A Unifying Framework for Situation Identification Methodologies

Olga Murdoch and Paddy Nixon

CLARITY: Centre for Sensor Web Technologies,
University College Dublin,
Ireland
olga.murdoch@ucdconnect.ie,paddy.nixon@ucd.ie

Abstract. Situation identification methodologies in pervasive computing aim to abstract low level context data into more meaningful high level contexts for use by context-aware application developers and users. Many situation identification techniques have been developed and successfully applied to limited scenarios. It is not possible to apply a single technique to a wide range of diverse applications. An ideal solution would allow the combination and and interchanging of techniques as appropriate. We propose a unifying framework for existing situation identification methodologies that will automatically select the best techniques for an application.

1 Introduction

Pervasive computing involves building highly responsive and self-adaptive systems which aim to increase the integration of technology into the fabric of everyday living [1] e.g., smart spaces. Context in pervasive systems comprises any information that characterises the situation of any person, place, or object that is considered relevant to the interaction between a user and an application [2]. Context data may be gathered from many sources including sensors deployed in the environment (sensing light or sound levels for example), digital sources such as calendar information, and static information such as a person's name, date of birth, etc. Sensors are inherently uncertain, providing an approximation of the real world rather than truely reflecting the variables they sense. They also may become damaged leading to inaccurate readings. This means that any action we decide to take may be wrong or may be invalidated by new information [3].

Individually, context data does not yield much information about the environment but when accumulated gives rise to higher-level contexts such as events, activities and situations. A higher-level context is the interpretation of context data, i.e., it describes a high level of abstraction in the environment in which the sensors are deployed. We refer to this process as situation identification. There is no general solution to the problem of situation identification for pervasive systems. Existing research provides us with an insight into how situation identification techniques can perform with high accuracy for limited scenarios [7–9]

but it is not possible to utilise a single technique for a broad range of diverse problems [4–6].

Research on situation identification methodologies for contextual systems is still in its early stages. The relationship between techniques and their performance given diverse context-aware applications is little understood. We propose a decision framework for the selection of situation identification techniques by determining the characteristics of distinct scenarios for context-aware applications and how those characteristics relate to the performance of situation identification techniques. This will form the basis for a design time automatic selection tool whereby the most suitable situation identification techniques will be chosen for the context-aware application developer based on a set of application specific inputs.

We discuss related work in section 2. In section 3 we characterise applications and describe their relationship to existing situation identification techniques. We present the architecture for our framework in section 4. We conclude with a summary of this research and an outline future work.

2 Related Work

2.1 Situation Identification Methodologies

Situation identification methodologies abstract low level context data into more meaningful high level contexts. Many techniques have been developed and successfully applied to specific scenarios.

A **Bayesian network** (BN) is a directed acyclic graph that can learn the relationships between contexts. It is a probabilistic model allowing higher level contexts to be inferred based on prior and conditional probabilities. Leaf nodes in the graph represent sensed contexts and internal nodes and the root node represent higher-level inferred contexts. BN's have been used for device location and activity recognition [12, 7] .

Neural networks have been used as a way to learn a mapping between context data and higher level abstractions. They comprise a network of weighted values and transfer functions used to derive output values. While they have been proved useful in identifying links between contexts and abstractions such as events or activities, they use a "black-box" learning process which is hidden from the user/application and developer. Applications of Neural networks include user location detection [13] and activity recognition [8].

In **Support Vector Machines** (SVM) classification is performed by construction of an N-dimensional hyperplane that seperates training samples. Qian et. al., [14] use SVM's to identify activities using video data.

Case Based Reasoning (CBR) is a way of solving new problems based on previous similar solved problems. Each case consists of a problem (set of features describing current state of environment) and a solution (correctly identified situation). CREEK is a case based reasoning architecture which has been used to identify situations of hospital staff [10, 11] . Knox et al. [15] used CBR to identify situations in a home environment.

Hidden Markov Models (HMM) are probabilistic models consisting of hidden and observable variables at specific time-steps. A HMM is a finite set of hidden states whose transitions are governed by transition probabilities and each state is associated with probability distribution. An observable outcome/evidence can be generated from a hidden state according to its probability distribution. HMMs have been used to infer user situations from wearable sensors [9] and for recognising activities in the home [23].

Decision trees model decisions and consequences in order to predict an outcome based on a set of input variables. They have been applied to activity detection in [20] and [21].

