A Case-based Reasoning Approach for Personalizing Location-aware Services

Olivier Coutand, Sandra Haseloff, Sian Lun Lau, Klaus David

Chair for Communication Technology
University of Kassel, Germany
{Lastname}@uni-kassel.de

Abstract. The popularization of mobile devices in the last years has led to the development of an increasing number of services. However the technical limitations of mobile devices call for services that require minimal interactions with the user and adapt their behaviors to the user's expectations. Context-awareness has proven to facilitate personalization of services by enabling the adaptation of the service to the user's situation. However, the adaptation is often carried out by using pre-defined rules that only apply to some contexts. In this paper, we present our approach that addresses this limitation in location-aware services by referring to the previous actions of the user. We have developed a metric to calculate the similarity between the current user's location and the previous ones. Based on this metric, our approach provides a personalized service by determining the service behavior expected by the user in the current location. A service, the Call Profiler service, details our approach.

1 Introduction

The proliferation of mobile devices has generated an increasing interest for services accessed anytime and anywhere, making the use of services always more ubiquitous and popular in people's everyday life.

This trend calls for services that are personalized, i.e. that are custom-designed to the user's preferences and environment. The making of personalized services aims to increase their acceptance by users. On the one hand, personalized services can provide users with more accurate information. On the other hand, they respond to users after a minimal set of user interactions. This results in a more effective use of the user's attention and concentration, e.g. by minimizing the number of key strokes typed by the user.

An approach for personalizing services is based on context-awareness. Information about the user's situation is gathered and used to determine how a system must properly react. This way, context-awareness facilitates the development of personalized services by custom-designing services to the user's situation, automatically selecting services and devices settings.

Context-aware systems that enable the adaptation of services to the user's situations have been widely developed [1],[2], [3]. These systems collect sensor data from the user's environment. The data are interpreted and are transformed in order to pro-

vide a human understandable description of the user's situation. Finally this description is used to decide on the actions carried out by the services.

However, most context-aware systems perform adaptation using rules, i.e. conditional expressions where an expected service behavior is related to a particular user's context (e.g. *WHEN I wake up at 7 am, on a work day, at home, THEN play rock music*). However, in contexts where rules no longer apply, no actions can be carried out until the current rules are modified or new rules are added. In order to perform adaptation, users of context-aware systems must then either manually select the action of the service or modify the existing rule to encompass the new context.

This is, nevertheless, a severe limitation. It appears that the success and the proliferation of context-aware services in daily life will depend on their acceptance by end-users. The effort to use context-aware services should be significantly lesser than their benefits to users. These systems, while carrying out service adaptation, must cause minimal user distraction and not constantly request for information to be keyed in to select service behaviors.

In this paper we present an approach that utilizes a case-based reasoning (CBR) method to address the limitation. The service adaptation to be performed is determined from the user's previous utilizations of the respective service. We describe our approach for location-based services, i.e. services whose behaviors can be adapted based on the current user's location. We use a similarity metric to compare the similarity between the current location and the previous ones. The action to be performed by a service is then determined from the action of the most similar location. Our approach is further detailed through the example of the Call Profiler Service.

The paper is structure as follow. In section 2 we discuss the issue of service personalization and we give a scenario featuring the Call Profiler Service. In section 3 we detail the limitation of the adaptation process in the current implemented context-aware systems. In section 4, our approach is described. The similarity metric is further discussed in section 5. In section 6 the specific example of the Call Profiler Service using our approach is described. Finally section 7 presents the further work and concludes the paper.

2 Personalization of services

Admittedly, personalization aims to increase the acceptance of services as it provides users with services that behave as they expect. At the same time the explicit interactions between the service and its user are minimized.

A service can be seen as an object with a well-defined interface. In non-personalized service, the user must explicitly input information to the service via e.g. the GUI or by voice or gesture commands. The resulting response of the service, i.e. the service behavior, is solely produced from these inputs and therefore conforms to the user's expectations. When it comes to personalizing services no (or few) explicit user's inputs are passed to the service. The response of the service is determined from data delivered by an underlying personalization system. Hence the interaction between the user and the service is implicit from the user viewpoint. To achieve this, a personalization process is structured around two tasks: user modeling and system

adaptation. User modeling consists of gathering and storing information about the current user's current goal, plans, background and knowledge. A user model serves as a description of the user and a prediction of how he will behave. System adaptation refers to the process that occurs as mathematical models or analytical techniques are applied to user model data in order to make decisions on the service behavior delivered to the user.

