UNIVERSITY OF NOVI SAD
FACULTY OF SCIENCE
DEPARTMENT OF
MATHEMATICS AND INFORMATICS

Aleksandra Klašnja-Milićević

Personalized Recommendation Based on Collaborative Tagging Techniques for an E-learning System

– Doctoral Dissertation –

Mentor:
dr Mirjana Ivanović

Novi Sad, 2012. god.

Contents

List of Tables

List of Figures

Acknowledgments

First of all, I am deeply grateful to my mentor prof. dr Mirjana Ivanović. She has supported me in all my endeavors and provided encouragement and excellent guidance throughout this work's lifetime. I enjoy the freedom of discussing many research ideas with her. She is always able to provide insightful feedback and prompt help so that I can move forward on the right track.

I would like to thank the members of the dissertation defense board, dr Zoran Budimac (University of Novi Sad, Serbia), dr Dragan Janković (University of Niš, Serbia), and dr Elvira Popescu (University of Craiova, Romania) for helpful comments which have given both new prospective and a stronger thesis.

I am greatly thankful to my parents: mother Radmila and father Đorđe, and my sister Marija for making the small and big sacrifices in their lives, so that I could achieve the small and big achievements of my life.

My deepest thank goes to my husband, Dejan, for his devoted love, understanding, and patience, and for listening to my complaints and encouraging me to complete this work. Also, special warm thanks go to my lovely son Marko, for his love and his devotion, helping me in any conceivable respect.

The presented research was supported by Ministry of Education, Science and Technological Development, through project: "*Intelligent techniques and their integration into wide-spectrum decision support*" (project no. 174023).

Novi Sad, October 2012 Aleksandra Klašnja - Milićević

Part I

Preliminaries

Chapter 1

Introduction

With the development of sophisticated e-learning environments which characterize the huge information, the strong interactivity, the great coverage and no space-time restrictions (Anane et al., 2004; Mallinson and Sewry, 2004), personalization is becoming an important feature in e-learning systems due to the differences in background, goals, capabilities and personalities of the large numbers of learners, the main users of such systems. Personalized learning occurs when e-learning systems are designed according to educational experiences that fit the needs, goals, talents, and interests of their learners. Personalization can be achieved using pre-defined rules that sequentially propose learning objects in a specified learning path (Koper and Oliver, 2004). It is also achieved by using heuristic rules, user models and recommendation techniques (Resnick and Varian, 1997).

Ideally, Recommender System (RS) in e-learning environments should assist learners in discovering relevant learning actions that perfectly match their profile, at the right time, in the right context, and in the right way, keep them motivated and enable them to complete their learning activities in an effective and efficient way (Tang and McCalla, 2005).

Different kind of algorithms such as user-based and item-based collaborative filtering have been used to establish a RS (Konstan et al., 2004). Although the assumption that collaborative filtering relied on works well in narrow domains, it is likely to fail in more diverse or mixed settings. The reason is obvious: people who have similar taste in one domain may behave quite differently in others.

To improve recommendation quality, metadata such as content information of items has typically been used as additional knowledge (Karen et al., 2008). With the increasing popularity of the collaborative tagging systems, tags could be interesting and useful information to enhance algorithms for RSs. Collaborative tagging systems have grown in popularity over the Web in the last years based on their simplicity to categorize and retrieve content using open-ended tags (Golder and Huberman, 2006). The increasing number of users providing information about themselves through social tagging activities caused the emergence of tag-based profiling approaches, which assume that users expose their preferences for certain contents through tag assignments (Klašnja-Milićević et al., 2010). Besides helping user to organize his or her personal collections, a tag also can be regarded as a user's or expert's personal opinion expression, while tagging can be considered as implicit rating or voting on the tagged information resources or items (Halpin et al., 2007). Thus, the tagging information can be used to make recommendations. Tag-based RSs in e-learning could support learners in their own learning path by recommending tags and learning objects, and also could promote the learning performance of individual learners. These systems can use different recommendation techniques in order to suggest online learning activities or optimal browsing pathways to learners, based on their preferences, learning style, knowledge level and the browsing history of other learners with similar characteristics (Godwin and Kaplan, 2008).

Computer technology has been used to develop a vast array of educational software, from early computer-based training systems to web-based adaptive hypermedia, multimedia courseware, and educational games (Klašnja-Milićević et al., 2011b). This dissertation studies specific kind of tutoring systems which apply two types of personalization:

1. personalization that adjust the course to learners' individual learning styles and
2. recommendation techniques which suggest the most appropriate online learning activities to learners, based on their knowledge, preferences and the tags information of other learners with similar characteristics.

Dissertation research aims to analyze capabilities of integration recommender systems based on collaborative tagging techniques into adaptive and intelligent web-based programming tutoring system – Protus (PRogramming TUtoring System) that takes into account pedagogical aspects of the learner. The research will be focused on appropriate selection of collaborative tagging techniques which could lead to applying the best results in terms of increasing motivation in learning process and understanding of the learning content. As a result personalized and the most likely preferred recommendations can be estimated to an active learner that will be in accordance to the learner's interests, his learning style and previously acquired knowledge. The structure of this dissertation is outlined in the following section.

1.1. Dissertation Outline

This dissertation is organized into three major parts. *Part I: Preliminaries*, which includes the first two chapters of the dissertation, introduces the motivation and objectives studied in the subsequently presented research, and provides an overview of techniques for recommender systems, folksonomy and tag-based recommender systems to assist the reader in understanding the material which follows. The overview, presented in Chapter 2 includes descriptions of content-based recommender systems, collaborative filtering systems, hybrid approach, memory-based and model-based algorithms, features of collaborative tagging that are generally attributed to their success and popularity, as well as model for tagging activities and tag-based recommender systems.

Part II: Personalized Recommendation for a Programming Tutoring System, which consists of Chapters 3 and 4, presents the most important requests for designing a recommender system in e-learning environments, as well as design, architecture and interface of the Protus system. Section 3.1 is dedicated to most important requirements and challenges for designing a recommender system in e-learning environments. Section 3.2 examines traditional recommendation methods: Collaborative filtering, Content - based techniques and Association rule mining and observes challenges and various limitations for each of these traditional recommendation methods. Applying tag-based RS to e-learning environments and suitable recommendation techniques using collaborative tags of learners are described in Section 3.3. The limitations of current folksonomy and possible solutions are given in Section 3.4. Chapter 4 consists of 3 sections. After reviewing and illustrating related work in Section 4.1, Section 4.2 outlines the overall system architecture and describes the recommendation module and process of personalization in details. Protus interface is described in the Section 4.3.

Part III: Evaluation and discussion, wich contains Chapter 5 and 6, highlights the results of the evaluation and discussion of analysis of the results regarding the validity of the system. Chapter 5 contains 7 sections. Section 5.1 describes data definition, while process data clustering is explained in section 5.2. Statistical properties of learners' tagging history are analyzed in section 5.3. Experimental protocol and evaluation metrics are referred to in Section 5.4. Experimental results are given in Section 5.5 Section 5.6 outlines the results of an experimental test which was created with the expert tag set. Section 5.7 describes Protus system from the educational point of view. Finally, Chapter 6 concludes the dissertation, summarizing the main contributions and discussing the possibilities for future work.

1.2. Research Objectives of the Dissertation

Research of the dissertation aims to analyze and define a model to select tags that reveal the preferences and characteristics of users required to generate personalized recommendations and options on the use of models for personalized tutoring system. Personalized learning occurs when e-learning systems make deliberate efforts to design educational experiences that fit the needs, goals, talents, learning styles, interests of their learners and learners with similar characteristics.

This thesis is devoted to analyze capabilities of integration recommender systems based on collaborative tagging techniques into Java Tutoring system. Appropriate selection of collaborative tagging techniques could lead to applying the best results in terms of increasing motivation in learning process and understanding of the learning content. The scientific objectives are summarized as follows.

1. a comprehensive survey of the state-of-the-art in collaborative tagging systems and folksonomy for tagging activities which can be used for extending the capabilities of recommender systems,
2. identification of limitations of the current generation of collaborative tagging techniques and discussion about some initial approaches for extending their capabilities,

3. theoretical overview of tag-based recommender systems in e-learning environments,
4. applying a recommender system based on collaborative tagging techniques in developing a tutoring system that adapts to the learner’s learning style and level of learners' knowledge in the field of programming languages and
5. experimental comparison of described techniques.

As a result personalized and the most likely preferred recommendations can be estimated to an active learner that will be in accordance with the learner's interests, his learning style, demographic characteristics and previously acquired knowledge.

Chapter 2

Recommender Systems, Folksonomy and Tag-based Recommender Systems

The information in the Web is increasing far more quickly than people can cope with. Users are forced review a number of choices before they discover what they need. This is often time consuming and frustrating. Given today's fast paced lifestyle, a slow and careful search for elusive item of choice is surely not a sustainable option. People would rather look at items that are customized to their interests and preferences. Personalized recommendation (Resnick and Varian 1997) can help people to overcome the information overload problem, by recommending items according to users' interests.

RSs can be defined as a platform for providing recommendations to users based on their personal likes and dislikes. These systems use specific type of information filtering (IF) technique that attempt to present to the user information items (movies, music, books, news, web pages, learning objects, etc.) the user is interested in (Ricci et al. 2011). To do this the user's profile is compared to some reference characteristics. These characteristics may be from the information item (the content-based approach) or the user's social environment (the collaborative filtering approach) (Adomavicius and Tuzhilin, 2005).

Typically, RSs apply personalization techniques, considering that different users have different preferences and different information needs (Konstan et al., 1997). In order to generate personalized recommendations that are tailored to the user's specific needs, recommender systems must collect personal preference information, e.g., the user's history of purchase, click-stream data, demographic information, and so forth. Traditionally, expressions of preference of users for products are generally called ratings. Two different types of ratings are distinguished.

1. *Explicit ratings.* Users are required to explicitly specify their preference for any particular item, usually by indicating their extent of appreciation on 5-point or 7-point likert scales (Alton-Scheidl et al, 1997).

2. *Implicit ratings.* Explicit ratings require additional efforts on users. Consequently, users often tend to avoid the burden of explicitly stating their preferences and either leave the system or rely upon “free-riding” (Avery and Zeckhauser, 1997). Alternatively, gathering preference information from observations of user behavior is less intrusive (Nichols, 1998).

A ratings database consists of pairs of users and items rated by them, along with additional information such as timestamps etc. Given such a ratings database and a set of preliminary ratings by a new user, the basic requirement of a recommendation system is to recommend the largest and the most significant set of recommendations to the new user (Miller, 2003). The largest set of recommendations refers to the maximum number of items in the items database that could be recommended to the user. The significant items are the ones that the new user is most likely to rate in the future.

Based on the nature of reference characteristics, two broad categories of information filtering for computing recommendations have emerged: content-based filtering, and collaborative filtering (Goldberg et al., 1992).

Content-based recommender systems recommend items “by comparing representations of content contained in an item to representations of content that interests the user” (Malone et al., 1987). In most cases, a keyword profile is built. Apparently, this works well in text domains but not in domains where there is not much content associated with the items or where a computer cannot easily analyze this content. Relying on rich descriptions, content-based recommender systems need significant knowledge engineering efforts to create substantial metadata for the items. Content-based systems “form profiles for each user independently” (Basilico and Hofmann, 2004). Even if two items were in two neighbor categories that people normally like both them, the item from category A would never be recommended to the user if he only rated items from category B. This problem is often addressed by introducing some unpredictability. Also, the user has to rate a suitable number of items before a content-based recommender system can really comprehend the user’s preferences and present the user with trustworthy recommendations. Therefore, a new user, having very few ratings, would not be able to get accurate recommendations.

Collaborative filtering systems compute profile similarity between the target user and the other users “by comparing users’ opinions of items” (Balabanović and Shoham 1997). Profile similarity is usually computed by comparing rating-vectors with various distance metrics, e.g. Pearson correlation or cosine similarity. They provide the user with the items he will most likely be interested in, either one single item or a “ranked list of items” – usually referred as *top-N-item* (Cosley et al., 2002; McLaughlin and Herlocker, 2004). In contrast to content-based systems, recommender systems based on collaborative filtering can provide the user with unexpected but fitting recommendations that do not have anything in common with aforerated items. Collaborative filtering is a very successful methodology in almost every domain – especially “where multi-value ratings are available” (McLaughlin and Herlocker, 2004). However, they suffer from two key problems: sparsity and first-rater problem. As most users only rate a small portion of all items, it is highly difficult to find users with “significantly similar ratings.” Furthermore an item cannot be recommended before one user has rated it. This can be the case if the item has newly been introduced to the system (Melville, Mooney and Nagarajan, 2002).

A number recommendation systems use a **hybrid approach** by combining collaborative and content-based methods, which helps to avoid certain limitations of content and collaborative-based systems (Balabanović and Shoham 1997, Basu et al., 1998, Claypool et al., 1999, Pennock and Horvitz 1999). Different ways to combine collaborative and content-based methods into a hybrid recommender system can be classified as follows (Adomavicius, G., and Tuzhilin, 2005):

1. implementing collaborative and content-based methods separately and combining their predictions,
2. incorporating some content-based characteristics into a collaborative approach,
3. incorporating some collaborative characteristics into a content-based approach, and
4. constructing a general unifying model that incorporates both content-based and collaborative characteristics.

According to Breese et al. (1998), algorithms for collaborative recommendations can be grouped into two general classes: memory-based (or heuristic-based) and model-based.

Memory-based algorithms (Breese et al. 1998, Delgado and Ishii, 1999, Resnick et al., 1994. Shardanand and Maes, 1995) essentially are heuristics that utilize the entire database of user preferences when computing recommendations. These algorithms tend to be simple to implement and require little training cost. They can also easily take new preference data into account. However, their online performance tends to be slow as the size of the user and item sets grow, which makes these algorithms as stated in the literature unsuitable in large systems. One workaround is to only consider a subset of the preference data in the calculation, but doing this can reduce both recommendation quality and the number of items that can be recommended due to data being omitted from the calculation. Another solution is to perform as much of the computation as possible in an offline setting. However, this may make it difficult to add new users to the system on a real-time basis, which is a basic necessity of most online systems. Furthermore, the storage requirements for the precomputed data could be high.

Model-based algorithms (Billsus and Pazzani, 1998, Goldberg et al., 2001, Hofmann, 2003) use the collection of ratings to learn a model, which is then used to make rating predictions. Often, the model building process is time-consuming and is only used periodically. The model is compact and can generate recommendations very quickly. The disadvantage of model-based algorithms is adding new users, items, or preferences which can be the same as recomputing the entire model.

The most important difference between collaborative model-based techniques and heuristic-based approaches is that the model-based techniques calculate utility predictions based not on some ad hoc heuristic rules, but, rather, based on a model learned from the underlying data using statistical and machine learning techniques (Adomavicius, G., and Tuzhilin, 2005). A method combining memory-based and model-based approaches was proposed in (Pennock and Horvitz, 2000). It was empirically confirmed that the use of this approach can afford better recommendations than pure memory-based and model-based collaborative approaches.

Over the past several years there has been much research done on recommendation technologies which use a variety of statistical, machine learning, information retrieval, and other techniques that have significantly advanced early recommender systems, collaborative and content-based heuristics. As was discussed above, recommender systems can be classified as 1) content-based, collaborative, or hybrid (based on the recommendation approach used), and 2) heuristic-based or model-based (based on the types of recommendation techniques used for the rating estimation). These two orthogonal dimensions are used to classify the recommender systems research in the 2 × 3 matrix, presented in Table 1 (Adomavicius, G., and Tuzhilin, 2005).

The recommendation techniques explained in this chapter have performed well in several applications, including the ones for recommending books, CDs, news articles or movies (Marlin, 2003; Rosset et al., 2002) and some of these methods are used in the "industrial-strength" recommender systems, such as the ones developed at Amazon[1], MovieLens[2], and Last.fm[3]. However, both collaborative and content-based methods have certain limitations. Recommender systems can be extended in several ways that include improving the understanding of users and items, incorporating the contextual information into the recommendation process, sustaining multicriteria ratings, and providing more flexible and less disturbing types of recommendations (Adomavicius and Tuzhilin, 2005).

Collaborative tagging is employed as an approach which is used for automatic analysis of user preference and recommendation. To improve recommendation quality, metadata such as content information of items has typically been used as additional knowledge. With the increasing reputation of the collaborative tagging systems, tags could be interesting and useful information to enhance algorithms for recommender systems. Collaborative tagging systems allow users to upload their resources, and to label them with arbitrary words, so-called tags (Golder and Huberman, 2005). The systems can be distinguished according to what kind of resources are supported. Flickr[4], for instance, allows the sharing of photos, del.icio.us[5] the sharing of bookmarks, CiteULike[6] and Connotea[7] the sharing of bibliographic references, and 43Things[8] even the sharing of goals in private life. These systems are all very similar. Once a user is logged in, he can add a resource to the system, and assign arbitrary tags to it. The collection of all his assignments is his personomy, the collection of all personomies constitutes the folksonomy. The user can explore his personomy, as well as the personomies of the other users, in all dimensions: for a given user one can see all resources he had uploaded,

[1] http://www.amazon.com

[2] http://www.movielens.umn.edu

[3] http://www.last.fm

[4] http://www.flickr.com

[5] http://www.del.icio.us

[6] http://www.citeulike.org

[7] http://www.connotea.org

[8] http://www.43things.com

together with the tags he had assigned to them (Jäschke et al., 2007). Besides helping user to organize his or her personal collections, a tag also can be regarded as a user's personal opinion expression, while tagging can be considered as implicit rating or voting on the tagged information resources or items (Liang et al., 2008). Thus, the tagging information can be used to make recommendations. The rest of this chapter will review in more detail the collaborative tagging systems, folksonomy and tag-based RSs, relevant to the results of the research presented in this dissertation. Section 2.1 provides comprehensive survey of the state-of-the-art in collaborative tagging systems and folksonomy. Section 2.2 presents a model for tagging activities. Tag-based recommender systems and approaches for extension and collecting tags are described in Section 2.3.

Table 2.1 Classification of RS research (Adomavicius, G., and Tuzhilin 2005)

Recommendation Approach	Recommendation technique	
	Heuristic based	Model based
Content - based	Commonly used techniques: • TF-IDF (information retrieval) • Clustering Representative research examples: • Lang 1995 • Balabanović, Shoham 1997 • Pazzani & Billsus 1997	Commonly used techniques: • Bayesian classifiers • Clustering • Decision trees Representative research examples: • Pazzani & Billsus 1997 • Mooney et al. 1998 • Mooney & Roy 1999 • Billsus & Pazzani 1999, 2000 • Zhang et al. 2002
Collaborative	Commonly used techniques: • Nearest neighbor (cosine, corellation) • Clustering • Graph theory Representative research examples: • Resnick et al. 1994 • Hill et al. 1994 • Shardannand and Maes 1995 • Breese et al. 1998 • Nakamura & Abe 1998 • Aggarwal et al. 1999 • Delgado & Ishii 1999 • Pennock & Horwitz 1999 • Sarwar et al. 2001	Commonly used techniques: • Bayesian networks • Clustering • Artificial neural networks • Linear regression • Probabilistic models Representative research examples: • Billsus & Pazzani 1998 • Breese et al. 1998 • Goldberg et al. 2001 • Ungar & Foster 1998 • Chien & George 1999 • Getoor & Sahami 1999 • Pennock & Horwitz 1999 • Pavlov & Pennock 2002 • Shani et al. 2002 • Hofmman 2003, 2004
Hybrid	Combining content – based and collaborative components by: • Linear combination of pred icted ratings • Various voting schemes • Incorporating one component as a part of the heuristic for the other Representative research examples: • Balabanović, Shoham 1997 • Pazzani 1999 • Billsus & Pazzani 1998 • Claypool et al. 1999 • Good et al. 1999 • Train & Cohen 2000	Combining content – based and collaborative components by: • Incorporating one component as a part of the model for the other • Building one unifying model Representative research examples: • Soboroff & Nicholas 1999 • Basu et al. 1998 • Condiff et al. 1999 • Popescul et al. 2001 • Schein et al. 2002 • Ansari et al. 2000

2.1. A Survey of Collaborative Tagging Systems and Folksonomy

Collaborative tagging is the practice of allowing users to freely attach keywords or tags to content (Golder and Huberman, 2005). Collaborative tagging is most useful when there is nobody in the "librarian" role or there is simply too much content for a single authority to classify. People tag pictures, videos, and other resources with a couple of keywords to easily retrieve them in a later stage. The following features of collaborative tagging are generally attributed to their success and popularity (Mathes, 2004; Quintarelli, 2005; Wu et al., 2006).

- *Low cognitive cost and entry barriers.* The simplicity of tagging allows any Web user to classify their favorite Web resources by using keywords that are not constrained by predefined vocabularies.
- *Immediate feedback and communication.* Tag suggestions in collaborative tagging systems provide mechanisms for users to communicate implicitly with each other through tag suggestions to describe resources on the Web.
- *Quick adaptation to changes in vocabulary.* The freedom provided by tagging allows fast response to changes in the use of language and the emergency of new words. Terms like Web2.0, ontologies and social network can be used readily by the users without the need to modify any pre-defined schemes.
- *Individual needs and formation of organization.* Tagging systems provide a convenient means for Web users to organize their favorite Web resources. Besides, as the systems develop, users are able to discover other people who are also interested in similar items.
- *Scalability*. Predefined vocabularies become imprecise when a domain grows. Instead, tags can reach a nearly unlimited granularity.
- *Serendipity*. Controlled vocabularies are designed to ease retrieval. Less popular content that resides in the so-called long-tail of the information space is hard to find. Tags enable users to discover long-tail information by browsing through the folksonomy network of items, tags, and users.
- *Inclusiveness*. The set of potential tags includes every user's views, preferences, or language as well as all potential topics.

Since tags are created by individual users in a free form, one important problem facing tagging is to identify most appropriate tags, while eliminating noise and spam. For this purpose, Noll et al. (2009) define a set of general criteria for a good tagging system.

- *High coverage of multiple facets.* A good tag combination should include multiple facets of the tagged objects. The larger the number of facets the more likely a user is able to recall the tagged content.
- *High popularity.* If a set of tags are used by a large number of people for a particular object, these tags are more likely to uniquely identify the tagged content and the more likely to be used by a new user for the given object.
- *Least-effort.* The number of tags for identifying an object should be minimized, and the number of objects identified by the tag combination should be small. As a

result, a user can reach any tagged objects in a small number of steps via tag browsing.

- *Uniformity (normalization).* Since there is no universal ontology, different people can use different terms for the same concept. In general, we have observed two general types of divergence: those due to syntactic variance, e.g., color, colorize, colorise, colourise; and those due to synonym, e.g., learner and pupil, which are different syntactic terms that refer to the same underlying concept. These kinds of divergence are a double-edged sword. On the one hand, they introduce noises to the system; on the other hand it can increase recall.
- *Exclusion of certain types of tags.* For example, personally used organizational tags are less likely to be shared by different users. Thus, they should be excluded from public usage. Rather than ignoring these tags, tagging system includes a feature that auto-completes tags as they are being typed by matching the prefixes of the tags entered by the user before. This not only improves the usability of the system but also enables the convergence of tags.

Another important aspect of tagging systems is how they operate. Marlow et al. (2006) explain some important dimensions of tagging systems' design that may have immediate effect on the content and effectiveness of tags generated by the system. Some of these dimensions are listed below.

1. Tagging Rights

The permission a user has to tag resources can effect the properties of an emergent folksonomy. Systems can determine who may remove a tag. Also, systems can choose the resources which users tag or specify different levels of permissions to tag. The spectrum of tagging permissions ranges from:

a. Self-tagging - users can only tag their own contributions (e.g. Technorati[9]), through

b. Permission-based - users make a decision who can tag their resources (e.g. Flickr), to

c. Free-for-all - any user can tag any resource

2. Tagging Support

One important aspect of a tagging system is the way in which users assign tags to items. They may assign arbitrary tags without prompting, they may add tags while considering those already added to a particular resource, or tags may be proposed. There are three different categories:

a. Blind tagging - user cannot see the other tags assigned to the resource they're tagging

b. Viewable tagging - users can see the other tags assigned to the resource they're tagging

---

[9] http://www.technorati.com

c. Suggestive tagging - user can see recommended tags for the resource they're tagging

3. Aggregation

The aggregation of tags around a given resource is an important consideration. The system may allow for a multiplicity of tags for the same resource which may result in duplicate tags from different users. Alternatively, many systems ask the group to collectively tag an individual resource. It is able to distinguish two models of aggregation.

a. Bag-model - the equal tag can be assigned to a resource multiple times, like in Delicious, allowing statistics to be generated and users to see if there is agreement among taggers about the content of the resource

b. Set-model - a tag can be applied only once to a resource, like in Flickr

4. Types of Object

The implications for the nature of the resultant tags are numerous. The types of resource tagged allow us to distinguish different tagging systems. Popular systems include simple objects, like: webpages, bibliographic materials, images, videos, songs, etc. Tags for text objects and multimedia objects can be varied. In reality, any object that can be virtually represented can be tagged or used in a tagging system. For example, systems exist that let users tag physical locations or events (e.g., Upcoming[10]).

5. Sources of Material

Some systems restrict the source through architecture (e.g., Flickr), while others restrict the source solely through social norms (e.g., CiteULike). Resources to be tagged can be supplied:

a. by the participants (YouTube[11], Flickr, Technorati, Upcoming)

b. by the system (ESP Game[12], Last.fm, Yahoo! Podcasts[13])

c. open to any web resource (Delicious, Yahoo! MyWeb2.0[14])

6. Resource Connectivity

Resources in a tagging system may be connected to each other independently of their tags. For example, Web pages may be connected via hyperlinks, or resources can be assigned to groups (e.g. photo albums in Flickr). Connectivity can be roughly categorized as: linked, grouped, or none.

---

[10] http://www.upcoming.yahoo.com

[11] http://www.youtube.com

[12] http://www.espgame.org

[13] http://podcasts.yahoo.com

[14] http://myweb.yahoo.com

7. Social Connectivity

Users of the system may be connected. Many tagging systems include social networking facilities that allow users to connect themselves to each other based on their areas of interest, educational institutions, location and so forth. Like resource connectivity, the social connectivity could be defined as linked, grouped, or none.

The term **folksonomy** defines a user-generated and distributed classification system, emerging when large communities of users collectively tag resources (Wal, 2005). Hotho et al. (2006a) are defined a folksonomy as follows.

A folksonomy is a quadruple $F := (U; T; I; Y)$, where U, T, I are finite sets of instances of users, tags, and items and Y defines a relation, the tag assignment, between these sets, that is, $Y \subseteq U \times T \times I$.

Folksonomies became popular on the Web with social software applications such as social bookmarking, photo sharing and weblogs. A number of social tagging sites such as Delicious, Flickr, YouTube, CiteULike have become popular. Commonly cited advantages of folksonomies are their flexibility, rapid adaptability, free-for-all collaborative customisation and their serendipity (Mathes, 2004). People can in general use any term as a tag without exactly understanding the meaning of the terms they choose. The power of folksonomies stands in the aggregation of tagged information that one is interested in. This improves social serendipity by enabling social connections and by providing social search and navigation (Quintarelli, 2005). Folksonomy shows a lot of benefits (Peters and Stock, 2007):

- represent an authentic use of language,
- allow multiple interpretations,
- are cheap methods of indexing,
- are the only way to index mass information on the Web,
- are sources for the development of ontologies, thesauri or classification systems,
- give the quality "control" to the masses,
- allow searching and – perhaps even better – browsing,
- recognize neologisms,
- can help to identify communities,
- are sources for collaborative recommender systems,
- make people sensitive to information indexing.

There are two types of folksonomies: broad and narrow folksonomies (Wal, 2005). The **broad** folksonomy, like Delicious, has many people tagging the same object and every person can tag the object with their own tags in their own vocabulary. Thus, in theory there is a great number of tags that all refer to the same object (item), because users might independently use very distinct tags for the same content. The **narrow** folksonomy, which a tool like Flickr represents, provides benefit in tagging objects that are not easily searchable or have no other means of using text to describe or find the object.

The narrow folksonomy is done by one or a few people providing tags that the person uses to get back to that information. The tags, unlike in the broad folksonomy, are singular in nature. The same tag cannot be associated with a single object multiple times; in other words, the creator or publisher of an object is often the person who creates the first tags (unlike in broad folksonomies), and the option to tag may be even restricted to that person. After all, a much smaller number of tags for one and the same object can be identified in a narrow folksonomy. The differences between narrow and broad folksonomies from a graph perspective are depicted in Figure 2.1, where *U* is the set of users, *T* is the set of available tags and *I* is the set of items. The figure also illustrates that narrow folksonomies are a special case of broad folksonomies with the constraint that each item links to exactly one user.

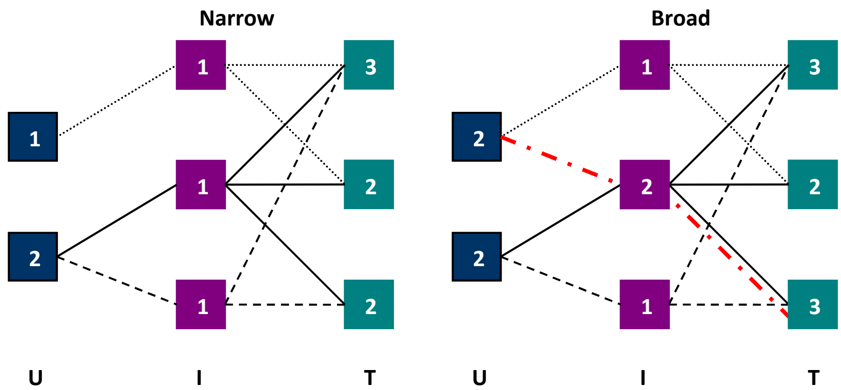

Figure 2.1 The structural difference between narrow and broad folksonomies

2.2. A Model for Tagging Activities

Social tagging systems allow their users to share their tags of particular resources. Each tag serves as a link to additional resources tagged in the same way by other users (Marlow et al., 2006). Certain resources may be linked to each other; at the same time, there may be relationships between users according to their own social interests, so the shared tags of a folksonomy come to interconnect the three groups of protagonists in social labeling systems: Users, Items, and Tags.

Many researchers (Mika, 2005; Halpin et al., 2007; Ciro et al., 2007) suggested a tripartite model that represents the tagging process:

$$Tagging: (U, T, I)$$

where *U* is the set of users who participate in a tagging activity, *T* is the set of available tags and *I* is the set of items being tagged. Figure 2.2 shows a conceptual model for social tagging system where users and items are connected through the tags they assign. In this model, users assign tags to a specific item; tags are represented as typed edges connecting users and items. Items may be connected to each other (e.g., as links between web pages) and users may be associated by a social network, or sets of affiliations (e.g., users that work for the same company).

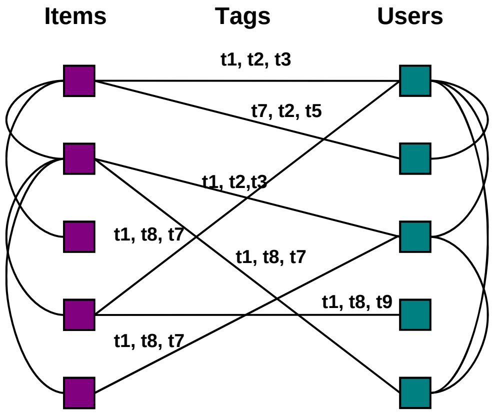

Figure 2.2 Conceptual model of a collaborative tagging system (Marlow et al., 2006)

Examination (Golder and Huberman, 2005) of the collaborative tagging system, such as Delicious, has revealed a rich variety in the ways in which tags are used, regularities in user activity, tag frequencies, and great popularity in bookmarking, as well as a significant stability in the comparative proportions of tags within a given url.

a. Tags may be used to identify the topic of a resource using nouns and proper nouns (i.e. photo, album, photographer).
b. To classify the type of resource (i.e. book, blog, article, review, event).
c. To denote the qualities and characteristics of the item (i.e. funny, useful, cool).
d. A subset of tags, such as myfavourites, mymusic and myphotos reflect a notion of self-reference.
e. Some tags are used by individuals for task organization (e.g. to read, job search, and to print).

Time is an important factor in considering collaborative tagging systems, in fact definitions and relationships among tags could vary over time. For certain users, the number of tags can become stable over time, while for others, it keeps growing. There are three hypotheses about tags behavior over time (Halpin et al., 2006):

a. **Tags convergence**: the tags assigned to a certain Web resource tend to stabilize and to become the majority.
b. **Tags divergence**: tag-sets that don’t converge to a smaller group of more stable tags, and where the tag distribution repeatedly changes.
c. **Tags periodicity**: after one group of users tag some local optimal tag-set, another group uses a divergent set but, after a period of time the new group's set becomes the new local optimal tag-set. This process may repeat and so lead to convergence after a period of instability, or it may act like a chaotic attractor.

2.3. Tag-Based Recommender Systems

Recommender systems in general recommend interesting or personalized information objects to users based on explicit or implicit ratings. Usually, recommender systems predict ratings of objects or suggest a list of new objects that the user hopefully will like the most. The approaches of profiling users with user-item rating matrix and keywords vectors are widely used in recommender systems. However, these approaches are used for describing two-dimensional relationships between users and items. In tag recommender systems the recommendations are, for a given user $u \in U$ and a given resource $r \in R$, a set $\hat{T}(u,r) \subseteq T$ of tags. In many cases, $\hat{T}(u,r)$ is computed by first generating a ranking on the set of tags according to some quality or relevance criterion, from which then the top *n* elements are selected (J¨aschke et al., 2007).

Personalized recommendation is used to conquer the information overload problem, and collaborative filtering recommendation is one of the most successful recommendation techniques to date. However, collaborative filtering recommendation becomes less effective when users have multiple interests, because users have similar taste in one aspect may behave quite different in other aspects. Information got from social tagging websites not only tells what a user likes, but also why he or she likes it. Tagging represents an action of reflection, where the tagger sums up a series of thoughts into one or more summary tags, each of which stands on its own to describe some aspect of the resource based on the tagger's experiences and beliefs (Bateman et al., 2007).

In the remainder of this section, we first describe the proposed extension with integrating tags information to improve recommendation quality. We then present two different ways for collecting tags which can work complementary to collaborative tagging, result to tag collections with improved quality.

2.3.1. Extension with Tags

The current recommender systems are commonly using collaborative filtering techniques, which traditionally exploit only pairs of two-dimensional data. As collaborative tagging is getting more widely used, social tags as a powerful mechanism that reveal three-dimensional correlations between users–tags–items, could also be employed as background knowledge in recommender system.

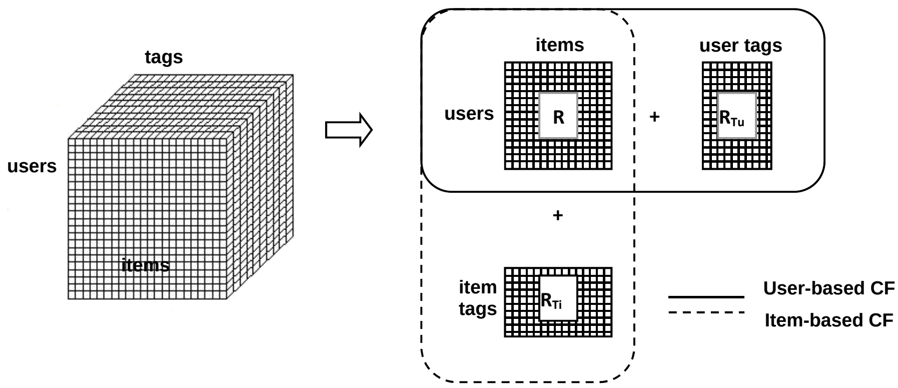

Figure 2.3 Extend user-item matrixes by including user tags as items and item tags as users (Tso-Sutter et al., 2008)

The first adaptation lies in reducing the three-dimensional folksonomy to three two-dimensional contexts: <*user, tag*> and <*item, tag*> and <*user, item*>. This can be done by augmenting the standard user-item matrix horizontally and vertically with user and item tags correspondingly (Tso-Sutter et al., 2008). User tags are tags that user *u* uses to tag items and are viewed as items in the user-item matrix. Item tags, are tags that describe an item *i*, by users and play the role of users in the user-item matrix (See Figure 2.3). Furthermore, instead of viewing each single tag as user or item, clustering methods can be applied to the tags such that similar tags are grouped together.

A tag based recommender system must approach several challenges to be successful in a real world application (Marinho et al., 2011):

- tags should describe the annotated item,
- items should awake the interest of the user,
- suggested items should be interesting and relevant,
- the suggestions should be traceable such that one easily understands why he got the items suggested,
- the suggestions must be delivered timely without delay,
- the suggestions must be easy to access (i.e., by allowing the user to click on them or to use tab-completion when entering tags),
- the system must ensure that recommendations do not obstruct the normal usage of the system.

Recommending tags can serve various purposes, such as: increasing the chances of getting an item annotated, reminding a user what an item is about and consolidating the vocabulary across the users.

2.3.2. Collecting Tags

The quality of tags can directly affect the recommendation process. Collaborative tagging exploits the “wisdom of crowds”. The following alternative ways for collecting tags can work complementary to collaborative tagging, resulting to tag collections with improved quality (Marinho et al., 2011).

Tagging based on experts: An expert is someone who possesses a high level of knowledge in a particular domain (Marinho et al., 2011). This implies that experts provide tags that are objective and cover multiple aspects. The main advantage of using experts is the high-quality of the resulting tags, especially for e-learning systems. An expert should be someone who is able to recognizes the usefulness of an item/document/learning object before the others do, thus becoming the first to assign tags to it and bring it to the attention of other users. Generally speaking, the earlier a user has tagged a document, the more credit he should receive for his actions. This comes, of course, to the cost of manual work, which is both time consuming and expensive.

Tagging based on content: Several items, like URLs, songs, etc., contain a rich content. By crawling associated information from the Web and by converting it into a suitable representation, tags can be collected using data mining algorithms. In the tag recommendation task some of the tags to be predicted in the test set never appeared in the training set, which forced the participants (Srikant et al., 2008) to use the textual content of the items to come up with new tags. The advantage of content-based tags is that no humans must be directly involved during the collection process. The disadvantages are that these tags can be noisy and that their computation is intensive.

Table 2.2 Characterization of tag collection methods (Marinho et al., 2011)

Method	Advantages	Disadvantages
Social tagging	Scalability, social context, “wisdom of crowds”	Polysemy, cold start
Experts	Accurate tags	Costly process, difficult scalability
Content-based	Automation, avoids cold start	Noise, computationally intensive

Compared to the alternative methods, social tagging has the advantage of producing large-scale tag collections. The quality of tags generally improves with a large number of taggers. Nevertheless, social tagging is prone to the cold-start problem, as new resources are seldom tagged. The main advantages and disadvantages of the described approaches are shown in Table 2.1 (Marinho et al., 2011).

Part II

Personalized Recommendation for a Programming Tutoring System

Chapter 3

Recommender Systems in E-learning Environments

RSs strongly depend on the context or domain they operate in, and it is often not possible to take a recommendation strategy (Drachsler et al., 2009) from one context and transfer it to another context or domain. The first challenge for designing a RS is to define the users and purpose of specific context or domain in a proper way (McNee et al., 2006). Learning process includes three components: learners, teachers/instructors, and learning materials. From teacher's point of view, teaching is an activity to deliver information and skill to learners with some goals to be achieved. From learners' point of view, learning is an activity to acquire information from teacher to achieve goals set by the teacher. Learners with their prior knowledge acquire new information from teacher. Here, social constructivism paradigm can help learners learn collaborative and sharing knowledge each other. Basically, knowledge which is needed to be achieved according to the course, mainly does not influenced by how many Learning Objects (LOs) the learners have read, but how relevant are LOs that have retrieved and learned. Learner who has high prior knowledge according to the course is different from other learners who have low prior knowledge.

In a virtual classroom, teachers provide resources such as text, multimedia and simulations, and moderate and animate discussions. Remote learners are encouraged to peruse the resources and participate in activities. However, it is very difficult and time consuming for educators to thoroughly track and assess all the activities performed by all learners on all tools. Moreover, it is hard to evaluate the structure of the course content and its effectiveness on the learning process. Resource providers do their best to structure the content assuming its efficacy (Za¨ıane, 2001). When instructors put together an on-line course, they may compile interactive course notes, simulations, demos, exercises, quizzes, asynchronous forums, chat tools, web resources, etc. This amalgam of on-line hyperlinked material could form a complex structure that is difficult to navigate. Hence, personalization features are needed which adaptively facilitate learner in monitoring their learning progress and provide any resources or learning material that suitable to what they need.

Personalized recommendation approaches are first proposed in Ecommerce area for product purchase (Balabanović and Shoham, 1997; Resnick and Varian 1997), which help consumers find products they would like to purchase by creating a list of recommended products for each given consumer (Cheung et al., 2003; Schafer et al., 2001). Literature review shows there are also many researchers have attempted to adopt recommender systems to e-learning environments. For example, Shen et al. (2005) described a mechanism focused on how to organize the learning materials based on domain ontology which can guide the learning resources recommendation according to learning status. A multi-attribute assessment method is proposed in (Lu, 2004) to justify a learner's need and deployed a fuzzy matching method to find suitable learning contents to best perform each learner need. Research paper (Luo et al., 2002) presented a method to organize components and courseware using the hierarchy and association rules of the concepts, which can recommend the relative contents to learners and also can help them to control the learning schedule. However, most of these methods missing one important issue in e-learning RS, that is, the natural learning behavior is not lonely but interactive which relying on friends, classmates, lecturers, and other sources to make the choices for learning. Designers and instructors, when devising the on-line structure of the course and course material, have a navigation pattern in mind and assume all on-line learners would follow a consistent path; the path put forth in the design and materialized by some hyperlinks. Learners, however, could follow different paths generating a variety of sequences of learning activities. Often this sequence is not the optimum sequence, and probably not the sequence intended by the designer. Instructors are in desperate need for non-intrusive and automatic ways to get objective feedback from learners in order to better follow the learning process and appraise the on-line course structure effectiveness. On the learner's side, it would be very useful if the system could automatically guide the learner's activities and intelligently recommend on-line activities or resources that would favour and improve the learning. The automatic recommendation could be based on the teacher's intended sequence of navigation in the course material, or, more interestingly, based on navigation patterns of other successful learners. For example, during the learning process, a learner read a useful material, summarized what he/she has learned or got the answer of a typical question, some learners with similar learning status are likely need these resources. The aim of the thesis research is to analyse capabilities of integration such a recommender system based on collaborative tagging techniques into (Java) programming tutoring system.

The rest of this chapter is organized as follows. The most important challenges for designing a recommender system in e-learning environments are presented in Section 3.1. Section 3.2 presents a survey of the state-of-the-art in recommendation techniques for RS in e-learning environments. Applying tag-based RS to e-learning environments and suitable recommendation techniques via collaborative tags of learners are described in Section 3.4. The limitations of current folksonomy and posible solutions are given in Section 3.5.

3.1. The Most Important Requirements and Challenges for Designing a Recommender System in E-learning Environments

A RS in e-learning environments utilizes information about learners and learning activities (LA) and recommend items such as papers, web pages, courses, lessons and other learning resources which meet the pedagogical characteristics and interests of learners (Drachsler et al., 2008). Such a RS could provide recommendations to online learning materials or shortcuts. Those recommendations are based on previous learners' activities or on the learning styles of the learners that are discovered from their navigation patterns. To design an effective RS in e-learning environments, it is important to understand specific learners' characteristics (Garcia et al., 2009; Drachsler et al., 2008):

1. learning goal,
2. prior knowledge,
3. learner characteristics,
4. learner grouping,
5. rated learning activities (LAs),
6. learning paths, and
7. learning strategies, desired in a RS.

E-learning systems should be able to recognize and exploit these learners' characteristics serve as guidelines for framework design and platform implementation for a good RS for e-learning (Angehrn et al., 2001; Zaïane, 2001; Savidis et al., 2006).

- *A good RS should be highly personalized.* Relevant learning materials should be chosen and presented to learners or researchers based on learner's learning style, interests, preferences, current activities, etc.
- *A good RS should recommend materials at the appropriate time and location*. A good RS should deliver relevant learning materials to learner at the most appropriate time and locations to facilitate learners' acquisition of knowledge and skills.
- *A good RS should support non-disruptive view of experience*. Non-disruptive means that learners have the option to either follow or discount relevant materials based on their learning needs.
- *A good RS should be socially situated*. A good RS should be able to recognize and exploit the learners' social networks, role models, levels of trust and influence, etc. RS should also help the learners to recognize their knowledge acquisition process in the context of the group.
- *A good RS should include the adoption phase*. A good RS should be able to monitor, understand and model the different phases of adoption of the knowledge by the learner. In particular it includes the phases in which the new concepts are experimented with, evaluated, internalized and finally applied.

- *A good RS should support the continuous learning process*. A good RS should support just-in-time learning, by better analyzing their current and future activities. Also it should provide motivational support and stimulation.
- *A good RS should provide high level of interactivity*. A good RS should provide very active, cognitive and diverse mode of interaction with the learner in the form of a rich choice of interaction strategies.
- *A good RS should provide appropriate course materials according to learners' learning style*. Each person learns differently and needs to develop his/her own learning skills in his/her own way. Learners have different backgrounds, strengths and weaknesses, interests, ambitions, senses of responsibility, levels of motivation, and approaches to studying and learning. For example, different learners prefer different presentation forms: some prefer multimedia contents (simulations, presentations, graphical material and hypertext documents); while others prefer traditional web pages (questionnaires, exercises, research studies).

