Development of Knowledge-Filtering Agent Along with User Context in Ubiquitous Environment

Takao Takenouchi[1], Takahiro Kawamura[2,3], and Akihiko Ohsuga[2,3]

[1] NEC Corporation, 2-11-5 Shibaura, Minato-ku, Tokyo, Japan
takenouchi@bu.jp.nec.com
[2] The Graduate School of Information Systems,
University of Electro-Communications, 1-5-1, Chofugaoka, Chofu-shi, Tokyo, Japan
{kawamura, ohsuga}@maekawa.is.uec.ac.jp
[3] Research & Development Center, Toshiba Corp.,
1, Komukai Toshiba-cho, Saiwai-ku, Kawasaki-shi, Kanagawa, Japan

Abstract. In this paper, we propose combination of Ubiquitous Computing and Semantic Web. Data and services will be annotated even in the ubiquitous devices, and should be connected to the web of the semantics near future. We call it Ubiquitous Semantics, where we would find huge amount of knowledge information, but also find most of them transitive along with user context. Therefore, in order for an agent to meet user's real-time query it is required to efficiently retrieve timely and useful piece of the knowledge from the Ubiquitous Semantics. Thus, this paper shows a knowledge-filtering agent, which quickly responds the query by dynamic classification of the necessary information along with the user context changing in the real world. Further, to evaluate our approach we validate the performance of an application: Recipe Recommendation Agent.

1 Introduction

Semantic Web[1] has gained attention for recent years. As the popularity of Semantic Web, it is gets for an agent to gather enormous knowledge from Semantic Web. Also, Ubiquitous Computing is expected to become much popular. In Ubiquitous Computing world, people can use computers and networks anywhere-anytime and detect everything with RFIDs.

In near future, data and services would be annotated even in the ubiquitous network, and connected to the web of the semantics. We call it Ubiquitous Semantics, which is an extension of the current Semantic Web. Ubiquitous Semantics is different from Semantic Web in the following points.

1. The agent can retrieve huge amount of knowledge from not only the networks but also people, object and places in the ubiquitous environment. However, most of them are *transitive*, which is described in the next section.
2. In the ubiquitous environment, it is necessary that the agent detects user context and responds quickly in order to support the user's behavior in the real world.

T. Enokido et al. (Eds.): EUC Workshops 2005, LNCS 3823, pp. 71–80, 2005.

In short, the agent can get huge amount of knowledge from Ubiquitous Semantics, but it is difficult to meet the user's real-time query. Therefore, it is required to retrieve timely and useful piece of the knowledge from the Ubiquitous Semantics according to the user context.

Thus, this paper proposes a knowledge-filtering agent, which quickly responds the query by dynamic classification of the useful information along with the user context changing in the real world. Here, the knowledge is metadata annotated to somethings, which is represented in a triple form including facts, rules, and ontologies.

The rest of this paper is organized as follows: section 2 describes *transitiveness*. Section 3 proposes the knowledge-filtering agent based on *transitiveness*. In section 4, we overview the architecture of our recipe recommendation agent for evaluation, and validate the performance of the application in section 5. Then, in section 6, we discuss related works, and section 7 concludes this paper.

2 Transitiveness of Knowledge

The knowledge of the current Semantic Web is sort of static such as web pages. However, in the ubiquitous environment, it is necessary to consider the knowledge changing along with the user's real-time context. In other words, the knowledge in Ubiquitous Semantics must be filtered along with the user's time, place, and so on. We call it *transitive* knowledge.

Therefore, we propose a method to classify the knowledge based on the transitiveness and to select a certain size of useful knowledge. This will enable the agent to reason on it efficiently and quickly. In order to classify knowledge based on the transitiveness, we define the following four factors of transitiveness.

First factor is Time. In the real world, there is much knowledge depending on time. Therefore it is important to select useful knowledge based on the time.

Second factor is Place. In the ubiquitous environment, the user would mainly need to know the knowledge related to the present time and place.

Third factor is Occasion. According to the user's current context, it is different whether the user wants to have a response quickly or not. If the user doesn't have so much time, the agent should inference for short time period and respond quickly. Thus, the user's occasion is an important factor to detect the transitiveness of Ubiquitous Semantics.

Fourth factor is Personalization. User's preference is also an important factor for selecting. Therefore, the agent should consider the user's preference.

Here, we take the initials of the four names, and call it TPO+P. In the next section, we describe an inference agent who classifies the transitive knowledge based on TPO+P.

