

# Towards a Formal Model of Context Awareness

Mikkel Baun Kjærgaard and Jonathan Bunde-Pedersen

Department of Computer Science, University of Aarhus,  
IT-parken, Aabogade 34, DK-8200 Aarhus N, Denmark,  
{mikkelbk, jbp}@daimi.au.dk

**Abstract.** There is a definite lack of formal support for modeling realistic context-awareness in pervasive computing applications. The CONAWA calculus presented in this paper provides mechanisms for modeling complex and interwoven sets of context-information by extending ambient calculus with new constructs and capabilities. The calculus is a step in the direction of making formal methods applicable in the area of pervasive computing.

## 1 Introduction

In the area of pervasive computing the great vision is moving computation from the desktop computer to a number of devices embedded in the environment of the user. Applications put forward for these devices depend heavily on the notion of context-awareness. The increasing interest in pervasive computing has heightened the need for a formal foundation for these applications and in particular for context-awareness. The existence of such a formal foundation could be utilized in a combined theory and systems building approach to reason about interesting properties for context-information such as privacy and consistency as well as more traditional properties.

For a formal model to be applicable in this setting it should closely reflect the reality of actual context-aware software systems. Generally the context information which can be modeled by approaches such as [1–3] are too limited to capture the diversity and interwovenness of real context information such as described in [4, 5]. An example that illustrate these qualities will be given later based on the AwarePhone application [6].

The message of this position paper is that the expressiveness of formal models of context awareness should be sufficient to model complex sets of context-information. This enables the models to reflect reality and be applicable in the area of pervasive computing.

The remainder of the position paper is organized around motivating key areas for furthering a combined theory and systems building approach explicitly for context-aware systems. This will be done through presenting the experience of making CONAWA: a formal model for context-awareness. The rest of the paper is organized as follows. The idea behind CONAWA is presented in Section 2. Some details of the syntax and semantic of CONAWA and an example of its use are given in Section 3. A discussion of the experience with CONAWA and

key areas for furthering the idea of a combined theory and systems building approach for context-aware systems is given in Section 4.

## 2 The ideas behind CONAWA

In the work with CONAWA the findings of [4] was used as a definition of context-awareness and a characterization of context-aware applications. In that paper four basic categories of context-aware applications are defined as the product of two points along two orthogonal dimensions. One of the dimensions describes if a task at hand is getting information or is carrying out a command; the other dimension describes if it is effected manually or automatically (see Table 1).

	Manual	Automatic
Information	proximate selection & contextual information	automatic contextual re-configuration
Command	contextual commands	context-triggered actions

Table 1. Four categories of context-aware applications

We have also been inspired by the ideas on mobility and context-awareness described in [6]. These ideas revolve around the concept of an “Aware-Phone” which is an application implemented on a mobile phone supporting context-mediated social awareness.

### 2.1 Scenarios

In order to evaluate the expresiveness of a formal model in terms of context information, we propose to use the following four scenarios. The scenarios were developed so each reflect one aspect of the four categories in figure 1 and contain interwoven context information.

**Contextual information** A nurse needs to find the *nearest* and *available* doctor.

**Contextual commands** A doctor migrates an X-ray image from the Aware-Phone to the *nearest wall-sized* display.

**Context-triggered actions** A doctor is automatically logged in and out when *approaching* a computer.

**Contextual reconfiguration** A doctor logs into a computer which is automatically reconfigured to his *current activity* and other *context*.

### 2.2 The Calculus

The CONAWA calculus for modeling context is inspired by the ambient calculus [7]. The ambient calculus deals with ambients which are tree-structured recursively defined entities. An ambient may be situated in another ambient and

itself contain ambients, and information between ambients may only flow between nearby ambients, i.e. parents or siblings. If context-information should be made available to an ambient it must be positioned near the ambient in the tree – but this is not necessarily a good idea. The problem is the very nature of context. For instance; one piece of context might relate to nearby doctors while another describes available displays. These two pieces of context are hard to model independently of each-other in a single tree. Modeling context in a flat structure is possible (see [1,2]) but this approach suffers from its simplicity such that it is difficult to express and navigate the contextual information.

