Online Proceedings of CLCE'94*, pages ISIS–Tutor: An Intelligent Learning Environment fo, 1994. URL http://cs.joensuu.fi/~{}mtuki/www_clce.270296/Brusilov.html. 5, 15, 16

Peter Brusilovsky and Leonid Pesin. Adaptive navigation support in educational hypermedia: An evaluation of the ISIS-Tutor. *Journal of Computing and Information Technology*, 6(1):27–38, 1998. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.12.7482. 17

Peter Brusilovsky and C Peylo. Adaptive and intellingent Web-based educational systems. *International Journal of Artificial Intelligence in Education*, 13(2-4):159–172, 2003. 7

Peter Brusilovsky, Elmar Schwarz, and Gerhard Weber. ELM-ART: An intelligent tutoring system on world wide web - Intelligent Tutoring Systems - Lecture Notes in Computer Science. In C Frasson, G Gauthier, and A Lesgold, editors, *Proceedings of the Third International Conference on Intelligent Tutoring Systems, ITS-96*, volume 1086, pages 261–269–269. Springer Berlin / Heidelberg, 1996. URL http://www.springerlink.com/content/fnj2687uxrj21640/. 17

Susan Bull and Judy Kay. Student Models that Invite the Learner In: The SMILI:() Open Learner Modelling Framework. *International Journal of Artificial Intelligence in Education*, 17(2):89–120, 2007. URL http://iospress.metapress.com/content/d136315752666874/. 24

Andrea Bunt, Giuseppe Carenini, and Cristina Conati. *Adaptive Content Presentation for the Web*, volume 4321, chapter 13, pages 409–432. Springer, 2007. URL http://www.springerlink.com/index/h1602222260221x4.pdf. 12, 13

Paul Clements, Felix Bachmann, Len Bass, David Garlan, James Ivers, Reed Little, Robert Nord, and Judith Stafford. *Documenting Software Architectures: Views and Beyond*, volume Vi of *The SEI Series in Software Engineering*. Addison-Wesley Professional, 2002. ISBN 0201703726. URL http://www.amazon.com/Documenting-Software-Architectures-Views-Beyond/dp/0201703726. 52

Owen Conlan. State of the Art: Adaptive Hypermedia. *State of Art Surveys*, 2(May):47–57, 2003. 7

Paul De Bra and Jan-Peter Ruiter. AHA! Adaptive Hypermedia for All. In W A Lawrence-Fowler and J Hasebrook, editors, *Proceedings of WebNet 2001 - World Conference on the WWW and Internet, Orlando, Florida*, pages 262–268. AACE, Chesapeake, VA, 2001. 19

Paul De Bra, Geert-Jan Houben, and Hongjing Wu. AHAM: A Dexter-based Reference Model for Adaptive Hypermedia. In *Proceedings of the tenth ACM Conference on Hypertext and Hypermedia*, pages 147–156. 1999. 7

Paul De Bra, Ad Aerts, David Smits, and Natalia Stash. AHA! Version 2.0: More Adaptation Flexibility for Authors. In M Driscoll and T C Reeves, editors, *Proceedings of World Conference on E-Learning, E-Learn 2002*, pages 240–246. AACE, 2002. 5, 13

Michael Eraut. Informal learning in the Workplace. *Studies in Continuing Education*, 26 (2):243–247, 2004. 1, 12

Michael Eraut and Wendy Hirsh. The Significance of Workplace Learning for Individuals, Groups and Organisations, 2007. URL http://www.skope.ox.ac.uk/publications/significance-workplace-learning-individuals-groups-and-organisations. 1

Josef Fink, Alfred Kobsa, and Andreas Nill. Adaptable and adaptive information provision for all users, including disabled and elderly people. *New Review Of Hypermedia And Multimedia*, 4(1):163–188, 1998. URL http://www.informaworld.com/10.1080/13614569808914700. 14

Gerhard Fischer. Lifelong learning—more than training. *Journal of Interactive Learning Research*, 11(3):265, 2000. URL http://portal.acm.org/citation.cfm?id=371937. 1

Gerhard Fischer. User Modelling in Human-Computer Interaction. *User Modeling and User-Adapted Interaction*, 11:65–86, 2001. 3

Mark Fox and Michael Gruninger. Enterprise Modeling. *AI Magazine*, 19(3):109–121, 1998. ISSN 07384602. URL http://www.aaai.org/ojs/index.php/aimagazine/article/viewArticle/1399. 26

Victor Manuel García-Barrios. User-centric Privacy Framework: Integrating Legal, Technological and Human Aspects into User-Adapting Systems. In *International Conference on Computational Science and Engineering CSE*, pages 176–181. IEEE, 2009. ISBN 9781424453344. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5283435. 83