3 Knowledge-Filtering Agent

The agent needs to select useful knowledge in considering transitive knowledge mentioned in section 2 in order for the agent to respond quickly in Ubiquitous

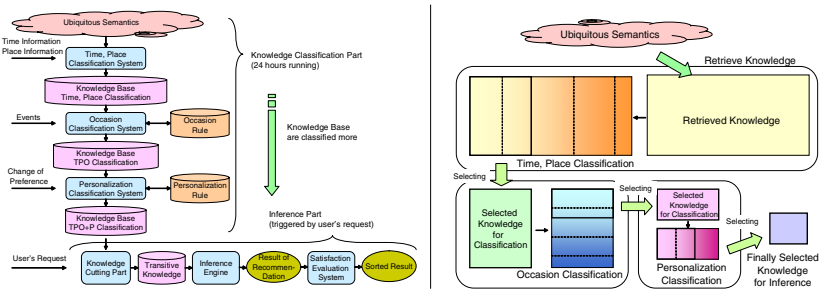

Fig. 1. Archtecture of the knowledge-filtering agent and knowledge filtering

Semantics. Figure 1 shows the architecture of the agent. This agent is mainly composed of Knowledge Classification part and Inference part.

The first part including knowledge base vertically connected from top to down is Knowledge Classification part. This part classifies transitive knowledge. The transitive knowledge are classified in three steps based on Time-Place, Occasion and Personalization. Here, we applied the strategy which is to process the simple classification first to make the size of knowledge passed to the next more complex classification smaller. In addition, each classification is processed independently. Therefore, it is possible to re-classify transitive knowledge quickly in case of the change of user's context (figure 1). Knowledge Classification part is always running, then receives input information of user's position, event, preference and so on. We describe each step of classification process and the example in the next section.

The second part including an inference engine horizontally from left to right is Inference part to provide decision support information for users. This part is executed on the user's demand. Knowledge is already retrieved and classified by Knowledge Classification part. Then, the inference part just selects the useful part of knowledge to pass it to inference engine for decision making support. Finally, the agent calculates the satisfaction ratio from the results of inference, and outputs the sorted results with the satisfaction ratio.

4 Recipe Recommendation Agent

We have developed recipes recommendation agent for evaluation. The agent recommends a recipe, for example, for homemakers to prepare dinner in consideration of sale information and children's preference and so on. Here, we assume the ubiquitous environment as follows. The information of user's position and what merchandise in user's hand is acquired by using GPS and a RFID reader in the portable device. Also, the agent acquires the necessary knowledge from Ubiquitous Semantics in cooperation with information appliances at home and makes the recommendation. Finally, the portable device displays the recommended recipes.

4.1 Overview of Recipe Recommendation Agent

In the followings, we show the process of classification.

1. Time-Place Classification
   Firstly, the agent detects user's position and retrieves knowledge of shops around the user and their sale information and so on, and classifies them based on time and place .
   The agent retrieves not only knowledge of business hours and regular holiday of the shop, but also all knowledge that depends on time such as time-sale, then classifies them.
   Knowledge Classification part is always running. Thus, the knowledge near the user such as local weather information is updated any time.
   The information of shops such as opening hours and position, etc. are assumed to be represented in RDF. In addition, we defined an ontology for shop description (e.g. opening hours, shop holiday, service time and so on). This ontology is defined with DAML-Time ontology[2].
2. Occasion Classification
   For example, consider that the user picks up a food stuff in a shop. In this case, it is thought that the user is interested in that food stuff. Thus, the agent should recommend some recipes using it. So the agent retrieves the name of the food stuff from the attached RFID or QR Code, and selects recipe, and recommends some of them. In Occasion classification process, Jess (Java Expert System Shell) [3] is used as an inference engine. Therefore, Occasion rules are represented in S-expression like Lisp.
   Occasion rules are divided into Common Rules and Condition Rules (table 1). Common rules describe some typical situations and are prepared by the system manager. On the other hand, Occasion Rules describe the situations depends on the user. Therefore, we will develop a tool to select and customize the Occasion Rules in the future.
3. Personalization Classification
   Finally, the agent classifies the useful knowledge based on the user's preference, and calculates a satisfaction ratio for recommendation. The user's

