Context-Aware Recommender Systems: A Service-Oriented Approach

Sofiane Abbar
PRiSM Lab, Versailles university
45 avenue des Etats-Unis
78035, Versailles, France
sofa@prism.uvsq.fr

Mokrane Bouzeghoub
PRiSM Laboratory, Versailles university
45 avenue des Etats-Unis
78035, Versailles, France
mok@prism.uvsq.fr

Stéphane Lopez
PRiSM Lab, Versailles university
45 avenue des Etats-Unis
78035, Versailles, France
hal@prism.uvsq.fr

ABSTRACT

Recommender systems are efficient tools that overcome the information overload problem by providing users with the most relevant contents. This is generally done through user's preferences/ratings acquired from log files of his former sessions. Besides these preferences, taking into account the interaction context of the user will improve the relevancy of recommendation process. In this paper, we propose a context-aware recommender system based on both user profile and context. The approach we present is based on a previous work on data personalization which leads to the definition of a Personalized Access Model that provides a set of personalization services. We show how these services can be deployed in order to provide advanced context-aware recommender systems.

1. INTRODUCTION

The last decade met a remarkable proliferation of P2P networks, PDMS, semantic web, communitarian websites, electronic stores, etc. resulting in an overload of available information. Current information systems deal with a huge amount of content, and deliver in consequence a high number of results in response to user queries. Thus, users are not able to distinguish relevant contents from secondary ones.

Recommender systems (RS) are efficient tools designed to overcome the information overload problem by providing users with the most relevant content [8]. Recommendations are computed by predicting user's ratings on some contents. Rating predictions are usually based on a user profiling model that summarizes former user's behaviour.

The importance of RS is now well established. Netflix organizes a contest [1] in which one million dollar is offered for any better recommendation engine. This contest shows, in one hand, the importance that industrials give to RS, and in another hand that better recommendations worth a lot. Two important questions arise from the Netflix contest: (i) What is a better RS? And (ii) How can a system provide the best recommendations?

In our vision, a better RS is the one which delivers recommendations that best match with users' preferences, needs and hopes at the right moment, in the right place and on the right media. This can't be achieved without designing a RS that takes into account all information and parameters that influence user's ratings. These information may concern demographic data, preferences about user's domain of interest, quality and delivery requirements as well as the time of the interaction, the location, the media, the cognitive status of the user and his availability. This knowledge is organized into the two concepts of user profile and context. The user profile groups information that characterizes the user himself while the context encompasses a set of features which describe the environment within which the user interacts with the system.

We claim that taking into account both profiles and contexts in a recommendation process benefits to any RS for many reasons: (*i.*) Users' preferences/ratings change according to their contexts [5, 12]. (*ii.*) The additive nature of traditional RS [6] does not consider multiple ratings of the same content. (*iii.*) RS may fail in providing some valuable recommendations as their similarity distance is uniformly applied to user preferences without analyzing the discrepancies introduced by the context.

In [2], we proposed a Personalized Access Model (PAM), a framework that provides a set of personalization concepts and services, generic enough to be applied to a large variety of applications. Based on the PAM, and following the efforts started by Adomavicius et al. [3, 4] and Anand et al. [6], we propose a Context-Aware Recommender System (CARS) which is based on both user profiles and contexts. In our approach the same user who interacts from different contexts is provided with different recommendations.

Our main contributions include the following: (*i.*) We present a general architecture for context-aware RS based on a set of personalization services. (*ii.*) We extend the PAM with a new context learning service that enables concrete construction of users' contexts from their log files. (*iii.*) The contextualization service proposed in [1] is improved by combining both *support* and *confidence* of ratings within a given context instead of considering their frequencies only. Notice that this paper reports a conceptualization effort for integrating context into RS. The evaluation of CARS approach is under study.

