Context Awareness by Case-Based Reasoning in a Music Recommendation System

Jae Sik Lee[1] and Jin Chun Lee[2]

[1] Division of e-Business, School of Business Administration, Ajou University
San 5, Wonchun-Dong, Youngtong-Gu, Suwon 443-749, Korea
leejsk@ajou.ac.kr
[2] Ubiquitous Convergence Research Institute
San 5, Wonchun-Dong, Youngtong-Gu, Suwon 443-749, Korea
giny777@empal.com

Abstract. The recommendation system is one of the core technologies for implementing personalization services. Recommendation systems in ubiquitous computing environment should have the capability of context-awareness. In this research, we developed a music recommendation system, which we shall call C^2_Music, which utilizes not only the user's demographics and behavioral patterns but also the user's context. For a specific user in a specific context, the C^2_Music recommends the music that the similar users listened most in the similar context. In evaluating the performance of C^2_Music using a real world data, it outperforms the comparative system that utilizes the user's demographics and behavioral patterns only.

Keywords: Music Recommendation System, Context-Awareness, Case-based Reasoning, Ubiquitous Data Mining, Personalization.

1 Introduction

In ubiquitous computing environment, various types of data will be obtained through various types of devices in any time and any place. Enormous amount of data generated from our everyday life will be collected by sensors and computers, and will be used to provide us with intelligent services.

The problem that we will be confronted with in this ubiquitous computing environment is 'information overload.' Therefore, we need some systems that alleviate this information overload problem effectively and efficiently, that is, recommendation systems. Recommendation can be defined as "the process of utilizing the opinions of a community of customers to help individuals in that community more effectively identify content of interest from a potentially overwhelming set of choices" [21]. Effective recommendation reduces the user's effort and time in making decisions. Researches on recommendation systems have been performed actively both in academy and practice. However, in making recommendation, most of the existing researches have focused on using the user's

H. Ichikawa et al. (Eds.): UCS 2007, LNCS 4836, pp. 45–58, 2007.

preference and/or transaction data. They rarely considered such information as user's context at the time of making recommendation.

Suppose a customer at his thirties visits a department store. Traditional recommendation systems will recommend the items that he might be interested in by analyzing his past purchase history and preferences. If he comes to the department store for the purpose of buying some items for himself, then the recommended items by the traditional systems would be appropriate. However, if his purpose is to buy some presents for his wife for their wedding anniversary, then anniversary items or ladies' accessories should be recommended. In other words, the items must be recommended considering the situation the customer is placed in, i.e., the customer's context.

In this research, we developed a music recommendation system that utilizes not only the user's demographics and behavioral patterns but also his/her context at the time of making recommendation. The underlying algorithm of our proposed system is case-based reasoning. The rest of the paper is organized as follows: The next section provides a brief overview of the related work, i.e., context, context-awareness, recommendation systems and case-based reasoning. Section 3 provides an overall structure of our proposed music recommendation system. Section 4 describes the implementation process of our proposed system. Section 5 presents the performance evaluation of the proposed system. The final section provides concluding remarks and directions for further research.

2 Related Work

2.1 Context and Context-Awareness

Context-awareness is to use information about the circumstances that the application is running in, to provide relevant information and/or services to the user [5]. The term 'context-awareness' was introduced by Schilit and Theimer [24]. Context awareness is a term that describes the ability of the computer to sense and act upon information about its environment, such as location, time, temperature or user identity [22].

Schilit and Theimer [24] defined 'context' through giving a number of examples of context, i.e., location, identities of nearby people and objects, and changes to those objects. The term context, however, has been defined by many researchers in various ways. Schmidt *et al.* [25] defined it using three dimensions: physical environment, human factors and time. Benerecetti *et al.* [2] classified context into physical context and cultural context. Physical context is a set of features of the environment while cultural context includes user information, the social environment and beliefs. Dey and Abowd [7] defined context as "any information that can be used to characterize the situation of an entity, where an entity can be a person, place, physical or computational object that is considered relevant to the interaction between a user and an application, including the user and the application themselves". Dey [6] presented four types of context, i.e., location, identity, time and activity, and provided a framework for defining a context by giving values to these four types.