Victor Manuel Garcia-Barrios, Günter Beham, and Barbara Kump. Scrutinising Competencies: Retraceable Clouds of Learning Goals in the APOSDLE System. In *Proceedings of 6th International Conference on Community Based Environments OpenACS and LRN Conference 2008 February 2008 Guatemala*, pages 34–45, 2008. 64

Victor Manuel García-Barrios, Ariane Hemmelmayr, and Helmut Leitner. Personalized Systems Need Adaptable Privacy Statements! How to Make Privacy-related Legal Aspects Usable and Retraceable. *2009 Second International Conference on Advances in HumanOriented and Personalized Mechanisms Technologies and Services*, pages 91–96, 2009. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5291012. 83

Cornelia Gerdenitsch. *User-centred Evaluation of the Adaptation Model in a Work-integrated Learning System: The Case of APOSDLE. Unpublished Master's Thesis, Institute for Psychology, University of Graz.* 2009. 76, 83

Chiara Ghidini, Marco Rospocher, Luciano Serafini, Andreas Faatz, Barbara Kump, and Viktoria Pammer. Collaborative enterprise integrated modelling. In *Knowledge Management*, 2008. 74

Chiara Ghidini, Barbara Kump, Stefanie Lindstaedt, Nahid Mahbub, Viktoria Pammer, Marco Rospocher, and Luciano Serafini. MoKi : The Enterprise Modelling Wiki Research Background : Enterprise Modelling. *The Semantic Web Research and Applications*, 5554: 831–835, 2009. URL http://www.springerlink.com/content/m535253m2wq666p5. 26

Jeremy Goecks and Jude Shavlik. Learning users' interests by unobtrusively observing their normal behavior. In H Lieberman, editor, *IUI 2000: International Conference on Intelligent User Interfaces*, pages 129–132, New York, 2000. ACM. 27

Frank Halasz and Mayer Schwartz. The Dexter Hypertext Reference Model. *Communications of the ACM*, 37(2):30–39, 1994. URL citeseer.ist.psu.edu/halasz94dexter.html. 7

Lynda Hardman, Dick C A Bulterman, and Guido Van Rossum. The Amsterdam hypermedia model: adding time and context to the Dexter model. *Communications of the ACM*, 37(2):50–62, 1994. ISSN 00010782. URL http://doi.acm.org/10.1145/175235.175239. 7

Laura P Hartman. Technology and Ethics: Privacy in the Workplace. *Business and Society Review*, 106(1):1–27, 2001. ISSN 00453609. URL http://doi.wiley.com/10.1111/0045-3609.00099. 83

Nicola Henze and Wolfgang Nejdl. A Logical Characterization of Adaptive Educational Hypermedia. *New Review on Hypertext and Hypermedia*, 10(1):77–113, 2004. ISSN 13614568. URL http://www.kbs.uni-hannover.de/Arbeiten/Publikationen/2004/logical_characterization_henze_nejdl.pdf. 7, 8

Christine Hofmeister, Robert Nord, and Dilip Soni. *Applied software architecture*. Addison Wesley, 2000. URL http://portal.acm.org/citation.cfm?id=322640. 52

Kai Michael Höver and Nils Faltin. Authoring of an adaptive group-oriented business English course with AHA! In *Proceedings of the First International Workshop on Group Adaptation in Collaborative Learning Environments*, 2008. URL http://ftp.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-384/POSTER-p1.pdf. 19

David Hutchison, Peter Brusilovsky, Peter Dolog, and Leonid Pesin. *Adaptive Hypermedia and Adaptive Web-Based Systems*, volume 5149 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2006. ISBN 9783540709848. URL http://www.springerlink.com/index/10.1007/11768012. 2

IEEE Computer Society. IEEE Recommended Practice for Architectural Description of Software-Intensive Systems, 2000. 52

Anthony Jameson. *Adaptive interfaces and agents*, chapter 15, pages 305–330. Lawrence Erlbaum Associates, Mahwah, New Jersey, 2003. 6, 11, 23

Sarah Jones and Stefanie N. Lindstaedt. *A Multi-Activity Distributed Participatory Design Process for Stimulating Creativity in the Specification of requirements for a Work-Integrated Learning System*. 2008. 34

Pythagoras Karampiperis and Demetrios Sampson. Adaptive learning resources sequencing in educational hypermedia systems. *Educational Technology & Society*, 8(4):128–147, 2005. ISSN 14364522. URL http://www.ifets.info/journals/8_4/13.pdf. 12