3.2. Recommendation Techniques for RS in E-learning Environments - A Survey of the State-of-the-Art

E-learning system uses different recommendation techniques in order to suggest online learning activities to learners, based on their preferences, knowledge and the browsing history of other learners with similar characteristics. RSs assist the natural process of relying on friends, classmates, lecturers, and other sources of make the choices for learning (Lu, 2004). In the educational setting, these recommendation systems can be classified according to their field of application or focus (Romero and Ventura, 2006):

1) learner-centered (Gaudioso et al., 2003; Zaiane, 2002), in order to suggest good learning experiences for the learners in accordance with their preferences, needs and level of knowledge; and
2) teacher-centered, with the aim of helping the teachers and/or authors of the e-learning systems to improve the functions or performance of these systems based on learner information (Chen et al., 2002; Romero et al., 2003). Some others examples of educational applications of these systems are: obtaining more feedback about teaching; finding out more about how learners learn on the Web; evaluating learners in terms of their browsing patterns; classifying learners into groups; or restructuring the contents of the website in order to personalize the course.

Each recommendation strategy has its own strengths and weaknesses. According to set of the most important requirements for a good RS in e-learning environment, have been explored and defined in the previous section, in the remainder of this section we present a survey of the state-of-the-art in RS for e-learning environment. We identify challenges and various limitations for each traditional recommendation method, then consider some tag-based profiling approaches for extending their capabilities.

3.2.1. Collaborative Filtering Approach

Collaborative systems track past actions of a group of learners to make a recommendation for individual members of the group (Tan et al., 2008). Based on the assumption that learners with similar past behaviors (rating, browsing, or learning path) have similar interests, a collaborative filtering system recommends learning objects the neighbors of the given learner have liked.

This approach relies on a historic record of all learner interests such as can be inferred from their ratings of the items (learning objects/learning actions) on a website. Rating can be explicit (explicit ratings or customer satisfaction questionnaires) or implicit (from the studying patterns or click-stream behavior of the learners). The proportion of actual studying hours to the total hours of the course is recorded as the implicit rating scores, and transformed to corresponding explicit rating scores, from 1 to 5. The learners' rating scores can be given in a m*n matrix, as it is shown in table 3.1, where $L = \{l_1, l_2, \ldots, l_m\}$ is a list of *m* learners, $O = \{o_1, o_2, \ldots, o_n\}$ is the list of *n* learning objects, and $R_{j,k}$ gives the rating of object O_k, given by learner j. Also, it can be rating of object o_k given by intelligent tutoring system for learner *j*. There exists a distinguished learner $l_a \in L$ called the active learner for whom the task of collaborative filtering algorithm is to find learning object likeliness.

Table 3.1 Learner's rating matrix

	O_1	…	O_k	…	O_n
l_1	$R_{1,1}$	…	$R_{1,k}$	…	$R_{1,n}$
…					
l_1	$R_{j,1}$	…	$R_{j,k}$	…	$R_{j,n}$
…					
l_1	$R_{m,1}$	…	$R_{m,k}$	…	$R_{m,n}$

The Neighborhood formation scheme usually uses Pearson correlation or cosine similarity as a measure of proximity (Shardanand and Maes, 1995; Resnick et al., 1997).

An exploratory study of a recommender system, using collaborative filtering to support (virtual) learners in a learning network, has been reported in (Koper, 2004). The author simulated rules for increasing/decreasing motivation and some other disturbance factors in learning networks, using the Netlogo tool. Closely related to this study is an experiment reported in (Janssen et al., 2007). The authors offered learners a similar recommendation system. The recommendations did not take personal characteristics of learners (or possible 'matching errors') into account. Another system implemented by Soonthornphisaj et al. (2006) allows all learners to collaborate their expertise in order to predict the most suitable learning materials to each learner. This smart e-learning system applies the collaborative filtering approach that has an ability to predict the most suitable documents to the learner. All learners have the chance to introduce new material by uploading the documents to the server or pointing out the Web link from the Internet and rate the currently available materials.

One of the first attempts to develop a collaborative filtering system for learning resources has been the Altered Vista (AV) system (Recker and Walker, 2003; Recker et al., 2003; Walker et al., 2004). The AV system (Walker et al., 2004) uses a database in which learner evaluations of learning resources are stored. Learners can browse the reviews of others and can get personalized learning resource recommendations from the system. AV does not aim to support learners directly by giving them feedback on their work. Instead, AV provides an indirect learning support in which suitable learning tools are recommended. The team working on AV explored several relevant issues, such as the development of non-authoritative metadata to store learner-provided evaluations (Recker and Walker, 2003), the design of the system and the review scheme it uses (Walker et al., 2004), as well as results from pilot and empirical studies from using the system to recommend to the members of a community both interesting resources and people with similar tastes and beliefs. A survey-based evaluation of AV showed a predominant positive feedback, but also identified issues with the system's incentive and with regard to privacy (Walker et al., 2004).

Another system of the educational collaborative filtering applications is the Web-based PeerGrader (PG) (Gehringer, 2001; Lynch et al., 2006). The purpose of this tool is to help learners improve their skills by reviewing and evaluating solutions of their fellow learners blindly. PG works in the following way: first, the learners get a task list and each learner chooses a task. Next, the learners submit their solutions to the system, where they are read by another learner who then provides feedback in form of textual comments. After that, the authors modify their solutions based on the comments they have received, and re-submit their modified solutions again to the system, where they will be reviewed by other learners. Then, the solutions' authors grade each review with respect to whether it was helpful or not. Finally, the system calculates grades for all learner solutions. One of PG's strengths is to provide learners with high-quality feedback also in ill-defined homework tasks that do not have clear-cut gold standard solutions (such as design problems). This kind of feedback could not be generated automatically. A disadvantage is the time required for the system to work effectively: due to the complexity of the reviewing process and the textual comments, the evaluation of a single learner answer is very time consuming. This may cause learner drop-outs and deadline problems (Lynch et al., 2006). Also, studies with PG revealed problems with getting feedback of high quality. An evaluation of subjective usefulness showed that the system was appreciated by its users (Lynch et al., 2006), yet a systematic comparison of PG scores to expert grades has not been conducted.

A newer web-based collaborative filtering system, the Scaffolded Writing and Rewriting in the Discipline (SWoRD) system (Cho and Schunn, 2007; Cho et al., 2006) addresses the problem of writing homework in the form of a long text, which cannot be reviewed in detail by a teacher for time reasons. Because of this, learners do often not receive any detailed feedback on their solutions at all. Having such feedback would be beneficial for learners though, since they could use it to improve their future work. To address this problem, SWoRD relies on peer reviews and implements an algorithm that follows the typical journal publication and reviewing process. An evaluation showed that the participants benefitted from multi-peers' feedback more than from single-peer's or single expert's feedback (Cho and Schunn, 2007).

A different approach is used by the LARGO system (Pinkwart et al., 2006), where learners create graphs of US Supreme Court oral arguments. Within LARGO, collaborative scoring is employed to assess the quality of a "decision rule" that a learner has included in his diagram. Since this assessment involves interpretation of legal argument in textual form, it cannot be automated reasonably. While the overall LARGO system has been tested in law schools and shown to help lower-aptitude learners (Pinkwart et al., 2007), empirical studies to test the educational effectiveness of the specific collaborative scoring components have not been conducted.

Rule-Applying Collaborative Filtering (RACOFI) Composer system (Anderson et al., 2003; Lemire, 2005; Lemire et al., 2005) combines two recommendation approaches by integrating a collaborative filtering engine, that works with ratings that learners provide for learning resources, with an inference rule engine that is mining association rules between the learning resources and using them for recommendation. RACOFI studies have not yet assessed the pedagogical value of the recommender, nor do they report some evaluation of the system by learners.

Manouselis et al. (2007) tried a typical, neighborhood-based set of collaborative filtering algorithms in order to support learning object recommendation. The examined algorithms have been multiattribute ones, allowing the recommendation service to consider multi-dimensional ratings that learners provide on learning resources. The performance of the same algorithms is changing, depending on the context where testing takes place. The results from the comparative study of the same algorithms in an e-commerce and a e-learning setting Manouselis et al. (2007) have led to the selection of different algorithms from the same set of candidate ones.

In summary, the relatively few educational technology systems with collaborative filtering components all have an underlying algorithm to determine solution quality based on collaborative scoring. Yet, existing systems are often specialized for a particular application area such as legal argumentation (LARGO), writing skills training (SWoRD), or educational resource recommendation (AV), or they involve a rather complicated and longterm review process (SWoRD, PG).

The CF-based techniques, in general, suffer from several limitations. Two serious limitations with quality evaluation are: the sparsity problem and the cold start problem (Lu, 2004). The sparsity problem occurs when available data is insufficient for identifying similar learners or items (neighbors) due to an immense amount of learners and items (Sarwar et al., 2001). It is difficult for collaborative filtering based recommender systems to precisely compute the neighborhood and identify the learning objects to be recommended even though learners are very active, each individual has only expressed a rating on a very small portion of the items (Linden et al., 2003). Also, an severe problem is the cold start problem (first-rater), which occurs when a new learner/learner object is introduced and thus has no previous ratings information available (Massa and Avesani, 2004). With this situation, the system is generally unable to make high quality recommendations.

The CF-based techniques rely heavily on explicit learner input (e.g., previous customers' rating/ranking of products), which is either unavailable or considered intrusive. With sparsity of such learner input, the recommendation precision and quality drop significantly. This is because without good and trusted ratings entered by the learners, recommendations become useless and untrustworthy. To recommend learning activities or learning objects it is better to use real past activities (history logs) by learners as input for their profiles. Also, in the case of intelligent tutoring system, collaborative filtering

(CF) approach can be carried out according to ratings (grades) for learners' knowledge level, provided by the tutoring system.

3.2.2. Content-based Techniques

Content-based techniques recommend items (learning objects/learning actions) similar to the ones the learners preferred in the past. They base their recommendations on individual information and ignore contributions from other learners (Billsus and Pazzani, 1998). In content-based systems, items are described by a common set of attributes. Learner's preferences are predicted by considering the association between the item ratings and the corresponding item attributes. Therefore, learner can receive proper recommendations without help from other learners. Content-based techniques can be classified into two different categories (Schmitt and Bergmann, 1999; Aguzzoli et al., 2002; Wilson et al., 2003):

1. Case based reasoning (CBR) techniques and
2. Attribute – based techniques

Case based reasoning techniques recommend items with the highest correlation to items the learner liked before. Case-based reasoning is useful to keep the learner informed about aimed learning goals. These techniques are domain-independent, do not require content analysis and the quality of the recommendation improves over time when the learners have rated more items. The disadvantage of the new learner problem also states to case-based reasoning techniques. Nevertheless specific disadvantages of case-based reasoning are overspecialization and sparsity, because only items that are highly correlated with the learner profile or interest can be recommended. Through case-based reasoning the learner is limited to a set of items that are similar to the items he already knows (Adomavicius and Tuzhilin, 2005).

Recent research papers present different facets of CBR in teaching or learning help. Pixed (Project Integrating eXperience in Distance Learning), which is an adaptive hypermedia ontology-based system implements case based reasoning method (Heraud, 2004). The Pixed approach assumes positions of a learner as a kind of expert of her/his own learning skills, or at least as a real practitioner of his own practices. The learner builds her/his knowledge by interacting with the learning environment, trying to benefit as much as possible from the available educational activities. Learning is considered as a problem-solving task. The goal is to learn a specific concept proposed in the domain knowledge ontology. The way to reach this goal is one particular path among the different available educational activities linked to that ontology. Sormo and Amodt (2002) propose building "a cognitive model of how humans solve problems in the domain and use this model in attempting to solve the problem, both from the point of view of the current learner (using the learner model) and of an expert (represented by an expert model)". The case-based reasoner has to evaluate the learner's solution and to explain why s/he does or does not fit the observed features of the problem. Funk and Conlan (2003) make research more closely related to Pixed. Their goal is the same: to use learner feedback in order to adapt the learning environment. The learner feedback can be exploited in two ways: direct feedback exploitation during the learning process, in the form of learners' comments, and feedback exploitation by authors and tutors after the learning process in order to integrate it into the proposed courses, by comparing the learners' result with the result of other cases. The authors associate CBR with filtering techniques by attempting to create learner profiles taking into account different feedbacks. Elorriaga and Fernandez-Castro (2000) propose to use CBR to deploy an instructional planner which adapts the sequences observed in logs in order to create instructional sequences for a complete course. In (Heraud et al., 2004), a case - based reasoning system was developed to offer navigational guidance to the learner. It is based on past user's interaction logs and it includes a model describing learning sessions.

Attribute – based techniques recommend items based on the matching of their attributes to the learner profile. Attributes could be weighted for their importance to learner. Adding new LA or learners to the network will not cause any problem. Attribute-based techniques are sensitive to changes in the profiles of the learners (Drachsler et al., 2008). They can always control the personalized RS by changing their profile or the relative weight of the attributes. A description of needs in their profile is mapped directly to available LA. A serious disadvantage is that an attribute-based recommendation is static and not able to learn from the network behavior. That is the reason why highly personalized recommendation can not be achieved. Attribute-based techniques work only with information that can be described in categories. Media types, like audio and video, first need to be classified to the topics in the profile of the learner. This requires category modeling and maintenance which could raise serious limitations for learning environments. Also the overspecialization can be a problem, especially if learners do not change their profile. Attribute-based recommendations are useful to handle the 'cold-start' problem because no behavior data about the learners is needed. Attribute-based techniques can directly map characteristics of learners (like learning goal, prior knowledge, and available study time) to characteristics of LA (Drachsler et al., 2007). There are several applications that tackle attribute – based techniques problems such as prediction and visualization. Attribute–based Ant Colony System (AACS) (Yang and Wu, 2009) uses a method of finding learning objects that would be suitable for a learner based on the most frequent learning trails followed by the previous learners. The system updates the trails pheromones from different knowledge levels and different styles of learners to create a powerful and dynamic learning object search mechanism. There are three prerequisites for achieving this:

a) the adaptive learning portal knows the learner's attributes which include the learner's knowledge level and learning style
b) the learner's attributes and learning object's attributes which have been annotated by teacher or content providers
c) matching the relationships between learners and learning object.

3.2.3. Association Rule Mining

Association rule mining techniques (Agrawal and Srikant, 1995) are one of the most popular ways of representing discovered knowledge and describe a close correlation between frequent items in a database. An association rule consists of an antecedent (left-hand side) and a consequent (right-hand side). The intersection between the antecedent and the consequent is empty. An:

$$X \Rightarrow Y$$

type association rule expresses a close correlation between items (attribute-value) in a database (Zheng et al., 2001). Most association rule mining algorithms require the user to set at least two thresholds, one of minimum support and the other of minimum confidence. The support *S* of a rule is defined as the probability that an entry has of satisfying both *X* and *Y*. Confidence is defined as the probability an entry has of satisfying *Y* when it satisfies *X*. Therefore the aim is to find all the association rules that satisfy certain minimum support and confidence restrictions, with parameters specified by the user. Therefore, the user must have a certain amount of expertise in order to find the right support and confidence settings to achieve the best rules.

Association rule mining has been applied to e-learning systems aims to intelligently recommend on-line learning activities to learners based on the actions of previous learners to improve course content navigation as well as to assist the on-line learning process (García et al., 2007).

Count the learners’ browsing records, learning path and testing grades and finding out the connection between learning objects, association rule can be used to calculate the learning profiles of the coming learners and perform the following tasks:

- building recommender agents for on-line learning activities or shortcuts (Zaiane, 2002),
- automatically leading the learner’s activities and intelligently recommend on-line learning activities or shortcuts in the course web site to the learners (Lu, 2004),
- identifying attributes of performance inconsistency between various groups of learners (Minaei-Bidgoli et al., 2004),
- discovering interesting learner’s usage information in order to provide feedback to course author (Romero et al., 2004),
- finding out the relation among the learning materials from a large amount of material data (Yu et al., 2001),
- finding learners’ mistakes that are often occur together (Merceron and Yacef, 2004),
- optimizing the content of an e-learning portal by determining the content of most interest to the learner (Ramli, 2005),
- deriving useful patterns to help educators and instructors evaluating and interpreting on-line course activities (Zaiane, 2002), and
- personalizing e-learning based on comprehensive usage profiles and a domain ontology (Markellou et al., 2005).

Most of the subjective approaches involve learner participation in order to express, in accordance with his or her previous knowledge, which rules are of interest. Hence, subjective measures are becoming increasingly important (Silberschatz and Tuzhilin, 1996). Some suggested subjective measures (Liu et al., 2000) are:

- *Unexpectedness*: Rules are interesting if they are unknown to the learner or contradict the learner’s knowledge.

- *Actionability*: Rules are interesting if learners can do something with them to their advantage.

There are several specific researches about the application association rule mining and recommender systems in e-learning systems. Association rules for classification applied to e-learning (Castro et al., 2007), have been investigated in the areas of learning recommendation systems (Chu et al., 2003; Zaiane, 2002), learning material organization (Tsai et al., 2001), learner learning assessments (Hwang et al., 2003; Kumar, 2005; Matsui and Okamoto, 2003; Resende and Pires, 2001), course adaptation to the learners' behavior (Hsu et al., 2003; Markellou et al., 2005), and evaluation of educational web sites (Dos Santos and Becker, 2003)

Wang et al. (2002) develops a portfolio analysis tool based on associative material clusters and sequences among them. This knowledge allows teachers to study the dynamic browsing structure and to identify interesting or unexpected learning patterns. Minaei-Bidgolie et al. (2004) propose mining interesting contrast rules for Web-based education systems. Contrast rules help one to identify attributes characterizing patterns of performance difference between various groups of learners. Markellou et al. (2005) propose an ontology-based framework and discover association rules, using the Apriori algorithm. The role of the ontology is to determine which learning materials are more suitable to be recommended to the learner. Li and Zaïane (2004) use recommender agents for recommending online learning activities or shortcuts in a course web site based on a learner's access history. Romero et al. (2004) propose to use grammar-based genetic programming with multi-objective optimization techniques for discovering useful association rules from learner's usage information. Merceron and Yacef (2004) use association rule and symbolic data analysis, as well as traditional SQL queries to mining learner data captured from a web-based tutoring tool. Their goal is to find mistakes that often occur together. Freyberger et al. (2004) use association rules to determine what operation to perform on the transfer model that predicts a learner's success.

Apriori algorithm (Agrawal et al., 1993) is a prominent algorithm for mining frequent itemsets for Boolean association rules. In Apriori algorithm, it is time-consuming that the database has been scanned for many times. Therefore, many algorithms, like the DIC algorithm (Brin et al., 1997), DHP algorithm (Park et al., 1995) and AprioriTid algorithm (Agrawal et al., 1993), etc., are proposed successively to improve the performance.

Association rule mining and frequent pattern mining were applied in (Zaïane, 2001) to extract useful patterns that might help teacher, educational managers, and Web masters to evaluate and understand on-line course activities. A similar approach can be found in (Minaei-Bidgoli et al., 2004), where distinguish rules, defined as sets of conjunctive rules describing patterns of performance difference between groups of learners, were used. A computer-assisted approach to diagnosing learner learning problems in science courses and offer learners advice was presented in (Hwang et al., 2003), based on the concept effect relationship (CER) model, a specification of the association rules technique.

A hypermedia learning environment with a tutorial component was described in (Costabile et al., 2005). It is called Logiocando and targets children of the fourth level of primary school (9-10 years old). It includes a tutor module, based on if-then rules, that emulates the teacher by providing suggestions on how and what to study. In Matsui and Okamoto (2003) it can be found the description of a learning process assessment method that resorts to association rules, and the well-known ID3 DT learning method. A framework for the employ of Web usage mining to support the validation of learning site designs was defined in (Dos Santos and Becker, 2003), applying association and sequence techniques (Srivastava et al., 2000)

In Markellou et al. (2005), a framework for personalized e-learning based on aggregate usage profiles and domain ontology were presented, and a combination of Semantic Web and Web mining methods was used. The Apriori algorithm for association rules was applied to capture relations among URL references based on the navigational patterns of learners. A test result feedback (TRF) model that analyzes the relationships between learner learning time and the corresponding test results was introduced in Hsu et al. (2003). The objective was twofold: on the one hand, developing a tool for supporting the tutor in reorganizing the course material; on the other, a personalization of the course tailored to the individual learner needs. The approach was based on association rules mining. A rule-based mechanism for the adaptive generation of problems in Intelligent Tutoring System (ITS) in the context of Web-based programming tutors was proposed in (Kumar, 2005). In (Hwang et al., 2003), a Web-based course recommendation system, used to provide learners with suggestions when having trouble in choosing courses, was described. The approach integrates the Apriori algorithm with graph theory.

Some of the main drawbacks of association rule algorithms are (Garcia et al., 2007):

- association rule mining algorithms normally discover a huge quantity of rules and do not guarantee that all the rules found are relevant,
- the used algorithms have too many parameters for somebody non expert in data mining and
- the obtained rules are far too many, most of them non-interesting and with low comprehensibility.

In order to provide better recommendations, and to be able to use recommender systems in more complex types of e-learning, most of the methods reviewed in this subsection would need significant extensions.

In the remainder of this chapter, we analyze a new approach which improves the understanding of learners, incorporating the tag information into the recommendation process. We first describe the proposed features of collaborative tagging that are generally attributed to their success in e-learning. We then present several different algorithms recommendation algorithms for developing tag-based recommender systems which are suitable for e-learning environments. The FolkRank algorithm, developed as a folksonomy search engine by using the graph model, is reported in Section 3.3.1. Probabilistic latent semantic analysis (PLSA), as a novel statistical technique for the analysis of two-mode and co-occurrence data, is described in Section 3.3.2. Section 3.3.3 reviews a method for tag-based profile construction with collaborative filtering based on collaborative tagging. 3.3.4. Tensor Factorization Technique for Tag Recommendation is shown in section 3.3.4.

3.3. Applying Tag-Based Recommender Systems to E-learning Environments

Collaborative tagging systems have grown in popularity over the Web in the last years on account of their ability to categorize and recover content using open-ended tags (Godoy and Amandi, 2008). The increasing number of users providing information about themselves through collaborative tagging activities caused the appearance of tag-based profiling approaches, which assume that users expose their preferences for certain contents through tag assignments. Thus, the tags could be interesting and useful information to enhance recommender system's algorithms. Tag-based recommender systems (Klašnja-Milićević et al., 2010) analyze tags, discover preferences of a given user and provide suggestions for the user which items could be interesting. The main advantage of the tag-based recommenders is that user preferences and interests are expressed by used tags of the given person. Therefore, these recommenders provide more accurate and personalized recommendations. On the other hand, majority of the tag-based recommenders consider only textual (syntactical) similarities among tags. It causes problems when there are tag synonyms and according to the syntactical similarity these relations will not be revealed. The similar problem can occur when a given tag has more different meanings – so called polysemy. These issues are handled by various techniques which extend standard tag-based recommenders and provide semantically more accurate recommendations (as analyzed in the section 3.3.4). In this chapter we investigate the suitability of tag-based recommender systems into a new context: e-learning. The innovation with respect to the e-learning system lies in their ability to support learners in their own learning path by recommending tags and learning items, and also their ability to promote the learning performance of individual learners (Manouselis et al., 2011).

Using tags enables useful item organization and browsing techniques, such as "pivot browsing" (Millen et al., 2006), which provides a simple and effective method for discovering new and relevant items. Learners could benefit from writing tags in two important ways: first, tagging is proven to be a meta-cognitive strategy that involves learners in active learning and engages them with more effectively in the learning process. As summarized by Bonifazi et al. (2002), tags could help learners to remember better by highlighting the most significant part of a text, could encourage learners to think when they add more ideas to what they are reading, and could help learners to clarify and make sense of the learning content while they try to reshape the information. Learners' tags could create an important trail for other learners to follow by recording their thoughts about specific tutorial resource and could give more comprehensible recommendation about the resources. While the viewing of tags used on a webpage can give a learner some idea of its importance and its content, it falls short of supporting a learner in finding the exact point of interest within the page. The following features of collaborative tagging are generally attributed to their success in e-learning (Bateman et al., 2007; Doush, 2011; Dahl and Vossen, 2008):

1. The information provided by tags makes available insight on learner's comprehension and activity, which is useful for both educators and administrators.
2. Collaborative tagging has potential to further enhance peer interactions and peer awareness centered on learning content.

3. Tagging, by its very nature, is a reflective practice which can give learners an opportunity to summarize new ideas, while receiving peer support through viewing other learners' tags/tag suggestions.
4. In e-learning there is a lack of the social cues that inform instructors about the understanding of new concepts by their learners. Collaborative tags, created by learners to categorize learning contents, would allow instructors to reflect at different levels on their learners' progress. Tags could be examined at the individual level to examine the understanding of a learner (e.g. tags that are out of context could represent a misconception), while tags examined at the group level could identify the overall progress of the class. Working with instructors of online courses employing tagging would help shed light on the perceived benefits of reflection based on tags.
5. Tagging provides possible solutions for learners' engagement in a number of different annotation activities - add comments, corrections, links, or shared discussion. E-learning systems currently lack sufficient support for self-organization and annotation of learning content (Bateman, 2007). However, walk through a university campus we can see learners engaged in a number of annotation activities. These include writing notes, creating marginalia in books, highlighting text, creating dog ears on pages or bookmarking pages. During lectures as many as 99% of learners take notes (Palmatier and Bennet, 1974), and 94% of learners at the post-secondary level believe that note-taking is an important educational activity (Wiley, 2000). In this sense tagging is beneficial to note-taking, since tags represent an aspect or cue to be used in the tagger's recall process.

Traditionally, e-learning systems intend to provide direct customized instruction to learners by finding the mismatches between the knowledge of the expert and the actions that reflect the assimilation of that knowledge by the learner (Santos and Boticario, 2008). Their main limitations are:

1. e-learning are specific of the domain for which they have been designed (since they have to be provided with the expert knowledge) and
2. it is unrealistic to think that it is possible to code in a system all the possible responses to cover the specific needs of each learner at any situation of the course.

In this sense, a dynamic support that recommends learners what to do to achieve their learning goals is desirable. Also, such systems should have capability to find appropriate content on the Web, and capability to personalize and adjust this content based on the system's examination of its learners and the collected tags given by the learners and domain experts.

3.3.1. FolkRank Algorithm

The FolkRank algorithm has been inspired from the PageRank algorithm which exploits the network structures of Web pages. The PageRank algorithm assumes that a hyperlink from one page to another is a vote from the former to the latter (Brin and Page, 1998). The more votes a page receives, the more important that page is assumed to be.

This idea is similar to an item which is tagged with important tags by important learners becomes important itself. For example, one definition/example/task could be tagged with important tags by important learner with high knowledge level. Such definition/example/task may be considered as an important definition/example/task. The same holds, symmetrically, for tags and learners. The distribution of weights can thus be described as the fixed point of a weight passing scheme on the web graph.

The hyperlinks indicate how important a learning object is. Tags, though, incorporate more information than does a simple hyperlink, which represents a learner created textual description of a LO. Thus, intuition would suggest that additional information can be harnessed in some way to create better search results. That tags can provide useful information for new statistical approaches which take into account human-based voting and knowledge, using algorithms similar to PageRank.

The FolkRank algorithm adopted the same weight spreading approaches as in the PageRank. The main difference, however, lies in the graph (Hotho et al., 2006a, b, c). In the FolkRank, the graph of tags has no direction, while the PageRank uses directed graphs.

Folksonomy-Adapted PageRank. The FolkRank algorithm transforms the hypergraph formed by the traditional tag assignments into an undirected, weighted tripartite graph $G_F = (V_F, E_F)$, which serves as input for an adaption of PageRank (Page et al., 1998). At this, the set of nodes is $V_F = L \cup T \cup I$ and the set of edges is given via $E_F = \{\{l,t\},\{t,i\},\{l,i\} \mid (l,t,i) \in Y\}$. The weight ω of each edge is determined according to its frequency within the set of tag assignments, i.e. $\omega(l,t) = |\{i \in I : (l,t,i) \in Y\}|$ is the number of items the learner *l* tagged with keyword *t*.

Accordingly, $\omega(t,i)$ counts the number of learners who annotated item *i* with tag *t*, and $\omega(l,i)$ determines the number of tags a learner *l* assigned to an item *i*. With G_F represented by the real matrix *A*, which is obtained from the adjacency matrix by normalizing each row to have a sum equal to 1, and starting with any vector $\vec{\omega}$ of non-negative reals, adapted PageRank iterates as $\vec{\omega} \leftarrow dA\vec{\omega} + (1-d)\vec{p}$.

Adapted PageRank utilizes vector $\vec{p}$, used to express learner preferences by giving a higher weight to the components which represent the learner's preferred web pages, fulfilling the condition $\|\vec{\omega}\|_1 = \|\vec{p}\|_1$. Its influence can be adjusted by d $\in$ [0; 1]. Based on this, FolkRank is defined as follows.

The FolkRank algorithm computes a topic specific ranking in folksonomies: If $\vec{p}$ specifies the preference in a topic (e.g. preference for a given tag), $\vec{\omega}_0$ is the result of applying the adapted PageRank with d = 1 and $\vec{\omega}_1$ is the result of applying the adapted PageRank with some d < 1, then $\vec{\omega} = \vec{\omega}_1 - \vec{\omega}_0$ is the final weight vector. $\vec{\omega}[x]$ denotes the FolkRank of $x \in V$ (Hotho et al., 2006a, b, c).

FolkRank yields a set of related learners and items for a given tag. Following these observations, FolkRank can be used to generate recommendations within a folksonomy system. These recommendations can be presented to the learner at different points in the usage of a folksonomy system (Hotho et al., 2006a, b, c):

- Learning objects that are of potential interest to a learner can be suggested to him. This kind of recommendation increases the chance that a learner finds useful items that he did not even know existed by "serendipitous" browsing.
- When using a certain tag, other related tags can be suggested. This can be used, for instance, to speed up the consolidation of different terminologies and thus facilitate the emergence of a common vocabulary.
- While folksonomy tools already use simple techniques for tag recommendations, FolkRank additionally considers the tagging behavior of other learners.
- Other learners that work on related topics can be made explicit, improving thus the knowledge transfer within organizations and fostering the formation of communities.

FolkRank is robust against online updates since it does not need to be trained every time a new learner, item or tag enters the system. However, FolkRank is computationally expensive and not trivially scalable. It is more suitable for systems where real-time recommendations are not a requirement. Hotho et al. (2006a) investigated FolkRank ranking in contrast to the Adapted PageRank. Results present that the Adapted PageRank ranking contains many globally frequent tags, while the FolkRank ranking provides more personal tags. While the differential nature of the FolkRank algorithm usually pushes down the globally frequent tags such as "web", though, this happens in a distinguished manner: FolkRank will keep them in the top positions, if they are indeed relevant to the learner under consideration.

3.3.2. PLSA

Probabilistic latent semantic analysis (PLSA) is a useful statistical technique for the analysis of two-mode and co-occurrence data, which has applications in information retrieval and filtering, natural language processing, machine learning from text, and in related areas. PLSA has been shown to improve the quality of collaborative filtering based recommenders (Hofmann, 1999) by assuming an underlying lower dimensional latent topic model.

Web users show different types of behavior depending on their information needs and their intended tasks. These tasks are captured implicitly by a collection of actions taken by users during their visits to a site. For example, in a dynamic e-learning Web site, user tasks may be reflected by sequences of interactions with application to browse course information, to register for courses, to read a tutorial, to study an example or to solve a test. The identification of intended learner tasks can shed light on various types of learner navigational behaviors. There may be many learner groups with different (but overlapping) behavior types. These may include learners who engage in reading content by browsing through a variety of learning objects in different categories; learners who are goal-oriented showing interest in a specific category; or learners who prefer to go through the course step by step, in a linear way with each step following logically from the previous one, or learners who tend to learn in large leaps. Most current Web usage mining systems use different data mining techniques, such as clustering, association rule mining, and sequential pattern mining to extract usage patterns from user historical navigational data (Pierrakos et al., 2003). Generally these usage patterns are standalone patterns at the pageview level. They, however, do not capture the intrinsic characteristics of Web users' activities, nor can they quantify the underlying and unobservable factors that lead to specific navigational patterns.

Thus, to better understand the factors that lead to common navigational patterns, it is necessary to develop techniques that can automatically characterize the users' underlying navigational objectives and to discover the hidden semantic relationships among users as well as between users and Web objects. A common approach for capturing the latent or hidden semantic associations among co-occurring objects is Latent Semantic Analysis (LSA) (Landauer et al., 1998). It is mostly used in automatic indexing and information retrieval (Hofmann, 1999), where LSA usually takes the (high dimensional) vector space representation of documents based on term frequency as a starting point and applies a dimension reducing linear projection, such as Singular Value Decomposition (SVD) to generate a reduced latent space representation (Deerwester et al., 1990).

Probabilistic latent semantic analysis (PLSA) models, proposed by Hofmann (1999, 2003), provide a probabilistic approach for the discovery of latent variables which is more flexible and has a more solid statistical foundation than the standard LSA. The basis of PLSA is a model often referred to as the aspect model [17]. Assuming that there exist a set of hidden factors underlying the co-occurrences among two sets of objects, PLSA uses Expectation-Maximization (EM) algorithm to estimate the probability values which measure the relationships between the hidden factors and the two sets of objects.

According to Hotho et al. (2006a, b, c) a folksonomy can be described as a tripartite graph whose vertex set is partitioned into three disjoint sets of users $U = \{u_1, ..., u_l\}$, tags $T = \{t_1, ..., t_n\}$ and items $I = \{i_1, ..., i_m\}$. This model can be simplified to two bipartite models where the collaborative filtering model IU is built from the item user co-occurrence counts *f(i, u)* and the annotation-based model IT derives from the co-occurrence counts between items and tags *f(i, t)*. In the case of social bookmarking IU becomes a binary matrix $(f(i,u) \in \{0,1\})$, as users can bookmark a given web resource only once.

The aspect model of PLSA associates the co-occurrence of observations with a hidden topic variable $Z = \{z_1. \; . \; . \; z_k\}$. In the context of collaborative filtering an observation corresponds to the bookmarking of an item by a user and all observations are given by the co-occurrence matrix IU (Wetzker et al., 2009). Users and items are assumed independent given the topic variable *Z*. The probability that an item was bookmarked by a given user can be computed by summing over all latent variables *Z*:

$$P(i_m \mid u_l) = \sum_k P(i_m \mid z_k) P(z_k \mid u_l),$$

Analog to (3), the conditional probability between tags and items can be written as:

$$P(i_m \mid t_n) = \sum_k P(i_m \mid z_k) P(z_k \mid t_n),$$

Following the Cohn's and Hofmann's procedure (2000), we can now combine both models based on the common factor $P(i_m \mid z_k)$ by maximizing the log-likelihood function:

$$L = \sum_m \left[\alpha \sum_l f(i_m, u_l) \log P(i_m \mid u_l) + (1-\alpha) \sum_n f(i_m, t_n) \log P(i_m \mid t_n) \right]$$

where α is a predefined weight for the influence of each twomode model. Using the Expectation-Maximization (EM) algorithm (Cohn and Hofmann, 2000) it can be performed maximum likelihood parameter estimation for the aspect model. The standard procedure for maximum likelihood estimation in latent variable models is the Expectation Maximization (EM) algorithm (Arenas-García et al., 2007). EM alternates two coupled steps:

(i) an expectation (E) step where posterior probabilities are computed for the latent variables,

(ii) an maximization (M) step, where parameters are updated. Standard calculations yield the E-step equation:

$$P(z_k \mid u_l, i_m) = \frac{P(i_m \mid z_k) P(z_k \mid u_l)}{P(i_m \mid u_l)}$$
$$P(z_k \mid t_n, i_m) = \frac{P(i_m \mid z_k) P(z_k \mid t_n)}{P(i_m \mid t_n)}$$

and then re-estimate parameters in the maximization (M) step as follows:

$$\begin{aligned} P(z_k \mid u_l) &\propto \sum_m f(u_l, i_m) P(z_k \mid u_l, i_m) \\ P(z_k \mid t_n) &\propto \sum_m f(t_n, i_m) P(z_k \mid t_n, i_m) \\ p(i_m \mid z_k) &\propto \alpha \sum_l f(u_l, i_m) P(z_k \mid u_l, i_m) \\ &\quad + (1-\alpha) \sum_n f(t_n, i_m) P(z_k \mid t_n, i_m) \end{aligned}$$

Based on the iterative computation of the above E and M steps, the EM algorithm monotonically increases the likelihood of the combined model on the observed data. Using the parameter, this model can be easily reduced to a collaborative filtering or annotation-based model by setting to 1.0 or 0.0 respectively.

It is possible to recommend items to a user u_l weighted by the probability $P(i_m/u_l)$. For items already bookmarked by the user in the training data this weight set to 0, thus they are appended to the end of the recommended item list.

PLSA as a hybrid approach to the task of item recommendation in folksonomies that includes user generated annotations produces better results than a standard collaborative filtering or annotation-based method.

3.3.3. Collaborative Filtering Based on Collaborative Tagging

Collaborative filtering is based on the assumption that people with similar tastes (i.e., people who agreed in the past) will prefer similar items (i.e., will agree in the future) (Shardanand and Maes, 1995). Traditionally, collaborative filtering techniques predict ratings of items or suggest a list of new items that the user will like the most. In the case of e-learning, collaborative systems track past actions of a group of learners to make a recommendation for individual members of the group (Tan et al., 2008). Based on the assumption that learners with similar past behaviors (browsing, learning path, item ratings or grades that they received by the system) have similar interests and similar appropriate level of knowledge, a collaborative filtering system recommends learning objects the neighbors of the given learner. This approach relies on a historic record of all learner interests such as can be inferred from their ratings of the items (learning objects/learning actions) on a website. Rating can be explicit (explicit ratings or learner satisfaction questionnaires) or implicit (from the studying patterns or click-stream behavior of the learners).

The learner profiles can be represented in a learner-item matrix $X \in R^{m \times n}$, for *m* learners and *n* items. The matrix can be decomposed into row vectors:

$$X := [x_1, \ldots, x_m]^T \;\; with\, x_l := [x_{l,1}, \ldots, x_{l,n}], \quad for\; l := 1, \ldots, m$$

where $x_{l,i}$ indicates that learner *l* rated item *i* by $x_{l,i} \in R$. Each row vector x_l corresponds thus to a learner profile representing the item's ratings of a particular learner. This decomposition usually leads to algorithms that leverage learner-learner similarities, such as the well-known user-based collaborative filtering (Resnick et al., 1994). Given two learners x_u and x_v, we then quantify learners' similarity $sim(x_u, x_v)$ as the cosine of the angle between their vectors:

$$sim(x_u, x_v) = \frac{\langle x_u, x_v \rangle}{\|x_u\| \|x_v\|}$$

Alternatively, Pearson Correlation (and its variations - e.g., weighted Pearson) (Herlocker et al., 2004) could be used.

The matrix X can alternatively be represented by its column vectors:

$$X := [x_1, \ldots, x_n] \;\; with\, x_r := [x_{u,1}, \ldots, x_{m,r}], \quad for\; u := 1, \ldots, n$$

in which each column vector x_r corresponds to a specific item's ratings by all *m* learners. This representation usually leverages item-item similarities and leads to item-based CF algorithms (Deshpande and Karypis, 2004).

Collaborative filtering for tag recommendations in folksonomies aim at modeling user interests based on their historical tagging behaviors, and recommend tags to a user from similar users or user groups (Golder and Huberman, 2006). Tags are used for navigation, finding resources and serendipitous browsing and thus provide an immediate benefit for users. CF tag-based RSs usually include tag recommendation mechanisms easing the process of finding good tags for a resource, but also consolidating the tag vocabulary across users. Specifically, during the collaborative step, users who share similar tagging behaviors with the user we want recommend tags to are chosen based on the between-user similarities, which are calculated based on the users' tagging history. This step usually requires a pre-computed look-up table for the between-user similarities, which is usually in the form of weighted symmetric matrices. After that, the filtering step selects the best tags from those similar users for recommendation.

In the case of e-learning, collaborative tags represent a form of practical metadata, which could supplement in part the needs for detailed learning object descriptions. Also, tagging provides possible solutions for learners' engagement in a number of different annotation activities - add comments, corrections, links, or shared discussion. Learners' tags could create an important trail for other learners to follow by recording their thoughts about specific tutorial resource and could give more comprehensible recommendation about the resources. Therefore, we can conclude that tag collection of like-minded learners offer active learners advice on what is important in a tutorial, what is difficult in a lesson, which example is useful etc. (e.g. learners could observe tag clouds describing course concepts).

Traditionally, collaborative filtering techniques exploit only pairs of two-dimensional data. Thus, because of the ternary relational nature of folksonomy which provides a 3-dimensional relationship between users, items and tags, traditional CF cannot be applied directly. The first adaptation lies in reducing the ternary relation *Y* to a lower dimensional space (Marinho and Schmidt-Thieme, 2007). In the case of user-based CF, we consider as matrix *X* alternatively the two 2-dimensional projections (Figure 3.1) for learner *l*, and item *i*:

$$\pi_{LI}Y \in \{0,1\}^{|L|\times|I|} \textit{with} \, (\pi_{LI}Y)_{l,i} := 1 \ \textit{ if there exist } t \in T \ \textit{s.t} \ (l,t,i) \in Y \textit{ and } 0 \textit{ else}$$

$$\pi_{LT}Y \in \{0,1\}^{|L|\times|T|} \textit{with} \left(\pi_{LT}Y\right)_{l,t} := 1 \textit{ if there exist } i \in I \ \textit{s.t} \ (l,t,i) \in Y \textit{ and } 0 \textit{ else}$$

We first compute the set N_l^k of the *k* learners that are most similar to learner *l*, based on the row decomposed version of X and for a given *k*:

$$N_l^k := \underset{v \in L \setminus \{l\}}{\overset{k}{\operatorname{argmax}}} \operatorname{sim}(x_u, x_v)$$

where the superscript in the *argmax* function indicates the number $k \in N$ of neighbors to be returned. Having the neighborhood computed, we can extract the set $\hat{T}(l,i)$ of *s* recommended tags for a given user *l*, a given item *i*, and some $s \in N$, as follows:

$$\hat{T}(l,i) := \underset{t \in T}{\overset{s}{argmax}} \sum_{v \in N_l^k} sim(x_u, x_v)\delta(v,t,i)$$

$$\textit{where } \delta(v,t,i) := 1 \textit{ if } \delta(v,t,i) \in Y \textit{ and } 0 \textit{ else}$$

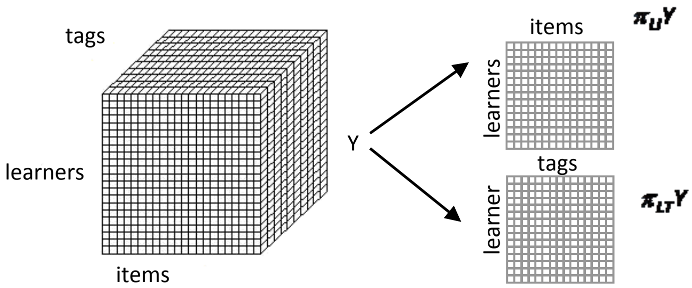

Figure 3.1 Projections of Y into the learner's item and learner's tag spaces (Tso-Sutter et al., 2008)

In order to apply collaborative filtering algorithms for tag recommendation in folksonomies, some data transformation must be performed. Such transformations lead to information loss, which can lower the recommendation quality, but collaborative filtering algorithms are robust against online updates since it does not need to be trained every time a new learner, item or tag enters the system. Especially in the learning process, consideration of like-minded learners that worked on related topics can be of great importance for active learner. Furthermore, as Sood et al. (2007) point out, tag recommendations "fundamentally change the tagging process from generation to recognition" which requires less cognitive effort and time.

3.3.4. Tensor Factorization Technique for Tag Recommendation

Most developing recommendation algorithms (Hotho et al., 2006a; Xu et al., 2006) try to exploit the provided data (users - u, items - i, tags - t) only in 2-dimensional relations. These pairs: (users, tags), (users, items), (tags, items) are analyzed by the different types of the algorithms which determine the most relevant and appropriate content – tags or items for the users. However, these algorithms do not consider the 3 dimensions of the problem altogether, and therefore they miss a part of the semantics that is carried by the 3-dimensions.

Researcher Symenonidis et al. (2008) recognized that involving and exploring existent relationships between tags, users and items can reveal more relevant effects. They suggested tensor based technique which can address the problem of recommendation by capturing the multimodal perception of items by particular users (learning materials by particular learners). It can perform 3-dimensional analysis on the social tags data, attempting to discover the latent factors that determine the associations among the triplets user–tag–item. Consequently, items can be recommended according to the captured associations. That is, given a learner and a tag, the purpose is to predict whether and how much the learner is likely to label with this tag a specific learning item.

As a simple example, let us consider the social tagging system of learners in Protus. Assume two learners. One would like to revision (study/repeat) the examples of examination task and therefore has tagged Example 4 as "useful" and Example 6 as "suitable". Another one learned studiously and has tagged introductory example as "useful" and "basics" for learning next, complex learning material. When wanting to study "useful" examples, both learners are recommended some examples, while the first learner is expecting the examples of examination task and the other prefers the introductory examples.

Recommendation algorithms based on tensor factorization generate their recommendations using ranking score which is computed according to spectral attributes extracted from the underlying folksonomy data structure. By representing $\mathcal{Y}$ as a tensor, one is able to exploit the underlying latent semantic structure in $\mathcal{A}$ formed by multi-way correlations between users, tags, and items. There are different ways to represent $\mathcal{Y}$ as $\mathcal{A}$. Symeonidis et al. (2008), for example, proposed to interpret $\mathcal{Y}$ as a sparse tensor (Figure 3.2 left) in which 1 indicates positive feedback and the remaining data as 0:

$$a_{u,t,i} = \begin{matrix} 1, & (u,t,i) \in Y \\ 0, & else \end{matrix}$$

Rendle et al. (2009) on the other hand, distinguish between positive and negative examples and missing values in order to learn personalized ranking of tags. The idea is that positive and negative examples are only generated from observed tag assignments. All other entries, i.e., all tags for an item that a user has not tagged yet, are assumed to be missing values (Figure 3 right).

