Toward the Next Generation of Recommender Systems: Applications and Research Challenges

Alexander Felfernig, Michael Jeran, Gerald Ninaus,
Florian Reinfrank, and Stefan Reiterer

Institute for Software Technology
Graz University of Technology
Inffeldgasse 16b, A-8010 Graz, Austria
{firstname.lastname}@ist.tugraz.at

Abstract. Recommender systems are assisting users in the process of identifying items that fulfill their wishes and needs. These systems are successfully applied in different e-commerce settings, for example, to the recommendation of news, movies, music, books, and digital cameras. The major goal of this book chapter is to discuss new and upcoming applications of recommendation technologies and to provide an outlook on major characteristics of future technological developments. Based on a literature analysis, we discuss new and upcoming applications in domains such as software engineering, data & knowledge engineering, configurable items, and persuasive technologies. Thereafter we sketch major properties of the next generation of recommendation technologies.

1 Introduction

Recommender systems support users in the identification of items that fulfill their wishes and needs. As a research discipline, *recommender systems* has been established in the early 1990's (see, e.g., [33]) and since then has grown enormously in terms of algorithmic developments as well as in terms of deployed applications. A recommender system can be defined as a system that guides users in a personalized way to interesting or useful objects in a large space of possible objects or produces such objects as output [8, 46]. Practical experiences from the successful deployment of recommendation technologies in e-commerce contexts (e.g., *amazon.com* [51] and *netflix.com* [74]) contributed to the development of recommenders in new application domains. Especially such new and upcoming application domains are within the major focus of this chapter.

On an algorithmic level, there exist four basic recommendation approaches. First, *content-based filtering* [58] is an information filtering approach where features of items a user liked in the past are exploited for the determination of new recommendations. Content-based filtering recommendation is applied, for example, by *amazon.com* for the recommendation of items which are similar to those that have already been purchased by the user. If a user has already purchased

a book related to the *Linux* operating system, new (and similar) books will be recommended to her/him in the future.

Second, *collaborative filtering* is applied to the recommendation of items such as music and movies [45, 47, 51]. It is based on the concept of analyzing the preferences of users with a similar item rating behavior. As such, it is a basic implementation of word-of-mouth promotion with the idea that a purchase decision is influenced by the opinions of friends and relatives. For example, if user *Joe* has purchased movies similar to the ones that have been purchased by user *Mary* then *amazon.com* would recommend items to *Joe* which have been purchased by *Mary* but not by *Joe* up to now. The major differences compared to content-based filtering are (a) no need of additional item information (in terms of categories or keywords describing the items) and (b) the need of information about the rating behavior of other users.

Third, high-involvement items such as cars, apartments, and financial services are typically recommended on the basis of *knowledge-based recommendation technologies* [8, 22]. These technologies are based on the idea of exploiting explicitly defined recommendation knowledge (defined in terms of deep recommendation knowledge, e.g., as rules and constraints) for the determination of new recommendations. Rating-based recommendation approaches such as collaborative filtering and content-based filtering are not applicable in this context since items are not purchased very frequently. A consequence of this is that no up-to-date rating data of items is available. Since knowledge-based recommendation approaches rely on explicit knowledge representations the so-called knowledge acquisition bottleneck becomes a major challenge when developing and maintaining such systems [22].

Finally, *group recommendation* [54] is applied in situations where there is a need of recommendations dedicated to a group of users, for example, movies to be watched by a group of friends or software requirements to be implemented by a development team. The major difference compared to the afore mentioned recommendation approaches lies in the criteria used for determining recommendations: while in the case of content-based filtering, collaborative filtering, and knowledge-based recommendation the major goal is to identify recommendations that perfectly fit the preferences of the current user, group recommendation approaches have to find ways to satisfy the preferences of a user group.

The major focus of this book chapter is to give an overview of new and upcoming applications of recommendation technologies and to provide insights into major requirements regarding the development of the next generation of recommender systems. Our work is based on an analysis of research published in workshops, conferences, and journals summarized in Table 1. In this context we do not attempt to provide an in-depth analysis of state-of-the-art recommendation algorithms [2, 41] and traditional applications of recommendation technologies [68] but focus on the aspect of new and upcoming applications [16, 46]. An overview of these applications is given in Table 2.

The remainder of this chapter is organized as follows. In Sections 2–6 we provide an overview of new types of applications of recommendation technolo-

gies and give working examples. In Section 7 we sketch the upcoming generation of recommender systems as *Personal Assistants* which significantly improve the overall quality of recommendations in terms of better taking into account preferences of users – in this context we discuss major issues for future research.

Table 1. Overview of journals and conferences (including workshops) used as the basis for the literature analysis (papers on recommender system applications 2005–2012).

Journals and Conferences
International Journal of Electronic Commerce (IJEC)
Journal of User Modeling and User-Adapted Interaction (UMUAI)
AI Magazine
IEEE Intelligent Systems
Communications of the ACM
Expert Systems with Applications
ACM Recommender Systems (RecSys)
User Modeling, Adaptation, and Personalization (UMAP)
ACM Symposium on Applied Computing (ACM SAC)
ACM International Conference on Intelligent User Interfaces (IUI)

2 Recommender Systems in Software Engineering

Recommender systems can support stakeholders in software projects by efficiently tackling the information overload immanent in software projects [66]. They can provide stakeholder support throughout the whole software development process – examples are the *recommendation of methodological knowledge* [9, 59], the *recommendation of requirements* [28, 57], and the *recommendation of code* [14, 55].

Recommendation of Methodological Knowledge. Appropriate method selection is crucial for successful software projects, for example, the waterfall process is only applicable for risk-free types of projects but not applicable for high risk projects. Method recommendation is already applied in the context of different types of development activities such as the domain-dependent recommendation of algorithmic problem solving approaches [9] and the recommendation of appropriate effort estimation methods [59]. These approaches are based on the knowledge-based recommendation paradigm [8, 22] since recommendations do not rely on the taste of users but on well-defined rules defining the criteria for method selection.