Alfred Kobsa. Generic User Modeling Systems. *User Modeling and UserAdapted Interaction*, 11(1):49–63, 2001. ISSN 09241868. URL http://www.springerlink.com/index/G5G2030040352416.pdf. 7

Alfred Kobsa. Privacy-enhanced personalization. *Communications of the ACM*, 50(8):24–33, 2007. ISSN 00010782. URL http://portal.acm.org/citation.cfm?doid=1278201.1278202. 83

Alfred Kobsa, Dietmar Müller, and Andreas Nill. KN-AHS : An Adaptive Hypertext Client of the User Modeling System BGP-MS. In *Readings in Intelligent User Interfaces*, volume 1, pages 99–105. Morgan Kaufmann, 1994. 14

Alfred Kobsa, Jürgen Koenemann, and Wolfgang Pohl. Personalised hypermedia presentation techniques for improving online customer relationships. *The Knowledge Engineering Review*, 16(02):111, 2001. ISSN 02698889. doi: 10.1017/S0269888901000108. URL http://www.journals.cambridge.org/abstract_S0269888901000108. 6, 12

Nora Koch and Martin Wirsing. The Munich Reference Model for Adaptive Hypermedia Applications. *Lecture Notes in Computer Science*, 2347(May):213–222, 2002. URL http://www.springerlink.com/index/4G5VUMEQ8UB16XYU.pdf. 7

Jose Kooken, Tobias Ley, and Robert de Hoog. How Do People Learn at the Workplace? Investigating Four Workplace Learning Assumptions. In Erik Duval, Ralf Klamma, and Martin Wolpers, editors, *Creating New Learning Experiences on a Global Scale*, volume 4753 of *Lecture Notes in Computer Science*, pages 158–171. Springer Berlin / Heidelberg, 2007. URL http://dx.doi.org/10.1007/978-3-540-75195-3_12DO-10.1007/978-3-540-75195-3_12. 25

Phillippe Kruchten. Architecture blueprints-the "4+1" view model of software architecture. In *Tutorial proceedings on TRI-Ada '91 Ada's role in global markets: solutions for a changing complex world - TRI-Ada '95*, volume 12, pages 540–555, New York, New York, USA, 1995. ACM Press. ISBN 0897917057. URL http://portal.acm.org/citation.cfm?doid=216591.216611. 52, 57

Barbara Kump. *Evaluating the Domain Model of Adaptive Work-Integrated Learning Systems (unpublished thesis)*. PhD thesis, University of Graz, 2010. 3, 74

Barbara Kump, Christin Seifert, Guenter Beham, Stefanie N. Lindstaedt, and Tobias Ley. MyExperiences: Visualizing Evidence in an Open Learner Model. In *Adjunct Proceedings* *of the 18th International Conference on User Modeling Adaptation and Personalization Posters and Demonstrations*, pages 16–18, 2010. 4, 81

Jean Lave and Etienne Wenger. *Situated learning: Legitimate peripheral participation*, volume 29 of *Learning in Doing: Social, Cognitive, and Computational Perspect*. Cambridge University Press, 1991. ISBN 0521423740. URL http://books.google.com/books?id=CAVIOrW3vYAC&pgis=1. 1

James C Lester, Rosa Maria Vicari, Fábio Paraguaçu, Soumaya Chaffar, and Claude Frasson. *Intelligent Tutoring Systems*, volume 3220 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2004. ISBN 9783540229483. URL http://www.springerlink.com/content/mmafayxjmwbwypur/. 5

Tobias Ley. Measuring Intellectual Capital: Experience and Reconsideration. In Ulrich Reimer, Abecker Andreas, Steffen Staab, and Gerd Stumme, editors, *WM 2003: Professionelles Wissesmanagement - Erfahrungen und Visionen, Beiträge der 2. Konferenz Professionelles Wissensmanagement*. Springer LNI, 2003. 24

Tobias Ley, Stefanie N Lindstaedt, and D Albert. *Supporting Competency Development in Informal Workplace Learning*, volume 3082 of *Lecture Notes in Computer Science*, pages 189–202. Springer, 2005. URL http://www.springerlink.com/content/f684734800253235/?p=0004a80a8cc24ce1b5183238edee70a2&pi=3. 1

Tobias Ley, Barbara Kump, and Dietrich Albert. A methodology for eliciting, modelling, and evaluating expert knowledge for an adaptive work-integrated learning system. *International Journal of Human-Computer Studies*, 68(4):185–208, 2010a. ISSN 10715819. URL http://linkinghub.elsevier.com/retrieve/pii/S1071581909001773. 27, 46

