Action Suggestion Using Situation Rules

Anton Benčič
Institute of Informatics and Software Engineering
Faculty of Informatics and Information Technologies, Slovak University of Technology
Ilkovičova 3, 842 16 Bratislava, Slovakia
bencican@live.com

Mária Bieliková
Institute of Informatics and Software Engineering
Faculty of Informatics and Information Technologies, Slovak University of Technology
Ilkovičova 3, 842 16 Bratislava, Slovakia
bielik@fiit.stuba.sk

Abstract— **Nowadays we can see a new era of mobile computing springing up. Mobile devices more often than not provide incomparably more relevant information and context about their users than was ever available on desktops or within classic web browsing. With this a new branch of research for autonomous software is forming. The aim is to recognize usage situations to let an application decide on performing a specific action autonomously. In this paper we describe our novel method for learning users' situation preferences to suggest the right moments for performing specific actions independently. Such action can be for example autonomous news push in a news application, but it can be just as easily applied to songs or microblog recommenders. User preferences are described with situations the users encounter throughout the time and rules that are based on either implicit or explicit feedback from the users. The focus of this paper is on the action suggestion method for the right moment for performing an action alongside which we also introduce a few usage scenarios, discuss on characteristics and limits of our method and present experiments that evaluate on our method's performance in various scenarios.**

Keywords- action suggestion; situation model; action model; situation rule; machine learning

I. Introduction and Related Work

We live in a world where the pervasive computing is already present in many different areas of our lives. Recommendations can be also considered one such area. Today however in most cases the user has to initiate the interaction by stating her intentions. This is by explicitly stating her query or even by opening a site of an internet shop. Other approach employs the concept of notifications, i.e. the interaction is initiated by the application. However, most of the time these notifications are not presented to the users in a sophisticated way, especially in the right moment, but rather naively, only considering the content and not the user's situation and preferences that can be estimated based on her activity in a particular context.

In this paper we propose a novel approach to action recommendation based on situation rules. We refer to an action as to anything an end-user service or application can perform on its own. In autonomous news recommendation for example pushing news to the user's device can be considered a typical action. Our method is designed to support autonomous decision making in context-aware services and applications that need to make decisions on actions without a user initiation. For example, a news service that decides on the type of presented content like the one presented [16] can be extended to identify whether it is the right time to push some news content to the user. This would even further maximize the chances that the user responds positively and that an overall satisfaction is achieved. A trivial example of such news service that attempts to identify suitable moments is presented in [17]. An example from a different domain is an entertainment service that can decide when it is the right time to open the store and propose some music albums, games or movies. Of course the situation suitability often depends also on the content recommended and our method is perfectly capable of handling these cases. In [18] the authors for example present an adaptive event notifier that can decide when it is the best moment to alert the user so that she does not miss anything.

The reason we need a rather sophisticated method is that clues for the decisions are often indirect and almost always user dependent. A simple example would be that of a travelling user. We do not know directly if we should or should not push some news right now, because we do not know right away if she is driving or taking a bus in a particular situation. That is why there are usually no simple decisions and why such end-user services and applications have to personalize their approach to every user.

There are a couple of methods already exploiting rules for automated decision making [1]. The main problem with existing approaches is that they require their rule base to be set up beforehand by an expert, often being the user herself. Moreover, these rules serve as explicit definitions of actions for specific situations making them useful only for simplistic and straightforward scenarios. Such example service is presented in [2] and [3]. In [2] the authors present a method that automatically switches sound profiles on the user's mobile device according to the sensed situation. In [3] the authors present a method for recommending leisure time activities based on what time of week it is and what venues are close to the reported GPS position by combining different recommendation models using pre-built rules.

This kind of straightforward approach is mainly suitable for context-aware applications that need no personalization and the actions performed by them depend purely on the contextual information. Classic examples of such applications may be adaptive mobile guides presented in [4] or vehicle interfaces that decide how and when to present information based on the driving situation at hand [5].

The novelty of our method is in providing means to automatically create rule base using contextual information and user's feedback and suggest actions based on it. The sophisticated combination of certainty factors, context information and rules makes our model well suited for discovering relationship among situations and actions even in scenarios where a human-defined rule base is impossible due to its potential complexity.

