Computing (2014) 96:223–257
DOI 10.1007/s00607-013-0351-z

Ubiquitous recommender systems

Christos Mettouris · George A. Papadopoulos

Received: 29 October 2012 / Accepted: 24 September 2013 / Published online: 8 October 2013

Abstract Ubiquitous recommender systems combine characteristics from ubiquitous systems and recommender systems in order to provide personalized recommendations to users in ubiquitous environments. Although not a new research area, ubiquitous recommender systems research has not yet been reviewed and classified in terms of ubiquitous research and recommender systems research, in order to deeply comprehend its nature, characteristics, relevant issues and challenges. It is our belief that ubiquitous recommenders can nowadays take advantage of the progress mobile phone technology has made in identifying items around, as well as utilize the faster wireless connections and the endless capabilities of modern mobile devices in order to provide users with more personalized and context-aware recommendations on location to aid them with their task at hand. This work focuses on ubiquitous recommender systems, while a brief analysis of the two fundamental areas from which they emerged, ubiquitous computing and recommender systems research is also conducted. Related work is provided, followed by a classification schema and a discussion about the correlation of ubiquitous recommenders with classic ubiquitous systems and recommender systems: similarities inevitably exist, however their fundamental differences are crucial. The paper concludes by proposing UbiCARS: a new class of ubiquitous recommender systems that will combine characteristics from ubiquitous systems and context-aware recommender systems in order to utilize multidimensional context modeling techniques not previously met in ubiquitous recommender systems.

Keywords Ubiquitous recommender systems · Ubiquitous computing · Context-aware recommender systems · Context modelling

Mathematics Subject Classification 68-02 · 68U99

C. Mettouris (✉) · G. A. Papadopoulos
Department of Computer Science, University of Cyprus,
1 University Avenue, P.O. Box 20537, 2109 Nicosia, Cyprus
e-mail: mettour@cs.ucy.ac.cy

1 Introduction

Ubiquitous recommender systems use ubiquitous devices in order to facilitate the user through her task in-situ by providing her with personalized recommendations; hence, they combine characteristics from both the ubiquitous domain and the recommendation domain. In order to provide an analysis on ubiquitous recommender systems research and classify it in terms of ubiquitous research and recommender systems research, we must first introduce the latter two research areas.

The term ubiquitous computing, first introduced in the nineties, refers to the shifting of the computing paradigm from the desktop PC to a more distributed and embedded form of computing [50]. Together with Pervasive Computing (for many people these terms are synonymous), ubiquitous computing introduced the concept of "anywhere, anytime computing", allowing users to interact with computers embedded in every-day objects in an "anywhere and anytime" manner. Moreover, ubiquitous computing specifies that the interaction of users with such devices must be straightforward to the degree that the user would not even notice such an interaction. In other words, in order for ubiquitous and pervasiveness to be achieved, computers must disappear from the front-end, be embedded to common objects that humans use daily and provide computational and informational services without expecting from users to explicitly and consciously interact with them.

A part of ubiquitous research deals with *Location-Based Information Systems*. Such systems utilize the user's location as context to provide users with the ability to produce and access information that is related to a location. Through these systems, users are able to produce and access information relevant to physical locations, their surroundings, as well as objects in their proximity. Many examples of such systems exist in the bibliography [5,10,11,35,48]. The digital annotations (electronic messages) in these systems may be used as pieces of information about the location, as recommendations about which locations are best offered for certain activities such as eating, drinking, studying, dancing, etc., and as reminders to others about things they have to do, places they need to be and people they have to meet.

As such systems gain acceptance by more and more people, the information overload is becoming the main problem that developers have to deal with. The amount of information a user has to access is so enormous that she is often lost and left with a feeling of disappointment and frustration. Imagine for example a Location-Based Service (LBS) that suggests restaurants in London based exclusively on the user's location. The recommendations can be too many and too variant for a user to remain satisfied. The solution is to utilize more contextual parameters than the location in order to provide more personalized recommendations to the user.

Traditional recommender systems (RS) on the other hand use filtering techniques and recommendation algorithms in order to opine about which information is best suited for a particular user. Such systems use data retrieved from user's usage, the system, the profile of the user and the items to be recommended to calculate the recommendations. The problem with the traditional recommendation approaches is that they do not utilize all available information for producing recommendations. More contextual parameters could be used in the recommendation process to result in more accurate recommendations. Adomavicius [1,2] was among the first to utilize the context into the recommendation process, introducing the Context-Aware Recommender Systems—CARS.

Ubiquitous recommender systems facilitate users on-location by providing them with personalized recommendations of items in the proximity via mobile devices. Such systems combine characteristics from both the ubiquitous domain and the recommendation domain, for each of which we will dedicate a section of this paper for description, analysis and challenges. In particular, Sect. 2 analyzes ubiquitous computing and challenges. Section 3, after introducing the main traditional recommendation methods, it focuses on the new trend in recommender systems research, the context-aware recommender systems. Section 4 is dedicated to ubiquitous recommenders: definition, related work and challenges are discussed, while an important classification is conducted about the correlation of the ubiquitous recommendation domain with the ubiquitous domain and the recommendation domain. Section 4 closes by discussing and comparing the challenges met in each type of systems discussed in the paper. Section 5 closes this paper by providing conclusions and introducing a new class of ubiquitous recommender systems called UbiCARS.

2 Ubiquitous computing

Ubiquitous computing focuses on promoting computing beyond the notion of the PC. It deals with integrating technology in everyday objects aiming to provide people with technological means that ease everyday life. The most inspiring work among all research works in ubiquitous computing comes from Weiser [51], in which he describes his ubiquitous vision. Mark Weiser, twenty years ago, envisioned a world in the future in which easy to use, context aware, pervasive electronic devices would be connected to one another and seamlessly embedded to the environment. The words “pervasive” and “context aware” are the most suitable ones to describe this vision. As depicted in his scenarios in [51], Weiser wanted those devices to be: (1) pervasive to the degree that people would not be aware that they are using them (easy to use, straight forward human-device interaction), and (2) context aware to the degree that devices would be aware of information regarding users (e.g. their characteristics, profile, background, social status, and their relations to each other), as well as other people in the proximity, various devices around, location information, etc. In his scenario in [51], Weiser proposes displaying information concerning other people located inside or outside the house on a window glass, something that is challenging even in our times, 20 years later. Weiser’s vision was adopted by the research community gaining hundreds of citations. Most of these works struggled to realize this vision of drawing away from the traditional desktop PC by implementing novel ubiquitous applications, by using mobile devices and networking technologies such as RFID, Wi-Fi and Bluetooth, by using sensors, wearable devices, public displays and more. The ultimate goal has always been to achieve context awareness and pervasiveness to the level of extent of Weiser’s thoughts.

However, as much as we would like to believe that technology has progressed regarding pervasiveness, the vision of Weiser as stated by him has not yet come to life. Even though devices and technologies similar to the ones foretold by him have already been developed and used, they are still not pervasive enough to be seamlessly embedded to the environment: "…'Ubiquitous computing' …does not just mean computers that can be carried to the beach, jungle or airport. Even the most powerful notebook computer, with access to a worldwide information network, still focuses attention on a single box" [51]. Literally every mobile device/computer people have built and used until now is a "computer that can be carried to the beach, jungle or airport", provided that there is a wireless network coverage at that place. We have successfully built devices that extend the notion of the desktop computer anyplace where networking availability exists, having "anywhere and anytime" access to information which can be offered in a personalized manner, but we have not succeeded in the two fundamental rules of Weiser: (1) to make these devices seamlessly embedded to the environment, and (2) to make the devices more context aware than the basic information gathered from a user's profile or a few sensors. Waller and Johnston [49] note that, because ubiquitous computing is mainly application driven utilizing technologies such as RFID, sensors and wearable devices, it endorses the risk of focusing on technical capabilities and challenges, ignoring at the same time Weiser's vision. They state that the common ubiquitous devices we use tend to get in the way of what we want to do and that they interact with a representation of the real world and not on the real world per se. This is true if we consider that most actions we are able to do via a computing device have a meaning only to other computing devices or to other people only if they are also using a computing device themselves. For example, when a person sends an email to a colleague she actually does not inform her colleague about the matter (no action occurs in the real world towards this direction); instead, her action reflects on her colleague's computer, which will keep the information for itself until the colleague checks her emails. In contrast to the aforementioned, Weiser's devices could automatically project information relevant to the context at the time needed and in a way that users did not have to interact at all with any strange, complex and not human friendly device. In fact, Weiser's scenarios minimize the human-computer interaction to a minimum, enabling at the same time the user to act on real objects instead of acting on representations of them [51]. The discussion above results in that, in order to achieve pervasiveness, we should move the user away from being a self conscious user to being a more abstracted recipient of information. By that it is meant that the user should not be bound to use any device that would make her conscious about using it in order to be informed. At this point a question arises: What are the challenges that researchers face in their attempt to realize Weiser's vision? Are there only technological challenges or do other kinds of challenges exist as well? We discuss this issue in the following section.

2.1 Ubiquitous computing challenges

A number of ubiquitous computing challenges exist, ranging from technological such as wireless technology limitations and power management issues, to challenges related to context-awareness, tracking user intentions and privacy concerns. In the following we list and categorize the most important challenges discussed in the bibliography.

2.1.1 General challenges

Want and Pering [50], categorize the challenges in ubiquitous computing to: (1) power management issues—how mobile devices deal with processing power and storage space and the kind of wireless technology to use in every given situation, (2) limitations in connecting devices—how are all these small devices going to be connected and managed, (3) user interface issues—since ubiquitous computing demands for many different small-scale devices of various types of interfaces and displays of various sizes, the challenge lies in developing user friendly interfaces (e.g. many functionalities of the PDA are not used because of the complexity of the interface: too many functionalities provided in too little space to display them), (4) issues related to Location Aware Computing—location-based APIs must be built to be used by a wide variety of devices, along with a variety of wireless technologies. Henricksen and colleagues [21] add to the above list the challenge of managing heterogeneous devices of different hardware and software specifications, such as sensors and actuators, embedded devices in objects (e.g. shoes), home and office appliances (e.g. videos), mobile devices and traditional desktop computers, in order for these devices to interact seamlessly. Another challenge they mention has to do with maintaining network connections while devices move between networks of different nature and characteristics. During this setting, network disconnections have to be managed in a way that is abstracted from the user, i.e. the user must have the feeling that he is continuously connected to the network despite any movement on his behalf. Besides connectivity problems, user mobility also demands for software and data mobility as well. In ubiquitous environments, people tend to use many devices simultaneously, therefore there is a need for these devices to communicate and exchange data. As Henricksen notes, program and data migration, as well as synchronization and coordination of components should not concern the developers; rather, a ubiquitous computing infrastructure facilitating interoperability is needed [21]. Davies and Gellersen [15] add that, since ubiquitous systems operate independently, each in its own context, the challenge is how to build integrated ubiquitous computing systems that will be able to easily communicate and decide together based on an integrated, common context.