Discussion Each of the described methodologies have been developed and tested by researchers focusing on specific application areas. Despite techniques successfully identifying situations in many of the evaluations performed, the same technique may not perform as well given another application setup. Also, techniques are not directly comparable unless identical evaluation setups and datasets are used [17]. As previously discussed, it is not possible to take one technique and solve the situation identification problem for a diverse range of applications. An ideal solution would allow us to interchange techniques on demand. Current research does not give a decisive description of what each technique is best at and what application characteristics they are best suited to.

2.2 Middleware and Frameworks

The fact that there is no one general solution to situation identification for context-aware applications has been taken into account in some systems where multiple techniques are employed. GAIA [4, 24] is middleware for context-aware application development that supports the use of fuzzy logic, probabilistic logic and Bayesian networks. A development framework is provided within which an application developer can define rules and build a BN according to their needs. Similarly the CAMUS middleware [5, 25, 26] supports the use of Bayesian, ontological and rule-based reasoning about context data. Dargie [6] discusses the benefits of incorporating multiple probabilistic situation identification techniques. Making multiple techniques available to a developer is very beneficial since one single technique will not meet every developers needs. Without decision aids the context-aware application developer chooses situation identification techniques based on intuition and experience. This is not an ideal solution.

Przybilski et. al., [18, 19] propose a reasoning framework for context-aware applications. They suggest an architecture encorporating multiple situation identification (reasoning) techniques in order to support a more diverse range of applications. The focus of the work presented however, is on the distribution of such a framework.

Discussion Research in situation identification does not identify the links between techniques and how and why they perform differently for diverse applications. A method for the automated selection of optimal situation identification

techniques for applications has not, to our knowledge, been implemented or proposed. Based on our research in the area an automated selection framework for situation identification methodologies is needed but such a tool has not yet been developed. We propose a unifying framework for existing situation identification methodologies. We define the important characteristics of context-aware application scenarios and use their relationship to the performance of situation identification techniques as a basis for automatic technique selection. This will allow a unifying framework to determine the situation identification technique(s) that are best suited to the characteristics of a diverse range of applications.

3 Understanding the Relationship Between Applications and Optimal Situation Identification Methodologies

To automatically select situation identification techniques we must first understand the characteristics of context-aware applications. We must identify the important characteristics that differentiate between applications and those that will have an impact on how a situation identification technique performs. By analysing evaluations of situation identification techniques over a range of scenarios we can begin to understand the links between scenario characteristics and technique performance.

3.1 Application Characteristics

Previous research has catalogued context-aware applications in terms of the types of context used (activity, identity, location, time) and context-aware features of the application (presentation, automatic execution, tagging) [2]. For the purpose of this research and based on analysis of existing context-aware applications and datasets we provide classifications on two levels, physical deployment and specific application scenarios. We choose this level of separation as the physical deployment is relevant to all aspects of the application. On the other hand, applications can often be split into independent tasks whose characteristics differ greatly, so it is necessary to characterise such scenarios individually.

First however, we define types of context used in context-aware applications. There have been other classifications used but in our version we expand on that of Dey et. al., [2]. As well as identity, location and time, we also include device, environment, and user-profile. In our classification we have left out activity since this is a more abstract concept. We use device to describe contexts related to specific objects, for example when state change sensors detect the use of household appliances. By environment we mean the information sensed such as temperature and light, and by user-profile we refer to information gathered about the user from sources such as calendars, emails, or instant messengers.

Deployment Characteristics We define the important characteristics of an application's physical deployment. The number of sensors, types of sensors, number of users, and duration of annotated dataset (if any) differ greatly between

Dataset	Size	Sensor Types	Users	Dataset
PlaceLab [22]	900+	heterogeneous	1	120h
CASL [15]	15	heterogeneous	3	5d
Van Kasteren [23]	14	State change	1	28d

Table 1. Application deployment characteristics

applications and deployments. This can be seen in table 1 where we characterise three diverse datasets. While it may seem obvious how these characteristics differ between applications, they were also selected on the basis that they influence the performance of situation identification techniques. For example, the size of the deployment may lead to scalability issues for some situation identification techniques.