By sensing context information, context-aware applications can present context information to users, or modify their behavior according to changes in the environment [23]. Dey [6] defined context-aware application as “a system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user’s task.” Three important context awareness behaviors are the presentation of information and services to a user, automatic execution of a service, and tagging of context to information for later retrieval [7].

Early investigations on context-aware application were carried out at the Olivetti Research Lab with development of the active badge system [27][29], sensing locality of mobile users to adapt applications to people’s whereabouts. Another research can be found at Xerox PARC with the ubiquitous computing experiment, from which a first general consideration of context-aware mobile computing emerged [22]. Dey *et al.* developed the Conference Assistant, a prototype context-aware application for assisting conference attendees and presenters [8]. The capabilities of this system is to help users decide which activities to attend, to provide awareness of the activities of colleagues, to enhance interactions between users and the environment, to assist users in taking notes on presentations and to aid in the retrieval of conference information after the conference concludes. Dey *et al.* [9] also developed a context toolkit to support rapid prototyping of certain types of context-aware applications. Many researchers have adopted this toolkit approach [11][12], while others have been developing a middleware infrastructure [16].

2.2 Music Recommendation Systems

The recommendation systems are to recommend items that users may be interested in based on their predefined preferences or access histories [4]. Many of the leading companies such as Amazon, Google, CDNOW, LA Times and eBay, are already using personalized recommendation systems to help their customers find products to purchase.

Ringo is a pioneer music recommendation system using collaborative filtering [26]. In Ringo, each user is requested to make ratings for music objects. These ratings constitute the personal profiles. For collaborative recommendation, only the ratings of the users whose profiles are similar to the target user are considered. Whether a music objects will be recommended is then based on the weighted average of the ratings considered.

Kuo and Shan [18] developed a content-based music recommendation system. In their system, the users’ preferences are learned by mining the melody patterns, i.e., the pitch information, of the music they listened. Chen and Chen [4] proposed a music recommendation system that employed three recommendation mechanisms, i.e., content-based method, collaborative filtering and statistics-based method, for different users’ needs. In their system, music objects are grouped according to the properties such as pitch, duration and loudness, and users are grouped according to their interests and behaviors derived from the access histories.

Celma *et al.* [3] proposed a music recommendation system called ‘Foafing the Music’. The Friend of a Friend (FOAF) project is about creating a Web of machine-readable homepages describing people, the things they create and do, their interests and the links between them. The 'Foafing the Music' system uses this FOAF information and the Rich Site Summary that publishes new releases or artists' related news, for recommending music to a user, depending on his/her musical tastes.

Park *et al.* [20] proposed a context-aware music recommendation system that employs fuzzy system, Bayesian networks and utility theory. Kim *et al.* [13] designed a music recommendation system that makes recommendation using user information such as sex, age and pulsation, and surrounding contexts such as weather, temperature and location. The profiles of music, not music listeners, are prepared using these factors and stored in Music Content Information Database. Then, suitable music is selected using a statistical filtering method.

2.3 Case-Based Reasoning

Case-based reasoning (CBR), a well-known artificial intelligence technique, has already proven its effectiveness in numerous domains. The fundamental concept in CBR is that similar problems will have similar solutions. CBR is a method of solving a new problem by analyzing the solutions to previous, similar problems [15][28][30]. Since CBR can provide answers just using accumulated previous cases, i.e., case base, it can be applied to complicated and unstructured problems relatively easily. The most distinguished advantage of CBR is that it can learn continuously by just adding new cases to the case base.

As shown in Figure 1, CBR is typically described as a cyclical process comprising the four REs [1]: (1) REtrieve the most similar case or cases, (2) REuse the information and knowledge in that case to attempt to solve the problem, (3) REvise the proposed solution if necessary, and (4) REtain the parts of this experience likely to be useful for future problem solving.