2.1.2 Tracking user intentions

Satyanarayanan [43] notes that tracking user intentions is important in Pervasive Computing in order for the system to understand what system actions could help the user and not hinder her. The author uses an example to make a point: suppose a user who is viewing a video over a network connection that suddenly drops; what should the system do: (1) reduce the fidelity of the video, (2) pause briefly to find another higher-bandwidth connection, or (3) advise the user that the task can no longer be accomplished? The correct choice will depend on what the user is trying to accomplish. According to [43], current applications fail to correctly track the intentions of the user, or they do not consider them at all. Davies and Gellersen [15] also categorize the process of tracking user intentions as challenging, as well as determining the user's task accurately and react based on them in order to assist the user. According to the authors, tracking user intentions has only been achieved in extremely limited application domains.

2.1.3 Context-awareness and adaptation

An important challenge in context-awareness is to build context-aware systems that detect and manipulate the context in a human-like manner, i.e. making decisions proactively based on the context and provoke actions based on those decisions that assist the user through her task; the aforementioned should be done without any user participation or disturbance, except maybe in case of emergency [43]. Another important issue is how to obtain contextual information. Contextual information can be any information related to the user, the computing system, the environment of the user and any other relevant information regarding the interaction of the user and the system [17]. The user's personal computing space [43] can be used as the user's context (any information regarding the user taken from her personal profile, calendars, to-do lists, etc.), various types of context can be sensed in real time like location, people and objects nearby, while contextual parameters could incorporate the current emotional and physiological state of the user as well [43]. Contextual challenges also include the way context is represented (ontologies can be used or other context modelling techniques), the way context information is combined with system information, as well as how frequently should context information be considered. Hinze and Buchanan [22] differentiate static context (e.g. user's profile information) from fluent context (dynamic, real-time context, e.g. time) and propose that a context model should be defined for each important entity, such as the user, the locations, etc. The authors mention as challenges the capturing of the context (should it be done automatically at particular times or manually by the user?) and the process of storing the context (should it be stored on the client, on the server or on both?). On the process of accessing contextual information, Hinze and Buchanan propose that context-awareness can help in reducing the amount of data to be accessed real time, by pre-retrieving any relevant pre-known data, e.g. the static context [22]. This increases efficiency. In addition, the environment and the services it can offer are also very important, as different environments support different services as well as different contextual parameters [43]. A question that arises is: what are the minimal services that an environment needs to provide to make context-awareness feasible?

The process with which a system adapts its behaviour to the needs and preferences of its users is called adaptation. Adaptation is based on user related and context related information. Any components/devices should adapt based on the context without interfering with user's task: no user explicit interaction should be necessary. These components/devices should adapt separately and at the same time, while the user maintains a consistent view of the system/application. In ubiquitous computing, the need for adaptation often stems from poor resource availability when such resources are in demand, e.g. poor bandwidth available to a user that desperately needs email access before boarding on a flight. Based on the aforementioned, three adaptation strategies can be mentioned [43]: (1) the system guides the adaptation of applications towards using less of a resource that is scarce, (2) the system asks from the environment to guarantee a minimum level of resource availability, aiming to meet the client's demand for that resource and (3) the system recommends to the user certain actions that the user can do to optimize system behaviour. Research issues in adaptation include the correct choice between the various adaptation strategies for a particular application, which factors should the ideal adaptation strategy consider, how these factors should be weighted, whether the user should play any role in this process and how should seamless transition between strategies happen [43].

An important challenge regarding context-aware and adaptation strategies in ubiquitous systems being used by non-expert individuals has to do with how to prevent feelings of anxiety and frustration that such users may feel due to sudden system changes. Kjeldskov and Skov [25] have extended and tested a ubiquitous system for the healthcare domain and concluded that users (nurses not familiarized with context-aware computing systems) demand for less automatic changes in system behaviour (even though such changes better adjust the system based on the environment), since such sudden changes seem inscrutable to them: even the smallest automatic information update could provoke unhappy feelings to them if the update is done without the user knowing what is happening, understands it, approves it and has at all times the ability to disable it.

2.1.4 Proactivity and transparency

A big issue in ubiquitous systems is the trade-off between proactivity and transparency. If a system is proactive,[1] then in many situations it will act proactively based on internal rules and procedures and unless carefully designed, it can easily annoy a user, reducing therefore its transparency levels [43]. A question that arises is how proactive should a system be, at what circumstances, and what is the right balance between proactivity and transparency? User preferences can play an important role in this decision, as well as user experience with the system, the process, the application domain and the environment, since an experienced or expert user may have less patience and tolerance for proactivity and may therefore expect more transparency from the system; the system can then act less proactively than with other inexperienced users.

2.1.5 Changes in user roles

In ubiquitous settings a user often changes roles according to the context and the current environment she acts within: one challenge is how to capture these changes and how to react on them. As an example, consider a user that is new in town, likes foreign cuisine and on Saturday nights she likes visiting good restaurants in the proximity with the help of her mobile device. However, one Saturday night she has to work for a few hours, therefore she seeks for easy to prepare, fast solutions for her dinner. The challenge lies in capturing and utilizing both personal context (she likes foreign cuisine, it is Saturday night) and business context (she has to work this Saturday) to infer changes in the user's role and react accordingly [22]. Therefore, to successfully capture changes

[1] A proactive system acts considering and anticipating any problematic situations and events that could happen in the future.

in the user's role, both capturing the current context (i.e. the environment and the various circumstances, events and facts) and user modelling (what possible role could a person play according to context) are necessary.

2.1.6 Location issues and mobile privacy concerns

Location, as an important contextual parameter, plays an important role in ubiquitous systems. The type of location sensing technology to be used is one issue, privacy is another—should user privacy be sacrificed for location awareness and to what extent?—while a third issue is the semantic (and contextual) representation of the location in order to utilize more contextual parameters than just the location itself. For example, by semantically representing locations, one can attach to them various information resources such as a webpage, a user profile, various objects with semantic representation, etc. Location-based annotations constitute an example of information resources attached to real locations. Annotations could be information of any format (text, pictures, video, audio, etc.). Ubiquitous annotation systems allow their users to attach such information resources not only to locations, but also to objects and people. Schilit and colleagues [44], propose the movement from the simplified concept of location to more contextually rich notions of place where people and activities should also be considered. Possible problems towards this concept include the difficult management of large scale positioning data, privacy concerns regarding location-awareness and the challenge of how to associate information resources with a real-world location. According to the authors, privacy issues regarding location-awareness are related to human psychology: users often consider privacy issues when their location is to be known by a system, but at the same time they provide private information such as credit card numbers and addresses to online systems without hesitation. This happens because in the first case they simply do not see the benefit of providing their location to be used by a simple application (e.g. finding friends in the proximity), while at the latter case they clearly see the benefit of buying goods online. The authors also argue that the centralized nature of most location tracking applications (having a central server on which all user personal data are stored) discourages users from providing any personalized information, because centralized data can be accessed by anyone, not only illegally (e.g. hackers) but also by the government, corporations with interest in user data (e.g. advertisers), etc. A solution could be a decentralised schema where any personal data are stored and calculated only on the client side (the user's device). An example of such a technology is the well known GPS: the client device uses satellite links to calculate locally the user's current position. Moreover, a qualitative study on mobile privacy by Mancini and colleagues [30] that revealed a number of privacy related user boundaries, while the users were mobile, these boundaries were related to socio-cultural factors rather than physical ones, such as a user's personal boundaries regarding the personal information that should be revealed to the public and to members of their social networks, or boundaries relevant to the physical proximity of people that are not members of their social networks.

Another important challenge is how to investigate mobile privacy issues: when do they occur, how do users feel when such issues arise and how does this affect their behavior. Mancini and colleagues [30] argue that privacy issues are sensitive, difficult to study and poorly understood, that survey methods on privacy such as questionnaires and interviews provide only limited insight into users' feelings and needs and that, instead, observing user behavior in real-time should be preferred. However, in situations where users are mobile, directly observing the user may lead to undesirable modification of what would otherwise be spontaneous behavior: "any observing agent that was following the participants around would hardly go unnoticed and would therefore end up intruding into their privacy and altering their behavior" [30]. Therefore, in order to obtain meaningful and useful information regarding user privacy in mobile settings, experience sampling is normally used: participants answer questions regularly or on particular events that concern their feelings and behavior in daily life situations. Since this method demands from mobile users to spend much time in answering questions at inconvenient times and environments, the authors propose a combination of experience sampling and semi-structured interviewing, during which the user is given a memory phrase in order to better recall certain past events and their context and, by that, assist her to answer privacy related questions relevant to the event [30].

2.1.7 Authenticity of information and trust

Another trust related issue met in ubiquitous systems is researched by Lenders and colleagues in [27]: in a system where users provide their own content from particular locations, how can one trust the authenticity and quality of the information published by individuals? They state that if spatial and temporal knowledge regarding user content is known (from where was the content provided and when), then this content is more trusted and valuable, and they propose an approach where user device location and content creation time is used in the user authentication process to ensure the authenticity and quality of the content.

2.1.8 Developing dynamic information systems

Another ubiquitous challenge is that the majority of location-based information systems are not dynamic annotation systems, meaning that their information is pre-authored and users are not given the ability to dynamically produce their own annotations [20]. This "read-only" functionality can indeed be met in the majority of location-based applications such as navigation systems and various kinds of guides such as tourist guides, restaurant guides, etc. There are many challenges in developing ubiquitous dynamic annotation systems instead of the "read-only" location-based systems and Hansen[2] [20] lists them as follows: anchoring, structuring, presentation, and authoring. Anchoring refers to the process of identifying the location. An issue is the precision of the technology to be used: good precision aids in better identification of the location and consequently, better representation of that location by the annotation to be attached. Structuring refers to how the annotation is to be attached to the location.

[2] The author uses the term "resource" for the object to be annotated and "information" for the annotation itself. The present work uses the term "information resource" to refer to the annotation and no special term for the object to be annotated.