Recommendation of Code. Due to frequently changing development teams, the intelligent support of discovering, locating, and understanding code is crucial for efficient software development [55]. Code recommendation can be applied, for example, in the context of (collaborative) call completion (which API methods from a large set of options are relevant in a certain context?) [55], recommending relevant example code fragments (which sequence of methods is needed in the

current context?) [38], tailoring the displayed software artifacts to the current development context [44], and classification-based defect prediction [56]. For an in-depth discussion of different types of code recommendation approaches and strategies we refer the reader to [35, 66].

Table 2. Overview of identified recommender applications not in the mainstream of e-commerce applications (Knowledge-based Recommendation: KBR, Collaborative Filtering: CF, Content-based Recommendation: CBR, Machine Learning:ML, Group Recommendation: GR, Probability-based Recommendation: PR, Data Mining: DM).

Domain	Recommended Items	Recommendation Approach	
Software Engineering	effort estimation methods	KBR	Peischl et al. [59]
	problem solving approaches	KBR	Burke and Ramezani [9]
	API call completions	CF	McCarey et al. [55]
	example code fragments	CBR	Holmes et al. [38]
	contextualized artifacts	KBR	Kersten and Murphy [44]
	software defects	ML	Misirli et al. [56]
	requirements prioritizations	GR	Felfernig et al. [28]
	software requirements	CF	Mobasher & Cleland-Huang [57]
Data & Knowledge Engineering	related constraints	KBR	Felfernig et al. [25, 27]
	explanations (hitting sets)	KBR	Felfernig et al. [26]
	constraint rewritings	KBR	Felfernig et al. [24]
	database queries	CF	Chatzopoulou et al. [10]
		CBF	Fakhraee and Fotouhi[17]
Knowledge-based Configuration	relevant features	CF	Felfernig and Burke [21]
	requirements repairs	CBF	Felfernig et al. [23]
Persuasive Technologies	game task complexities	CF	Berkovsky et al. [5]
	development practices	KBR	Pribik and Felfernig [61]
Smart Homes	equipment configurations	KBR	Leitner et al. [49]
	smart home control actions	CF	LeMay et al. [50]
People	criminals	PR	Tayebi et al. [70]
	reviewers	CF	Kapoor et al. [43]
	physicians	CF	Hoens et al. [36]
Points of Interest	tourism services	CF	Huang et al. [39]
	passenger hotspots	PR	Yuan et al. [79]
Help Services	schools	CBR	Wilson et al. [76]
	financial services	KBR	Fano and Kurth [19]
	lifestyles	KBR	Hammer et al. [34]
	receipes	CBR	Pinxteren et al. [60]
	health information	CBR	Wiesner and Pfeifer [75]
Innovation Management	innovation teams	KBR	Brocco and Groh [7]
	business plans	KBR	Jannach and B.-Joergensen [40]
	ideas	DM	Thorleuchter et al. [73]

Recommendation of Requirements. Requirements engineering is one of the most critical phases of a software development process and poorly implemented requirements engineering is one of the major reasons for project failure [37]. Core requirements engineering activities are elicitation & definition, quality assurance,

negotiation, and release planning [69]. All of these activities can be supported by recommendation technologies, for example, the (collaborative) recommendation of requirements to stakeholders working on similar requirements [57] and the group-based recommendation of requirements prioritizations [28].

Example: Content-based Recommendation of Similar Requirements. In the following, we will exemplify the application of *content-based filtering* [58] in the context of *requirements engineering*. A recommender can support stakeholders, for example, by recommending requirements that have been defined in already completed software projects (requirements reuse) or have been defined by other stakeholders of the same project (redundancy and dependency detection). Table 3 provides an overview of requirements defined in a software project. Each requirement req_i is characterized by a *category*, the number of estimated *person days* to implement the requirement, and a textual *description*.

Table 3. Example of a content-based filtering recommendation problem: recommendation of similar requirements based on *category* and/or *keyword* information.

requirement	category	person days	description
req_1	database	170	store component configuration in DB
req_2	user interface	65	user interface with online help available
req_3	database	280	separate tier for DB independence
req_4	user interface	40	user interface with corporate identity

If we assume that the stakeholder currently interacting with the requirements engineering environment has already investigated the requirement req_1 (assigned to the category *database*), a content-based recommender system would recommend the requirement req_3 (if this one has not been investigated by the stakeholder up to now). If no explicitly defined categories are available, the textual description of each requirement can be exploited for the extraction of keywords which serve for the characterization of the requirement. Extracted keywords can then be used for the determination of similar requirements [62]. A basic metric for determining the similarity between two requirements is given in Formula 1. For example, $\text{sim}(\text{req}_1, \text{req}_3) = 0.17$, if we assume $keywords(\text{req}_1) = \{store, component, configuration, DB\}$ and $keywords(\text{req}_3) = \{tier, DB, independence\}$.

$$sim(req_1, req_2) = \frac{|keywords(req_1) \cap keywords(req_2)|}{|keywords(req_1) \cup keywords(req_2)|} \tag{1}$$

Example: Group-based Recommendation of Requirements Prioritizations. Group recommenders include heuristics [54] that can help to find solution alternatives which will be accepted by the members of a group (with a high probability). Requirements prioritization is the task of deciding which of a given set of requirements should be implemented within the scope of the next software release – as such, this task has a clear need of group decision support: a stakeholder group has to decide which requirements should be taken into account.

Different heuristics for coming up with a group recommendation are possible, for example, the *majority heuristic* proposes a recommendation that takes the majority of requirements-individual votes as group recommendation (see Table 4). In contrast, the *least misery* heuristic recommends the minimum of the requirements-individual ratings to avoid misery for individual group members. For a detailed discussion of different possible group recommendation heuristics we refer the reader to [54].