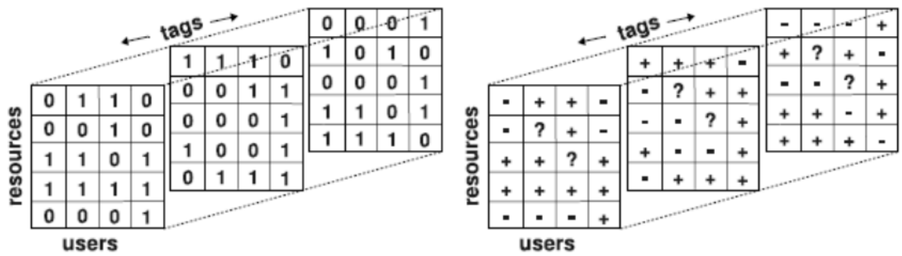

Figure 3.2 Tensor representations - Left (Symeonidis et al., 2008), Right (Rendle et al., 2009)

In this section, we will analyze the recommendation systems based on tensor factorization using Higher Order Singular Value Decomposition (HOSVD). We first provide an outline of SVD approach (Singular Value Decomposition), tensor and HOSVD (Higher Order Singular Value Decomposition) method. Next, we analyze the steps of the RTF (`Ranking with Tensor Factorization') algorithm. In the following, we denote tensors by calligraphic uppercase letters (e.g., $\mathcal{A}$, $\mathcal{B}$), matrices by uppercase letters (e.g., A, B), scalars by lowercase letters (e.g., a, b), and vectors by bold lowercase letters (e.g., **a**, **b**).

3.3.4.1. SVD

The tensor reduction technique based on a SVD (Berry et al, 1994) calculates matrix approximation. The SVD of a matrix $F_{I_1 \times I_2}$ can be written as a product of three matrices:

$$F_{I_1 \times I_2} = U_{I_1 \times I_1} \cdot S_{I_1 \times I_2} \cdot V^T_{I_2 \times I_2}$$

where U is the matrix with the left singular vectors of F, V^T is the transpose of the matrix V with the right singular vectors of F and S is the diagonal matrix of singular values of F. Visualization of the matrix SVD is shown in Figure 3.3.

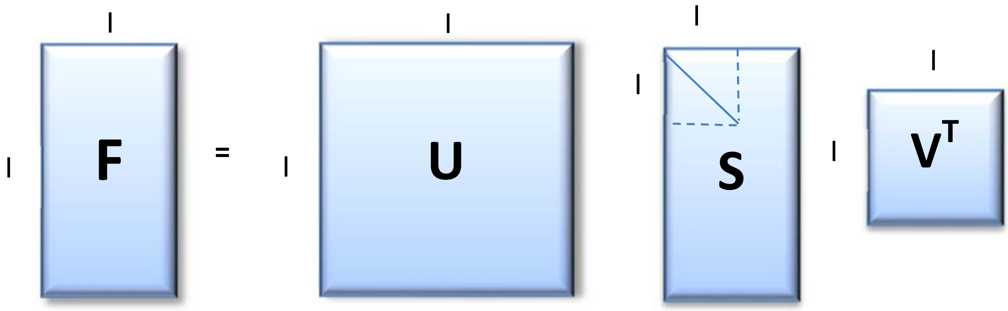

Figure 3.3 Visualization of the matrix SVD

By preserving only the largest $c < \{I_1, I_2\}$ singular values of S, SVD results to matrix $\hat{F}$, which is an approximation of F. The tuning of c is empirically determined by the information percentage that is preserved compared to the original matrix (De Lathauwer et al., 2000).

3.3.4.2. Tensors and HOSVD

A tensor is a multi-dimensional matrix. N-order tensor $\mathcal{A}$ is denoted as $\mathcal{A} \in R^{I_1 \times I_2 \times \ldots \times I_N}$, with elements $a_{i_1,\ldots,i_N}$.

Definition. The n-mode product of a tensor $\mathcal{A} \in R^{I_1 \times I_2 \times \ldots \times I_N}$ by a matrix $U \in R^{J_n \times I_n}$, denoted by $\mathcal{A} \times_n U$, is an $(I_1 \times I_2 \times \ldots \times I_{n-1} \times J_n \times I_{n+1} \times \ldots \times I_N)$- tensor of which the entries are given by (De Lathauwer, 1997):

$$(\mathcal{A} \times_n U)_{i_1 i_2 \ldots i_{n-1} j_n i_{n+1} \ldots i_N} = \sum_{i_n} a_{i_1 i_2 \ldots i_{n-1} i_n i_{n+1} \ldots i_N} u_{j_n i_n}$$

We only use 3-order tensors (the three dimensions are: *u*-users, *i*-items and *t*-tags) where $\mathcal{A} \in R^{u \times t \times i}$. Each tensor element measures the preference of a (user *u*, tag *t*) pair on an item *i*. Tensor $\mathcal{A}$ can be metricized i.e., to build matrix representations in which all the column (row) vectors are stacked one after the other.

Thus, after the unfolding of tensor $\mathcal{A}$ for all three modes, we create 3 new matrices A_1, A_2 and A_3 as follows (De Lathauwer et al., 2000):

$$A_1 \in R^{I_u \times I_t I_i},$$
$$A_2 \in R^{I_t \times I_u I_i},$$
$$A_3 \in R^{I_u I_t \times I_i}$$

Where A_1, A_2, and A_3 are called the 1-mode, 2-mode, 3-mode matrix unfolding of $\mathcal{A}$, respectively. The unfolding of $\mathcal{A}$ *in* the three modes, is illustrated in Figure 3.4.

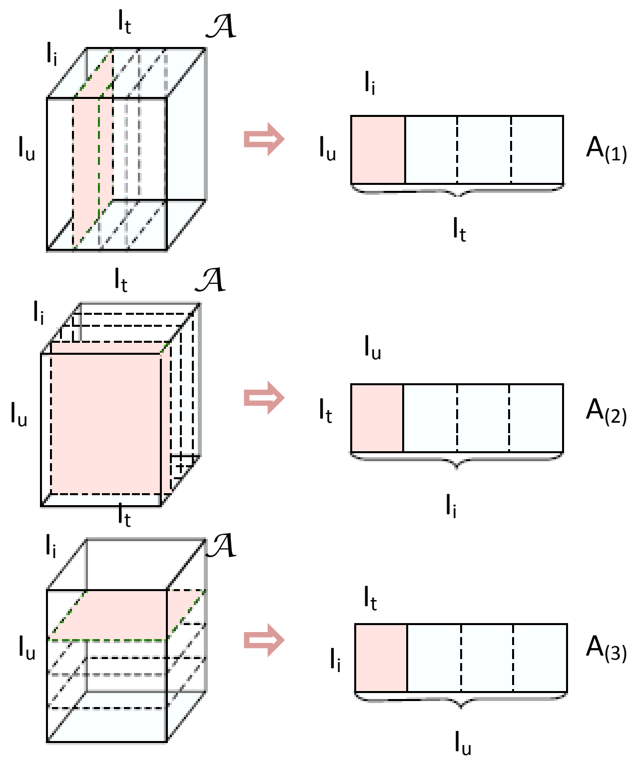

Figure 3.4 Visualization of the three unfoldings of a 3-order tensor A

In terms of n-mode products, SVD on a regular two-dimensional matrix (i.e., 2-order tensor), can be rewritten as follows (De Lathauwer et al., 2000):

$$F = S \times_1 U^{(1)} \times_2 U^{(2)}$$

where $U^{(1)} = (u_1^{(1)} u_2^{(1)} \dots u_{I_u}^{(1)})$ is a unitary ($I_u \times I_u$)-matrix, $U^{(2)} = (u_1^{(2)} u_2^{(2)} \dots u_{I_u}^{(2)})$ is a unitary ($I_t \times I_t$) -matrix and S is an ($I_u \times I_t$)-matrix with the following properties:

- Pseudodiagonality ($S = diag(\sigma_1, \sigma_2, \dots \sigma_{\min\{I_u, I_t\}})$)
- Ordering ($\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_{\min\{I_u, I_t\}} \geq 0$)

By extending this form of SVD, the HOSVD of 3-order tensor $\mathcal{A}$ can be written as follows (De Lathauwer et al., 2000):

$$\mathcal{A} = S \times_1 U^{(1)} \times_2 U^{(2)} \times_3 U^{(3)}$$

where $U^{(1)}$, $U^{(2)}$, $U^{(3)}$ contain the orthonormal vectors (called the 1-mode, 2-mode and 3-mode singular vectors, respectively) spanning the column space of the A_1, A_2, A_3 matrix unfoldings. S is the core tensor and has the property of all orthogonality. This process is illustrated in Figure 3.5.

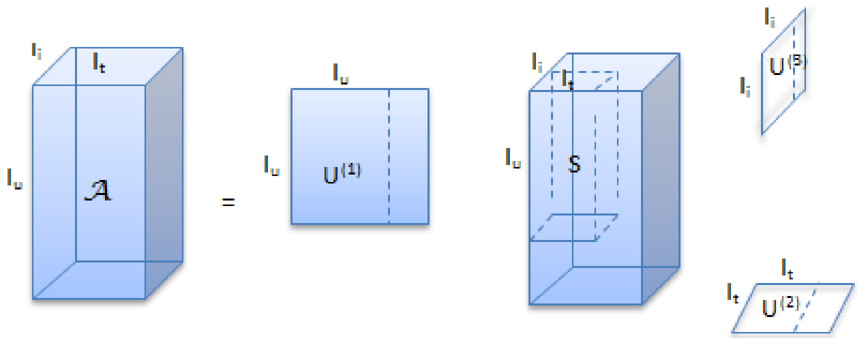

Figure 3.5 Visualization of HOSVD

An initial 3-order tensor $\mathcal{A} \in R^{u \times t \times i}$ is created from the usage data triplets (user, tag, and item). Each tensor element measures the preference of a (user *u*, tag *t*) pair on an item *i*. This initial tensor $\mathcal{A}$ is matricized in all three modes. Thus, after the unfolding of tensor $\mathcal{A}$ for all three modes, we create 3 new matrices A_1, A_2, A_3, as follows.

$$A_1 \in R^{I_u \times I_t I_i},$$
$$A_2 \in R^{I_t \times I_u I_i},$$
$$A_3 \in R^{I_u I_t \times I_i}$$

SVD is applied on these three matrix unfoldings. We result to total 9 new matrices.

$$A_1 = U^{(1)} \cdot S_1 \cdot V_1^T$$
$$A_2 = U^{(2)} \cdot S_2 \cdot V_2^T$$
$$A_3 = U^{(3)} \cdot S_3 \cdot V_3^T$$

For tensor dimensionality reduction, there are three parameters to be determined. The numbers c_1, c_2, and c_3 of left singular vectors of matrices $U^{(1)}$, $U^{(2)}$, $U^{(3)}$ which are determinative for the final dimension of the core tensor S. Since each of the three diagonal singular matrices S_1, S_2 and S_3, are calculated by applying SVD on matrices A_1, A_2, and A_3, respectively, we use different c_1, c_2, and c_3 values for each matrix $U^{(1)}$, $U^{(2)}$, $U^{(3)}$. The numbers c_1, c_2, and c_3 are empirically chosen by maintaining a percentage of information of the original S_1, S_2 and S_3 matrices after appropriate modification. Usually the percentage is set to 70% of the original matrix.

The core tensor S governs the interactions among user, item and tag entities. Since we have selected the dimensions of $U^{(1)}$, $U^{(2)}$, $U^{(3)}$ matrices, we proceed to the construction of the core tensor S, as follows (Symenonidis et al., 2008) :

$$S = \mathcal{A} \times_1 {U_{c_1}^{(1)}}^T \times_2 {U_{c_2}^{(2)}}^T \times_3 {U_{c_3}^{(3)}}^T$$

where $\mathcal{A}$ is the initial tensor, ${U_{c_1}^{(1)}}^T$ is the transpose of the c_1-dimensionally reduced $U^{(1)}$ matrix, ${U_{c_2}^{(2)}}^T$ is the transpose of the c_2-dimensionally reduced $U^{(2)}$, and ${U_{c_3}^{(3)}}^T$ is the transpose of the c_3-dimensionally reduced $U^{(3)}$. Finally, tensor $\hat{\mathcal{A}}$ is built by the product of the core tensor S and the mode products of the three matrices $U^{(1)}$, $U^{(2)}$, $U^{(3)}$ as follows:

$$\hat{\mathcal{A}} = S \times_1 U_{c_1}^{(1)} \times_2 U_{c_2}^{(2)} \times_3 U_{c_3}^{(3)}$$

where S is the c_1, c_2, and c_3 reduced core tensor, $U_{c_1}^{(1)}$ is the c_1-dimensionally reduced $U^{(1)}$ matrix, $U_{c_2}^{(2)}$ is the c_2-dimensionally reduced $U^{(2)}$ matrix, $U_{c_3}^{(3)}$ is the c_3 - dimensionally reduced $U^{(3)}$ matrix.

The reconstructed tensor $\hat{\mathcal{A}}$ measures the associations among the users, tags and items. The model parameters to be learned are then the quadruple $\hat{\theta} := (S, U_{c_1}^{(1)}, U_{c_2}^{(2)}, U_{c_3}^{(3)})$.

The basic idea is to minimize an element-wise loss on the elements of $\hat{\mathcal{A}}$ by optimizing the square loss, i.e.,

$$\arg\min_{\hat{\theta}} \sum_{(u,t,i) \in U \times T \times I} (\hat{a}_{u,t,i} - a_{u,t,i})^2$$

3.3.4.3. Ranking with Tensor Factorization

Rendle et al. (2009) propose RTF (Ranking with Tensor Factorization), a method for learning an optimal factorization of $\mathcal{A}$ *for* the specific problem of tag recommendations. First, the observed tag assignments are divided in positive and negative. All other entries (e.g. all tags for an item that a user has not tagged yet) are assumed to be missing values, as described in Section 3.3.4 (see right-hand side of Figure 3.2). Let $P_{\mathcal{A}} := \{(u,i) \mid \exists t \in T : (u,t,i) \in Y\}$ be the set of all distinct user/item combinations in $\mathcal{Y}$, the sets of positive and negative tags of a particular $(u,i) \in P_{\mathcal{A}}$ are then defined as:

$$T_{u,i}^{+} := \{t \mid (u,i) \in P_{\mathcal{A}} \wedge (u,t,i) \in Y\}$$
$$T_{u,i}^{-} := \{t \mid (u,i) \in P_{\mathcal{A}} \wedge (u,t,i) \notin Y\}$$

From this, pairwise tag ranking constraints can be defined for the values of $\hat{\mathcal{A}}$:

$$a_{u,t_1,i} > a_{u,t_2,i} \Leftrightarrow (u,t_1,i) \in T_{u,i}^{+} \wedge (u,t_2,i) \in T_{u,i}^{-}$$

From a semantically point of view this scheme makes more sense as the user/item combinations that have no tags are the ones that the recommender system will have to predict in the future. Thus, instead of minimizing the least-squares as in the HOSVD-based methods, an optimization criterion that maximizes the ranking statistic AUC (area under the ROC-curve) is proposed. The AUC measure for a particular $(u,i) \in P_{\mathcal{A}}$ is defined as:

$$AUC(\hat{\theta},u,i) := \frac{1}{|T_{u,i}^{+}||T_{u,i}^{-}|} \sum_{t^{+} \in T_{u,i}^{+}} \sum_{t^{-} \in T_{u,i}^{-}} H_{0,5}(\hat{a}_{u,t^{+},i} - \hat{a}_{u,t^{-},i})$$

where H_{α} is the Heaviside function:

$$H_{\alpha} := \begin{cases} o, & x < 0 \\ \alpha, & x = 0 \\ 1, & x > 0 \end{cases}$$

The overall optimization task with respect to the ranking statistic AUC and the observed data is then:

$$\underset{\hat{\theta}}{\arg\max} \sum_{(u,i) \in P_{\mathcal{A}}} AUC(\hat{\theta},u,i)$$

3.3.4.4. Multi-mode Recommendations

Once $\hat{\mathcal{A}}$ is computed, the recommendation list with the N highest scoring tags for a given user *u* and a given item *i* can be calculated by:

$$Top(u,i,N) := \underset{t \in T}{\arg\max}^{N} \; \hat{a}_{u,t,i}$$

Recommending N items to a given user *u* for a particular tag *t* can be done in a similar manner. Moreover, other users can be recommended to a particular user *u* given a specific tag *t*, according to the total score that results by aggregating all items that are tagged with *t* by *u*. Thus, according to the data representation, tensor modeling permits multi-mode recommendations in an easy way.

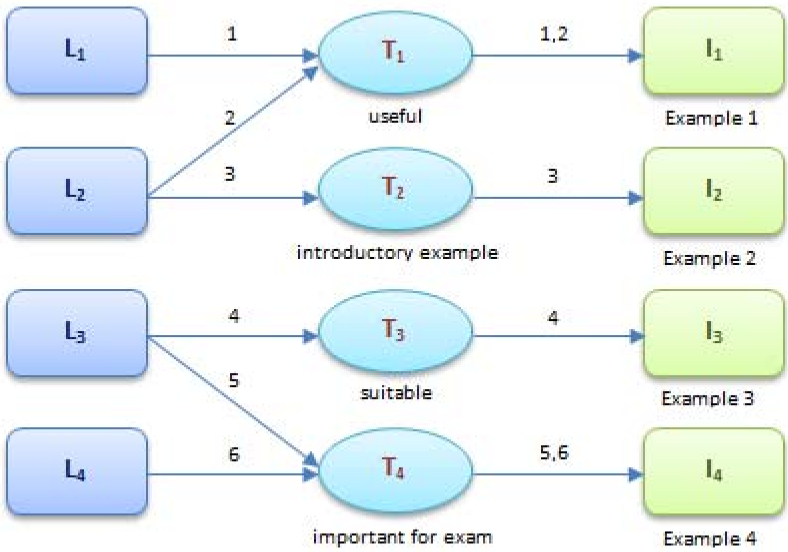

Figure 3.6 Running example

To exemplify this approach, we apply the RTF algorithm to a running example, which is illustrated in Figure 3.6. As it can be seen, 4 learners tagged 4 different items. In the figure, the arrow lines and the numbers on them give the correspondence between the three types of entities. For example, learner L_1 tagged with tag "useful" (denoted as T_1) the item "Introductory example" (denoted as I_1). From Figure 3.6, we can see that learners L_1 and L_2 have common interests on introductory example, while learners L_3 and L_4 have common interests in the examples of examination task. A 3-order tensor $A \in R^{4\times4\times4}$ can be created from these usage data. We use the co-occurrence frequency of learner, tag and item as the elements of tensor A, which are given in Table 3.2.

Table 3.2 Tensor created from the usage data

Arrow line	Learner	Tag	Item	Weight
1	L_1	T_1	I_1	1
2	L_2	T_1	I_1	1
3	L_2	T_2	I_2	1
4	L_3	T_3	I_3	1
5	L_3	T_4	I_4	1
6	L_4	T_4	I_4	1

After performing the tensor reduction analysis, we get the reconstructed tensor of $\hat{A}$. Table 3.3 gives the output of the tensor reduction algorithm, which is also illustrated in Figure 3.7. We can notice that the algorithm outputs new associations between the included entities (the last rows in the Table 3.3 and the dotted lines in Figure 3.7). Even though in the original data, learner L_1 did not tag items I_2, the algorithm is capable to conclude that if L_1 would tag them, then L_1 would likely (likelihood 0.35) use tag "introductory example". As well, the algorithm can assume that if L_4 would tag item I_4 with another tag, then L_4 would likely (likelihood 0.44) use the tag "suitable".

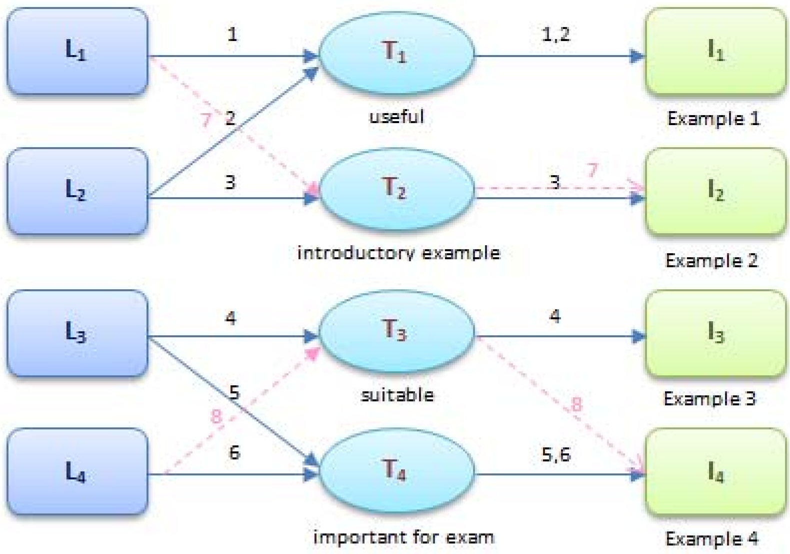

Figure 3.7 Illustration of the tensor reduction algorithm output for the running example

The tensor reduction approach is able to capture the latent associations among the multi-type data entities: learners, tags and items. These associations can further be used to improve the recommendation procedure.

Table 3.3 Tensor constructed from the usage data of the running example

Arrow line	Learner	Tag	Item	Weight
1	L_1	T_1	I_1	0.72
2	L_2	T_1	I_1	0.5
3	L_2	T_2	I_2	1.18
4	L_3	T_3	I_3	0.35
5	L_3	T_4	I_4	0.35
6	L_4	T_4	I_4	0.44
7	L_1	T_2	I_2	1.18
8	L_4	T_3	I_4	0.72

3.3.5. Most Popular Tags

The web algorithms Most Popular Tags (Most Popular Tags by Item and Most Popular Tags by User) are based on tag counts. In the rest of this section it will be presented that these methods are particulary cheap to compute and therefore might be good candidates for online computation of recommendations.

If we want to compute, for a given pair (u, i), the most popular tags of the user u (or the item i), we need to linearly scan Y to calculate the occurrence counts for u's tags (or i's tags) and afterwards sort the tags we gathered by their count.

For a user $u \in U$, the set of all his tag assignements is $Y_u := Y \cap (\{u\} \times T \times I)$. The sets Y_i (for any item $i \in I$) and Y_t (for any tag $t \in T$) are defined accordingly. Similiarly, for $t \in T$ and $i \in I$, $Y_{t,u} := Y \cap (\{u\} \times \{t\} \times R)$ and $Y_{t,i}$ accordingly. Finally, for a user $u \in U$, the set of all his tags can be defined as $T_u := \{t \in T \mid \exists i \in I : (u,t,i) \in Y\}$. The set T_i (for any item $i \in I$) is defined accordingly.

There are three types of "Most Popular Tags" algorithms:

1. Recommending the most popular tags of the folksonomy is the most simplistic approach. It recommends, for any user $u \in U$ and any item $i \in I$, the same set:
$$\hat{T}(u,i) := \underset{t \in T}{\arg\max^{n}}(|Y_t|)$$
2. Tags that globally are most specific to the item will be recommended when using the most popular tags by item:
$$\hat{T}(u,i) := \underset{t \in T}{\arg\max^{n}}(|Y_{t,i}|)$$
3. Since users might have specific interests for which they already tagged several items, using the most popular tags by user is another option:
$$\hat{T}(u,i) := \underset{t \in T}{\arg\max^{n}}(|Y_{t,u}|)$$

None of the aforementioned methods alone will in general provide the best recommendations. Nevertheless, the simplicity and cost efficiency of algorithms based on tag counts make them a favored approach for use in existing folksonomy systems. J¨aschke et al. (2007) experimented with a mix of the recommendations generated by variants 2 and 3 which are called most popular tags mix.

3.3.5.1. Mix of "Most Popular Tags" Recommenders

The main idea of this approach is to recommend a mix of the most popular tags of the user with the most popular tags of the item. The simplest way to mix the tags is to add their counts and then sort them by their count:

$$\hat{T}(u,i) := \underset{t \in T}{\overset{n}{\arg\max}}(|Y_{t,u}| + |Y_{t,i}|)$$

This way of mixing are called most popular tags mix 1:1, since we just add the counts as they are. For instance, if the item has been tagged three times with „popular" by other users and the user has used the tag „popular" four times on other items, the tag „popular" would get a count of seven.

Although this method already contributes good results, the influence of the user-based recommendation will be very small compared to the item-based recommendation if many people have tagged this item. On the contrary, if a user has tagged many items, his most popular tags might have counts that are much higher than the counts provided by the items. Therefore, J¨aschke et al. (2007) introduced another mix variant, where the tag counts of the two participating sets are normalized and weighted before they are added. Normalization function is defined for each tag $t \in T_i$:

$$norm_i(t) := \frac{|Y_{t,i}| - \min_{t' \in T} |Y_{t',i}|}{\max_{t' \in T} |Y_{t',i}| - \min_{t' \in T} |Y_{t',i}|}$$

For $t \in T_u$, the normalization $norm_u(t)$ is defined in analogue manner. After normalization the weights of all tags in T_i and T_u lie between zero and one – with the most popular tag(s) having weight 1 and the least important tag(s) having weight 0. A pre-defined factor $\rho \in [0,1]$ allows us to balance the influence of the user and the item:

$$\hat{T}(u,i) := \underset{t \in T}{\overset{n}{\arg\max}}(\rho\, norm_i(t) + (1-\rho) norm_u(t))$$

This method is called *The Most Popular Tags* ρ *– Mix*. T*he Most Popular Tags 0 – Mix* is just the most popular tags by user strategy, since the normalization does not change the order of the tags. Similarly, *The Most Popular Tags 1 – Mix* is just the most popular tags by item strategy. However, due to normalization *The Most Popular Tags 0.5 – Mix* is not identical to *The Most Popular Tags Mix 1:1*.

3.4. Limitations of Current Folksonomy and Possible Solutions

Tagging systems have the potential to improve search, recommendation and personal organization while introducing new modalities of social communication. As described in this section, there has been much research done on tag-based recommendation technologies that have significantly advanced the state-of-the-art in comparison to early recommender systems utilized collaborative and content-based heuristics. Despite the rapid expansion of applications that support tagging of items, the simplicity and ease of use of tagging however, lead to problems with current folksonomy systems, which hinder the growth or affect the usefulness of the systems. The problems can be classified in some categories (Mathes, 2004; Shepitsen et al., 2008; Pluzhenskaia, 2006; Gordon-

Murnane, 2006). We consider set of limitations which can directly affect the tag-based recommendation process in e-learning environments.

1. Tags have little semantics and many variations. Thus, even if a tagging activity can be considered as the learner's cognitive process, the resulting set of tags does not always correctly and consistently represent the learner's mental model.
2. As an uncontrolled vocabulary that is shared across an entire system, the terms in a folksonomy have inherent ambiguity, as different learners apply terms to items in different ways. Tag ambiguity, in which a single tag has many meanings, can falsely give the impression that items are similar when they are in fact unrelated.
3. Tag redundancy, in which several tags have the same meaning, can obfuscate the similarity among items. Redundant tags can hinder algorithms that depend on identifying similarities between items.
4. The use of different word forms such as plurals and parts of speech also exacerbate the problem.

There are some different approaches aiming to solve the mentioned problems. First one tries to educate learners to improve "tag literacy" (Guy and Tonkin, 2006). An important condition for this way of resolving problems is to establish learner researches about folksonomies (Bar-Ilan et al., 2006; Winget, 2006; Lin et al., 2006), concerning the "deep nature" of tags (Veres, 2006a), discussing aspects of the folksonomy interoperability (Veres, 2006b) and the "semiotic dynamics" of folksonomies in terms of tag co-occurrences (Cattuto et al., 2007). For training the learner's selection of "good" tags it may be useful that the system would suggest some tags (MacLaurin, 2007). Tag-suggestions can operate on a syntactical level (e.g., a learner attaches "graph" and the system suggests "graphics") or even on a relational level (e.g., a learner attaches "graphics" and the system suggests "image", because both words do often co-occur in items' tag clouds (Xu et al., 2006). Also, tag-suggestion can be based on experts' opinions, providing high-quality of the resulting tags that are objective and cover multiple aspects.

These extensions leave ample opportunity for future work in this area. They can improve tag-based recommendation capabilities and make collaborative tagging systems applicable to en even broader range of applications.

Chapter 4

Design, Architecture and Interface of Protus System

Protus is a tutoring system designed to help learners in learning essentials of programming languages. The Protus architecture evolved from existing web-based Java tutoring system called Mag (Ivanović et al., 2008) that is developed at Department of Mathematics and Informatics, Faculty of Science, Novi Sad. Mag has been multifunctional educational system that fulfills three primary goals, identified by earlier exploration in this field (Jones et al., 2006). The first goal was to provide intelligent tutoring system for learners in a platform independent manner. The second goal was to provide the teachers with useful reports identifying the strengths and weaknesses of learner's learning process. Finally, the third goal was to provide a rapid development tool for creating basic elements of tutoring system: new learning objects, units, tutorials and tests.

The main purpose of the extension of Mag system is to recommend useful and interesting contents to e-learners based on their different backgrounds, preferences, learning purposes and other meaningful attributes (Klašnja-Milićević et al., 2011a). In spite of the fact that this system is designed and implemented as a general tutoring system for different programming languages, the first completely implemented and tested version has been used for an introductory Java programming course (Vesin et al., 2008). Java has been chosen because it is a programming language widely used at Novi Sad University, and because it is a clear example of an object-oriented language and therefore suitable for teaching the concepts of object-orientation.

This chapter will summarize the general setup of the Protus system before discussing the recommendation module in detail. After reviewing and illustrating related work in Section 4.1., Section 4.2 outlines the overall system architecture and describes the recommendation module and process of personalization in detail. Protus interface is described in the final section 4.3.

4.1. Related Work

Computer technology has been used to develop a wide range of educational software, from early computer-based training systems to web-based adaptive hypermedia, multimedia courseware, and educational games. These systems have given learners access to a great variety of pedagogical approaches that supplement classroom learning and provide items outside the classroom. This variety has been helpful in reaching learners who don't do well with traditional lecture and textbook instruction. During our research, we focused our attention only on a specific kind of tutoring systems. In the remainder of this section, we first describe programming tutoring systems in general and we then present tutoring systems that use different recommendation techniques in order to suggest the most appropriate online learning activities to learners, based on their preferences, learning style, knowledge and the browsing history of other learners with similar characteristics.

4.1.1. Programming Tutoring Systems

Most of the tutoring systems for learning programming languages found on the Web are more or less only well-reformatted versions of lecture notes or textbooks (Vesin et al., 2009). As a consequence these systems don't have implemented interactivity and adaptivity.

The functions that such systems can perform vary. Some of them are used for learner assessment like JavaBugs (Suarez and Sison, 2008) and JITS (Sykes and Franek, 2003; Sykes, 2007). Also, some of them are adaptive web-based tutorials (Romero and Ventura, 2006). One step further in implementation of adaptation was made by systems like JOSH-online (Bieg and Diehl, 2004), iWeaver (Wolf, 2003) and CIMEL ITS (Wei et al., 2005; Glenn et al., 2004). JavaBugs examines a complete Java program and identifies the most similar correct program to the learner's solution among a collection of correct solutions. After that it builds trees of misconceptions using similarity measures and background knowledge (Suarez and Sison, 2008). They focused on the construction of a bug library for novice Java programmer errors, which is a collection of commonly occurring errors and misconceptions.

The WWW-based introductory LISP course ELM-ART (ELM Adaptive Remote Tutor) is based on ELM-PE (Brusilovsky et al., 1996), an on-site intelligent learning environment that supports example-based programming, intelligent analysis of problem solutions, and advanced testing and debugging facilities. For annotating the links, the authors use the traffic light metaphor. A red ball indicates pages which contain information for which the user lacks some knowledge, a green ball indicates suggested links, etc. Java Intelligent Tutoring System - JITS is a tutoring system designed for learning Java programming (Sykes and Franek, 2003). JITS implements JECA (Java Error Correction Algorithm), an algorithm for a compiler that enables error correction intelligently changing code, and identifies errors more clearly than other compilers. This practical compiler intelligently learns and corrects errors in learners' program (Sykes, 2007). iWeaver is an interactive web-based adaptive learning environment, developed as a multidisciplinary research project at RMIT University Melbourne, Australia (Wolf, 2003). iWeaver was designed to provide an environment for the learner by implementing adaptive hypermedia techniques to teach the Java programming language. It implements several established adaptation techniques, including link sorting, link hiding and conditional page content. The current version of iWeaver does not support adaptive navigation, which is one of the best researched areas of adaptive environments. JOSH is an interpreter for the Java programming language (Bieg and Diehl, 2004) originally designed to make easier teaching Java to beginners. Recently the interpreter was restructured into a server based interpreter applet and integrated into an online tutorial on Java programming called JOSH-online. CIMEL ITS is an intelligent tutoring system that provides one-on-one tutoring to help beginners in learning object-oriented analysis and design. It uses elements of UML before implementing any code (Glenn et al., 2004). A three-layered Learner model is included which supports adaptive tutoring by deducing the problem-specific knowledge state from learner solutions, the historical knowledge state of the learner and cognitive reasons about why the learner makes an error (Wei et al., 2005). This Learner model provides an accurate profile of a learner so that the intelligent tutoring system can support adaptive tutoring.

Most of the existing e-learning platforms for teaching programming have not yet taken the advantage of adaptivity (Emurian, 2006; Holland et al., 2009; Sykes and Franek, 2003), possibly because the expected profit has not justified the high effort of implementing and authoring adaptive courses. Moreover, most of the adaptive tutoring systems do not support e-learning standards. Our system recommends a media experience that is most likely to be chosen in the current learning context by the current learner. This recommendation mechanism is attempting to accommodate to a possible variation in a learner’s learning style profile. Also, up to now most if not all systems do not take into consideration the important aspect of learning styles preferences or how and when to adjust the presented topic based on the preferred presentation method of the learner.

4.1.2. Tutoring Systems with Implemented Recommendation

A personalized recommender system that uses web mining techniques for recommending a learner which (next) links to visit within an adaptable educational hypermedia system was described in Romero and Ventura (2006) paper. They presented a specific mining tool and a recommender engine that they have integrated in the AHA! system, in order to help the teacher to carry out the whole web mining process. They made several experiments with real data in order to show the suitability of using together, clustering and sequence mining algorithms, for discovering personalized recommendation links (Brusilovsky et al., 1996). Another system described in (Soonthornphisaj et al., 2006) allows all learners to collaborate and exchange their expertise in order to predict the most suitable learning materials to each learner. This smart e-Learning system applies the collaborative filtering approach (Soonthornphisaj, 2006) that has an ability to predict the most suitable documents to the learner. All learners have the chance to introduce new material by uploading the documents to the server or pointing out the web link from the Internet and rate the currently available materials. My Online Teacher 2.0 (MOT 2.0) successfully combines Web 2.0 features (such as tags, rating system, feedback, etc.) in order to support both learners and teachers in personalized systems (Ghali and Cristea, 2009). Ghali and Cristea (2009) focus on a study which can explain how to more effectively use and combine the recommendation of peers and content adaptation to enhance the learning outcome in e-learning systems.

In the last few years, some research studies have been conducted on developing an approach that identifies learning styles from learners' behavior in an online course (García et al., 2007; Graf et al., 2010b). The rationale is that adapting courses to the learning preferences of the students has a positive effect on the learning process, leading to an increased efficiency, effectiveness and/or learner satisfaction (Popescu, 2010). The adaptive response of existing environments is often restricted to pictures and text instead of multimedia presentations, with some exceptions like the iWeaver (Wolf, 2003). Systems like Logic-ITA, ProGuide and Jeliot 3 gave us good ideas and perspective which functionalities could be included in new web-based tutoring system (Merceron and Yacef, 2004; Myller, 2007). Compared to current tutoring systems which are only executed on standalone machine (JavaBugs, JITS, CIMEL ITS, Jeliot 3) or have just basic interactivity and adaptivity implemented (JOSH-online, JavaBugs, Logic-ITA), the Protus system integrates content and link adaptation in order to accomplish completely functional web-based tutoring system with personalization capability. Protus, as an e-learning system, offers a constant on-the-fly adaptation of the course units and their presentation to the current needs and preferences of the individual learner. This guarantees a significant, individual success of a learner. None of the above mentioned systems implement full use of the recommender techniques (like collaborative filtering, association rule mining and clustering), just the basic data mining techniques. Second, besides learning content ranking and tagging, Protus also supports learning path generation and personalization based on the learning styles identification. Our work differs from previous mentioned papers in several aspects. First, we combine several adaptation techniques, both recommendations of material and adaptive hypermedia, in order to personalize lessons presentation to learners. Second, besides learning content ranking and tagging, Protus also supports learning material clustering and learning path generation. Third, despite of the great variety of tutoring systems in the literature we chose to focus our attention on programming tutoring system that defines scalable and adaptable architecture. Protus provides possibility to import knowledge from various domains in our system so that the process of learning can be performed in whatever domain of knowledge. This choice enabled us to develop a system for knowledge presentation and acquisition that tries to be independent of the specific domain.

4.2. System Architecture and Design

System architecture of Protus was designed in a form of centralized architecture that was proven to be the most effective for constructing tutoring systems (Chen, 2008). Figure 4.1 shows a graphical representation of the Protus system architecture. This architecture presents adapted architecture of first version of Mag system (Vesin et al., 2007) based on experiences of similar web-based learning systems, (Jones et al., 2006), (Šimić, 2004), (Chen, 2008).

4.2.1. *Design of Protus System*

Besides being beneficial for providing learners with personalized learning experience, the implemented architecture and the reasoning that is performed over it, are also useful for generating feedback for other key participants in the learning process - teachers. Likewise, the framework can be used to provide feedback to teachers about the learners' activities, their performance, achieved knowledge level and so on. In both cases, the feedback can help in improving the learning process. To support this statement, several goals were fulfilled in the Protus system (Klašnja-Milićević, et al., 2009):

- separation of the two different interfaces - for learners and teachers,
- a strict separation of different modules: domain, application, adaptation and learner, in order to ensure a good modularization of the system components,
- permanent administration of learning progression, preferences and personal data of learners within sharable and dynamic learner model,
- enabling communication and collaboration among learners and between learners and teachers,
- assessment of knowledge and increasing competency level of learners,
- functionalities for creation of new learning content and migration of content from external sources,
- semantically rich descriptions of the components' functionality, in order to allow effective interoperability among system components, and
- providing effective coordination and communication between the system components.

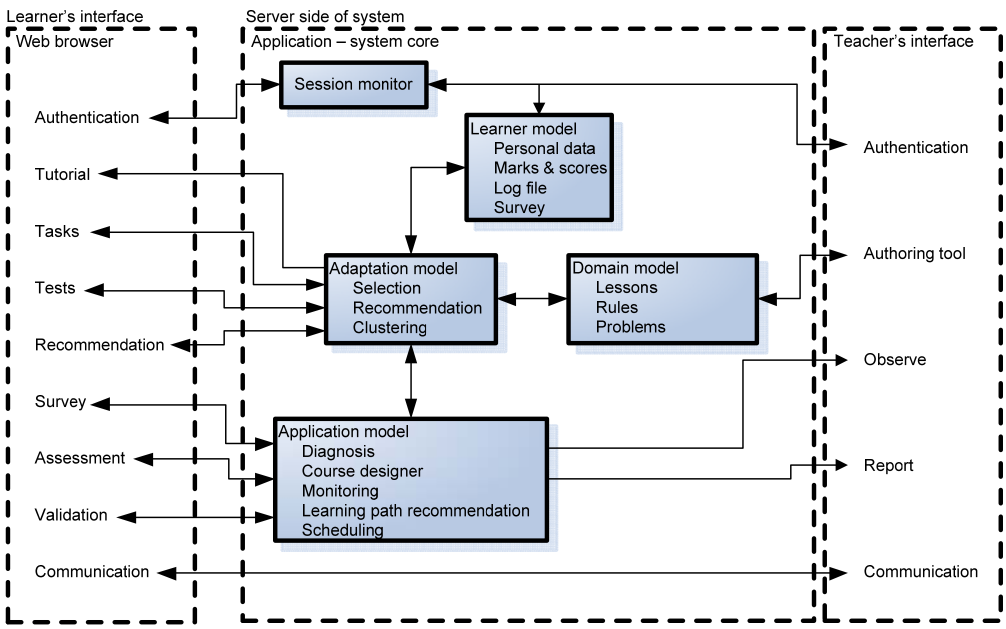

Figure 4.1 Protus system architecture

4.2.2. *Protus System Architecture*

As a result of Protus design, essentially a centralized architecture of the system has been reached (Klašnja-Milićević et al., 2009). Two separated user interfaces are provided for both learner and his/her teacher (Figure 4.1). The teacher's interface is a windows application with functionalities for managing data about learners and data of course materials. The basic functionalities of this application are:

- review and update the database of learners (adding and deleting data about learners - participants of the course),
- receiving reports about the progress of the learner (the number of passed lesson, success in individual lessons, etc.)
- review and update the data about teachers,
- easy search data about learners by different criteria,
- communication with learners (review received messages and send new ones),
- adding new units and lessons, as well as adding new examples and tests within the lesson.

The learner's interface is a series of web *jsp* pages that provides options for taking lessons and testing learner's knowledge. All data about the learners and their progress in the course as well as data about lessons are stored in the system's server.

The proposed architecture has numerous benefits. Integrated development environment – NetBeans, used as a primary tool for the Protus system construction, is platform-independent, lightweight and scalable. Learners do not need to install software on their own machine and they do not need a high-speed network connection to use the Protus. Other benefits include fast execution, since all processing is done on the J2EE server that typically has much faster and more efficient hardware than typical PCs.

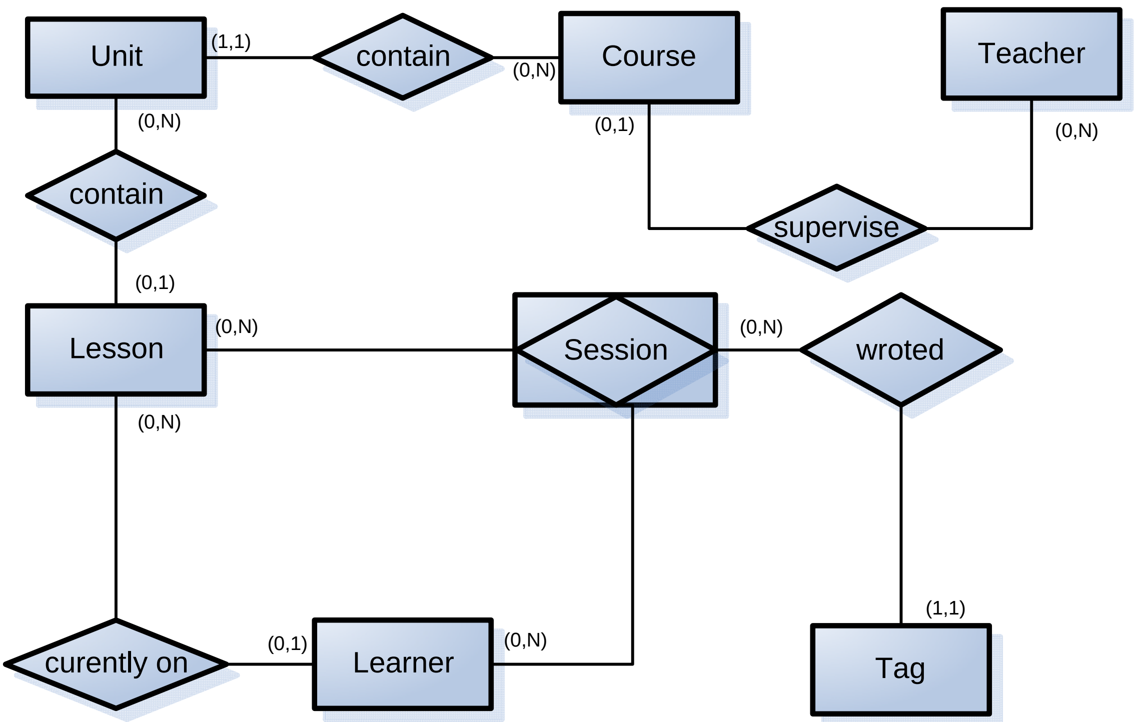

Figure 4.2 ER diagram of the Protus system database

The architecture uses a JDBC (Java DataBase Connectivity) connection to database which stores and retrieves specific information about learners, including their progress history and performance statistics, teachers, course, unit, lessons and tagging, as will be explained in the next section.

Table 4.1 Table *Learner*

Field	Type	Description
idlearner	**Int**	**Identification number of learner**
name	varchar	Learner' s name
surname	varchar	Learner' s surname
username	varchar	Learner' s username
password	varchar	Learner' s password
gender	varchar	Gender of learner
year	int	Year of birth
adr	varchar	Learner' s adress
processing	int	Category within 'Information Processing' domain
perception	int	Category within 'Information Perception' domain
reception	int	Category within 'Information Reception' domain
understanding	int	Category within 'Information Understanding' domain
begin_time	date	Date of the course beginning
overall_time	time	Total duration of the course
avg_grade	decimal	Average grade of learner
percentage	decimal	Percentage of course completed
lesson	int	Number of completed lesson
curlesson	int	Current lesson

4.2.2.1. Protus System Database

The Protus system uses and updates the database with data about learners, teachers, course, unit, lessons, tagging and evaluation process. The ER diagram of the database is shown in Figure 4.2. The database consists of seven tables:

1. Learner. It contains basic information about the learner as well as some information about the learning styles and learner's progress (Table 4.1).
2. Teacher. It contains basic information about the teacher (Table 4.2).
3. Lesson. It contains information about the lessons (Table 4.3).
4. Unit. It contains information about the unit, lesson and learning objects (resources) from which lesson is consisted (Table 4.4).
5. Course. It contains information about the course, units, lessons, number of learners attending the course and duration of the course (Table 4.5).
6. Session. It contains information about learner sessions that the learner has completed during the course and the grades he/she earned for them (Table 4.6).

