

# Case-Based Situation Assessment in a Mobile Context-Aware System

Anders Kofod-Petersen and Agnar Aamodt

Department of Computer and Information Science  
Norwegian University of Science and Technology  
NO-7491 Trondheim  
Norway

**Abstract.** This paper describes how to utilise Case Based Reasoning for identifying and user situations assessment in a context-aware mobile system. The Case Based Reasoning mechanism attempts to identify what situation the user is in, and utilises a Multi Agent System, consisting of information suppliers, to provide the user with personalised and context-sensitive information.

**Keywords :** Case-Based Reasoning, Context-Awareness, Multi-Agent System, Mobile System.

## 1 Introduction

The average computer user is becoming more and more mobile. The users are bringing an increasing amount of data and computational power along, and are more and more likely to have net access everywhere.

With this movement of the computer from the desktop to the ubiquitous paradigm described by Weiser [1], the computer system now has to adapt to the user's situation, instead of the user adapting to the computer.

Even though there have been a lot of research within the field of context-awareness, the term context is still not well defined. Much of the research on context-awareness has focused on the location part of context, partly due to its importance and partly due to the fact that it is the best understood parameter in a context.

This paper describes an approach to automatic situation assessment in a mobile environment. The context-awareness mobile system consists of three major parts: an open-ended context ontology, Case Based Reasoning (CBR) based interpretation of the contextual data available, and a Multi Agent System (MAS) of information supplying agents.

This work proposes an open context model where a taxonomic structure of context types is given in the design phase. This enables the users to dynamically add, or remove, the types of contextual data available to the system. The context model is linked to a domain model that enables high-level semantic interpretation of situations.

This work proposes the use of Case Based Reasoning as the main reasoning part of the system. This CBR engine will, based on the information available and the interpretation enabled by the domain model, determine the users situation and the possible goals associated with these situations.

With the openness of the context model, and the adaptivity of the reasoning mechanism, it must be equally easy to introduce new services, which the system can utilise to supply a context-aware service to the user. To allow for this, the whole system is based on a Multi Agent System, where each service is accessed through an agent.

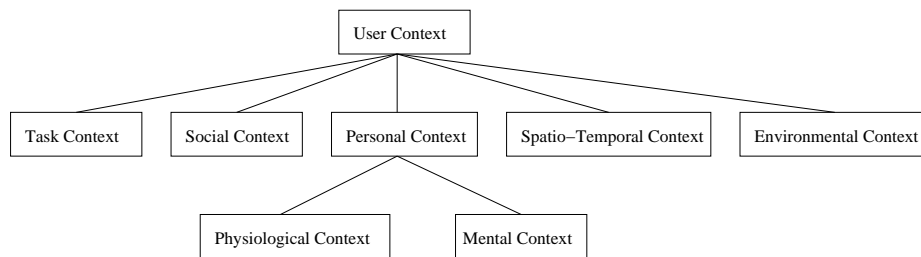
## 2 Context

Through out the brief period of context-aware computing as a research field, several more or less elaborate context models have been proposed (for a overview see for an example Dey and Abowd [2]). Even though a lot of research has been put into the different aspects of context, the predominant part of context in most applications is location.

The word context is still not a well defined term in the context-aware field of research. To ensure that no misunderstandings will occur, a definition of context versus situation used in this article is in order.

As in the work by Day and Abowd [2], the term context is used in the broadest possible sense; it encompasses any information that might be useful for defining the user's situation. There are potentially many more types of contextual information available than what is used to define a given situation. Hence, a *situation* is described by a *context*, which is an instance of the *contextual information* available. In our approach a situation is stored in a case.

To facilitate this degree of flexibility in the definition of context an open context model is in place. This model only defines the taxonomic structure is in the design phase (see Figure 1).



**Fig. 1.** Context Hierarchy

The context model for the system is part of the ongoing EU founded AmbieSense research project (a more thorough discussion can be found in [3]). This context model is divided into five main categories: *i*) Task context: The task context describe what the user is doing, it can describe the user's goals, tasks, activities, etc. *ii*) Social context: This describes the social aspects of the user, such as information about friends and relatives, the role the user plays, etc. *iii*) Personal context: This part describes the mental and physical information about the user, such as mood, expertise, disabilities, and weight. *iv*) Spatio-temporal context: This type of context is concerned with attributes like: time, location, and movement. *v*) Environmental context: This part captures the users surrounding, such as things, services, light, people, and information accessed by the user.