Table 1. Occasion rules

Rule Type	Description	Rule
Common Rules	If an user is in a shop, then Agent cut the knowledge small size (because the user would want to be recommended quickly).	(defrule (user-in-shop) => (cut-knowledge-small))
	If an user is in a shop, and the shop will be closed soon, then Agent re-check the prices (because it will be saved).	(defrule (user-in-shop) (closing-time-soon ?shop) => (check-price ?shop))
	If an user has a car, and is at home, then the user can go shopping by car.	(defrule (user-have-car) (user-in-home) => (use-car-enable))
Condition Rules	If an user is in a shop, and picks up merchandise, then the Agent recommend recipes using it.	(defrule (user-in-shop) (event-have ?item) => (recomend-recipe-use ?item))
	If an user is near a station, and it is the time of going home, then the Agent recommends to buy at the shops in his way home.	(defrule (user-near-station) (time-evening) => (area-shop-station-home))
	If an user is reading a shop handbill, then the Agent recommends to buy at the shop.	(defrule (event-read-handbill ?shop) => (area-shop ?shop))

preference is complex, then it is written as rules. Personalization classification is most time-consumption, so it is executed at the end.
For example, if the user is on a diet, the agent should retrieve knowledge of nutrition information in order to calculate Calorie.
Personalization rules in which the user's preferences are described are specified with URL, and downloaded from the network.
The user's preferences are gathered by questionnaires in advance, then converted to the Personalization rules. We will also develop a tool as well as the Occasion rules in the future, so that the users can describe the Personalization rules by themselves.

4.2 Motivation of Recipe Recommendation Agent

The recipe recommendation is one of the best applications for the evaluation as follows.

First, there is much of transitive knowledge in the recipe recommendation. For example, sale information on each day and time service discount information are transitive knowledge depends on time information. In addition, user's preference depends the physical condition on everyday. Therefore, the preference is also transitive knowledge.

Second, lots of sites show several recipes on the web, and various terms are used. Thus, it is necessary to use the ontology in recipe recommendation agent. For example, "Potato" is necessary as a food stuff for a menu, and a certain merchandise is labeled as "White Potato", then the agent should recognize that "White Potato" is one of "Potato" and can be used as the food stuff of the menu. A food stuff ontology is described in Web Ontology Language (OWL) [4].

5 Implementation

The mobile device of the recipe recommendation agent is assumed as an advanced cellular phone (Smart Phone). However, the evaluation system was implemented in Pocket PC due to the problem of the development environment (figure 2).

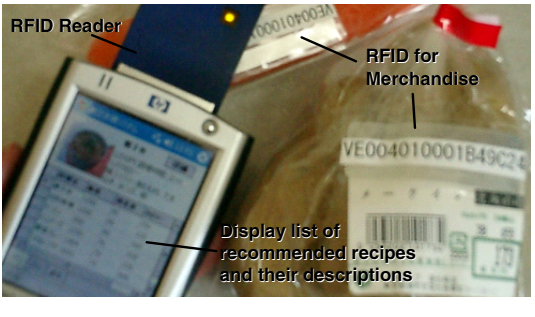

Fig. 2. Mobile device for evaluation

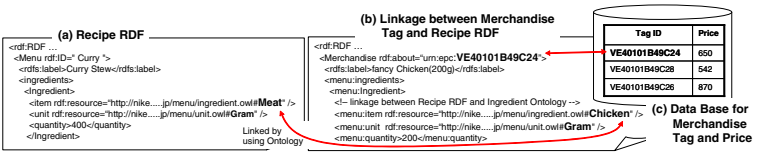

Fig. 3. Knowledge on merchandises and recipes

We use HP iPAQ h2210 in which RFID Reader and GPS Reader are installed. PDA OS is Pocket PC 2003.

We developed the recipe recommendation agent with Java, and installed it to PC instead of a home server. We used Jena 2 Semantic Web Framework [5] to process OWL ontology, and use Jess[3], which is a forward chaining inference engine in Java. Thus, the user's preferences are written in Jess rules. Web services are provided in some of servers by using Axis[6], and the mobile device communicates with the servers via the agent.

Further, we prepared the knowledge which links the merchandise tag with the recipe written in RDFs as shown in figure 3. Information on tag IDs such as prices are stored in merchandise DB, and the knowledge also links the merchandise tag ID to the food ontology.

6 Evaluation

In this section, we evaluate the knowledge-filtering agent. The evaluation was done on the response time and the accuracy.

The knowledge used in the experiment is the real data which is published on a food company [7]. We converted it into RDF and used it as Ubiquitous Semantics. Also, we converted part of the thesaurus which is made by [8] into OWL, and used it as the food ontology.

6.1 Response Time

Table 2 shows the result of the response time comparing classifications and non classification. The results are the averages over 3 times sending the same query. We had an experiment in the occasion that the user is in a shop and thinks of today's dinner then picks up a merchandise at 2 shops, 93 merchandise in each shop, and 40 recipes. The user sends the request to prefer the lowest price.

The agent classifies the knowledge based on shops by "Time-Place classification", and classifies it based on the merchandise which the user picked up by "Occasion classification". Moreover, by "Personalization classification" the agent classifies the knowledge based on preference for lower price recipes.