Table 4. Example of a group recommendation problem: recommendation of requirements prioritizations to a group of stakeholders (used heuristics = majority voting).

requirement	stakeholder_1	stakeholder_2	stakeholder_3	stakeholder_4	*recommendation*
req_1	1	1	1	2	1
req_2	5	4	5	5	5
req_3	3	3	2	3	3
req_4	5	5	4	5	5

3 Recommender Systems in Data & Knowledge Engineering

Similar to conventional software development, knowledge-based systems development suffers from continuously changing organizational environments and personal. In this context, recommender systems can help database & knowledge engineers to better cope with the size and complexity of knowledge structures [10, 23]. For example, recommender systems can support an improved understanding of the knowledge base by actively suggesting those parts of the knowledge base which are candidates for increased testing and debugging [23, 25]. Furthermore, recommender systems can propose repair and refactoring actions for faulty and ill-formed knowledge bases [24, 27]. In the context of information search, recommender systems can improve the accessibility of databases (knowledge bases) by recommending queries [10].

Knowledge Base Understanding. Understanding the basic elements and the organizational structure of a knowledge base is a major precondition for efficient knowledge base development and maintenance. In this context, recommender systems can be applied for supporting knowledge engineers, for example, by (collaboratively) recommending constraints to be investigated when analyzing a knowledge base (recommendation of navigation paths through a knowledge base) or by recommending constraints which are related to each other, i.e., are referring to common variables [25].

Knowledge Base Testing and Debugging. Knowledge bases are frequently adapted and extended due to the fact that changes in the application domain have to be integrated into the corresponding knowledge base. Integrating such changes into a knowledge base in a consistent fashion is time-critical [22]. Recommender systems can be applied for improving the efficiency of knowledge base

testing and debugging by recommending minimal sets of changes to knowledge bases which allow to restore consistency [23, 26]. Popular approaches to the identification of such *hitting sets* are model-based diagnosis introduced by Reiter [65] and different variants thereof [26].

Knowledge Base Refactoring. Understandability and maintainability of knowledge bases are important quality characteristics which have to be taken into account within the scope of knowledge base development and maintenance processes [4]. Low quality knowledge structures can lead to enormous maintenance costs and thus have to be avoided by all available means. In this context, recommender systems can be applied for recommending relevant refactorings, i.e., semantics-preserving structural changes of the knowledge base. Such refactorings can be represented by simple constraint rewritings [24] or by simplifications in terms of recommended removals of redundant constraints, i.e., constraints which do not have an impact on the semantics of the knowledge base [27].

Recommender Systems in Databases. Chatzopoulou et al. [10] introduce a recommender application that supports users when interactively exploring relational databases (*complex SQL queries*). This application is based on collaborative filtering where information about the navigation behavior of the current user is exploited for the recommendation of relevant queries. These queries are generated on the basis of the similarity of the current user's navigation behavior and the navigation behavior of nearest neighbors (other users with a similar navigation behavior who already searched the database). Fakhraee et al. [17] focus on recommending database queries which are derived from those database attributes that contain the keywords part of the initial user query. In contrast to Chatzopoulou et al. [10], Fakhraee et al. [17] do not support the recommendation of complex SQL queries but focus on basic lists of keywords (*keyword-based database queries*).

Example: Collaborative Recommendation of Relevant Constraints. When knowledge engineers try to *understand a given set of constraints* part of a knowledge base, recommender systems can provide support in terms of showing related constraints, for example, constraints that have been analyzed by knowledge engineers in a similar navigation context. In Table 5, the constraints $\{\text{constraint}_{1..4}\}$ have already partially been investigated by the knowledge engineers $\{\text{knowledge engineer}_{1..3}\}$. For example, knowledge engineer_1 has already investigated the constraints constraint_1 and constraint_3. Collaborative filtering (CF) can exploit the ratings (the rating = 1 if a knowledge engineer has already investigated a constraint and it is 0 if the constraint has not been investigated up to now) for identifying additional constraints relevant for the knowledge engineer.

User-based CF [45] identifies the k-nearest neighbors (knowledge engineers with a similar knowledge navigation behavior) and determines a prediction for the rating of a constraint the knowledge engineer had not to deal with up to now. This prediction can be determined, for example, on the basis of the majority of the k-nearest neighbors. In the example of Table 5 $\text{knowledge engineer}_2$ is the nearest neighbor (if we set k=1) since $\text{knowledge engineer}_2$ has analyzed (or changed) all the constraints investigated by $\text{knowledge engineer}_1$. At the same

Table 5. Example of a collaborative recommendation problem. The entry ij with value 1 (0) denotes that fact that knowledge engineer$_i$ has (not) inspected constraint$_j$.

	constraint$_1$	constraint$_2$	constraint$_3$	constraint$_4$
knowledge engineer$_1$	1	0	1	?
knowledge engineer$_2$	1	0	1	1
knowledge engineer$_3$	1	1	0	1

time, knowledge engineer$_1$ did not analyze (change) the constraint constraint$_4$. In our example, CF would recommend the constraint$_4$ to knowledge engineer$_1$.

4 Recommender Systems for Configurable Items

Configuration can be defined as a basic form of design activity where a target product is composed from a set of predefined components in such a way that it is consistent with a set of predefined constraints [67]. Similar to knowledge-based recommender systems [8, 22], configuration systems support users in specifying their requirements and give feedback in terms of solutions and corresponding explanations [18]. The major difference between configuration and recommender systems is the way in which these systems represent the product (configuration) knowledge: configurators are exploiting a configuration knowledge base whereas (knowledge-based) recommender systems are relying on a set of enumerated solution alternatives (items). Configuration knowledge bases are especially useful in scenarios with a large number of solution alternatives which would make an explicit representation extremely inefficient [18].

In many cases, the amount and complexity of options presented by a configurator outstrips the capability of a user to identify an appropriate solution. Recommendation technologies can be applied in various ways to improve the usability of configuration systems, for example, filtering out product features which are relevant for the user [18, 31], proposing concrete feature values and thus helping the user in situations where knowledge about certain product properties is not available [18], and by determining plausible repairs for inconsistent user requirements [23].