This work postulates that there exist a goal or problem in any situation. It would be futile to identify a situation unless there is some task connected to it - no matter how mundane. This is most obvious when dealing with users, where a situation implies that there is a problem that needs to be solved; such as the possible situation "hungry user", which implies the goal of *not hungry user*, hence food should be located.

### 3 System overview

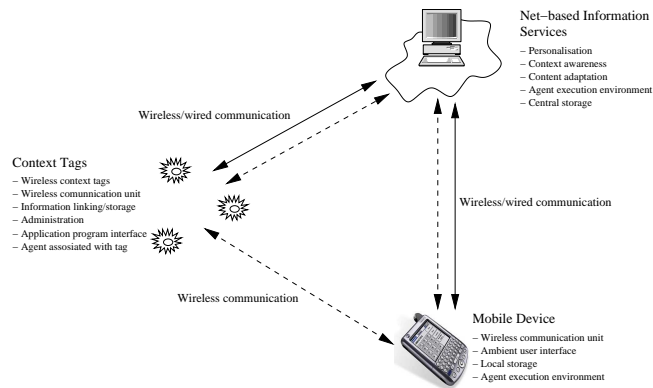


Fig. 2. Overall Architecture

The mobile system is one compound in an architecture that also contains “Context Tags” (Bluetooth beacons) primarily used for communications and location, and Net-Based Information Services (see Figure: 2). Since this work is concerned with contextual understanding on mobile devices, the overall architecture will only be touched briefly. Interested parties are directed to the AmbieSense website<sup>1</sup> or a more thorough description of the architecture in Myrhaug and Göker [4].

One of the Context Tag’s primary assignments is to supply the location. It can also distribute localised information, such as the menu of the particular restaurant, where it is installed. The more advanced Context Tags can offer both local services or a connection to services located on the net.

The Net-Based Information Services are divided into two major categories: *i*) The AmbieSense connected services that offer personalised information services tailored to the AmbieSense system. *ii*) The more general information services that are available on the Internet.

The mobile system is divided into three major parts: The domain model, the multi agent system, and the reasoning mechanism.

#### 3.1 Domain model

One of the major requirements for communications and reasoning across entities is a common understanding of the concepts and relations that are to be used.

In this system a top-level taxonomy for context has been defined (see Figure 1). This ontology is concerned with the types of information that are related to the definition and understanding of context. This context model is not concerned with a complete model from the beginning, but rather imposes a structure that all suppliers of contextual information must abide to.

The context model is integrated into a more general domain ontology, which describes concepts of the domain (e.g. Airport Hall, Gate, Restaurant, Newsstand) as well

<sup>1</sup> [www.AmbieSense.com](http://www.AmbieSense.com)