As discussed in [4] future communication systems and services are required to adapt not only to the specific demand of individuals, but also to the individual's situational and environmental conditions. Indeed, individuals have different preferences and settings in different situations.

An approach for having systems adapt to the user's situation is provided by context-awareness. According to [2], a system is context-aware when it uses context to provide relevant information to the user. Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves. The information can be an activity, a location, the time, etc.

When adapting a service, we should distinguish between the gross and the detailed behaviors of a service [5]. A service is designed to provide one or a set of gross behaviors (playing music, enabling chat, displaying messages, etc). When it comes to adaptation, the gross behavior of a service should remain the same. For instance, a music player must remain a music player irregardless of the context. However the detailed behavior of the service can change with the context. The type of music played or the algorithm that is used to construct a play list may be adapted. Behavior changes result then in a parameterization or a selection rather than a change in the code of the function that is provided by the service.

In the following we give a scenario that features a personalized service: the Call Profiler Service. The Call Profiler Service is a service that enables the user to manage incoming calls on his mobile device. The service provides two types of behaviors. It can notify the user of an incoming call via several modes, e.g. loud or low ring tones, vibration, etc. The service can also block incoming calls and divert them to mailboxes, e.g. a mailbox where callers are presented with a message asking them to call back later, or a mailbox offering them to leave a message. This service is context-aware. Indeed, the service behavior is selected according to the user's preference in his current context.

Paul is a researcher and he is currently involved in many projects. Therefore he receives many phone calls every day from different people on his mobile device. Mobile phones have facilitated people's communication means, enabling people to be reached easily. But it has also a serious drawback: a phone call can interrupt and disturb the phone's owner at any time. To avoid such an inconvenience, Paul has started using the Call Profiler Service.

Paul has been using the Call Profiler for some weeks now and this afternoon he has an important meeting at the university. As he arrives in the morning, he decides to meet up with a colleague to give the final touches to the presentation he holds in

the afternoon. He joins his colleague in his office and together the work on the presentation. When someone tries to reach him few minutes later, the call is block and switched to Paul's mailbox. Indeed, Paul does not wish to accept calls when he is in a colleague's office. On the way back to his office, in the corridor of the department, he is notified about the blocked call. In the corridor, Paul wants to be reached again. During the day, the Call Profiler Service continues working. At noon he receives another call in the cafeteria, whereas the two callers that tried to reach him, while Paul is in the meeting room, received a message asking them to try again later.
In the evening Paul goes out to town, and continues using the service. In the tram, he receives a call from a friend to fix an appointment. His phone has been switched to the vibration mode, since Paul usually does not want to disturb people around him when using public transportations. Later, while Paul is having a drink with friends in a pub, he is notified of two incoming calls by a louder ring tone.

The scenario describes how a user can take advantage of a personalized service. In any place Paul is, the Call Profiler provides him with the behaviors he wants. When Paul is in the cafeteria, he requests the service to pass all incoming calls, whereas when he is in the meeting room, he expects calls to be blocked. Obviously the service could behave another way with another user.

According to Paul's location, one behavior offered by the service is selected. The selection is driven by Paul's preferences in each location.
In the following we discuss how current context-aware systems address such a scenario.

3 Current Context-aware systems

Since [1], context awareness has attracted quite some attention for computing applications. Many context-aware frameworks have been developed to provide a middleware layer to adapt applications and services to the user's current context. As pointed out in [6], issues addressed by these frameworks can be classified into three categories: gathering context; interpreting and managing context; and adapting the service behaviors.

Context gathering approaches aim to collect raw context data from heterogeneous sensors and augment these data. Interpreting context consists of transforming raw sensor data into human understandable high-level contexts. High-level contexts are composed of data from different context data sources or of different context types (location, temperature, etc). Interpretation is carried out by using predefined rules as in [7], [8]. A slightly different approach is presented in [3], where context is modeled and interpreted based on ontologies. Following this approach context is represented as predicates written in OWL. Context interpretation is performed by a context reasoning engine that supports RDF-S and OWL reasoning and general rule based reasoning.