Our method is based on symbols which makes it applicable in various scenarios and usable within a range of domains and applications. The only requirement for an end-user service or application is to identify situation classes that might be important for autonomous decision making on actions in the particular domain and our method is then capable of identifying the particular situations that are suitable for individual users.

The paper is structured as follows. In Section II the situation model which represents the user's environment through a set of symbols is described. The action model that consists of a set of automatically generated rules is described in Section III. We present our novel method for action suggestion in Section IV. Section V is devoted to the evaluation. We describe here simulations that demonstrate basic and some advanced characteristics of our rule-based action recommendation method. The paper is concluded with discussion and conclusions.

II. Situation Model

Symbolic situation representation allows for situation models that consist of relatively simple strings (symbols), but allow for representation of a variety of situation classes, from the most simple to complex ones. Because of its flexibility we opted to represent the user's environment through a set of symbols that also serve as a basis for creating rules.

Each symbol represents a particular atomic situation. Every symbol also consists of two parts, where the first part represents the situation class and the second part represents a particular situation within that class. An example may be situation class *Weather* with value *Clear* that would go as *Weather:Clear*.

Every user-situation instance (situation observation) has a certainty value associated with it that represents how certain are we that the user is in such situation at a particular time. The values are in (0.0, 1.0> range, where 0.0 would mean that we are absolutely uncertain about the observation while 1.0 means that we are absolutely certain about it. This gives us more flexibility by allowing us to have more than one situation from any class at the same time, each with its own assigned certainty. This kind of fuzzyfication is very handy for symbol-based models, because it allows us to represent uncertainty as well as position in a segmented continuous interval. For example at 6 AM we can say that it is morning, but at 10 AM it is starting to be lunch time as well. The concept of parallel situations with assigned certainty allows us to represent this proximity more correctly.

Situations are fed into our method for action suggestion at arbitrary times that is dependent mostly on the target client service or application and the conditions, like internet connectivity, service availability or battery state. The situations that are fed represent the user's environment state at a particular time and the certainty attached to them at that time should not be the same sometime later (considering another situation update of the same class has not been delivered yet). That is why we embedded a concept of *time sensitivity* that decreases situations' certainty values as the time passes and it does it using the following formula:

$$CF_t = \frac{CF_b}{(1+r)^t} \qquad (1)$$

where CF_t is the resulting certainty factor at a time t, CF_b is the base certainty initially assigned to a situation, r is the rate of cease that controls how fast is the situation update losing its certainty and t is the time in hours that has elapsed since the observation was made.

One intrinsic advantage of such representation of user's situation is that the symbols bear no meaning with them thus eliminating privacy concerns that we might otherwise experience with some users. This allows for the recommendation engine to be easily deployed within any web service to allow for possible future collaborative models, where rules can be transferred among users based on their similarity to further speed-up the training process.

III. Action Model

The action model consists of a set of rules that are automatically generated based on the user's performance. This includes her implicit and explicit feedback. The first part of any particular rule is a set of antecedents that define in which situation the particular rule applies.

Every antecedent has a certainty value that represents the certainty of an observation at the time the rule was defined. Second part of every rule is its consequence (or action) that is a symbol similar to the situation symbols (section II). The only difference is that conclusion symbols are not defined in classes.

Every rule is also assigned a certainty value that defines what weight a rule holds. For example, considering a news recommendation service, a user opening a news application is a strong clue that this is the right situation for presenting news in future, while a user not responding to a notification is only a weak clue of the opposite. In such case for the positive feedback we define a rule with conclusion like *ShowArticles* and with certainty closer to 1.0 while in case of the weak negative feedback we define a rule with conclusion like *DontShowArticles*, but with certainty closer to 0.0.