Scenario Characteristics We also need to further dissect the application in terms of the scenarios involved. A smart office application, for example, may rely on the identification of high level situations such as meeting, presentation, or coffee break. The contexts required for identifying the different situations in an application vary. This can also be said for the same situation across different applications where the physical deployment differs. We refer to these high level situations and their application specific characteristics as scenarios. We define the important characteristics of scenarios as frequency, duration, regularity, and required contexts (Time, Location, Identity, Device, Application). We illustrate this classification in table 2, using four scenarios that occur in the PlCouple1PlaceLab dataset [22].

Scenario	Frequency	Avg. Duration	Est. Regularity	Req. Contexts
Eating	197	1.58 mins	4 times per day	kitchen sensors, location
Hygeine	38	3.05 mins	2 times per day	bathroom sensors, location
Watching TV	22	33.29 mins	none	tv sensor, location
Using Computer	132	19.24 mins	3 times per day	computer sensor, location

Table 2. Scenario characteristics example using Plcouple1 PlaceLab dataset

3.2 Relating Situation Identification Performance to Application Characteristics

We have identified the evaluations by Knox et. al., [15, 16] as being one of the most comprehensive in terms of comparing the performance of techniques us-

ing diverse datasets. They evaluated the performance of Case Based Reasoning (CBR), Naive Bayes (NB), Suppor Vector Machines(SVM), and C4.5 Decision Trees (DT) in situation identification using three differing datasets. The PlaceLab home dataset [22], the CASL office dataset [15] and the home dataset by Van Kasteren et. al., [23]. The differences between these datasets can be seen in Table 1. Since these techniques were evaluated using identical setups they are directly comparable. We further analyse these evaluation results as a basis for understanding the relationship between scenarios and situation identification techniques.

The evaluation results presented show that one solution does not outperform the others in all scenarios. By scenario we mean identifying a specific situation given the characteristics of the dataset. By analysing these results in terms of how each technique performed for each scenario we can begin to understand the links between the physical characteristics and requirements of scenarios and how this effects the performance of a technique for that scenario.

Technique	**Scenario**	**F-Measure**
CBR	Eating+PlaceLab	0.39
DT	Eating+PlaceLab	0.41
NB	Eating+PlaceLab	0.0
SVM	Eating+PlaceLab	0.0
CBR	Watching TV+PlaceLab	0.46
DT	Watching TV+PlaceLab	0.47
NB	Watching TV+PlaceLab	0.73
SVM	Watching TV+PlaceLab	0.65
CBR	Go to bed + Van Kasteren	0.02
DT	Go to bed + Van Kasteren	0.52
NB	Go to bed + Van Kasteren	0.58
SVM	Go to bed + Van Kasteren	0.63

Table 3. Performance of techniques for PlaceLab Scenarios. Taken from [16]

SVMs and NB performed very well in a limited number of scenarios that occur frequently, i.e., have more available training data. This is illustrated in table 3 where the performance in terms of f-measure is listed for each technique for four scenarios. The first scenario identifies the situation where someone is eating and the second identifies someone watching TV, both using the PlaceLab dataset. From table 2 it is clear that there is many more instances of the eating scenario compared to that of watching TV, but the watching TV scenario has a much longer average duration meaning that there is more data relevant to that scenario. This suggests that SVM's and NB are better solutions for scenarios that are longer in duration. Both techniques failed to identify any of the brief scenarios in this from the PlaceLab dataset and this further demonstrates the inability of a single technique to solve all problems.

CBR and DT's were identified as the most consistent techniques across scenarios although CBR struggled to distinguish between similar scenarios in some cases. The "go to bed" situation from the Van Kasteren dataset relies the context-type time to be identified. As can be seen in table 3 CBR is the only technique that performed poorly for this scenario.Interestingly however, CBR was by far the most successful in identifying the "Prepare breakfast" situation from the same dataset. This scenario relied on time and device contexts. This demonstrates that CBR is not the optimal solution for solely time based scenarios. For scenarios where multiple contexts are available, and the scenario is distinct from others, CBR performs well.

Discussion Analysis of the evaluations performed by Knox et. al., [16] shows the relationship between the situation identification techniques and application characteristics. Support Vector Machines and Naive Bayesian were identified (out of the techniques in question) as being optimal solutions for scenarios that have long lasting durations where there is more training data. They should be avoided however for brief scenarios as they often fail to identify the situation at all. Case Based Reasoning was shown to perform well for scenarios that involve multiple context types, but poorly for scenarios that were solely time based. Case Based Reasoning also struggles to distinguish between similar scenarios. These evaluations have given us an initial insight into how the characteristics of applications relate to the performance of situation identification techniques. This understanding forms the basis for a profile of situation identification techniques which we will use in order to automatically select the most suitable solutions.