Service behavior adaptation is performed by determining the adequate reactions to events and interpreted contexts. Mobile services adapt their behavior to the context by applying predefined rules. In [3] actions are triggered by a set of rules whenever the

current context changes. Service developers write pre-defined First Order Logic rules and specify what method is invoked when a condition becomes true. All the rules are saved in a file and pre-loaded into the "context reasoner" component of the system. A similar approach is carried out in [8] where an inference engine component based on CLIPS which is able to select rules to fire is used. In [9], a customization (i.e. personalization) model is presented that enables the context-based adaptation of web application services. Context is used to trigger the customization as soon as a context change occurs. To specify the customization, ECA (Event/ Condition/ Action) rules are used. A rule consists of event and a condition parts that together specify the context the rule is applied to. The event part determines the change that triggers the rule (e.g. change of bandwidth). The condition part is evaluated as soon as the rule is triggered by an event. It determines whether an adaptation is required. For example, a condition would be that the bandwidth falls below a minimum. Eventually, the action part activates the adaptation of the service.
Current context-aware frameworks use rules to adapt services. This does not allow performing actions when the user's context have not been envisioned beforehand Even though it may be easy to characterize and describe some of the user's situations, for which a specific service behavior can be requested, it is arduous to exhaustively define all user's contexts where the user expects a service behaviors. Then, in contexts where the rules (no longer) apply, no action can be carried out until the current rules are modified. Rules can be newly input by the user. But this is not acceptable, since the aim of context-awareness is to reduce the cognitive load on the user and preventing the user to type of series of key strokes when using the system.

4 CBR for adapting location-based services

In this paper we present an approach to overcome the limitations of current implementations of context-aware systems that perform rule-based adaptation of services.

A rule is a simple scheme for expressing how the user's context and an action, which would be carried out by the system are associated. Any other association (e.g. a cross reference in a database, etc) that would have been defined beforehand by the user or the service developer would impose exactly the same limitation. Performing such an association between a user's context and a service action implies that most of the user's contexts must be foreseen. However, this is an exercise that seems very difficult for human beings. Envisioning and describing all the possible contexts a user is going to experience can only been managed for a well specified scenario. Rather, human beings have a tendency to solve problems as they are confronted. To do so, they prefer to refer to their previous experiences and examples, and try to come out with an acceptable solution.

We follow a similar approach to support the personalization of services. We propose to determine the way a service is adapted by comparing the current user's context (the problem) with his past experiences. When a past user's current context is found to be similar to a previous one, the same adaptation scheme is applied. Indeed,

it is most likely (but not mandatory though) that the user will behave analogously in similar contexts.

Context, as defined [2] and as we envision it, is a manifold and complex concept. Many elements can come into picture if ones tries to characterize a user's context, e.g. location, activity, nearby people, nearby objects, mood, environment characteristics (temperature, lightening), humidity, etc. This makes context potentially very difficult to deal with. However, all context-aware services do not have to be adapted by taking into account all these elements for the context. Some context elements may be more relevant for adapting services than others. For example, the above scenario features the personalization of the Call Manger service, where the user's location appears as a critical context element, allowing personalization to be carried out based on the user's location only. Therefore, in the remainder of this paper, we discuss the adaptation of the Call Manager service based on this context element only.

Comparing past user experiences is referred to in the literature as CBR – Case-based reasoning - [10]. Case-based reasoning uses a memory of relevant past cases to interpret or solve a new problem case. In case-based reasoning a problem is solved based on similar solutions of past problems. Case-based reasoning utilizes the specific knowledge of previously experienced problem situations, referred to as cases.

In our approach, a case refers to a user's location, for which the user's expectation is known. Location typically represents the site, i.e. the point in space, where the user of the application is located. As discussed in [11] there are different *plausible* and *correct* ways to answer the question: *where is A. located?*, and consequently different ways to characterize a location. In this paper we consider the following properties as relevant for characterizing a site: *the absolute position (name space)* (with respect to a coordinate system), *its class or function (named class)*, *ownership* (*subject static space*), and the *attention*. These properties are further discussed in section 5.

A distinction is also made between known and unknown locations. A known location is defined as a location where the system knows what the user expects. Indeed, in this location, the user may already have used the service, or may have specified which service behavior he wants. As such the known locations constitute the user's previous cases. Inversely, an unknown location is defined as a location for which the user's expectation has not been expressed.