Selecting Relevant Features. In many cases users are not interested in specifying all the features offered by a configurator interface, for example, some users may be interested in the GPS functionality of digital cameras whereas this functionality is completely uninteresting for other users. In this context, recommender systems can help to determine features of relevance for the user, i.e., features the user is interested to specify. Such a feature recommendation can be implemented, for example, on the basis of collaborative filtering [21].

Determining Relevant Feature Values. Users try to avoid to specify features they are not interested in or about which they do not have the needed technical knowledge [18]. In such a context, recommender systems can automatically recommend feature values and thus reduce the burden of interaction for the user. Feature value recommendation can be implemented, for example, on the

basis of collaborative filtering [18, 45]. Note that feature value recommendation can trigger biasing effects and – as a consequence – could also be exploited for manipulating users to select services which are not necessarily needed [52].

Determining Plausible Repairs for Inconsistent Requirements. In situations where no solution can be found for a given set of customer requirements, configuration systems determine a set of repair alternatives which guarantee the recovery of consistency. Typically many different repair actions are determined by the configurator which makes it nearly infeasible for the user to find one which exactly fits his/her wishes and needs. In such a situation, knowledge-based and collaborative recommendation technologies can be exploited for personalizing the user requirements repair search process [23].

Example: Collaborative Recommendation of Features. Table 6 represents an interaction log which indicates in which session which features have been selected by the user (in which order) – in session_1, feature_1 was selected first, then feature_3 and feature_2, and finally feature_4. One approach to determine relevant features for (the current) user in session_5 is to apply collaborative filtering. Assuming that the user in session_5 has specified the $\text{features}_{1,2}$, the most similar session would be session_2 and feature_3 would be recommended to the user since it had been selected by the nearest neighbor (user in session_2). For a discussion of further feature selection approaches we refer the reader to [18].

Table 6. Example of collaboratively recommending relevant features. A table entry x denotes the order in which a user specified values for the given features.

	feature_1	feature_2	feature_3	feature_4
session_1	1	3	2	4
session_2	1	2	3	4
session_3	1	1	4	3
session_4	1	3	2	4
session_5	1	2	?	?

5 Recommender Systems for Persuasive Technologies

Persuasive technologies [29] aim to trigger changes in a user's attitudes and behavior on the basis of the concepts of human computer interaction. The impact of persuasive technologies can be significantly increased by additionally integrating recommendation technologies into the design of persuasive systems. Such an approach moves persuasive technologies forward from a one-size-fits all approach to a personalized environment where user-specific circumstances are taken into account when generating persuasive messages [6]. Examples of the application of recommendation technologies in the context of persuasive systems are the enforcement of physical activity while playing computer games [5] and encouraging software developers to improve the quality of their software components [61].

Persuasive Games. Games focusing on the motivation of physical activities include additional reward mechanisms to encourage players to perform real physical activities. Berkovsky et al. [5] show the successful application of collaborative filtering recommendation technologies [45] for estimating the personal difficulty of playing. This recommendation (estimation) is exploited to adapt the difficulty level of the current game session since the perceived degree of difficulty is correlated with the preparedness of a user to perform physical activities.

Persuasive Software Development Environments. Software development teams are often under the gun of developing software components under high time pressure which often has a direct impact on the corresponding software quality. However, in the long term software quality is strongly correlated with the degree of understandability and maintainability. In this context, software quality improvements can be achieved by recommendation technologies, for example, knowledge-based recommenders can be applied to inform programmers about critical code segments and also recommend actions to be performed in order to increase the software quality. Pribik et al. [61] introduce such an environment which has been implemented as an Eclipse plugin (*www.eclipse.org*).

Example: Collaborative Estimation of Personal Game Level Difficulties. Table 7 depicts a simple example of the application of collaborative filtering to the determination of personal difficulty levels. Let us assume that the current user in session_5 has already completed the tasks 1..3; by determining the nearest neighbor of session_5 we can infer a probable duration of task_4: session_1 is the nearest neighbor of session_5 and the user of session_1 needed 7 time units to complete task_4. This knowledge about the probable time efforts needed to complete a task can be exploited to automatically adapt the complexity level of the game (with the goal to increase the level of physical activity [5]).

Table 7. Collaborative filtering based determination of personal game difficulty levels. A table entry x denotes the time a user needed to complete a certain game task.

	task_1	task_2	task_3	task_4
session_1	4	6	5	7
session_2	1	2	2	2
session_3	4	5	5	6
session_4	1	1	1	2
session_5	4	6	4	?

6 Further Applications

Up to now we discussed a couple of new and innovative application domains for recommendation technologies. Admittedly, this enumeration is incomplete since it does not reflect wide-spread e-commerce applications – for a corresponding overview the interested reader is referred to [51, 68]. In this section we discuss

further new and upcoming application domains for recommendation technologies that have been identified within the scope of our literature analysis.

Recommender Systems for Smart Homes. Smart homes are exploiting information technologies to improve the quality of life inside the home. Leitner et al. [49] show the application of knowledge-based recommendation technologies in the context of ambient assisted living (AAL) scenarios where on the one hand recommenders are applied to the design a smart home, on the other hand to control the smart home equipment. During the design phase of a smart home, the user is guided through a preference construction process with the final outcome of a recommendation of the needed technical equipment for the envisioned smart home. For already deployed smart home installations recommendation technologies support the people living in the smart home by recommending certain activities such as activating the air conditioning or informing other relatives about dangerous situations (e.g., for some relevant time period the status of the elderly people living in the house is unclear). A further application of recommendation technologies within the context of AAL is reported by LeMay et al. [50] who introduce an application of collaborative recommendation technologies for supporting the control of complex smart home installations.