4 Architecture: Unifying Framework for Existing Situation Identification Methodologies

Our architecture aims to provide a decision framework for the selection of situation identification techniques on the behalf of the developer. It is divided into three layers as illustrated in figure 1. The developer layer supports application specific inputs by providing templates to be filled in by the developer. These templates are based on the important characteristics of application deployments/scenarios defined in section 3.1. This layer also displays the output from lower layers, i.e., the recommended situation identification technique(s). The decision layer contains the logic for selection of technique(s) from the situation identification layer, based on the application profile. We describe these layers in more detail using a smart home application example.

4.1 Developer Layer

The deployment template supports the input of information unique to the physical deployment for the application. The required information may be provided as domain knowledge, e.g., information provided from the proposed user of the system, and/or from datasets gathered from the deployment.

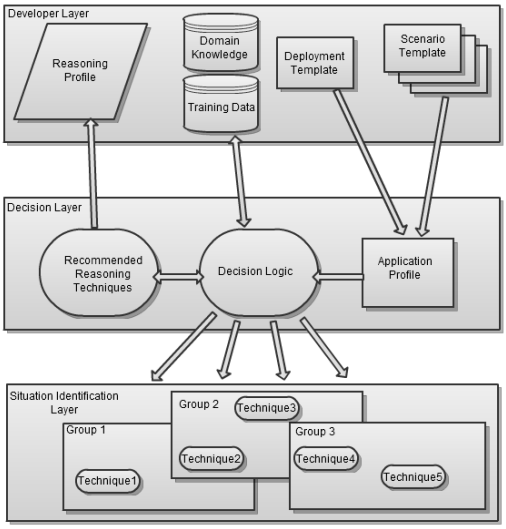

Fig. 1. Architecture of a unifying framework for situation identification methodologies

In our smart home example the deployment consists of 30 state change sensors attached to household objects and 5 location sensors monitoring the users movements throughout the house. The state change sensors are present in the kitchen, living room and bathroom. There is a dataset with 100 hours of annotated data available.

```
<Deployment name = "Case Study Deployment">
    <Characteristics>
        <Size = "40"  />
        <SensorTypes = "statechange, location"  />
        <NumParticipants = "1"  />
        <DatasetDuration = "100h"  />
    </Characteristics>
</Deployment>
```

Scenario templates support the input of scenario specific characteristics. These characteristics have been chosen as they have been shown in previous evaluations to have a connection with the performance of specific situation identification techniques. Using the scenario templates, the developer describes each of the scenarios that need to be identified in the application. This information comes from domain knowledge and/or training data. Training data can also be used to validate any manually entered domain knowledge.

We describe three common scenarios for home applications, eating, hygiene, and gone to bed. Watching TV occurs about once every two days lasting 30 minutes on average, there is no pattern to it's occurrence and the contexts it relies

on are location and device. Gone to bed lasts 8 hours on average, usually occurs once daily between 11pm and 1am and relies on location and time contexts. The eating scenario template is illustrated below:

```
<Scenario name = "Eating" >
    <Characteristics>
        <AverageDuration = " 10m"  />
        <ExpectedFrequency = " 2/1d "  />
        <Regularity = "~9am, ~6pm "  />
    </Characteristics>
    <ContextTypes>
        <Location = "yes"  />
        <Time = "yes"  />
        <Identity = "no"  />
        <Device = "yes"  />
        <Application = "no"  />
    </ContextTypes>
</Scenario>
```

The reasoning profile displays the output of the decision framework to the developer. The situation identification technique(s) selected will be presented to the developer illustrating which scenario(s) each technique is most suited to. We give an example of this output after describing the decision making process.

4.2 Decision Layer

The application profile component combines information provided in the deployment and scenario templates and builds a profile for the application as a whole. Here, scenarios and relevant deployment characteristics are analysed and grouped according to the characteristic similarities and contextual similarities. This step allows us to identify similar scenarios which can prove difficult to identify for some situation identification techniques. This was shown to be an issue for Case Based Reasoning for example. This profile will serve as a more concise description of application characteristics for use by the decision logic.