Our work is driven by the assumption that in similar locations the user demands towards a service are likely to be identical. Thus, in order to determine the service behavior, we propose to analyze the similarities between an unknown location and other, known locations. By considering the similarities, we can then retrieve the locations that best matches. Consequently, we deduce the service behavior of the unknown location from the service behavior of the known location

In order to achieve the above goal, several processes have to be consecutively carried out. Now we discuss how these processes can ease the adaptation of location-aware services. They follow the processes of the CBR cycle described in [10].

- *Identifying the various service behaviors.* The first task consists of identifying the service behaviors that are offered by the service. To personalize the service, a selec-

tion has to be made between these behaviors according to the user's location. The behaviors offered by a service are specified in the service description.

- *Identifying the current case (user's location).* The user's location characterizes the current problem (case). Data are gathered from the user's environment e.g. from sensors to obtain a description of the user's location. When different types of sensors are jointly used, for example to determine the user location both indoors and outdoors, the component also processes data to provide a uniform and system-manageable description of the user locations. Indeed location data must be represented the same way, in order to ensure that the similarities can be further calculated.

- *Retrieving the most similar case.* To determine the expected service behavior in the current user's location, a similarity metric is used. The metric is detailed in the next section.

- *Reusing the information and knowledge of that case to solve the problem.* When the most similar location has been found, its differences with the current location are abstracted away. In similar locations the user demands towards a service are likely to be the same. Then the same service behavior is displayed to the user being in his current location.

- *Revising the proposed solution.* After the service behavior has been displayed to the user, he has still the choice to interrupt and request for a behavior's change. The interruption from the user is considered as a *negative* user feedback, while no interruption is considered as a *positive* user feedback. In the case of a *negative* feedback and depending on the service, the user may select a service behavior, or he may give a feedback to a newly proposed service behavior.

- *Retaining the case.* After a *positive* user feedback, the new case is stored along with the previous ones. The new case consists of the user location and the service behavior the user has given a *positive* feedback to. Hence, an unknown location can then be added as a known location.

5 Computing similarities

Similarity is a cornerstone in case-based reasoning systems. It serves as a principle to select the past experiences and solve the problem of the new case [12].

They are two ways to obtain a measure of similarity between objects. First similarity can be obtained from the objects. For example a marketing survey may ask respondents to rate pairs of objects according to their similarity [13]. In the same way, the user could be asked for rated all the potential locations in his environment and this way rate the similar locations, i.e. locations for which they expect an application to behave similarly. Alternatively, measures of similarity may be obtained indirectly from a vector of measurements or characteristics describing each object.

It is then important to define what we mean by similarity, so that we can calculate formal similarity measures. For this it is important to remember that our goal is to determine the behavior of a service in a location for which the user did not specify his

expectation(s). Hence, a measure of similarity between two locations must not necessarily depict how "physically similar" these two locations are. Rather a measure of similarity depicts how similar the expectations of the user towards the application are for the two locations. As a consequence, the similarity measure aims to determine how similar location's characteristics that drive user's expectations are.

Often, location is regarded as a concept only characterized by the physical coordinates with respect to a coordinate system. The physical coordinates are given by means of sensors (e.g. GPS for outdoor locations and Bluetooth for indoor locations). However, physical coordinates, though relevant for characterizing a location, are not the only characteristics we consider when defining a location. Indeed, when deciding on a service behavior, the user is influenced by his current location. However, here the location refers to the concept as perceived by the user. The position in space characterized by some coordinates is important. But the user can also perceive the function of the location (i.e. he perceives he is in the meeting room and as such that he needs to switch off his mobile devices), as well as other characteristics.

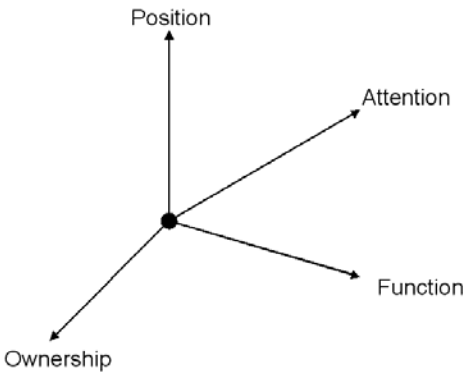

Figure 1: Location dimensions

Figure 1 depicts the dimensions of locations we consider in this work.

- *Function.* The function of the location refers to the purpose for which the location exists and is used. It corresponds to the question: "What is the location for?" As such, the function is described by the name of the location. For example, the site where I work is my office, while the site where I sleep is my bedroom.

- *Position.* The position of the location indicates the geographical situation (place) of the location. It is characterized by the location's coordinates defined in a coordinate system that is associated to a location model.