Table 2 shows the response time improvement. It is caused by classifying the knowledge based on TPO+P and selecting suitable knowledge of recipe according to the users context. This result shows that the response time is getting faster by increasing the classification factors.

Table 2. Response time with classification factors (ms)

		No Classification	TP	TPO	TPO+P
Response Time		11550	6940	4137	291
	Initialization	781	0	0	0
	Non-Transitive Knowledge	471	0	0	0
	TP Classification	0	317	0	0
	TPO Classification	0	0	531	0
	Pre-Inference Process	2033	1088	154	0
	Price Calculation	7968	5261	3211	0
	TPO+P Classification	0	0	0	50
	Inference	297	274	241	241

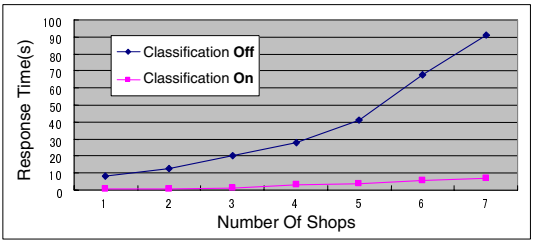

Fig. 4. Response time with knowledge size

By looking at the internal processing time, it is found that the calculation on the total price of the recipe takes so much time. The agent infers with the food ontology like section 4.2 by using Jena and calculates the price. Therefore, as the knowledge of the merchandise and the recipes increase, their combinations increase and the processing time grows. However, as the classification factors of knowledge information increase, the combinations become smaller, and the processing time is getting faster.

In addition, we had an experiment on the response time when the size of knowledge is changed. Figure 4 shows the result of classification. When not classifying it, the combinations of the merchandise of the shop and the food stuff of recipes increase explosively. Thus, the response time is getting worse rapidly. In contrast, the response time is almost stable when classifying it.

Further, to confirm whether the classification order TPO+P is appropriate, we shuffled the order. Table 3 shows the results of the time for each order. It is confirmed by this result that the order of "Time-Place", "Occasion", and "Personalization" is the fastest, and appropriate as the classification order.

Finally, we had an experiment on a processing time for re-classification with 100 recipes, 200 merchandises, and 4 shops. Figure 4 shows the result. When the re-classification is done at "Time-Place classification", it is necessary to do re-classification at "Occasion classification" and "Personalize classification" that are below it. That is, it costs the longest time to re-classify the knowledge at "Time-Place classification". The result shows it takes about 6 seconds to do

Table 3. Reclassification time (ms) with TPO+P order

	TimePlace	Occasion	Personalization	Total
TP,O,P	497	251	448	1196
TP,P,O	538	808	855	2201
O,TP,P	631	1519	581	2731
O,P,TP	644	1519	12939	15102
P,TP,O	2507	240	39587	42334
P,O,TP	648	1385	40899	42932

Table 4. Reclassification time (ms)

Change level	Reclassification Time
Time-Place changed	6023
Occasion changed	5488
Personalization changed	5021

re-classification at "Time-Place classification". However, considering with the current PC spec; Pentium M 1.5GHz, the agent would be able to follow enough the user's real movement.

As the result, the agent can decrease the number of combinations, reduce the size of knowledge for inference, and improve the response time. Also, as the size of ubiquitous semantics increases, the effectiveness of the agent will become higher.

6.2 Accuracy of Recommendation

This section shows that the accuracy does not get worse by cutting out the transitive knowledge. The verification method is as follows. First of all, each tester recommends 20 recipes that he/she wants to eat in some occasions from 100 recipes. Then, the recipe agent recommends 20 recipes. Finally, we examines how many recipes which the agent recommended are matched to his / her recommendations, and calculate the precision ratio of the recommendation.

The occasions are the followings.

Occasion A. In a shop at 3:00 PM, selecting a food stuff for today's dinner. At that time, the user picks up a savory carrot.

Occasion B. Around the train station at 10:00 PM. The user buys a food stuff at a convenience store.

Occasion C. At home at 3:00 PM, thinking of the menu of today's dinner.

Here, the user's preference is "cooking time is shorter", "low calorie", "dislikes fishes". We had an experiment with 230 merchandise, 231 food ontology, and 7 testers. Table 5 shows the result of the average. (No-classification experiment was not able to be done, because the system rised memory shortage error.)

Table 5. Precision ratio (%)

	TP	TPO	TPO+P
Occasion A	43.3	83.3	83.3
Occasion B	35.0	46.7	51.7
Occasion C	30.0	40.0	43.3

The result shows that the precision ratio improves as the classification factor increases. In each classification, obviously unsuitable recipes are cut out, so the precision ratio of the recipe has improved.