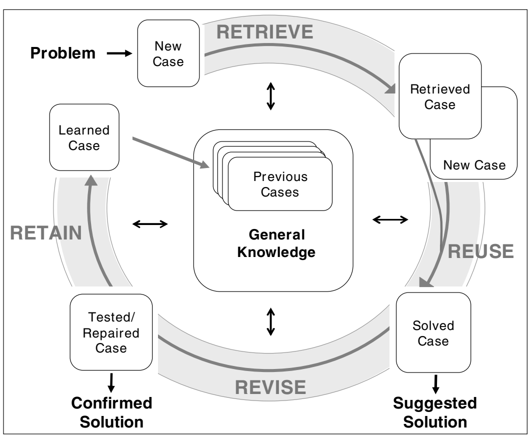

Fig. 1. Problem Solving Cycle of CBR

CBR is basically based on the k-Nearest Neighbors (NN) algorithm. Therefore, in order to build the CBR model, we need to define the similarity function that will be used to find k previous cases similar to the new case. The similarity score between a new case N and a previous case C is calculated using the Equation (1) in general,

$$Similarity\ (N,C) = \frac{\sum_{i=1}^{n} f(N_i, C_i) \times W_i}{\sum_{i=1}^{n} W_i} \tag{1}$$

where N_i is the i^{th} feature value of the new case, C_i is the i^{th} feature value of the old case, n is the number of features, $f(N_i, C_i)$ is the distance function between N_i and C_i, and W_i is the weight of i^{th} feature. The value of similarity score is between 0 and 1. The more the two cases N and C are similar, the more similarity score becomes close to 1.

Recently, CBR became applied to the development of context-aware applications. Kofod-Petersen and Aamodt [14] incorporated context information as cases in CBR for user situation assessment in a mobile context-aware system. Kumar *et al.* [17] proposed a context enabled Multi-CBR approach. It consisted of two CBRs, i.e., user context CBR and product context CBR, for aiding the recommendation engine in retrieving appropriate information for e-commerce applications. Leake *et al.* [19] presented three potential areas in which the use of CBR may provide benefits for context-aware applications, especially in smart homes, i.e., supporting personalization, supporting interactive adjustment of the system by the user, and facilitating customization and knowledge acquisition by the developer.

3 A Case-Based Context-Aware Music Recommendation System

In this section, we describe the structure and the components of the proposed system, we shall call C^2_Music. Let us think about the following scenario:

> "The user A is a white-collar worker at his early thirties who enjoys listening to music. It is Friday and has been raining from morning, and it is chilly for a day in late June. When he enters his room after returning home from work, his audio system is turned on automatically and plays the Creedence Clearwater Revival's 'Who'll Stop the Rain' that he usually listened on a day like today. Even though Rod Stewart revived the song recently, the user A prefers the original song by CCR back in 1970."

In the above scenario, the music is selected by C^2_Music. The C^2_Music consists of three layers, i.e., Interface Layer, Application Layer and Repository Layer as depicted in Figure 2.

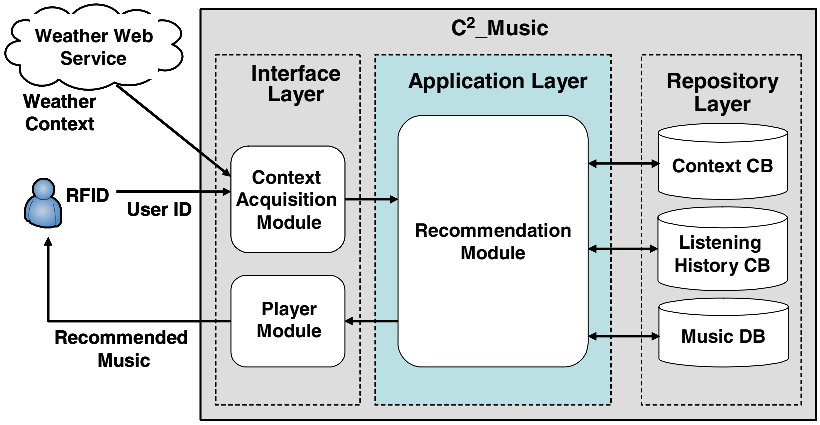

Fig. 2. The Structure of C^2_Music

The Interface Layer is to identify the user, to collect the context data and to deliver the recommended music to the user. The Application Layer is to select the music for recommendation. The Repository Layer is to store relevant cases and data. The detailed roles and functions of the three layers and their modules will be explained as we describe the music recommendation process of C^2_Music in section 4.2. Figure 3 shows the entity-relationship diagram of the cases and data stored in the Repository Layer that consists of Context Case Base (CB), Listening History CB and Music Data Base (DB).