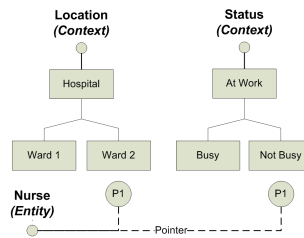


Fig. 1. Illustration of a nurse (entity) and the contexts which she is part of

Our basic idea is to model context using several ambient-like trees. An ambient which represents an entity will have a presence in one or more trees. For instance an ambient modeling a nurse in a hospital might both be present in a location tree (in ‘Ward 2’) and the status tree (she might be ‘not busy’), as shown in Figure 2. We realize that an entity cannot physically be part of more than one tree, but propose to use a solution inspired from bigraphs [8] where links (a la pointers) are used to indicate presence. An ambient utilizing the context information would then navigate the relevant trees in order to specify preferences and indicate which context it was interested in. Take the example where a another ambient moves near the nurse-ambient in both the location and status tree. This would give the ambient a physical proximity to the nurse(in the first tree) but also allow it to see the status of the nurse-ambient(in the second tree). The ambient positions itself in the context-trees such that it is able to locate nurses which are physically close and ‘not busy’. We propose to extend existing ambient-constructs such as *in* and *out* to enable navigating in several contexts at once and also extending the basic tree-structure to use several tree-structures for context-information.

### 3 A formal presentation of the calculus

In this section a more formal presentation of CONAWA is given. Before the general syntax and semantics are presented we provide an example showing how the calculus may be used.

### 3.1 Context-information scenario

The following example shows how CONAWA can be applied to model one of the scenarios presented earlier. The basic idea in the scenario is that when a query for information is issued, then the resulting answer is adapted based on the current context. This is summarized in the following scenario in terms of the Aware-Phone application:

*A young doctor or a nurse needs to contact a more experienced doctor to consult him on some issue. So the Aware-Phone is queried for where the nearest doctor which is not occupied by some other work task is located. The application then returns the best suited doctor in the current context of the location and activities of the doctors on duty.*

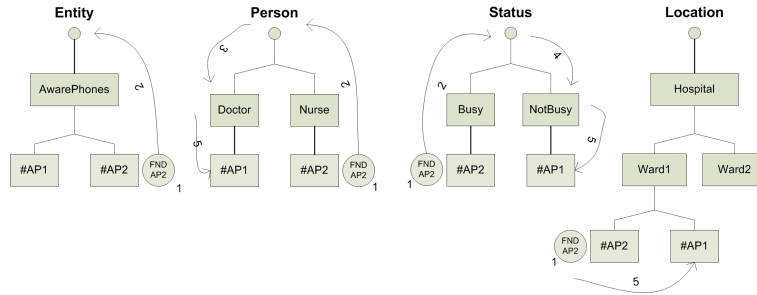


Fig. 2. Illustration of the example scenario

The example scenario is described using the textual syntax in Figure 3 . An illustration of how the scenario could execute is given in Figure 2. The interpretation of the formalized scenario is as follows:

1. The subambient of AP2 performs an  $out\{*\}$  which places it in all contexts of AP2 (as a sibling)
2.  $out\{Person \wedge Status \wedge Entity\}$  moves FNDAP2 upwards in the contexts
3.  $in\{Person\}$  Doctor moves it into the 'Doctor' branch of the context 'Person'
4. Similarly,  $in\{Status\}$  NotBusy makes FNDAP2 a child of 'NotBusy'
5. Then the "agent" enters the ambient which matches  $Person \wedge Status \wedge Location \wedge \neg Entity$  which happens to be AP1

The scenario goes on a bit further and the result is that FNDAP2 returns to AP2 with the ambient name 'Hans' as the nearest *and* available doctor. We have not depicted this in Figure 2, because the main motivation for the scenario is finding the doctor.

```

1 Entity:[ Awarephones[#AP1 | #AP2] ]
2 Person:[ Doctor[#AP1] | Nurse[#AP2] ]
3 Status:[ Busy[#AP2] | NotBusy[#AP1] ]
4 Location:[ Hospital[ Ward1[#AP1 | #AP2] | Ward2[] ] ]
5 ;
6 AP1:[!<Hans>]
7 AP2:
8 [
9   FNDAP2 //Find Nearest Doctor
10  [
11    out{*}.out{Person&Status&Entity}.in{Person} Doctor.in{Status} NotBusy
12    .in{Person&Status&Location&~Entity} ?
13    .GetMsg[out{*}.(name).enter FNDAP2.<name>]
14    .coenter{*} GetMsg.open GetMsg.in {Entity} Awarephones.enter {Entity} AP2]
15    | coenter{*} FNDAP2.open FNDAP2
16    | !<Grethe>
17  ]
18 ]

```

Fig. 3. Textual description of the example scenario.

