Heil, M. Gaedke; Environment-Awareness: Quantitative Processing of Context Changes, Sixth Annual IEEE International Conference on Pervasive Computing and Communications, Pages 423-428, Hong Kong, 17-21 March 2008.

Environment-Awareness: Quantitative Processing of Context Changes

Andreas Heil[1,2] and Martin Gaedke[2]

[1]Microsoft Research
European Science Initiative
Cambridge, United Kingdom
v-aheil@microsoft.com

[2]Chemnitz University of Technology
Faculty of Computer Science
Chemnitz, Germany
firstname.lastname@cs.tu-chemnitz.de

Abstract

Comparing context of different entities is not easy at all. Since context depends on the situation of a particular entity it can be understood in different ways by various entities. In this paper we introduce an approach to consider all entities as environment in which the formal interpretation of all available context and its changes can be quantified and used to find optimal decisions within such an environment-aware system.

1. Introduction

Today's computing systems are built out of a variety of devices (computers, electronically units, embedded computers, mobile phones and personal digital assistants) running a various number of services. Each device within such a system provides a certain amount of resources in form of computing power, memory or, sensing capabilities. Due to the mobility of these devices, the setup of such a computing system and therefore, the resource allocation is changing constantly. The possibility to interconnect these devices and services in a uniform way is the goal of many attempts nowadays. However, choosing the right device or service for a certain objective within such a system is an open question. It is the question about location, time and situational information of devices and services in a given situation. In a nutshell it is the question about an *context* [1] on a global scale.

In this paper we propose an approach for quantifying changes in context, based on a structured environment hosting *environment-aware* applications.

2. A Formal Approach to Context

Following Dey, context can be understood as *any information that can be used to characterize the situation of an entity* [1]. Combining the location of entities as described by Rodden et al [2] and their current environment and situation as proposed by Shilit and Theimer [3], context can be formally expressed in the Ambient Calculus [4].

The Ambient Calculus is a process calculus where processes reside within a hierarchy of locations with the purpose of studying mobility. Entities in the Ambient Calculus are represented by named ambients whereas an ambient is defined as a bounded place where computation happens. Each ambient can have a set of unlimited subambients. Thus, the ambient system constitutes a tree structure. The tree in Figure 1 depicts such hierarchical structured ambients. The illustration on the right shows the same scenario in a more intuitive way to visualize the nesting of the corresponding ambients. There, we see ambient a, containing two distinct ambients, b and c, where b contains d and e as subambients.

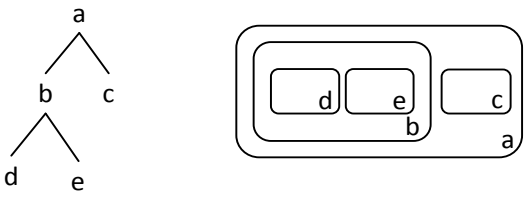

Figure 1. Ambient example in informal tree and intuitive notation.

Computation in the ambient calculus takes place in form of parallel executed processes $(P|Q)$ whereas the $\mathbf{0}$ represents the void process. A named ambient n may contain these processes as in $n[P]$. Furthermore, simultaneous existing ambients are composed in the same way as parallel executed processes. The example above can thus be described as

$$a[b[d[\mathbf{0}]|e[\mathbf{0}]]|c[\mathbf{0}]]$$

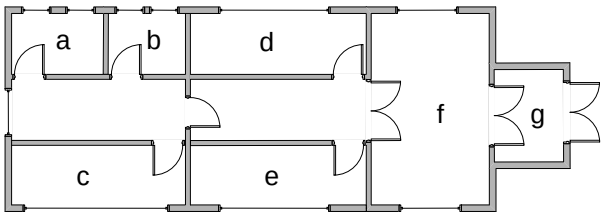

Figure 2. Building plan including office space (a, b and c) and meeting venues (d, e, f and g).

The Ambient Calculus defines an ambient by its boundaries. Thus, we can express entities such as persons, places, or objects as ambients. With these basic concepts of the calculus we can describe real world entities and constructs such as the building plan in Figure 2 in terms of the calculus.

$$g[f[e[\mathbf{0}]|d[\mathbf{0}]|r[c[\mathbf{0}]|b[\mathbf{0}]|a[\mathbf{0}]]]]$$

To separate the locations in this example furthermore, we introduced an additional ambient r. This ambient is used to identify places with restricted access rights and gives an idea about how to use virtual ambients to create fine granular constructs. In a visual way, we can depict this system in the form of a tree as seen in Figure 3. Thus, we can use Mobile Ambients to describe context as introduced before in a rather formal way.