7. Tag. It contains information about tags and information about lessons and learning objects for which the tag is placed (Table 4.7).

Table 4.2 Table *Teacher*

Field	Datatype	Description
idteacher	**Int**	**Identification number of teacher**
firstname	varchar	Teacher's first name
lastname	varchar	Teacher's last name
username	varchar	Teacher's username
password	varchar	Teacher's password
title	varchar	Teacher's title

Table 4.3 Table *Lesson*

Field	Datatype	Description
idlesson	**int**	**Identification number of lesson**
name	varchar	Name of the lesson
resources	int	Overall number of learning objects
intro	int	Number of resources of introduction type
basic info	int	Number of resources of basic info type
example	int	Number of resources of example type
explanation	int	Number of resources of explanation type
theory	int	Number of resources of theory type
activity	int	Number of resources of activity type
syntax	int	Number of resources of syntax type
unit	int	Identification number of unit

Table 4.4 Table *Session*

Field	Datatype	Description
idsession	**int**	**Identification number of session**
learner	int	Identification number of learner who has completed session
lesson	int	Lesson visited during session
sessiontime	time	Duration of the session
grade	int	Earned grade

Table 4.5 Table *Unit*

Field	Datatype	Description
idunit	**int**	**Identification number of unit**
name	varchar	Name of the unit
course	int	Identification number of course
lesson	int	Number of lessons

Table 4.6 Table *Course*

Field	Datatype	Description
idcourse	**int**	**Identification number of course**
name	varchar	Name of the course
unit	int	Number of units
lesson	int	Number of lessons
LSsuported	int	Learning styles which are suported
learner_num	int	Number of learners attending the course
duration	varchar	Duration of the course
teacher	int	Id of supervisor

Table 4.7 Table *Tag*

Field	Datatype	Description
idtag	**int**	**Identification number of tag**
idlearner	int	Identification number of learner
lesson	int	Identification number of lesson
Learning_object	int	Identification number of learning object
value	varchar	Entered tag
session	int	Identification number of session

4.2.2.2. The Structural Components of the Protus System Core

The core of the system includes five major components: ‘learner model’, ’session monitor’, ‘domain module’, ’application module’, and ‘adaptation module’, as can be seen in Figure 4.1.

Learner model. This module is a representation of information about an individual learner that is essential for an adaptive system to provide the adaptation effect - building of the learner model and tracking related cognitive processes are important aspects in providing personalization. All relevant data are part of a database as it is explained in previous chapter. Data from learner model in Protus is classified along three layers (Figure 4.3) that are suggested in (Klašnja-Milićević et al., 2011a):

- *Objective information*, which includes data supplied directly by the learner like: personal data, previous knowledge, preferences, etc. The learner edits this data during his/her registration to the system.

- *Learner's performance*, which includes data about cognitive style, level of knowledge of the subject domain, his/her misconceptions, progress and the overall performance for particular learner.
- *Learning history*, which includes information about lessons and tests learner has already studied, his/her interaction with system, the assessments he/she underwent, etc.

The system uses that information in order to predict the learner's behavior, and thereby adapt course to his/her individual needs.

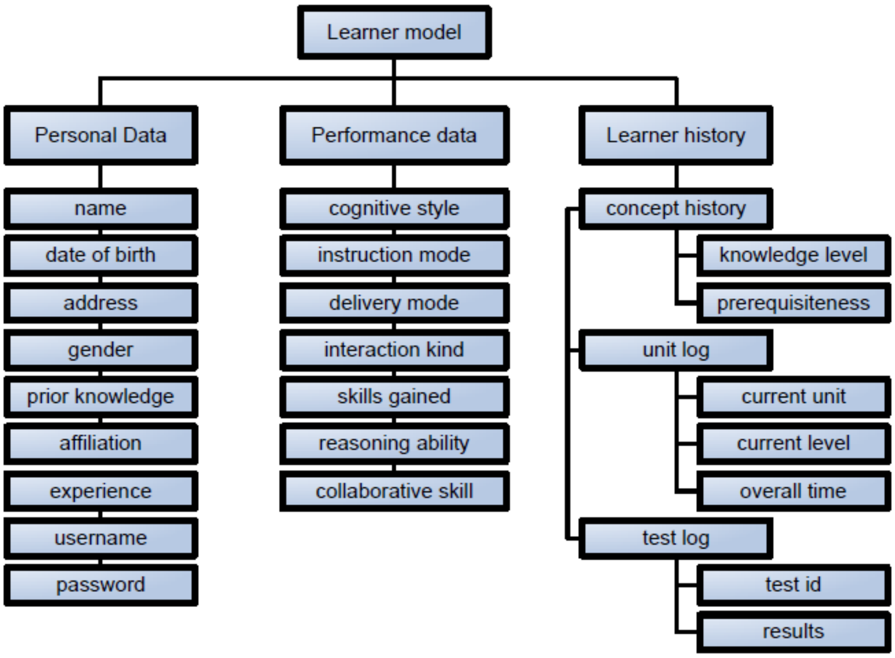

Figure 4.3 Layers in learner model

Session monitor. Within session monitor the system gradually builds the learner model during each session, in order to keep track of the learner's actions and his/her progress, detect and correct his/her errors and possibly redirect the session accordingly.

Domain module. This module is in fact storage for all essential learning concepts and objects in the domain, tutorials and tests. It describes how the information content is structured. The whole course is divided into six units, each of which consists of several lessons (Figure 4.4) (Klašnja-Milićević et al., 2011b). Every lesson (out of eighteen) contains several learning objects (LOs): theory session (tutorials), illustrative examples that supports theoretical part and additionally illustrates key concepts and numerous tests intended for checking learners acquired knowledge (Figure 4.5). All lessons, examples and tests are located in the file system on the server side, from where can be quickly loaded into the *jsp* page and present to learners. To every lesson unlimited number of examples and tests can be attached. Teachers can add new learning material using appropriate authoring tool.

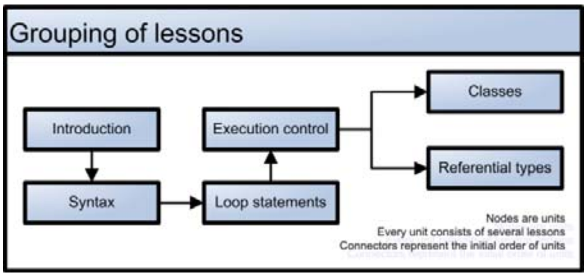

Figure 4.4 Grouping some of lessons and their interaction

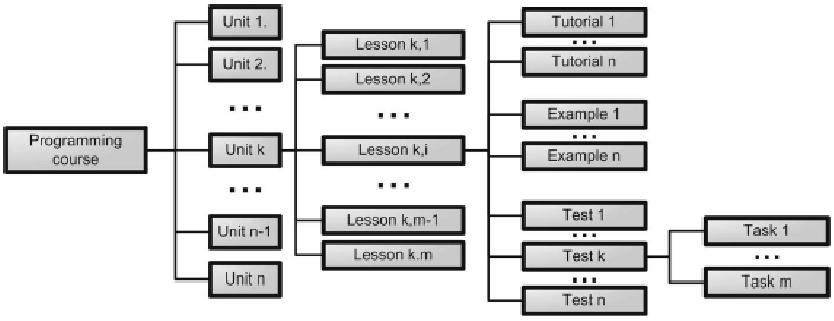

Figure 4.5 Course material hierarchy

Application module. This module applies different strategies and techniques to ensure tailoring of the learning content to the individual learners and personalized task and navigation sequencing. It supports a given pedagogical strategy. For example, that strategy often consists of selecting or computing a specific navigation sequences among the items based on the information contained in the learner model.

Adaptation module. This module follows the instructional directions specified by the application module and creates navigation sequence of learning objects recommended for the particular learner. These two components are separated in order to make adding new content clusters and adaptation functionalities easier. This module is responsible for building and updating learner model characteristics and for personalization of the application to the learner. It processes changing of learner's characteristics based on learner's activities and it provides an adaptation of visible aspects of the system for specific learner. Its main tasks also include storage and management of course data, ways of presenting course to learners, provision of reports and test results, etc. The adaptation module provides two general categories of personalization:

1. Personalization based on ratings of the frequent sequences, provided by the Protus and
2. Tag-based personalization.

Explanations for these two types of personalization, as well as extended consideration about the component for making recommendations will be given in the next section.

4.2.2.3. The Component for Making Recommendations

The ultimate goal of developing the Protus system has been increasing the learning opportunities, challenges and efficiency. Two important ways of increasing the quality of

Protus service are to make it intelligent and adaptive. The tags could be interesting and useful information to enhance recommender system's algorithms. They have ability to support learners in their own learning path by recommending tags and learning items, and also their ability to promote the learning performance of individual learners (Manouselis et al. 2011). The proposed framework for building automatic recommendations is composed of three modules (Figure 4.4):

- *A learner-system interaction module*, which pre-processes data to build learner models. The information from learners' registration form and learning style survey are collected in order to create an initial personal profile. The data about all learners' activities like sequential patterns, visited pages (tutorials, theory session or examples), test results and grades earned are collected within this module and saved into the server logs. The functionality available by clicking on an active learning object includes searching and categorization, as well as the ability to add tags or notes, and to modify/delete selected tags or notes. All information about tagging process are kept in tags repository and used for tag-based recommendation.
- *An off-line module*, which uses learner models on-the-fly to recognize learners' goals and content profiles. After appropriate learning style is determined for each learner, based on the initial survey, learning content is filtered, depending on the current status of the course, learner's affiliation and learners' tags. The offline module is launched periodically to perform all necessary calculations (user/cluster assignment, evaluation computation) – usually once a day. Calculations are performed using a current snapshot of input data, and all results are stored until all procedures are finished. The previously computed results are left intact at this stage. Upon completion, the old results are overwritten with the newly calculated data. This allows the recommendation engine, to be very quick, with computational complexity kept at minimum. That is because all requirements to be done online are to look up the user in the users/clusters assignment database and retrieve ranks and recommendations for that given cluster.
- *A recommendation engine*, which produces a recommendation list according to the:
    - learners' and experts' tags for each generated cluster and
    - the ratings of the frequent sequences, provided by the Protus system

From the filtered list of learning content the list of recommended actions is sent to alter learner–system interaction within a new session.

In the rest of this chapter, it will be explain an example of recommendation procedure. The learner signs up by using the registration form in order to create an initial personal profile or just logs in, if previously used the system. Each profile stores personal information supplied directly by the learner, i.e.: last name, first name, login, and previous knowledge in the form of ratings produced by the system, etc. (known as static information), and information about interests, dominant meaning words, and behavior in the form of learning sequences (known as dynamic information). The learner may change the static information at any time by editing it. When learners are registered to the system, their learning styles need to be tested. The Felder-Silverman learning style model

(FSLSM) is considered the most appropriate to be used in a computer-based educational system (Graf, 2010a). It describes the learning style on a more detailed level than the other models. By using dimensions instead of types, the strengths of students' preference towards a particular learning style can be represented. Moreover, FSLSM is based on tendencies, enabling the learning style model to consider exceptional behaviour. Furthermore, FSLSM is widely used in adaptive learning systems focusing on learning styles and some researchers even argue that it is the most appropriate model for the use in adaptive learning systems (Kuljis and Liu, 2005; Carver et al., 1999). Based on this model a corresponding psychometric assessment instrument was created. It was called the Felder-Solomon's Index of Learning Styles (ILS). It is a 44-item questionnaire where learners' personal preferences for each dimension are expressed with values between +11 to -11 per dimension, with steps +/-2. This range comes from the eleven questions that are posed for each dimension (Graf, 2007). This style indicates a preference for some presentation methods over others.

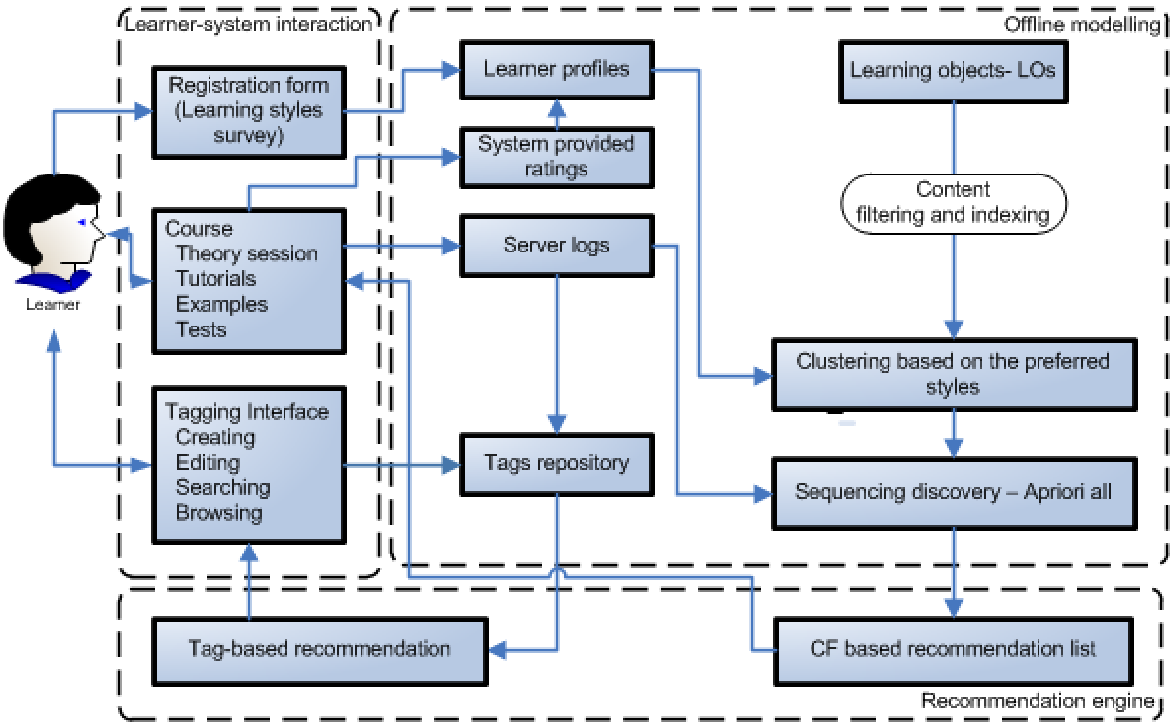

Figure 4.6 The recommendation component

According to the different combinations of learning styles it is possible to define clusters, which determined learner profiles. These results are stored in the learner model, which are used for the adaptation in Protus (Klašnja-Milićević et al., 2011b).

When a learner is registered or logged in, he/she can begin the process of learning. A learning session is initiated based on the learner's specific learning style and sequence of lessons is recommended to him/her. The learner can change the order of lessons he/she is attending. After selecting a lesson, from the collection available in Protus, the system chooses a presentation method for the lesson based on the preferred style. For the rest of the lesson, learners are free to switch among presentation methods using the media experience bar, which will be explained in detail in chapter 4.3.1. When the learner completes the sequence of learning contents, the system evaluates the learner's knowledge degree. The test contains several multiple-choice questions and code completion tasks. Protus then provides feedback to the learner on his/her answers and gives the correct solutions after the test. Recommendations cannot be made for the entire set of (all) learners in the same way, because even for learners with similar learning interests, their ability to solve a task can vary due to variations in their knowledge level. In our approach, we perform a data clustering technique as a first step to cluster learners based on their learning styles. These clusters are used to identify coherent choices in frequent sequences of learning activities.

Recommendation list can be created according to the ratings of these frequent sequences, provided by the Protus system. Also, a recommendation list can be created according to the learners' and experts' tags in every cluster, separately. During the learning process learner can tag each learning object. While Protus provides these two categories of personalization, experimental analyzes of this dissertation focus on the tag-based recommendation. The details of the whole process are presented in the rest of the paper.

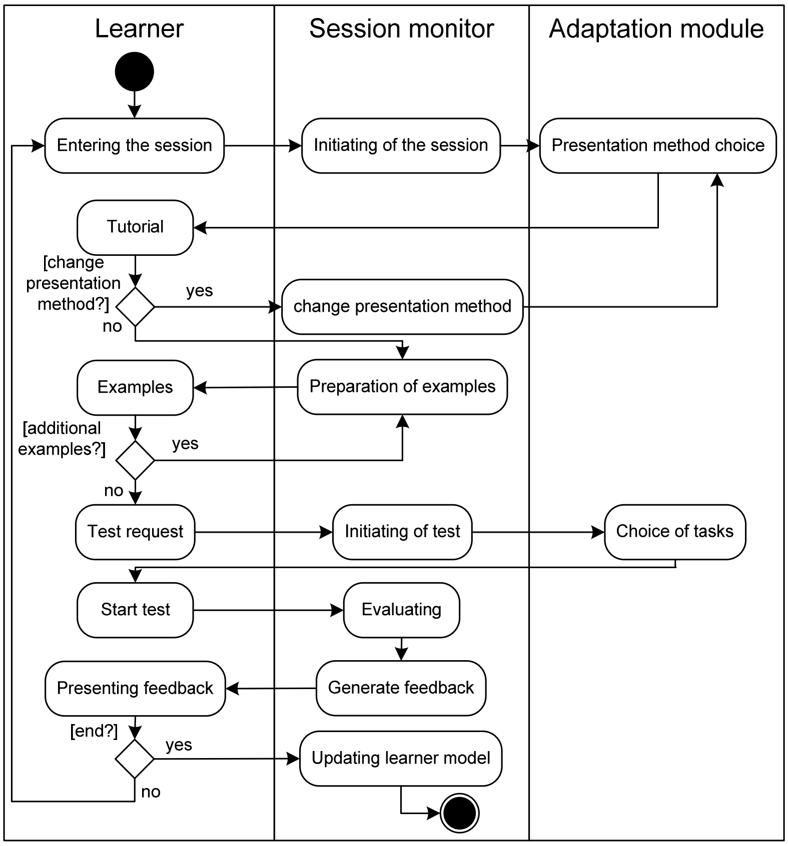

Figure 4.7 Structuring of lessons in Protus system

4.2.3. Content Filtering and Indexing

The learning content is divided into six units, each of which consists of several lessons (Figure 4.5). Every lesson (out of eighteen) contains several learning objects (LOs): theory session (tutorials), illustrative examples that supports theoretical part and additionally illustrates key concepts and numerous tests intended for checking learners acquired knowledge (Figure 4.7). Tutorials can contain various learning objects with different purposes (introductory, syntax rule presentation, block diagrams, etc.). To every lesson an unlimited number of examples and tests can be attached (Vesin et al., 2009). Their number can be increased by teachers using an appropriate authoring tool.

4.2.4. Adaptation to Learning Styles

It is obvious that different learners have different preferences, needs and approaches to learning. Psychologists call these individual differences learning styles. Therefore, it is very important to accommodate for the different styles of learners through learning environments that they prefer and find more efficient.

Learning styles can be defined as unique manners in which learners begin to concentrate on, process, absorb, and retain new and difficult information (Dunn et al., 1984). They are distinctive individual patterns of learning, which vary from person to person. It is necessary to determine what is most likely to trigger each learner's concentration, how to maintain it, and how to respond to his or her natural processing style to produce long term memory and retention. There are over seventy identifiable approaches to investigate and/or describe learning style preferences. One such data collection tool for investigating learning styles is Index of Learning Styles (ILS) by Felder and Soloman (Felder and Soloman, 1996).

Table 4.8 Characteristics of ILS based on Felder and Soloman (1996)

Active	**Reflective**
Work in groups	Work alone
Preference to try out new material immediately (Ask, discuss, and explain)	Preference to take time to think about a problem
Practical (Experimentalists)	Fundamental (Theoreticians)
Sensing	**Intuitive**
More patient with details	More interested in overviews and a broad knowledge (bored with details)
By standard methods	Innovations
Senses, facts and experimentation	Perception, principles and theories
Visual	**Verbal**
Preference to perceive materials as pictures, diagrams and flow chart	Preference to perceive materials as text
Global	**Sequential**
Prefer to get the big picture first	Prefer to process information sequentially
Assimilate and understand information in a linear and incremental step, but lack a grasp of the big picture	Absorb information in unconnected chunks and achieve understanding in large holistic jumps without knowing the details

The ILS is a 44-question, freely available, multiple-choice learning styles instrument, which assesses variations in individual learning style preferences across four dimensions or domains. These are Information Processing, Information Perception, Information Reception, and Information Understanding. Within each of the four domains of the ILS there are two categories (see Table 4.8):

- Information Processing: Active and Reflective learners,
- Information Perception: Sensing and Intuitive learners,
- Information Reception: Visual and Verbal learners,
- Information Understanding: Sequential and Global learners.

The preferred learning style can be investigated by offering the learner a free choice between an example, an activity or an explanation at first, and by observing a pattern in the choices he/she makes.

4.2.4.1. Information Processing: Active and Reflective Learners

Within Information Processing domain, we can distinguish example-oriented learners, called Reflectors, and activity-oriented learners, called Activists (Felder and Silverman, 2002). Active learners tend to retain and understand information best by doing something active with it - discussing or applying it or explaining it to others. Reflectors are people who tend to collect and analyze data before taking an action. They may be more interested in reviewing other learners' and professional opinions than doing real activities. In the Protus system, a learner with the active learning style is shown an activity first, then an example, explanation and theory. For the learner with the reflective style this order is different – he/she is shown an example first, then an explanation and theory, and finally he/she is asked to perform an activity.

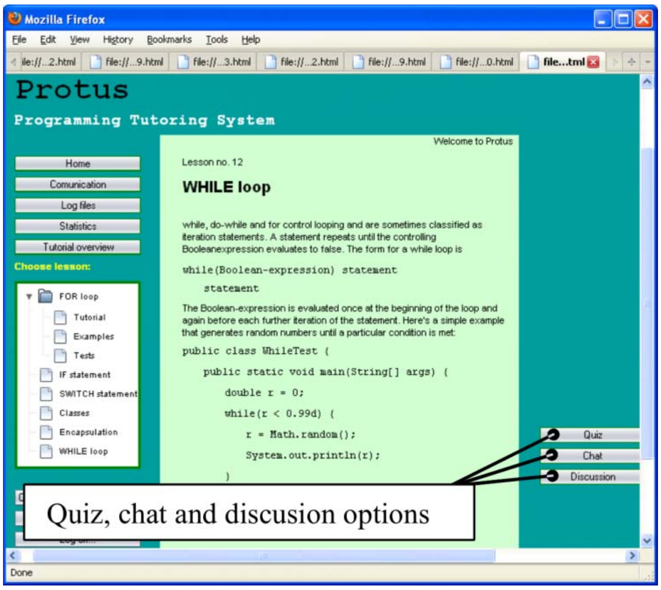

Figure 4.8a Active style presentation

For example, Figure 4.8 (a, b) shows a presentation of the lesson to the learner with active and reflective styles. A learner with the active learning style (Figure 4.8a) can participate in activities such as quiz, chatting, and discussion options whereas a learner with the reflective style (Figure 4.8b) is shown an example first before they are offered an action.

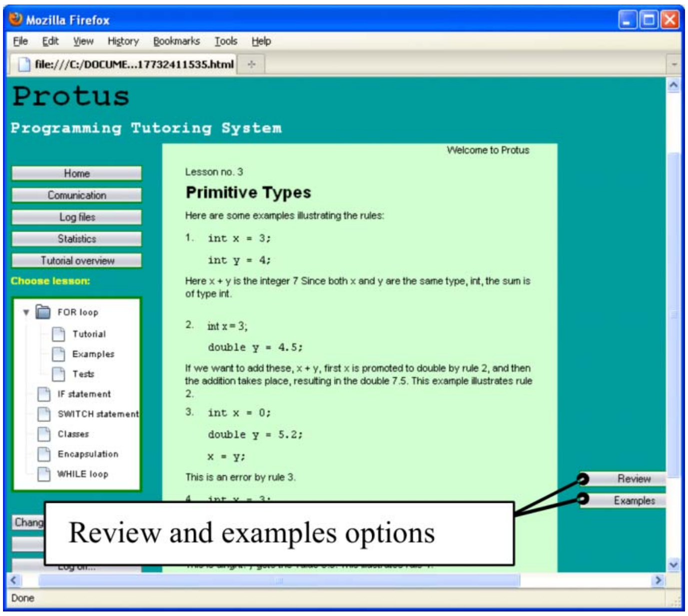

Figure 4.8b Reflective style presentation

4.2.4.2. Information Perception: Sensing and Intuitive Learners

Within Information Perception domain, sensing learners, called Sensors, tend to be patient with details and good at memorizing details and doing hands-on (laboratory) work. On the other hand intuitive learners, called Intuitors, may be better at understanding new concepts and are often more comfortable with abstractions and mathematical formulations than sensing learners.

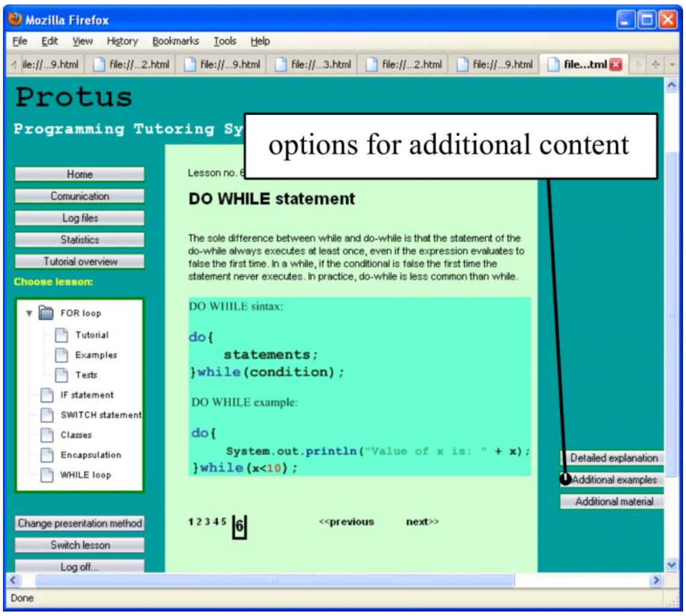

Figure 4.9a Sensing styles presentation

Sensors often prefer solving problems using well-established methods, and dislike complications and surprises. On the other hand, Intuitors like innovation and dislike repetition. Sensors tend to be more practical and careful than Intuitors. Intuitors tend to work faster and to be more innovative than Sensors. Presentation of the lesson to the learner with sensing and intuitive styles is given in Figure 4.9 (a,b) For example, it is assumed that sensing learners will be interested in additional contents. Therefore they may click the button for additional content on the interface (Figure 4.9a). Intuitors are provided with abstract content, formulas and concepts as shown in Figure 4.9b. Adequate explanations are given in form of block diagrams or exact syntax rules.

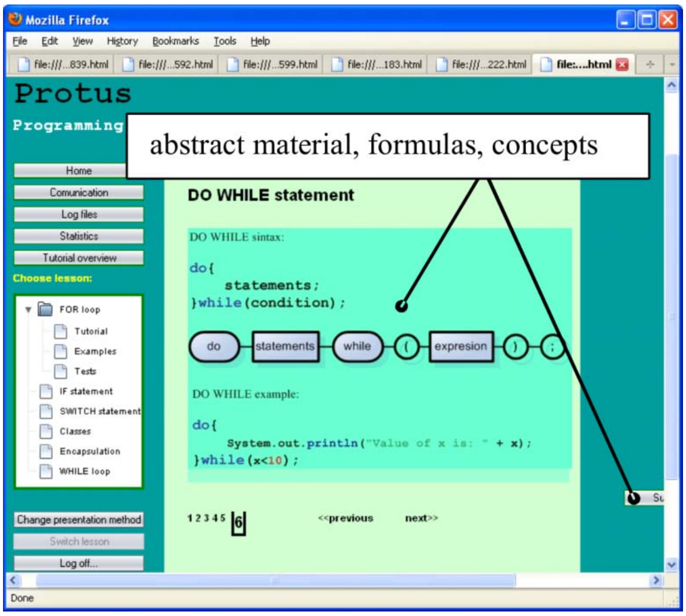

Figure 4.9b Intuitive styles presentation

4.2.4.3. Information Reception: Visual and Verbal Learners

Within Information Reception domain, Visual learners remember greatest what they see - pictures, diagrams, flow charts, time lines, and demonstrations. Verbal learners get more out of words - written and verbal explanations. Figure 4.10a shows a presentation of the topic of “String Declarations and Initialization” to a learner with a preference for textual material (verbalizer style). Figure 4.10b shows the presentation of the material to a learner with a visual preference. Based on the visual preference, the topic about the “FOR loop” is presented as a block-diagram.

4.2.4.4. Information Understanding: Sequential and Global Learners

Within Information Understanding domain Sequential learners tend to follow logical stepwise paths in finding solutions. On the other hand, Global learners may be able to solve complex problems quickly or put things together in novel ways once they have comprehended the big picture, but they may have difficulty explaining how they did it.

Sequential learners prefer to go through the course step by step, in a linear way with each step following logically from the previous one, while global learners tend to learn in large leaps, sometimes skipping learning objects and jumping to more complex material (Figure 4.11a,b). According to these characteristics of Sequential learning style, learners go through Protus’ lessons by a predefined order (Figure 4.11a). On the other hand, Global learners are provided with overall view of the course, with short explanations of each unit and options for accessing the unit they are interested in by clicking the unit hyperlinks rather than following sequential order (Figure 4.10b).

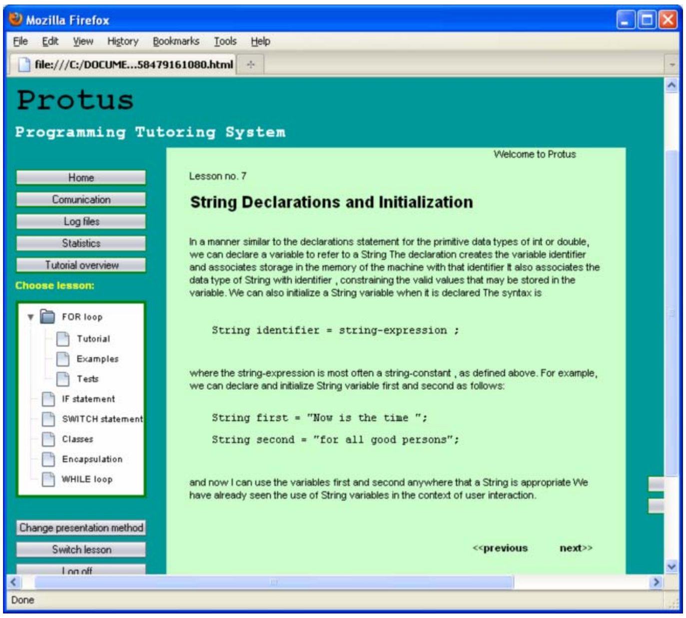

Figure 4.10a Verbal styles presentation

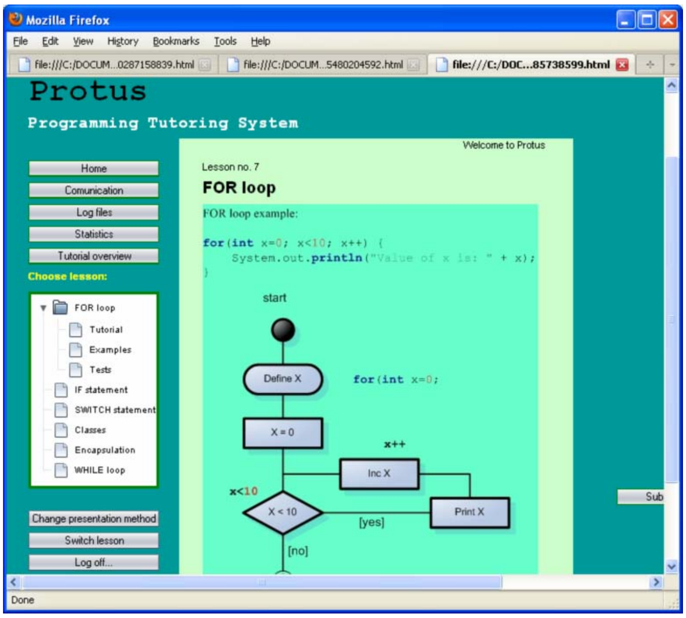

Figure 4.10b Visual styles presentation

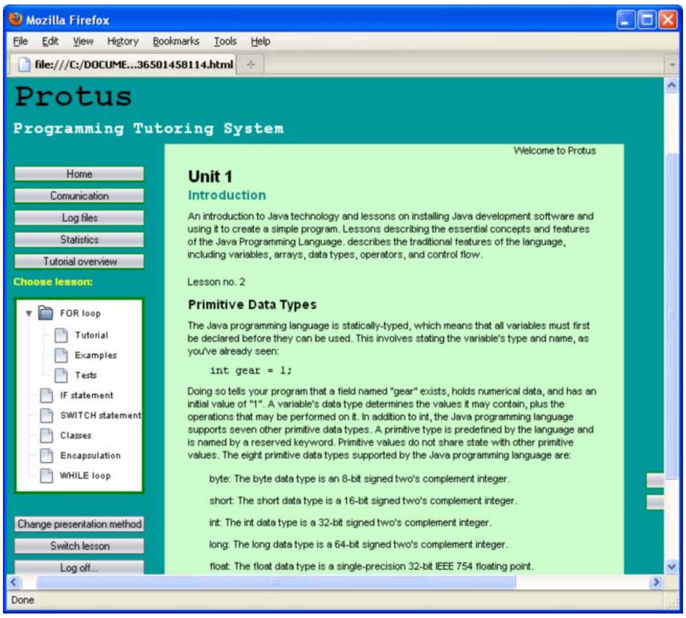

Figure 4.11a Sequential styles presentation

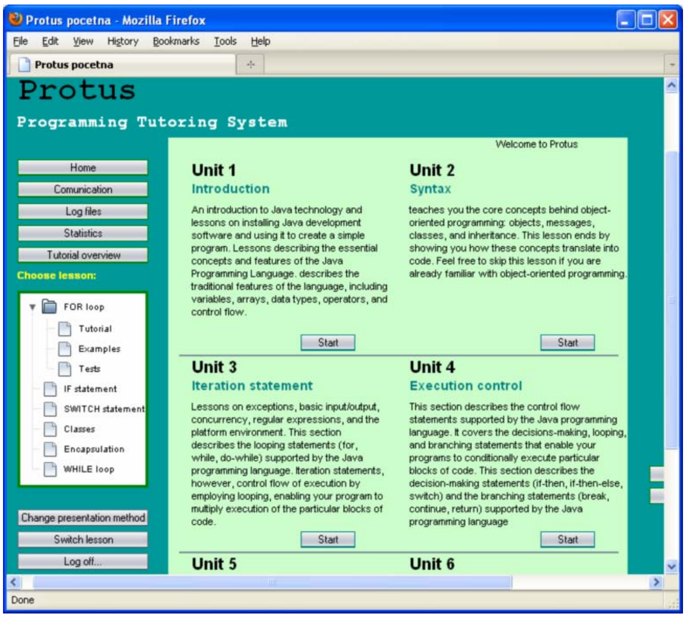

Figure 4.11b Global styles presentation

4.2.5. Clustering Based on Preferred Learning Styles

Based on the results of the ILS questionnaires it was possible to define appropriate clusters, which determined groups of learners with the similar learning style preferences.

In our approach, clusters were formed for different combinations of learning styles within the three categories, so we have 8 (2^3) clusters. Category *Information processing* was omitted in order to increase the number of learners in a separate cluster and to obtain more relevant data for recommendations. In future research, when increasing the number of learners participating in the experiment it can be taken into account.

4.2.6. Identification of Sequences of Learning Activities and Personalized Recommendation

In contrast to traditional classroom-based learning, the learning behavior in web-based environments is more determined by the learner's own decisions how to organize learning process (Northrup, 2001). Learners could follow different paths based on their preferences and generate a variety of learning activities. All these variations in series of learning activities are noted down by the Protus system.

In order to investigate learning activities in detail, sequential pattern mining algorithm of AprioriAll (Tong and Pi-Lian, 2005) is adopted to extract behavioral (interaction) patterns from the log file. These patterns will be useful to analyze how learners evolve from the beginning of learning of particular unit, until they successfully finish it, or less successfully, give up. Learners with different learning styles have different sets of frequent sequences. Hence, learners were clustered based on their learning styles and then behavioral patterns were discovered for each learner by AprioriAll algorithm.

4.2.6.1. The Process of Mining Sequential Patterns by Apriori All Algorithm

Let $I = \{i_1, i_2, \ldots, i_m\}$ be a set of learning objects, called *items*. An *itemset* is a non-empty set of *items*. A sequence is an ordered list of *itemsets*. We denote an *itemset* i by $i_1, i_2, \ldots, i_m$, where i_j is an item. We denote a sequence *s* by $(s_1, s_2, \ldots, s_n)$, where s_j is an itemset. A sequence $\langle a_1, a_2, \ldots, a_n \rangle$ is contained in another sequence $\langle b_1, b_2, \ldots, b_n \rangle$ if there exist integers $\langle k_1 < k_2 < \ldots < k_n \rangle$ such that $a_1 \subseteq b_{k_1}, a_2 \subseteq b_{k_2}, \ldots, a_n \subseteq b_{k_n}$.

In a set of sequences, a sequence *s* is *maximal* if *s* is not contained in any other sequence (Agrawal and Srikant, 1995). A learner supports a sequence *s* if *s* is contained in the learner-sequence for this learner. The support for sequence is defined as the fraction of total learners who support this sequence.

Given a database D of learners' access transactions, the problem of mining sequential patterns is to find the maximal sequences among all sequences that have a certain learner-specified minimum support. Each such maximal sequence represents a *sequential pattern*. We call a sequence satisfying the minimum support constraint a *large* sequence.

The process of mining sequential patterns can be split into five phases (Agrawal and Srikant, 1995). To conveniently explain them, we use a small part of the database, as shown in Table 4.9. Each transaction consists of the following fields: learner-id, access-time, and the access path in the transactions. Phases are:

- Sort phase: The original database is sorted with learner-id as the major key and access time as the minor key. Table 4.9 shows the result set of learner sequences after sorting.
- Large-itemsets (l-itemsets) phase: In this phase we find the set of all large itemsets. Without loss of generality, we assume that the set of l-items is mapped to a set of consecutive letters. Suppose the minimal support is 60%, and the minimal support customer sequence is thus 3. The result of large 1-itemsets is listed in Table 4.10.
- Transformation phase: In a transformed learner sequence, each transaction is replaced by the set of l-itemsets contained in that transaction. If a transaction

does not contain any l-itemset, it is not retained in the transformed database. This transformed database is shown in Table 4.11.

- Sequence phase: This is an essential phase of the process. In this phase, an algorithm uses a set of large itemsets to find the desired sequences. The idea is that, given the l-itemsets, the set of all the sequences with minimum support should be found. In each pass, we use the large sequences from the previous pass to generate the candidate sequences, and then measure their support by making a pass over the database. The first pass over the database is made in the l-itemset phase, and we determine the large 1-sequences shown in Table 4.10. The large sequences together with their support at the end of the third and fourth pass are shown in Table 4.12 and Table 4.13, respectively.
- Maximal phase: to reduce information redundancy, the sequential patterns contained in other sequential patterns are pruned (see algorithm in Listing 1). Table 4.14 shows Maximal Large 5-Sequences, after pruning.

Table 4.9 Database sorted by learner-id and transaction time

Learner-id	Access-time	Access path
1	2010. January 20.	Lesson1 introduction, Lesson1 overview, Lesson 1 theory
1	2010. January 22.	session Lesson1 exercise, Lesson 1 syntax rule,
1	2010. January 23.	Lesson 1 example 2, Lesson 1 example 3
1	2010. January 24.	Lesson 1 example 1, Test 1
2	2010. January 15.	Lesson 1 overview, Lesson 1 theory session
2	2010. January 16.	Lesson 1 example 3, Lesson 1 example 1,
2	2010. January 17.	Test 1
3	2010. January 18.	Lesson 1 theory session
3	2010. January 20.	Lesson 1 syntax rule
3	2010. January 21.	Lesson 1 example 3, Lesson 1 example 1
3	2010. January 22.	Lesson 1 exercise, Test 1
4	2010. January 21.	Lesson1 introduction, Lesson1 overview, Lesson 1 theory
4	2010. January 23.	session, Lesson1 exercise,
4	2010. January 24.	Lesson 1 syntax rule,
4	2010. January 26.	Lesson 1 example 2, Lesson 1 example 3, Test 1
5	2010. January 16.	Lesson1 introduction, Lesson1 overview
5	2010. January 17.	Lesson 1 theory session
5	2010. January 18.	Lesson1 exercise
5	2010. January 19.	Lesson 1 example 1, Lesson 1 example 2, Test 1

Listing 1. Algorithm for pruning of sequential patterns

```
for all sequences  c ∈ C_k  do
   for all (k−1)- subsequences s of c do
         if (s ∉ L_{k−1})then
               delete c from C_k;
```

Table 4.10 Large Itemsets

Large itemsets	Mapped to
Lesson1 introduction	a
Lesson1 overview	b
Lesson 1 theory session	c
Lesson1 exercise	d
Lesson 1 syntax rule	e
Lesson 1 example 1	f
Lesson 1 example 2	g
Lesson 1 example 3	h
Test 1	i

Table 4.11 Transformed database

Learner id	Mapped to
1	<(abc)(de)(gh)(fi)>
2	<(bc)(hf)i>
3	<ce(hf)(di)>
4	<(abc)de(ghi)>
5	<(ab)cd(fgi)>

Table 4.12 Large 3-sequences

Sequence	Support
abc	3
abd	3
abg	3
abi	3
abh	3
bcd	3
bce	3
bcg	3
bch	3
bcf	3
bdg	3
bdh	3
bgh	3
bhi	4
bci	4
ceh	4
cdg	3
cdi	4
chi	3
ghi	3
hfi	3
ehf	3
dgi	3
dgh	3

Table 4.13 Large 4-sequences

Sequence	Support
abcd	3
abcg	3
abch	3
abgh	3
abhi	3
abdg	3
abdh	3
bcgh	3
bchi	3
bghi	3
cdgh	3
cehf	3
cdgi	3
dghi	3

Table 4.14 Maximal Large 5-Sequences (after pruning)

Sequences
<abcgh>
<abchi>
<abghi>

4.2.6.2. Evaluation Process

When the learner completes the sequence of learning materials, the Protus system evaluates the learner's acquired knowledge. The learners' ratings can be interpreted according to the percentage of correct answers, as follows:

- 10 (excellent) - (90–100%)
- 9 (good) - (80–89%)
- 8 (average) - (70–79%)
- 7 (passing) - (60–69%)
- 6 (marginal) - (50–59%)

This grading scale is based on our university grading system. Consequently, learners have a better sense of having mastered the material using this system of evaluation. The system can be easily transformed and adapted to other standards of grading. Two learners are said to be similar to each other if they are evaluated by the system with the same ratings for a similar navigational sequence. Recommendation process can be carried out according to these learning sequences based on the collaborative filtering (CF) approach.

4.2.6.3. Recommendation Process Based on Collaborative Filtering

The task of a collaborative filtering system is to predict the usefulness rating of a particular learner *l* for a similar learner *l'* (Herlocker et al., 2004). Therefore, the rating vector of a learner *l* is represented by $R_l = (r_{l1}, r_{l2}, \ldots, r_{li})$. The entry r_{lj} of R_l is provided by the *Protus* system to indicate the learner's knowledge degree for the unit he/she is currently used in learning process.

The collaborative filtering system compares the learner's ratings with the ratings of all other learners, who have been rated. Then a weighted average of the other learners rating is used as a prediction. If S_l is set of frequent sequences that a learner *l* has been rated for, then we can define the mean rating of learner *l* as:

$$\overline{r_l} = (1/|S_l|)\sum_{i \in S_l} r_{li}$$

When Pearson correlation (Herlocker et al., 1999) is used, similarity is determined from the correlation of the rating vectors of learner *l* and the other learner *l'*. This value measures the similarity between the two learners' rating vectors.

$$\rho(l,l') = \left(\sum_{i \in S_l \cap S_{l'}} (r_{li} - \overline{r_l}) \cdot (r_{l'i} - \overline{r_{l'}})\right) \Bigg/ \left(\sqrt{\sum_{i \in S_l \cap S_{l'}} (r_{li} - \overline{r_l})^2 \cdot \sum_{i \in S_l \cap S_{l'}} (r_{l'i} - \overline{r_{l'}})^2}\right)$$

The prediction formula is based on the assumption that the prediction is a weighted average of the other learners' ratings.

$$p^{col}(l,i) = \overline{r_l} + k_{li} \sum_{l \in L_i} \rho(l,l')(r_{l'i} - \overline{r_{l'}})$$

where *Li* - is the set of learners who were rated for sequence *i*; the factor k_{li} is used to normalize the weights.

$$k_{li} = 1 \Big/ \left(\sum_{l' \in L_i} \rho(l,l')\right)$$

When this procedure is executed, the Protus system can recommend relevant links and actions to target learner during the learning process based on similarities with other learners. The system can be considered successful if the observed learner is rated with a similar grade.

4.2.7. Tag-based Personalized Recommendation Using Ranking with Tensor Factorization Technique

The task of tag-based personalized recommendation is to provide a learner with a personalized ranked list of tags for a specific item. We have implemented the recommendation component of the Protus system that recommends the most popular tags and experimentally compare it with the previous version of the system, which will be shown in chapter 5. On the basis of comprehensive comparisons of techniques that can be used to recommend tags, in the rest of this chapter we will show the possibility of implementing the system using Ranking with Tensor Factorization (RTF) technique, as analyzed in the section 3.3.4. The recommendation process consists of three phases:

- Generating initial tensor
- Computing tensor factorization
- Generating a list of recommended items

4.2.7.1. Generating Initial Tensor

To generate the initial tensor, we have been used 3-dimensional data of learners, items (learning objects) and tags. The third-order tensor $\mathcal{A} \in R^{I \times J \times K}$ represents this data where I, J and K are the dimensions of the data of learners, items and tags, respectively. A value $(\mathcal{A})_{ijk} = a_{ijk}$ can represent, for example, how many times learner i tagged an item k with a tag j. In this phase following steps can be recognized:

- A learner set is generated.
- Set of tags is resolved. These are the tags used by the learners.
- Item set is resolved. These are the items tagged with the tags by the learners.
- Iterate through all the learners, tags and items. Resolve if a current item is tagged by the current tag and learner. If so – mark the existing relationship in the tensor.
- Store the empty relations (if a learner does not tag an item with a tag) of a current learner. These relations will be used to resolve the recommendations.