The rules, similarly to the situations, employ a *time sensitivity* principle where the rules lose their certainty over time. In this case we however have to adjust the formula (1) so that we keep a steady rule base and not lose all rules due to sparse updates. Hence the parameter r is not constant but rather computed as follows:

$$r = r_b \cdot \left(\left(\frac{\sum CF_{s_{a_1}}}{m \cdot CF_{a_1}} \right) \cdot \left(\frac{\sum CF_{s_{a_2}}}{m \cdot CF_{a_2}} \right) \cdot \ldots \cdot \left(\frac{\sum CF_{s_{a_n}}}{m \cdot CF_{a_n}} \right) \right) \qquad (2)$$

where r_b is the base cease rate associated with the rule (this is similar to situations), CF_{a_n} is certainty of the n-th antecedent in the particular rule, m is the base antecedent certainty cease multiplier that roughly defines how many situation updates it takes to completely cease the rule's certainty and $\sum CF_{s_{a_n}}$ is the sum of certainties of situation updates that match the situation of the n-th antecedent. The parameters that control the resulting cease rate are r_b and m. Their best values may differ by domain, specifically depend on how often a feedback is registered and are best to be found empirically.

IV. Method for Action Suggestion

Our method is based on a set of automatically generated rules that are based on feedback received from a particular user. All rules contain an antecedent part that describes the situation in which the rule was defined and a conclusion part that describes what action the rule suggests when the situation corresponds. All the rules are used within the computation model that aggregates them and calculates the final recommendations.

A. Rule Generation

We define rules upon a client service or application notifies of an action suggestion. In such case the most recent situation update of all situation classes is taken to form antecedents for the newly created rules and the suggested action will be their conclusion. As we already mentioned there can be more than one valid situation from any situation class. In such case more than one rule is created in a way that every rule has exactly one antecedent from every situation class and there are rules for any possible combination of situations among classes. The base suggestion certainty provided by the target service is distributed among the new rules based on certainty of their antecedents compared to the certainty of antecedents from all newly defined rules:

$$CF_r = CF_b \left(\frac{\sum CF_{a_r}}{\sum CF_a} \right) \tag{3}$$

where CF_r is the final rule certainty, CF_b is base certainty assigned by the client service or application, $\sum CF_{a_r}$ is the sum of antecedent certainties from the particular rule and $\sum CF_a$ is the sum of antecedent certainties from all newly created rules.

After all new potential rules are created, it has to be decided which ones will be included in the user's rule base. All rules that have no existing match (both their antecedent set and conclusion match one of the rules) are included in the rule base. If a rule has a match in the user's rule base it is only factored in if it has greater final calculated certainty than the rule that is already present in the model and. In such case the old rule is retired from the rule base. The calculation for the final rule certainty goes as follows:

$$CF_f = \left(CF_{a_1} \cdot CF_{a_2} \cdot \ldots \cdot CF_{a_n} \right) \cdot CF_r \tag{4}$$

where CF_f is the final computed certainty, CF_{a_n} is certainty of the n-th rule antecedent and CF_r is the certainty initially assigned to the rule.

B. Recommendation Calculation

The recommendations for action suggestions are calculated from the rules of a particular user. These are however first adjusted using the user's situation model to decide on their final weight.

The recommendation calculation is a three-step process: (1) compute the importance of situation classes for a particular user situation, (2) modify the rule certainties based on their antecedent certainties, current user's situation and importance of the situation classes, (3) compute the final recommendations based on the present rules with their modified certainties.

1) Computing the importance of situation classes

Not all of the situation classes are important for a particular user and therefore their full inclusion in the computation process degrades recommendation results. To avoid this we have to identify how important a situation class is for a particular user, therefore finding its importance. To achieve this we compare how much is situation distribution from any particular class similar to a possible uniform distribution.

The first part is calculating ratio for any situation in a situation class in possible uniform distribution of situations (situation class ratio):

$$r_{sc} = \frac{1}{|SC|} \tag{5}$$

where r_{sc} is the situation class ratio and $|SC|$ is cardinality of the situation class.

Second part is calculating the real individual situation ratios by comparing the presence of a particular situation in the rule base to the presence of all situations from the situation class:

$$r_{s_m} = \frac{\sum CF_{s_m}}{\sum CF_{s_1} + \sum CF_{s_2} + \cdots + \sum CF_{s_n}} \tag{6}$$

where r_{s_m} is the computed situation ratio for situation *m*, $\sum CF_{s_m}$ is the sum of certainties of rule antecedents with situation *m* and $\sum CF_{s_1} + \sum CF_{s_2} + \cdots + \sum CF_{s_n}$ is the sum of certainties of rule antecedents with any situation from the situation class that is being processed (class of situation *m*). Calculation of the ratio and thus all subsequent calculations are performed only with situations that have presence in the user's rule base. Situations that do not are after computing the *situation class ratio* left out.