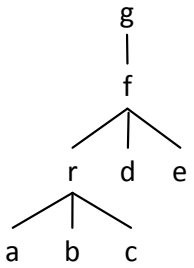

Figure 3. Hierarchical ambient model depicting context based on the example building plan.

By adding processes to certain ambients, we can describe their current situation in more detail. Currently running processes describe a part of the recent context of an entity while previously running processes can be retained in the notation of the Ambient Calculus, describing previous situations. Both together allow us to reason about the current context of an entity, based on its recent situation as well as on its history. For the rest of this paper, however, we focus on the aspects of the location and movement of ambients rather than on processes, while the concepts introduced in this paper can be easily applied also to processes and their movements.

3. Formal Change of Context

The system of ambients can be expressed as graph $G = \{V, E\}$ where the nodes $V(G)$ represent the ambients and the edges $E(G)$ represent their spatial relationships. Mobile Ambients provide a set of mobility primitives to describe the movement of ambients and processes within the system. Based on these primitives, we define a set of operations to be applied on G.

Definition (open): *A node v with parent v' is deleted from its parent by revealing its boundaries. The children of v are inserted as subset as children of v'.*

Definition (insert): *A node v is inserted as child into node v'. v' thus becomes the parent node of v.*

Definition (remove): *A node v is removed from its parent node v' without revealing its content.*

The *open* operation is equivalent to its corresponding capability in the Ambient Calculus. The movement capabilities (namely *in* and *out*) of an ambient can be expressed as combinations of *insert* and *remove* operations along the edges of the tree structure. Therefore, each edge is marked with a weight function $\omega(e)$ where $e \in E(G)$, representing the costs by moving along the particular edge. Since each ambient requires a unique name, the graph can be understood as a rooted, labeled unordered tree. Tree edit operations such as *insert* and *delete* are well studied in the tree-to-tree correction problem [5]. This approach allows determining the optimal costs for transforming a tree T into an a tree T'. These costs can be computed for ordered labeled trees [6]. For unordered labeled trees these problems are hard to solve, some of the problems are even NP-complete [5]. Introducing a constrained edit distance, e.g. Zhang proposed an algorithm to solve these costs for unordered labeled trees in sequential time [7].

Due to the movement capabilities in the Ambient Calculus, an ambient has to follow a distinct path within the ambient hierarchy. The movement of an ambient can be identified as the shortest path between its origin and destination within the graph [8]. However, not every ambient is supposed to be moved such as the rooms in Figure 2. If necessary, we can address these circumstances using immobile ambients as introduced by Levi and Sangiorgi [9]. As stated by Cardelli and Gordon, the movement of an ambient in the real world can affect physical entities such as a laptop but also of other constructs such as an electronic document that is copied on such a laptop. While the

movement capabilities of ambients are limited by their corresponding mobility in the real world (e.g. rooms vs. laptops). Furthermore, the *open* capability cannot be applied to physical entities since it would imply the destruction of the corresponding entity. Based on these assumptions we can calculate the costs for altering the context of ambients as shown in the next section.

4. Quantifying Environment-Awareness

While context-awareness is, e.g. understood to be based on relevant information of an solely entity [1], it varies by different perspectives of different entities. Such different corresponding collections of nearby objects and their changes over time [3] are depicted in Figure 4. Consequently, the edge between nodes n_1 and n_2 can have different meanings to either n_1 or n_2.

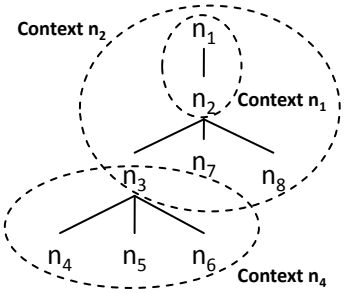

Figure 4. Different Context due to the relevant information of an entity providing different meanings to entities.

Environment-awareness however shall provide a holistic approach to deal with information in a global manner, based on contexts available all over the system. Therefore, it should allow a direct comparison of situation regardless of the particular perspective. Hence, we have to put forward a way to evaluate context of different entities in a consistent way as proposed in the following section.

4.1 Ambient-based Costs-Function

The approach to quantify context changes in a ambient-based hierarchy is based on the research by Tai on the tree-to-tree correction problem [6].