4.2.7.2. Computing Tensor Factorization

To compute a tensor factorization, the initial tensor has to be defined, and then the following steps should be applied:

- Firstly, the initial tensor is split into the three mode matrices.
- Secondly, the dimensions are reduced for each mode matrix. These reduced matrices are multiplied to compute a core tensor.
- Finally, the reduced matrices are transformed, multiplications are applied. The factorized tensor is computed.

4.2.7.3. Generating a List of Recommended Items

When the factorized tensor is computed, the recommendations can be determined. The task of tag recommendation is to predict which tags a learner is most likely to use for tagging an item. That means a tag recommender has to predict the numerical values of the factorized tensor indicating how much the learner likes a tag for an item. Instead of predicting single elements the system should provide the learner a personalized list of the best N tags for the item.

4.3. Protus Interface

Two main roles exist in the system, intended for two types of system's users:

• learners - they are taking the Java programming course and will be using the system in order to gain certain knowledge and

• teachers and content authors - the lesson and learner database administrator; they track learning process of learners and help them with their assignments.

Therefore, separated user interfaces are provided for learner and teacher (Vesin et al., 2007). Teacher's interface helps in process of managing data about a learner and course material. Learner's interface is a series of web pages that provide two options: taking lessons and testing learner's knowledge. All data about learner and his progress in the course, as well as data about tutorials, tests and examples are stored on the system's server.

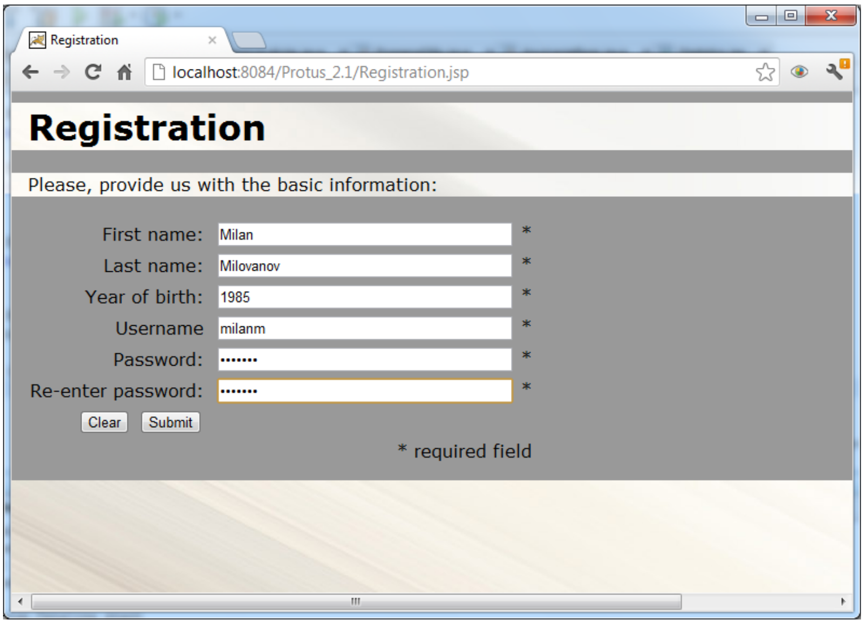

Figure 4.12 Registration form of the Protus system

4.3.1. Learner's Interface

This section describes the user interface and explains the guidelines that were taken into account for its design. The new learner registers in the system by using an appropriate form (Figure 4.12) in order to create a personal profile. After that, learners are ought to answer short optional survey where they need to choose their own preferred learning style that indicates a preference for some previously mentioned presentation methods over others. These results are stored in a learner model, which will be used for the initial adaptation in Protus (Klašnja-Milićević et al., 2009).

When a learner is logged in, a session is initiated based on learner's specific data and sequence of lessons is recommended to him/her. Lessons are grouped into units. Initial order of lessons in implemented Java programming course is presented in Figure 4.5. A learner has possibility to change order he/she will attend lessons. After selecting a lesson, from the collection of lessons available in Protus (Figure 4.13), system chooses presentation method of lesson based on the learner's preferred style.

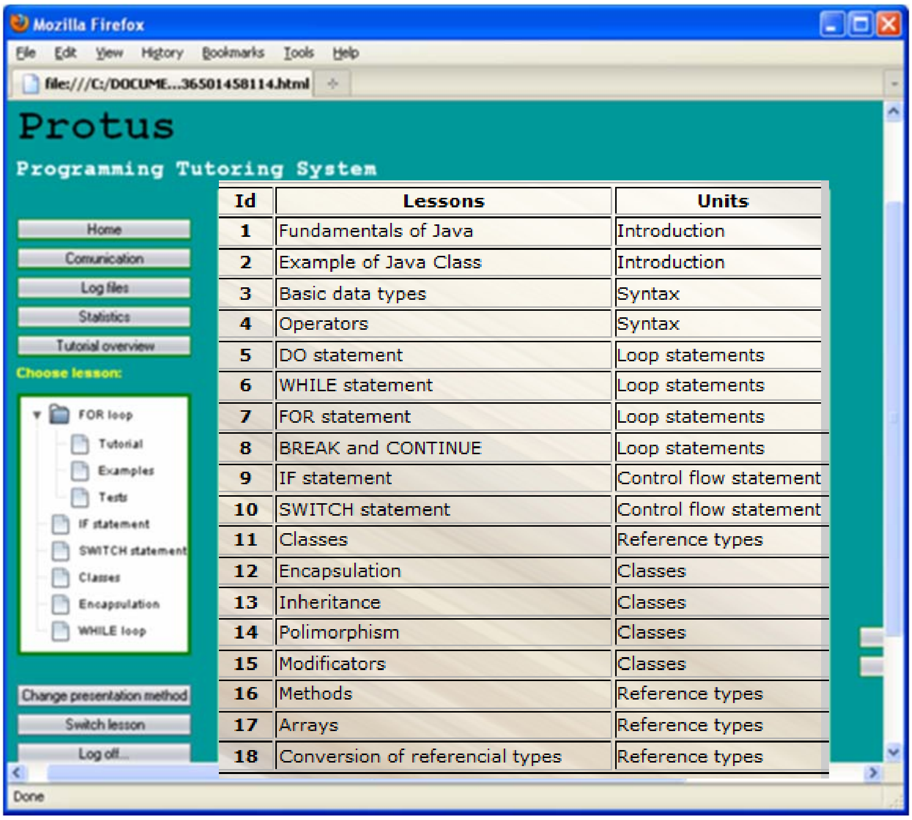

Figure 4.13 Collection of available lessons in the Protus system

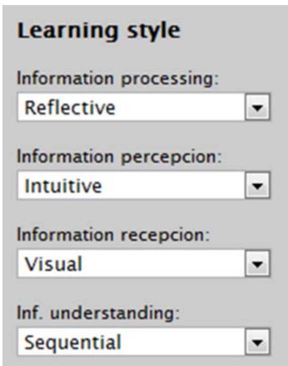

Figure 4.14 Experience bar

Different researches have shown that learning style may change depending on the task that the learner has mastered. Also, learning style may change according to the content of learning. Therefore, it is counterproductive to leave the user's learning style unchanged throughout the whole course. For the rest of the lesson, learners were free to switch between presentations methods by using the experience bar (Figure 4.14).

For every lesson the same sequence of activities has to be followed. At the beginning of a lesson, participants are pre-tested with multiple choice questions and fill-in appropriate entering questionnaire. In the next step, participants are shown a short introductory text on the lesson's topic (Figure 4.15), additionally explained with appropriate examples (Figure 4.16).

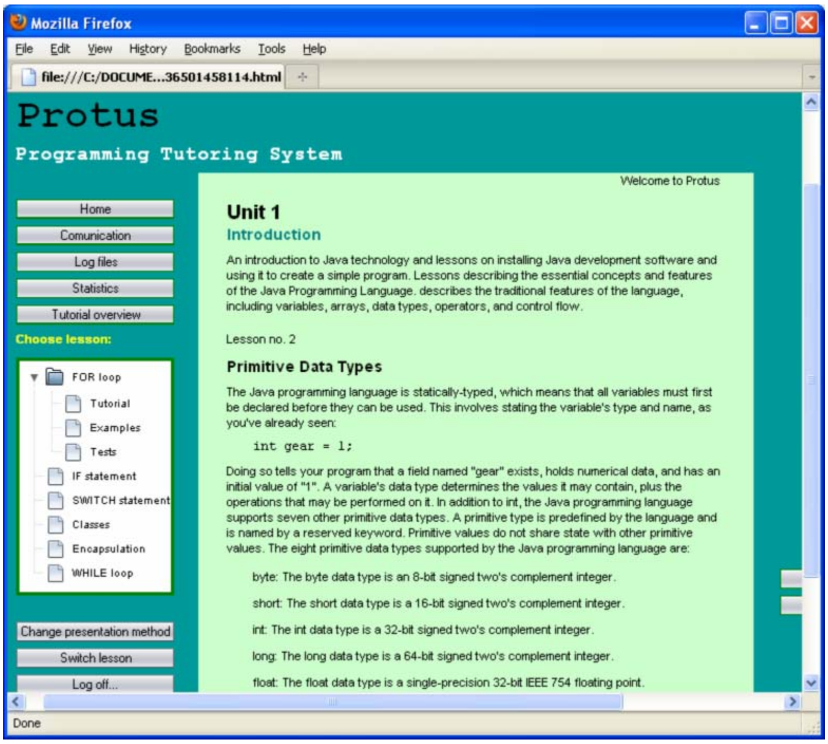

Figure 4.15 Lessons tutorial

At the end of each lesson a post-test should be conducted. Test contains several multiple-choice questions and code completion tasks. This time Protus provides feedback on their answers and gives the correct solutions after the test. The post-test section is followed by a lesson summary and a lesson feedback form. Through this form (Table 5.5 in section 5.7), participants rate the presentation method(s) and answer questions about their perceived enjoyment, progress, and motivation. Given grades are in range from 1 (lowest satisfaction) to 5 (highest satisfaction). When the learner completes the sequence of learning materials, the system evaluates the learner's knowledge degree. When the learners have visited all lessons within one unit they have to fill-in a final test to evaluate their accepted knowledge about the unit.

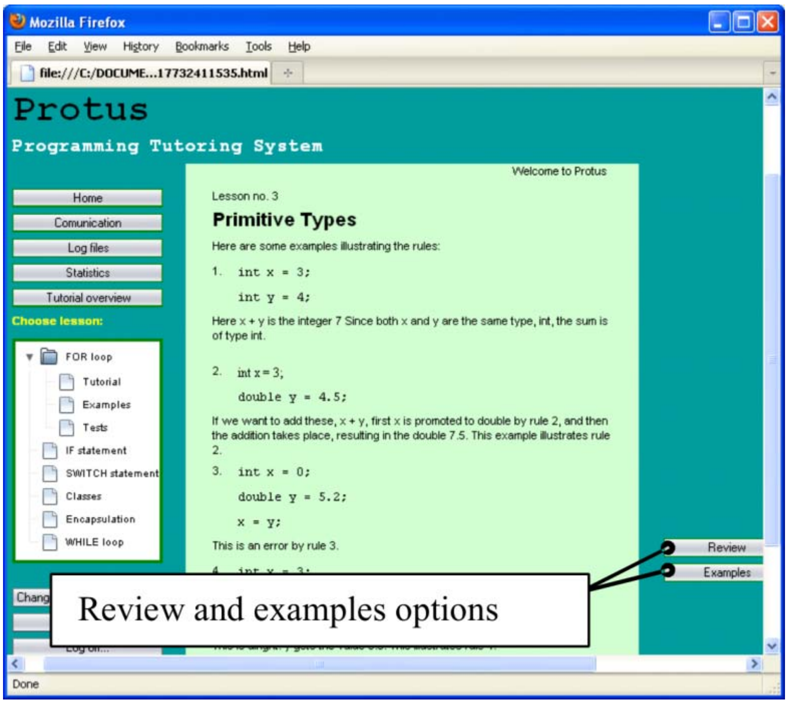

Figure 4.16 Lessons example

4.3.2. Teacher's Interface

Besides being beneficial for providing learners with personalized learning experience, the Protus system is also useful for generating feedback for other participants in the learning process - administrator and/or teachers (teachers). Protus can be used to provide feedback to teachers about the learners' activities, their performance, achieved collaboration level and the similar activities. In both cases, the feedback helps in improving the learning process. Teachers have access to special functions within Protus system. There are two levels of privilege. First is the higher level - a level with unlimited possibilities. A teacher with this privilege level can enter information about new teachers and data necessary for their connection. Other teachers are limited to only change the data of learners. All teachers can access and have access to the data of all active learners.

When teacher logs into a system, it has to enter a username and password (Figure 4.17). After correctly entered the username and password, the application launches and the main screen of teacher's part of the system appears (Figure 4.18).

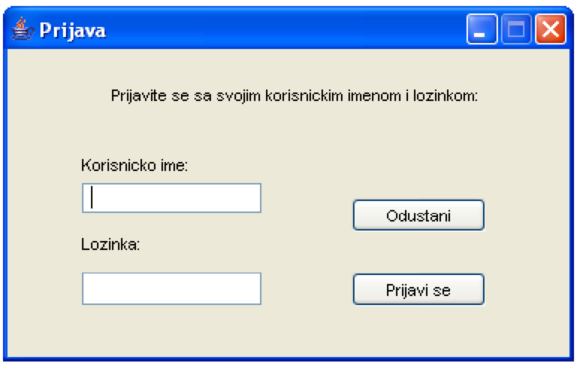

Figure 4.17 Login window for teachers

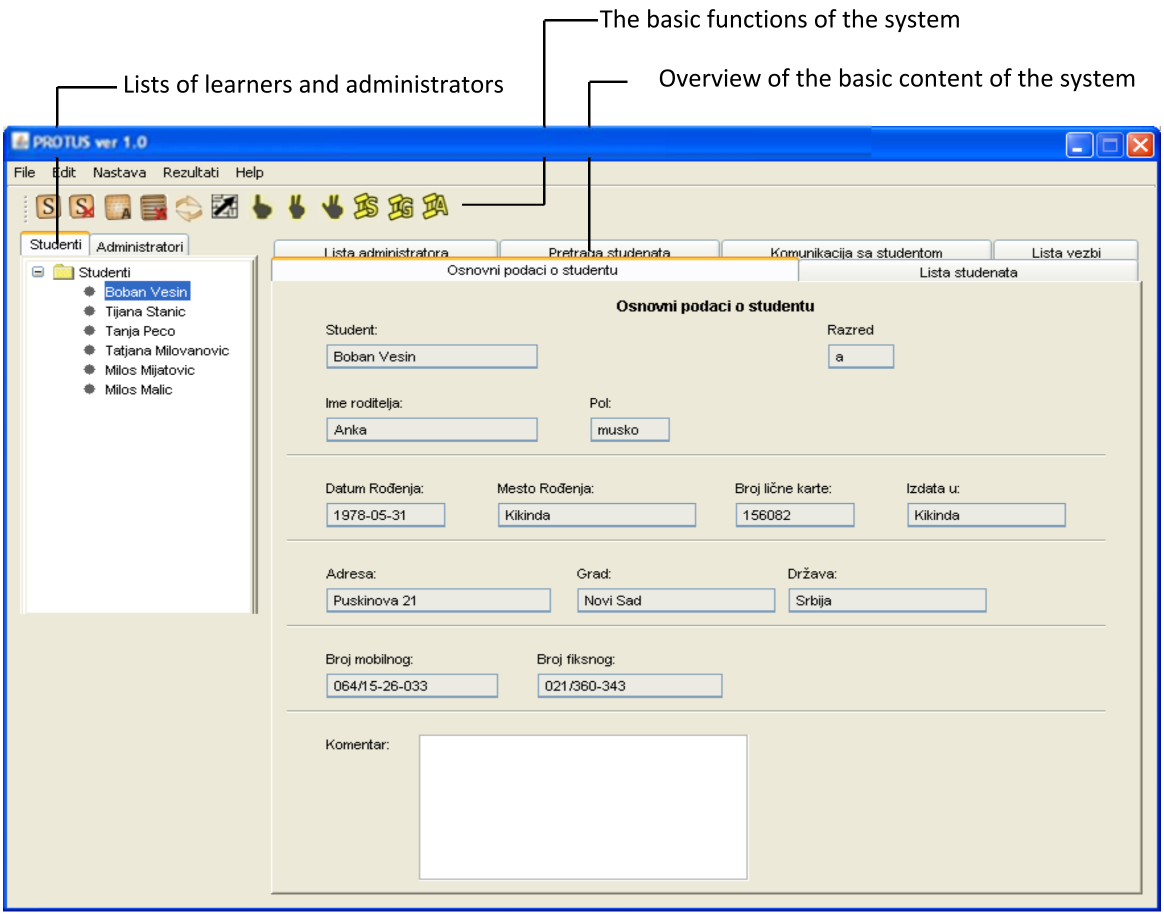

Figure 4.18 The main window of the system

During the startup of the database basic information about learners and teachers is loaded automatically. These data are entered into the appropriate lists. Protus aims at helping teachers rethink the quality of the learning content and learning design of the course they teach. To this end, the system provides teachers with feedback about the relevant aspects of the learning process taking place in the online learning environment they use. The provided feedback is based on the analyses of the context data collected in the learning environment.

In panel *'A list of learners'* (Figure 4.19), list of all learners can be loaded from the database. By entering arbitrary data into the appropriate field (name or learner identification number) and press the search button, the remaining data about learner are entered into the proper lists. Edit button opens a dialog for editing information about the learner (Figure 4.20). Each of the fields is ready to enter the new data. Data on gender, class, and country of residence of the learner is entered by selecting one of the options. If an unauthorized user wants to make changes to the data, the system prevents it and he/she receives a warning message.

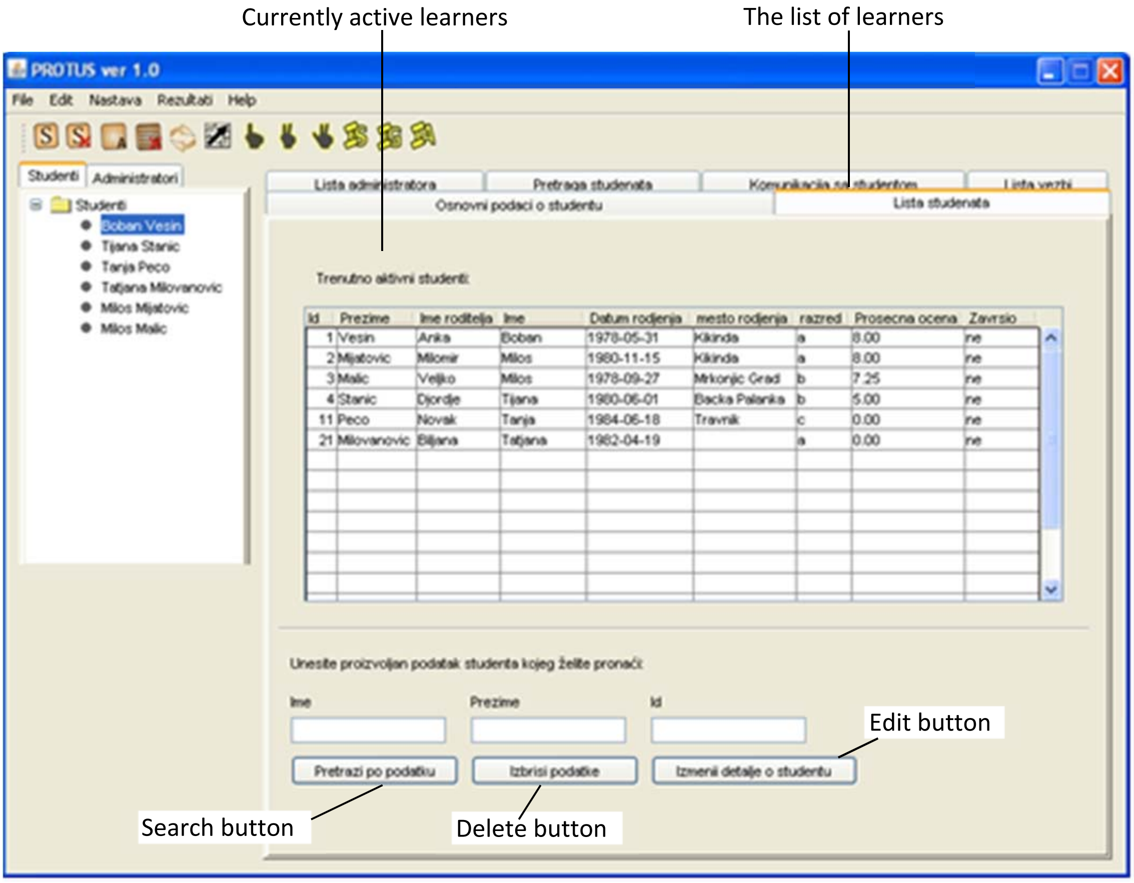

Figure 4.19 Panel 'List of learners'

Panel 'A list of administrator' displays a list of administrators with basic information and allows searching in the same way as for learners.

In the case of a large number of learners, selecting specific groups of learners who meet specific criteria is difficult. For this purpose, 'Search' panel is very useful (Figure 4.21). The database searching can be performed based on the following criteria:

1. Selecting the gender of required learners. It is also offered a search feature, regardless of gender.
2. Choosing the average score from which the required learner must have a higher or lower grade point. It is also offered a search feature, regardless of ratings.
3. Choosing a search among graduates and among those learners whose course is still in progress.
4. Selecting the number of passed lessons of required learners.

Search results (list of learners who meet selected criteria) are presented in the table (Figure 4.21).

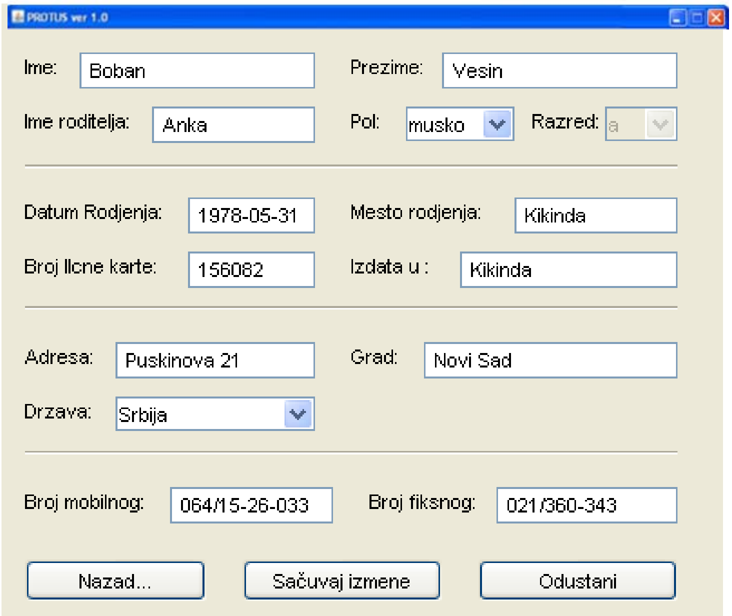

Figure 4.20 Framework for editing learner

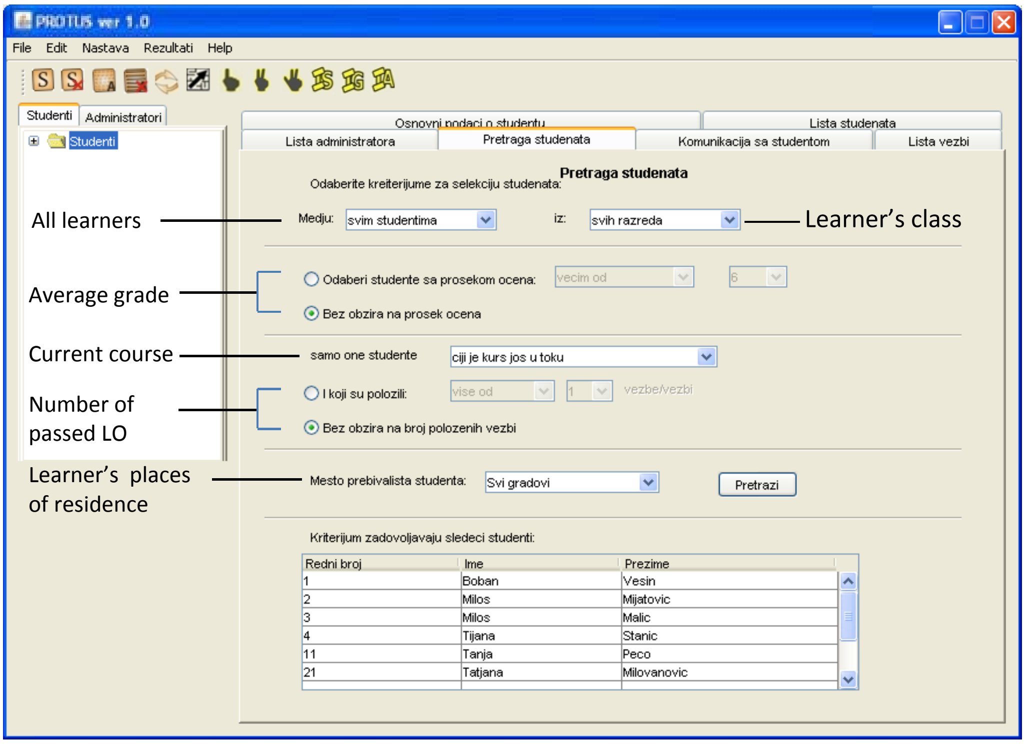

Figure 4.21 Panel 'Learners search'

In particular, Protus provides statistical reports to the teacher. These reports can inform teacher about the activities the learners performed during the learning process. Figure 4.22 depict the graphic interface where the assessment and tracking data and statistics are generally shown. This form facilitates data retrieval and provides appropriate results for the teacher. The teacher can combine parameters and filters in order to obtain reports that will be presented in form of charts and tables. The chart type varies according to the selected filters. For example, the teacher could know what specific material was more used by learners, what kind of learning style they preferred or what grades they earned for every particular lesson. These reports can show results for group of learners or for every learner separately.

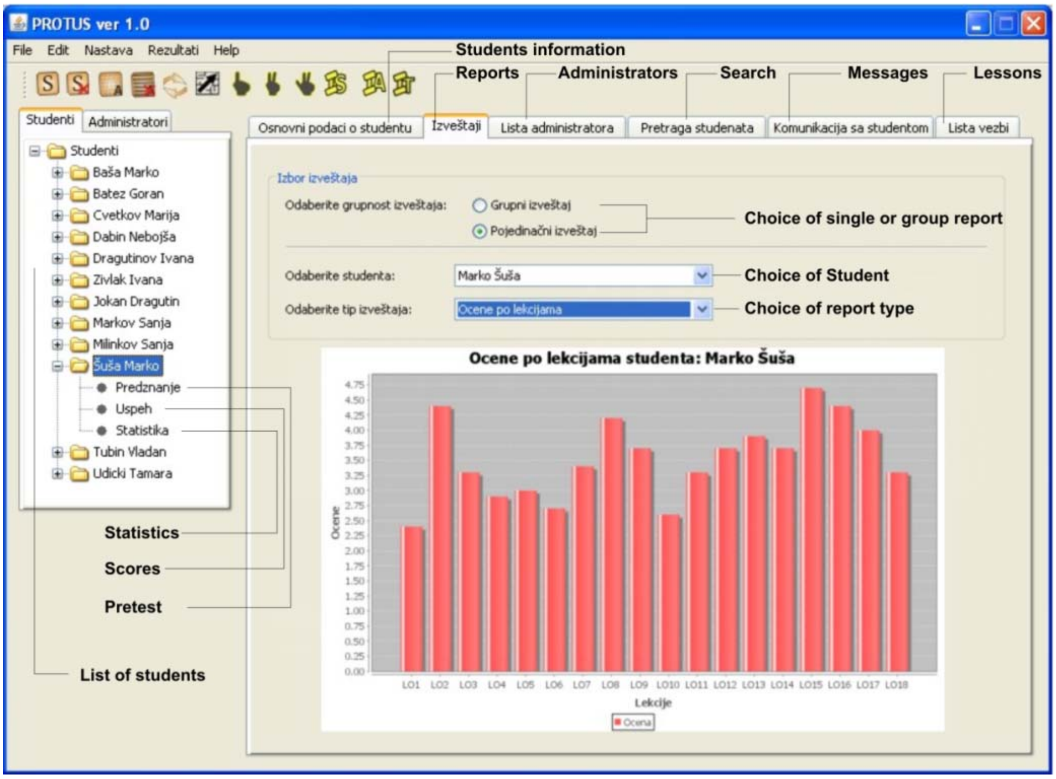

Figure 4.22 Teacher's interface

4.3.3. Tagging Interface

As we have summarized in Chapter 3.3 collaborative tagging activities caused the appearance of tag-based profiling approaches, which assume that users expose their preferences for certain contents through tag assignments. Thus, the tags could be interesting and useful information to enhance recommender system's algorithms. The innovation with respect to the e-learning system lies in their ability to support learners in their own learning path by recommending tags and learning items, and also their ability to promote the learning performance of individual learners (Manouselis et al., 2011).

Learners could benefit from writing tags in several important ways. Tagging is proven to be a meta-cognitive strategy that involves learners in active learning and engages them with more effectively in the learning process. As summarized by Bonifazi et al. (2002), tags could help learners to remember better by highlighting the most significant part of a text, could encourage learners to think when they add more ideas to what they are reading, and could help learners to clarify and make sense of the learning content while they try to reshape the information. Learners' tags could create an important trail for other learners to follow by recording their thoughts about specific learning material and could give more comprehensible recommendation about the learning process. The viewing of tags used on a webpage can give a learner some idea of its importance and its content.

The information provided by tags makes available insight on learner's comprehension and activity, which is useful for both learners and teachers. Tagging, by its very nature, is a reflective practice which can give learners an opportunity to summarize new ideas, while receiving peer support through viewing other learners' tags/tag suggestions. Tagging provides possible solutions for learners' engagement in a number of different annotation activities - add comments, corrections, links, or shared discussion.

To create a tag in Protus the learner simply starts by clicking on active learning object in the content and enter arbitrary keywords in the appropriate textfield. The system allows participants to enter as many tags as they wish, separated by commas. This makes it possible to use spaces in tags, rather than restricting the participant to a single word. This is in contrast to many popular tagging systems which only allow single word tags. We allow the use of multi-word tags to eliminate the problem of establishing a convention for word combination.

Whenever the learner returns to that particular learning object, the list of tags he/she has previously made will re-appear, as shown in Figure 4.23. The functionality available by clicking on an active learning object includes searching and categorization, as well as the ability to add tags or notes, and to modify/delete selected tags or notes.

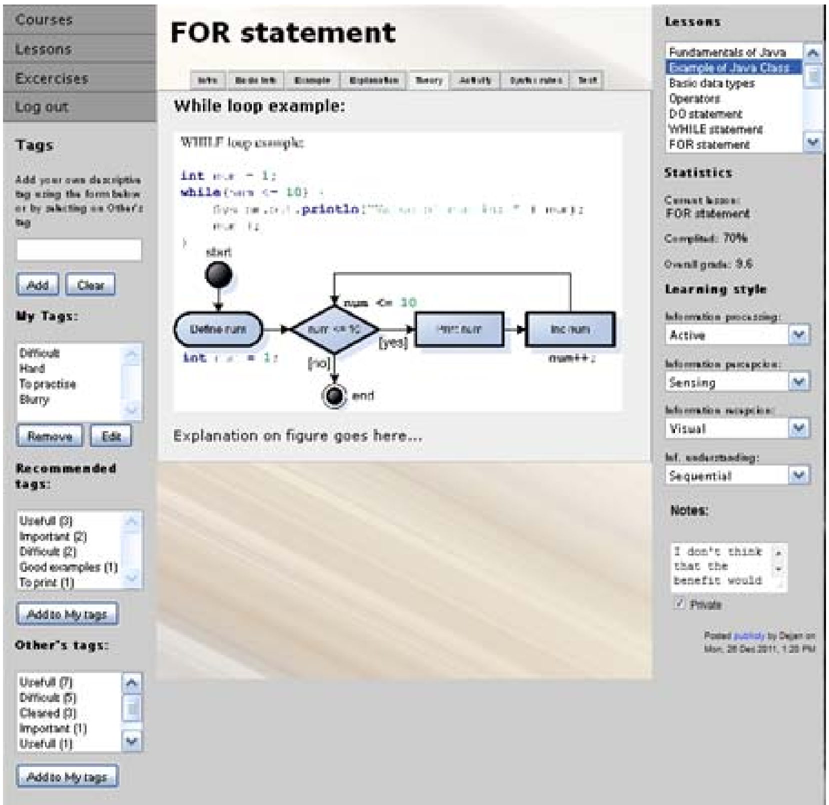

Figure 4.23 Protus's interface for tagging

The learner may add many tags by listing them in the text box and separating them by comas. Information provided by the individual learner is located under "My Tags" in the interface. By clicking on "My Tags" a list of all tags the learner has used is revealed, which are ordered from the most to least frequently used tag (Figure 4.24a). By clicking on each individual tag a list of three options pertaining to the tag are presented: "Search for *this tag*", which links the learner to the search interface; "View learning objects you've tagged with *this tag*", which shows all learning objects which have been described with the given tag; and "edit *this tag*", which give learner option to modify/delete this tag. The other tags, determined by providing the most popular tags added to learning objects of other learners, appear under "Others' Tags". Learner has ability to add any tags from the "Others' tags" to "My tags" list. An example of these functionalities is shown in Figure 4.24b. According to the research that we have made by comparative analysis of tag-based recommender algorithms, the "recommended tags" list is generated according to the learners' and experts' tags based on Ranking with Tensor Factorization model which produced more accurate recommendations then existing state-of-the-art algorithms.

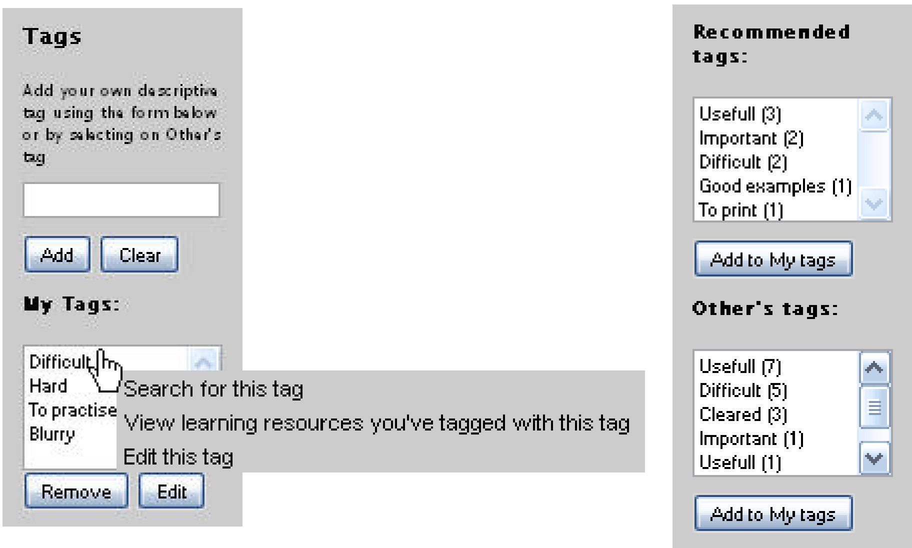

Figure 4.24 Viewing "My tags" (a), "Recommended tags" and "Others' tags" (b) list in Protus interface

Note-taking in Protus refers to adding longer pieces of free text (Figure 4.25). When compared to tags, notes have absolutely no rules, and are used for the purpose of adding normally structured sentence-based messages. To add a note the learner simply types a message in the text area. She/he has the added capability of making notes "public" (globally viewable) or "private" (viewable only by her/him); the default is public which can be changed by checking the "Private" checkbox. Also, the other notes, added to learning objects of other learners, appear under "Others' Notes". As shown in Figure 4.25, all notes contain information on when and who added the note.

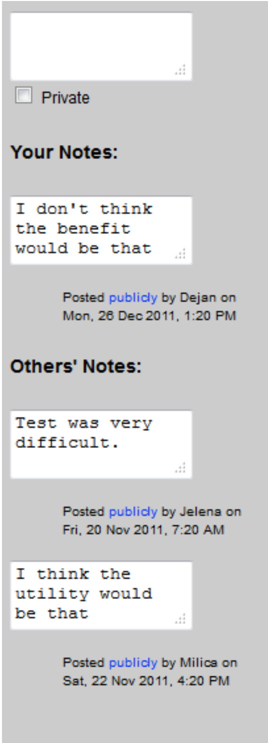

Figure 4.25 Note taking in Protus

4.3.3.1. Tag Browsing

Tag Browsing, in terms of an individual learner, is an interface to automatically categorize information based on tags. In the community sense it is a way to gain a "global view" of the tagging of the entire community, while still allowing learners to browse the individual contributions of peers. This functionality can be accessed through the tag menu, and provides three options: "My Tags", "Others' tags", and "Community Tags".

Information provided by the individual learner is located under "My Tags" in the interface. By clicking on "My Tags" a list of all the tags the learner has used is revealed, which are ordered from the most to least frequently used tag (Figure 4.26). By clicking on each individual tag a list of three options pertaining to the tag are presented: "Search for *this tag*", which links the learner to the search interface; "View learning objects you've

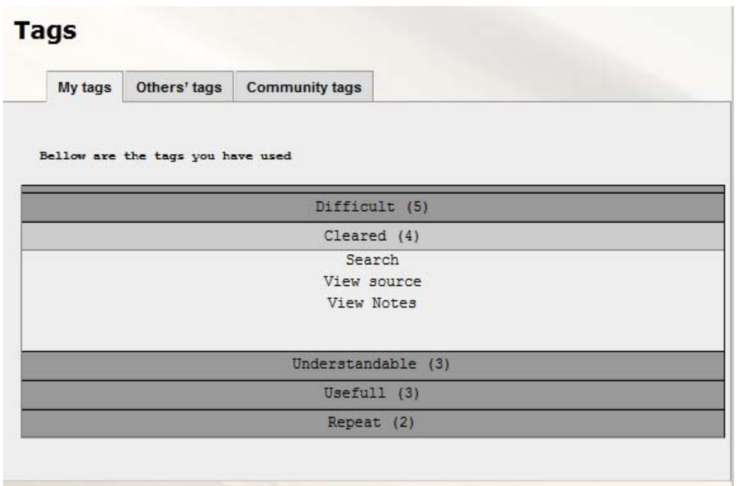

Figure 4.26 Viewing "My Tags" list in Tags interface

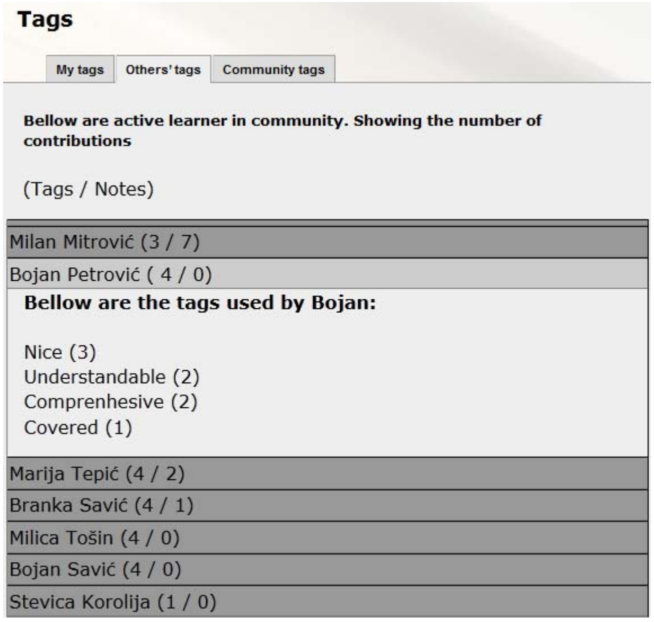

Figure 4.27 Viewing "Others' Tags" list in Tags interface

By expanding the "Others' Tags" section, the list of active learners in community and the number of contributions are shown. Selection of one learner's tag section (see Figure 4.27) displays the most popular tags added by this learner in descending order of number of times used.

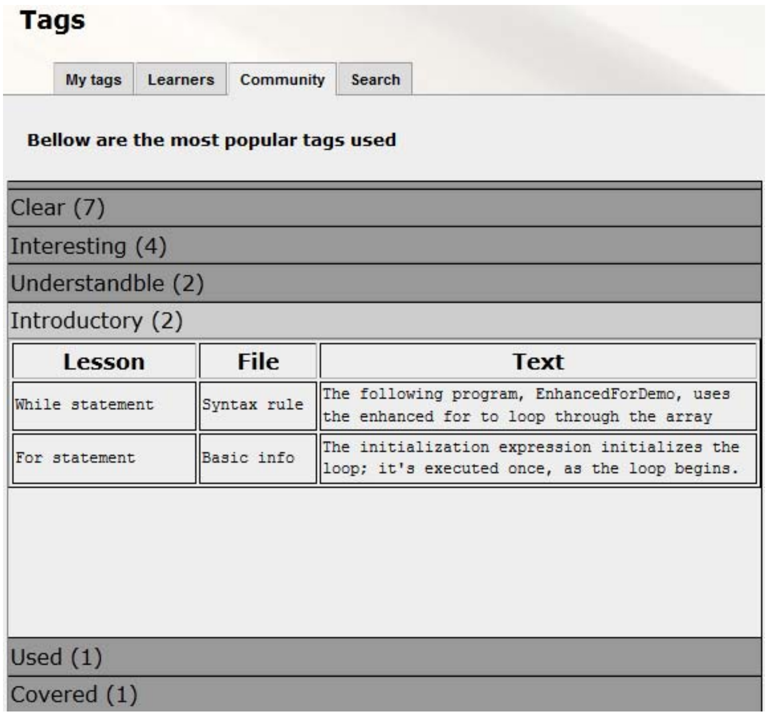

Figure 4.28 Viewing "Community Tags" list in Tags interface

By expanding the "Community Tags" section (see Figure 4.28), the most popular tags are shown in descending order of number of times used according to the overall use of tags, independent of individual learner who specifies tags. This gives the learner an idea, at a much higher level, the overall view of all the content. The learner can get a sense of what are the most important terms and/or ideas at a course level. By clicking on one of the tags the learner can select to add the tag to the learning object in context, or to search by the given tag. If a learner chooses to search for any selected tag lesson and learning object for which the tag is placed will be displayed. For each tag, the number of occurrences is printed in brackets.

4.3.3.2. Tag Searching

Searching for pages with certain tags is an important aspect of the system. Along with tag browsing, the searching functionality would potentially provide the learner with more motivation to contribute to use the system.

Figure 4.29 shows searching for a learning object by tags. The results are returned ordered by “score”. The score simply counts all occurrences of the tag in any learner’s learning objects on the given page. The learning object with the highest count of tag occurrences is presented first.

The system generated query results for the search box using the *Ranking with Vector Spaces* algorithm provided by MySQL9 (MySQL internals algorithms, 2008). This algorithm uses a variation on the well-known TF-IDF term-weighting method. In this method the database system assigns each term in the collection a weight, according to its frequency in the collection and its frequency in each learning objects. When a participant performs a query, the algorithm selects and ranks learning objects according to the matched query terms and their individual weights. That is, any learning object with the relevant keyword was included in the search results.

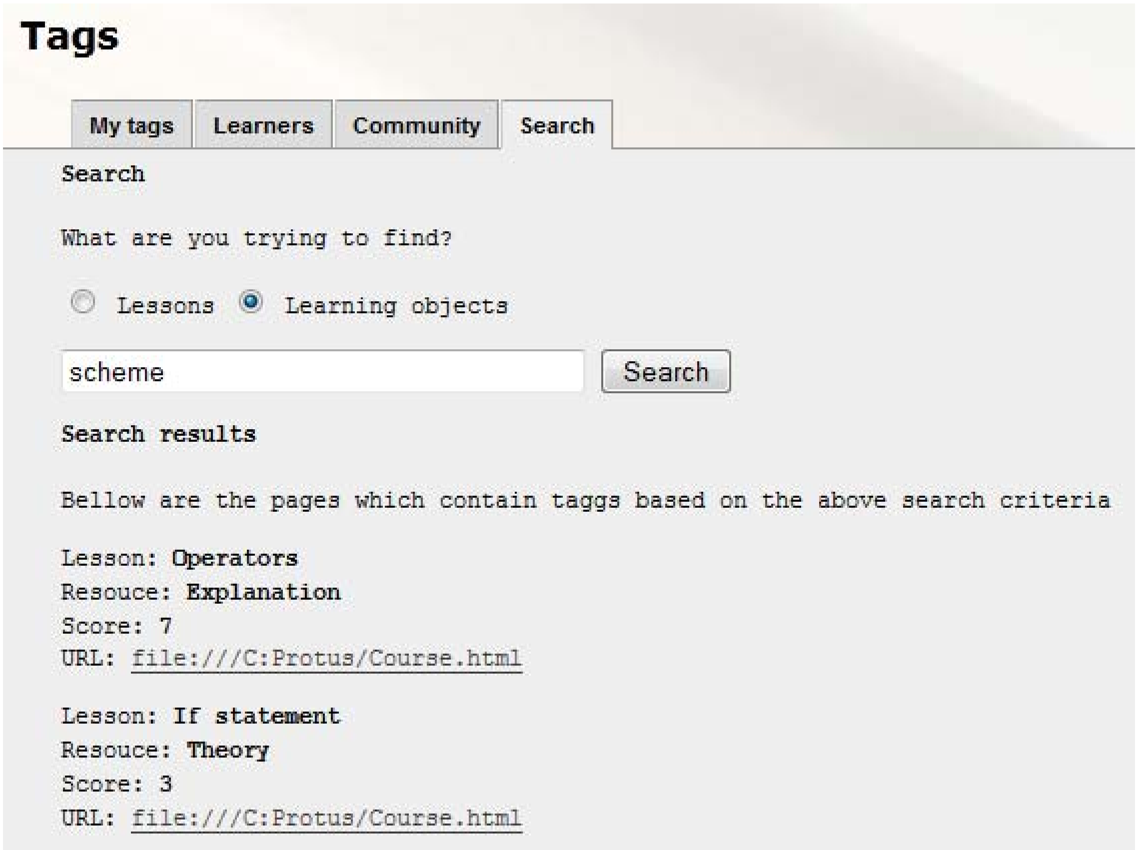

Figure 4.29 Searching for lessons by tags

Part III

Evaluation and Discussion

Chapter 5

Experimental Research

The experiments were realized on an educational dataset, consists of 440 learners, 3rd year undergraduate students of the Department of Information technology at Higher School of Professional Business Studies, University of Novi Sad. The experiment lasted for two semesters, from September 2010 until May 2011. Involved learners were programming beginners that successfully passed the basic computer literacy course at previous semester. They were divided into two groups: the experimental group and the control group. Learners of the control group learned with the previous version of the system (Vesin et al., 2009) and did not receive any recommendation or guidance through the course, while the learners of the experimental group were required to use the Protus system. Learners from both groups did not take any parallel traditional course and they were required not to use any additional material or help.