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

VLDB '09, August 24-28, 2009, Lyon, France

This paper is organized as follow: Section 2 presents a high level architecture of CARS and poses the main requirements that CARS should satisfy. Section 3 recalls concepts and services provided by the PAM. Sections 4 and 5 focus on the services applied to CARS. Section 6 provides a global view on CARS deployment using PAM services. Section 7 summarizes related works. Section 8 concludes the paper with further research.

2. REQUIREMENTS FOR CARS

We focus on RS which combine content-based techniques and Collabotrative Filtering (CF). The content-based approach permits to learn users' profiles by analyzing content descriptors. Resulting profiles are, then, matched and compared to determine similar users in order to make collaborative recommendations. The CF approach allows exploiting the ratings given by the Top K neighbors of the active user in order to derive the missing rating by aggregation function. Figure 1 gives a flavour of the CARS global architecture.

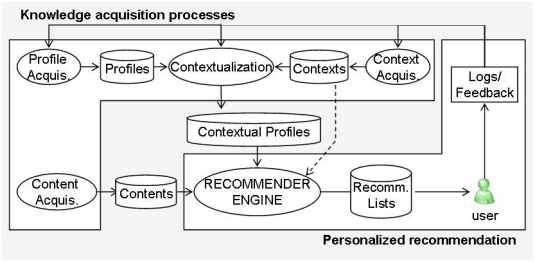

Figure 1: Global architecture of context-aware RS

This architecture is composed of: the left-upper block which concerns knowledge acquisition and the right-bottom block which concerns personalized recommendation.

Knowledge acquisition processes: This block is responsible for the acquisition and the management of knowledge a CARS needs to process recommendations. This knowledge is grouped into three entities: user profiles, content descriptors, and contexts. We focus in this paper on: (i) context acquisition from log files, and (ii) discovery of relationships between these contexts and user profiles elements (called contextual profiles).

Personalized recommendation: This block encompasses actual RS operations. The recommender engine (RE) takes as inputs (i) contextual profiles generated in the knowledge acquisition block and (ii) the current context of the target user to compute a list of contextual recommendations. The behaviour of the target user when consuming these recommendations is stored in log files on which acquisition processes are based to update profiles, contexts and their mappings; hence, closing the cycle between knowledge acquisition and their exploitation.

The design of such CARS is driven by the following requirements:

1. Distinguish profiles from contexts for a well-understood definition and evolution of both entities.
2. Enable contexts discovery through a concrete approach and a widely available data (standard log files).
3. Consider both profiles and contexts into recommendation such that a user interacting from different contexts would be provided with appropriate recommendations.

As a consequence of the separation of profile and context, applications may be only aware on profiles or contexts or on both. The next section details a service-oriented approach for designing CARS that satisfy above requirements.

3. OVERVIEW OF PAM CONCEPTS AND SERVICES

Our CARS proposal is based on a former work [2] in which we defined a Personalized Access Model (PAM) that provides a set of personalization services. The PAM aims at providing a generic set of concepts and services which can be deployed over a given architecture to make applications aware of users' profiles and contexts.

The components of the PAM are built around profile and context meta models which are adaptable to a wide range of applications and which are open to integrate specific knowledge not included initially. Profile and context meta models are organized into dimensions and sub dimensions which are described by (attribute, value) couples.

3.1 Main Concepts

This section provides an informal definition of the concepts that will be used in the following sections.

User profile. Provides an extensive definition of the preferences that a user has in a given domain of interest. It is a set of (attribute,value) couples rated by the user or the RS on the basis of user's actions.

Active User. In RS, an active user is the one for whom recommendations are calculated.

Context. Set of features characterizing the environment within which users interact with RS.

Active context. It is the context within which the active user interacts with RS. It is also called current context.

Contextualized profile. It's a set of mappings that relate a subset of profile preferences to the context in which they are defined.

Operational profile. It's a runtime binding of a contextualized profile to the current context of the user.

3.2 PAM Services

The PAM provides several services which have been progressively extended, following application requirements [2]. Among these services, we distinguish offline (Contextualization, Context discovery) and online services (Binding, Matching). The former set is used at the design time of a personalized application, while the latter is used at the execution time. We list hereafter those which are worth to be used in personalized RS.