The last part is computing the final importance of a situation class by comparing the average distance of situations computed in the previous step to the maximum possible average distance:

$$I_{SC} = \frac{\left(\frac{d_{s_1} + d_{s_2} + \cdots + d_{s_m}}{m} \right)}{\left(\frac{(1 - r_{SC}) + (m-1) \cdot r_{SC}}{m} \right)} \tag{7}$$

where I_{SC} is the final computed situation class importance, $d_{s_m} = \left| r_{SC} - r_{s_m} \right|$ is the distance of each situation ratio from its situation class ratio, $\left(\frac{d_{s_1}+d_{s_2}+\cdots+d_{s_m}}{m}\right)$ is the average distance of situations ratios from their class ratio and $\left(\frac{(1-r_{sc})+(m-1)\cdot r_{sc}}{m}\right)$ is the maximum possible average distance.

The final computed values range from 0.0 to 1.0, where exact 0.0 is achieved when all the situations are present in the user's rule base equally (exact uniform distribution) and exact 1.0 is achieved when only one situation from a class is present in the whole rule base.

2) Modifying rule certainties

In this step the rule certainties are modified based on the match between certainties of their antecedents and corresponding current situation certainties with the situation class ratio I_{SC} also being factored in. The first part is modifying the base rule certainty with regards to the *time sensitivity principle* presented in Section III.

The second part is computing how much each rule antecedent influences the final rule certainty:

$$F_a = 1 - I_{SC} \cdot \left(1 - Min\left(1, \frac{CF_S}{CF_a}\right)\right) \tag{8}$$

where F_a is the final influence factor of an antecedent a, I_{SC} is the computed situation class importance for the situation of antecedent a, CF_a is the antecedent a certainty and CF_S is the certainty of the current situation corresponding to the situation of antecedent a. The resulting factor ranges from 0.0 to 1.0 with 0.0 being achieved when the importance of a situation class is 1.0 (most important) and the situation from antecedent a is not present in the current situation. Factor of value 1.0 is achieved whenever a situation class has 0.0 importance or there is an exact or higher match between the antecedent's certainty and certainty of the corresponding situation in the user's current situation. The minimum function is employed for cases when the certainty of antecedent is lower than the corresponding situation certainty.

The third part is factoring the rule antecedent factors into the final rule certainty:

$$\mathrm{CF}_f = \left(F_{a_1} \cdot F_{a_2} \cdot \ldots \cdot F_{a_n}\right) \cdot CF_r \tag{9}$$

where CF_f is the final rule certainty, F_{a_1}, F_{a_2}, F_{a_n} are factors of antecedents 1 to n and CF_r is the original rule certainty.

This model ensures that the more important are the situations that are present in the rule antecedent set, but missing from the current situation the more is the rule certainty decreased. On the other hand if a situation with low importance is missing, the rule certainty is decreased just slightly.

3) Computing the final recommendations

Computing the final recommendations groups rules from the user's rule base with a matching conclusion (action) and sums their final certainties according to the summation formula proposed by David McAllister's model of working with certainties:

$$CF = CF_a + CF_b(1 - CF_a) \tag{10}$$

where CF_a and CF_b are two positive certainty factors that range from 0.0 to 1.0. If the conditions are held, the formula ensures that the final computed value is also within the range from 0.0 to 1.0 by adding the second certainty factor reduced by the remaining certainty. The parameters in the equation are interchangeable. Also if there are more than two certainties to sum together the order in which they are processed is not important, thus any order gives the same result.

C. Discussion

We designed our method in a way that it can be used in any domain as both situation and action models are domain independent. One important concept in our method is the *time aspect* which allows for a change of the user's action model once her preferences change. The rate at which our method is able change the model depends on the value of the base antecedent certainty cease multiplier introduced in Section III. The lower the parameter is, the faster the model can change. However too low values can cause an adverse loss of rules even when there is no need for the model to change. Thus the most suitable value for this parameter needs to be found and set experimentally.