An appropriate location structure is needed in order for every location to be able to be annotated and linked. Presentation refers to whether the annotation is to be presented on or off the location and attached or detached to the location. On/off location refers to whether the user is at the actual location or not, while attached/detached to the location refers to whether the annotation is attached on the actual object/location or on a digital representation of it. Finally, the challenge of authoring and editing of annotations: how to design appropriate user interfaces for authoring and editing, which are tasks with different requirements than that of presenting the annotations.

2.1.9 Scalability

Domnitcheva [19] notes that scalability is also an important challenge in Location aware systems. Such systems should be able to cope, on the one end with many sensors providing raw data and, on the other end with many devices consuming information. The number of sensors and devices must not be fixed; rather, truly ubiquitous systems must be able to facilitate an arbitrary number of them without compromising efficiency or security.

2.2 Summary

The section above describes the most important challenges and issues in ubiquitous computing research. Table 1 summarizes these challenges by categorizing them accordingly. As expected, the ubiquitous nature of ubiquitous recommender systems brings researchers and developers in the area against most of these challenges. We discuss this issue in Sect. 4.

3 Recommender systems

3.1 Traditional recommender systems

Recommender systems, as the name implies, are systems that use a variety of filtering techniques and recommendation methods to provide personalized recommendations to their users. Recommender systems are important due to the information overload modern life experiences at all fields, and their importance will be further increased as this overload is expanding exponentially. The terms “traditional” or “un-contextual” are used to denote typical recommender systems that use limited or no contextual information to produce recommendations, as opposed to the Context-Aware Recommender Systems (CARS) that aim in using many contextual parameters to provide better recommendations [2]. Context-aware recommender systems will be studied in following sections.

Traditional recommender systems use information retrieved from the user profile, from user’s usage history, as well as information related to the items to be recommended in order to calculate the recommendations. The user profile includes valuable information regarding the user and her personality, her preferences and dislikes, her habits and more. Such information could be used by recommender systems to filter out

Table 1 Challenges in ubiquitous computing

Technological
Power (energy concerns) and storage issues due to the small size of mobile devices
Wireless technologies issues
Connectivity issues: how to connect and manage many small mobile devices
Networking issues: manage resources, scalability, devices must operate across different types of networks
HCI related
User interfaces in small devices
Devices should be able to function without user action or attention
User friendly devices/interfaces: any ordinary user should be able to use them
Aim to improve user satisfaction
Usability: new novel types of interaction
Any system adaptation should not interfere with the user's task
Track user intentions: the system should be able to infer what the user wants to do in order to assist her
The amount of input that is needed from users should be reduced by using context-awareness
Proactivity vs. transparency: proactive system knows how to react to an event. Risk: may not be transparent to user
Presentation: how will annotations be presented—on the object or not, attached or detached from the location
Anchoring: how annotations are being attached to places/objects
Authoring/editing of annotations: how, when, why?
Wearable computing: may be proved inconvenient for users
Context-awareness and adaptation related
Modelling the context: which method is more appropriate to use
Observing the context: automatically or manually?
Context sensing: retrieving context data from various sources (e.g. sensors), data inconsistencies may occur
Accuracy of contextual information should be well known during the design of ubiquitous systems
Storing the context: on server (privacy issues), on client or on both?
Systems should be more context-aware than just the location. A place is more than a location
How will the application modify its behaviour (be adapted) based on the context
Devices should not operate based only on their own context, but based on the context of the whole system
Context-awareness should be used to reduce the amount of input that is needed from users
Capture changes in the user's role by (1) capturing the current context, (2) user modelling
Components adapt based on context without interfering with user's task; user maintains a consistent view of system
Privacy and trust related
Users do not want to give up their location privacy, unless for a significant reason: no motive
Centralized location tracking is not as easily trusted by users as client side location tracking is (e.g. GPS)

Table 1 continued

Sensitive user data should be distributed only with user's consent
Due to the wireless networks, ubiquitous applications are difficult to be trusted
User boundaries regarding privacy are often related to socio-cultural factors rather than physical ones
Mobile privacy issues are difficult to investigate: need to observe user mobile behavior in real-time

any recommendations not suitable for the user. User's usage data are data produced from user actions. User actions reflect user preferences and needs, thus analyzing such actions could be valuable during the recommendation process. For example, web recommender systems may use any browser history information and logging data to opine what the user likes. Finally, information related to the items to be recommended is used. Such information depends on the particular items that are recommended. For example, in movie recommendations where the items under study are movies, such information may include the title of the movie, its genre, its duration, the actors etc. The most well known recommendation approaches are the collaborative filtering (CF), the Content-based filtering and Hybrid recommendation techniques.

3.1.1 Collaborative filtering

The *Collaborative filtering technique* recommends items that similar users to the active user[3] have highly rated (hence like). The method uses information from other users in the system to measure their similarity to the active user; the most similar to her are selected to form a "neighborhood", hence all similar users are the active user's "neighbors". The assumption made here it that those who agreed in the past tend to also agree in the future. Thus, since the active user's neighbors tend to agree with her (they are similar users), the items these neighbors like the most might be included highly in the list of preferences of the active user as well, and hence can be recommended to her. The Collaborative filtering technique is composed of two basic steps: the neighborhood formation and the recommendation extraction [41]. During neighborhood formation, the similarity between users is computed. For each user, an ordered list of similar users is computed. To compute user similarities, the proximity measures can be used [1,41]. Proximity measures measure the distance between users: users that are closer have more similar preferences [41]. The proximity measure is usually calculated using the Correlation Measure (for users a and b, the distance among them is measured by calculating the Pearson correlation) or the Cosine-based approach (users a and b are perceived as vectors in the m dimensional item space—proximity is calculated by measuring the cosine of the angle between the two vectors) [1,41]. After finding similar users, the neighborhood formation can follow the Center-based schema or the Aggregate one [41]. The Center-based schema gives higher priority to those users that are closest (more similar) to the active user. The Aggregate neighborhood schema on the other hand gives higher priority to the users that are closer to the centre of the

[3] We will refer to the user who is to be provided with recommendations as the "active user".

current neighborhood (schematically the neighborhood changes form each time a user is being added). The second step, the recommendation extraction, extracts the top-N recommendations which are the items that have the highest scores among the list of neighbors of the active user.

Besides the User-based Collaborative filtering (explained above), a similar technique called Item-based Collaborative filtering was also proposed [42]. Item-Based CF is similar to User-Based CF but instead of using similarities among users, similarities among items are used. The similarity between 2 items is computed by finding the users who have rated (or bought) both of these items and then applying a similarity computation method [42]. Similarity computation methods rely on user ratings on items, and not on item characteristics, as is the case in Content-Based filtering (discussed in the next section). Once the similarity of items is calculated and similar items have been specified, the process uses prediction techniques to recommend to the user the most appropriate items [42].

The biggest advantage of Collaborative Filtering is that it does not depend on any system representations of the items to be recommended and can function well with complex items such as music and movies [9]. The most well known problem of this method is the "New user" problem, which states that a new user has to rate a certain amount of items before the system may effectively apply the algorithm. This is true if we consider that neighborhood formation, which finds user similarities, is based on previous user ratings. Therefore, a new user with no previous ratings is left with no recommendations. Moreover, the "New item" problem is also very common where the recommender cannot recommend a new item until it has been rated by a number of users. Finally, scalability issues have been reported, as well as performance problems for users with large information set [42].

3.1.2 Content-based filtering

Content-based filtering suggests that the active user will be recommended with those items that are most similar to the items she has highly rated (or bought the most) in the past. Items are similar in terms of their content (characteristics, features and attributes of the items are used), and therefore the algorithm differs from Item-based CF. Some of the most important Content-based issues are [1,9]: items suffer from over-specialization, since the algorithm focuses only on items already rated (or purchased) by a user as well as other items similar to those, excluding in this manner other, different types of items. The "New user" problem also applies here, where a new user has to rate a certain amount of items before the system can apply the algorithm. This happens because in order for the system to learn any user preferences, the user must rate (or buy) a number of items. Moreover, Content-based filtering is limited to use only features that are explicitly associated with the items to be recommended.

3.1.3 Hybrid and other recommenders

Hybrid recommenders constitute a combination of CF and Content-based recommenders. The combination can exist either within a system, i.e. the system uses a combination of CF and content-based methods, or by using two separate systems, i.e. a CF recommender and a Content-based recommender accordingly [1]. In the first case, an example would be to construct user profiles via content-based techniques and then directly compare these profiles to opine about the similarity of users to provide collaborate recommendations [1]. In the latter case, one can either combine the output of the two systems in one common recommendation list, or choose which of the two recommendation sets is going to be displayed according to appropriate metrics.

Besides CF, Content-based filtering and Hybrid recommenders, other methods have been proposed in the bibliography such as Demographic recommender systems, Utility-based recommenders and Knowledge-based recommenders [9]. For a review of these approaches, as well as a more extended review on the three methods described in this section, the reader is referred to [9,16,24,41,42].

3.2 Context-awareness in recommender systems

Recommender systems have attracted the research community's interest for the past fifteen years. Many techniques have been proposed, as well as many extensions and improvements, but it was not until recently that the research community realized that recommenders have only been using a part of the available information for producing recommendations. The problem was that traditional recommenders do not utilize the context. Instead, they focus on two dimensions: the user and the items (also called two-dimensional recommenders), excluding other contextual data that could be used in the recommendation process, such as the day/time, with whom the user is with, weather conditions, etc. Adomavicius and colleagues were among the first to prove that contextual information incorporated in the recommendation process indeed improves recommendations; they proposed that the recommendation procedure should not be two-dimensional but rather multi-dimensional, introducing the Context-Aware RecommenderSystems—CARS [1,2].

According to Adomavicius and Tuzhilin [2], contextual information can be used in two ways for producing recommendations. They named the first "Recommendation via Context-Driven querying and search", where systems use contextual information from the environment (e.g. location), the user (e.g. profile information, user actions) and the system as searching parameters to search for the most relevant items in a repository to recommend. Examples of such systems are ubiquitous systems and location-based systems that utilize contextual information, often from the environment by using sensors, to recommend appropriate items in the proximity (e.g. restaurants, touristic attractions, etc.). The second way for producing recommendations is called "Recommendation via Contextual preference elicitation and estimation" [2] where systems focus in modelling user preferences by using various methods, e.g. observing the user while interacting with a system or by receiving appropriate feedback from the user regarding the recommendations.

Using contextual information to provide recommendations is a very important procedure followed by ubiquitous recommender systems as well. Therefore, in order to properly analyze and categorize these systems, a review on the two aforementioned ways for producing recommendations is mandatory. In the remaining of this section we will focus on each of the two aforementioned ways and on related challenges.