Context discovery service. Discovers, for each user, regular contexts within which he interacts with RS (e.g. Holiday context, Office context).

Contextualization Service. Aims at identifying mappings between user ratings and contexts in which these ratings were defined. Mappings are of the form (R_i, c_j, s_{ij}), where $R_i = Item_i \times r_i$, and s_{ij} represents the importance of the rating r_i over $item_i$ within the context c_j.

Binding Service. Exploits the mappings generated by the contextualization service to derive the operational profile.

Matching Services Allow capturing the semantic similarity between two concepts. The similarity value takes into account the structure of compared concepts, user ratings and

ontologies with which items can be described. We distinguish four matching services [2]: profile/profile, context/context, item/item, and profile/item. Due to space limitation, matching services are not devellopped in this paper.

4. OFFLINE SERVICES

This section details the two services that acquire the necessary knowledge to efficiently run the CARS. These services are an answer to the first two requirements imposed in section 2. For each service, we give its definition, its deployment on CARS, and details on its operational semantics.

4.1 Context Discovery Service

4.1.1 Service definition

The goal of the context discovery service is to learn from user log files, the most recurrent contexts (situations) within which the user interacts in general.

$$C : Context\_Discovery(\alpha, L) \quad (1)$$

The context discovery service takes as input the user log file (L) tracked from former recommendations and returns a set of regular contexts C within which a user, whose profile is P_u, interacts with the personalized application.

4.1.2 Service deployment on CARS

The approach we propose to discover contexts is based on the analysis of logs to capture regular contexts. Hereafter, a motivating example is given to understand the approach.

Example. Assume that a user, John, interacts frequently with RS. Then, thanks to information contained in his log files, the system can capture that John's interests (activities) change according to the IP address, the date and time, the device, etc. This can be pointed by a lookup to his log file from where two contexts can be derived: office context during working days and home context during weekends.

John / IP 192.168.53.25 /May 4 17h:00 /winXP,En,Firefox5.0
John / IP 192.168.53.25 /May 5 17h:30 /winXP,En,Firefox5.0
John / IP 192.168.53.25 /May 4 17h:33 /winXP,En,Firefox5.0
...
John / IP 192.168.70.1 /May 9 12h:00 /MacOS X,En,Safari
John / IP 192.168.70.1 /May 9 20h:00 /MacOS X,En,Safari

Obviously, the content to be recommended change according to John's context (e.g. scientific papers for office context and movies for home context). Context discovery from logs assumes the manipulation of well structured log file. Hopefully, the analysis of the most known log file formats such as Apache CLF, IIS LF and W3C LF [11] reveals that most of them already contain contextual information (fields) that correspond to the context meta model proposed in [2]. Among these fields there are, for example, the IP Address, the DATE of the request, and the USER-AGENT (browser, OS, ...).

4.1.3 Log File Format

The log file format we used is based on the W3C log file format [11] which is enough extensible (thanks to ten W3C General Purpose Fields) to adapt it to various applications. field: $Date\ Time\ c-ip\ c-dns\ c-auth-id\ cs-method\ cs-uri-stem\ cs-uri-query\ cs(UserAgent)\ sessionId\ c-Action\ c-Device$. Notice that this log file format defines two new fields only which are $c-action$, and $c-device$. The first one is well known in application server log files, it contains usually the id of the action (e.g. Buy) that user applies to a given content. The second field informs about the device used (e.g. Laptop, a mobile phone, a remote control, etc.).

Id	IP	Date	Time	Agent	Device	Content	...
1	x	y	t	a	d	Item1	...
2	e	r	f	d	c	Item2	...
3	r	g	f	q	s	Item1	...

Figure 2: Log file

Among the log file format fields, five are contextual:

$Date$: the domain of date is organized into a hierarchy, in such a manner that we can know if it is a working day or a weekend, holiday or not, etc.