One important problem when working with loads of situation data or context is selection of the important context. Generally when automated methods work with much more information (be it context or anything else) than necessary, they tend to work incorrectly, because there are too many variables that do not matter and thus obscure what is important and invalidate the results. Thanks to the aggregative model our method performs well in this case and an experiment for such case is presented in section V.

An important note to point out is that our recommendation method and models described in this paper are not sensitive to the level at which the rule certainties are set, but the end-user service or application just needs to be consistent and assign comparably higher certainties to feedback that is a stronger clue of an action suitability and lower certainties when observing weaker clues. Because of this there is also no specific certainty at which the end-user service or application should perform a specific action, but the best time is when the certainty reaches for a significant local maximum in its development through time or is in a significant incline.

V. Evaluation

For evaluation of our method's performance we have implemented a simulation framework and a mobile news recommendation application. The purpose of simulations is to demonstrate basic and some advanced characteristics of our rule-based action recommendation method. The simulation framework works in two stages. The first stage is responsible for generating an environment (situations) of an imaginary user for a set period of time (i.e. one month). The second stage is then responsible for simulating feedback based on the type of chosen user. This can be a user that prefers to read news during mornings, a user who only reads news on showery Monday evenings or a user whose preference changes over time. These are just a few examples and the simulations we have performed with different environments and types of users are presented in the following section.

The situations in the situation classes are represented through symbols that cluster real or discrete values (the purposes and approaches to raw context clustering are presented in [11] and [13]). For example the *TimeOfDay* situation class contains twelve symbols that cluster 24 hours of a day into 2 hour segments.

We first ran a couple of simulations with one and two situation classes of a user whose preference was to read news in the morning from 7:00 AM to 11:00 AM. We then scaled this up to nine different situation classes with the same user preference model. The purpose of this simulation was to testify our method's capability to identify important situation class when there is overload of unimportant ones. The results are shown in Fig.1.

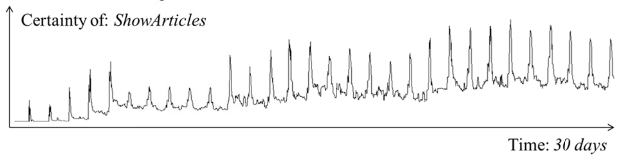

Figure 1. Nine-class, simple preference simulation results.

The number and variety of unimportant situation classes employed in this simulation showed its traces in the overall appearance of the output certainty graph, but the peaks are still significant and thus reliably identifiable.

In next simulations we have kept all the situation classes from our previous simulations but exchanged the simulated user for one whose preference, e.g. reading news only on Monday morning and Friday evening. The purpose of these simulations was to examine characteristics of our method when two situations from two different classes are important only when together. The sample results are shown in Fig. 2.

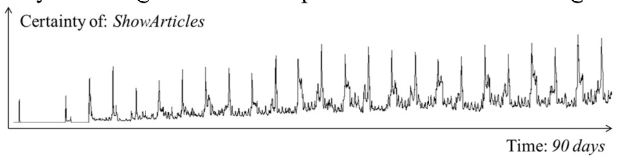

Figure 2. Nine-class, multiple composed preferences.

Even though this time we have two different pairs of important situations within nine different classes our method is still able to reliably identify the important ones after only a few of their occurrences. The gap in the first week's Friday is caused by our simulated user not providing feedback on that day. There are a couple other days where no feedback is provided, however rules from the past weeks are still in force and thus the peaks are present in the final results.

Our last series of experiment simulations tests out the rule retirement model by simulating a user who prefers to read news on Monday morning, but after one and a half month her preferences change to Friday evening. The sample results are shown in Fig. 3.

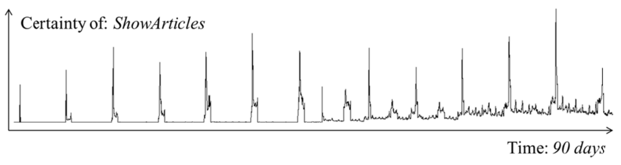

Figure 3. Nine-class, change of preferences.