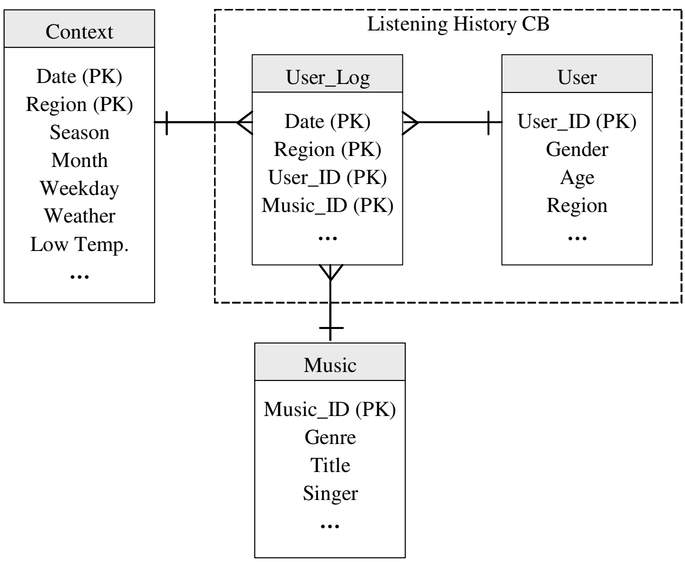

Fig. 3. ERD for the Repository Layer

4 Implementation of C^2_Music

4.1 Data Description

The data used in this research are listening history data set and weather data set. The listening history data set was obtained from a streaming music service company in Korea, and it contains the list of songs listened by 659 customers for 6 months. Table 1 shows the features of user's characteristics stored in the Listening History CB. The weather data set was obtained from the Weather Bureau. Table 2 shows the weather data stored in the Context CB.

Table 1. Features of User's Characteristics Stored in Listening History CB

Feature Name	Description	Type
User_ID	User's ID Number	Categorical
X1	Gender	Categorical
X2	Age	Numerical
X3	Region	Categorical
X4	Number of Listening Days for Last One Month	Numerical
X5	Number of Listening Times for Last One Month	Numerical
X6	Number of Listened Songs for Last One Month	Numerical
….	….	….
X12	Ratio of Rock and Metal Songs for Last One Month	Numerical
X13	Ratio of Korean Trot Songs for Last One Month	Numerical

Table 2. Features Stored in Context CB

Feature Name	Description	Type
Date	Music Listening Date	Categorical
Region	Residence Area of User	Categorical
Season	Spring, Summer, Fall, Winter	Categorical
Month	January, February, March, April, May, June, July, August, September, October, November, December	Categorical
Weekday	Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday	Categorical
Weather	Sunny, Partly Cloudy, Mostly Cloudy, Cloudy, Rainy, Snow	Categorical
Avg_Temp	Average Temperature During a Day (Unit: Celsius)	Numeric
High_Temp	Highest Temperature During a Day (Unit: Celsius)	Numeric
Low_Temp	Lowest Temperature During a Day (Unit: Celsius)	Numeric

4.2 Implementation of the Recommendation Module

The C^2_Music recommends the music to the user in the following four steps:

Step 1: Identify the user and collect the context data

The Context Acquisition Module identifies the user A using the RFID sensor, and collects the weather data from web service. The collected weather data includes 'season,' 'month,' 'day of the week,' 'atmospheric condition,' 'the lowest temperature,' 'the highest temperature' and 'the average temperature' of the corresponding date. The Context Acquisition Module constructs an input case using these collected data and delivers it to the Recommendation Module. This constructed case is stored in the Context CB for later use.

The following steps, i.e., steps 2, 3 and 4 performed by the Recommendation Module are simply depicted in Figure 4.