In summary, it was confirmed that the agent is able to respond quickly keeping the accuracy by classifying transitive knowledge based on TPO+P.

6.3 Applicability of Other Applications

We have concluded the knowledge filtering agent is effective by evaluating the recipe recomentdation system only. However, the knowledge filtering agent is applicable to other applications. For example, man navigation system which changes the destination along with user's preference is one of target applications. This system has too many possible destination for users goal. Therefore, it is difficult to consider the whole knowledge. Furthermore, it is necessary to recomend quickly in the case of changing the user's plan. Also, there are many kinds of transitive knowledge, such as vacant seat in the theater and so on. For the above reasons, the knowledge filtering agent would be applicable for other systems.

7 Related Works

Several studies have been made on context awareness in ubiquitous environment. [9] aims at providing Web services that fit to ubiquitous computing and proposes an architecture with middle agents who determine the best matched services and location-ontology for ubiquitous computing. However, it doesn't classify the knowledge information from huge amount of transitive knowledge.

[10] proposes a system which infers user's context from the knowledge in Semantic Web and information from sensors, and provides appropriate information to the user. However, it don't classify massive transitive knowledge and not consider the performance.

In addition, several methods of acquiring knowledge to respond quickly are proposed. [11, 12] propose an agent who acquires the knowledge on the Web using caching and planning technology. However, they don't deal with transitive knowledge in ubiquitous environment. Our research aims to respond quickly by classifying transitive knowledge information.

Furthermore, several studies have been made on recipe recommendation. [13] proposes a system to recommend new recipes from some basic recipes by using Case-Based Reasoning and propose a substitute food stuff by using a food ontol-

ogy. However, it doesn't consider the transitive knowledge. If the size of knowledge is large, then it would become necessary to select the useful knowledge.

8 Conclusion

We defined four factors that characterize the transitive knowledge as TPO+P, and proposed the method of efficiently selecting the useful knowledge part from the huge amount of knowledge in ubiquitous environment.

Then, we developed the recipe recommendation agent, and evaluated the response time and the accuracy.

References

1. Tim Berners-Lee, James Hendler, and Ora Lassila. The Semantic Web. *Scientific American*, May 2001.
2. Harry Chen, Filip Perich, Tim Finin, and Anupam Joshi. SOUPA: Standard Ontology for Ubiquitous and Pervasive Applications. In *MobiQuitous 2004*, August 2004.
3. Jess (Java Expert System Shell). Sandia National Laboratories. `http://herzberg.ca.sandia.gov/jess/`.
4. Deborah L. McGuiness and Frank. van Harmelen. OWL Web Ontology Language Overview, December 2003. `http://www.w3.org/TR/owl-features`.
5. Jeremy J. Carroll, Ian Dickinson, Chris Dollin, Dave Reynolds, Andy Seaborne, and Kevin Wilkinson. Jena: Implementing the semantic web recommendations. Technical Report HPL-2003-146, HP Lab, 2003.
6. Apache AXIS. Apache Web Services Project. `http://ws.apache.org/axis/`.
7. Ajinomoto, Co., Inc. Recipe *DAIHYAKKA*. `http://www.ajinomoto.co.jp/recipe/`.
8. Institute of Language Engineering. Thesaurus. Japan, `http://www.gengokk.co.jp/thesaurus/`.
9. Akio Sashima, Koichi Kurumatani, and Noriaki Izumi. Location-mediated service coordination in ubiquitous computing. In *the Third International Workshop on Ontologies in Agent Systems(OAS-03)*, pages 39–46. AAMAS2003, 2003.
10. Harry Chen, Tim Finin, Anupam Joshi, Filip Perich, Dipanjan Chakraborty, and Lalana Kagal. Intelligent Agents Meet the Semantic Web in Smart Spaces. *IEEE Internet Computing*, 8(6):69–79, November 2004.
11. Victor Lesser, Bryan Horling, Frank Klassner, Anita Raja, Thomas Wagner, and Shelley XQ. Zhang. BIG: An Agent for Resource-Bounded Information Gathering and Decision Making. *Artificial Intelligence*, 118(1–2):197–244, May 2000.
12. Victor Lesser, Bryan Horling, Frank Klassner, Anita Raja, Thomas Wagner, and Shelley XQ. Zhang. BIG: A Resource-Bounded Information Gathering and Decision Support Agent. Technical Report 1998-52, Multi-Agent Systems Laboratory Computer Science Department University of Massachusetts, January 1999.
13. Kristian J. Hammond. CHEF: A Model of Case-Based Planning. *AAAI*, pages 267–271, 1986.