The results of our last experiment demonstrate our method's capabilities to change when users change their behavior. The spike right after for Monday morning is still around the same size, but is gradually retired as the new rules take over.

In all of the experiments the graph results show significant spikes for suitable moments. The most suitable moments represented by these spikes are then easily identifiable using a simple time series analysis algorithm so our method provides results with both recall and precision close to 100%.

VI. Discussion

The purpose of the experiment simulations was to testify that our rule-based action recommendation method and especially its mathematical models that underline it are correct. We have chosen to perform simulations with their advancing complexity to discover our method characteristics as the data complexity scales up in different directions.

The experiments have shown that our rule time sensitivity model works much more reliably than a simplistic model like the one used with situations work here. In multiple scenarios we have seen that the certainty does not drop to zero even when it is clear that it is not the right situation for a particular action. This is okay, because as we already stated we are interested in significant spikes and possibly inclines of the resulting recommendation certainty.

Even though we have seen our method perform well even when scaled up, the wobbliness is present more significantly when more unimportant situation classes are employed. This is a well-known problem in recommendation where adding more inputs does not always produce better results and it is thus important to identify and use only the relevant context [15]. But again due to the underlying mathematical models of our method we are able to deal with reasonable amount of clutter very well. On the other side to make our method perform its best it is up to the target end-user service or application to define and use the classes that are important in the domain. For example, in case of a real push news service we would employ situation classes that tell us how long ago the last session was, how many articles the user has read during it or how long it lasted.

VII. Conclusions

Our method aims at providing a simple and flexible means for context-aware services and applications that need to provide specific personalized content or perform specific personalized actions autonomously in specific situations. The main strength of our method is

- the use of abstract symbols and
- an intelligent time-sensitivity model.

The use of abstract symbols makes it domain-independent and simple to use while our time sensitivity model makes it possible to recognize and accommodate to the user behavior or preference changes. An example of this is a user who travels during the spring and summer season with public transport making it an ideal situation for reading some news but during autumn and winter commutes by driving her car making it impossible to read any news during that time.

To evaluate our method's performance we have created a simulation framework that we used to run both simple and more complex simulations to ensure the correctness of our method's underlying mathematical models. Besides that, we implemented our method within a mobile news application and, using the work in [6] and [7] for news content recommendation, plan a long term live experiment.

There are a couple of possible enhancements to our method as well. The first one is including a support for collaborative models, where rules could be transferred and shared among similar users just as content is recommended in this collaborative way [8]. This may significantly counter the known cold-start problem for new users, after a sufficient user-base is already active in the target service.

Another extension is to include a base importance factor for situation classes that would help us identify important times for different actions faster and more reliably even with a plenty of situation classes that are not generally important in a particular domain, but may be for some users and thus have to be present.

A possible extension is inclusion of a lightweight ontologies or similar linking schemes as presented in [9], [10] and [12]. This would allow for rule derivation using similarity or parental links among concepts, in this case situations and actions which could also lead to quicker learning and more reliable results. On the other hand such extension would significantly complicate the overall simple method concepts and thus make it less accessible in development of context-aware end-user services or applications. A trade-off can be achieved by only exploiting hierarchical dependencies as presented in [14] as such hierarchy among classes and their situations can be easily employed using just our symbolic representation of situations.

We have designed our method to work in a variety of domains and scenarios. Our method can be used for identification of suitable times for content presentation in mobile device, for automatic sound profile switching, for webpage pre-loading or in any other scenario when there is a need for automated actions performed by an end-user service or application. We believe that our method and frameworks based on it have potential due to their simplicity encourage more services and applications to become context-aware and more intelligent.

Acknowledgment

This work was partially supported by the grants APVV-0208-10, VG1/0675/11/2011-2014, and it is the partial result of the Research & Development Operational Programme for the project ITMS 26240220039, co-funded by the ERDF.

References

[1] P. Korpipää, J. Häkkilä, J. Kela, S. Ronkainen, and I. Känsälä, "Utilising Context Ontology in Mobile Device Application Personalization," Proc. 3rd Int. Conf. on Mobile and ubiquitous multimedia, ACM, New York, 2004, pp.133-140.