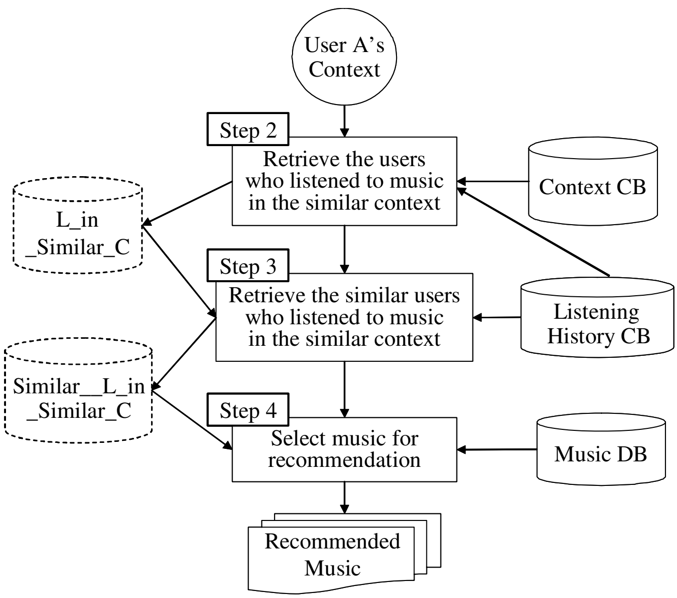

Fig. 4. The Music Recommendation Process of C^2_Music

Step 2: Retrieve the users who listened to music in the similar context

The Recommendation Module retrieves the top *k* past contexts similar to the user A's present context from the Context CB, and identifies the dates that correspond to the retrieved contexts. Input features for CBR model are presented in Table 2. Then, it retrieves the users who listened to music on these dates from the Listening History CB and makes a temporary case base named 'L_in_Similar_C' (meaning 'Listeners in the Similar Context'). In other words, in this step, we retrieve the users who listened to music in the context similar to the user A's present context from the Listening History CB.

In calculating the similarity scores for finding similar contexts, Equation (2) is used for numerical features.

$$f(N_i, C_i) = \begin{cases} 1-d & \text{if } 0 \le d \le 1 \\ 0 & \text{if } d > 1 \end{cases} \tag{2}$$

where,

$$d = \frac{|N_i - C_i|}{Max - Min}$$

Max : maximum value among i^{th} feature values for all cases in case base.
Min : minimum value among i^{th} feature values for all cases in case base.

For categorical features, a partial matching scheme is devised using the domain knowledge as shown in Table 3.

Table 3. Similarity Score Metrics for Categorical Features

<table>
<tr><th>Feature</th><th>Similarity Score Metrics</th></tr>
<tr><td>Season</td><td>
<table>
<tr><td>New Case / Old Case</td><td>Spring</td><td>Summer</td><td>Fall</td><td>Winter</td></tr>
<tr><td>Spring</td><td>1</td><td>0.2</td><td>0.5</td><td>0.2</td></tr>
<tr><td>Summer</td><td>0.2</td><td>1</td><td>0.2</td><td>0</td></tr>
<tr><td>Fall</td><td>0.5</td><td>0.2</td><td>1</td><td>0.2</td></tr>
<tr><td>Winter</td><td>0.2</td><td>0</td><td>0.2</td><td>1</td></tr>
</table>
</td></tr>
<tr><td>Month</td><td>
<table>
<tr><td>Similarity Value</td><td>Condition</td></tr>
<tr><td>1</td><td>$N_i = C_i$</td></tr>
<tr><td>0.5</td><td>Distance Between N_i and C_i = 1 Month</td></tr>
<tr><td>0.2</td><td>Distance Between N_i and C_i = 2 Months</td></tr>
<tr><td>0</td><td>Otherwise</td></tr>
</table>
</td></tr>
<tr><td>….</td><td>….</td></tr>
<tr><td>….</td><td>….</td></tr>
</table>

Step 3: Retrieve the similar users who listened to music in the similar context

The Recommendation Module retrieves the top *k* users whose demographics and behavioral patterns are similar to those of the user A from ‘L_in_Similar_C’, and makes a temporary case base named ‘Similar_L_in_Similar_C’ (meaning ‘Similar Listeners in the Similar Context’). Input features for CBR model are presented in Table 1.