Tobias Ley, Barbara Kump, and Cornelia Gerdenitsch. *Scaffolding Self-directed Learning with Personalized Learning Goal Recommendations*, volume 6075 of *Lecture Notes in Computer Science*, pages 75–86. Springer Berlin / Heidelberg, 2010b. URL http://dx.doi.org/10.1007/978-3-642-13470-8_9DO-10.1007/978-3-642-13470-8_9. 76, 83

Ching-Yung Lin, Kate Ehrlich, Vicky Griffiths-Fisher, and Christopher Desforges. SmallBlue: People Mining for Expertise Search. *IEEE MultiMedia*, 15(1):78–84, 2008. ISSN 1070-986. URL http://portal.acm.org/citation.cfm?id=1399296. 28

Stefanie N Lindstaedt, Tobias Ley, and Harald Mayer. Integrating Working and Learning with APOSDLE. *Innovation*, pages 1–5, 2006. 33

Stefanie N. Lindstaedt, Tobias Ley, Peter Scheir, and Armin Ulbrich. Applying Scruffy Methods to Enable Work-integrated Learning. *Upgrade: The European Journal of the Informatics Professional*, 9(3):44–50, 2008a. 28, 74

Stefanie N. Lindstaedt, Peter Scheir, Robert Lokaiczyk, Barbara Kump, Günter Beham, and Viktoria Pammer. Knowledge Services for Work-integrated Learning. *Times of Convergence Technologies Across Learning Contexts*, pages 234–244, 2008b. URL http://www.springerlink.com/index/j5388087600kr1r6.pdf. 4, 23

Stefanie N. Lindstaedt, Mario Aehnelt, and Robert de Hoog. Supporting the Learning Dimension of Knowledge Work. In *Proceedings of the 4th European Conference on Technology Enhanced Learning: Learning in the Synergy of Multiple Disciplines*, EC-TEL '09, pages 639–644, Berlin, Heidelberg, 2009a. Springer-Verlag. ISBN 978-3-642-04635-3. URL http://dx.doi.org/10.1007/978-3-642-04636-0_61. 1

Stefanie N. Lindstaedt, Günter Beham, Barbara Kump, and Tobias Ley. Getting to Know Your User - Unobtrusive User Model Maintenance within Work-Integrated Learning Environments. In Ulrike Cress, Vania Dimitrova, and Marcus Specht, editors, *Learning in the Synergy of Multiple Disciplines*, volume 5794, pages 73–87. Springer Berlin / Heidelberg, 2009b. URL `http://portal.acm.org/citation.cfm?id=1691911`. 4, 11, 23, 27, 30, 33

Stefanie N. Lindstaedt, Barbara Kump, Günter Beham, Viktoria Pammer, Tobias Ley, Amir Dotan, and Robert de Hoog. Providing Varying Degrees of Guidance for Work-Integrated Learning. In Martin Wolpers, Paul A. Kirschner, Maren Scheffel, Stefanie Lindstaedt, and Vania Dimitrova, editors, *Sustaining TEL: From Innovation to Learning and Practice*, volume 6383 of *Lecture Notes in Computer Science*, pages 213–228, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. ISBN 978-3-642-16019-6. URL `http://www.springerlink.com/content/g17m006200212511/`. 1, 4

Robert Lokaiczyk, Eicke Godehardt, Andreas Faatz, Manuel Goertz, Andrea Kienle, Martin Wessner, and Armin Ulbrich. *Exploiting Context Information for Identification of Relevant Experts in Collaborative Workplace-Embedded E-Learning Environments*, volume 4753 of *Lecture Notes in Computer Science*, pages 217–231. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007. ISBN 978-3-540-75194-6. URL `http://www.springerlink.com/content/y427117k24467140/`. 44

Mark T. Maybury. *Expert Finding Systems.* The MITRE Corporation, Bedford, Massachusetts, 2006. 24, 29, 47

Mark T. Maybury, Raymond J. D'Amore, and David House. *Automated Discovery and Mapping of Expertise*, pages 359–382. MIT Press, Cambridge, 2002. 28

Wolfgang Nejdl, P Fröhlich, and M Wolpers. KBS-Hyperbook - An Open Hyperbook System for Education. In *10th World Conf on Educational Media and Hypermedia EDMEDIA98 Freiburg Germany*, 1998. 13

Reinhard Oppermann. *Adaptive user support: ergonomic design of manually and automatically adaptable software.* Series on computers, coginition, and work. Lawrence Erlbaum Associates, Inc., 1994. ISBN 0805816550. URL `http://portal.acm.org/citation.cfm?id=213146`. 6

Christine Pabst. *Evaluating a User Model with Implicit Knowledge Diagnosis: A Field Study. Unpublished Master's Thesis. Institute for Psychology, University of Graz.* 2011. 76, 83