[2] P. Ala-Siuru, R. Tapani, "Understanding and recognizing usage situations using context data available in mobile phones," Proc. 2nd Int. Workshop on Personalized Context Modeling and Management for UbiComp Applications, 2006.

[3] M. Roberts, N. Ducheneaut et al., "Scalable Architecture for Context-Aware Activity-Detecting Mobile Recommendation Systems," Proc. 9th IEEE Int. Symposium on a World of Wireless Mobile and Multimedia Networks. IEEE, 2008, pp.1-6.

[4] A. Krüger, J. Baus, D. Heckmann, M. Kruppa, R. Wasinger, "Adaptive Mobile Guides," The Adaptive Web: Methods and Strategies of Web Personalization, LNCS 4321. Springer, Berlin, 2007, pp. 521-549.

[5] F. Bellotti, A. Gloria, R. Montanari, N. Dosio, D. Morreale, "COMUNICAR: designing a multimedia, context-aware human-machine interface for cars," Cognition, Technology & Work, vol. 7, no. 1, Mar. 2005, pp.36-45.

[6] J. Suchal, P. Návrat, "Full Text Search Engine as Scalable k-Nearest Neighbor Recommendation System," IFIP Advances in ICT. Artificial Intelligence in Theory and Practice, 2010, pp. 165-173.

[7] M. Bieliková, M. Kompan, D. Zeleník, "Effective Hierarchical Vector-Based News Representation for Personalized Recommendation," Computer Science and Information Systems, vol. 9, no. 1, 2012, pp. 303-322.

[8] A. Soller, "Adaptive Support for Distributed Collaboration," The Adaptive Web: Methods and Strategies of Web Personalization, LNCS 4321. Springer, Berlin, 2007, pp. 573-595.

[9] A. Costa, R. Guizzardi, G. Guizzardi, J. Filho, "COReS: Context-aware, Ontology-based Recommender system for Service recommendation," Proc. CAiSE'07 Workshop on Ubiquitous Mobile Information and Collaboration Systems, 2007.

[10] L. Liu, F. Lecue, N. Mehandjiev, L. Xu, "Using Context Similarity for Service Recommendation," Proc. 4th Int. Conf. on Semantic Computing, IEEE, 2010, pp. 277-284.

[11] Y. Zhang, S. Cai, M. Hu, F. Liang, "A Study on the Method of Mobile Content Recommendation Based on Situations," Proc. 3rd Int. Symp. on Comp. Intelligence and Design, Hangzhou, 2010, pp. 23-26.

[12] H. Xiao, Y. Zou, J. Ng, L. Nigul, "An Approach for Context-aware Service Discovery and Recommendation," Proc. 17th Int. Conf. on Web Services, IEEE, 2010, pp. 163-170.

[13] D. Shin, J. Lee, J. Yeon, S. Lee, "Context-Aware Recommendation by Aggregating User Context," Proc. 11th Congress on Evolutionary Computing, IEEE, 2009, pp. 423-430.

[14] G. Biegel, V. Cahill, "A Framework for Developing Mobile, Context-aware Applications," Proc. 2nd Int. Conf. on Pervasive Comp. and Communications, IEEE, 2004, pp. 361-365.

[15] G. Yap, A. Tan, H. Pang, "Discovering and Exploiting Causal Dependencies for Robust Mobile Context-Aware Recommenders," IEEE Trans. on Knowledge and Data, vol. 19, 2007, pp. 977-992.

[16] I. Cantador, P. Castells, "Semantic Contextualisation in a News Recommender System," Proc. 3rd ACM Conf. on Recommender systems, ACM New York, 2009.

[17] K. Yeung, Y. Yang, "A Proactive Personalized Mobile News Recommendation System," Proc. 3rd Int. Conf. on Developments in eSystems Engineering, IEEE, 2010, pp. 207-212.

[18] D. Zeleník, "An Approach to Context Aware Event Reminding. Information Sciences and Technologies," Bulletin of the ACM Slovakia, Vol. 3, No. 2, 2011, pp. 126-130.