3.2.1 Recommendation via context-driven querying and search

The first way of including contextual information in the recommendation process refers to ubiquitous approaches, where context is sensed (e.g. via sensors) or retrieved (e.g. from the user profile, the web, etc.) in order to be included in the search for appropriate recommendations. Early works included only few contextual parameters such as the location, user identity and people and objects around, aiming at informing people about things in their proximity[4] [5,10,11,48]. In the area of location-based notification systems (LBNS), many works offer awareness through location-based notifications in various domains like police patrolling, firefighting, military and tourism. In [46] for example, a mobile service is presented that notifies police officers about warrants, agreements and police focal points in their vicinity.

Besides trivial contextual parameters such as the location, many ubiquitous systems use more contextual information in their search for recommendations, such as date, time, season, temperature, user's interests, user's emotional status, etc. Cena and colleagues [14] propose a system called UbiquiTo that recommends items (places to visit, accommodations, restaurants) by using three types of context: user preferences, current context (location, time, etc.) and device type. The recommender calculates a score to each item and then orders these items to produce a list to be presented to the user. The scoring procedure takes under account the user's interest (e.g. if she does not like visiting museums then she will not be recommended such items) and the user's location (only items close to her location will be recommended). van Setten [45] offer context-aware recommendations based on the context (user location, time) and user interests. Böhmer and Bauer [7] propose a system for recommending applications for mobile devices. Due to the nature of the system (i.e. recommending mobile applications to mobile device users), much of the context can be captured automatically by the system via the mobile device. For example, a system that recommends books cannot be aware of information like how often the user reads each book or for how long she reads it; on the contrary, this can be achieved with mobile applications. Therefore, the system acquires context automatically (implicitly) via an application which runs in the background as a service and acquires as much contextual information as possible (user's location, time, what applications she is currently using or has been using in the past, when and for how long, etc.).

Current research on "context-driven querying and search" ubiquitous systems is mainly driven by context sensing and context identification and focuses at utilizing as much of the available contextual information as possible in order to provide better, more personalized results. However, as Jannach mentions [23] such systems mostly filter the presented information content according to users' current location and preferences in a rather static approach, not utilizing any of the sophisticated recommendation

[4] Things in the proximity could be other people, objects, or places.

algorithms with machine learning aspects met in traditional recommender systems, such as Content-based and Collaborative filtering.

3.2.2 Recommendation via contextual preference elicitation and estimation

Regarding "Recommendation via Contextual preference elicitation and estimation", three approaches were proposed: the Pre-filtering approach, the Post-filtering approach and the Contextual Modelling approach [1,2]. CARS in this group do not use two-dimensional datasets as with traditional recommenders: Users $\times$ Items $\rightarrow$ Ratings; rather they face the challenge of coming against multidimensional datasets that include additional contextual dimensions besides "users" and "items": Users $\times$ Items $\times$ Context $\rightarrow$ Ratings [2].

The *Contextual Pre-filtering approach* aims at pre-processing the input data without affecting the actual recommendation process. It filters the initial multidimensional contextual dataset before it is provided as input to the recommender system. This dataset is multidimensional because many contextual parameters have been considered. For example, a two-dimensional (2D) dataset includes only users and items as data, a three-dimensional (3D) dataset may also include the time, while a multidimensional one (MD) may include an arbitrary number of contextual parameters. The reduction-based approach is an example of Contextual Pre-filtering [2]. The aim of the reduction-based approach is to filter this MD dataset in order to produce a 2D dataset which can be used as input in any of the classical pre-existing 2D recommendation methods. In this way the actual recommendation method does not change (it is still 2D). The filtering of the initial MD dataset is done by assigning specific values to particular contextual parameters and then selecting only the dataset records that satisfy this assignment; the result is that these contextual parameters are being excluded from the initial MD dataset of the particular application, resulting in a 2D dataset. For example, let's consider a movie recommender with a 4D initial dataset and contextual parameters the day/time a movie is watched and the company with which it is watched (the other two entities/dimensions are the user and the movies). Suppose a user would like to watch a movie on Wednesday night with her boyfriend and wants to be provided with recommendations. The reduction-based approach will assign specific values to the contextual parameters day/time and company as follows: day/time = "weekday/after 8 p.m." and company = "boyfriend" and select only the dataset records that include day/time = "weekday/after 8" and company = "boyfriend". By this, the reduction-based approach not only reduces the dimensions of the initial 4D dataset to two (by assigning specific values to contextual parameters the corresponding contextual dimensions in the initial dataset are eliminated and the resulting dataset is reduced, in this example from 4D it becomes 2D and can be provided as input to any pre-existing recommendation method), but also to exclude any irrelevant data from participating in the recommendation process (all records considering day/time = "weekend" are irrelevant because the user is not interested in watching a movie at the weekend; considering irrelevant information in the recommendation process may lead to ineffective recommendations).

Another Pre-filtering example is the work of Baltrunas and Amatriain [3] who proposed a Pre-filtering method for recommending music to users. As they note, music recommendations have different characteristics than other types of recommendations because users tend to repeatedly use the same items more than once, i.e. when they listen to the same songs repeatedly (in book recommenders for example this does not apply). Moreover, besides user ratings on music items, a music recommender may also consider implicit feedback retrieved automatically: the songs a user has listened to, the number of times she has listened to each song, the artists she chooses more often, etc. According to the authors, an issue considering implicit feedback data is that information regarding negative user preferences could be missed, because the recommender only knows the items the user likes but is not aware of what applies to the rest of them: does the user like them or not? The authors propose a time-aware RS where the number of times a user has listened to an artist suggests how much she likes the particular artist. Instead of the whole user profile, they use micro-profiles which are basically snapshots of the user profile in certain time periods, e.g. morning, noon, night. The main problem with this method is how to divide a continuous contextual variable, such as time, into many distinct slots. For example, for one person the morning period can be 6–9 a.m. while for others be 8–12 a.m., etc. By using only the time-based micro-profile of the user instead of the whole profile the authors report that they have better accuracy results. This is a Pre-filtering approach since by using micro-profiles the input dataset of the recommendation algorithm is reduced. More works that use the Pre-filtering method are [4,18,29].

The *Contextual Post-filtering approach* does not involve any contextual filtering of the input dataset, nor does it involve the context in the recommendation process: recommendations are produced in the same way as in the traditional 2D recommendation systems. Rather, the Contextual Post-filtering approach filters the results of the 2D recommender based on some contextual parameters [2]. The filtering excludes any irrelevant recommendations based on the context and then prioritizes the resulted recommendations according to their relevance to the context.

The *Multidimensional Contextual Modelling approach*, as the name implies, is the only approach of the three that incorporates the multidimensional context in the actual recommendation process. According to Adomavicius and Tuzhilin [2], while the Pre-filtering and Post-filtering approaches can use traditional 2D recommendation methods, the contextual modelling approach promotes truly multidimensional recommendation methods, which essentially represent predictive models or heuristic calculations that incorporate contextual information in addition to the user and item data. The input data include more dimensions besides users and items, thus there is a need for developing appropriate methods that will include these dimensions in the recommendation process.

As a Contextual Modelling example we note the work of Oku and colleagues [33]. Their approach extends a 2D classifier method called Support Vectors Machine—SVM to support contextual parameters. Their method is called Contextual-SVM and adds a general contextual dimension to the 2D method, thus making it 3D. For given contextual information, the method results in a plane (the one dimension of the 3D is known—the context) which determines the preferences of each user for that context. To opine about how similar the preferences of two users are, the authors apply a similarity function that is related to the positive and negative data of each user (positive data include items the user likes, while negative data include items that she does not like). The similarity function is used in order to find the most similar users to the active user and form a neighborhood, as in the CF recommendation method. To test their approach, the authors applied it to a recommender system for restaurants with generally good results. As they note, it is possible to classify user preferences based on the context by using the Contextual-SVM.

3.2.3 Challenges and issues

Research issues regarding the Contextual Pre-filtering approach include choosing the right generalized pre-filter in order to obtain the right dataset that will produce the best recommendations [2], dealing with potential computational complexity due to context granularity (context has often great detail which introduces computational complexity in the recommendation algorithms) and choosing the appropriate contextual parameters in application domains where context has great granularity and allows many possibilities. In addition, an open issue relates to combining the reduction-based approach with many 2D recommendation techniques (CF, Content-based filtering, Hybrid methods, etc.) to infer which combination is best for all application domains, if such a combination exists, or whether different combinations apply better in different domains.

As with the Pre-filtering approach, research issues related to the Post-filtering approach include dealing with potential computational complexity due to context granularity, choosing the appropriate contextual parameters in application domains where context has great granularity and allows many possibilities, as well as combining this approach with many 2D recommendation techniques to infer which combination is best.

Research issues concerning the contextual modelling approach have to do with what dimensions should be included in the MD recommendation model, how can classical 2D recommendation techniques be extended into multidimensional and what new multidimensional techniques can be developed. For more information on context-aware recommender systems the reader is referred to [2].

3.3 Summary

Table 2 categorizes the challenges met in the different recommendation methods in traditional recommender systems and CARS, as these were presented in Sect. 3. As discussed in this section, CARS are divided to two types of systems according to the way recommendations are produced: systems that search repositories according to the context and systems that model user preferences. While the second way specifies systems closely related to the CARS research area, the first one specifies systems in the area of ubiquitous computing where contextual information from the environment (e.g. location), the system and the user is considered in the search for items

Table 2 Challenges in traditional recommender systems and CARS

Challenges/issues
CF
"New user" problem: a new user has to rate a certain amount of items for the algorithm to be applied
"New item" problem: the recommender cannot recommend a new item until it has been rated by a number of users
Scalability issues
Performance problems for users with large information set
Content-based filtering
"New user" problem: a new user has to rate a certain amount of items for the algorithm to be applied
Over-specialization of items: focus only on items rated or purchased and similar ones. Exclude other different types
Limited use of information: use only features that are explicitly associated with the items to be recommended
Pre-filtering
Choosing the right generalized pre-filter to obtain the right dataset that will produce the best recommendations
Pre-filtering, post-filtering
Computational complexity due to context granularity
Choose appropriate context parameters in domains where context has great granularity: allows many possibilities
Combining the approach with 2D recommendation techniques:
(1) which combination is best for all application domains?
(2) can different combinations apply better in different domains?
Contextual modelling
Choosing the appropriate dimensions to be included in the MD recommendation model
Extending classical 2D recommendation techniques to multidimensional (MD)
Develop new multidimensional techniques

suitable to be recommended. We state that this latter set of systems, in combination with traditional recommendation methods, define the ubiquitous recommender systems.