Alfred Radl. *Reflexionsintegration im arbeitsintegrierten Lernen. Unpublished Master's Thesis, Knowledge Management Institute, Graz University of Technology.* 2009. 83

Daniel Resanovic. *Configurable User Profiling Service and Simulation of User Behavior in APOSDLE. Unpublished Master's Thesis. Knowledge Management Institute, Graz University of Technology.* 2010. 76, 80, 83

Peter Scheir, Stefanie N. Lindstaedt, and Chiara Ghidini. A Network Model Approach to Retrieval in the Semantic Web. *International Journal on Semantic Web and Information Systems*, 4(4):56–84, 2008. 46, 61, 78

Andreas Schmidt. Bridging the Gap between Knowledge Management and E-Learning with Context-Aware Corporate Learning. In K.-D. Althoff, A Dengel, R Bergmann, and T Roth-Berghofer, editors, *Professional Knowledge Management. Third Biennal Conference, WM 2005, Kaiserslautern, Germany, April 2005. Revised Selected Papers, Lecture Notes in Artificial Intelligence (LNAI)*, volume 3782, pages 203–213. Springer, 2005. 28

Ingo Schwab and Alfred Kobsa. Adaptivity through Unobstrusive Learning. *KI Special Issue on Adaptivity and User Modeling*, 3:5–9, 2002. 27

Daniel Schwabe and Gustavo Rossi. Developing Hypermedia Applications using OOHDM. In *Workshop on Hypermedia Development Processes Methods and Models Hypertext98*, volume 98, pages 1–20. ACM Press, 1998. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.40.4780. 7

Peter J Smith. Workplace Learning and Flexible Delivery. *Review of Educational Research*, 73(1):53–88, 2003. 1

Sarah Spiekermann and Lorrie Faith Cranor. Engineering Privacy. *IEEE Transactions on Software Engineering*, 35(1):67–82, 2009. ISSN 00985589. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4657365. 24

Theophanis Tsandilas and M C Schraefel. Usable Adaptive Hypermedia Systems. *New Review Of Hypermedia And Multimedia*, 10(1):25, 2004. URL http://eprints.ecs.soton.ac.uk/9333/. 14

Adriana Vivacqua and Henry Lieberman. Agents to assist in finding help. In *Conference on Human Factors in Computing Systems*, pages 65–72. ACM-Press, New York, 2000. URL http://portal.acm.org/citation.cfm?id=332408. 28

Yang Wang and Alfred Kobsa. *Respecting Users' Individual Privacy Constraints in Web Personalization*, pages 157–166. Springer, 2007. URL http://dx.doi.org/10.1007/978-3-540-73078-1_19. 24

Gerhard Weber and Peter Brusilovsky. ELM-ART: An Adaptive Versatile System for Web-based Instruction. *International Journal of Artificial Intelligence in Education*, 12:351–384, 2001. 5

Gerhard Weber and Antje Möllenberg. ELM-PE: A knowledge-based programming environment for learning LISP. In *Proceedings of ED-MEDIA 94-World Conference on Educational Multimedia and Hypermedia*, volume Proceeding, pages 557–562, 1994. 17

Gerhard Weber and Marcus Specht. User Modeling and Adaptive Navigation Support in WWW-Based Tutoring Systems. In *Proceedings of the Sixth International Conference on User Modeling, UM97*, pages 289–300. Springer, 1997. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.48.1139. 17

Gerhard Weber, H.-C. Kuhl, and S Weibelzahl. Developing adaptive internet based courses with the authoring system NetCoach. In P De Bra, P Brusilovsky, and A Kobsa, editors, *Proceedings of the third workshop on Adaptive Hypertext and Hypermedia*, pages 35–48. 2001. 17

Martin Wolpers. Contextualized Attention Metadata in Learning Environments. *Learning*, 9(3):57–61, 2008. URL http://www.upgrade-cepis.org/issues/2008/3/up9-3-Wolpers.pdf. 27

Dawit Yimam-Seid and Alfred Kobsa. Expert Finding Systems for Organizations: Problem and Domain Analysis and the DEMOIR Approach. *Journal of Organizational Computing and Electronic Commerce*, 13(1):1–24, 2003. 24, 29, 48, 83

Andreas Zinnen, Sybille Hambach, Andreas Faatz, Günter Beham, Stefanie N Lindstaedt, Manuel Görtz, and Robert Lokaiczyk. Datenschutzfragen bei der Etablierung einer Arbeitsprozess-integrierten e-Learning-Lösung. In *DeLFI'08*, pages 341–352, 2008. 78