In calculating the similarity scores for finding similar contexts, Equation (2) is used for numerical features. For categorical features, an exact matching scheme is used. Similarity score 1 is assigned if two features have the same values, 0 is assigned otherwise.

Step 4: Select music for recommendation

The Recommendation Module first retrieves the songs listened by the users in 'Similar_L_in_Similar_C' from the Listening History CB and composes the candidates set for recommendation. Before selecting the songs for recommendation from the candidates set, we need to determine the selection criterion and the number of songs to be recommended. In this research, we select songs on the basis of frequency and recency. Therefore, we sort the candidate songs in the descending order of the listening frequencies by similar listeners and the last dates when they are listened. Since the number of songs normally contained in one CD is 15, we select 15 songs for recommendation.

In other words, the Recommendation Module selects the top 15 songs that the users in 'Similar_L_in_Similar_C' listened most and recently, from the Listening History CB. Then, it retrieves the corresponding music files from Music DB and provides them for the user A through the Player Module.

5 Performance Evaluation of C^2_Music

In order to evaluate the performance of C^2_Music, we implemented a comparative system. The comparative system is C_Music (meaning conventional Case-Based Music Recommendation System) that employs CBR technique and makes recommendations using the other users' listening histories stored in the Listening History CB. In other words, while the input features of C^2_Music are the features presented in Tables 1and 2, the input features of C_Music are the features in Table 1 only. The two systems were implemented using Microsoft Visual Basic 6.0 on PC.

For performance evaluation of the two systems, we performed 10-fold cross validation. The 10-fold data sets were prepared as shown in Figure 5. In each fold, we divided the data set into the training data set and the test data set in the ratio of 6 and 4.The training data set was used for constructing the case base and optimizing the CBR models. The test data set was used for evaluating the performances of the two final systems.

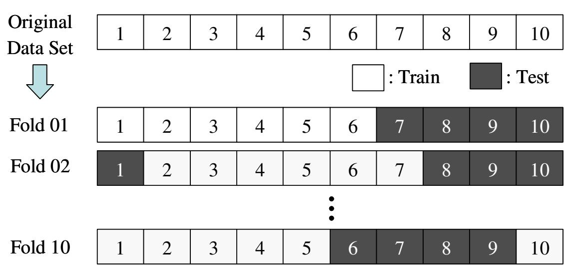

Fig. 5. Configuration of 10-Fold Data Sets

CBR is basically based on the *k*-NN algorithm. Therefore the performance of CBR model is affected by the value of *k*. Another factor that affects the performance of

CBR is the weight vector of the input features. For implementing the best model, we first generated hundred weight vectors, and then experimented with the resulting CBR model by varying the value of *k* from 10 to 100 at the increment of 5 for each weight vector.

Evaluation metrics are essential in order to judge the quality and performance of recommendation systems. There has been considerable research in the area of recommendation systems evaluation [10]. In this research, we use Precision calculated using Equation (3) as our choice of evaluation measure because the number of recommended songs is fixed as 15 and our research focus is to improve the accuracy of the recommendation system [4].

$$\mathrm{Pr}\,ecision = \frac{R^L}{R} \tag{3}$$

where,

R^L : the number of songs, among the recommended songs, actually listened.
R : the number of songs recommended by the recommendation system.

Table 4 shows the precisions of the two systems, i.e., C_Music and C^2_Music on the test data set.

Table 4. Precision Comparison of C_Music and C^2_Music on Test Data Sets

Fold	**C_Music**	**C^2_Music**
Fold 01	0.464	0.534
Fold 02	0.493	0.533
Fold 03	0.481	0.595
Fold 04	0.470	0.572
Fold 05	0.420	0.490
Fold 06	0.410	0.500
Fold 07	0.480	0.551
Fold 08	0.467	0.545
Fold 09	0.457	0.559
Fold 10	0.465	0.548
Average	0.461	0.542

As shown in Table 4, on average, the precision of C^2_Music is 0.542 that is 8% point higher than the precision of C_Music, 0.461. In order to examine if our results are statistically significant, the Wilcoxon Signed Rank Sum Test is performed.