Let Σ be the alphabet over the ambient names in a given system while ε is an empty symbol with $\varepsilon \notin \Sigma$. That way we can define $\Sigma_\varepsilon := \Sigma \cup \varepsilon$. Following [6], we define an edit operation s as $n_1 \to n_2$ where $(n_1, n_2) \in (\Sigma_\varepsilon \times \Sigma_\varepsilon) \backslash (\varepsilon, \varepsilon)$ on names $n_1, n_2 \in \Sigma_\varepsilon$. The operation is an insertion if $n_1 = \varepsilon$ and an open operation if $n_2 = \varepsilon$. If $n_1, n_2 \in \Sigma$ the operation is a movement operation where n_1 is the subject moved in or out of $n_2$where n_2 is an ancestor of n_1 for an *out* movement or n_1, n_2 are siblings for an *in* movement.

Let S be a sequence of edit operations $s_1, s_2, \dots, s_m$where $m = |S|$ is the number of steps in the sequence. We define a cost function $\gamma: (\Sigma_\varepsilon \times \Sigma_\varepsilon) \backslash (\varepsilon, \varepsilon) \to \mathbb{R}$. Thus, $\gamma(n_1 \to n_2)$ assigns to $s(n_1 \to n_2)$ a real number representing the costs of this edit operation and $s(n_1 \to n_2)$ gives us the edge e involved in this movement operation. For each movement operation along an edge e we have to consider the eventual additional costs $\omega(e)$ arising by this operation. Furthermore, two general assumptions can be made about the cost-function γ:
A reflexive edit operation on a node n_1 does not cause costs at all:

$$\gamma(n_1 \to n_1) = 0$$

Secondly, the tree structure implicates that the composition of edit operations cannot exceed the costs of their single operations:

$$\gamma(n_1 \to n_2) + \gamma(n_2 \to n_3) \geq \gamma(n_1 \to n_3)$$

Finally, for a sequence S of edit steps we get the total costs

$$\gamma(S) = \sum_{i=1}^{m} \gamma(s_i) + \omega(s_i)$$

where $\omega(e_j) = 0 \; \forall e_j = \emptyset$ with $j \in [1, |E(G)|]$.

In the next section we show how these costs allow us to reason about the potential context changes in a given system.

4.2 Cost-based Decisions

The quantification of ambient movements in the form of the metric cost-function γ allows us directly to compare the utility of possible changes in context as shown in the following example, describing the movement of an ambient n_3 through the ambient hierarchy.

$$n_1[\, mv\, n_3 in\, n_5 | n_2[mv\, n_3 out\, n_2.\, \mathbf{0} | n_3[\,] | n_4[\,]] | n_5[\,]]$$
$$\to\; n_1[\, mv\, n_3 in\, n_5 | n_2[n_4[\,]] | n_5[\,] | n_3[\,]]$$
$$\to\; n_1[\, n_2[n_4[\,]] | n_5[\, n_3[\,]]]$$

For the movements depicted in Figure 5 we can calculate the costs as

$$\gamma_1 = \gamma_{out}(n_3, n_2) + \omega(e_1) + \gamma_{in}(n_3, n_5) + \omega(e_5).$$

With n_4 being a possible alternative to n_3 we state a second cost function based on the movements of n_4

$$\gamma_2 = \gamma_{out}(n_4, n_2) + \omega(e_3) + \gamma_{in}(n_4, n_5) + \omega(e_5).$$

Comparing the various costs γ_p for p possibilities, we can chose from a set of competitive alternatives.

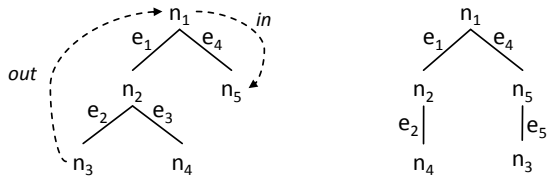

Figure 5. Simple ambient movement

4.3 Addressing Ambiguous Movements

Movement capabilities in the Ambient Calculus are ambiguous. Thus, it is not foreseeable which capability eventually executes.

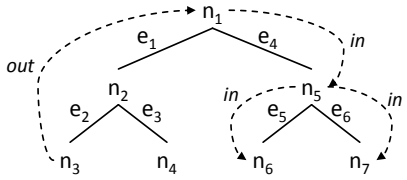

Figure 6. Ambiguous Movements.