People Recommender. Social networks such as *facebook.com* or *linkedin.com* are increasingly popular communication platforms. These platforms also include different types of recommender applications that support users in retrieving interesting music, travel destinations, and connecting to other people. Recommendations determined by these platforms are often exploiting the information contained in the underlying social network [32]. A new and upcoming application which exploits the information of social networks is the identification of crime suspects. In this context, social networks are representing the relationships between criminals. The *CrimeWalker* system introduced by Tayebi et al. [70] is based on a random-walk based method which recommends (provides) a list of the top-k potential suspects of a crime. Similar to *CrimeWalker*, *TechLens* is a recommender system which focuses on the recommendation of persons who could act – for example – as reviewers within the scope of a scientific conference organization. In contrast to *CrimeWalker*, the current version of *TechLens* does not exploit information from social networks – it is based on a collaborative filtering recommendation approach. Yuan et al. [79] present their approach to the recommendation of passenger hotspots, i.e., the recommender system is responsible for improving the prediction quality of hotspots and with this decreases the idle time of taxi drivers. Finally, Hoens et al. [36] present their approach to the recommendation of physicians – the major goal behind this application is to provide mechanisms which improve the quality of physician selection which otherwise is based on simple heuristics such as the opinion of friends or the information found on websites.

RFID based Recommenders. Personalized services become ubiquitous, for example, in tourism many destinations such as museums and art galleries are providing personalized access to help customers to better cope with the large amount of available services. An alternative to force users to explicitly declare

their preferences before receiving a recommendation of potentially interesting services is to observe a user's navigation behavior and – on the basis of this information – to infer plausible recommendations. Such a recommendation approach is in the need of location and object information in order to be able to store the user navigation behavior. A collaborative filtering based handheld guide for art museums is introduced in [39] where Radiofrequency Identification (RFID) serves as a basis for preference data acquisition. RFID is a non-contact tracking system which exploits radio-frequency electromagnetic fields to transfer tag data for the purposes of object identification and object tracking [72].

Help Agents. Howto's are an important mechanism to provide guidance for users who are non-experts in a certain problem domain – such a support can be implemented, for example, on the basis of recommendation technologies [71]. One example for such a type of *help agent* is *SmartChoice* which is a (content-based) recommender system that supports representatives of low-income families in the choice of a public school for their children. Such a recommendation support is crucial since in many cases (a) parents do not dispose of detailed knowledge about the advantages and disadvantages of the different school types and (b) a false decision can have a very negative impact on the future life of the children. Another example of a help agent is *Personal Choice Point* [19] which is a financial service recommendation environment focusing on the visualization of the impact of different financial decisions on a user's life. Hammer et al. [34] present *MED-StyleR* which is a lifestyle recommender dedicated to the support of diabetes patients with the goal of improving care provision, enhancing the quality of a patient's life, and also to lower costs of public health institutions and patients. Another lifestyle related recommender application is presented by Pinxteren et al. [60] which focuses on determining health-supporting recipes that also fit the lifestyle of the user. In the same line, Wiesner and Pfeifer [75] introduce a content-based recommender application dedicated to the identification of relevant health information for a specific user.

Recommender Systems in Open Innovation. Integrating consumer knowledge into a company's innovation processes (also denoted as Open Innovation [11]) is in many cases crucial for efficient new product and service development. Innovation process quality has a huge impact on the ability of a company to achieve sustainable growth. Innovations are very often triggered by consumers who are becoming active contributors in the process of developing new products. Platforms such as *sourceforge.net* or *ideastorm.com* confirm this trend of progressive customer integration. These platforms exploit community knowledge and preferences to come up with new requirements, ideas, and products. In this context, the size and complexity of the generated knowledge (informal descriptions of requirements and ideas as well as knowledge bases describing new products on a formal level) outstrips the capability of community members to retain a clear view. Recommender systems can provide help in terms of finding other users with similar interests and ideas (*team recommendation* [7]) and to (semi-) automatically filter out the most promising ideas (*idea mining* [40, 73]).

7 Issues for Future Research

Recommendation technologies have been successfully applied for almost two decades primarily with the goal of increasing the revenue of different types of online services. In most of the existing systems the primary focus is to help to achieve business goals, rarely the viewpoint of the customer is taken into account in the first place [53]. For example, *amazon.com* recommenders inform users about new books of interest for them; a more customer-centered recommendation approach would also take into account (if available) information about books that have been bought by friends or relatives and thus are potentially available for the customer [53]. As a consequence, we are in the need of new recommendation technologies that allow more customer-centered recommendation approaches. In this context, the following research challenges have to tackled.

Focusing on the User Perspective. There are many other scenarios quite similar to the above mentioned *amazon.com* one where the recommender system is clearly focused on increasing business revenues. For example, consumer packaged goods (CPG) are already offered on the basis of recommender systems [15], however, these systems are domain-specific, i.e., do not take into account information regarding goods and services offered by the grocer nearby. Digital camera recommenders recommend the newest technology but in most cases do not take into account the current portfolio of the user, for example, if a user has a complete lens assortment of camera provider X it does not make sense to recommend a new camera of provider Y in the first place. An approach which is in the line of the idea of a stronger focus on the quality of user support is the RADAR personal assistant introduced by Faulring et al. [20] that supports multi-task coordination of personal emails.

Sharing Recommendation Knowledge. Besides commercial interests, one of the major reasons for the low level of customer orientation of todays recommender solutions is the lack of the needed recommendation knowledge. In order to recommend books already read by friends the recommender would need the information of the social network of the customer. The global availability of CPG goods information seems to be theoretically possible but is definitely in the need of a corresponding cloud and mobile computing infrastructure. More customer-centered recommender systems will follow the paradigm of personal assistants which does not focus on specific recommendation services but rather provides an integrated and multi-domain recommendation service [12, 53]. Following the idea of *ambient intelligence* [64], such systems will be based on global object information [63, 72] and support users in different application contexts in a cross-domain fashion.

Context Awareness. New recommendation technologies will intensively exploit the infrastructure of mobile services to determine and take into account the context of the current user [2]. Information such as the users shorttime and longterm preferences, geographical position, movement data, calendar information, information from social networks can be exploited for detecting the current context of the person and exploit this information for coming up with intelligent recommendations [3].