- *Ownership.* The ownership characterizes the individual that is used to occupying the location. For example, John's office is "owned" by John since it is the office he has been assigned to. The ownership dimension can have a great influence on the choice of service behavior. Indeed, the user may act differently whether he is in his

office, the office of one of his colleague, or the office of someone he has no social relationships with.

- *Attention.* Finally, attention determines how much attention the user can give to a service in the location. This dimension is relevant for site where interactions with a service must be restricted, since it may endanger the user.

These four dimensions are the only ones we consider in this work; even though, other dimensions arising from specific scenarios might also be perceived.

Consequently a location can be represented as a vector whose features (X^i) are the values assigned to each dimensions.

$$loc^i = \left(X^i_{Position}, X^i_{Fucntion}, X^i_{Ownership}, X^i_{Attention} \right) \quad (1)$$

It is important to note that the perception of a location may be specific to an individual. It can therefore happen that values assigned by two persons to the dimensions of a same location differ. For example, a location can be perceived by someone as "office" whereas it is a "secretariat" for someone else.

Having defined how a location is characterized, we now introduce how the similarity between two locations is computed.

Locations can be seen as points in a space directed by the dimensions (as represented in Figure 1). The similarity (or dissimilarity) is measured by the metric distance of the two locations.

The distance expressing the similarity between two locations is calculated as the weighted city-block distance between their feature vectors, as follows:

$$Sim(loc^i, loc^j) = \sum_k \alpha_k \cdot sim_k(loc^i(X_k), loc^j(X_k)) \quad (2)$$

Where sim_k is the similarity metric related to the k^{th} feature of a location vector, and α_k the weight assigned to sim_k.

These metrics are discussed below.

- *Similarity for the function dimension.* The function of the location refers to the purpose for which the location exists and is used. Typical examples of features for this dimension are: office, corridor, kitchen, etc. The similarity between two of these values depends on how closely related and interchangeable the concepts are. Indeed, the similarity between the function: office and the function: secretariat is greater than the similarity between the function: office and the function: corridor. This similarity is strongly related to the semantic of the concepts.

In computer science, an ontology serves to express the semantic of concepts and, hence, provides a representation of a domain. It specifies the concepts that are part of the domain and their relationships. Among the many ontologies covering various domains, one major work is WordNet [14]. WordNet provides a description of the semantic relations between words in the English language. Its design inspired psycho-

linguistic theories of human lexical memory makes WordNet relevant for our purpose of similarity computation.

To compute similarities between concepts in WordNet, we use the similarity measure proposed in [15]: the similarity is based on the shortest path between two concepts and scales that value by the maximum path length found in the hierarchy in which they occur.

- *Similarity for the position dimension.* For this similarity we first calculate the physical distance between the two locations. Computing this similarity is particularly relevant when locations are relatively close to each other. The increase of the distance beyond a certain value does not make locations be more different from each other. Therefore beyond a distance Δ, the similarity measure is set to 0.

The similarity metric for the position dimension is then:

$$Sim_{Position}(X^i, X^j) = \begin{pmatrix} 1 - D/\Delta & , for \quad D \le \Delta \\ 0 & , for \quad D > \Delta \end{pmatrix} \tag{4}$$

$$with \quad D = \left| X^i_{Position} - X^j_{Position} \right|$$

- *Similarity for the attention and ownership dimensions.* Similarities for the attention and ownership dimensions are computed in the same way. We take as an example the ownership dimension. For any location loc^i the feature $X^i_{Ownership}$ is an element of defined list *{private, family, colleague, …}*.

The function *f* maps the list into a set of positive integers $\{a_1, a_{2,\ldots}\ a_n\}$, where $a_i \neq a_j$ for $i \neq j$.

For example, the function *f* maps *private* to a_1 (=1), *family* to a_2 (=2),etc.

Then the similarity metric is determined as follows:

$$Sim_{Ownership}(X^i, X^j) = 1 - \frac{\left| f(X^i_{Ownership}) - f(X^j_{Ownership}) \right|}{\max_{(k,l)\in n} \left| a_k - a_l \right|} \tag{5}$$

Where $f(X^i_{Ownership})$ is an integer assigned by *f* to the ownership feature of location loc^i and $\max_{(k,l)\in n} \left| a_k - a_l \right|$ is the greatest difference between two of the positive integers.