### 3.2 Syntax

The syntax of CONAWA focuses on constructs which makes it possible to describe and navigate context information. Therefore we have limited the syntax in areas such as scopes of names and general output paths.

A description in the CONAWA calculus consists of two parts. The first part describes a number of contexts which are placeholders for context information. They are ambient structures and represent context information as hierarchical and independent sets of information. The contexts are restricted in their capabilities as well as structure compared to original ambients [7]. These restricted ambients called *context ambients* are static in the sense that they do not have capabilities to move, but only to be created or opened. The second part consist of a number of *reference ambients* which can be used to denote entities which are embedded in certain contexts. These ambients can be present in contexts by a reference which is only a “pointer”. This means that a reference ambient can be represented in multiple contexts<sup>1</sup>. The reference ambients have capabilities which allow them to move in and out of the different contexts.

We have kept the context and reference ambients consistent with those of Cardelli’s on the syntactic level. The only exception is reference ambients which can be positioned in several contexts. To make this possible a reference is used to position a reference ambient  $n$  in a context. The ambient is denoted by  $\#n$  inside the context structure as shown in Figure 3 for AP1 and AP2.

The syntax for the reference ambient’s capabilities *in*, *out*, and *open* have been extended with braces  $\langle$  and  $\rangle$  to indicate in which context or contexts the

<sup>1</sup> But only one representation per context for consistency reasons

capability should be exercised. The context will not be aware of any ambients entering or leaving it this way. The context ambient can use the cocapabilities *coenter* and *coexit* to be able to detect the coming and going of referenced ambients, if they do so by using *enter* and *exit*. The expression between the braces can be a simple boolean expression. An example is given in figure 4. The example will match all ambients which are present in the same context in the *Location* and *Activity* contexts but is a different type of device. By not explicitly naming the ambient on which to perform the capabilities and using the wildcard-character '?', we can model a simple matching mechanism.

... *in*(*Location*  $\wedge$  *Activity*  $\wedge$   $\neg$ *DeviceType*) ?

Fig. 4. Simple matching mechanism

The calculus focuses on extending the standard ambient syntax with constructs and capabilities which lets ambients navigate in complex context information. For a complete formal definition of the syntax see [9].

### 3.3 Semantics

The semantics of the CONAWA calculus extends the ambient semantics for the new capabilities as explained above. For context ambients the following constructs have the same semantic as ambients: parallel, inactivity, exercise capability and open. The semantic of *coenter* and *coexit* are extended to take the context trees into account.

The semantics of reference ambients extends the ambient semantics of several capabilities: *in*, *out*, *enter*, *exit*, *coenter* and *coexit*. The semantic of the constructs have small variations depending on whether a reference ambient is placed in another reference ambient or in a number of contexts by references. In connection with the following constructs, reference ambients have the same semantics as ambients; parallel, replication, inactivity, input, output, exercise capability and *open*.

An example of how the semantic can be defined for a simplified *in* rule is shown in Figure 5. A more complete treatment of the semantic of the CONAWA calculus is given in [9]. The rule allows a reference ambient to perform an *in* in one context. The ellipsis character "... " is used to indicate that the ambient and references can be arbitrary deeply nested within a context. The full *in* rule also handles several contexts and the ? which makes it possible to make a move into an arbitrary ambient.