Figure 6 shows such ambiguousness. Therefore n_5 is defined as below. When n_3 enters the scope of n_5 either of the possible capabilities execute.

$$n_5[n_3 mv\ in\ n_6.\mathbf{0}\ |n_3 mv\ in\ n_7.\mathbf{0}]$$

Such issues are addressed as *plain* and *grave interferences* by Levi and Sangiorgi in [9]. The proposed solution in form of co-actions binds each action to its counterpart. However, this is hard to be applied to real systems, since it requires a-prior knowledge of the system's components interactions. Instead, computing the costs valid alternatives of these actions provides an additional factor to make the decision which opportunity to choose.

The decision to follow the path e_5 or e_6 can thus be supported by evaluating the costs of the corresponding movements of node n_3.

4.4 Influenced by Limited Resources

In many theoretical approaches, resources are considered to be available unlimited. For example in the Ambient Calculus, the replication primitive $!P$ can be used to replicate a unlimited number of processes P. In a real system, however, we face very strict constraints in terms of resources such as memory, bandwidth, CPU cycles and of course power consumption. When thinking of costs to modify the setup of an environment, we should think of available budgets, budget restrictions as well as the utility of resources. Therefore, we consider the systems of interest as closed ecosystems with a limited number of resources and participants. That way, we can study interesting mechanisms within such a system.

For our model, the assumption is made that no system component offers resources to its own benefit. Rather the ambition of the system is to avoid idle resources. Thus, irrespectively of the costs, the available devices do not vary. Based on this postulation, we can depict the supply curve as a vertical line.

The demand curve represents the required amount of resource for any possible costs. One possible demand curve such as in Figure 7 shows a negative gradient. The number of required resources though is limited by x_{max}, representing the actual needed resources within the system. Additional resources appear to be idle, not being used within the system nevertheless if they have no costs. On the other side, the minimum of required resources is indicated by x_{min}. This amount of resources is required to provide the minimum functionality of the system. This minimum can be understood as an indispensable requirement of resources, for example to avoid system failures. Hence, the actual costs to satisfy these needs are insignificant.

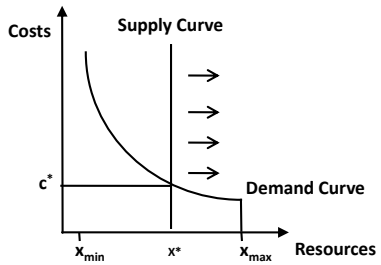

Figure 7. Satisfying the demand by adding further resources to a system.

A lack of resources can be simply resolved by scaling-up or scaling-out of the required resources. Due to a higher availability, the supply curve shifts to the right. Satisfying all demands by simply increasing the amount of available resources is an intuitive approach but in generally not to be accomplishable.

Increasing the amount of available resources is not always possible. For example time to get a respond cannot be affected directly and many other resources such as power consumption are exogenous factors not to be influenced at all. Allocation of additional

hardware for satisfying requests is often related to monetary expenditure or not possible in a short-term. Therefore, an optimum choice of resources already provided is required.

The availability of resources within such a system is well studied in different areas of research such as the transport economy [10]. There, a utility function $U(\vec{c})$ of attributes is used to describe the demands of individuals. The vector $\vec{c} = (c_1, c_2, \dots, c_n)$ thus can be used to describe the various costs of the resources. The attributes might be CPU cycles, memory, storage or certain sensing capabilities, latency but also qualitative attributes such as simplicity to use a certain API or interoperability of a system. The utility function can thus be expressed as a linear function with unknown coefficients $\alpha_1, \alpha_2, \dots, \alpha_n$. These coefficients describe the weight of the costs of each attribute.

$$U(c_1, c_2, \dots, c_n) = \alpha_1 c_1 + \alpha_2 c_2 + \cdots + \alpha_n c_n$$

With a sufficient large set of observations, the coefficients can be identified based on statistical methods as proposed in [10]. The application of a utility function thus, is also qualified to support decisions based on the quantification of resources. Consequently, the evaluation is not only restricted to a single resource, furthermore a combination of all required resources can be evaluated to draw a decision.

5. Related Work

The CONAWA approach proposes a mechanism for modeling complex and interlinked sets of context-information by extending the Ambient Calculus with additional constructs [11, 12]. In this approach, multiple tree structures are used for describing the context, e.g. by providing one tree for location and a further one for activities. However, the approach differs from similar approaches where context-awareness is put on one level with location-awareness and does not limit itself to locations. In contrast to the approach proposed in this paper, CONAWA does extend the Ambient calculus with new constructs and capabilities. The Context UNITY [13, 14] approach allows to reason about mobility and explores the process by which the Mobile UNITY [15] model is applied. In this formal model for context-aware systems, location entailing agents are represented by state transition systems that are used for state-based formal reasoning. In such a system, context-changes are noticed as spontaneous state transitions outside of the agent's control. Also the Ambient Calculus does use mobile agents to describe mobility and interactions of the system's components. However, the Ambient Calculus is not limited to agent-based systems and allows us to describe various real-world scenarios in the presence of any kind of agents.