6 The Call Profiler Service

In the previous section we detail our approach for personalizing services and overcoming the limitation of current context-aware systems in adapting services, as de-

tailed in section 3. The approach is further discussed with the example of the Call Profiler Service.

As highlighted by the scenario in section II, the Call Profiler Service provides several functionalities that enable the management of incoming calls for a mobile phone's user. The Call Profiler can enable the blocking of incoming calls and can divert them to one of the user's mailboxes, (e.g. professional vs. private mailboxes). Alternatively the service passes the call and notifies the user with a specific ring tone or with vibration.

The personalization of the Call Profiler is enabled by a middleware system that implements our approach. In order to enable the determination of the service behavior and therefore to provide a personalized service, the system must identify the behaviors offered by the service. At the time of the first use, the service provides the system with a description of its capabilities, where the available service behaviors are described.

Existing approaches for service descriptions have already provided means for semantically describing the capabilities of services [16]. However, a semantic description of the service behavior is not requested in our approach. Indeed, personalization is enabled by selecting the service behavior expected by the user in his current location. The service behavior is determined from the most similar location. Therefore the system must only be able to identify it from the list of behaviors offered by the service. A semantic description would be needed if relationships between various service behaviors had to be inferred.

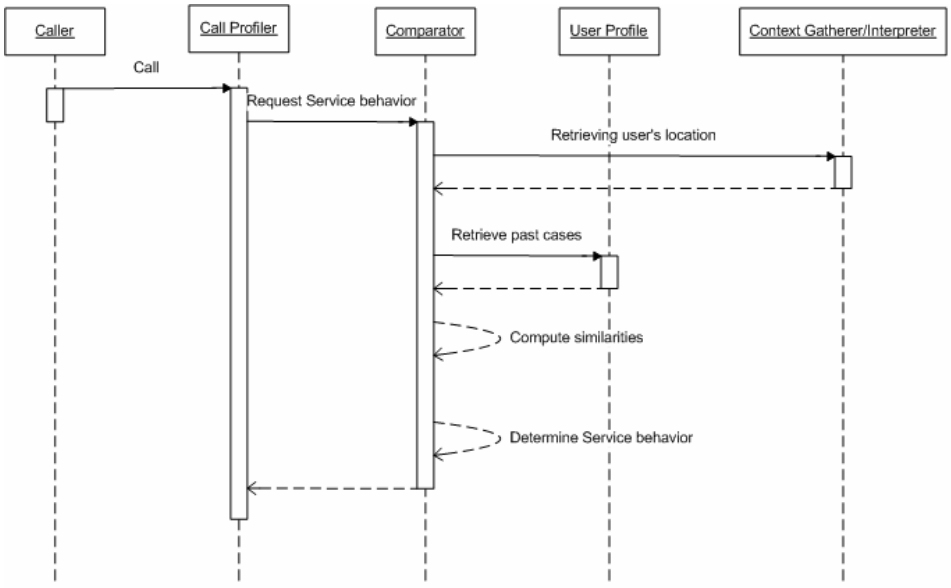

Figure 2: Interactions with the Call Profiler Service

Figure 2 illustrates the interactions of the system's components that cooperatively determine the expected service behaviors in a location. The system is activated when a call reaches the user's device. The *Call Profiler*, which has no clue on how to handle the call, requests at the *Comparator* component the service behavior expected by the user. The *Comparator* first retrieves a description of the current user's location from a *Context Gatherer and Interpreter* component, which constantly maintains an up-to-date description of the user's location. Then, the *Comparator* retrieves known locations from the *User Profile*. Since, the *User Profile* potentially contains many known locations, only those that are willing to present some similarity with the current location have to be retrieved (we do not detail the mechanism here). The *Comparator* calculates the degree of similarity of each identified known location with the current one by applying the similarity metric, described in section V. The most similar one (and potential identical one) is then used to determine the expected service behavior. Finally, the determined service behavior (e.g. block call, switch to mailbox #1) is expressed in an xml-based format and is passed to the *Call Profiler*.

It is worth noting that the personalization, i.e. the determination of the expected service behavior, is not performed by the Call Profiler service per se. Rather, the service applies the service behavior that is determined by the underlying system.

7 Conclusion and further work

Personalization aims to provide users with services that match their preferences. User preferences often differ according to the current user's situation. Context awareness allows the development of personalized services that automatically adapt to the user's situation. Therefore it reduces user's attention when interacting with the service.