Based on the application and technique profiles, the decision logic component selects technique(s) based on their ability to effectively perform situation identification for the application as a whole, i.e., the applications physical and scenario specific characteristics. For example, a scenario may rely on context from sensors that are prone to error/failure such as environmental sensors. This needs to be accounted for in the decision making process. The selected technique should be able to identify such a situation while managing the uncertainty of the data. Techniques that can not manage a particular scenario's characteristics with the training data available are ruled out for that scenario, but not for others.

A profile of the relevant situation identification technique(s) is built incrementally by the recommended reasoning techniques component. This again is

assessed by the decision logic component, reducing the number of techniques available based on the similarities of scenarios. In order to prevent the recommendation of an unnecessarily large number of techniques the decision logic component analyses the trade-offs between techniques, recommending as few techniques as possible while maintaining an optimal solution. The complete recommendation forms the basis for a reasoning profile in the developer layer. The final reasoning profile is presented to the developer via the reasoning profile in the decision layer, illustrating the grouping of similar scenarios and the optimal solutions for each group. The following is a sample output for our smart home example.

```
<ReasoningProfile>
    <RecommendedTechnique name = "Case Based Reasoning">
        <Scenario name = "Watching TV" />
    </RecommendedTechnique>
    <RecommendedTechnique name = "Support Vector Machines" >
        <Scenario name = "Eating" />
        <Scenario name = "Gone to bed"/>
    </RecommendedTechnique>
</ReasoningProfile>
```

4.3 Situation Identification Layer

This layer contains the situation identification techniques employed by the framework and a profile of each. Techniques are grouped according to their previously discussed capabilities and may belong to more than one group. It is important here, to allow for additional techniques to be added to the framework as they become available. The technique profiles provide a description of the strengths and weaknesses of each technique.

5 Summary and Future Work

Situation identification techniques are limited in terms of the variety of applications they can be successfully applied to. We identified an ideal solution to be one where techniques can be interchanged according to application needs. We characterised applications in terms of their deployment and scenario characteristics. We analysed evaluations of situation identification techniques in relation to these characteristics and gave an initial description of the relationship between application scenarios and how the applied techniques perform. We proposed a unifying framework that automatically selects the most suitable combination of techniques for a given set of application characteristics. It is argued that a unifying framework will support a single development platform for a diverse range of context-aware applications.

While we used a smart home example to illustrate the workings of our architecture the framework aims to be accessible to a diverse range of applications.

Our initial evaluations analysis identified some relationships between techniques and the scenarios they are applied to. We will perform further evaluations focussing on diverse scenarios in order to identify further the characteristics (and combinations of characteristics) that affect the performance of situation identification techniques. We also intend to include application requirements into the decision making process. Until now we assume that all scenarios simply require the best solution available. Realistically however, some scenarios may be safety critical and as such may require the combination of techniques used in that scenario in order to reach a satisfactory level of certainty. While our current aim is to gain a greater understanding of the technique-scenario relationships, it would also be interesting to apply machine learning to the decision making process in order to learn optimal solutions. The recommended reasoning techniques from this architecture have been described using XML. The presentation of such output to the user, illustrating the benefits of one technique over the others for particular scenarios, will be further investigated.

References

1. M. Weiser, The Computer for the 21st Century, Scientific Am., pp. 94- 104. (1991)
2. Dey, Anind K. and Gregory D. Abowd (2000b). Towards a better understanding of context and context awareness. In the Workshop on the What, Who, Where, When and How of Context-Awareness, affiliated with the 2000 ACM Conference on Human Factors in Computer Systems (CHI 2000), The Hague, Netherlands. April 1-6, 2000
3. Simon Dobson and Paddy Nixon. Whole-system programming of adaptive ambient intelligence. In Proceedings of HCI International. LNCS. Springer-Verlag. Beijing, CN. 2007.
4. Anand Ranganathan, et al. A Middleware for Context- Aware Agents in Ubiquitous Computing Environments, USENIX International Middleware Conference, 2002.
5. Truong BA, Lee Y-K, Lee S-Y (2005) Modeling and Reasoning about Uncertainty in Context- Aware Systems. IEEE International Conference on e-Business Engineering 2005: 102-109.
6. H. Hagras, V. Callaghan, M. Colley, G. Clarke, A. Pounds-Cornish, and H. Duman. Creating an ambient-intelligence environment using embedded agents. IEEE Intelligent Systems, 19(6):12-20, 2004.
7. Panu Korpipaa, Miika Koskinen, Johannes Peltola, Satu-Marja Mkel, and TapioSeppanen. Bayesian Approach to Sensor-Based Context Awareness, Per- sonal and Ubiquitous Computing J., vol. 7, no. 4, 2003.
8. L. R. Rabiner and B. H. Juang, An introduction to hidden Markov models, IEEE Acoust., Speech, Signal Processing Mag., pp. 4-16, Jan.1986.
9. Englebienne VanKasteren, Noulas and Krose. Accurate activity recognition in a home setting. In In Proceedings of Tenth International Conference on Ubiquitous Computing, South Korea, September 2008.
10. Kofod-Petersen, A., and Aamodt, A. 2009. Case-based reasoning for situation-aware ambient intelligence: A hospital ward evaluation study. In ICCBR, 450464.
11. Cassens, J. and Kofod-Petersen, A. (2007b). Explanations and case-based reasoning in ambient intelligent systems. In Coyle, L. and Schwarz, S., editors, Case-Based Reasoning and Context-Awareness, The Seventh International Conference on