$Time$: a day is partitioned into periods which are organized into a hierarchy. The time can inform about the day period: day, night, morning, afternoon, etc.

$c-ip$: The IP address in used to localize the user, the localities are organized into a hierarchy of continent, region, country, city, town, etc.

$cs(UserAgent)$: gives information about the user Browser and OS (operating system) further than their languages.

$c-Device$: characterizes the used device for the user interaction, devices can be segmented into a hierarchy.

Conjunction of these attribute represent contexts within which users interacts with the RS. We propose to group these contexts into clusters, each representing a regular context or situation such as home, labs, etc.

4.1.4 Context Discovery Algorithm

Log files are transformed and uploaded into a database relation having the schema of figure 2. Based on this table, contexts are discovered by mining the five contextual attributes given above.

Let a context tuple be a conjunction of the five contextual attributes which corresponds to one row (see figure 2).

The idea is to cluster the set of all tuples into a finite set of clusters, each cluster representing a particular regular context. Thus, we start by applying the *Agglomerative Hierarchical Clustering (AHC)* [10] in order to estimate the number of clusters k, then, we apply the *k-means* algorithm with the k center of resulting clusters in AHC algorithm instead of choosing arbitrarily the initial k centers.

Algorithm 1 starts by extracting context instances from the user log file. In line 3, AHC is applied on these instances in order to compute the initial k clusters. Then, from lines 4 to 13, *k-means* is applied as an iterative relocation algorithm in order to improve the initial clustering obtained with AHC. The algorithm iterates until Squared Error $\mathcal{E}$ achieves its minimal value.

4.2 Contextualization Service

4.2.1 Service definition

The main idea of the contextualization is to check whether there are correlations between the user profile elements and the user feedback within a given context.

$$M(P_u, C) : Contextualization(P_u, H) \quad (2)$$

Contextualization process takes as input a user profile P_u and the user history H corresponding to the user feedback in contexts C discovered previously. It returns a set of contextual mappings M representing dependencies between el-

Algorithm 1 Automatic Contexts Discovery

Input: the user log file $\mathcal{L}$, the threshold α.
Output: the set of user contexts $\mathcal{C} = \{c_1, \ldots, c_m\}$,

$\mathcal{C} \leftarrow \emptyset$, $\mathcal{E} := \infty$
$\mathcal{CI} \leftarrow CONTEXT(\mathcal{L})$ {extract context instances}
$\mathcal{C} \leftarrow AHC(\mathcal{CI}, \alpha)$ {apply Agglomerative Hierarchical Clustering on $\mathcal{CI}$. α a stopping threshold}
for all cluster $c_i \in \mathcal{C}$ **do**
  update its mean $m_i := NewMean(c_i)$
repeat
  $\mathcal{E}_{old} := \mathcal{E}$
  for all context instance $t_j \in \mathcal{CI}$ **do**
    assign t_j to its closest cluster $c^* \in \mathcal{C}$ such that $\forall c_i \in \mathcal{C}, MATCH(t_j, c_i) \leq MATCH(t_j, c^*)$
  for all cluster $c_i \in \mathcal{C}$ **do**
    update its mean $m_i \leftarrow NewMean(c_i)$
  compute $\mathcal{E} := \sum_{i=1}^{k} \sum_{t \in c_i} |MATCH(t, m_i)|^2$
until $\mathcal{E} \geq \mathcal{E}_{old}$
return $\mathcal{C}$

ements in P_u and contexts in C.

4.2.2 Contextual profile

In traditional RS, a user profile is a set of ratings $P_u = \{R_1, \ldots, R_n\}$, where each rating R_i is composed of a predicate pr_i and a rate (weight) r_i, i.e. $R_i = (pr_i, r_i)$. The rate r_i is a real number expressing the importance of the predicate pr_i for the user. pr_i is a triplet $< concept, operator, value >$, e.g. *Genre = 'Drama'*. *Concepts* may be *items* (content) that a user consumed, $features$ of these items or both of them.