5.1. Data Definition

The experimental group consisted of 340 learners, while the control group consisted of 100 learners.

In order to assess whether the means of two groups are statistically different from each other, the t-test was utilized. Both groups of learners completed the Norm-referenced test which allows us to compare learners' intellectual abilities (Glaser, 1963). Results of this test were combined with grades that learners earned at a basic computer literacy course at the first semester of their studies. The aim of a computer literacy course was to teach data structures and algorithms by presenting exercises of algorithm simulations to the learners.

Programming coursework in any programming language was not assessed. The most important outcome was therefore the introduction of general problem solving concepts, rather than focusing on teaching the syntax of a specific programming language. The predetermined alpha level adopted for hypothesis testing was 0.05, as significance levels of less than 0.05 are considered statically significant, degrees of freedom (df) for the test was 438. Table 5.1 reports the obtained t-test results. Since the calculated value of t (1.23) is not greater than table value of t (1.96), we can conclude that the differences between the experimental and the control group are negligible and there is no need for additional equalization of groups.

Table 5.1 The analysis of the test score difference

Type of test	Group	N	Df	Mean	t-calculated value	t-table value
Intellectual abilities	Experimental	340	438	117,25	1.23	1.96
	Control	100		111,69		

Level of significance α=0.05

5.2. Data Clustering

The learners from the experimental group filled out the Felder-Soloman Index of Learning Styles Questionnaire (Felder and Soloman, 1996). This psychological questionnaire maps a set of 44 questions over 4 dimensions (Figure 5.1) representing learning preferences and styles: Active versus Reflective, Sensing versus Intuitive, Visual versus Verbal, Sequential versus Global. The aim was to cluster learners from the experimental group into a sub-class according to the learning style revealed by the ILS questionnaire. Figure 5.2 shows the comparison of learners' stated preferences corresponding to learning styles across all four domains.

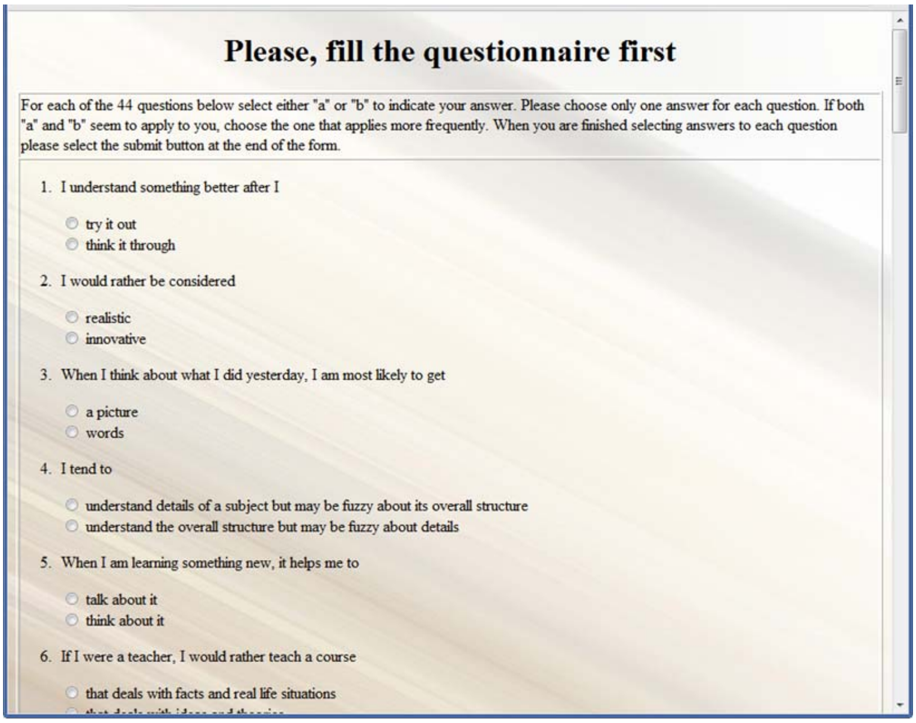

Figure 5.1 ILS Questionnaire

Based on the results of the questionnaires it was possible to define appropriate clusters, which determined learner profiles for 340 learners from the experimental group. Clusters were formed for different combinations of learning styles within the three categories (Table 5.2). Category *Information processing* was omitted in order to increase the number of learners in a separate cluster and to obtain more relevant data for recommendations. In future research, when increasing the number of learners participating in the experiment it can be taken into account. The summary of usage data per each cluster is shown on Table 5.3. Number of learners, number of LOs, number of tags, average number of tags per learners and average number of tags per LO were measured. In order to understand the characteristics of learner tags, and learner tagging behavior, in the next section, we will examine tag characteristics of learners in the Protus system.

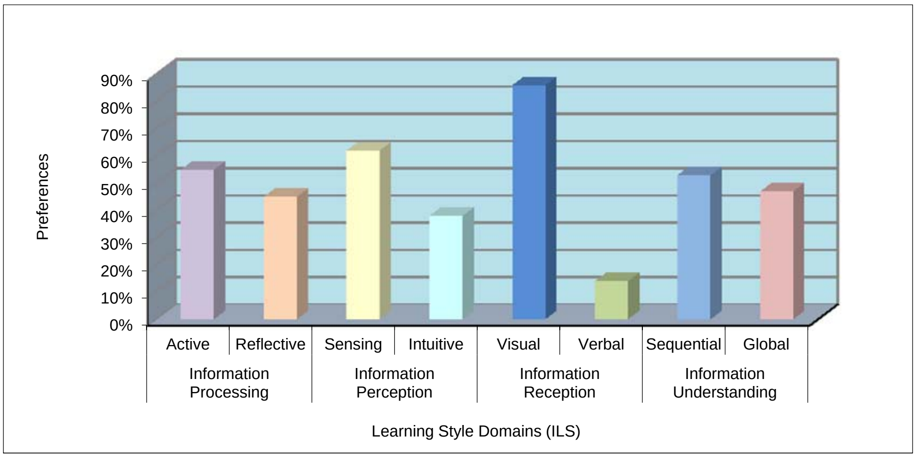

Figure 5.2 Learning styles results

Table 5.2 Cluster identification based on different styles

Cluster1	Cluster2	Cluster 3	Cluster 4	.....	Cluster 8
Sensing	Sensing	Sensing	Sensing		Intuitive
Visual	Visual	Verbal	Verbal		Visual
Sequential	Global	Sequential	Global		Global
49 learners	**46 learners**	**39 learners**	**42 learners**	**.....**	**48 learners**

Table 5.3 Characteristics of the data sets per cluster

	Clust1	Clust2	Clust3	Clust4	Clust5	Clust6	Clust7	Clust8
Num. of Learners	49	46	39	42	35	42	39	48
Num. of LO	72	72	72	72	72	72	72	72
Num. of Tags	2402	2707	3283	2380	2243	2486	2289	2268
Avg. Num. of Tags per Learners	54,6	57,3	67	49,6	64,1	59,2	58,7	63
Avg. Num. of Tags per LO	33,4	37,6	45,6	33,6	31,6	34,5	31,8	31,5

5.3. Statistical Properties of Learners' Tagging History

This section investigates how Protus learners utilize tags in order to organize their collections of learning objects. It further discusses global, as well as item and user-level, patterns that emerge from this collaborative tagging activity.

When we analyzed the dataset in terms of learners' activity and tags' usage, all clusters were considered together.

5.3.1. Learners' Activities

We studied how many LO were tagged on average by each learner in the system (Figure 5.3) and found that 12% of the learners tagged less than 10 LOs (low activity), 23% tagged between 10 and 50 LOs (medium activity) and 65% tagged between 50 and 72 LOs (high activity). We also analyzed the tagging vocabulary, i.e., how many different tags each learner used to define her/his preferred LOs (Figure 5.4). We found that 21% of the learners used less than 20 different tags, 71% used between 20 and 35 tags and the remaining 8% used between 35 and 65 tags.

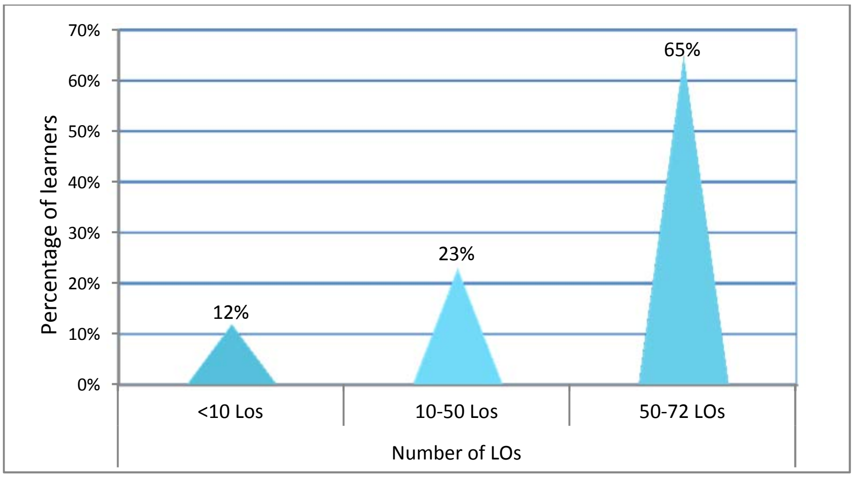

Figure 5.3 Learner activities on LOs

5.3.2. Tag Usage

In order to understand the characteristics of learner tags, and learner tagging behavior, we examined tag characteristics of learners in the Protus system. Figure 5.5 illustrates the frequency of use of the first fifteen popular tags in the learner tag set. The X-axis denotes the number of learners entered over time. The Y-axis denotes the proportion of tag usage in a specific time. Here, each line represents a tag.

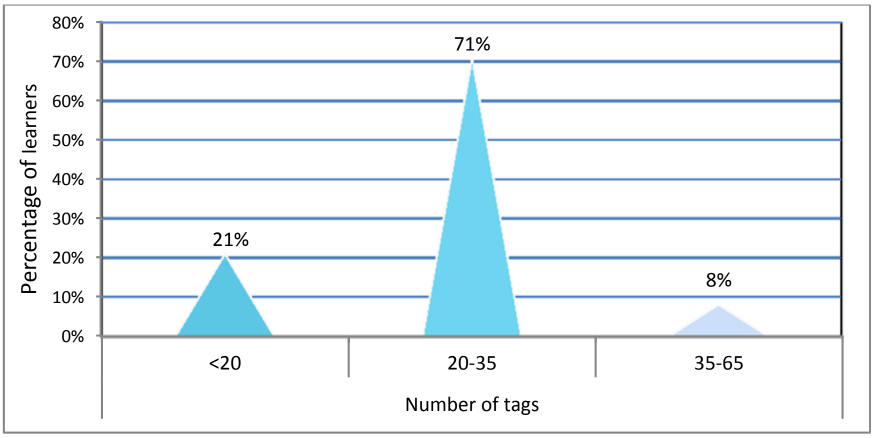

Figure 5.4 Learners activities on tags

Overall, these tag patterns reveal that after examining about 20 learners, the frequency of learner tags tends to remain stable at fixed proportions of the total tag frequency. In other words, as the number of learners increases and a tagged object receives more and more tags, the frequency of at which a tag is selected tends to become fixed (Golder and Huberman, 2006).

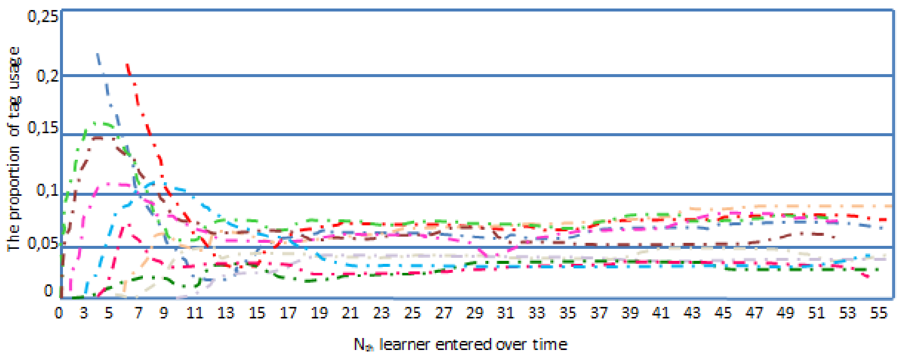

Figure 5.5 The stabilization of learner tags' relative proportions over time

In their work, Golder and Huberman (2006) proposed the concept of convergence or stabilization, and indicated that stability has important implications for the collective usefulness of individual tagging behavior. Likewise, this stabilization might appear during instances of shared knowledge, as well as when users imitate the tag selection of other users. Thus, the tag extraction process of our research design helps demonstrate how our tag-based system might facilitate knowledge sharing among learners. Furthermore, as group ideas or opinions on a reading change, this should be reflected by a corresponding change in the previously stable tag frequencies.

5.3.3. Tag Entropy over Time

Research on collaborative tagging process itself has found that the position of a tag correlates with its expressiveness. Tag entropy is a measurement of specificity where more general tags should have higher entropies because they might appear in different topics, whereas seldom tags are often more specific to a topic, thus have lower entropies. The entropy of a tag is defined as:

$$H(T) = -\sum_{t \in T} p(t) \log_2 p(t)$$

Here, T is the set of tags in the profile of learner, p(t) is the probability that the tag t was utilized by learner and $\log_2 p(t)$ is called selfinformation. Using base 2 for the computation of the logarithm allows for measuring self-information as well as entropy in bits.

Figure 5.6 shows the strong correlation between the position and the informativeness of a tag.

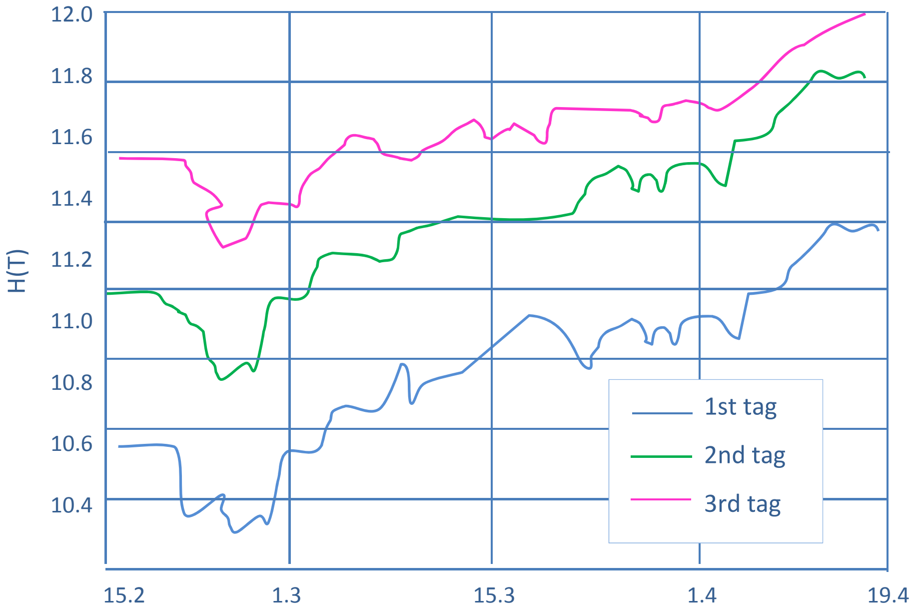

Figure 5.6 Tag entropy H(T) over time

It appears that learners tend to assign common tags at the beginning of the tagging process and more specific tags later. There exist at least three potential explanations for this effect (Wetzker et al., 2010):

1. The affinity to label from general to specific could be a universal behavioral pattern of humans that persists in other domains.
2. The effect could also consequence from users' intention to classify new content into a set of relatively constant categories. Adding frequent category tags in the beginning and content specific tags later would result in an increase in entropy as the one observed in Figure 5.6.
3. Finally, the perceived association between tag position and entropy could be initiated by the tag recommendation functionality.

5.3.4. Semantic Analysis of Tags

A semantic analysis of tags was performed to better understand different utilization of tags. Tags were classified according to Sen et al. (2006) that is also based on the categories of Golder and Huberman (2005), which are:

1. **Factual tags** - tags may be used to identify the topic of a object using nouns and proper nouns (e.g. operators, loop, arrays) or to classify the type of object (e.g. tutorial, task, example, basic info, explanation, definition),

2. **Subjective tags** - tags may be used to denote the qualities and characteristics of the item (e.g. useful, interesting, difficult, easy, understandable, blurry) and
3. **Personal tags** - item ownership, self-reference, tasks organization - a subset of tags often used by individuals to organize their own learning objects. Much like self-referencing tags, some tags are used by individuals for task organization (e.g. to read, to practise, to print).

When we analyzed how these tags were used and re-used among learners, we found the vast majority of the tags (Table 5.4) were of the personal (44% of tags) and subjective type (40% of tags). The rest of the tags (16% of tags) were factual in their nature and could be used to identify the topic of a learning object. The obtained distribution indicates the fact that learners adapt learning objects themselves and organize them for easily managing. In a MovieLens study (Sen et al., 2006), for comparison, the distribution was 63% factual, 29% subjective, 3% personal and 5% other.

Table 5.4 Types analysis of each tag

PERSONAL	SUBJECTIVE	FACTUAL
44%	40%	16%

5.4. Experimental Protocol and Evaluation Metrics

The performance of the proposed models is evaluated by holding out a part of the data set as ground-truth data (the test set), and building prediction models from the remaining data (the training set).

We randomly divided the data set into training set and a test set with sizes 80 and 20 percent of the original set, respectively. As performance measures for item and tag recommendations, we use the classic metrics of precision and recall which are standard in such scenarios (Herlocker et al., 2004). Precision and recall have been in use to evaluate information retrieval systems for many years. Mapping into recommender system manner of speaking, precision and recall have the following definitions regarding the evaluation of top-N recommendations.

For a test user that receives a list of N recommended tags (top-N list), precision and recall are defined as follows:

- **Precision** is the ratio of the number of relevant tags in the top-N list (i.e., those in the top-N list that belong in the future set of tags posted by the test user) to N.
- **Recall** is the ratio of the number of relevant tags in the top-N list to the total number of relevant tags (all tags in the future set posted by the test user).

With *i* being the item from the randomly picked post of user *u* and $\hat{T}(u,i)$ the set of recommended tags, recall and precision can be calculated as:

$$recall(\hat{T}(u,i)) = \frac{1}{|U|}\sum_{u \in U}\frac{|tags(u,i) \cap \hat{T}(u,i)|}{|tags(u,i)|}$$

$$precision(\hat{T}(u,i)) = \frac{1}{|U|}\sum_{u \in U}\frac{|tags(u,i) \cap \hat{T}(u,i)|}{|\hat{T}(u,i)|}$$

All experiments are repeated 10 times and we report the mean of the runs. For each run, we use exactly the same train/test splits.

5.5. Experimental Results

The classical measures precision and recall were chosen to evaluate the performance of several suitable recommendation techniques for RS in e-learning environments.

First, we describe the specific settings used to run evaluated algorithms. Then we present and discuss the results of our evaluation.

5.5.1. Settings of the Algorithms

Before starting full experimental evaluation of selected algorithms we determined the sensitivity of appropriate parameters to different algorithms and from the sensitivity plots we fixed the optimum values of these parameters and used them for the rest of the experiments. The analysis was performed for all eight clusters. Given the similarity of the obtained values of parameters the results for the first cluster are shown only.

Most Popular Tags. We counted in how many posts tag occurs globally and used the top tags as recommendations.

Most Popular Tags by Item. For a given item we counted for all tags in how many posts they occur together with that item. We then used the tags that occured most often together with that item as recommendation.

Most Popular Tags ρ - Mix. Before comparing *Most Popular Tags ρ - Mix algorithm with* the others, we focused on finding an appropriate size of parameter ρ . Hence, we observed a similar precision/recall behavior for all values of $\rho \in \{0, 0.1, ..., 0.9, 1\}$. As can be seen in Figure 5.7, variation of algorithm with the most popular tags by user $(\rho = 0)$ performs worse than variation of algorithm with the most popular tags by item $(\rho = 1)$ for all numbers of recommended tags. All mixed versions perform better than most popular tags by user and all mixed versions with $\rho \geq 0.5$ perform better than most popular tags by item. The best performance is obtained if $\rho = 0.6$.

Adapted PageRank. With the parameter d = 0.7 we stopped computation after 10 iterations or when the distance between two consecutive weight vectors was less than 10^{-6}. In $\vec{p}$, we gave higher weights to the user and the item from the post which was chosen. While each user, tag and item got a preference weight of 1, the user and item from that particular post got a preference weight of 1 + |U| and 1 + |I|, resp.

FolkRank. The same parameter and preference weights were used as in the Adapted PageRank.

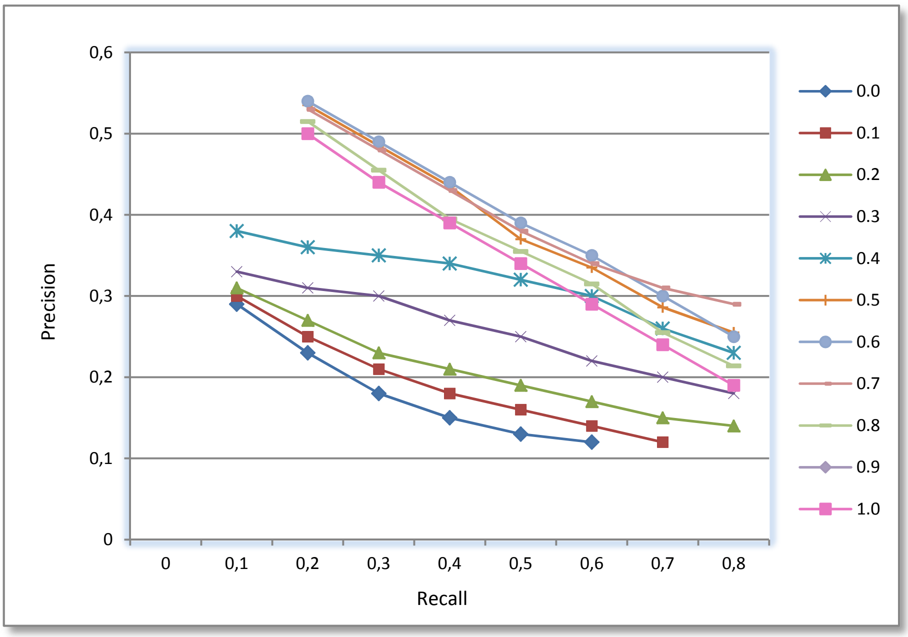

Figure 5.7 Precision and recall of most popular tags ρ mix for $\rho \in \{0, 0.1, \ldots, 0.9, 1\}$

Collaborative Filtering based on Tags. For Collaborative Filtering algorithm the neighborhood is computed based on the user-tag matrix $\pi_{UT}Y$. The only parameter to be tuned in the CF based algorithms is the number *k* of best neighbors (Sarwar et al., 2001). We examine the effect of the variation of recalls according to the neighborhood size k which is closely connected with tag preference generation. Figure 5.8 shows a graph of how recall changes as the neighbor size grows from 10 to 90. Recommender quality initially improves as we increase the neighborhood size from 10 to 30. However, after the neighborhood of size 30, increasing the value of k did not lead to statistically significant improvements. That is, once the number of nearest neighbors, k, is sufficiently large, the recommendation quality for each user is not changed by any further increases in the number of nearest neighbors. Considering this trend we selected 30 as our optimal choice of the neighborhood size.

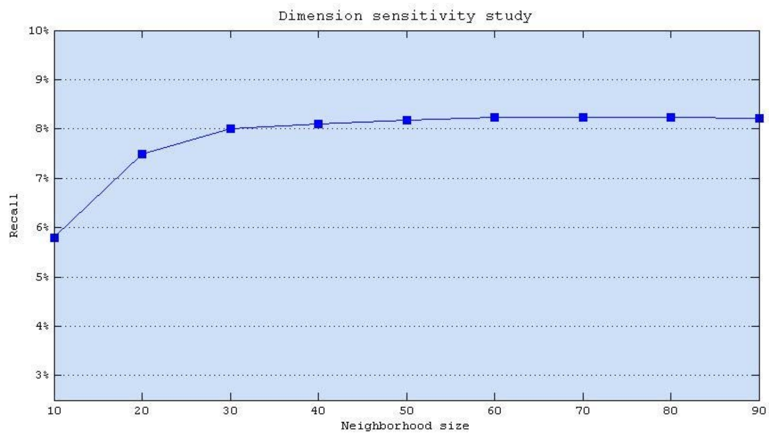

Figure 5.8 Recall of collaborative tag-based CF according to the variation of neighborhood size

HOSVD - based model. Since there is no straightforward way to find the optimal values for c_1, c_2 and c_3, we follow the way according to (Symeonidis et al., 2010) that a 70% of the original diagonal of $X_{(1)}$, $X_{(2)}$ and $X_{(3)}$ matrices can give good approximations. Thus, c_1, c_2 and c_3 are set to be the numbers of singular values by preserving 70% of the original diagonal of $X_{(1)}$, $X_{(2)}$ and $X_{(3)}$ respectively in each run.

RTF. We ran RTF with $(k_u, k_i, k_t) \in \{(8,8,8); (16,16,16); (32,32,32)\}$ dimensions, as in (Rendle et al., 2009). The corresponding model is called "RTF 8", "RTF 16", and "RTF32". The other hyper parameters are: learning rate $\alpha = 0.5$, regularization $\gamma = \gamma_c = 10^{-5}$, iterations $iter = 500$. The model parameters $\hat{\theta}$ are initialized with small random values drawn from the normal distribution $N(0,\ 0.1)$.

5.5.2. Results

In the following subsection, we will present and discuss the results of selected methods evaluation. First, we compare simple methods based on counting tag occurrences (Most Popular Tags), specific approaches for improving the performance of such methods and an adaptation of User-based Collaborative Filtering, named Collaborative Filtering based on Tags. Then, we analyze the prediction quality of graph-based approaches, Adapted PageRank and FolkRank, and tensor based approaches, HOSVD and RTF. Finally, we give a comparative analysis of the best representatives of these considered techniques, together.

5.5.2.1. Comparison of Methods Based on Counting Tag Occurrences and Standard Collaborative Filtering Based on Tags

In this section we compare simple methods based on counting tag occurrences: Most Popular Tags, Most Popular Tags by Item, Most Popular Tags by User, Most Popular Tags 0.6–mix and standard Collaborative Filtering based on Tags.

There are two types of diagrams, which are used for all eight clusters of the Protus system. The first type of diagram (Figure 5.9) shows in a straightforward manner how the recall depends on the number of recommended tags. The other diagrams are usual precision/recall plots. Here a data point on a curve stands for the number of tags recommended, starting with the highest ranked tag on the left of the curve and ending with ten tags on the right. Therefore, the steady decrease of all curves in those plots means that the more tags of the recommendation are regarded, the better the recall and the worse the precision will be. Figure 5.9 shows how the recall increases, when more tags of the recommendation are used. All algorithms perform significantly better than the baseline Most Popular Tags and the Most Popular Tags by User strategy, whereas it is much harder to beat the Most Popular Tags by Item. The idea to suggest the most popular tags by item results in a recall which is very similar to using the CF recommender based on user's item similarities. In contrast to these two approaches, the Most Popular Tags ρ Mix-recommender includes also the user's tags in the recommendations. As the diagrams show, it is successful and could gain results better than those of CF. The precision-recall plots in Figure 5.10 extend diagrams from Figure 5.9 with the precision measure. All algorithms perform significantly better than the baseline Most Popular Tags and the Most Popular Tags by User strategy. It is remarkable that the Most Popular Tags 0.6–Mix recommender provides on average better precision and recall than both Collaborative Filtering algorithms and Most Popular Tags by Item.

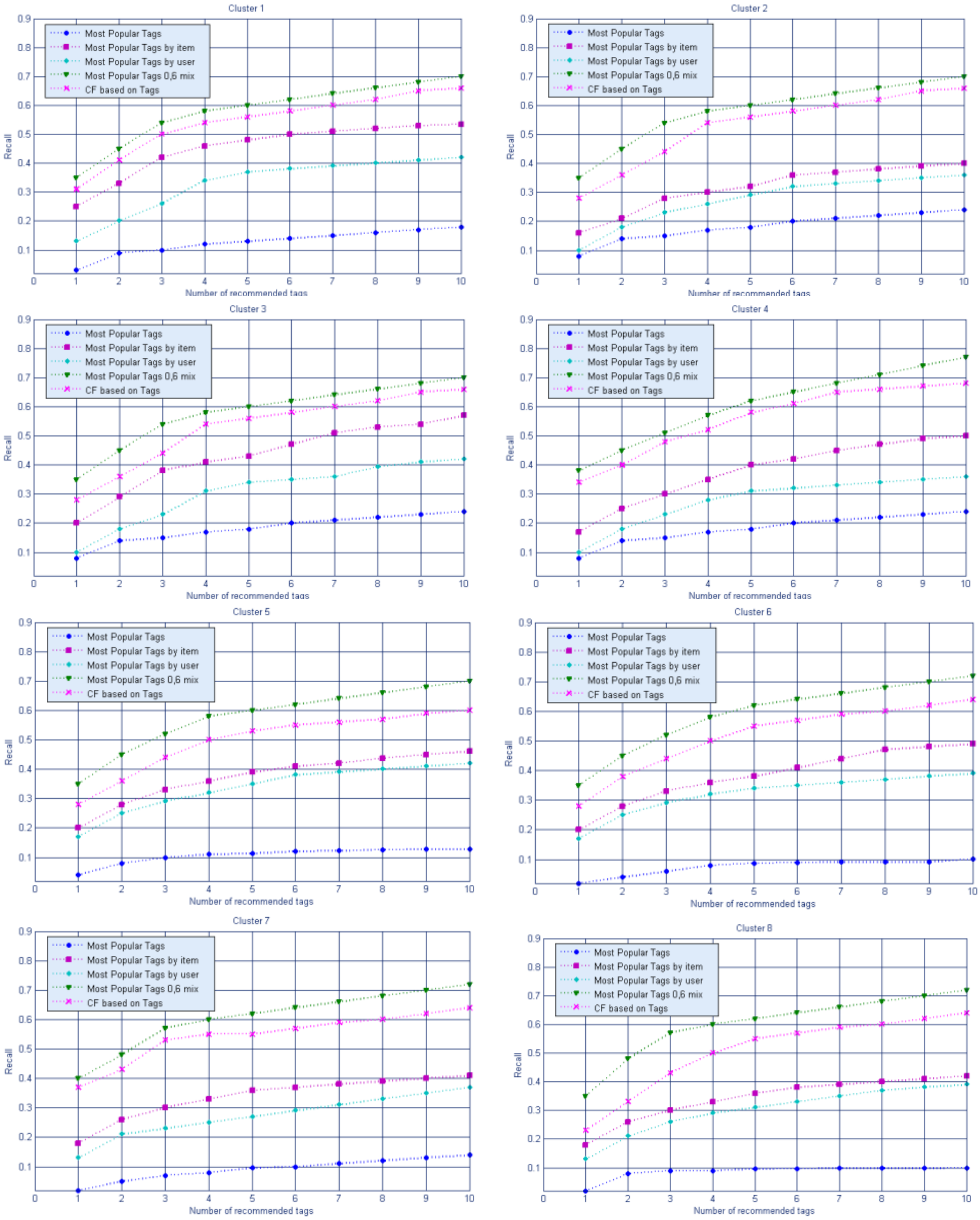

Figure 5.9 Recall for methods based on counting tag occurrences as a function of number of recommended tags for the eight clusters of the Protus system

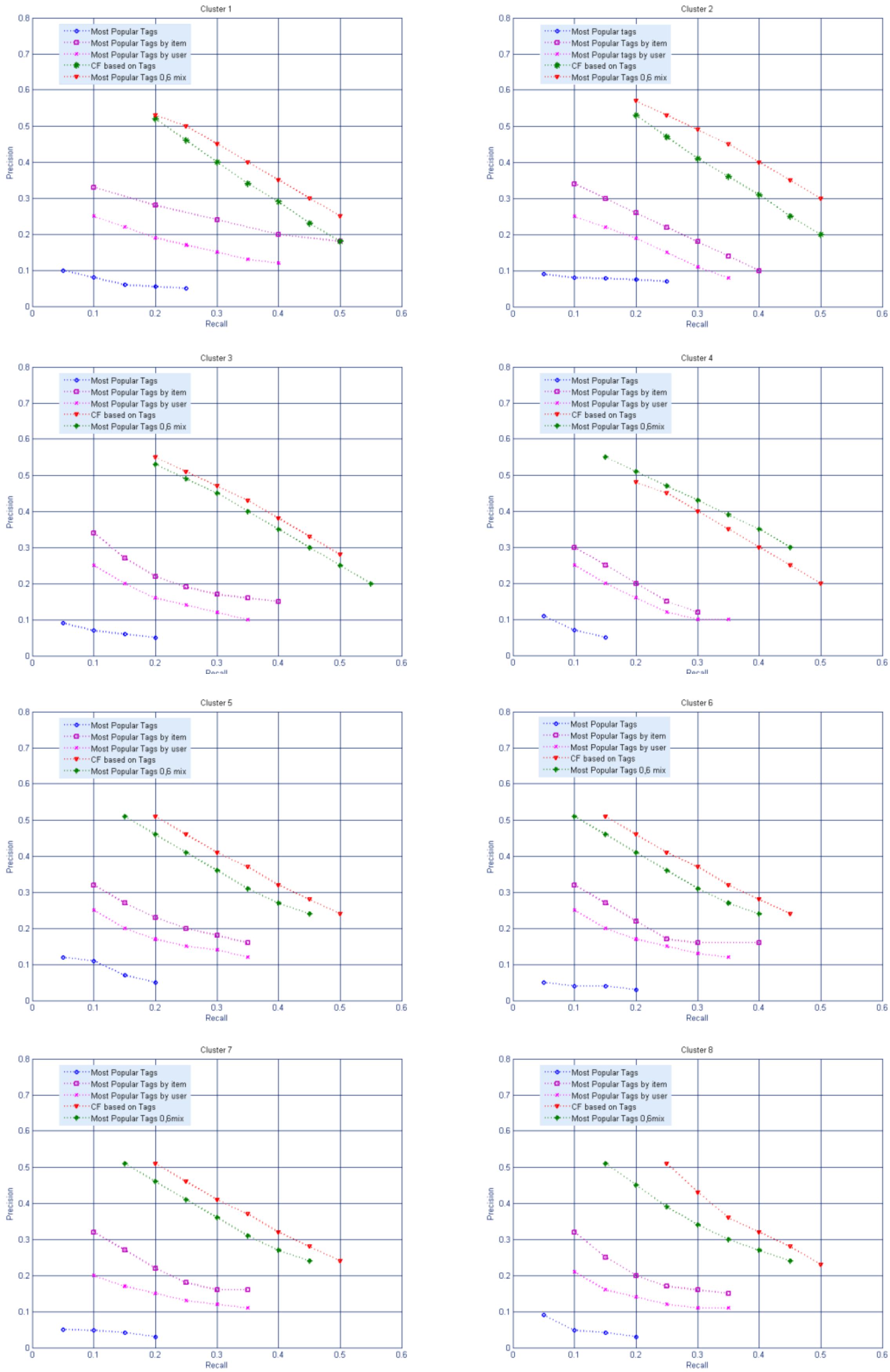

Figure 5.10 Recall and Precision for methods based on counting tag occurrences for eight clusters of the Protus system

5.5.2.2. Comparison of Graph-based Approaches

We saw in Section 3.2 that in order to apply standard CF-based algorithms to folksonomies, some data transformation must be performed. Such transformations lead to information loss, which can lower the recommendation quality. Another well-known problem with CF-based methods is that evaluation large projection matrices must be kept in memory, which can be time/space consuming and thus compromise real time recommendations. Also, for each different mode to be recommended, the algorithm must be eventually changed, demanding an additional effort for offering multi-mode recommendations.

FolkRank builds on PageRank and proved to give significantly better tag recommendations than CF, because FolkRank has ability to exploit the information that is appropriate to the specific user together with input from other users via the integrating structure of the underlying hypergraph. When comparing the prediction quality of CF, Adapted PageRank and FolkRank (Figure 5.11) one can see that FolkRank outperform both two. FolkRank is able to predict, additionally to globally relevant tags, the exact tags of the user which CF could not. This is due to the fact that FolkRank considers, via the hypergraph structure, also the vocabulary of the user himself, which CF by definition doesn’t do. This method also allows for mode switching with no change in the algorithm. Moreover, as well as CF-based algorithms, FolkRank is robust against online updates since it does not need to be trained every time a new user, item or tag enters the system. However, FolkRank is computationally expensive and not trivially scalable, making it more suitable for systems where real-time recommendations are not a requirement.

5.5.2.3. Comparison of Methods Based on Tensor Factorization

Similarly to FolkRank, tensor factorization methods also work directly over the ternary relation of the folksonomy. Although the tensor reconstruction phase can be expensive, it can be performed offline. After the lower dimensional tensor is computed, the recommendations can be done fast, making these algorithms suitable for real-time recommendations. A potential disadvantage of tensor factorization methods is that easy mode switching can only be achieved if one considers that the different recommendation problems, i.e., user/item/tag, can be addressed by minimizing the same error function. If one chooses HOSVD for example, the reconstructed tensor can be used for multi-mode recommendations with trivial mode switching, but at the cost of eventually solving the wrong problem: HOSVD minimizes a least-square error function while social tagging RS are more related to ranking. If one tries to optimally reconstruct the tensor with regard to an error function targeted to a specific recommendation mode on the other hand, accuracy is eventually improved, but at the cost of making mode switching more involved. Even though RTF and HOSVD have the same prediction method and thus prediction complexity, in practice RTF models are much faster in prediction than comparable HOSVD models, because RTF models need much less dimensions than HOSVD for attaining better quality. Also, for the task of personalized ranking HOSVD has three major drawbacks to RTF (Rendle et al., 2009):

1. HOSVD doesn’t take into account missing values. For tag recommendation the missing values are usually filled with zeros (Symeonidis et al., 2008).

2. HOSVD optimizes for minimal element-wise error. But for the ranking problem of tag recommendation we are interested in another objective function.

3. HOSVD has no regularization. For machine learning tasks preventing overfitting is very important so HOSVD is predisposed to overfitting.

A final problem with HOSVD is sensitivity to the number of dimensions and that they have to be chosen carefully. Also HOSVD is sensitive to the relations between the user, item and tag dimensions (e.g. choosing the same dimension for all three dimensions leads to poor results). In contrast to this, for RTF it can be chosen the same number of dimensions for user, item and tag. Besides this theoretical analysis, in Figure 5.11 it can be seen that the prediction quality of RTF is clearly better to the one of HOSVD. Also, Figure 5.11 shows that even with a very small number of 8 dimensions, RTF achieves almost similar results as HOSVD. Increasing the dimensions of RTF to 16 dimensions it already outperforms HOSVD in quality. Furthermore for RTF, by increasing the number of dimensions we get better results. When comparing the prediction quality of RTF and FolkRank (Figure 5.11) one can see that RTF with 8/16 dimensions achieves comparable results whereas 32 dimensions outperform FolkRank in quality.

5.5.2.4. Summary of Algorithms' Advantages and Disadvantages

According to the conducted experiments on real-life dataset, in this section, we briefly discuss the main advantages and disadvantages of the aforementioned algorithms. Standard CF-based algorithms need some data transformation in order to apply to folksonomies. Such transformations lead to information loss, which can lower the recommendation quality. Another problem with CF-based methods is that large projection matrices must be kept in memory, which can be time and space overwhelming and thus compromise real-time recommendations. Also for each different mode to be recommended, the algorithm must be eventually improved, demanding an additional effort for offering multi-mode recommendations. As well as CF-based algorithms, FolkRank is robust against online updates since it does not need to be trained every time a new user, item or tag enters the system. However, FolkRank is computationally expensive and not trivially scalable, making it more suitable for systems where real-time recommendations are not a requirement. FolkRank also allows mode switching with no change in the algorithm. Similarly to FolkRank, tensor factorization methods work directly over the ternary relation of the folksonomy. Although the tensor reconstruction phase can be expensive it can be performed offline. After the lower dimensional tensor is computed, the recommendations can be done quickly, making these algorithms appropriate for real-time recommendations. A possible drawback of tensor factorization methods is that easy mode switching can only be achieved if one considers that the different recommendation problems, i.e., user/item/tag, can be addressed by minimizing the same error function. If one chooses HOSVD for example, the reconstructed tensor can be used for multi-mode recommendations with simple mode switching, but at the cost of solving the wrong problem: HOSVD minimizes a least-square error function while social tagging RS are more related to ranking. Figure 5.12 shows a comparison between some of the aforementioned algorithms. We selected only the best representatives of the considered techniques. We can conclude that the best method is RTF followed by FolkRank and HOSVD.