Current context-aware frameworks have limited capabilities when it comes to dealing with unknown contexts, i.e. contexts where the user demand has not been explicitly expressed. These frameworks use rules that associate a context to an action to be performed by the service. However, in contexts where the rules (no longer) apply, no action can be carried out until the current rules are modified.

In this paper, we present our approach that overcomes this limitation by using a case-based reasoning method. The approach is discussed through the example of the Call Profiler service. The expected service behavior in the current user's location is determined from his past actions. The similarities between the user's current location and the locations where the user already used the service are computed. The most similar location is identified and the requested service behavior at this time is proposed to the user. To compute similarities between locations, we propose a metric that takes into account various location's dimensions (function, position, ownership and attention).

In order to demonstrate our approach and address some technical questions we will next carry out experiments with the Call Profiler Service. First, the similarity metric must be validated against some real-life scenarios. These scenarios must also enable

the investigation of the user interactions with the service, e.g. stating how many experiences are needed for the system in order for the system to make the first determination (cold start problem) as well as the handling of user feedbacks.

References

[1] B. N. Schilit, N. I. Adams, R. Want, "Context-Aware Computing Applications", In Proceedings of the Workshop on Mobile Computing Systems and Applications, Santa Cruz, CA, December 1994, pages 85-90.

[2] A. K. Dey, "Providing Architectural Support for building Context-Aware Applications", PhD thesis, College of Computing, Georgia Institute of Technology, December 2000.

[3] T. Gu, H. K. Pung, D.Q. Zhang, "A middleware for building context-aware mobile services", In Proceedings of IEEE Vehicular Technology Conference (VTC), Milan, Italy, May 2004.

[4] S. Arbanowski, P. Ballon, K. David, O. Droegehorn, H. Eertink, W. Kellerer, H. van Kranenburg, K. Raatikainen, and R. Popescu-Zeletin, "I-centric Communications: Personalization, Ambient Awareness, and Adaptability for Future Mobile Services", in IEEE Communications Magazine, September 2004, pages 63 – 69.

[5] S. Dobson and P. Nixon, "More principled design of pervasive computing systems", Engineering for Human-Computer Interaction and Design, Engineering Human Computer Interaction and Interactive Systems, Joint Working Conferences EHCI-DSVIS 2004, Hamburg, Germany, July 2004, pages 292-305.

[6] M. Baldauf, S. Dustdar, F. Rosenberg, "A Survey on Context Aware Systems. International Journal of Ad Hoc and Ubiquitous Computing", forthcoming, 2006.

[7] M. Roman, C. Hess, R. Cerqueira, A. Ranganathan, R.H. Campbell and K. Nahrstedt. "A middleware infrastructure for active spaces", IEEE Pervasive Computing, 1(4), 2002, pages 74–83.

[8] G. Biegel, V. Cahill, V, "A framework for developing mobile, context-aware applications", in Proceedings of the 2nd IEEE Conference on Pervasive Computing and Communication, 2004, pages 361-365.

[9] G. Kappel, W. Retschitzegger, E. Kimmerstorfer, B. Pröll, W. Schwinger, Th. Hofer, "Towards a Generic Customisation Model for Ubiquitous Web Applications", in International Workshop on Web Oriented Software Technology IWWOST'2002, Málaga, Spain, 2002.

[10] A. Aamodt, E. Plaza. "Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches", in Artificial Intelligence Communications 7, no. 1, 1994, pp 39-52.

[11] S. Dobson. Leveraging the subtleties of location, in Proceedings of Smart Objects and Ambient Intelligence, 2005, pp 189-193.

[12] M. Richter, "Introduction", pp1-15, in Case-Based Reasoning Technology, from Foundation to Applicationm, in In Lecture Notes in Artificial Intelligence, Springer-Verlag, 1998, ISBN 3-540-64572-1

[13] D. J. Hand, H. Mannila, P. Smyth, "Principles of Data Mining", The MIT Press, 2001, ISBN 026208290X.

[14] http://wordnet.princeton.edu/ (last visited 26.05.2006)

[15] C. Leacock, M. Chodorow, "Combining local context and wordnet similarity for word sense identification", in C. Fellbaum (Ed.), Wordnet: An electronic lexical database, MIT Press, 1998, pages 265-283.

[16] http://www.daml.org/services/owl-s/1.0/ (last visited 26.06.2006)