Unobtrusive Preference Identification. Knowledge about user preferences is a key preliminary for determining recommendations of relevance for the user. A major issue in this context is the development of new technologies which allow the elicitation of preferences in an unobtrusive fashion [30, 48, 77]. The three major modalities to support such a type of preference elicitation are the detection of facial expressions, the interpretation of recorded speech, and the analysis of physiological signals. An example of the derivation of user preferences from the analysis of eye tracking patterns is presented by Xu et al. [78] who exploit eye tracking technologies by interpreting attention times to improve the quality of a content-based filtering recommender. An approach to preference elicitation from physiological signals is presented by Janssen et al. [42] who exploit the information about skin temperature for measuring valence which is applicable to mood measurement.

Psychological Aspects of Recommender Systems. Building efficient recommendation algorithms and the corresponding user interfaces requires a deep understanding of human decision processes. This goal can be achieved by analyzing existing psychological theories of human decision making and their impact on the construction of recommender systems. Cosley et al. [13] already showed that the style of item rating presentation has a direct impact on a users' rating behavior. Adomavicius et al. [1] showed the existence of anchoring effects in different collaborative filtering scenarios. With their work, Adomavicius et al. [1] confirm the results presented by Cosley et al. [13] but they show in more detail in which way rating drifts can have an impact on the rating behavior of a user. As already mentioned, recommendation technologies improve the quality of persuasive interfaces ([5, 61]). Future recommenders should exploit the information provided by the mentioned preference elicitation methods [42].

8 Conclusions

With this chapter we provide an overview of new and upcoming applications of recommendation technologies. This overview does not claim to be complete but is the result of an analysis of work published in recommender systems related workshops, conferences, and journals. Beside providing insights into new and upcoming applications of recommendation technologies we also provide a discussion of issues for future research with the goal of advancing the state of the art in recommender systems which is characterized by a more user-focused and personal assistance based recommendation paradigm.

References

1. Adomavicius, G., Bockstedt, J., Curley, S., Zhang, J.: Recommender systems, consumer preferences, and anchoring effects. In: RecSys 2011 Workshop on Human Decision Making in Recommender Systems. pp. 35–42 (2011)
2. Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommander systems: A survey of the state-of-the-art and possible extensions. IEEE Transactions on Knowledge and Data Engineering 17(6), 734–749 (2005)

3. Ballatore, A., McArdle, G., Kelly, C., Bertolotto, M.: RecoMap: an interactive and adaptive map-based recommender. In: 25th ACM Symposium on Applied Computing (ACM SAC 2010). pp. 887–891. Sierre, Switzerland (2010)
4. Barker, V., O'Connor, D., Bachant, J., Soloway, E.: Expert systems for configuration at Digital: XCON and beyond. Communications of the ACM 32(3), 298–318 (1989)
5. Berkovsky, S., Freyne, J., Coombe, M., Bhandari, D.: Recommender algorithms in activity motivating games. ACM Conference on Recommender Systems (RecSys'09) pp. 175–182 (2010)
6. Berkovsky, S., Freyne, J., Oinas-Kukkonen, H.: Influencing Individually: Fusing Personalization and Persuasion. ACM Transactions on Interactive Intelligent Systems 2(2), 1–8 (2012)
7. Brocco, M., Groh, G.: Team Recommendation in Open Innovation Networks. In: ACM Conference on Recommender Systems (RecSys'09). pp. 365–368. NY, USA (2009)
8. Burke, R.: Knowledge-based recommender systems. Encyclopedia of Library and Information Systems 69(32), 180–200 (2000)
9. Burke, R., Ramezani, M.: Matching recommendation technologies and domains. Recommender Systems Handbook pp. 367–386 (2010)
10. Chatzopoulou, G., Eirinaki, M., Poyzotis, N.: Query Recommendations for Interactive Database Exploration. In: 21st Intl. Conference on Scientific and Statistical Database Management. pp. 3–18 (2009)
11. Chesbrough, H.: Open Innovation: The New Imperative for Creating and Profiting from Technology
12. Chung, R., Sundaram, D., Srinivasan, A.: Integrated personal recommender systems. In: 9th ACM Intl. Conference on Electronic Commerce. pp. 65–74. Minneapolis, MN, USA (2007)
13. Cosley, D., Lam, S., Albert, I., Konstan, J., Riedl, J.: Is seeing believing how recommender system interfaces affect users opinions. In: CHI03. pp. 585–592 (2003)
14. Cubranic, D., Murphy, G., Singer, J., Booth, K.: Hipikat: A Project Memory for Software Development. IEEE Transactions of Software Engineering 31(6), 446–465 (2005)
15. Dias, M., Locher, D., Li, M., El-Deredy, W., Lisboa, P.: The value of personalized recommender systems to e-business. In: 2nd ACM Conference on Recommender Systems (RecSys'08). pp. 291–294. Lausanne, Switzerland (2008)
16. Ducheneaut, N., Patridge, K., Huang, Q., Price, B., Roberts, M.: Collaborative Filtering Is Not Enough? Experiments with a Mixed-Model Recommender for Leisure Activities. In: 17th Intl. Conference User Modeling, Adaptation, and Personalization (UMAP 2009). pp. 295–306. Trento, Italy (2009)
17. Fakhraee, S., Fotouhi, F.: TupleRecommender: A Recommender System for Relational Databases. In: 22nd Intl. Workshop on Database and Expert Systems Applications (DEXA). pp. 549–553. Toulouse, France (2011)
18. Falkner, A., Felfernig, A., Haag, A.: Recommendation Technologies for Configurable Products. AI Magazine 32(3), 99–108 (2011)
19. Fano, A., Kurth, S.: Personal choice point: helping users visualize what it means to buy a BMW. In: 8th Intl. Conference on Intelligent User Interfaces (IUI 2003). pp. 46–52. Miami, FL, USA (2003)
20. Faulring, A., Mohnkern, K., Steinfeld, A., Myers, B.: The Design and Evaluation of User Interfaces for the RADAR Learning Personal Assistant. AI Magazine 30(4), 74–84 (2009)