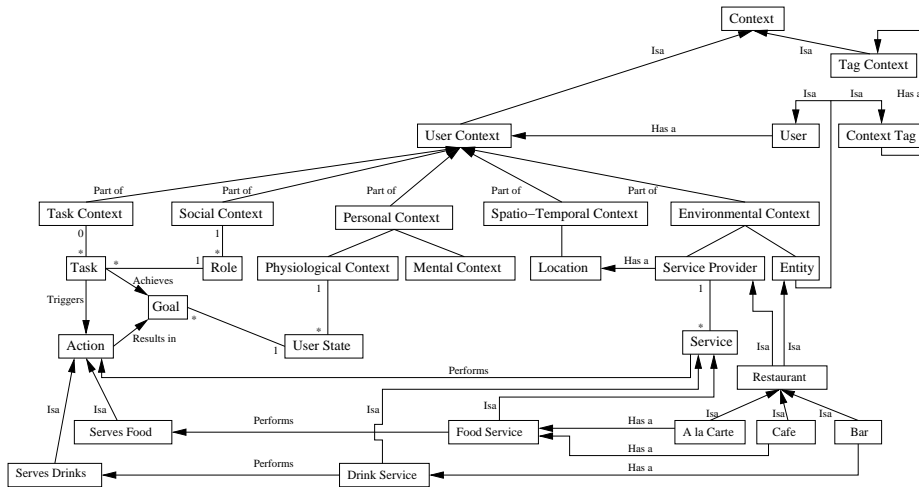


Fig. 3. Airport Domain

as more generic concepts (Task, Goal, Action, Physical Object) in a multi-relational semantic network. The model enables the system to infer relationships between concepts by constructing context-dependent paths between them. One important use of this is to be able to match two case features that are syntactically different, by explaining why they are similar ([5], [6]). For example, the concept Magazine matches Journal in the context of Newsstand.

A part of the domain model –in which the context model is integrated– is illustrated in Figure 3.

### 3.2 Agents

As a part on the AmbieSense system, the mobile device holds the agency that identifies user situations and solves the problems associated with the current situation.

Agents have been chosen for various reasons. One of the most important is the modularity that an agency supports. This flexibility is a great benefit when new application agents are implemented into the system.

The agency consists of three kinds of agents:

- The framework agents supplied by the agent platform; Jade [7]
- The core agents handling the situation detection and goal decomposition.
- The application agents, each handling one application specific chore, such as map construction or news gathering.

The most important core agents are the context agent, and the facilitator. The context agent maintains the space of contextual data, and utilises CBR to identify the current situation. Based on this identification, the agent notifies the facilitator about this particular situation. This notification is the set consisting of the situation with the corresponding contextual information, and the goal associated with. Once the facilitator has reached it's goal, the solution will be returned to the context agent.

The facilitator uses the Unified Problem-solving Method description Language (UP-ML) [8] to maintain an overview of the application agents' services and for handling the

task decomposition. It will receive the situation and task description from the context agent, decompose the task into the different sub-tasks required, and recruit the correct application agents for solving the tasks.

Application agents are responsible for solving their own particular tasks. At present four different application agents exist:

- Map agent, who can access the map server and supply map suited to the particular context information.
- News agent, who can gather relevant news.
- Information agent, who can gather information that are generally available on the net.
- Airport agent, who can solve airport related problems.

When new application agents are to be introduced into the system, the programmer needs to know the context ontology and must specify the capabilities of this agent in UPML, and the agent is basically ready to be a part of the system.

### 3.3 Reasoning

In most of the research in context-aware systems, the problem of filtering the vast amount of contextual information that are available, in such a way that the identification of important constellations of the contextual information are feasible, has not been thoroughly addressed. Case Based Reasoning is a promising method for this.

Case Based Reasoning [9] is concerned with adapting to new situations by remembering similar earlier experienced situations (cases). CBR has historically been used in large monolithic systems. This work applies CBR as a lightweight reasoning mechanism that is capable of running on a small mobile device.

The reasoning mechanism is split into two different parts. The on-line part that resides on the user's mobile device, and the off-line reasoning that resides on the user's backbone system.

**On-line reasoning** The CBR mechanism is encapsulated in the context agent. This agent maintains a dynamic structure of the contextual information available. Different types of contextual information can arrive in a very diffuse fashion, e.g. time is continually flowing into the system, whereas location might be pseudo static. Since CBR works on discreet cases the continuously values flowing into the system must be made discreet. This is handled by the context agent, following the suggestion of Zimmermann [10]. The agent takes "snapshots" of the contextual information at certain time intervals, i.e. the state of the context structure, and stores them as cases.

Once a snapshot has been taken, the CBR cycle is activated (see Figure 4). The system will try to retrieve a known context or case, and classify the current situation based on the retrieved one. When the situation has been classified, the associated goal is presented to the task decomposition agent. After the agency has handled the problem the case will be stored in the case base in a triplet consisting of: i) the contextual information describing the situation, ii) the problem associated with the situation, and iii) the solution constructed by the application agents.

Since the user is expected to experience at least a few different situations a day, the storage of the cases will quickly fill up the mobile device with cases. This potential vast amount of cases will also severely hamper the searching process for the CBR mechanism. To remedy this, some of the reasoning process is moved into the user's backbone servers.