Case-Based Reasoning (ICCBR 07), Workshop Proceedings, pages 167176, Belfast, Northern Ireland. University of Ulster
12. P. Castro, P. Chiu, T. Kremenek, R. Muntz. A Probabilistic Room Location Service. Proceedings of Ubicomp 2001: Ubiquitous Computing. Atlanta, Georgia, September 2001.
13. Randell, C., Muller, H., Context Awareness by Analyzing Accelerometer Data, Fourth International Symposium on Wearable Computers (ISWC'00), p. 175-176, Atlanta, Georgia, October 18 - 21, 2000.
14. Huimin Qian, Yaobin Mao, Wenbo Xiang, and Zhiquan Wang. Recognition of human activities using svm multi-class classifier. Pattern Recogn. Lett., 31(2):100111, 2010.
15. Stephen Knox, Lorcan Coyle and Simon Dobson. Using ontologies in casebased activity recognition. In Proceedings of the 23rd International Conference of the Florida Artificial Intelligence Research Society (FLAIRS- 23). Daytona Beach, FL. May 2010.
16. S. Knox, Combining Case-Based Reasoning and the Semantic Web in Recognising Situations. PhD thesis, University College Dublin, School of Computer Science and Informatics, 2010
17. Juan Ye, Lorcan Coyle, Susan McKeever and Simon Dobson. Dealing with activities with diffuse boundaries. In Proceedings of the Workshop on How to do good activity recognition research: Experimental methodologies, evaluation metrics and reproducility issues at PERVASIVE 2010. Helsinki, FI. May 2010.
18. Michael Przybilski, Petteri Nurmi, Patrik Floren: A Framework for Context Reasoning Systems Proceedings of the 23rd IASTED International Conference on SOFTWARE ENGINEERING (SE 2005), pages 448 - 452
19. P. Nurmi, P. Floren, M. Przybilski and G. Linden, A Framework for Distributed Activity Recognition in Ubiquitous Systems In proceedings of the International Conference on Artificial Intelligence (ICAI '05), pp. 650 - 655, Las Vegas, 27 - 30 June, 2005.
20. Nishkam Ravi, Nikhil Dandekar, Preetham Mysore, and Michael L. Littman. Activity recognition from accelerometer data. In AAAI, pages 1541-1546, 2005.
21. L. Bao and S. Intille. Activity Recognition from User-Annotated Acceleration Data. Proc. Pervasive, 1-17, Vienna, Austria, 2004.
22. B. Logan, J. Healey, M. Philipose, E. M. Tapia, and S. S. Intille. A long-term evaluation of sensing modalities for activity recognition. In Ubicomp, pages 483500, 2007.
23. T. van Kasteren, A. Noulas, G. Englebienne, and B. Krose. Accurate activity recognition in a home setting. In UbiComp 08: Proceedings of the 10th international conference on Ubiquitous computing, pages 19, New York, NY, USA, 2008. ACM.
24. Hung Q. Ngo et al: Developing Context-Aware Ubiquitous Computing Systems with a Unified Middleware Framework. In Proceedings of the EUC 2004: 672-681.
25. B. A. Truong, Y.-K. Lee, S.-Y. Lee: Modeling Uncertainty in Context-aware Computing. 4th Annual ACIS International Conference on Computer and Information. Science (ICIS05)
26. Waltenegus Dargie. The role of probabilistic schemes in multisensor contextawareness. IEEE International Conference on Pervasive Computing and Communications Work-shops, pages 27 32, 2007.