21. Felfernig, A., Burke, R.: Constraint-based Recommender Systems: Technologies and Research Issues. In: 10th ACM Intl. Conference on Electronic Commerce (ICEC'08). pp. 17–26. Innsbruck, Austria (2008)
22. Felfernig, A., Friedrich, G., Jannach, D., Zanker, M.: An Integrated Environment for the Development of Knowledge-based Recommender Applications. Intl. Journal of Electronic Commerce (IJEC) 11(2), 11–34 (2006)
23. Felfernig, A., Friedrich, G., Schubert, M., Mandl, M., Mairitsch, M., Teppan, E.: Plausible Repairs for Inconsistent Requirements. In: IJCAI'09. pp. 791–796. Pasadena, CA (2009)
24. Felfernig, A., Mandl, M., Pum, A., Schubert, M.: Empirical Knowledge Engineering: Cognitive Aspects in the Development of Constraint-based Recommenders. In: 23rd Intl. Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems (IEA/AIE 2010). pp. 631–640. Cordoba, Spain (2010)
25. Felfernig, A., Reinfrank, F., Ninaus, G.: Resolving Anomalies in Feature Models. In: 20th Intl. Symposium on Methodologies for Intelligent Systems. pp. 1–10. Macau, China (2012)
26. Felfernig, A., Schubert, M., Zehentner, C.: An Efficient Diagnosis Algorithm for Inconsistent Constraint Sets. Artificial Intelligence for Engineering Design, Analysis, and Manufacturing (AIEDAM) 25(2), 175–184 (2011)
27. Felfernig, A., Zehentner, C., Blazek, P.: CoreDiag: Eliminating Redundancy in Constraint Sets. In: 22nd Intl. Workshop on Principles of Diagnosis. Munich, Germany (2011)
28. Felfernig, A., Zehentner, C., Ninaus, G., Grabner, H., Maalej, W., Pagano, D., Weninger, L., Reinfrank, F.: Group Decision Support for Requirements Negotiation. Springer Lecture Notes in Computer Science (7138), 1–12 (2011)
29. Fogg, B.: Persuasive Technology – Using Computers to Change What We Think and Do. Morgan Kaufmann Publishers (2003)
30. Foster, M., Oberlander, J.: User Preferences Can Drive Facial Expressions: Evaluating an Embodied Conversational Agent in a Recommender Dialog System. User Modeling and User-Adapted Interaction (UMUAI) 20(4), 341–381 (2010)
31. Garcia-Molina, H., Koutrika, G., Parameswaran, A.: Information seeking: convergence of search, recommendations, and advertising. Communications of the ACM 54(11), 121–130 (2011)
32. Golbeck, J.: Computing with social trust. Springer (2009)
33. Goldberg, D., Nichols, D., Oki, B., Terry, D.: Using Collaborative Filtering to weave an information Tapestry. Communications of the ACM 35(12), 61–70 (1992)
34. Hammer, S., Kim, J., Andr, E.: MED-StyleR: METABO diabetes-lifestyle recommender. In: 4th ACM Conference on Recommender Systems. pp. 285–288. Barcelona, Spain (2010)
35. Happel, H., Maalej, W.: Potentials and Challenges of Recommendation Systems for Software Engineering. In: Intl. Workshop on Recommendation Systems for Software Engineering. pp. 11–15. Atlanta, GA, USA (2008)
36. Hoens, T., Blanton, M., Chawla, N.: Reliable Medical Recommendation Systems with Patient Privacy. In: 1st ACM Intl. Health Informatics Symposium (IHI 2010). pp. 173–182. Arlington, Virginia, USA (2010)
37. Hofmann, H., Lehner, F.: Requirements engineering as a success factor in software projects. IEEE Software 18(4), 58–66 (2001)
38. Holmes, R., Walker, R., Murphy, G.: Approximate structural context matching: An approach to recommend relevant examples. IEEE Transactions on Software Engineering 32(12), 952–970 (2006)

39. Huang, Y., Chang, Y., Sandnes, F.: Experiences with RFID-based interactive learning in museums. Intl. Journal of Autonomous and Adaptive Communication Systems 3(1), 59–74 (2010)
40. Jannach, D., Bundgaard-Joergensen, U.: SAT: A Web-Based Interactive Advisor for Investor-Ready Business Plans. In: Intl. Conference on e-Business (ICE-B 2007). pp. 99–106 (2007)
41. Jannach, D., Zanker, M., Felfernig, A., Friedrich, G.: Recommender Systems – An Introduction. Cambridge University Press (2010)
42. Janssen, J., Broek, E., Westerink, J.: Tune in to your emotions: a robust personalized affective music player. User Modeling and User-Adapted Interaction (UMUAI) 22(3), 255–279 (2011)
43. Kapoor, N., Chen, J., Butler, J., Fouty, G., Stemper, J., Riedl, J., Konstan, J.: Techlens: a researcher's desktop. In: 1st Conference on Recommender Systems. pp. 183–184. Minneapolis, Minnesota, USA (2007)
44. Kersten, M., Murphy, G.: Using task context to improve programmer productivity. In: 14th ACM SIGSOFT Intl. Symposium on Foundations of Software Engineering. pp. 1–11 (2010)
45. Konstan, J., Miller, B., Maltz, D., Herlocker, J., Gordon, L., Riedl, J.: GroupLens: applying collaborative filtering to Usenet news. Communications of the ACM 40(3), 77–87 (1997)
46. Konstan, J., Riedl, J.: Recommender systems: from algorithms to user experience. User Modeling and User-Adapted Interaction (UMUAI) 22(1), 101–123 (2012)
47. Koren, Y., Bell, R., Volinsky, C.: Matrix Factorization Techniques for Recommender Systems. IEEE Computer 42(8), 30–37 (2009)
48. Lee, T., Park, Y., Park, Y.: A time-based approach to effective recommender systems using implicit feedback. Expert Systems with Applications 34(4), 3055–3062 (2008)
49. Leitner, G., Fercher, A., Felfernig, A., Hitz, M.: Reducing the Entry Threshold of AAL Systems: Preliminary Results from Casa Vecchia. In: 13th Intl. Conference on Computers Helping People with Special Needs. pp. 709–715. Linz, Austria (2012)
50. LeMay, M., Haas, J., Gunter, C.: Collaborative Recommender Systems for Building Automation. In: Hawaii Intl. Conference on System Sciences, Waikoloa, Hawaii, 2009. pp. 1–10. Hawaii, USA (2009)
51. Linden, G., Smith, B., York, J.: Amazon.com Recommendations – Item-to-Item Collaborative Filtering. IEEE Internet Computing 7(1), 76–80 (2003)
52. Mandl, M., Felfernig, A., Tiihonen, J., Isak, K.: Status Quo Bias in Configuration Systems. In: 24th Intl. Conference on Industrial Engineering and Other Applications of Applied Intelligent Systems (IEA/AIE 2011). pp. 105–114. Syracuse, NY, USA (2011)
53. Martin, F., Donaldson, J., Ashenfelter, A., Torrens, M., Hangartner, R.: The Big Promise of Recommender Systems. AI Magazine 32(3), 19–27 (2011)
54. Masthoff, J.: Group recommender systems: Combining individual models. Recommender Systems Handbook pp. 677–702 (2011)
55. McCarey, F., Cinneide, M., Kushmerick, N.: Rascal – A Recommender Agent for Agile Reuse. Artificial Intelligence Review 24(3–4), 253–273 (2005)
56. Misirli, A., Bener, A., Kale, R.: AI-Based Software Defect Predictors: Applications and Benefits in a Case Study. AI Magazine 32(2), 57–68 (2011)
57. Mobasher, B., Cleland-Huang, J.: Recommender Systems in Requirements Engineering. AI Magazine 32(3), 81–89 (2011)
58. Pazzani, M., Billsus, D.: Learning and revising user profiles: The identification of interesting web sites. Machine Learning 27, 313–331 (1997)