4 Ubiquitous recommender systems

4.1 Definition

Ubiquitous recommender systems facilitate users on-location by providing them with personalized recommendations of items in the proximity via mobile devices. Intelligent tourist guides, navigation aids, and shopping recommenders that recommend based upon user activities and behavior patterns are examples of such systems [31]. In the sense of facilitating the users on-location by recommending items in the proximity, ubiquitous recommenders are similar to typical context-aware ubiquitous systems that provide "Recommendations via Context-Driven querying and search" (see

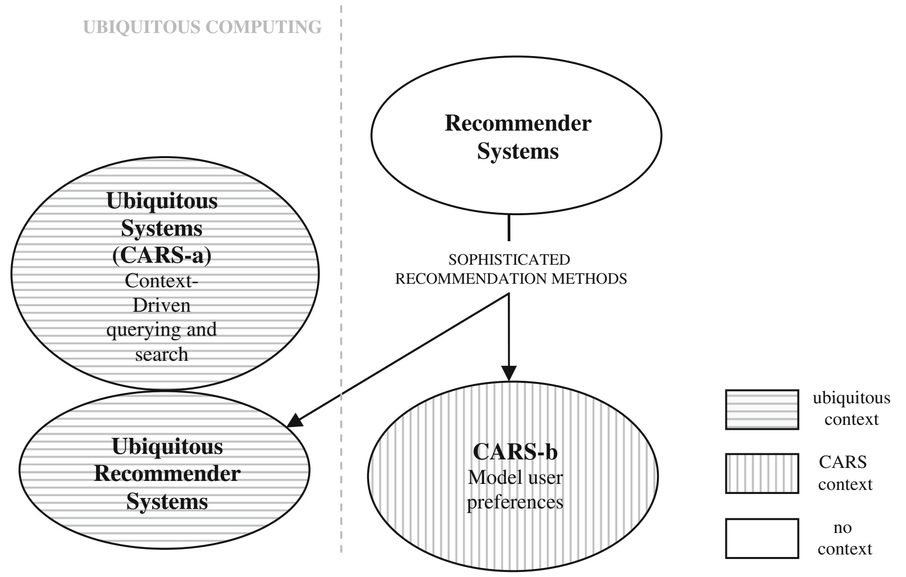
UBIQUITOUS COMPUTING
Recommender Systems
SOPHISTICATED RECOMMENDATION METHODS
Ubiquitous Systems (CARS-a) Context-Driven querying and search
Ubiquitous Recommender Systems
CARS-b Model user preferences
ubiquitous context
CARS context
no context

Fig. 1 Classification of Ubiquitous Recommender Systems domain

Sect. 3.2.1). However, we chose to categorize ubiquitous recommenders alone in this section due to two main differences: (1) ubiquitous recommenders do not focus mainly on context the way typical ubiquitous systems do and most important: (2) the recommendation procedure they follow does not rely on just a "search in repositories" (see Sect. 3.2.1); rather, they use more complex recommendation methods, similar to those traditional recommender systems use. The aforementioned are depicted in Fig. 1. The broad area of Ubiquitous Computing includes two sets: "Ubiquitous systems" and "Ubiquitous Recommender Systems". The set "Ubiquitous Systems" depicts systems that provide "Recommendations via Context-Driven querying and search", the first way for producing recommendations in CARS, as stated in [2] and Sect. 3.2.1 (in this paper "Ubiquitous Systems" could be also referred to as "CARS-a"). The set "Recommender Systems" depict traditional un-contextual recommender systems, as discussed in Sect. 3.1. The set "CARS-b" depicts systems that provide "Recommendations via Contextual preference elicitation and estimation" (the second way for producing recommendations in CARS) as stated in [2] and Sect. 3.2.2. The systems in "CARS-b" utilize the context in traditional recommendation methods met in traditional recommender systems (set "Recommender Systems"); the use of the arrow from the set "Recommender Systems" to the set "CARS-b" depicts the usage of such algorithms. Note that, as discussed in Sects. 3.2.1 and 3.2.2, "CARS-b" systems use different context types and methods than ubiquitous approaches ("Ubiquitous Systems"), hence the different patterns in the corresponding sets in Fig. 1.

The set "Ubiquitous Recommender Systems" includes ubiquitous systems that use the context to facilitate users on-location via mobile devices in the same way as systems in the set "Ubiquitous Systems" do, but with the important difference that the recommendation procedure they follow does not rely on just a "search in repositories".

In this aspect, we cannot include the “Ubiquitous Recommender Systems” set as a part of the “Ubiquitous Systems” set; instead, the two sets are positioned adjacent to denote their similarities, while they also share the same context pattern. The set “Ubiquitous Recommender Systems” as already stated uses more complex and sophisticated recommendation methods than “searching in repositories”, similar to those traditional recommender systems use. This is depicted via the “SOPHISTICATED RECOMMENDATION METHODS” arrow connecting the set “Recommender systems” with the set “Ubiquitous Recommender Systems”.

4.2 Related work

As Takeuchi and Sugimoto state, the focus of ubiquitous recommenders should be in applying the techniques used in E-commerce recommendation systems to real life activities [47]. A good example of a ubiquitous recommender system is the mobile city guide called CityVoyager proposed by Takeuchi and Sugimoto in [47]. CityVoyager is an intelligent city guide system for mobile devices equipped with GPS which retrieves and recommends shops in the proximity that match each user’s preferences. The authors differentiate from typical context-aware city guides on that they focus on applying the techniques used in E-commerce recommendation systems to real shopping. In particular, they use the item-based collaborative filtering algorithm to filter shops in a similar way traditional recommenders filter items. Instead of user ratings on items, the authors use the location history of a user on locations. Similarities between shops are calculated using a similarity function. The basic function of the system is to estimate users’ preferences from the history of their location data and offer tailored shop recommendations upon request.

The most representative and well known ubiquitous recommender systems are the *ubiquitous recommender systems for products*. Such systems manage to effectively combine the benefits of the very popular, widely used E-commerce web recommenders with the benefits ubiquitous systems offer. In the remainder of this section we chose to focus on ubiquitous recommender systems for products; however, the outcomes of this study are valid for and can be extended to apply to any type of ubiquitous recommender systems.

4.2.1 Ubiquitous recommender systems for products

Reischach and colleagues [37,38] have researched how item-related ratings, textual comments and recommendations of users to other users can be applied in real settings by using ubiquitous computing. Such ratings, comments and recommendations regard critics on items that have been bought by various users, as well as suggestions on what to buy and what not to, both personalized (a user recommends to a particular user) or generalized (a user recommends to everyone). Such recommendations are more often met in E-commerce settings where users buy online through websites that also use for posting their feedback. The idea of ubiquitous recommender systems for products is to facilitate users *during their actual shopping process* (i.e. at the actual store) by recommending them with products they might like, providing them with comments, suggestions and ratings on products they are about to buy, as well as providing them with the ability to comment in real time on any item in front of them. These recommendations should be provided explicitly (the user is asking for recommendations on a particular item) rather than implicitly (recommendations are provided automatically by the system on items the user may not be interested at—in the way an advertisement is presented) [37]. Reischach and colleagues [37] argue that explicit access of recommendations can be applied while shopping at an actual shopping place via a user's mobile device: the mobile device must be able to identify whatever product the user shows interest in (e.g. by having it in front of her or holding it) and provide her with ratings, comments and suggestions on that particular item. While research on ubiquitous computing offers many recommendation systems related to mobile phones (m-commerce and mobile-shopping assistance), such ubiquitous recommendation systems yet do not exist in practice [37]. Most related works provide users with product information according only to their preferences via a mobile device on-location (but do not consider the products found on the location), or facilitate her in editing her shopping lists and browse items of her interest [26,32]. The most interesting approaches are works that deploy NFC (Near Field Communication), a mobile phone technology that enables mobile devices to read RFID tags, in order to identify real products in-situ and provide the user with information on that products [36,38–40]. More particular, Reischach and Michahelles [38] research on how mobile devices could be enabled to recommend products and to receive product recommendations in real-time and on-location (e.g. at the actual store). They argue that mobile internet connection, mobile barcode recognition and NFC are the technologies that will facilitate the interconnection of the physical world with the virtual world.

However, the users of such ubiquitous product recommenders have a completely different task than users of the traditional on-line E-commerce recommenders. The former users have more limited mental and physical resources since being mobile forces them to do other things at the same time such as shopping, talking to others, observing their surroundings, etc. Moreover, the amount of time they are willing to spend on the mobile device for receiving recommendations is much less—a study suggests that mobile users get distracted in average after 4 s, when waiting for a computational result in a mobile context [38]. Finally, mobile devices have many interface limitations (small screen, small buttons, no mouse or keyboard), connectivity limitations and resources limitations that traditional PCs do not. From the aforementioned we can summarize that not only technical, but also important HCI related challenges arise when designing a truly ubiquitous recommender system for products.

4.2.2 The APriori example

In an attempt to develop a truly ubiquitous recommendation system for products, the authors of [38] have designed and developed a ubiquitous system called APriori that enables users to scan any RFID tagged product by using a mobile device and receive/produce recommendations, reviews and ratings on that product. The important feature here is that of scanning RFID tagged products with a mobile phone: since on one hand, all products in the market are tagged with RFID technology and on the other, mobile devices are the most well known, mostly used and most successful pervasive devices recently created, one can assume that such a product recommender could easily gain wide acceptance and be proved revolutionary. The particular system differs from other related works in that user ratings can be applied not only on standard, predefined parameters of a product, but also on user-defined ones that can be set dynamically on-the-fly. In this way users are more flexible in providing their ratings; a drawback is that user-defined product parameters can be too many or irrelevant to the product. Another important feature the authors propose is enabling the user to comment on an item during usage in real-time via a mobile device, since such functionality would be more effective than commenting on the item at a later time off-location: user emotions will affect the user into expressing her real opinion in-situ which the system will be able to capture. User experience with the system revealed trust and motivation issues: some users would not trust the opinion of random users while others stated that they did not believe that people would actually go through the trouble of rating products without additional motives [38]. A user study on the above system [36] revealed that almost all users would appreciate the system on their mobile phone, while some were reluctant on relying on other people's opinion on products. Some users were in general reluctant towards mobile applications, but as they mentioned, they would try the system at least once. This is a typical trust issue ubiquitous systems face (see Sect. 2.1: Ubiquitous Computing Challenges).