### 3.4 Further scenarios

In the start of this section the formalization of one of the scenarios were given. The three other scenarios has also been modelled in CONAWA together forming a model of some of the central features of the AwarePhone application. The

$$\begin{array}{l}
C : [\dots\#ra|ca[P]\dots] \\
; \\
ra : [in\{C\}ca.Q|R] \\
\longrightarrow \\
C : [\dots ca[\#ra|P]\dots] \\
; \\
ra : [Q|R]
\end{array}$$

Fig. 5. The rule for *in* in one context

three other models will not be presented here in full detail but can be found in [9]. Below for each of the scenarios it is outlined how they have been formalized.

**Contextual commands** The four contexts: Entity, Screens, Status and Location are modelled and reference ambients representing AwarePhones and displays are used. The images are modelled as ambients that act as an ‘agent’ wrapping the images.

**Context-triggered actions** The two contexts: AwarePhones and Location are modelled and reference ambients representing AwarePhones and displays are used.

**Contextual reconfiguration** The three contexts: AwarePhones, Screens and Location are modelled and reference ambients representing AwarePhones and displays are used.

## 4 Discussion

As stated in the introduction it is our position that for formal models of context awareness to be applicable in the area of pervasive computing their expressiveness should be sufficient to model complex and interwoven sets of context-information. We have developed the CONAWA calculus to make such modeling possible. Using this calculus we have been able to model a number of scenarios of central features of a real context-aware application. The calculus however restricts context information to a number of trees which limits the expressiveness but to a lesser degree than prior research. An issue that the calculus does not take into account is the modeling of probabilistic context information. It is important for calculi to be able to handle such information because it is the typical output of most sensors of context information. This is not only an issue for the present calculus but an issue that should be solved to make a combined theory and systems building approach possible for context-aware systems.

We believe that any model should also form the base for simulation, verification and software prototyping. The present CONAWA calculus does not explicitly contribute to this area. We feel that it is the next logical direction we could take our calculus in by (i) building a simulator which can interpret CONAWA descriptions, (ii) defining a logic which make it possible to state properties which

could be verified and (iii) building a code generation facility that makes it possible to build skeleton code from the models. One important issue in respect to defining a logic is how to reason about formal properties of context information. Here a logic should be developed that allows one to express properties such as: “This awarephone must never reveal my location to other people than the ones I have explicitly granted this privilege”?

To summarize, this paper identified the key area of the expressiveness of formal models of context awareness. Here the models expressiveness should be sufficient to model complex and interwoven sets of context-information. The CONAWA calculus was given as an example of a calculus that is a step in the direction of making such modeling possible.

## References

1. Braione, P., Picco, G.P.: On calculi for context-aware coordination. In Nicola, R.D., Ferrari, G., Meredith, G., eds.: Proceedings of the 7<sup>th</sup> International Conference on Coordination Models and Languages (COORDINATION 2004). Volume 2949 of Lecture Notes in Computer Science., Springer (2004) 38–54
2. Roman, G.C., Julien, C., Payton, J.: A formal treatment of context-awareness. In Wermelinger, M., Margaria, T., eds.: FASE. Volume 2984 of Lecture Notes in Computer Science., Springer (2004) 12–36
3. Zimmer, P.: A calculus for context-awareness. Technical Report RS-05-27, BRICS Report Series (2005)
4. Schilit, B., Adams, N., Want, R.: Context-aware computing applications. In: IEEE Workshop on Mobile Computing Systems and Applications, Santa Cruz, CA, US (1994)
5. Dey, A.K., Salber, D., Abowd, G.D.: A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human-Computer Interaction (HCI) Journal* **16 (2-4)** (2001) 97–166
6. Bardram, J.E., Hansen, T.R.: The aware architecture: supporting context-mediated social awareness in mobile cooperation. In: CSCW '04: Proceedings of the 2004 ACM conference on Computer supported cooperative work, New York, NY, USA, ACM Press (2004) 192–201
7. Cardelli, L.: Mobility and security. Lecture Notes for the Marktoberdorf Summer School 1999 (1999)
8. Milner, R.: Bigraphical reactive systems. In Larsen, K.G., Nielsen, M., eds.: CONCUR. Volume 2154 of Lecture Notes in Computer Science., Springer (2001) 16–35
9. Kjærgaard, M.B., Bunde-Pedersen, J.: A formal model for context-awareness. Technical Report RS-06-2 (2006) 26 pp.