Table 5. Wilcoxon Signed Rank Sum Test of C_Music and C^2_Music

Comparison	**z-Statistic**
C_Music – C^2_Music	-2.806^*

*: Significant at 5% level

As shown in Table 5, C^2_Music outperforms C_Music at the 5% significance level. This fact shows that the precision of recommendation can be increased by utilizing the environmental context data.

6 Conclusion

In ubiquitous computing environment, the need for context-aware recommendation systems that provides the users with personalized services using various contextual information will be increased. In this research, we developed a context-aware music recommendation system using case-based reasoning, that we shall call C^2_Music. The distinguished features of C^2_Music are twofold. First, for selecting the music for recommendation, it utilizes not only the user's demographics and behavioral patterns but also his/her context at the time of making recommendation. Secondly, it employs 2-step case-based reasoning, i.e., the first step for retrieving the similar contexts and the second step for retrieving the similar users.

The C^2_Music consists of three layers, i.e., Interface Layer, Application Layer and Repository Layer. The Interface Layer is to identify the user, to collect the context data and to deliver the recommended music to the user. The Application Layer is to select the music for recommendation. The Repository Layer that consists of Context Case Base, Listening History Case Base and Music Data Base is to store relevant cases and data.

In order to evaluate the performance of C^2_Music, we implemented a comparative system C_Music that also employs case-based reasoning technique but does not utilize the user's context. As a result of 10-fold cross validation with a real world data, the average precision of C^2_Music is 0.542 that is 8% point higher than the average precision of C_Music, 0.461. This result shows that the precision of recommendation can be increased by utilizing the context data.

For further research, we plan to continue our study on the following issues: First, we will consider and collect more features that can represent the user's context, and then select the appropriate features by feature selection process. Secondly, we need to compare our proposed system with other context-aware recommendation systems that employ some techniques other than case-based reasoning, such as ontology. Finally, we will do our research on the methods for speeding up the recommendation system, because in ubiquitous computing environment, not only accuracy but also speed of the recommendation is important.

Acknowledgments. This research is supported by the Ubiquitous Autonomic Computing and Network Project, the Ministry of Information and Communication (MIC), 21st Century Frontier R&D Program in Korea.

References

1. Aamodt, A., Plaza, E.: Case-based Reasoning: Fundamental Issues, Methodological Variations, and System Approaches. Artificial Intelligence Communication 7, 39–59 (1994)
2. Benerecetti, M., Bouquet, P., Bonifacio, M.: Distributed Context-Aware System. Human-Computer Interaction 16, 213–228 (2000)