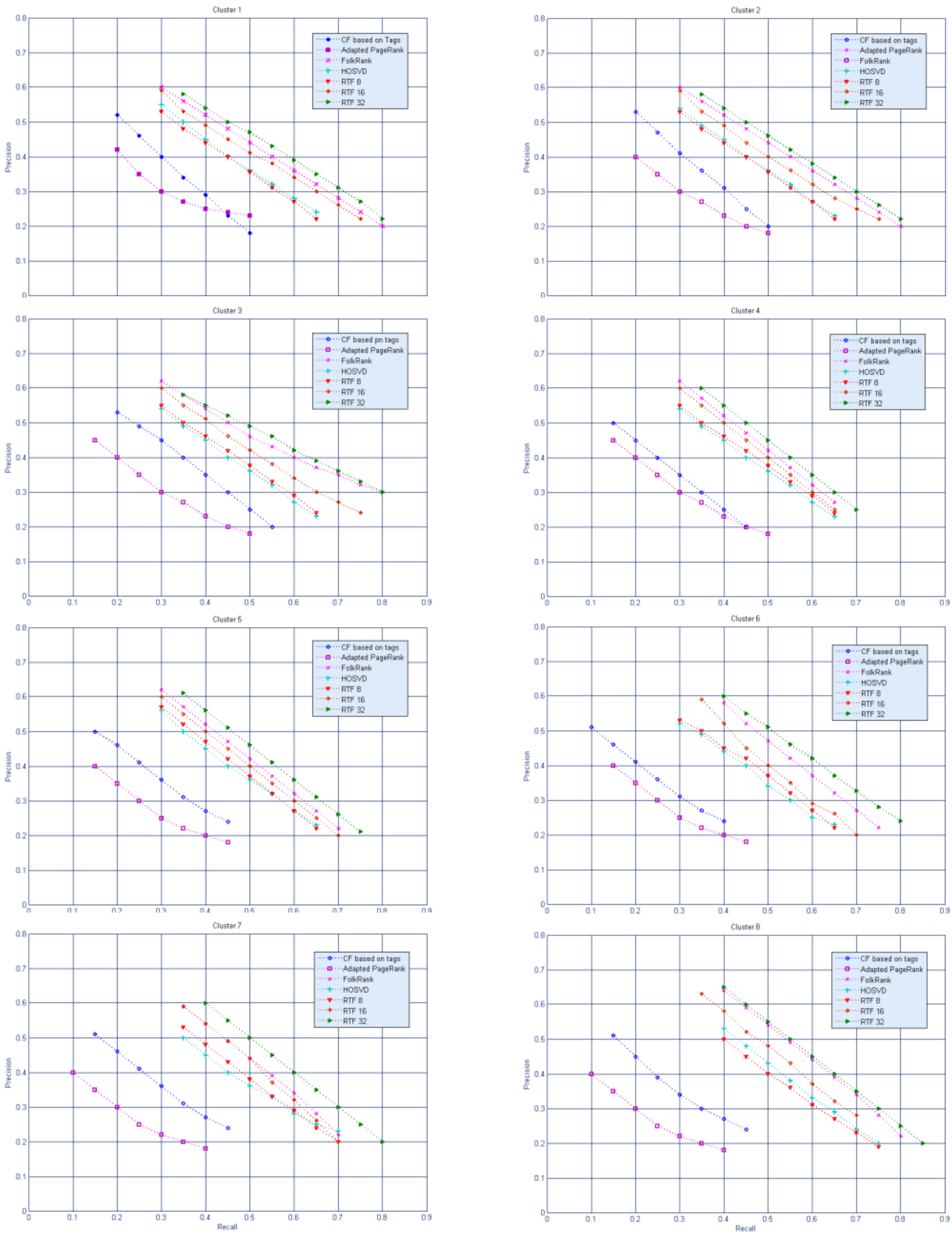

Figure 5.11 Recall and Precision for graph-based and tensor-based methods for eight clusters of the Protus system

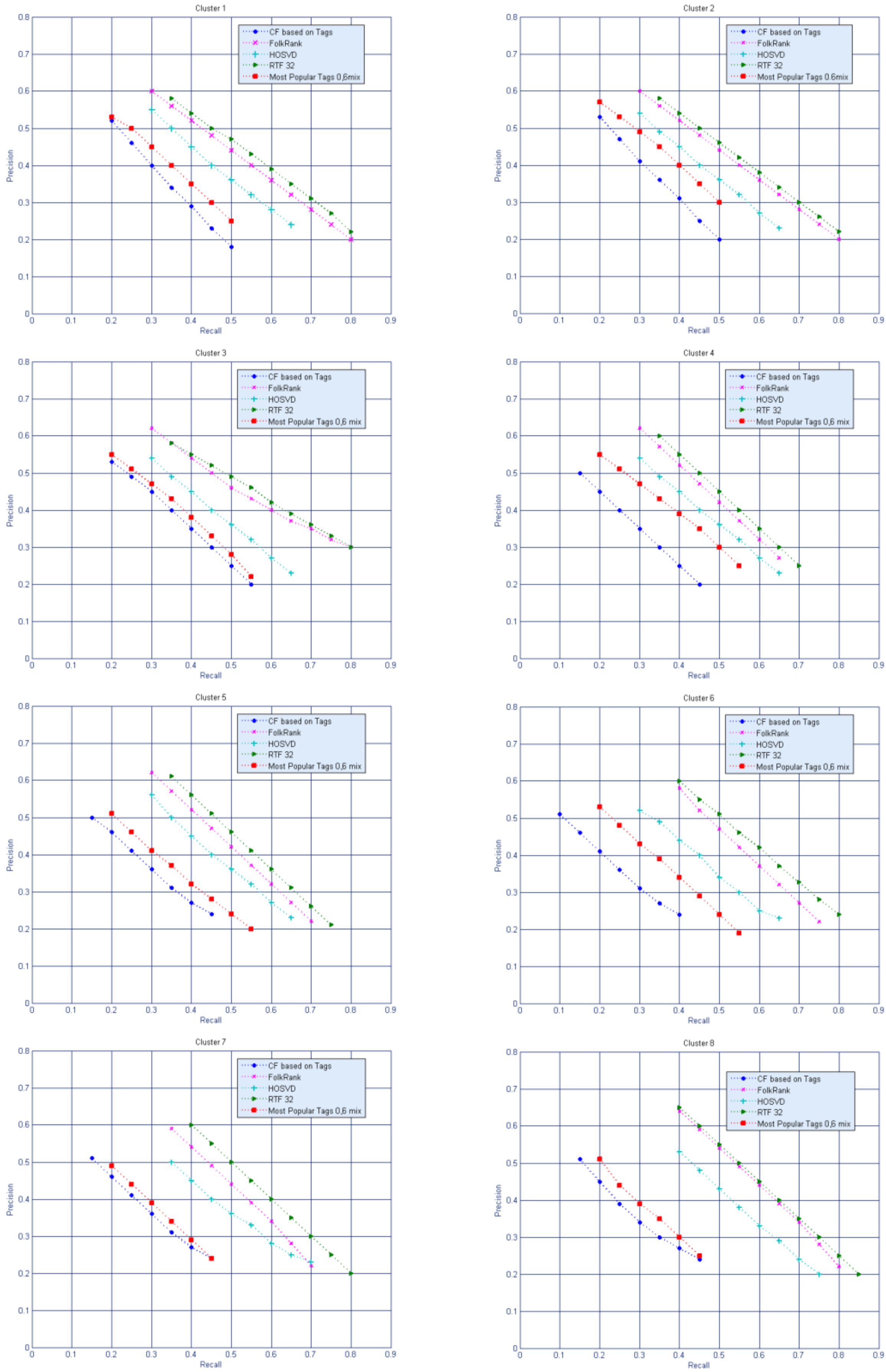

Figure 5.12 Recall and Precision for the best representatives of the considered techniques

5.6. Expert Validity Survey

In order to systematically verify the relationship between learning comprehension and learner data tagging, and in order to help teachers' evaluate learner knowledge, an experimental test was created with the expert tag set.

The collection of learner tags was compared with the tags given independently by 4 experts in the field. The expert tag set was comprised of 165 tags of which 100 were different.

Within expert tag set, we elaborated two research questions:

1. Which learning objects can be found by a simulated query with the expert tags on the complete set of learner tags and which relevance (number of matching tags) does it have? With respect to this question, we found that the ratio of matches was in average 45% of the expert tags also assigned to a learning object by the learners.
2. How many keywords assigned as tags are already present as text in the LO? This question addresses if the tags given to the learning items stay close to the original item. The results was that experts tend to tag more abstractly and conceptual then learners. According to Sen et al. (2006) categories (as we desribed in section 5.3.4), the distribution was 73% factual, 16% subjective, 4% personal and 7% other.

Given that 55% of the expert tags were tags not within the body of tags used by learners, we question the benefits of providing these tags to learners at all. The lack of expert time and willingness to fill in metadata has been cited (Friesen, 2001; IMS meta-data best practice guide for IEEE, 2004), as a significant hurdle to deploying learning objects. If expert tags provide limited value to learners, it may be more appropriate to bootstrap data sets with automatic tagging features and reduce the load on those who are creating content. We note the potential pedagogical benefits of collaborative tagging as suggested by (Jones et al., 2006): that the tags themselves represent the expertise of the users. This proposes that at a collaborative level, a tag set can be observed as the course is being given by the experts to improvement an insight into the topics and concepts that learners are filtering from the online material.

Beyond the issue of expert time is the issue of control in the classroom. Unlike the open web, where individual success is evaluated by the individual, success in e-learning systems is typically dictated through a series of educator prepared exams. It have observed (Bateman et al., 2007) that educators are hesitant to change their teaching to adopt new methods in the classroom (virtual or otherwise), because of a loss of control. By engaging educators actively in the process of creating tags, it may reduce their fears of these new technologies. However, our results showed only 45% of the expert tags were represented in the tags of the learners. Also Halpin et al. (2007) suggested that unlike open a web system, the educator in the classroom is not merely a peer, and their tags may be more relevant to the examinations, which may be useful to learners. Therefore end-use of tags in an educational context is of significant interest to us.

5.7. Evaluation of the Protus System from the Educational Point of View

Educational research measures are needed to evaluate whether learners actually do benefit from the usage of the recommender system. From the educational point of view, learners only benefit from learning technology when it makes learning more effective, efficient or attractive. Efficiency indicates the time that learners needed to reach their learning goal. Effectiveness is a measure of the total amount of completed, visited, or studied lessons during a learning phase (Drachsler et al., 2009). In our study, we track only lessons that are successfully completed, meaning that learners passed the appropriate test at the end of the particular lesson. It is related to the efficiency variable through counting the actual study time. To answer this question, we randomly selected a sample of 100 learners from the experimental group and 100 learners from the control group. The results of the experiment showed that the learners in the experimental group should be able to complete a course in less time than learners in the control group who learned with the previous version of the system (Figure 5.13). Figure 5.14 shows that the experimental group continuously completed more lessons successfully than the control group.

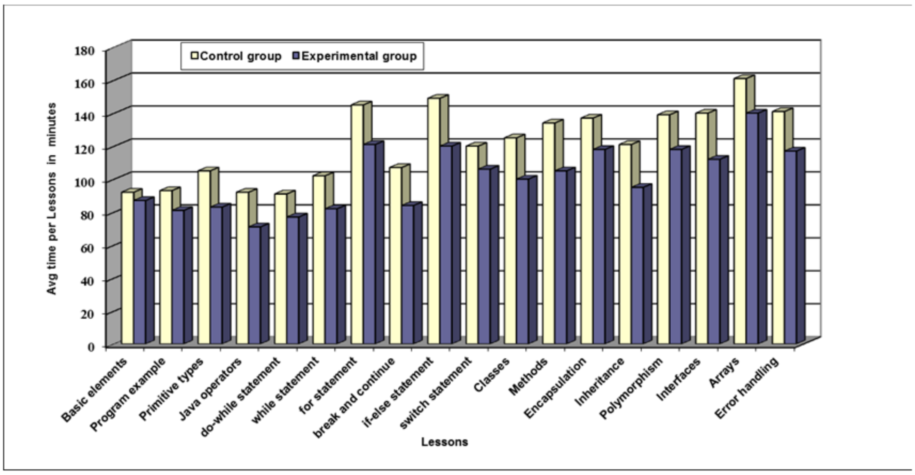

Figure 5.13 Efficiency comparison between groups

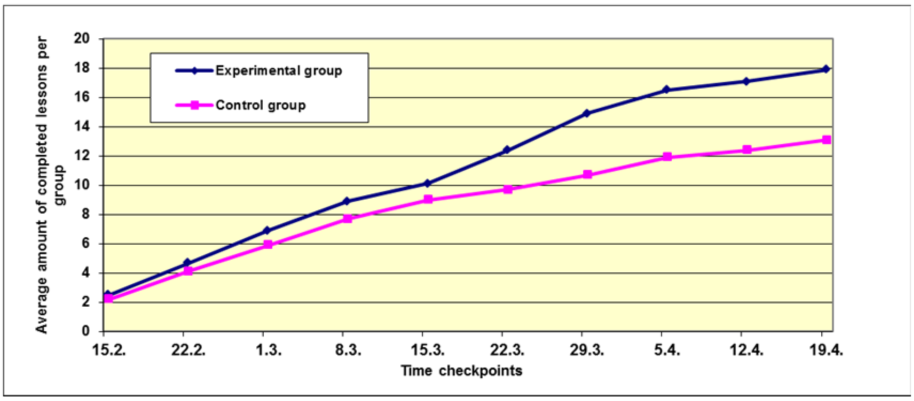

Figure 5.14 Average completions of lessons per group

Satisfaction reflects the individual satisfaction of learners with the given recommendations. Satisfaction is closely related to the motivation of the learner and therefore a rather important measure for learning. To get subjective evaluation of our system, at the end of the course we organized a non-mandatory questionnaire that collected learners' (from the experimental group) opinions about the main features of the system. The results of the questionnaire were used to improve the quality of lessons. This questionnaire (Table 5.5) maps a set of 16 questions over 4 dimensions: 'ease of tagging', 'usefulness of tagging', 'usefulness of the Protus system and tag exchange', and 'Protus system is easy-to-use'.

Out of 100 learners, 75 filled in the questionnaire. Participants are asked to give a level of agreement to each question on a 1-to-5 scale (strongly disagree, disagree, neutral, agree, and strongly agree). For example, the fifth question in Table 5.5 is about the relevance and unexpectedness of tags suggested by the Protus. A response of 1 would mean a learner strongly disagree, while 5 would mean a learner strongly agree with statement: "the tags suggested by the Protus are both relevant and unexpected".

Table 5.5 Questionnaire - Analysis of satisfaction with Protus system

Question	Response				
	1	2	3	4	5
1. I can easily construct meaningful words or phrases to represent the learning objects with tags.					
2. I can clearly indicate the meaning of tags which describes the context of learning objects.					
3. The proposed tags allow me to express the right term when entering tags.					
4. Reviews of my previous own tags facilitate the recall of learning objects' ideas and information.					
5. The tags suggested by the Protus are both relevant and unexpected.					
6. I am comfortable with others knowing what I mean about learning object.					
7. I am interested in seeing others who are likeminded to me regarding specific topics.					
8. Having a high level of influence on my neighbors is important to me.					
9. I think the using tags enable me to easily grasp the structure and concepts of learning objects.					
10. I think that exploiting tagging information provide me with effective feedback during the learning process.					
11. The tagging activities inspire me to make new ideas.					
12. I think that Protus's user interface is simple to learn (understand) and efficient to use.					
13. I can start quickly with the Protus system.					
14. I believe that Protus is effective at learning my preferences.					
15. I am satisfied with Protus's recommendations.					
16. I trust that Protus has ability to make correct recommendations for me.					

To examine the internal consistency and content validity of this survey, Cronbach's alpha coefficient was calculated for the 20-item questionnaire. Cronbach's α (alpha) (Bland and Altman, 1997) is a coefficient of reliability. It is usually used as a measure of the internal stability or consistency of a psychometric test score for a sample of examines. It was first named alpha by Lee Cronbach in 1951, as he had intended to continue with further coefficients. Alpha is not robust against omitted data. Several other Greek letters have been used by later researchers to assign other measures used in a similar context (Cortina, 1993). Cronbach's α is defined as:

$$\alpha = \frac{K}{K-1}\left(1 - \frac{\sum_{i=1}^{K}\sigma_{Y_i}^2}{\sigma_X^2}\right)$$

where *K* is the number of *items* or *testlets*, σ_X^2 is the variance of the observed entire test scores, and $\sigma_{Y_i}^2$ the variance of item *i* for the current sample of persons (Devellis, 1991). In order to determine if a question item is correlated with a factor, we applied the distinguish validity test by using the factor analysis method to observe each question item. Four factors among these items are shown in table 5.6. The eigenvalues of the four factors are greater than 1.00 with variance 68.12% explained. From the experimental results, it was found that some question items, were not correlated with factors (that is, their load was less than 0.5). As a result, 4 questions items were dropped, reducing the overall number to 16. In addition, the experiment shows that the internal reliability indexes of the four factors are 0.776, 0.833, 0.716, and 0.768, respectively.

The alpha coefficient is 0.814 after deleting non-correlated factors. Therefore, these results suggest that these factors were sufficiently reliable for representing learner tagging behaviors, when the Cronbach's α is higher than 0.7 (Hwang et al., 2008; Nunnally, 1978).

Table 5.6 Rotated factor loadings and Cronbach's *α* value for four factors

Items	Factor1	Factor2	Factor3	Factor4
Factor 1: Easy to tagging α=0.776				
I_1	0.672			
I_2	0.727			
I_3	0.812			
Factor 1: Usefulness of tagging				
α=0.833		0.614		
I_6		0.718		
I_7		0.588		
I_8				
Factor 1: Usefulness of tagging				
α=0.716			0.731	
I_9			0.708	
I_{10}			0.645	
I_{11}			0.811	
I_{12}				
Factor 1: Usefulness of tagging				
α=0.768				0.871
I_{15}				0.738
I_{16}				0.645
I_{17}				0.821
I_{18}				0.672
I_{19}				

α=0.814, total variance explained is 68,12%

The statistical analysis of the survey results is summarized in Table 5.7. The major findings are presented as follows:

(1) 97% of the learners indicated that creating tags was easy, and that it was easy to construct meaningful words or phrases to represent the learning objects with tagging objects.

(2) 85% of the learners thought that tagging activity can help learners summarize new ideas and quickly grasp the structure and concepts. Some learners indicated that their tags were more accurate after sufficient tagging practice.

(3) 93% of the learners agreed that Protus is capable of helping them to easily comprehend the context of learning objects, and can help them improve their learning efficiency.

(4) 94% of the learners regarded that Protus system is easy-to-use

Table 5.7 Statistical results of the questionnaire for evaluating the Protus system

Questionnaire item (four factors)	Strongly disagree (%)	Disagree (%)	Neutral (%)	Agree(%)	Strongly agree(%)
Easy to tagging in Protus	-	16.81	42.19	38.31	2.69
Usefulness of tagging in Protus	-	4.35	27.19	53.14	15.32
Usefulness of Protus system	-	-	25.71	68.1	6.19
Protus is easy-to-use	-	-	46	47.8	6.1

Chapter 6

Conclusions and Future Directions

Recommender systems made significant progress over the last decade when numerous content-based, collaborative, and hybrid methods were proposed and several “industrial-strength” systems have been developed. However, despite all of these advances, the current generation of recommender systems still requires further improvements to make recommendation methods more effective in a broader range of applications. With the increasing popularity of the collaborative tagging systems, surveyed in this dissertation, tags could be interesting and useful information to enhance recommender systems’ algorithms. Besides helping user organize his or her personal collections, a tag also can be regarded as a user’s personal opinion expression, while tagging can be considered as implicit rating or voting on the tagged information or items. Thus, the tagging information can be used to make recommendations.

6.1. Summary of Research

The research presented in this dissertation analyzed a model to select tags that reveal the preferences and characteristics of users required to generate personalized recommendations and options on the use of models for personalized tutoring system. The main research directions of the dissertation can therefore be summarized as follows.

Chapter 1 started with an introduction of the dissertation research aim. The chapter further highlighted research objectives of the dissertation.

Chapter 2 provided comprehensive survey of the state-of-the-art in recommender systems, collaborative tagging systems and folksonomy for tagging activities which can be used for extending the capabilities of recommender systems.

Chapter 3 presented theoretical overview of tag-based recommender systems in e-learning environments and identified the limitations of the current generation of collaborative tagging techniques and discussed about some approaches for extending their capabilities.

Chapter 4 considered adaptation based on learning styles and possibilities for applying a recommender system based on collaborative tagging techniques in developing a Protus tutoring system that adapts to the interests and level of learners' knowledge in the field of programming languages.

Chapter 5 analyzed statistical properties of learners' tagging history and presented evaluation of the performance of several suitable recommendation techniques for RS in e-learning environments and comparison of described techniques. Also, this chapter considered expert tag set in order to systematically verify the relationship between learning comprehension and learner data tagging, and in order to help teachers' evaluate learner knowledge. Finally, in this chapter evaluation of the Protus system from the educational point of view is performed.

Chapter 6 concludes the dissertation, summarizing the main contributions, and discussing the possibilities for future work.

6.2. Contributions of the Dissertation

As a part of Web 2.0, collaborative tagging is getting popular as an important tool to classify dynamic content for searching and sharing. We analyzed the potential of collaborative tagging systems, including personalized and biased user preference analysis, and specific and dynamic classification of content for applying collaborative tagging techniques into Java Tutoring system. Appropriate selection of collaborative tagging techniques could lead to applying the best results in terms of increasing motivation in learning process and understanding of the learning content. The scientific contributions are summarized as follows. First, in this thesis, we demonstrated how programming tutoring systems can be enabled to provide adaptivity based on learning styles. We introduced a general concept for tutoring system to automatically generate course that fit to the learning styles of the learners. The only additional effort for the teachers and course developers is to provide some meta-data in order to annotate the learning material. Furthermore, learners were asked to fill out the ILS questionnaire for detecting their learning styles. The concept was implemented and an experiment with 340 learners was performed to show the effectiveness of the realized concept. Then, we evaluated statistical properties of learners' tagging history. We studied how many LO were tagged on average by each learner in the system and found that even 65% learners show high activity, tagged between 50 and 72 LOs. In order to understand the characteristics of learner tags, and learner tagging behavior, we examined tag characteristics of learners in the Protus system. We have noted: as the number of learners increases and a tagged object receives more and more tags, the frequency of at which a tag is selected tends to become fixed. This concept of convergence or stabilization has important implications for the collective usefulness of individual tagging behavior. Likewise, this stabilization might appear during instances of shared knowledge, as well as when learners imitate the tag selection of other learners. Research on collaborative tagging process itself has found that the position of a tag correlates with its expressiveness. Tag entropy is a measurement of specificity where more general tags should have higher entropies because they might appear in different topics, whereas seldom tags are often more specific to a topic, thus have lower entropies. It appears that learners tend to assign common tags at the beginning of the tagging process and more specific tags later. A semantic analysis of tags was performed to better understand different utilization of tags. When we analyzed how these tags were used and re-used among learners, we found the vast majority of the tags were of the personal (44% of tags) and subjective type (40% of tags).

The most significant part of the research is focused on appropriate selection of collaborative tagging techniques which could lead to applying the best results in terms of increasing motivation in learning process and understanding of the learning content. As a result personalized and the most likely preferred recommendations can be estimated to an active learner that are in accordance with the learner's interests, his learning style, demographic characteristics and previously acquired knowledge. First, we compare simple methods based on counting tag occurrences (Most Popular Tags), specific approaches for improving the performance of such methods and an adaptation of User-based Collaborative Filtering, named Collaborative Filtering based on Tags. All algorithms perform significantly better than the baseline Most Popular Tags and the Most Popular Tags by User strategy, whereas it is much harder to beat the Most Popular Tags by Item. The idea to suggest the Most Popular Tags by Item results in a recall which is very similar to using the CF recommender based on user's item similarities. In contrast to these two approaches, the Most Popular Tags 0,6 Mix-recommender includes also the learner's tags in the recommendations. It is successful and could gain results better than those of CF.

Then, we analyzed the prediction quality of graph-based approaches, Adapted PageRank and FolkRank, and tensor based approaches, HOSVD and RTF. FolkRank builds on PageRank and proved to give significantly better tag recommendations than CF, because FolkRank has ability to exploit the information that is appropriate to the specific learner together with input from other learners via the integrating structure of the underlying hypergraph. When comparing the prediction quality of CF, Adapted PageRank and FolkRank one can see that FolkRank outperform both two. This method also allows for mode switching with no change in the algorithm. Moreover, as well as CF-based algorithms, FolkRank is robust against online updates since it does not need to be trained every time a new learner, item or tag enters the system. However, FolkRank is computationally expensive and not trivially scalable, making it more suitable for systems where real-time recommendations are not a requirement. Similarly to FolkRank, tensor factorization methods also work directly over the ternary relation of the folksonomy. Although the tensor reconstruction phase can be expensive, it can be performed offline. After the lower dimensional tensor is computed, the recommendations can be done quickly, making these algorithms appropriate for real-time recommendations. A possible drawback of tensor factorization methods is that easy mode switching can only be achieved if one considers that the different recommendation problems, i.e., learner/item/tag, can be addressed by minimizing the same error function. If one chooses HOSVD for example, the reconstructed tensor can be used for multi-mode recommendations with simple mode switching, but at the cost of solving the wrong problem: HOSVD minimizes a least-square error function while social tagging RS are more related to ranking. We selected only the best representatives of the considered techniques. We concluded that the best method is RTF followed by FolkRank and HOSVD.

Also, we have carried out other experiments to evaluate the performance of the system from the points of view of both teacher and learners. The results demonstrated the potential pedagogical benefits of collaborative tagging that the tags themselves represent the expertise of the users. This proposes that at a collaborative level, a tag set can be observed as the course is being given by the experts to improvement an insight into the topics and concepts that learners are filtering from the online material. The general opinion of experts has been very positive. They have demonstrated a high degree of motivation and have especially liked the novelty of using learners' data to improve e-learning courses, to be able to apply modifications to courses directly from the system and have the possibility of working and sharing information with other teachers and educational experts. However, experts have indicated that the creation of the repository or knowledge database is a hard task.

From the educational point of view, learners only benefit from learning technology when it makes learning more effective, efficient or attractive. The results of the experiment showed that the learners who were required to use the Protus system should be able to complete a course in less time than learners in the control group who learned with the previous version of the system. Also, these learners continuously completed more lessons successfully than the control group. To get subjective evaluation of our system, at the end of the course we organized a non-mandatory questionnaire that collected learners' opinions about the main features of the system. Results are very encouraging:

1. Learners indicated that creating tags was easy, and that it was easy to construct meaningful words or phrases to represent the learning objects with tagging objects.
2. Learners thought that tagging activity can help learners summarize new ideas and quickly grasp the structure and concepts. Some learners indicated that their tags were more accurate after sufficient tagging practice.
3. Learners agreed that Protus is capable of helping them to easily comprehend the context of learning objects, and can help them improve their learning efficiency.
4. Learners regarded that Protus system is easy-to-use.

6.3. Future Work

The rapid development of collaborative tagging system and related emerging technology suggests new ideas for personalized recommendation and determine a great number of challenges for future work.

Future studies could focus more specifically on measuring the impact of prior learner experience (with computers and the Internet) and interest (in the knowledge domain) on the effect of creating tags. Additionally, future studies could investigate whether there are more factors which also have an influence on the effect of choice of tags. Possible candidates could be mood or stress level.

Improvements in the experimental design could verify the findings reported in this dissertation and increase their external validity. Similar comparative studies could be carried out involving more learners and more experts and teachers from other areas (unrelated to computer science) in order to obtain a more heterogeneous teacher's profile. This will allow the study of other interesting questions such as: Is it possible that different teachers in different areas might coincide in their evaluation of patterns?; What is the behavior of experts and teachers as they progress through a course?; Can tuples that are found to be valid and useful in one course later be applied to another course with a different profile? These aspects could lead to a confirmation that would focus uniquely on a detailed analysis of the changes made and whether the process is efficient and likely to be corresponding to non-guided course content revision.

Even though the source code was written specifically for the Java programming course that was used in the experimental evaluation, it is feasible that with adequate programming effort, adapted versions of Protus can be created for other knowledge domains. We can, also, integrate other sequence mining algorithms (Han et al., 2005) such as SPADE, FreeSpan, CloSpan and PSP, and other clustering algorithm without demanding the learner to specify any parameter. We plan to evaluate the quality of the recommendations based on feedback from learners as well as on results using a testing set of data. Finally, it would be very useful to develop a real-time feedback loop between data mining and the recommendation system. We can use, for example, intelligent agents for doing on-line data mining automatically and for communicating with the recommender systems. In this way the system could work completely autonomously. The agents can mine data only when they notice enough volume of new data. And the authors do not have to preprocess and apply mining algorithms; they only have to organize the new recommender links if they want.

In the domain of tensor factorization for social tagging RS as a recent and prominent field, the research study on this area has just begun to expose the benefits that those methods have to proposal. A mainly interesting research direction considers investigating tensor factorization models that highlight both high recommendation accuracy and easy mode switching. As emphasized before, folksonomies usually do not contain numerical ratings, but recently the GroupLens[15] research group released a folksonomy dataset in which numerical ratings for the tagged items are also given (Marinho et al, 2011). This represents several research opportunities on how to exploit the item's rating information in order to improve recommendations. In this case, a single data structure for all the modes, such as tensors or hyper-graphs, would eventually fail since the ratings are only related to user-item pairs and not to tags. Similar issues can be investigated for content-based methods. It is proven that content-based methods usually neglect the user information, but past research shows that hybrid methods that combine user preferences with item's content usually conduct to better recommenders. Here, again, tensor or hyper-graph representations would be unsuccessful since items' content is only related to the items but not to the users or tags. So hybrid-based methods that achieve some kind of synthesis between folksonomy representations and items' content would be appreciated contribution to the area.

Finally, Protus proposed a new, dynamic approach to adaptive behavior in learning style-responsive environments. Future work will deal with an in-depth analysis of the results with respect to different learning style dimensions as well as the different adaptation features. We also plan to add more adaptation features to our concept and implement them. Another future direction will be to combine the proposed concept with an automatic learner modeling approach so that the system is able to automatically detect the learning styles of the learners based on their behavior and actions in the system.

[15] http://www.grouplens.org/

Bibliography

[1] Adomavicius G., and Tuzhilin A. (2005). 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*, v.17 n.6, 734-749.

[2] Agarwal A., Chakrabarti S., and Aggarwal S., (2006). Learning to rank networked entities, in *SIGKDD Conference*, 14–23.

[3] Agrawal R., Imielinski T., and Swami A., (1993). Database mining: A performance perspective. *IEEE Transactions on Knowledge and Data Engineering, Special Issue on Learning and Discovery in Knowledge-Based Databases*, 5(6), 914-925.

[4] Agrawal R., and Srikant R. (1995). Mining sequential patterns. In *Proceeding of 11th International Conference on Data Engineering*, 3–14.

[5] Aguzzoli S., Avesani P., and Massa P. (2002). *Collaborative case-based recommendation systems.* Lecture Notes in Computer Science, p. 2416

[6] Alton-Scheidl R., Schumutzer R., Sint P.P., and Tscherteu G. (Eds.), (1997), *Voting and rating in Web4Groups and other projects: approaches and first experiences*, Vienna, Austria: Oldenbourg, 13-103.

[7] Anane R., Crowther S., Beadle J., and Theodoropoulos G., (2004). e-learning Content Provision, *Proceedings of the 15th International Workshop on Database and Expert Systems Applications* (DEXA'04), 420-425.

[8] Anderson M., Ball M., Boley H., Greene S., Howse N., Lemire D., and McGrath S. (2003). RACOFI: a rule-applying collaborative filtering system. In *Proc. IEEE/WIC COLA'03* (eds A. Ghorbani & S. Marsh), National Research Council Canada, Halifax, NS. 53-72.

[9] Angehrn A., Nabeth T., Razmerita L., and Roda C. (2001). K-InCA: Using Artificial Agents for Helping People to Learn New Behaviours, *Proc. IEEE International Conference on Advanced Learning Technologies* (ICALT 2001), IEEE Computer Society, August 2001, Madison USA, 225 -226.

[10] Arenas-García J., Meng A., Petersen B., Schiøler L., Hansen K., and Larsen J. (2007). Unveiling music structure via PLSA similarity fusion. In: *arenas*. IEEE Press, 419–424.

[11] Avery C., and Zeckhauser, R. (1997). Recommender systems for evaluating computer messages*. Communications of the ACM 40, 3*, 88–89.

[12] Balabanović M., and Shoham Y. (1997). Fab: Content-based, collaborative recommendation. *Communications of the ACM 40*, 66–72.

[13] Bar-Ilan J., Shoham S., Idan A., Miller Y., and Shachak A. (2006). Structured vs. unstructured tagging - a case study. *In: Proceedings of the 15th international WWW conference*. Collaborative Web Tagging Workshop

[14] Basilico J., and Hofmann T. (2004). Unifying collaborative and content-based filtering. In *Proceedings of the 21st International Conference on Machine Learning. ACM Press*, Banff, Canada. 65–72.

[15] Basu C., Hirsh H., and W. Cohen, (1998). Recommendation as Classification: Using Social and Content-Based Information in Recommendation, Recommender Systems. Papers from 1998 Workshop, Technical Report WS-98-08, AAAI Press, 11-15.

[16] Batagelj V., and Zaversnik M. (2002). Generalized cores. CoRR cs.DS/ 0202039. Retrieved from http://arxiv.org/abs/cs/0202039

[17] Bateman S., McCalla G., and Brusilovsky P. (2007). Applying Collaborative Tagging to E-Learning. *Workshop Proc. of Tagging and Metadata for Social Information Organization in conjunction with the International World Wide Web Conference* (WWW 2007), Banff, Canada, 3-12.

[18] Bednarik R., Moreno A., and Myller N., (2006). Program visualization for programming education - Case of Jeliot 3, *Association for Computing Machinery New Zealand Bulletin*, 2 (2), Avalailable online from http://is-alt.massey.ac.nz/acmnz/bulletin/vol2/issue2

[19] Bieg C., and Diehl S. (2004). Education and technical design of a Web-based interactive tutorial on programming in Java. *Science of Computer Programming*, 25–36.

[20] Billsus D., and Pazzani M., (1998). Learning Collaborative Information Filters, *Proc. Int'l Conf. Machine Learning*, 46-54.

[21] Billsus D., and Pazzani M. (1999). A Hybrid User Model for News Story Classification, *Proceedings of the Seventh International Conference on User Modeling* (UM '99), Banff, Canada, June 20-24.

[22] Bland J.M., and Altman D.G. (1997). Statistics notes: Cronbach's alpha. BMJ 1997, 314-572.

[23] Bonifazi F., Levialdi S., and Rizzo P. (2002). A web-based annotation tool supporting e-learning, *AVI '02 Proceedings of the Working Conference on Advanced Visual Interfaces,* 123-128.

[24] Breese J. S., Heckerman D., and Kadie C. (1998). Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, Madison, WI, 43–52.

[25] Brin S., Motwai R., Ullman D., and Tsur S., (1997). Dynamic Itemset Counting and Implication Rules for Market Basket Data, *ACM SIGMOD Conference on Management of Data*, 265-276.

[26] Brin S., Page L. (1998). The anatomy of a large-scale hypertextual web search engine. *Computer Networks ISDN Syst* 30(1–7). 107–117.

[27] Brusilovsky P., Schwarz E., Weber G. (1996). ELM-ART: An Intelligent Tutoring System on the World Wide Web, Third International Conference on Intelligent Tutoring Systems, Montreal, Canada, 261-269.

[28] Carman M., Baillie M., and Crestani F. (2008). Tag data and personalized information retrieval, In *SSM '08: Proc. of the 2008 ACM workshop on Search in social media*. ACM, 27-34.

[29] Carver C.A., Howard R.A. and Lane W.D (1999). Addressing different learning styles through course hypermedia, IEEE Transactions on Education, 42, pp. 33-38.

[30] Castro F., Vellido A., Nebot A., and Mugica F. (2007). Applying Data Mining Techniques to e-Learning Problems, Evolution of teaching and learning paradigms in intelligent environment 62, Springer, 183–221.

[31] Cattuto C., Loreto V., and Pietronero L. (2007). Semiotic dynamics and collaborative tagging. *Proc Natl Acad Sci* USA 104(5), 1461–1464.

[32] Chen W., and Wasson B. (2002). Coordinating Collaborative Knowledge Building. *Proc of International Conference Applied informatics*. Vol. 25(1), 1-10.

[33] Chen C.M. (2008). Ontology-based concept map for planning a personalized learning path. *British Journal of Educational Technology*, Blackwell publishing, 1-31.

[34] Cheung K.W., Kwok J.T., Law M.H., and Tsui K.C., (2003). Mining customer product ratings for personalized marketing, *Decision Support Systems* Vol. 35, 231-243.

[35] Cho K., Schunn C. D., and Wilson R. W. (2006). Validity and Reliability of Scaffolded Peer Assessment of Writing from Instructor and Student Perspectives. *Journal of Educational Psychology*, Vol. 98, No. 4, 891—901.

[36] Cho K., and Schunn C. D. (2007). Scaffolded writing and rewriting in the discipline: A web-based reciprocal peer review system. *Computers and Education*, 48(3), 409-426.

[37] Chu K., Chang M., and Hsia Y. (2003). Designing a Course Recommendation System on Web based on the Students' Course Selection Records. In: *World Conference on Educational Multimedia, Hypermedia and Telecommunications* 14-21.

[38] Ciro C., Schmitz C., Baldassarri A., Servedio V., Loreto V., and Hotho A. (2007). Network properties of folksonomies. *AI Commun* 20(4), 245–262.

[39] Claypool M., Gokhale A., Miranda T., Murnikov P., Netes D., and Sartin M., (1999). Combining Content-Based and Collaborative Filters in an Online Newspaper, *Proc. ACM SIGIR '99 Workshop Recommender Systems: Algorithms and Evaluation*, 119–128.

[40] Cohn D., and Hofmann T. (2000). The missing link - a probabilistic model of document content and hypertext connectivity. *In Leen TK, Dietterich TG, Tresp V (eds) NIPS*. MIT Press, 430–436

[41] Cortina J. M. (1993). What is coefficient alpha? An examination of theory and applications. *Journal of Applied Psychology, 78*, 98- 104.

[42] Cosley D., Lawrence S., and Pennock D. (2002). An open framework for practical testing of recommender systems using ResearchIndex. In *28th International Conference on Very Large Databases*. Morgan Kaufmann, Hong Kong, China, 35–46.

[43] Costabile, M. F., De Marsico, M., Lanzilotti, R., Plantamura, V. L., and Roselli, T. (2005). *On the Usability Evaluation of E-Learning Applications*. IEEE

[44] Dahl D., and Vossen G. (2008). Evolution of learning folksonomies: social tagging in e-learning repositories, *International Journal of Technology Enhanced Learning* 1(1), Inderscience, 35-46.

[45] De Bra P., Aroyo L., and Chepegin V. (2004). The next big thing: adaptive Web-based systems. *Journal of Digital Information 5*, Article No. 247, p.12

[46] De Lathauwer L., (1997). *Signal Processing based on Multilinear Algebra*, doctoral dissertation, University of Leuven, Belgium

[47] De Lathauwer L., De Moor B., and Vandewalle J. (2000). Multilinear singular value decomposition. *SIAM Journal of Matrix Analysis and Applications*, 21(4): 1253–1278.

[48] Deerwester C., Dumais T., Landauer K., Furnas W., and Harshman A. (1990). Indexing by latent semantic analysis. *J. ASIS 41*, 6, 391–407.

[49] Deshpande M., and Karypis G. (2004). Item-based top-n recommendation algorithms. *ACM Trans. Inf. Syst*., 22(1), 143–177.

[50] Devedžić V. (2004). Education and the Semantic Web. *International Journal of Artificial Intelligence in Education 14,* IOS Press, 39-65.

[51] Devellis R.F. (1991). *Scale Development*, Sage Publications, pp. 24–33.

[52] Dos Santos M.L., and Becker K. (2003). Distance Education: a Web Usage Mining Case Study for the Evaluation of Learning Sites. In: *The 3rd IEEE International Conference on Advanced Learning Technologies, ICALT'03.* IEEE Computer Society. Athens Greece, 360-361.

[53] Doush I. (2011). Annotations, Collaborative Tagging, and Searching Mathematics in E-Learning IJACSA *International Journal of Advanced Computer Science and Applications* 2(4), 30-39.

[54] Drachsler H., Hummel H., and Koper R. (2007). Recommendations for learners are different: Applying memory-based recommender system techniques to lifelong

learning, *Proceedings of Workshop on Social Information Retrieval for Technology-Enhanced Learning* (SIRTEL'07) at the EC-TEL conference. 18-26.

[55] Drachsler H., Hummel H., and Koper R. (2008). Personal recommender systems for learners in lifelong learning networks: the requirements, techniques and model *International Journal of Learning Technology* 3 (4), 404-423.

[56] Drachsler H., Hummel H., and Koper R. (2009). Identifying the Goal, User model and Conditions of Recommender Systems for Formal and Informal Learning. *Journal of Digital Information (JODI) 10(2)*, 4-24.

[57] Dunn R., Dunn K., and Freeley M. E. (1984). Practical applications of the research: responding to students' learning styles-step one. *Illinois State Research and Development Journal*, 21(1), 1–21.

[58] Elorriaga J. A., and Fernandez-Castro I. (2000). Using Case-Based reasoning in Instructional Planning. Towards a Hybrid Self-improving Instructional Planner. In: *International Journal of Artificial Intelligence in Education*, vol. 11, 416-449.

[59] Emurian H. (2006). A web-based tutor for java: evidence of meaningful learning. *International Journal of Distance Education Technologies*, Vol.4, No. 2, 10-30.

[60] Felder M., and Soloman A. (1996). *Index of learning styles questionnaire*. Retrieved from http://www.engr.ncsu.edu/learningstyles/ilsweb.html.

[61] Freyberger J., Heffernan N., and Ruiz C. (2004). Using association rules to guide a search for best fitting transfer models of student learning. *In: Workshop on Analyzing Student-Tutor Interactions Logs to Improve Educational Outcomes at ITS Conference*, 1-10.

[62] Friesen N. (2001). What are educational objects? *Interactive Learning Environments*, vol 9, no. 3, 219-230.

[63] Funk P., and Conlan. O. (2003). Using Case-Based Reasoning to Support Authors of Adaptive Hypermedia Systems. In *AH2003: Workshop on Adaptive Hypermedia and Adaptive Web-Based Systems*, 113-120.

[64] García P., Amandi A., Schiaffino S., and Campo M. (2007). Evaluating Bayesian networks' precision for detecting students' learning styles. *Computers & Education*, 49, 794–808.

[65] García E., Romero C., Ventura S., and de Castro C. (2009). An architecture for making recommendations to courseware authors using association rule mining and collaborative filtering. *User Modeling and User-Adapted Interaction*. 19(1-2), 99-132.

[66] Gaudioso E., Santos O., Rodriguez A., and Boticario J. (2003). A Proposal for Modeling a Collaborative Task in a Web-Based Collaborative Learning Environment, in Proc. *9th. International Conference on User Modeling.* USA.

[67] Gehringer E. F. (2001). Electronic Peer-Review and Peer Grading in Computer-Science Courses. In *Proceedings of the 32nd SIGCSE Technical Symposium on Computer Science Education*, Charlotte, North Carolina, United States, pp. 139 – 143.

[68] Ghali F., and Cristea A. (2009). MOT 2.0: A Case Study on the Usefuleness of Social Modeling for Personalized E-Learning Systems. *Frontiers in Artificial Intelligence and Applications*, Vol. 200, 333 - 340.

[69] Glaser R. (1963). Instructional technology and the measurement of learning outcomes. *American Psychologist*, 18, 510–522.

[70] Glenn B., Shahida P., Wei F., and Moritz S. (2004). A Web-based ITS for OO design. *12th International Conference on Artificial Intelligence in Education*, Amsterdam, 59–64.

[71] Godoy D., and Amandi A. (2008). Hybrid content and tag-based profiles for recommendation in collaborative tagging systems. *In: Proceedings of the 6th Latin American web congress (LA-WEB 2008)*. IEEE Computer Society Vila Velha, Brazil, 58–65.

[72] Godwin L., and Kaplan S. (2008). Designing ee-Learning Environments: Lessons from an Online Workshop, *Innovate 4(4)*, Retrieved from http://www.innovateonline.info/index.php?view=article&id=489

[73] Golder A., and Huberman A. (2005). *The structure of collaborative tagging systems*. HPL Technical Report

[74] Golder S., and Huberman A. B. (2006). The Structure of Collaborative Tagging Systems, *Journal of Information Sciences 32(2)*, April 2006, 198—208.

[75] Goldberg D., Nichols D., Oki B., and Terry D. (1992). Using collaborative filtering to weave an information tapestry. *Communications of the ACM* 35, 12, 61–70.

[76] Goldberg K., Roeder T., Gupta D., and Perkins C., (2001). Eigentaste: A Constant Time Collaborative Filtering Algorithm, *Information Retrieval J.*, vol. 4, no. 2, pp. 133-151.

[77] Golder A, and Huberman A. (2005). *The structure of collaborative tagging systems*. HPL Technical Report

[78] Gordon-Murnane L (2006). Social bookmarking, folksonomies, and Web 2.0 tools. *Searcher* 14(6): 26–38.

[79] Graf S., (2007). *Adaptivity in Learning Management Systems Focussing on Learning Styles,* Vienna University of Technology, Ph.D. Thesis

[80] Graf S. and Kinshuk C. (2010a). A Flexible Mechanism for Providing Adaptivity Based on Learning Styles in Learning Management Systems. In 2010 10th IEEE International Conference on Advanced Learning Technologies, pages 30–34. IEEE.

[81] Graf S., Liu T. C., and Kinshuk (2010b). Analysis of learners' navigational behavior and their learning styles in an online course. *Journal of Computer Assisted Learning*, Vol.26, No. 2, 116-131.

[82] Guy M., and Tonkin E. (2006). Folksonomies: tidying up tags? *D-Lib Magazine*, 12(1), Retrieved from http://www.dlib.org/dlib/january06/guy/01guy.html

[83] Halpin H., Robu V., and Shepherd H. (2007). The Complex Dynamics of Collaborative Tagging, *Proceedings of the 16th international conference on World Wide Web,* ACM, USA, 211 – 220.

[84] Halpin H., Robu V., and Shepard H. (2006). The dynamics and semantics of collaborative tagging. In: *Proceedings of the 1st semantic authoring and annotation workshop* — SAAW06

[85] Han J., Pei J., and Yan X. (2005). Sequential Pattern Mining by Pattern-Growth: Principles and Extensions. StudFuzz. Springer. 183-220.

[86] Herlocker J., Konstan J., Borchers A., and Riedl J. (1999). An algorithmic framework for performing collaborative filtering. In: *Proceedings of ACM SIGIR conference '99*. 230–237.

[87] Herlocker J.L., Konstan J. A., Terveen L. G., and Riedl J. (2004). Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst. 22(1)*. 5-53.

[88] Heraud J., France L., and Mille A. (2004). Pixed: an ITS that Guides Students with the Help of Learners' Interaction Log. In: *International Conference on Intelligent Tutoring Systems, Workshop Analyzing Student Tutor Interaction Logs to Improve Educational Outcomes*. Maceio, Brazil 57-64.

[89] Holland J., Mitrovic A., and Martin B. (2009). J-Latte: a constraint-based tutor for java. *In 17th International Conference on Computers in Education*, Hong Kong, 142-146.

[90] Hofmann T. (1999). Probabilistic latent semantic analysis. In: *Proceedings of uncertainty in artificial intelligence* - UAI '99.

[91] Hofmann T. (2003). Collaborative Filtering via Gaussian Probabilistic Latent Semantic Analysis, *Proc. 26th Int'l ACM SIGIR Conf. Research and Development in Information Retrieval* (SIGIR '03), 259-266.

[92] Hotho A., Jäschkes R., Schmitz C., and Stumme G. (2006a). Information retrieval in folksonomies: search and ranking. In: Sure Y, Domingue J (eds) *The Semantic web: research and applications*, vol 4011 of LNAI. Springer, Heidelberg, 411–426.

[93] Hotho A., Jäschke R., Schmitz C., and Stumme G. (2006b). BibSonomy: a social bookmark and publication sharing system. In: *Proceedings of first conceptual structures tool interoperability workshop*. Aalborg, 87–102.

[94] Hotho A., Jäschke R., Schmitz C., and Stumme G. (2006c). FolkRank: a ranking algorithm for folksonomies. In: *Proceedings of workshop on information retrieval (FGIR)*. Germany

[95] Hsu H.H., Chen C.H., and Tai W.P. (2003). Towards Error-Free and Personalized Web-Based Courses. In: The 17th International Conference on Advanced Information Networking and Applications, AINA'03. March 27-29, Xian, China, 99-104.

[96] Hwang J., Hsiao L., and Tseng R. (2003). A Computer-Assisted Approach to Diagnosing Student Learning Problems in Science Courses. *Journal of Information Science and Engineering* 19, 229-248.