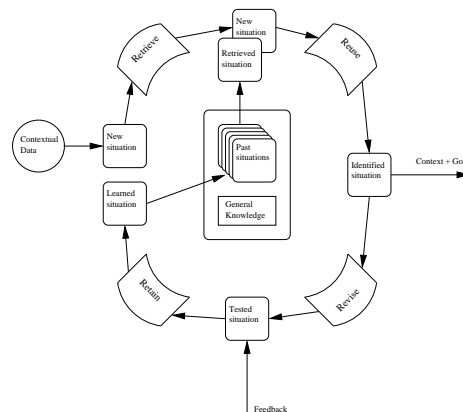


Fig. 4. CBR Cycle

**Off-line reasoning** There are potential two main problems with the use of Case-Based Reasoning for identifying situations; the storage problem, and the problem of indexing and searching.

First and foremost is the problem of storing the, potentially vast, amount of cases constructed during run time. To solve this the user will have personal persistence storage available on the user's home network. This storage will be used for storing the cases, and will be synchronised when the user has an up-link. This large amount of data does not only affect the amount of storage space needed, it will also severely affect the indexing and matching algorithm used.

To remedy this a generalisation process will occur on the home net. Similar cases will be grouped into prototypical cases, e.g. everything that the 600 business meetings has in common will constitute the prototypical business meeting situation. These prototypical cases will be part of the on-line case base, and be used in the every day reasoning process. Generalisation of cases is a well known research area within CBR, for an overview see for an example [11]. This case base is structured by prototypes, which are generated on the basis of the amount of similar parameters in the point-cases.

#### 4 A Hungry User Example

To illustrate how the system works an example is in order. This example covers the hungry user case mentioned before. For the system to work, different kinds of contextual information flow into the system. In this example the system must detect that the user is hungry and assist him in finding food to his liking.

Lets assume that the user is at an airport. The user's personal preferences, located in personal context (see Figure: 3), states that the user has a hang to Italian food. Part of the spatio-temporal context shows that the time is a quarter past one and that the user is at Oslo airport; The environmental context shows that there is a Context Tag nearby; Part of the personal context shows that it has been more than five hours since he last used his credit card at a restaurant.

The CBR system can now take this situation case (see Figure: 5) and try to find a matching case in it's case base The Identify User Need task matches Get Food, which is a subclass of the former in the domain model. A hungry user prototypical

case is matched, based on the information presented above. The matching case (Figure 6) is a prototypical case, in which the difference between current time and the time of the last restaurant visit, as well as the current time itself, have been generalised

The CBR engine has now identified the task of `get-food`, and can leave it to the facilitator to accomplish it.

```
Case 0
TASK: Identify User Need
TASK STATE: In Process
SOCIAL CONTEXT: Departing
PERSONAL CONTEXT:
  PREFERENCE:
    Italian Food
    Guardian Newspaper
    Spanish Wine
  FINANCIAL:
    Last-CC:
      08:04:00
      [Joe's Breakfast Club]
SPACIO-TEMPORAL:
  Current Location : OSL
  Current Date: 21032003
  Current Time: 13:15:00
ENVIRONMENTAL:
  Entity: OSL-CT-TAG-12
```

**Fig. 5.** Situation case

```
Case 1
TASK: Get Food
GOAL: Not Hungry
TASK STATE: Accomplished
SOCIAL CONTEXT: Departing, Hungry
PERSONAL CONTEXT:
  PREFERENCE:
    Italian Food
    Le Figaro Newspaper
  FINANCIAL:
    Last-CC:
      diff(Current Time. Last-CC) >= 6 h
SPACIO-TEMPORAL:
  Current Location : OSL
  Current Time: range[11:00 - 14:00]
ENVIRONMENTAL:
```

**Fig. 6.** Prototypical case

The facilitator knows what agent are available on the mobile system, and what services they offer. The facilitator can furthermore look up what services that are available, via the Context Tag, under the Service Providers in the Environmental Context. This information is then used to decompose the task of `get-food` into `find-restaurants`, `match-restaurants-to-preferences`, `check-related-news`, and `construct-map`. In this case the airport's Context Tag supplies the `find-restaurants`, and the information necessary for the map agent to execute `construct-map`. The agents on the mobile device will execute the `match-restaurants-to-preferences`, and through the up-link supplied by the Context Tag, the news agent will execute `check-related-news`.