3. Celma, O., Ramírez, M., Herrera, P.: Foafing the Music: A Music Recommendation System based on RSS Feeds and User Preferences. In: Proceedings: 6th International Conference on Music Information Retrieval, London, UK (2005)
4. Chen, H.C., Chen, A.P.: A Music Recommendation System Based on Music and User Grouping. Journal of Intelligent Information Systems 24, 113–132 (2005)
5. Cuddy, S., Katchabaw, M., Lutfiyya, H.: Context-Aware Service Selection based on Dynamic and Static Service Attributes. In: Wireless and Mobile Computing, Networking and Communications, IEEE International Conference (2005)
6. Dey, A.K.: Understanding and Using Context. Personal and Ubiquitous Computing 5, 4–7 (2001)
7. Dey, A.K., Abowd, G.D.: Towards a Better Understanding of Context and Context-Awareness. In: Proceedings of CHI 2000 Workshop on the What, Who, Where, When, Why, and How of Context-Awareness, The Hague, Netherlands, pp. 1–6 (2000)
8. Dey, A.K., Futakawa, M., Salber, D., Abowd, G.D.: The Conference Assistant: Combining Context-Awareness with Wearable Computing. In: Proceedings Symposium on Wearable Computers, pp. 21–28 (1999)
9. Dey, A.K., Salber, D., Abowd, G.D.: A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications. Human-Computer Interaction 16, 97–166 (2001)
10. Herlocker, J., Konstan, J., Tervin, L.G., Riedl, J.: Evaluating Collaborative Filtering Recommender Systems. ACM Transactions on Information Systems 22, 5–53 (2004)
11. Hong, J.: The Context Fabric. Berkeley, USA (2003), http://guir.berkeley.edu/projects/confab/
12. Jang, S., Woo, W.: Ubi-UCAM: A Unified Context-Aware Application Model. In: Proceedings of Context 2003, Stanford, CA, USA (2003)
13. Kim, J.H., Song, C.H., Lim, K.W., Lee, J.H.: Design of Music Recommendation System Using Context Information. In: Shi, Z.-Z., Sadananda, R. (eds.) PRIMA 2006. LNCS (LNAI), vol. 4088, pp. 708–713. Springer, Heidelberg (2006)
14. Kofod-Petersen, A., Aamodt, A.: Case-based Situation Assessment in a Mobile Context-Aware Systems. In: AIMS2003. Workshop on Artificial Intelligence for Mobile Systems, Seattle (October 2003)
15. Kolodner, J.L.: Case-based Reasoning. Morgan Kaufman, San Mateo, CA (1993)
16. Kumar, M., Shirazi, B.A., Das, S.K., Sung, B.Y., Levine, D., Singhal, M.: PICO: a Middleware Framework for Pervasive Computing. IEEE Pervasive Computing 2, 72–79 (2003)
17. Kumar, P., Gopalan, S., Sridhar, V.: Context Enabled Multi-CBR based Recommendation Engine for E-Commerce. In: ICEBE 2005. Proceedings of IEEE International Conference on e-Business Engineering, pp. 237–244. IEEE Computer Society Press, Los Alamitos (2005)
18. Kuo, F.F., Shan, M.K.: A Personalized Music Filtering System Based on Melody Style Classification. In: Proceedings of IEEE international Conference on Data Mining, pp. 649–652. IEEE Computer Society Press, Los Alamitos (2002)
19. Leake, D., Maguitman, A., Reichherzer, T.: Cases, Context, and Comfort: Opportunities for Case-Based Reasoning in Smart Homes. In: Augusto, J.C., Nugent, C.D. (eds.) Designing Smart Homes. LNCS (LNAI), vol. 4008, pp. 109–131. Springer, Heidelberg (2006)
20. Park, H.S., Yoo, J.O., Cho, S.B.: A Context-Aware Music Recommendation System Using Fuzzy Bayesian Networks with Utility Theory. In: Wang, L., Jiao, L., Shi, G., Li, X., Liu, J. (eds.) FSKD 2006. LNCS (LNAI), vol. 4223, pp. 970–979. Springer, Heidelberg (2006)

21. Resnick, P., Varian, H.R.: Recommender Systems. Communications of the ACM 40, 56–58 (1997)
22. Ryan, N.: Mobile Computing in a Fieldwork Environment: Metadata Elements. Project Working Document, Version 0.2 (1997)
23. Salber, D., Dey, A.K., Orr, R.J., Abowd, G.D.: Designing For Ubiquitous Computing: A Case Study in Context Sensing. GVU Technical Report GIT-GVU, 99–129 (1999)
24. Schilit, B., Theimer, M.: Disseminating Active Map Information to Mobile Hosts. IEEE Network 8, 22–32 (1994)
25. Schmidt, A., Beigl, M., Gellersen, H.W.: There is More to Context Than Location. Computers and Graphics 23, 893–901 (1999)
26. Sharadanand, U., Maes, P.: Social Information Filtering: Algorithms for Automating 'Word of Mouth'. In: Proceedings of CHI 1995 Conference on Human Factors in Computing Systems, pp. 210–217 (1995)
27. Want, R., Hopper, A., Falcao, V., Gibbons, J.: The Active Badge Location System. ACM Transactions on Information Systems 10, 91–102 (1992)
28. Wang, H.C., Wang, H.S.: A Hybrid Expert System for Equipment Failure Analysis. Expert Systems with Applications 28, 615–622 (2005)
29. Ward, A., Jones, A., Hopper, A.: A New Location Technique for the Active Office. IEEE Personal Communications 4, 42–47 (1997)
30. Watson, I.: Applying Case-based Reasoning: Techniques for Enterprise System. Morgan Kaufmann, San Francisco, CA (1997)