Based on this profile, a contextualized user profile CP_u is defined as a set M of contextual mappings which relates user ratings to a set of contexts C:

$$CP_u = \{m_{ij}\,(R_i, c_j, s_{ij})\,|R_i \in P_u, c_j \in C, s_{ij} \in [-1, 1]\}$$

Each contextual mapping m_{ij} is defined by a rating R_i, a context c_j and a score s_{ij}. The score s_{ij} is a real number expressing the importance of taking into account the Rating R_i when the user interacts from the context c_j. Hereafter, we describe the way a contextual user profile is constructed using log files.

4.2.3 Service deployment on CARS

The user history H on which the contextualization service is based, is obtained by transforming the user log file L (figure 2). A sample of this history is presented in figure 3.

Id	Context- cluster	Content	Feature	C-Action	...
1	c_1	Item1	f1	+	...
2	c_1	Item2	f1	-	...
3	c_2	Item1	f2	-	...

Figure 3: a sample of user history H

User behaviour is captured in a relational table where each row is of the form: `<id, Context_id, Content_id, Feature, Action_type>` expressing that a user consumed a content (item) having some features (predicates) in a specific way `(Action_type)` within a given context `(context_id)`. `Action_type` specifies whether the action applied by a user when consuming the content is positive (e.g. Buy) or negative (e.g. Ignore). `Content_id` and `Features` are obtained by transforming the URI of the consumed content (which is a log file field) into more significant information (e.g. http://www.imdb.com/title/tt0448157/ becomes Title: *Hancock*, Genre:*Action, Comedy, Crime*). `Context_id` specifies the context cluster within which the content was consumed; it is computed by the context discovery service.

4.2.4 Contextualization Algorithm

The algorithm of contextual profile construction captures for each user rating (more precisely, predicate), its importance in each user context. Notice that, in each context, user activities may be of two kinds: positive (belongs to liked contents) and/or negative (relative to disliked contents), according to the type of actions a user applied on contents. Therefore the importance of each predicate must be captured in both positive and negative activities. As presented in [1], the importance of a predicate pr_i within a context c_j is captured by computing its frequency in this context. However, we claim that the frequency does not reflect the real importance of a predicate for one user. Thus, we propose to model the importance of each predicate pr_i for the positive (resp. negative) activity within a given context as an association rules of the form $\langle pr_i \rightarrow + \rangle$ (resp. $\langle pr_i \rightarrow - \rangle$). Then, the importance of pr_i is obtained by combining both the *support* (sp) and the *confidence* (cf) of its corresponding rules.

Many approaches can be used as merging function, we mention hereafter one possible way that consists in measuring the conviction of the rule.

$$conv(pr_i \rightarrow *) = \frac{1 - sp(*)}{1 - cf(pr_i \rightarrow *)} \quad (3)$$

Where $*$ takes one value among $\{+, -\}$ at a time.

Algorithm 2 shows the way contextual profiles are constructed based user's interaction histories.

Algorithm 2 Automatic contextual mappings discovery

Input: the initial user profile $P_u = \{p_1, \ldots, p_n\}$, the user behavior $\mathcal{H}$.
Output: the contextualized user profile CP_u

$CP_u \leftarrow \emptyset$, $cf := 0, sp := 0$
$\mathcal{C} \leftarrow CONTEXT(\mathcal{H})$
for all $R_i \in P_u$ **do**
  for all $c_j \in \mathcal{C}$ **do**
    compute $cf^+ := confidence(pr_i \rightarrow +, c_j, \mathcal{H}')$
    compute $sp^+ := support(pr_i \rightarrow +, c_j, \mathcal{H}')$
    compute $s_{ij}^+ := merge(cf^+, sp^+)$
    if $s_{ij}^+ \geq \gamma$ **then**
      $CP_u \leftarrow CP_u \cup (R_i, c_j, s_{ij}^+)$
    compute $cf^- := confidence(pr_i \rightarrow -, c_j, \mathcal{H}')$
    compute $sp^- := support(pr_i \rightarrow -, c_j, \mathcal{H}')$
    compute $s_{ij}^- := merge(cf^-, sp^-)$
    if $s_{ij}^- \geq \gamma$ **then**
      $CP_u \leftarrow CP_u \cup (R_i, c_j, -s_{ij}^-)$
return CP_u

The contextual profile constructed in algorithm 2 allows positioning each profile rating in each context through a score. The next section discusses the manner this score is exploited to make contextual recommendations.