Once the solution has been returned to the Context agent, it can remind the user that it might be time for lunch. The suggestion is that there is a nice Italian restaurant five minutes away (as shown on the map), the local restaurant guide claims that the service is acceptable, the kitchen is above average, and that a decent "eat all you like" lunch offer is available for only 11 €.

## 5 Conclusion and Further Work

This work is part of ongoing research, and experimental results are still to come. However, work by Zimmermann [10] suggests that CBR is a promising method for identifying the correct combinations of contextual information that leads to a good situation understanding.

We are currently developing test scenarios together with Oslo Airport Gardermoen. In addition, some experiments are also planned in relation to a local project in our university within the medical field. Here, the hospital staff members are to be equipped with PDAs and a context aware system that will support the work on the ward. In this project the occurring situations are well known and documented, hence it will be a controlled way to tune the CBR mechanism for identifying similar situations.

Finally, the main parts of this project will be integrated into the test scenarios used in the AmbieSense project.

## 6 Acknowledgement

This work is part of the AmbieSense project, which is supported by the EU commission (IST-2001- 34244).

## References

1. Mark Weiser. Some computer science issues in ubiquitous computing. *Communications of the ACM*, 36(7), 1993.
2. A. K. Day and G. D. Abowd. Towards a better understanding of context and context-awareness. In *CHI 2000 Workshop on The What, Who, Where, When, Why, and How of Context-Awareness*, April 2000.
3. Ayse Göker and Hans Inge Myrhaug. User context and personalisation. In *Workshop proceedings for the 6<sup>th</sup> European Conference on Case Based Reasoning*, 2002.
4. Hans Inge Myrhaug and Ayse Göker. Ambiesense – interactive information channels in the surroundings of the mobile user. In Constantine Stephanidis, editor, *Universal Access in HCI, 10<sup>th</sup> International Conference on Human-Computer Interaction*, volume 4, pages 1158–1162. Lawrence Erlbaum Associates, 2003.
5. Agnar Aamodt. Explanation-driven case-based reasoning. In S. Wess, K. Althoff, and M. Richter, editors, *Topics in Case-based reasoning*, pages 274–288. Springer Verlag, 1994.
6. Martha Dørum Jære, Agnar Aamodt, and Pål Skalle. Representing temporal knowledge for case-based prediction. In *Advances in case-based reasoning*, 6th European Conference, ECCBR 2002, Lecture Notes in Artificial Intelligence, LNAI 2416, pages 174–188. Springer Verlag, September 2002.
7. Fabio Bellifemine, Agostino Poggi, and Giovanni Rimassa. Jade - a fi pa-compliant agent framework. Technical report, Centro Studi e Laboratori Telecomunicazioni, 1999. Part of this report has been also published in Proceedings of PAAM'99, London, April 1999, pagg.97-108.
8. D. Fensel, E. Motta, F. van Harmelen, V. R. Benjamins, M. Crubezy, S. Decker, M. Gaspari, R. Groenboom, W. Grosso, M. A. Musen, E. Plaza, G. Schreiber and R. Studer, and R. Wielinga. The unified problem-solving method development language upml. *Knowledge and Information Systems Journal (KAIS)*, 5(1), 2003.
9. Agnar Aamodt and Enric Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Communications*, 7(1):39–59, 1994.
10. Andreas Zimmermann. Context-awareness in user modelling: Requirements analysis for a case-based reasoning application. In Kevin D. Ashley and Derek G. Bridge, editors, *IC-CBR 2003, Case-Based Reasoning Research and Development*, LNAI 2689, pages 718–732. Springer-Verlag, 2003.
11. Kerstin Maximini, Rainer Maximini, and Ralph Bergmann. An investigation of generalized cases. In Kevin D. Ashley and Derek G. Bridge, editors, *ICCBR 2003, Case-Based Reasoning Research and Development*, LNAI 2689, pages 261–276. Springer-Verlag, 2003.