The APriori ubiquitous system described in this section served as a good example and representative of ubiquitous recommender systems. By focusing on its ubiquitous capabilities that are able to facilitate a user on-location by using state of the art technology, we were able to describe not only the technologies and methods used, but also the many challenges and issues met in this demanding research area, as well as the way they were overcome. Moreover, the system's recommendation methods, although not adequately described in this work, could successfully provide recommendations to users in order to gain feedback [36]. In the following section we provide the "big picture" regarding ubiquitous recommender systems research by summarizing challenges and issues related to the area.

4.3 Ubiquitous recommender systems challenges

Many challenges related to ubiquitous computing research and recommender systems research apply also to ubiquitous recommender systems.

4.3.1 Challenges related to ubiquitous computing that also apply to ubiquitous recommender systems

Due to operating in ubiquitous environments via mobile devices, ubiquitous recommender systems face important technological challenges such as energy concerns, storage limitations, wireless technologies issues, connectivity issues and networking issues (network connections must be maintained while devices move between various networks of different characteristics).

The mobile device must be able to track user intentions in order for the system to understand what system actions could help the user accomplish her goals. For example, the mobile device of a ubiquitous recommender system for products must be able in real time to identify the item the user is interested in, i.e. she is having in front of her or holding at any given time. Appropriate state of the art technologies should be used effectively towards this aim.

Users interact with small, often non usable devices that need much user attention. Devices may not be transparent since they operate on-field by considering various contextual parameters. Not transparent devices often provoke feelings of frustration to the users.

Context sensing: appropriate technologies and sensors must be utilized in order to infer the context in real-time. Context sensing should be done automatically and system actions based on context changes must be transparent to the user.

Appropriate usage of all available context: context is any information related to the user, the computing system, the environment of the user and the interaction of the user with the system [17]. Therefore, in a ubiquitous setting, context is not only information from the user environment, such as her location and time of day, but also relevant information from the user profile, the mobile device, etc. For example, a ubiquitous recommender system for products will consider (among other) the user preferences (what the user likes) in combination with environmental context (what the user prefers at the current day/time of day and at the particular location) in order to provide her with personalized recommendations.

Privacy concerns may arise: the user must trust the system in order to agree to provide sensitive information such as location and preferences.

Apply appropriate authentication mechanisms in order to ensure the authenticity and quality of any user created information.

Users should be able to dynamically produce their own information. Therefore, besides pre-authored information, the system must be able to support real-time authoring of information by the users; e.g. if a user would like to provide a comment/critique on a product she has purchased and tried in the past, she should be able to do so in-situ via appropriate system interfaces.

Scalability is also an issue: the system must be able to support many devices that will consume information at the same time without compromising efficiency or privacy.

4.3.2 Challenges related to recommender systems that also apply to ubiquitous recommender systems

Building appropriate user models in order to effectively store and use as much user information as possible, such as preferences, habits, rating data, etc.

The "New user" and "New item" problems are two of the most important ones when ubiquitous recommenders use the CF or Content-based filtering.

The users of ubiquitous recommenders face particular challenges not met in traditional E-commerce recommenders. This is mainly due to the small mobile devices they use instead of the common PC. Therefore, mobile users have more limited mental and physical resources due to their mobility, and the amount of time they are willing to spend on the mobile device for receiving recommendations is much less than a user on a PC.

4.4 Summary

Table 3 summarizes the challenges discussed in this work and notes whether these challenges apply to the research areas under study. Note that in the table, CARS refers only to systems that provide "Recommendations via Contextual preference elicitation and estimation". We have categorized challenges as Technological related, Human Computer Interaction (HCI) related, Context-Awareness and Adaptation related, Privacy and Trust related and Recommendation methods related. The "$\surd$" symbol indicates that the particular challenge/issue applies to the corresponding area while the "$\times$" symbol indicates it does not (not applicable).

Table 3 depicts just how demanding and challenging ubiquitous computing really is: ubiquitous systems face most of the challenges described (all but the recommendation methods related), while ubiquitous recommenders face all of the challenges mentioned in the Table since they are categorized in both areas, the ubiquitous and the recommender systems.

Regarding recommender systems and CARS, these systems do not face any Technological challenges related to mobile and wireless technologies; however, few HCI related challenges exist, in particular those that have to do with interface issues such as limitations, usability, achieving user satisfaction, novel types of interaction, etc.

All Context-Awareness and Adaptation related issues concern ubiquitous systems and ubiquitous recommenders, while few of them also concern CARS, especially issues relevant to context modelling and manipulating the context to enhance the outcomes of the recommendation methods.

Privacy and Trust related challenges are mostly met in ubiquitous environments where the user, while being mobile, is being asked to provide sensitive data such as the location via wireless networks of limited security in order to benefit from the application at hand. Centralized, server side location tracking systems are not easily trusted by users. Recommender systems and CARS are less concerned with location privacy issues (although some risk exists since computers connected to the internet may be located via the IP and MAC addresses); however, at all times any user private data should not be shared without the user's consent.

Finally, recommendation methods related issues concern all recommendation related systems, with ubiquitous recommenders having the additional challenge of facilitating mobile users with less amount of time available and less patience: the recommendation methods to be used should be considerably fast and effective, more user friendly, demand the least user attention and should not interfere with the user's task, e.g. shopping.

In Table 4 we classify the most important systems in the relevant literature. Some of the referenced works in the CARS domain (Pre-, Post- filtering, Contextual Modelling) may have not been described in this work due to little relevance with the main topic under study; however, we classify them for completeness and for the interested reader. Although "Context-Driven querying and search" ubiquitous systems are related to ubiquitous recommenders, from the classification of Table 4, as well as Fig. 1 it is depicted that these two types of systems have fundamental differences. A "Context-Driven querying and search" ubiquitous system will focus on sensing the context and use the contextual information in order to filter a pool of items aiming to find the

Table 3 Challenges in ubiquitous recommender systems, ubiquitous systems, recommender systems and CARS

Challenges/issues	Ubiquitous RS	Ubiquitous systems	Recommender systems	CARS
Technological				
Power (energy concerns) and storage issues due to the small size of mobile devices	√	√	×	×
Wireless technologies issues.	√	√	×	×
Connectivity issues: how to connect and manage many small mobile devices	√	√	×	×
Networking issues: manage resources, scalability, devices must operate across different types of networks	√	√	×	×
HCI related				
User interfaces in small devices.	√	√	×	×
Devices: able to function without user attention	√	√	×	×
User friendly devices/interfaces: any ordinary user should be able to use them	√	√	√	√
Aim to improve user satisfaction	√	√	√	√
Usability: new novel types of interaction.	√	√	√	√
Any system adaptation should not interfere with the user's task	√	√	√	√
Track user intentions: the system should be able to infer what the user wants to do in order to assist her	√	√	√	√
The amount of input that is needed from users should be reduced by using context-awareness	√	√	×	√
Proactivity vs. transparency: a proactive system is prepared for an event and is able to act effectively and efficiently, but the system is not transparent to the user, and thus may become annoying to her	√	√	√	√
Presentation: how will annotations be presented—on the object or not, attached or detached from the location.	√	√	×	×
Anchoring: how annotations are being attached to places/objects	√	√	×	×
Authoring/editing of annotations: how, when, why?	√	√	×	×
Wearable computing: may be proved inconvenient for users	√	√	×	×

Table 3 continued

Challenges/issues	Ubiquitous RS	Ubiquitous systems	Recommender systems	CARS
Context-awareness and adaptation related				
Modelling the context: which method is more appropriate to use	√	√	×	√
Observing the context: automatically or manually?	√	√	×	×
Context sensing: when retrieving context data from various sources (e.g. sensors), how will data inconsistencies be resolved	√	√	×	×
Accuracy of contextual information should be well known during the design of ubiquitous systems	√	√	×	×
Storing the context: on server (privacy issues), on client or on both?	√	√	×	×
Systems should be more context-aware than just the location. A place is more than a location	√	√	×	√
How will the application modify its behaviour (be adapted) based on the context	√	√	×	×
Devices should not operate based only on their own context, but based on the context of the whole system	√	√	×	√
Context-awareness should be used to reduce the amount of input that is needed from users	√	√	×	√
Capture changes in the user's role by: (1) capturing the current context (i.e. the environment and the various circumstances, events and facts) and (2) user modelling (what possible role could a person play according to context)	√	√	×	√
Any components/devices should adapt based on the context without interfering with user's task: no user explicit interaction should be necessary. Components/devices will adapt separately and at the same time, while the user maintains a consistent view for the system/application.	√	√	×	×
Privacy and trust related				
Users do not want to give up their location privacy, unless there is a very significant reason to do so. They do not see the reason or the benefits, no motive	√	√	√	√

Table 3 continued

Challenges/issues	Ubiquitous RS	Ubiquitous systems	Recommender systems	CARS
Centralized (server side) location tracking systems are not easily trusted by users; users do not trust all their data to be stored somewhere remotely. Client side location tracking on the other hand is trusted by them (e.g. GPS) since none of their personal data are been stored elsewhere besides their own device	√	√	×	×
Sensitive user data should be distributed only with user's consent	√	√	√	√
Due to the wireless networks, ubiquitous applications are difficult to be trusted	√	√	×	×
User boundaries regarding privacy are often related to socio-cultural factors rather than physical ones	√	√	√	√
Mobile privacy issues are difficult to investigate since observing user behavior in real-time is difficult due to mobility, while it will also modify the user's spontaneous behavior. Moreover, questionnaire methods in mobile settings demand from mobile users to spend much time in answering questions at inconvenient times and environments	√	√	×	×
Recommendation methods related				
Building appropriate user models to store and use as much user information as possible (preferences, habits, rating data, etc.)	√	×	√	√
"New user" and "New item" problems exist when CF or content-based filtering is used	√	×	√	√
Users of ubiquitous RS have more limited mental and physical resources than users of traditional E-commerce recommenders due to their mobility (willing to spend less amount of time on the mobile device for receiving recommendations. This greatly affects the recommendation method to be used.)	√	×	×	×

Note that in this table CARS refer only to systems that provide "Recommendations via Contextual preference elicitation and estimation". Systems that provide "Recommendation via Context-Driven querying and search" are included in the table as "Ubiquitous Systems"

most appropriate one (or few) to recommend to its users. An example is a ubiquitous system that recommends nearby restaurants based on current user location (via GPS technology) and user preferences. The aforementioned recommendation functionality varies in comparison to the one classic recommender systems offer (and ubiquitous recommenders use): they use more sophisticated methods to opine about the utility of the items as recommendation candidates. For example, a classic recommender that uses the CF (Sect. 3.1) to recommend movies to users will measure the similarity of

Table 4 A classification of reviewed systems

Ubiquitous recommender systems	Ubiquitous approaches (context-driven querying and search)		CARS pre-, post- filtering and contextual modelling (contextual preference elicitation and estimation)
	Utilize many contextual parameters	Utilize few contextual parameters (early works and LBNS)	
[36,38–40,47]	[7,13,14,26,32, 45]	[5,10, 11,46, 48]	[2–4,6,8,12, 18,28,29,33, 34,52]

the active user to all other users by using appropriate algorithms in order to find the most similar ones to form the user's neighborhood, will then form the neighborhood of the user by choosing among the available techniques, and will finally extract the top-N recommendations: the movies that have the highest scores among the members of the active user's neighborhood.