5. ONLINE SERVICE: BINDING

This section details a runtime service that constitutes an answer to the last requirement imposed in section 2.

5.1 Service definition

The binding consists in identifying, at a runtime, user profile parts which are related to a given context.

$$OP_u : Binding(CP_u, c_i) \quad (4)$$

The profile binding takes as input the contextual user profile CP_u and the active context of the user c_i. It returns an operational profile OP_u which contains only profile elements which have to be considered by applications within c_i.

5.2 Service deployment on CARS

The Operational Profile is produced by filtering user ratings which aren't relevant to the active context, and by combining the remaining user ratings with contextual mapping scores.

Operational profile is always related to one context c_a (active context), it can be defined as being a set of contextual ratings (and not mappings): $OP_u = \{CR_{11}, \ldots, CR_{nk}\}$. Each contextual rating is of the form: $CR =< pr_i, w >$, and is derived from a contextual mapping $m_{ia}(R_i, c_a, s_{ia})$ with $R_i = pr_i \times r_i$. Notice that $w = aggregate(r_i, s_{ia})$ is the weight (contextual rating) of the predicate pr_i (e.g. $item_i$) within the context c_a.

The two reals r_i and s_{ia} have particular semantics. In fact, r_i represents the absolute importance of a given predicate pr_i. In other words, it is the preference that a user has on the predicate independently of contexts. However, a daily analysis of the user behaviour can reveal that the importance of the predicate pr_i on which r_i was expressed changes w.r.t contexts, leading to the definition of a contextual importance s_{ia}.

$$Aggregate(r_i, s_{ia}) = s_{ia} \times r_i \quad (5)$$

These two numeric can be aggregated with different manners, one possibility is given in equation 5 were r_i is multiplied by the contextual score s_{ia}. In this way, more the importance of the contextual score of a given pr_i is high, more its related rating r_i is preserved.

6. TOWARD A SERVICE-BASED CARS

Once contexts are learned and contextual profiles are constructed for each user in the RS, we show how runtime services (Binding and Matching) can be deployed in CARS to provide users with contextual recommendations. The next sections present the CARS algorithm, and a simple example scenario of its usage.

6.1 Contextual Top k neighbors

The key idea behind the CARS we propose is to base the Top k neighbors detection only on the profile parts relevant to a given context (i.e. operational profile) instead of on whole user profile. Top k neighbors of the active user (u_a) are the k most similar users to him in term of their profiles. This means that profiles of all users are filtered and adapted to the context of the active user before comparing them. Algorithm 3 details the process of computing the Top k contextual neighbors.

The algorithm is explained through the simple scenario given bellow.

Algorithm 3 Contextual Top K Neighbors

Input: $CP_{ua} = \{m_{11}, \ldots, m_{nm}\}$: Contextual active user profile, $\mathcal{U}$: set of all users, c_a: the active context, $\mathcal{C}$: the set of all contexts, $Item$: the item candidate to recommendation, k: the number of neighbors to consider, γ: the threshold.