[97] Hwang G. J., Tsai P. S., Tsai C. C., and Tseng Judy. C. R. (2008). A novel approach for assisting teachers in analyzing student web-searching behaviors. *Computers & Education*, 51(2), 926-938.

[98] Ivanović M., Pribela I., Vesin B., and Budimac Z. (2008). Multifunctional Environment for E-Learning Purposes. Novi Sad Journal of Mathematics, *NSJOM* Vol. 38, No. 2, 153-170.

[99] Janssen J., Van den Berg B., Tattersall C., Hummel H., and Koper R. (2007). Navigational support in lifelong learning: enhancing effectiveness through indirect social navigation. *Interactive Learning Environments*, 15(2), 127–136.

[100] Jäschke R., Marinho L., Hotho A., Schmidt-Thieme L., and Stumme G. (2007). Tag Recommendations in Folksonomies, *Knowledge Discovery in Databases PKDD*, Volume: 4702, Issue: May, Springer, 506-514.

[101] Jones N., Macasek M., Walonoski J., Rasmussen K., and Heffernan N. (2006). Common Tutor Object Platform – an e-Learning Software Development Strategy. *Proceedings of the 15th international conference on World Wide Web*, Edinburgh, Scotland, pp. 307-316.

[102] Karen H. L., Tso-Sutter, Marinho L., and Schmidt-Thieme L. (2008). Tag-aware recommender systems by fusion of collaborative filtering algorithms*, SAC '08 Proceedings of the 2008 ACM symposium on Applied computing*, 1995-1999.

[103] Klašnja-Milićević A., Vesin B., Ivanović M., and Budimac Z. (2009). Integration of recommendations into Java tutoring system. *The International Conference on Information Technology*, Amman, Jordan, CD proceedings, paper no. 524.

[104] Klašnja-Milićević A., Nanopoulos A., and Ivanović M. (2010). Social tagging in recommender systems: a survey of the state – of –the - art and possible extensions, *Artificial Intelligence Review*, Volume: 33, Issue: 3, Springer Netherlands, 187-209.

[105] Klašnja-Milićević A., Vesin B., Ivanović M., and Budimac Z. (2011a). Integration of recommendations and adaptive hypermedia into java tutoring system. *Computer Science and Information Systems*, Volume 8, Number 1, 211-224.

[106] Klašnja-Milićević A., Vesin B., Ivanović M., and Budimac Z. (2011b). e-learning personalization based on hybrid recommendation strategy and learning style identification, *Computers & Education, Volume 56, Issue 3*, 885–899.

[107] Kolb, D. (1984). Individuality in learning and the concept of learning styles. *In Experimential learning,* Englewood Cliffs, New Jersey: Prentice-Hall, 61–98.

[108] Konstan J., Herlocker J., Terveen Loren G., and Riedl T. (2004). Evaluating collaborative filtering recommender systems. *ACM Trans Inf Syst* 22(1), 5–53.

[109] Koper R., and Olivier B. (2004). Representing the learning design of units of learning', *Education, technology and society*, 7(3), 97-111.

[110] Kuljis J., and Liu, F. (2005). A comparison of learning style theories on the suitability for elearning, in M. H. Hamza, ed., Proceedings of the IASTED Conference on Web Technologies, Applications, and Services, ACTA Press, Calgary, Alberta, pp. 191-197.

[111] Kumar A. (2005). Rule-Based Adaptive Problem Generation in Programming Tutors and its Evaluation. In: *12th International Conference on Artificial Intelligence in Education*. 36-44.

[112] Landauer T., Foltz P., and Laham D. (1998). Introduction to Latent Semantic Analysis, *Discourse Processes*, 25(25), 259-284.

[113] Lemire D. (2005). Scale and translation invariant collaborative filtering systems. *Information Retrieval* 8, 129–150.

[114] Lemire D., Boley H., McGrath S., and Ball M. (2005). Collaborative filtering and inference rules for context-aware learning object recommendation. *International Journal of Interactive Technology and Smart Education* 2, 179–188.

[115] Li J., and Zaïane O.R. (2004). Combining usage, content, and structure data to improve web site recommendation. *The 5th International conference on e-commerce and web technologies*, Zaragoza, Spain, Springer, LNCS 3182, 305–315.

[116] Liang H., Xu Y., Li Y., and Nayak R. (2008). Collaborative filtering recommender systems using tag information. In: *IEEE/WIC/ACM international conference on web intelligence and intelligent agent technology*. 59–62.

[117] Lin X., Beaudoin J.E., Bui Y., and Desai K. (2006). Exploring characteristics of social classification. In: 17th ASIS&T SIG/CR classification research workshop, (19 pages). Austin, TX, Retrieved from http://en.scientificcommons.org/43542511pp

[118] Linden G., Smith B., and York J., (2003). Amazon.com Recommendations: Item-to-Item Collaborative Filtering, *IEEE Internet Computing 7(1)*, 76-80.

[119] Liu B., Wynne H., Shu C., and Yiming M. (2000). Analyzing the Subjective Interestingness of Association Rules. IEEE Intelligent System 15:5, 47-55.

[120] Lu J., (2004). A Personalized e-Learning Material Recommender System, *Proceedings of the 2nd International Conference on Information Technology for Application* (ICITA 2004). 374-380.

[121] Luo S., Sha S., Shen D., and Jia W.J., (2002). Conceptual Network Based Courseware Navigation and Web Presentation Mechanisms, In *Proceedings of Advanced in Web-Based Learning* (ICWL 2002), LNCS 2436, 81-91.

[122] Lynch C., Ashley K., Aleven V., and Pinkwart N. (2006). Defining Ill-Defined Domains; A Literature Survey. In *Proceedings of the Workshop on Intelligent Tutoring Systems for Ill-Defined Domains at the 8th International Conference on Intelligent Tutoring Systems*, 1-10.

[123] MacLaurin B. (2007). Selection-based item tagging. *Patent Application* no. US 2007/0028171 A1.

[124] Mallinson B., and Sewry D. (2004). E-learning at Rhodes University – a case study, *Proceedings of the IEEE International Conference on Advanced Learning Technologies* (ICALT'04), 708-711.

[125] Malone T., Grant K., Turbak F., Brobst S., and Cohen M. (1987). Intelligent information-sharing systems. *Communications of the ACM* 30, 5, 390–402.

[126] Manouselis N., and Costopoulou C. (2007). Experimental analysis of design choices in a multi-attribute utility collaborative filtering system. *International Journal of Pattern Recognition and Artificial Intelligence* 21, 311–331.

[127] Manouselis N., Drachsler H., Vuorikari R., and Hummel H. (2011). *Recommender Systems in Technology Enhanced Learning*, In: Recommender Systems Handbook, 387-415.

[128] Marinho B., and Schmidt-Thieme L. (2007). Collaborative tag recommendations. *In Proceedings of 31st Annual Conference of the Gesellschaft f¨ur Klassifikation* (GfKl), Freiburg, pp 533–540. Springer

[129] Marinho B., Nanopoulos A., Schmidt-Thieme L., Jäschke R., Hotho A., Stumme G., and Symeonidis P. (2011). *Social Tagging Recommender Systems*. In: Recommender Systems Handbook: 615-644.

[130] Markellou P., Mousourouli I., Spiros S., and Tsakalidis A. (2005). Using Semantic Web Mining Technologies for Personalized e-Learning Experiences. *In: Proc. of the Int. Conference on Web-based Education*, pp. 1-10.

[131] Marlin B. (2003). Modeling User Rating Profiles for Collaborative Filtering, *Proc. 17th Ann. Conf. Neural Information Processing Systems (NIPS '03)*,

[132] Marlow C., Naaman M., Boyd D., and Davis M. (2006). Ht06, tagging paper, taxonomy, flickr, academic article, to read. In: *HYPERTEXT '06: proceedings of the seventeenth conference on Hypertext and hypermedia*. ACM Press, New York, 31–40.D.A.

[133] Massa P., and Avesani P. (2004). Trust-aware Collaborative Filtering for Recommender Systems, *CoopIS/DOA/ODBASE* (1), 492-508.

[134] Mathes A. (2004). *Folksonomies - cooperative classification and communication through shared metadata*. Computer Mediated Communication, Graduate School of Library and Information Science, University of Illinois Urbana-Champaign. Retrieved from http://www.adammathes.com/academic/computer-mediatedcommunication/folksonomies.html

[135] Matsui T., and Okamoto T. (2003). Knowledge Discovery from Learning History Data and its Effective Use for Learning Process Assessment under the e-Learning Environment. In: Crawford, C., et al. (eds.): *Society for Information Technology and Teacher Education International Conference*, 3141-3144.

[136] McLaughlin M., and Herlocker J. (2004). A collaborative filtering algorithm and evaluation metric that accurately model the user experience, Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval ACM New York, USA

[137] McNee M., Riedl J., and Konstan A. (2006). Making Recommendations Better: An Analytic Model for Human-Recommender Interaction. A Work-In-Progress paper in the *Extended Abstracts of the 2006 ACM Conference on Human Factors in Computing Systems (CHI 2006)*, Montreal, Canada, 1003-1008.

[138] Melville P., Mooney R., and Nagarajan R. (2002). Content-boosted collaborative filtering for improved recommendations. In *Eighteenth National Conference on*

Artificial Intelligence. American Association for Artificial Intelligence, Edmonton, Canada, 187–192.

[139] Merceron A., and Yacef K. (2004). Mining student data captured from a web-based tutoring tool*. Journal of Interactive Learning Research*, 15:4 319–346.

[140] Mika P. (2005). Ontologies are us: a unified model of social networks and semantics. In: *Proceedings of the 4th international semantic web conference, ISWC 2005*. Springer, Galway, 522–536.

[141] Millen D., Feinberg J., and Kerr B. (2006). Dogear: Social Bookmarking in the enterprise. in *Proceedings of the SIGCHI conference on Human Factors in computing systems* Montreal, Canada, doi10.1145/1124772.1124792

[142] Miller B. (2003). *Toward a personalized recommender system*. Ph.D. thesis, University of Minnesota, Minneapolis, MA, USA.

[143] Minaei-Bidgoli B., Tan P., and Punch W. (2004). Mining interesting contrast rules for a web-based educational system. *In: Proc. of the Int. Conf. on Machine Learning Applications*, 1-8.

[144] Myller N. (2007). Automatic prediction question generation during program visualization. *Electron. Notes Theor. Comput. Sci.* 178, 43–49.

[145] MySQL_Internals_Algorithms (2008). Retrieved from http://forge.mysql.com/wiki/MySQL_Internals_Algorithms

[146] Nichols D. (1998). Implicit rating and filtering. In *Proceedings of the Fifth DELOS Workshop on Filtering and Collaborative Filtering. ERCIM*, Budapest, Hungary, 31–36.

[147] Noll M., and Meinel C. (2007). Web search personalization via social bookmarking and tagging. In *Proc. of the 6th Int. Semantic Web Conf. and 2nd Asian Sem. Web Conf.* (ISWC/ASWC2007). Springer, 367-380

[148] Noll M., Au Yeung C., Gibbins N., Meinel C., and Shadbolt N. (2009). Telling experts from spammers: expertise ranking in folksonomies, *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieva*l, 612-619.

[149] Northrup P. A. (2001). Framework for designing interactivity into web-based instruction. *Educational Technology*, 41(2), 31–39.

[150] Nunnally J. C. (1978). *Psychometric theory* (2nd ed.). New York: McGraw-Hill

[151] Page L., Brin S., Motwani R., and Winograd T. (1998). The PageRank citation ranking: Bringing order to the web (Tech. Rep.). Stanford, CA: Stanford Digital Library Technologies Project.

[152] Palmatier A., and Bennet M. (1974). Note-taking habits of college students. *Journal of Reading*. Vol. 18, 215-218.

[153] Park S., Ming-Syan C., and Philip Yu., (1995). An Effective Hash Based Algorithm for Mining Association Rules, *Proc. of ACM SGMOD*, 175-185.

[154] Pennock M., and Horvitz E. (1999). Analysis of the Axiomatic Foundations of Collaborative Filtering, *AAAI Workshop on Artificial Intelligence for Electronic Commerce, National Conference on Artificial Intelligence (AAAI-99)*, Orlando, Florida

[155] Pennock M., Horvitz E., Lawrence S., and Giles L. (2000). Collaborative filtering by personality diagnosis: A hybrid memory and modelbased approach. *In Proc. of UAI*, 473–480.

[156] Peters I., and Stock G. (2007). Folksonomy and information retrieval. In: *Proceedings of the 70th annual meeting of the American society for information science and technology*, vol 45 CD-ROM

[157] Pierrakos D., Paliouras G., Papatheodorou C., and Spyropoulos C. (2003). Web usage mining as a tool for personalization: A survey. *User Modeling and User-Adapted Interaction*, 13, 311–372.

[158] Pinkwart N., Aleven V., Ashley K., and Lynch C. (2006). Toward Legal Argument Instruction with Graph Grammars and Collaborative Filtering Techniques. In *Proceedings of the 8th International Conference on Intelligent Tutoring Systems.* Lecture Notes in Computer Science 4053, 227-236. Berlin (Germany), Springer.

[159] Pinkwart N., Aleven V., Ashley K., and Lynch C. (2007). Evaluating Legal Argument Instruction with Graphical Representations Using LARGO. In *Proceedings of the 13th International Conference on Artificial Intelligence in Education*, 101-108. IOS Press.

[160] Pluzhenskaia M. (2006). Folksonomies or fauxsonomies: how social is social bookmarking? *17th ASIS&T SIG/CR classification research workshop*. Abstracts of Posters, 23–24.

[161] Popescu, E. (2010). Adaptation Provisioning with respect to Learning Styles in a Web-Based Educational System: An Experimental Study. *Journal of Computer Assisted Learning*, 26(4), Wiley, pp. 243–257.

[162] Quintarelli E. (2005). Folksonomies: power to the people. *ISKO Italy-UniMIB.* Retrieved from: http://www.dimat.unipv.it/biblio/isko/doc/folksonomies.htm

[163] Ramli A. (2005). Web usage mining using apriori algorithm: UUM learning care portal case. *In: Proc. of the Int. Conf. on Knowledge Management*, pp. 1-19.

[164] Recker M., and Walker A. (2003). Supporting 'word-of-mouth' social networks via collaborative information filtering. *Journal of Interactive Learning Research* 14, 79–98.

[165] Recker M., Walker A., and Lawless K. (2003). What do you recommend? Implementation and analyses of collaborative filtering of Web resources for education. *Instructional Science* 31, 229–316.

[166] Rendle S., Marinho B., Nanopoulos A., and Thieme S. (2009). Learning optimal ranking with tensor factorization for tag recommendation. In: KDD '09: *Proceeding of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, New York, NY, USA, ACM, 727-736.

[167] Resende D., and Pires T. (2001). An Ongoing Assessment Model for Distance Learning. In: Hamza, M.H. (ed.): *Fifth IASTED International Conference Internet and Multimedia Systems and Applications*. Acta Press 17-21.

[168] Resnick P., Iacovou N., Sushak M., Bergstrom P., and Riedl J., (1994). GroupLens: An Open Architecture for Collaborative Filtering of Netnews, *Proc. 1994 Computer Supported Cooperative Work Conf.* 175–186.

[169] Resnick P., and Varian H. (1997). *Recommender systems*. Communications of the ACM 40.

[170] Ricci F., Rokach L., Shapira B. and Kantor P. (2011). *Recommender Systems Handbook,* Springer, 842 p.

[171] Romero C., Ventura S., Bra D., and de Castro C. (2003). Discovering prediction rules in AHA! Courses', *En 9th International User Modeling Conference* (Vol. 2702), Berlin: Springer Verlag, 25-34.

[172] Romero C., Ventura S., and Bra D. (2004). Knowledge discovery with genetic programming for providing feedback to courseware author. User Modeling and User-Adapted Interaction: *The Journal of Personalization Research*, 14:5, 425–464.

[173] Romero C., and Ventura S. (2006). Educational data mining: a survey from 1995 to 2005. *Expert systems with Applications*, 33(1), 135-146.

[174] Rosset S., Neumann E., Eick U., Vatnik N., and Idan Y. (2002). Customer Lifetime Value Modeling and Its Use for Customer Retention Planning, *Proc. Eighth ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining* (KDD-2002), 56-69.

[175] Santos O., and Boticario J. (2008). Recommender Systems for Lifelong Learning inclusive scenarios. *ECAI 2008 - Workshop on Recommender Systems*, Patras, Greece, 45-49.

[176] Sarwar B. M., Karypis G., Konstan J. A., and Riedl J. (2001). *Item-based collaborative filtering recommendation algorithms.* Paper presented at the *World Wide Web Conference* (WWW10).

[177] Savidis A., Grammenos D., and Stephanidis C. (2006). Developing inclusive e-learning and e-entertainment to effectively accommodate learning difficulties. *Universal Access in the Information Society – International Journal,* 5(4), Springer 401-419.

[178] Schmitt S., and Bergmann R. (1999). Applying case-base reasoning technology for product selection and customization in electronic commerce environments, *12th Bled Electronic Commerce Conference*. Bled, Slovenia,

[179] Schafer B., Konstan A., and Riedl J., (2001). E-commerce recommendation applications, Data Mining and Knowledge Discovery, Vol. 5, 115–153.

[180] Sen S., Lam S., Rashid A., Cosley D., Frankowski D., Osterhouse J., Harper M., and Riedl J. (2006). Tagging, community, vocabulary, evolution. In the Proceedings of Conference on Computer Supported Cooperative Work (CSCW '06). November, 2006. Banff, Canada. 181-190.

[181] Shardanand U., and Maes P. (1995). Social Information Filtering: Algorithms for Automating "Word of Mouth", *In Proc. ACM CHI'95 Conf.*, 10-217.

[182] Shen L., and Shen R, (2005). Ontology-based Learning Content Recommendation, *International Journal of Continuing Engineering Education and Life-Long Learning*, v 15, n 3-6, 308-317.

[183] Shepitsen A., Gemmell J., Mobasher B., and Burke R. (2008). Personalized recommendation in social tagging systems using hierarchical clustering. In *RecSys '08: Proc. of the 2008 ACM conf. on Recommender systems.* ACM,

[184] Silberschatz A., and Tuzhilin A. (1996). What makes pattterns interesting in Knoledge discovery systems, *IEEE Trans. on Knowledge and Data Engineering*. 8(6), 970-974.

[185] Sood S., Owsley S., Hammond K., and Birnbaum L. (2007). Tagassist: Automatic tag suggestion for blog posts. In *ICWSM '07: Proceedings of the International Conference on Weblogs and Social Media*, Citeseer

[186] Soonthornphisaj N., Rojsattarat E., and Yim-ngam S., (2006). Smart E-Learning Using Recommender System, in: *International Conference on Intelligent Computing*, 518–523.

[187] Sormo F., and Aamodt A. (2002). Knowledge communication and CBR. In: *Workshop on Case-based Reasoning for Education and Training*, ECCBR, 47-61.

[188] Srikanth M., Tatu M., and D'Silva T. (2008). Tag recommendations using bookmark content. *In Proceedings of the ECML PKDD Discovery Challenge at 18th Europ. Conf. on Machine Learning (ECML'08) / 11th Europ. Conf. on Principles and Practice of Knowledge Discovery in Databases* (PKDD'08), 96-107.

[189] Srivastava J., Cooley R., Deshpande M., and Tan P. (2000). Web Usage Mining: Discovery and Applications of Usage Patterns from Web Data. ACM *SIGKDD Explorations*. 1(2), 12-23.

[190] Suarez M., and Sison R. (2008). Automatic construction of a bug library for objected-oriented novice Java programmer errors. ITS 2008, Springer-Verlag Berlin Heideberg, 184–193.

[191] Sykes E. R., and Franek F. (2003). An intelligent tutoring system prototype for learning to program Java. *The third IEEE International Conference on Advanced Learning Technologies ICALT'03 Athens*, Greece, 485–492.

[192] Sykes E. (2007). Developmental process model for the Java intelligent tutoring system. *Journal of Interactive Learning Research Chesapeake*, Vol. 18, No. 3, 399–410.

[193] Symeonidis P., Ruxanda M., Nanopoulos A., and Manolopoulos Y. (2008). Ternary semantic analysis of social tags for personalized music recommendation. *In Proc. the 9th International Conference on Music Information Retrieval*, Pennsylvania, USA, 219-224.

[194] Symeonidis P., Nanopoulos A., and Manolopoulos Y., (2010). Unified framework for providing recommendations in social tagging systems based on ternary

semantic analysis. *IEEE transactions on knowledge and data engineering*. 22(2), 179-192.

[195] Šimić G. (2004). The Multi-courses Tutoring System Design. *ComSIS* Vol. 1, No. 1, 141-155.

[196] Tan H., Guo J., and Li Y. (2008). E-learning Recommendation System, *International Conference on Computer Science and Software Engineering, CSSE,* vol. 5, 430-433.

[197] Tang T. Y., and McCalla G. (2005). Smart recommendation for an evolving e-learning system: architecture and experiment. *International Journal on e-Learning, 4(1)*, 105–129.

[198] Tong W., and Pi-Lian H. (2005). Web log mining by an improved Aprioriall algorithm. In: *Proceeding of world academy of science, engineering, and technology*, 97–100.

[199] Tsai C.J., Tseng S.S., and Lin C.Y. (2001). A Two-Phase Fuzzy Mining and Learning Algorithm for Adaptive Learning Environment. In: Alexandrov, V.N., et al. (eds.): *International Conference on Computational Science, ICCS 2001.* LNCS Vol. 2074. Springer-Verlag, Berlin Heidelberg New York 429-438.

[200] Tso-Sutter L., Marinho B., and Schmidt-Thieme L. (2008). Tag-aware recommender systems by fusion of collaborative filtering algorithms. In: *Proceedings of the 2008 ACM symposium on applied computing*. ACM, USA, 1995–1999.

[201] Veres C. (2006a). The language of folksonomies: what tags reveal about user classification. *Lecture Notes in Computer Science*, vol 3999, 58–69.

[202] Veres C. (2006b). Concept modeling by the masses: folksonomy structure and interoperability*. Lecture Notes in Computer Science*, vol 4215, 325–338.

[203] Vesin B. Ivanović M., Budimac Z., and Pribela I. (2007). Tutoring System for Distance Learning of Java Programming Language. *10th Symposium on Programming Languages and Software Tools SPLST*, Dobogókő, Hungary, pp. 310-320.

[204] Vesin B., Ivanović M., Budimac Z., and Pribela I. (2008). MILE - multifunctional integrated learning environment. In: *IADIS Multi Conference on Computer Science and Information Systems MCCSIS'2008*, Amsterdam, Netherlands. 104–108.

[205] Vesin B., Ivanović M., and Budimac Z. (2009). Learning management system for programming in java. *Sectio Computatorica, 31, Annales Universitatis Scientiarum De Rolando Eötvös Nominatae*, 75–92.

[206] Wal V. (2005). *Folksonomy definition and wikipedia*.

[207] Walker A., Recker M., Lawless K., and Wiley D. (2004). Collaborative information filtering: a review and an educational application. *International Journal of Artificial Intelligence in Education* 14, 1–26.

[208] Wang D., Bao Y., Yu G., and Wang G. (2002). Using Page Classification and Association Rule Mining for Personalized Recommendation in Distance Learning.

In: Fong, J., et al. (eds.): *International Conference on Web Based Learning*, ICWL 2002. LNCS Vol. 2436. Springer-Verlag, Berlin Heidelberg New York, 363-374.

[209] Wei F., Moritz S. H., Parvez S. M., and Blank G. D. (2005). A student model for object-oriented design and programming. *Journal of Computing Sciences in Colleges*, 260–273.

[210] Wetzker R., Said A., and Zimmermann C. (2009). Understanding the user: Personomy translation for tag recommendation, ECML *PKDD Discovery Challenge 2009 DC09,* Volume: 497, Citeseer, 275-*284.*

[211] Wetzker R., Zimmermann C., and Bauckhage C. (2010). I tag, you tag: Personomy translation for advanced user models. In *WSDM '10: Proc. Int. Conf. on Web search and data mining*, 71-80.

[212] Wiley D. A. (2000). Connecting learning objects to instructional design theory: A definition, a metaphor, and a taxonomy. Retrieved from http://reusability.org/read/chapters/wiley.doc

[213] Wilson D., Smyth B., and Sullivan D. (2003). Sparsity reduction in collaborative recommendation: a case-based approach. *Int'l Journal of Pattern Recognition and Artificial Intelligence*, 17(5), 863-884.

[214] Winget M. (2006). User-defined classification on the online photo sharing site Flickr…or, how I learned to stop worrying and love the million typing monkeys. *17th ASIS&T SIG/CR classification research workshop,* 1-16.

[215] Wolf, C. (2003). iWeaver: towards 'learning style' - based e-Learning in computer science education. *Australasian Computing Education Conference*, Adelaide, Australia 20, 273–279.

[216] Wu X., Zhang L., and Yu Y. (2006). Exploring social annotations for the semantic web. In: *WWW'06: proceedings of the 15th international conference on World Wide Web*. ACM Press, New York, 417–426.

[217] Xu Z., Fu Y., Mao J., and Su D. (2006). Towards the semantic web: collaborative tag suggestions. In: *Proceedings of the 15th international WWW conference*. Collaborative Web Tagging Workshop

[218] Yang Y. and Wu C. (2009). Attribute-based ant colony systems for adaptive learning object recommendation, *Expert Systems with Applications 36(2)*, Elsevier Ltd, 3034-3047.

[219] Yu P., Own C. and Lin L. (2001). On learning behavior analysis of web based interactive environment. *In: Proc. of the Int. Conf. on Implementing Curricular Change in Engineering Education,* 1-10.

[220] Zaïane O.R. (2001). Web usage mining for a better web-based learning environment, In: *Proceedings of Conference on Advanced Technology for Education*, 450-455.

[221] Zaiane O.R. (2002). Building a Recommender Agent for e-Learning Systems, *Proc. of the International Conference on Computer in Education*, 55-59.

[222] Zaiane O.R. (2006). Recommender systems for e-learning: toward non-intrusive web mining. *Data mining in e-learning*, WIT press, 79-96.

[223] Zanardi V., and Capra L. (2008). Social ranking: uncovering relevant content using tag-based recommender systems. *In RecSys '08: Proc. of the 2008 ACM conf. on Recommender systems*. ACM, 51-58.

[224] Zheng Z., Kohavi R. and Mason L. (2001). Real world performance of association rules'. *Sixth ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* 2(2), 86-98.

Sažetak

Sistemi koji generišu personalizovane preporuke (*recommender systems*) koriste različite vrste tehnika kako bi korisniku preporučili sadržaje (filmove, TV program, muziku, knjige, vesti, Veb stranice, kurseve i sl.) koji će mu biti interesantni. Tipično, sistemi za generisanje preporuka porede korisnikov profil sa referentnim karakteristikama predviđajući koje će ocene korisnik dati prilikom pregledanja posmatranog sadržaja. Ukoliko se posmatraju karakteristike sličnih sadržaja primenjuje se *content-based* pristup. Ukoliko sistemi za generisanje preporuka koriste mišljenje grupe sličnih korisnika kako bi pojedincima te grupe omogućili da efikasnije identifikuju interesantan sadržaj primenjuje se tehnika kolaborativnog filtriranja (*collaborative filtering*) kao jedna od najuspešnijih tehnika za generisanje preporuka. Mišljenje korisnika može se odrediti eksplicitno, postavljanjem pitanja ili implicitno na osnovu praćenja aktivnosti korisnika.

Novija istraživanja pokazala su da se unapređenje kvaliteta preporuka može postići uvođenjem tagova koji predstavljaju korisne informacije za poboljšanje algoritama namenjenih generisanju preporuka. Sistemi kolaborativnog tagovanja omogućavaju korisnicima da označe proizvode proizvoljnim rečima (tzv. tagovima) po sopstvenom izboru. Pored toga što tagovi korisnicima pružaju pomoć u organizaciji sadržaja, oni su korisni i u izražavanju mišljenja korisnika. Veliki broj informacija koje korisnici pružaju o sebi kroz aktivnosti tagovanja otvorio je mogućnost primene tagova u generisanju preporuka. Tagovi mogu da se koriste u organizovanju ličnih sadržaja, u izražavanju mišljenja korisnika pa i u davanju ocena.

Sistemi u kojima se tehnike za generisanje preporuka mogu značajno unaprediti uvođenjem kolaborativnog tagovanja jesu sistemi elektronskog učenja. U oblasti elektronskog učenja personalizovane preporuke omogućavaju korisniku pronalaženje relevantnih aktivnosti u procesu učenja, koje odgovaraju njegovom profilu, njegovom talentu i interesovanjima, stilu učenja i nivou znanja, što pospešuje njegovu motivaciju za učenjem i omogućava mu da realizuje svoje aktivnosti učenja na najefikasniji način.

Istraživanje disertacije je usmereno na analizu i definisanje poboljšanih modela za odabir tagova koji otkrivaju sklonosti i osobine korisnika potrebne za generisanje personalizovanih preporuka i mogućnosti primene tako dobijenih modela za

personalizaciju tutorskih sistema. Personalizovani tutorski sistemi korisniku pružaju optimalne putanje kretanja i adekvatne aktivnosti učenja na osnovu njegovih osobina, njegovog stila učenja, znanja koje on poseduje u toj oblasti, kao i prethodnog iskustva korisnika sistema koji imaju slične karakteristike.

Disertacija je organizovana u tri dela. Prvi deo sadrži opširan uvod i pruža pregled tehnika sistema za generisanje preporuka koje služe kao polazna osnova za rezultate dobijene u kasnijim poglavljima.

U poglavlju 1 dat je uvid u motivaciju i probleme proučavane u prikazanom istraživanju, prikazana je struktura i istaknuti su očekivani ciljevi doktorske disertacije.

U poglavlju 2 detaljno su opisani sistemi za generisanje preporuka i tehnike koje oni najčešće koriste kao i model i tehnike kolaborativnog tagovanje koje se koriste pri generisanju preporuka.

Drugi deo, koji sadrži poglavlje 3 i 4, obuhvatio je detaljan prikaz personalizacije procesa elektronskog učenja primenom sistema za generisanje preporuka zasnovanog na tehnikama kolaborativnog tagovanja (*collaborative tagging techniques*).

U poglavlju 3 prikazani su najvažniji zahtevi koje moraju da ispune sistemi za generisanje preporuka u oblasti elektronskog učenja. Potom su detaljno objašnjene klasične tehnike za generisanje preporuka: *collaborative filtering*, *content-based technique*, *association rule mining* kao i tehnike kolaborativnog tagovanja: *FolkRank*, *PLSA*, *collaborative filtering based on collaborative tagging*, *tensor factorization technique for tag recommendation*. Istraživanja su obuhvatila i izazove, nedostatke i mogućnosti proširenja.

U poglavlju 4 je, nakon pregleda literature iz stručne oblasti, prikazan dizajn, interfejs i generalna arhitektura programerskog tutorskog sistema – Protus, a potom je detaljno opisan modul za generisanje preporuka kao i sam proces personalizacije koji obuhvata identifikaciju stilova učenja, koja je neophodna za formiranje klastera, u okviru kojih su personalizovane preporuke generisane na osnovu tagova učenika i eksperata u oblasti programiranja.

Eksperimentalna istraživanja i opsežna diskusija dobijenih rezultata koji ukazuju na validnost sistema prikazani su u trećem delu disertacije, koji sadrži poglavlja 5 i 6.

U poglavlju 5 data je definicija upotrebljenih podataka, opisan je proces klasterovanja, statističke karakteristike procesa tagovanja i eksperimentalni rezultati dobijeni poređenjem selektovanih algoritama, kao i rezultati dobijeni analizom seta tagova koji su generisali eksperti.

Na kraju, poglavlje 6 zaključuje disertaciju, sumirajući glavne rezultate, doprinose i predstavljajući mogućnosti za dalji rad.

Naučni doprinosi disertacije mogu se sagledati u okvirima pravaca istraživanja prikazanim u delovima II i III.

Prvo, u tezi je prikazana adaptacija programerskog tutorskog sistema stilovima učenja na osnovu rezultata dobijenih obradom upitnika koji popunjavaju učenici. Potom su ocenjene statističke karakteristike procesa tagovanja. Proučene su aktivnosti tagovanja učenika i uočeno je da je većina učenika visoko aktivna u procesu tagovanja, jer taguje između 50 i 70 objekata učenja. Kako bismo mogli bolje razumeti karakteristike učenikovih tagova, i navika tagovanja, analizirali smo i entropiju tagova, kao meru specifičnosti, koja ukazuje na opštost pojmova ako je visoka i na njihovu specifičnost ako je niska. Istraživanje je pokazalo da učenici na početku procesa tagovanja koriste opštije pojmove, a vremenom počinju da upotrebljavaju pojmove koji su specifičniji za određene teme. Semantičkom analizom tagova uočeno je da se najčešće koriste personalni (44% tagova) i subjektivni (40% tagova) tagovi.

Najznačajniji deo istraživanja obuhvata odabir tehnika kolaborativnog tagovanja čijom bi se primenom postigli najbolji rezultati u pogledu povećanja motivacije u procesu učenja, znanja i razumevanja gradiva. Generisane preporuke su personalizovane i kvalitetno odabrane tako da su u skladu sa korisnikovim interesovanjima, njegovim stilom učenja, i prethodno stečenim znanjem. U istraživanjima koje je obuhvatila disertacija, prvo je napravljeno poređenje jednostavnih metoda zasnovanih na najpopularnijim tagovima (*most popular tags*), specfičnim pristupima za poboljšanje performansi tih metoda kao i adaptirane tehnike kolaborativnog filtriranja zasnovanog na tagovima. Svi algoritmi pokazali su značajno bolje rezultate u odnosu na najednostavniju preporuku najpopularnijih tagova. Međutim, uvođenje kombinovane tehnike eksperimentalna istraživanja sa različitim dimenzijama faktora u osnovni model preporuke najpopularnijih tagova pokazalo je da se odabirom vrednosti ρ u iznosu 0,6 postižu značajno bolji rezultati od standardne *CF* tehnike. Potom, su analizirane tehnike zasnovane na grafovima, *Adapted PageRank* i *FolkRank*, kao i tehnike zasnovane na tenzorima, *HOSVD* i *RTF*. U poređenju kvaliteta preporuka, *FolkRank* pruža, znatno bolje rezultate od CF i *Adapted PageRank* tehnike jer ima sposobnost da iskoristi informacije koje odgovaraju specifičnim osobinama korisnika i upotrebi integrisane strukture osnovnog hipergrafa. Tehnika *FolkRank* postiže dobre rezultate, jer nije potrebna obuka prilikom postavljanja novog korisnika, novih tagova ili novih objekata u sistem. Međutim, izračunavanja su skupa i teško se postiže skalabilnost, tako da je njegovo optimalno korišćenje moguće u sistemima u kojima preporuke u realnom vremenu nisu neophodan uslov.

Metode koje su zasnovane na tenzorima, takođe koriste ternarni odnos veličina u folksonomiji. Iako faza rekonstrukcije tenzora može biti skupa, ona ne mora da se realizuje preko Interneta, što ovim tehnikama omogućava efikasnu upotrebu u generisanju preporuka u realnom vremenu. Radi poređenja *HOSVD* tehnike i nekoliko tipova *RTF* tehnika korišćena je modifikacija i kombinacija različitog broja parametara, što je u svim okvirima istraživanja ukazalo na izraženiju efikasnost *RTF* metoda čak i sa nižim brojem parametara.

Eksperimentalna istraživanja obuhvatila su i ocenu performansi sistema sa aspekta učenika i mentora. Mentori su bili motivisani mogućnostima koje im sistem pruža u pogledu pripreme i modifikacije materijala za učenje, i razmene informacija sa ostalim mentorima i ekspertima iz oblasti. Međutim, mentori smatraju da je kreiranje baze znanja ipak težak zadatak. Eksperimentalno istraživanje značaja sistema u pogledu efikasnosti, efektivnosti i atraktivnosti za učenika realizovano je na eksperimentalnoj i kontrolnoj grupi učenika. Na osnovu obrade upitnika za subjektivnu procenu kvaliteta sistema koji su popunili samo učenici eksperimentalne grupe, može se zaključiti:

1. Kreiranje tagova za učenika je jednostavan proces.
2. Aktivnosti tagovanja mogu da pomognu učenicima u razumevanju novih struktura i koncepata, kao i da ih podstaknu na kreiranje sopstvenih ideja.

3. Personalizovane preporuke koje pruža Protus sistem omogućavaju učeniku pronalaženje relevantnih aktivnosti u procesu učenja koje odgovaraju njegovom profilu, stilu učenja i nivou znanja, što pospešuje njegovu motivaciju za učenjem i omogućava mu da realizuje svoje aktivnosti učenja na najefikasniji način.
4. Učenici smatraju da je Protus sistem jednostavan za upotrebu.

Dobijeni eksperimentalni rezultati pokazali su da učenici koji koriste Protus sistem postižu bolje rezultate nego učenici koji su za učenje koristili prethodnu verziji sistema, kako u pogledu količine obrađenog materijala tako i u pogledu utrošenog vremena.

Kratka biografija

Aleksandra Klašnja-Milićević je rođena 6. juna 1977. godine, u Bačkoj Topoli. Na Univerzitetu u Novom Sadu 1996. godine upisala je Fakultet tehničkih nauka, odsek Elektrotehnika i računarstvo, smer Elektronika i telekomunikacije, gde je i diplomirala 2002. godine sa prosečnom ocenom 8,76. odbranivši diplomski rad pod nazivom „Sinteza govora na osnovu parametara spektralne analize“ sa ocenom 10. Magistarske studije informatike upisala je 2003. godine na Prirodno-matematičkom fakultetu Univerziteta u Novom Sadu. Sve ispite predviđene planom i programom fakulteta položila je sa prosečnom ocenom 10. Iste godine zaposlila se na Visokoj poslovnoj školi strukovnih studija u Novom Sadu, na smeru Informatika. Magistrirala je 31. maja 2007. godine, odbranivši magistarsku tezu pod naslovom „Dizajn i implementacija elektronske prodavnice knjiga“. U zvanje predavača za predmete “Projektovanje Veb aplikacija” i “Primena Internet tehnologija” izabrana je na Visokoj poslovnoj školi strukovnih studija u Novom Sadu 2008. godine. Aktivno radi na naučnom projektu koje finansira Ministarstvo za obrazovnje, nauku i tehnološki razvoj Republike Srbije. Koautor je je jednog priručnika za primenu informacionih tehnologija. Autor je 25 radova iz oblasti internet tehnologija, sistema za generisanje preporuka, obrazovnih sistema i elektronskog poslovanja od kojih je šest objavljeno u časopisima sa SCI liste.

Short Biography

Aleksandra Klašnja-Milićević was born on June 6, 1977 in Bačka Topola, Serbia. She attended Faculty of Technical Sciences at the University of Novi Sad, Department for Electrical Engineering and Computer Science, receiving a B.S. in 2002, defending her thesis entitled "Speech synthesis based on the parameters of spectral analysis" with a score of 10. In 2003 she enrolled master studies of Computer Science at the Faculty of Science, University of Novi Sad. All the exams from the curriculum of the faculty passed with the average grade 10. In the same year, she was employed in the Higher School of Professional Business Studies in Novi Sad, at Department for Computer Science. She received her master degree in 2007, defending her thesis titled „Design and Implementation of an Online Bookstore“. Her research interests include internet technologies, recommender systems, e-learning and personalization, and electronic commerce. She has taught Internet technologies and Web applications design at the Higher School of Professional Business Studies in Novi Sad. She actively participates in scientific project financed by the Ministry of Education, Science and Technological Development of the Republic of Serbia. She coauthored one university textbook. She has published 25 scientific papers in proceedings of international conferences and journals in the field of internet technologies, recommender systems and e-commerce, six of which were published in journals indexed in the Science Citation Index (SCI) database.

UNIVERZITET U NOVOM SADU
PRIRODNO – MATEMATIČKI FAKULTET
KLJUČNA DOKUMENTACIJSKA INFORMACIJA

Redni broj:

PBR

Identifikacioni broj:

IBR

Tip dokumentacije: Monografska dokumentacija

TD

Tip zapisa: Tekstualni štampani materijal

TZ

Vrsta rada: Doktorska disertacija

VR

Autor: Aleksandra Klašnja-Milićević

AU

Mentor: prof. dr Mirjana Ivanović

MN

Naslov rada: Personalized Recommendation Based on Collaborative Tagging Techniques for an e-Learning System

Personalizacija procesa elektronskog učenja primenom sistema za generisanje preporuka zasnovanog na tehnikama kolaborativnog tagovanja

MR

Jezik publikacije: Engleski

JP

Jezik izvoda: Srpski/engleski

JI

Zemlja publikovanja: Srbija

ZP

Uže geografsko područje: Vojvodina, Novi Sad

UGP

Godina: 2012.

GO

Izdavač: Autorski reprint

IZ

Mesto i adresa: Novi Sad, Trg Dositeja Obradovića 4

MA

Fizički opis rada: (6/148/224/26/57/0/0)

(broj poglavlja/strana/lit. citata/tabela/slika/grafika/priloga)

FO

Naučna oblast: Računarske nauke

NO

Naučna disciplina: Sistemi za generisanje preporuka

ND

Ključne reči: Sistemi za generisanje preporuka, personalizacija, elektronsko učenje, stilovi učenja, kolaborativno tagovanje

PO

UDK:

Čuva se:

ČU

Važna napomena: nema

VN

Izvod: Predmet istraživanja disertacije obuhvata personalizaciju tutorskih sistema za elektronsko učenje primenom tehnika kolaborativnog tagovanja (collaborative tagging techniques) integrisanih u sisteme za generisanje preporuka (recommender systems). Tagovi, kao oblik meta podataka, predstavljaju proizvoljne ključne reči ili fraze koje korisnik može da upotrebi za označavanje različitih sadržaja. Pored toga što tagovi korisnicima pružaju pomoć u organizaciji sadržaja, oni su korisni i u izražavanju mišljenja korisnika. Veliki broj informacija koje korisnici pružaju o sebi kroz aktivnosti tagovanja otvorio je mogućnost primene tagova u generisanju preporuka.

Istraživanje disertacije je usmereno na analizu i definisanje poboljšanih modela za odabir tagova koji otkrivaju sklonosti i osobine korisnika potrebne za generisanje personalizovanih preporuka. Razmatrane su i mogućnosti primene tako dobijenih modela za personalizaciju tutorskih sistema. Personalizovani tutorski sistemi korisniku pružaju optimalne putanje kretanja i adekvatne aktivnosti učenja na osnovu njegovih osobina, njegovog stila učenja, znanja koje on poseduje u toj oblasti, kao i prethodnog iskustva korisnika sistema koji imaju slične karakteristike. Modeli definisani u disertaciji u praksi su evaluirani na tutorskom sistemu za učenje programskog jezika Java.

IZ

Datum prihvatanja teme od strane NN veća: 25. maj 2011. god.

DP

Datum odbrane:

DO

Članovi komisije:

KO

Predsednik: dr Zoran Budimac, redovni profesor, Prirodno-matematički fakultet, Novi Sad

Mentor: dr Mirjana Ivanović, redovni profesor, Prirodno-matematički fakultet, Novi Sad

Član: dr Dragan Janković, redovni profesor, Elektronski fakultet, Univerzitet u Nišu

Član: dr Elvira Popescu, docent, Univerzitet u Krajovi, Rumunija

UNIVERSITY OF NOVI SAD
FACULTY OF SCIENCE
KEY WORDS DOCUMENTATION

Accession number:

ANO

Identification number:

INO

Document type: Monograph type

DT

Type of record: Printed text

TR

Contents Code: Doctoral dissertation

CC

Author: Aleksandra Klašnja-Milićević

AU

Mentor: prof. dr Mirjana Ivanović

MN

Title: Personalized Recommendation Based on Collaborative Tagging Techniques for an e-Learning System

XI

Language of text: English

LT

Language of abstract: Serbian/English

LA

Country of publication: Serbia

CP

Locality of publication: Vojvodina, Novi Sad

LP

Publication year: 2012.

PY

Publisher: Author's reprint

PU

Publ. Place: Novi Sad, Trg Dositeja Obradovića 4

PP

Physical description: (6/148/224/26/57/0/0)

(no. of chapters/pages/bib. refs/tables/figures/graphs/appendices)

PD

Scientific field: Computer Science

SF

Key words: *recommender systems, personalization, e-learning, learning style, colaborative taging*

UC:

Holding data:

Hd Note:

Abstract: The research topic involves personalization of an e-learning system based on collaborative tagging techniques integrated in a recommender system. Collaborative tagging systems allow users to upload their resources, and to label them with arbitrary words, so-called tags. The systems can be distinguished according to what kind of resources are supported. Besides helping user to organize his or her personal collections, a tag also can be regarded as a user's personal opinion expression. The increasing number of users providing information about themselves through social tagging activities caused the emergence of tag-based profiling approaches, which assume that users expose their preferences for certain contents through tag assignments. Thus, the tagging information can be used to make recommendations.

Dissertation research aims to analyze and define an enhanced model to select tags that reveal the preferences and characteristics of users required to generate personalized recommendations. Options on the use of models for personalized tutoring system were also considered. Personalized learning occurs when e-learning systems make deliberate efforts to design educational experiences that fit the needs, goals, talents, learning styles, interests of their learners and learners with similar characteristics. In practice, models defined in the dissertation were evaluated on tutoring system for teaching Java programming language.

AB

Accepted by the Scientific Board on: May 25, 2011

Defended:

Thesis defend board:

President: dr Zoran Budimac, full professor, Faculty of Science, Novi Sad

Mentor: dr Mirjana Ivanović, full professor, Faculty of Science, Novi Sad

Member: dr Dragan Janković, full professor, Faculty of Electronic Engineering, Niš

Member: dr Elvira Popescu, assistant professor, Faculty of Automation, Computers and Electronics, Craiova, Romania