6. Summary and Outlook

In this paper we introduced the quantitative construction of context changes based on the Ambient Calculus. The Ambient Calculus provides the capability to reason about the location and the mobility of entities such as persons, places or objects in the form of ambients. The contextual information is directly educed from the structure of such ambients. By calculating the costs for acquiring resources, we propose an approach for choosing from different alternatives. The quantification leads directly to comparability of context, which is a first step towards environment-awareness, where different context is observed in a collective way.

The approach presented in this paper is an introduction into quantifying context-changes arising within our current models of Web-based applications. Within these systems, Web-based systems provide access to physical data providers such as real-world sensor. Due to its nature, however, the Web provides an abstract view on resources where resources are identified by their Uniform Resource Identifiers rather than by their physical location. While knowing about the hierarchical structure of the overall system, the proposed approach allows us to reason about the context-changes and thus the appropriate usage of resources in such a hierarchically ordered system, even without knowing about the exact physical location of the corresponding resources. In our future research we want to create a representative set of observations to determine the coefficients for the proposed utility function. The utility function was chosen in different fields of research as linear function since statistical methods can be easily applied to this function. However, we have to evaluate if different types of utility functions can express the preferences within a computing system in a better way.

7. References

[1] A. K. Dey, "Understanding and Using Context," *Personal and Ubiquitous Computing*, vol. 5, pp. 4-7, January 2001.

[2] T. Rodden, K. Cheverst, N. Davies, and A. Dix, "Exploiting Context in HCI Design for Mobile Systems," in *First Workshop on Human Computer Interaction with Mobile Devices*, Glasgow, Scotland, 1998.

[3] B. N. Shilit and M. M. Theimer, "Disseminating active map information to mobile hosts," *IEEE Network*, vol. 8, pp. 22-32, Sep/Oct 1994 1994.

[4] L. Cardelli and A. D. Gordon, "Mobile Ambients," in *First International Conference on Foundations of Software Science and Computation Structures (FoSSaCS '98) at ETAPS'98*, Lissabon, Portugal, 1998, pp. 140-155.

[5] P. Bille, "A Survey on Tree Edit Distance and Related Problems," *Theoretical Computer Science*, vol. 337, pp. 217 - 239, June 2005.

[6] K.-C. Tai, "The Tree-to-Tree Correction Problem," *Journal of the ACM*, vol. 26, pp. 422 - 433, July 1979.

[7] K. Zhang, "A Constrained Edit Distance Between Unordered Labeled Trees," *Algorithmica*, vol. 15, pp. 205-222, 1996.

[8] E. W. Dijkstra, "A Note on Two Problems in Connexion with Graphs," *Numerische Mathematik*, vol. 1, pp. 269-271, December 1959.

[9] F. Levi and D. Sangiorgi, "Controlling Interference in Ambients," in *27th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, Boston, Masachusets, USA, 2000, pp. 352 - 364.

[10] T. A. Domencich and D. McFadden, *Urban Travel Demand: A Behavioral Analysis* Amsterdam, Oxford: North-Holland Publishing Company, 1996.

[11] M. B. Kjærgaard and J. Bunde-Pedersen, "Towards a Formal Model of Context Awareness," in *International Workshop on Combining Theory and Systems Building in Pervasive Computing, PERVASIVE 2006* Dublin, Ireland, 2006.

[12] M. B. Kjærgaard and J. Bunde-Pedersen, "A Formal Model for Context-Awareness," Department of Computer Science, University of Aarhus, Aarhus RS-06-2, February 2006.

[13] C. Julien, J. Payton, and G.-C. Roman, "Reasoning About Context-Awareness in the Presence of Mobility," *Electronic Notes in Theoretical Computer Science* vol. 97, pp. 259-276, 22 July 2004.

[14] G.-C. Roman, C. Julien, and J. Payton, "Modeling Adaptive Behaviors in Context UNITY," *Theoretical Computer Science*, vol. 376, pp. 185-204, May 2007.

[15] G.-C. Roman and P. J. McCann, "A Notation and Logic for Mobile Computing," *Formal Methods in System Design* vol. 20, pp. 47-68, 2002.