Output: the TOP k neighbors

1: $CN \leftarrow \emptyset$ {Set of Candidate Neighbors}
2: $c^* := c \in \mathcal{C} | \forall c_i \in \mathcal{C}, MATCH(c_i, c_a) \leq MATCH(c^*, c_a)$
3: $OP_u := BIND(CP_u, c^*)$
4: **for all** $CP_{ui} \in \mathcal{U}$ **do**
5: **if** $m < (Item, *), *, * > \in CP_{ui}$ **then**
6: $OP_{ui} := BIND(CP_{ui}, c_a)$
7: $CN \leftarrow CN \cup \{OP_{ui}\}$
8: $TOPK \leftarrow \{OP_{ui} \in CN, i = \overline{1,k}\}$ such that $\forall OP_{uj} \in CN, OP_{uj} \notin TOPK, MATCH(OP_{uj}, OP_{ua}) \leq MATCH(OP_{ui}, OP_{ua})$
9: **return** $TOPK$

6.2 Aggregating Neighbor Ratings

Once Top k neighbors of the active user u_a are determined, ratings they gave to the item (It) to be recommended must be aggregated. The result of this aggregation allows deciding whether it is relevant or not to recommend It to the user u_a. There exist several techniques for aggregating these ratings [8, 6], the one we considered is the *Weighted Mean Aggregation* (equation 6) where the contribution of each neighbor rating is weighted with the similarity between this neighbor and the active user u_a.

$$Rate(u, It) = \alpha \sum_{u_i \in TopK} Match(u, u_i) \times Rate(u_i, It) \quad (6)$$

α is normalizing factor.

According to the value of the aggregated rating, the RS decides whether It could be or not recommended.

6.3 Simple scenario

The scenario below gives an intuition about the way CARS provides users with contextual recommendations. Figure 4 focuses on the recommender engine (RE) sketched in figure 1. Dashed arrows represent the inputs of the RE.

When recommendations are requested, explicitly by users (here John) or implicitly by the applications, RE computes them following three processes A, B, and C shown in fig 4.

First of all, a parser extracts the active context (c_a) of the user, and designates the candidate content to recommendation (step A1). In step B1, the active context (c_a) is matched with all context clusters in order to determine that to which c_a belongs (line 2 of the algorithm). This context cluster is used in step B2 for binding John's contextual profile and producing his operational profile (line 3). At the same time, RE looks for the candidate neighbors of John (process C: lines 4-7). A candidate neighbor is a user who rated the content to be recommended (filtering: step C1), in a context c_r similar to the active context c_a (step C2). In step C3, the contextual profile of each candidate user is bound to its corresponding context c_r, resulting in a set of operational profiles. Step A2 synchronizes the two processes B and C. It consists in determining the top k neighbors of John among the candidate users. This is done by capturing

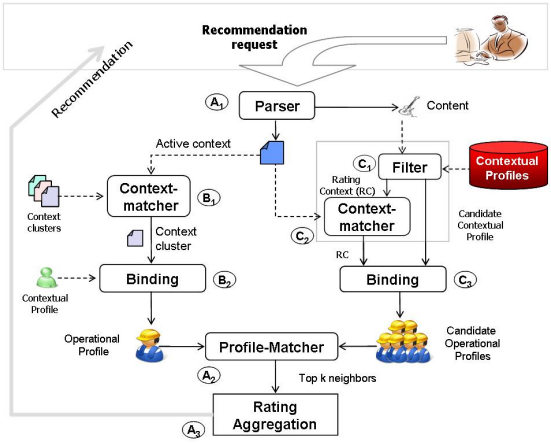

Figure 4: Contextual CF-RS

the similarity between each candidate profile and John profile; only top k similar profiles are retained (line 8). Step A3 aggregates ratings given to the recommended content by the top k returned neighbors. Finally, a decision is made to know whether it is interesting or not to recommend the candidate content (through a threshold for example).