59. Peischl, B., Zanker, M., Nica, M., Schmid, W.: Constraint-based Recommendation for Software Project Effort Estimation. Journal of Emerging Technologies in Web Intelligence 2(4), 282–290 (2010)
60. Pinxteren, Y., Gelijnse, G., Kamsteeg, P.: Deriving a recipe similarity measure for recommending healthful meals. In: 16th Intl. Conference on Intelligent User Interfaces. pp. 105–114. Palo Alto, CA, USA (2011)
61. Pribik, I., Felfernig, A.: Towards Persuasive Technology for Software Development Environments: An Empirical Study. In: Persuasive Technology Conference (Persuasive 2012). pp. 227–238 (2012)
62. R. Mooney, L.R.: Content-based book recommending using learning for text categorization. User Modeling and User-Adapted Interaction 14(1), 37–85 (2004)
63. Ramiez-Gonzales, G., Munoz-Merino, P., Delgado, K.: A Collaborative Recommender System Based on Space-Time Similarities. IEEE Pervasive Computing 9(3), 81–87 (2010)
64. Ramos, C., Augusto, J., Shapiro, D.: Ambient Intelligence – the Next Step for Artificial Intelligence. IEEE Intelligent Systems 23(2), 15–18 (2008)
65. Reiter, R.: A theory of diagnosis from first principles. AI Journal 23(1), 57–95 (1987)
66. Robillard, M., Walker, R., Zimmermann, T.: Recommendation Systems for Software Engineering. IEEE Software 27(4), 80–86 (2010)
67. Sabin, D., Weigel, R.: Product Configuration Frameworks - A Survey. IEEE Intelligent Systems 14(4), 42–49 (1998)
68. Schafer, J., Konstan, J., Riedl, J.: E-Commerce Recommendation Applications. Journal of Data Mining and Knowledge Discovery 5(1–2), 115–153 (2011)
69. Sommerville, I.: Software Engineering. Pearson (2007)
70. Tayebi, M., Jamali, M., Ester, M., Glaesser, U., Frank, R.: Crimewalker: A Recommender Model for Suspect Investigation. In: ACM Conference on Recommender Systems (RecSys'11). pp. 173–180. Chicago, IL (2011)
71. Terveen, L., Hill, W.: Beyond Recommender Systems: Helping People Help Each Other. In: HCI in the New Millennium. pp. 487–509. Addison-Wesley (2001)
72. Thiesse, F., Michahelles, F.: Building the Internet of Things Using RFID. IEEE Internet Computing 13(3), 48–55 (2009)
73. Thorleuchter, D., VanDenPoel, D., Prinzie, A.: Mining ideas from textual information. Expert Systems with Applications 37(10), 7182–7188 (2010)
74. Tuzhilin, A., Koren, Y.: 2nd KDD Workshop on Large-Scale Recommender Systems and the Netflix Price Competition. pp. 1–34 (2008)
75. Wiesner, M., Pfeifer, D.: Adapting Recommender Systems to the Requirements of Personal Health Record Systems. In: 1st ACM Intl. Health Informatics Symposium (IHI 2010). pp. 410–414. Arlington, Virginia, USA (2010)
76. Wilson, D., Leland, S., Godwin, K., Baxter, A., Levy, A., Smart, J., Najjar, N., Andaparambil, J.: SmartChoice: An Online Recommender System to Support Low-Income Families in Public School Choice. AI Magazine 30(2), 46–58 (2009)
77. Winoto, P., Tang, T.: The role of user mood in movie recommendations. Expert Systems with Applications 37(8), 6086–6092 (2010)
78. Xu, S., Jiang, H., Lau, F.: Personalized Online Document, Image and Video Recommendation via Commodity Eye-tracking. In: ACM Conference on Recommender Systems (RecSys'08). pp. 83–90 (2008)
79. Yuan, N., Zheng, Y., Zhang, L., Xie, X.: T-Finder: A Recommender System for Finding Passengers and Vacant Taxis. IEEE Transactions on Knowledge and Data Engineering (TKDE) pp. 1–14 (2012)