The difference in the recommendation process between the two types of systems can be explained if we observe the initial datasets and the expected outcomes of each type of system. The "Context-Driven querying and search" ubiquitous system discussed above needs to select and recommend the top-N items (restaurants) among a list of relatively few items, while this selection will be based on few factors that are hard to acquire. The number of restaurants in the proximity of any user in any city is relatively small, while the factors that the system will be based on are contextual factors such as location and weather which are, on one hand, hard to acquire, meaning that the use of specialized devices and sensors will be needed, but on the other hand, the fact that these context factors are few, along with the fact that any similar users would be easily spotted through similar food preferences, suggest no need for complex recommendation algorithms such as the CF. Regarding the classic movie recommender discussed above, this system needs to choose and recommend the top-N items among a list of thousands of movies, while this selection will be based on many factors such as user preferences, movie genre, actors, movie length, time of day to watch the movie, company to watch the movie with, etc., that most of them are nevertheless easy to acquire. Moreover, similar users are very hard to be calculated because of the many user characteristics, e.g. preferences, that could be important for the recommendation process, as well as the great number of items (movies) and their characteristics. In this setting, CF could be proved an ideal solution.

5 Conclusions

Many works in the literature exist that focus on ubiquitous recommender systems [31,36–40,47]; however, to the best of our knowledge no literature work has attempted to analyze in detail and classify ubiquitous recommenders in terms of ubiquitous computing and recommender systems research so that their similarities, but more important their fundamental differences are presented. In particular, this paper discussed how ubiquitous recommender systems facilitate the user on location with recommendations via mobile devices in a similar way as ubiquitous systems do; however, we have shown that ubiquitous recommender systems focus mainly on using sophisticated recommendation methods met in traditional recommender systems rather than offering recommendations by querying and searching in repositories as ubiquitous systems do. Moreover, we have discussed research trends, important methods used and related work regarding ubiquitous recommender systems, having first provided introduction to ubiquitous computing, recommender systems and CARS. The work was finalized by classifying the aforementioned areas as well as systems under study.

As systems originated from two very different and solid research areas, ubiquitous recommender systems are not easily classified, as one may observe from Fig. 1. As systems in both ubiquitous and recommendation areas become more and more complex, more subclasses of systems appear that tend to combine characteristics, methods and solutions from both areas to provide novel applications. The ubiquitous recommender systems have managed to effectively combine state of the art solutions from the ubiquitous computing research area with well defined, effective and solid recommendation methods to provide a new user experience that may improve our lives, as well as stimulate researchers to continue this research trend.

5.1 UbiCARS

The following research question is, in our opinion, of particular importance: since ubiquitous recommender systems use sophisticated recommendation methods most often met in traditional recommender systems instead of context driven searches (see Fig. 1), could these recommendation methods become enhanced with the type of context modelling methods used in CARS-b systems, i.e. systems that recommend "via Contextual preference elicitation and estimation"? The aforementioned are depicted in Fig. 2. The new set of systems called *Ubiquitous Context-Aware Recommender Systems*, *UbiCARS*, would constitute a subset of "Ubiquitous Recommender Systems" and would utilize both types of context, the ubiquitous context (horizontal lines pattern in Fig. 2) to enable the provision of recommendations on location via mobile devices (e.g. identification of near-by products), as well as the CARS context (vertical lines pattern in Fig. 2) which refers to the multidimensional dataset used in CARS containing not only the dimensions of "users" and "ratings", but also the corresponding dimensions of the "context": Users $\times$ Items $\times$ Context $\rightarrow$ Ratings (see Sect. 3.2.2). The resulting context pattern for UbiCARS includes both horizontal and vertical lines, as depicted in Fig. 2. UbiCARS systems would be ubiquitous and context-aware in terms of sensing information from the environment and react on it, as well as context-aware in terms of considering the context in the recommendation process by using multidimensional contextual datasets. Note that in Fig. 2 the UbiCARS set differs from the CARS-b set in two points: (1) UbiCARS lies under Ubiquitous Computing and (2) it uses the ubiquitous context as well. The two sets have been positioned adjacent to depict their similarities: using the CARS context and sophisticated recommendation methods from traditional recommender systems. Considering the above discussion, we may assume that such systems would be able to provide better recommendations

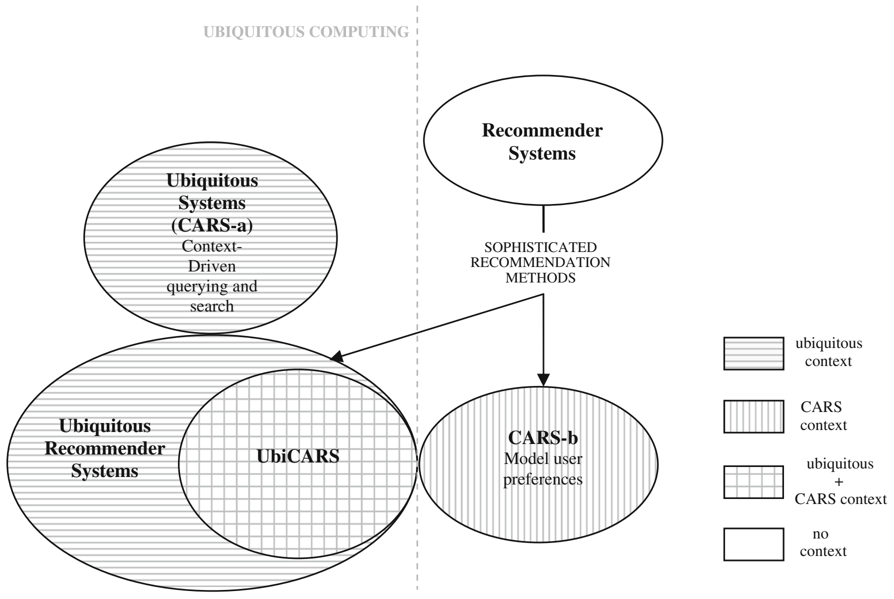
UBIQUITOUS COMPUTING
Recommender Systems
Ubiquitous Systems (CARS-a) Context-Driven querying and search
SOPHISTICATED RECOMMENDATION METHODS
Ubiquitous Recommender Systems
UbiCARS
CARS-b Model user preferences
ubiquitous context
CARS context
ubiquitous + CARS context
no context

Fig. 2 Ubiquitous context-aware recommender systems—UbiCARS

than ubiquitous recommender systems that currently do not consider the context in the actual recommendation process.

As a theoretic example of a UbiCARS system, we propose a ubiquitous recommender system for products that is able to recommend items in the user's proximity via the user's mobile device while she is shopping (utilizing the ubiquitous context), but is also able to consider in its advanced recommendation algorithm enriched context information not used by common ubiquitous recommender systems (CARS context). The advanced recommendation algorithm and enriched context information will be derived by CARS research, and more particular by CARS that provide "recommendations via Contextual preference elicitation and estimation". Regarding the ubiquitous context, the UbiCARS for products will deploy NFC (or other similar technologies) to read RFID tags and identify products in the user's proximity, as well as product information that can be useful (e.g. price and expiring information). Regarding the advanced recommendation algorithm and enriched context information, the system will use information on purchases the user has done in the past such as: the items she has bought in each visit to the store, the day and time of each purchase and with who the user was with (provided that such information is available). The UbiCARS for products will utilize the reduction-based approach [2], which is a Contextual Pre-filtering method aiming to produce a 2D dataset from the MD dataset (suppose it is 5D: user, item, day of purchase, time of purchase, the person who was with the user) and use it as input in any of the classical pre-existing 2D recommendation methods, e.g. CF (see also Sect. 3.2.2). Let's consider a user who is at a store on a Friday afternoon with her boyfriend and is looking to by a certain type of tomato juice, the only type that her boyfriend can consume due to allergies. The reduction-based approach will assign specific values to the contextual parameters day of purchase, time of purchase and company as follows: day = "Friday", time = 5–8 p.m." and company = "boyfriend" and select only the dataset records that include day = "Friday", time = 5–8 p.m." and company = "boyfriend". By this, the reduction-based approach reduces the dimensions of the initial 5D dataset to two, as well as excludes any irrelevant data from participating in the recommendation process (e.g. any records regarding other days are not as relevant as the records regarding Fridays). The above process takes into account enhanced context information, enabling the system to provide more customized to the particular circumstances recommendations; e.g. the presence of her boyfriend can guide the system to recommend only tomato juice brands that are suitable for him, since the past records that satisfy the context: company = "boyfriend" will include only such brands, provided that such purchases exist. In addition to the above, the system via the NFC method will identify all products the user is currently observing, i.e. canned tomato juices, registering important information such as price and expiry date. Alternatively, other recommendation strategies from CARS research similar to the reduction based approach can be used, such as the one by Baltrunas and Ricci [4], who instead of neglecting rating data not relevant to the current context (as the reduction-based approach), they split the data into two sets based on the contextual parameters. Moreover, the Pre-filtering approach by Domingues and colleagues [18] can be used, whose approach focuses on inserting the context in the recommendation input dataset by adding it as virtual items: each value of a contextual parameter is inputted in the dataset matrix as a new row of data. The values of the contextual parameters are therefore perceived as items and their similarities with regular items are calculated. The technique does not alter the actual recommendation algorithm to be used; it only changes the input dataset (Pre-filtering). Similarly, many algorithms derived from CARS research may be used, provided that appropriate enhanced context information is available for the UbiCARS system to utilize. The resulted recommendations (products) from the above process can then be combined with the results from the NFC method in order to recommend only appropriate products that are also in the user's proximity, are cheaper and have extended expire date.