7. RELATED WORK

Adomavicius et al. [4] discussed what a next generation of RS might be and proposed some possible extensions that will improve recommender capabilities such as the integration of contextual information into recommendation processes. In [3], a multidimensional approach is presented to incorporate the context into recommendations. The approach is based on ratings that are captured in OLAP cubes and which are sensitive to contextual information such as *Time, Place,* and *Accompanying people*. The same multidimensional cube-based approach was proposed to manage contextual preferences in the database field [13]. Contextual preferences are usually of the form: $Pref(item, c_1, \ldots, c_n) = w$, where c_i is a contextual attribute, and the weight $w \in [0, 1]$ expresses the interest a user has on the *item*. Notice that Contextual preferences were subject to extensive research in database field [12, 5, 2]. Hence, understanding these researches can help in designing more meaningful CARS. The difference between these approaches and our is that, in the latter one, contextual attributes are clustered to form a finite set of regular contexts of users (e.g. Lab context, Walking context) while, in the former, each instance of contextual attribute is considered itself as a context.

Inspired by human memory, Anand and Mobasher [6] proposed a contextual recommendation where users are modeled through a short-term memory (STM) which stores current interactions and a long-term memory (LTM) which stores previous interactions. Contextual Cues are generated from STM to extract relevant ratings from LTM according to the context (i.e. cues). These ratings are merged with the STM to provide user with contextual recommendations. The Binding service we presented has some similarities with this approach in the manner it computes the operational profile using both contextual profile (LTM) and active context (Cue). However, unlike Anand et al. who consider the interactional view of the context, we have considered the representational view in which the context is an information, stable, delineable, and separable from the activity [9]. More recently, there was attempts to define architectures for context-aware recommender [7]. However, author don't give details about the deployment of such architectures.

8. CONCLUSION

In this paper, we have proposed a set of personalization services that improve RS by introducing the notion of context. Among these services, two of them (Context Discovery and Contextualization) are design services and two others (Binding and Matching) are recommendation services. For each service, we have defined its operational semantics through one or several algorithms. Finally, we have shown how these services can be combined to form a CARS. Further research on this topic will concentrate on the evaluation of this approach by comparing traditional RS with CARS. To this end, a significant benchmark has to be defined.

9. REFERENCES

[1] S. Abbar, M. Bouzeghoub, D. Kostadinov, and S. Lopes. A contextualization service for a personalized access model. In *EGC*, pages 265–270, 2009.

[2] S. Abbar, M. Bouzeghoub, D. Kostadinov, S. Lopes, A. Aghasaryan, and S. Betge-Brezetz. A personalized access model: concepts and services for content delivery platforms. In *Proceedings of the 10th iiWas, Linz, Austria*, pages 41–47, 2008.

[3] G. Adomavicius, R. Sankaranarayanan, S. Sen, and A. Tuzhilin. Incorporating contextual information in recommender systems using a multidimensional approach. *ACM Trans. Inf. Syst.*, 23(1):103–145, 2005.

[4] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE TKDE*, 17(6):734–749, 2005.

[5] R. Agrawal, R. Rantzau, and E. Terzi. Context-sensitive ranking. In *ACM SIGMOD*, pages 383–394, 2006.

[6] S. S. Anand and B. Mobasher. Contextual recommendation. *Discovering and Deploying User and Content Profiles. Berlin, Germany*, pages 142–160, 2007.

[7] L. Baltrunas. Exploiting contextual information in recommender systems. In *RecSys '08*, pages 295–298, New York, NY, USA, 2008. ACM.

[8] J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. Technical report, MSR-TR-98-12. Microsoft research, Redmond, WA 98052, 1998.

[9] P. Dourish. What we talk about when we talk about context. *Pers. Ubiq. Comput.*, pages 19–30, 2004.

[10] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers, 550 pages. ISBN 1-55860-489-8, 2000.

[11] B. B. Phillip M. Hallam-Baker. Extended log file format. Technical report, W3C Draf WD- logfile -960323. http://www.w3.org/TR/WD-logfile, 1996.

[12] K. Stefanidis and E. Pitoura. Fast contextual preference scoring of database tuples. In *EDBT, Nantes, France*, pages 344–355, 2008.

[13] K. Stefanidis, E. Pitoura, and P. Vassiliadis. A context-aware preference database system. *Interl. Journal of PCC*, pages 439–460, 2007.