Based on the above discussion, we argue that the UbiCARS system for products will be able to provide better recommendations than common ubiquitous recommender systems that currently do not consider the CARS context in their recommendation process and do not use context modelling methods derived from the "Contextual preference elicitation and estimation" approach (such as the reduction-based approach). The reduction-based approach used above constitutes only a rather simple example of how recommendations can be enhanced by enriched context information; more specialized and advanced recommendation algorithms can be used based on the Multidimensional Contextual Modelling approach (see Sect. 3.2.2), while also new multidimensional techniques can be developed especially for UbiCARS. However, UbiCARS systems would also face additional challenges, as these are described in this work: ubiquitous recommender systems challenges as well as CARS related challenges. Nevertheless, it would be very interesting to witness the realization of such systems and their comparison with existing ones, in order to discover whether UbiCARS indeed have a significant advantage due to the enhanced context-aware functionality they provide.

References

1. Adomavicius G, Sankaranarayanan R, Sen S, Tuzhilin A (2005) Incorporating contextual information in recommender systems using a multidimensional approach. ACM Trans Inf Syst (TOIS) 23:103–145
2. Adomavicius G, Tuzhilin A (2008) Context-aware recommender systems. In: Proc. RecSys 2008, the 2008 ACM conference on recommender systems, pp 335–336
3. Baltrunas L, Amatriain X (2009) Towards time-dependant recommendation based on implicit feedback, workshop on context-aware recommender systems, CARS 2009. In: ACM RecSys, vol 2009
4. Baltrunas L, Ricci F (2009) Context-dependent items generation in collaborative filtering, workshop on context-aware recommender systems, CARS 2009. In: ACM RecSys, vol 2009
5. Bilandzic M, Foth M, Luca AD (2008) CityFlocks: designing social navigation for urban mobile information systems. In: Proc. of the 7th ACM conference on designing interactive systems. ACM, Cape Town, pp 174–183
6. Bogers T (2010) Movie Recommendation using random walks over the contextual graph. In: Proc. CARS 2010, the 2nd workshop on context-aware recommender systems
7. Böhmer M, Bauer G, Krüger A (2010) Exploring the design space of context-aware recommender systems that suggest mobile applications. In: Proc. CARS 2010, the 2nd workshop on context-aware recommender systems
8. Bourke S, McCarthy K, Smyth B (2010) The social camera: recommending photo composition using contextual features. In: Proc. CARS 2010, the 2nd workshop on context-aware recommender systems
9. Burke R (2002) Hybrid recommender systems: survey and experiments. User Model User Adapt Interact 12(4):331–370
10. Burrell J, Gay GK (2001) Collectively defining context in a mobile, networked computing environment, CHI ’01 extended abstracts on human factors in computing systems. ACM, Seattle
11. Burrell J, Gay GK (2002) E-Graffiti: evaluating real-world use of a context-aware system, interacting with computers
12. Cantador I, Castells P (2009) Semantic contextualisation in a news recommender system, workshop on context-aware recommender systems, CARS 2009. In: ACM RecSys, vol 2009
13. Carolis BD, Mazzotta I, Novielli N, Silvestri V (2009) Using common sense in providing personalized recommendations in the tourism domain, workshop on context-aware recommender systems, CARS 2009. In: ACM RecSys, vol 2009
14. Cena F, Console L, Gena C, Goy A, Levi G, Modeo S, Torre I (2006) Integrating heterogeneous adaptation techniques to build a flexible and usable mobile tourist guide. AI Commun 19:369–384
15. Davies N, Gellersen H (2002) Beyond prototypes: challenges in deploying ubiquitous systems. IEEE Pervasive Comput 1:26–35
16. Deshpande M, Karypis G (2004) Item-based top-n recommendation algorithms. ACM Trans Inf Syst 22:143–177
17. Dey AK, Abowd GD, Abowd D (2001) A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications, Hum Comput Interact 16:97–166
18. Domingues MA, Jorge AM, Soares C (2009) Using contextual information as virtual items on top-N recommender systems, workshop on context-aware recommender systems, CARS 2009. In: ACM RecSys, vol 2009
19. Domnitcheva S (2001) Location modeling: state of the art and challenges. In: Proc. the workshop on location modeling for ubiquitous computing, pp 13–19
20. Hansen FA (2006) Ubiquitous annotation systems: technologies and challenges. In: Proc. ACM, the seventeenth conference on hypertext and hypermedia, pp 121–132
21. Henricksen K, Indulska J, Rakotonirainy A (2001) Infrastructure for pervasive computing: challenges. In: Proc. workshop on pervasive computing informatik, vol 01, pp 214–222
22. Hinze A, Buchanan G (2005) Context-awareness in mobile tourist information systems: challenges for user interaction. In: Proc. international workshop on context in mobile HCI at the conference for 7th international conference on human computer interaction with mobile devices and services
23. Jannach D, Zanker M, Felfernig A, Friedrich G (2010) Recommender systems: an introduction. Cambridge University Press, Cambridge, pp 289–298
24. Karypis G (2000) Evaluation of item-based top-N recommendation algorithms. In: Proc. the 10th international conference on information and, knowledge management, pp 247–254
25. Kjeldskov J, Skov MB (2007) Exploring context-awareness for ubiquitous computing in the healthcare domain. Person Ubiquitous Comput 11:549–562

26. Kourouthanassis P, Spinellis D, Roussos G, Giaglis GM (2002) Intelligent cokes and diapers: Mygrocer ubiquitous computing environment. In: Proc. the 1st international mobile business conference
27. Lenders V, Koukoumidis E, Zhang P, Martonosi M (2008) Location-based trust for mobile user-generated content: applications, challenges and implementations. In: Proc. the 9th workshop on mobile computing systems and applications, ACM, pp 60–64
28. Loizou A, Dasmahapatra S (2006) Recommender systems for the semantic web. In: ECAI 2006 recommender systems workshop
29. Lombardi S, Anand S, Gorgoglione M (2009) Context and customer behavior in recommendation, workshop on context-aware recommender systems, CARS 2009. In: ACM RecSys, vol 2009
30. Mancini C, Thomas K, Rogers Y, Price BA, Jedrzejczyk L, Bandara AK, Joinson AN, Nuseibeh B (2009) From spaces to places: emerging contexts in mobile privacy. In: Proceedings of the 11th international conference on ubiquitous computing, New York, vol 2009, pp 1–10
31. McDonald DW (2003) Ubiquitous recommendation systems, computer, vol 36, no 10, p 111
32. Miller BN, Albert I, Lam SK, Konstan JA, Riedl J (2003) MovieLens unplugged: experiences with a recommender system on four mobile devices. In: Proc. the 8th international conference on intelligent user interfaces
33. Oku K, Nakajima S, Miyazaki J, Uemura S (2006) Context-aware SVM for context-dependent information recommendation. In: IEEE international conference on mobile data management, p 109
34. Oku K, Nakajima S, Miyazaki J, Uemura S, Kato H, Hattori F (2010) A recommendation system considering users' past/current/future contexts. In: Proc. CARS 2010, the 2nd workshop on context-aware recommender systems
35. Persson P, Espinoza F, Fagerberg P, Sandin A, Cöster R (2002) GeoNotes: a location-based information system for public spaces. In: Hook K, Benyon D, Munro A (eds) Readings in social navigation of information space, pp 151–173
36. Reischach FV, Guinard D, Michahelles F, Fleisch E (2009) A mobile product recommendation system interacting with tagged products. In: Proc. the 2009 IEEE international conference on pervasive computing and, communications, pp 1–6
37. Reischach FV, Michahelles F, Schmidt A (2009) The design space of ubiquitous product recommendation systems. In: Proc. the 8th international conference on mobile and ubiquitous multimedia, pp 1–10
38. Reischach FV, Michahelles F (2008) Apriori: a ubiquitous product rating system. In: PERMID '08, workshop on pervasive mobile interaction devices at pervasive conference
39. Resatsch F, Karpischek S, Sandner U, Hamacher S (2007) Mobile sales assistant: NFC for retailers. In: Proc. mobile HCI '07, the 9th international conference on human computer interaction with mobile devices and services
40. Resatsch F, Sandner U, Leimeister JM, Krcmar H (2008) Do point of sale RFID-based information services make a difference? Analyzing consumer perceptions for designing smart product information services in retail business. Electron Markets 18(3):216–231
41. Sarwar B, Karypis G, Konstan J, Riedl J (2000) Analysis of recommendation algorithms for e-commerce. In: Proc. the 2nd ACM conference on electronic commerce, pp 158–167
42. Sarwar B, Karypis G, Konstan J, Riedl J (2001) Item-based collaborative filtering recommendation algorithms. In: Proc. the 10th international conference on World Wide Web, pp 285–295
43. Satyanarayanan M (2001) Pervasive computing: vision and challenges. IEEE Person Commun 8:10–17
44. Schilit BN, Lamarca A, Borriello G, Griswold WG, Mcdonald D, Lazowska E, Hong J, Iverson V (2003) Challenge: ubiquitous location-aware computing and the Place Lab initiative. In: Proc. WMASH '03, the 1st ACM international workshop on wireless mobile applications and services on WLAN hotspots, pp 29–35
45. Setten MV, Pokraev S, Koolwaaij J (2004) Context-aware recommendations in the mobile tourist application compass. In: Nejdl W, De Bra P (eds) Adaptive hypermedia, pp 235–244
46. Streefkerk JW, Esch-Bussemakers MPV, Neerincx MA (2008) Field evaluation of a mobile location-based notification system for police officers. In: Proc. the 10th international conference on Human computer interaction with mobile devices and services, ACM, pp 101–108
47. Takeuchi Y, Sugimoto M (2007) A user-adaptive city guide system with an unobtrusive navigation interface. Person Ubiquitous Comput 13(2):119–132
48. Tungare M, Burbey I, Pérez-Quiñones MA (2006) Evaluation of a location-linked notes system. In: Proc. the 44th annual Southeast regional conference, ACM, pp 494–499
49. Waller V, Johnston RB (2009) Making ubiquitous computing available. Commun ACM 52:127–130

50. Want R, Pering T (2005) System challenges for ubiquitous and pervasive computing. In: Proc. the 27th international conference on Software engineering, ACM, pp 9–14
51. Weiser M (1991) The computer for the 21st century. Scientific American
52. Yu Z, Zhou X, Zhang D, Chin C, Wang X, Men J (2006) Supporting context-aware media recommendations for smart phones. IEEE Pervasive Comput 5(3):68–75