Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Seminarausarbeitung AW1

Jens Ellenberg

Ein Wecker in einem ubicom Haus

Fakultät Technik und Informatik
Department Informatik

Jens Ellenberg

Ein Wecker in einem ubicom Haus

Seminarausarbeitung AW1 eingereicht
im Studiengang Informatik Master
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Abgegeben am 27. Februar 2010

Inhaltsverzeichnis

1 Einleitung

1.1 Motivation

Die Vision eines Hauses, in dem Computer den Menschen unterstützen, gibt es schon seit mehreren Jahrzehnten. Marc Weiser hatte in den 90 Jahren eine interessante Vision veröffentlicht [9], welche die Welt der Informatik sehr geprägt hat. In dieser Vision beschreibt er einen Tag aus dem Leben von Sal, einem Menschen der in einer möglichen Zukunft lebt. Sal wird morgens von dem Computer der Wohnung geweckt. Es wurde aber offen gelassen, wie der Computer darauf kommt Sal zu wecken. Dieses ist eine interessante Frage, die in der nachfolgenden Arbeit beantwortet werden soll. Was sollte so ein Wecker der Zukunft können und wie könnte der Computer diese Aufgaben durchführen?
Um die Aufgaben eines Weckers zu beschreiben sei hier ein beispielhafter Verlauf eines Morgens an einem Wochentag gegeben:

Abbildung 1.1: Ein handelsüblicher Wecker

6:00 Aufstehen

6:15 Waschen/Duschen

6:45 Frühstücken

7:15 Zur Arbeit fahren

8:00 Arbeitsbeginn

Nehmen wir nun an, dass für das Wecken ein handelsüblicher Wecker[1] eingesetzt werden soll. Dann können verschiedene Dinge einen regulären Ablauf verhindern. Beispielsweise könnte der Wecker am Abend nicht gestellt werden; der Arbeitsbeginn könnte sich verschoben haben und man vergisst diesen Umstand beim Stellen des Weckers; man wacht schon

[1] handelsüblicher Wecker meint hier das, was man gegenwärtig bekommt, wenn man in einem Kaufhaus nach einem „Wecker“ fragt.

um 5:00 auf und will daher nicht um 6:00 geweckt, sondern um 7:15 an das Verlassen der Wohnung erinnert werden oder man schläft nicht im Bett, an dem der Wecker klingelt.

Damit diese Probleme bei einem Wecker der Zukunft nicht auftreten, ist es notwendig den Wecker mit künstlicher Intelligenz auszustatten. Dadurch soll der Wecker die folgenden Fähigkeiten aufweisen:

- Sich selbst stellen
- Sich selbst den Terminen anpassen
- Sich dem Bewohner anpassen
- Da aktiv werden, wo er gebraucht wird

1.2 Problemstellung

Im iFlat [7] der HAW Hamburg soll im Rahmen des Living Place Hamburg [3] ein „Wecker 2.0“ realisiert werden. Der Wecker soll durch Computer gesteuert werden, die in dieser Wohnung vorhanden sein werden. Der Wecker soll den Bewohner auf Termine hinweisen, wenn der Bewohner dieses erwünscht. Die Termine können der Beginn der täglichen Arbeit sein oder es kann sich um einen Besuch von Freunden am Abend handeln. Hierbei sollen die Zeitpunkte dynamisch berechnet werden, an welche der Bewohner erinnert werden soll. In die Berechnung sollen die Gewohnheiten des Bewohners aufgenommen werden, wie beispielsweise das morgendliche Waschen. Auch andere Bedingungen, die den Zeitpunkt verschieben können, sollen berücksichtigt werden, beispielsweise wenn der Bewohner nicht mehr im Bett schläft, sondern schon wach ist oder es bei den öffentlichen Verkehrsmitteln zu Ausfällen kommt. Des Weiteren soll der Wecker immer dort die Aufmerksamkeit des Bewohners erregen, wo er sich gerade in der Wohnung aufhält.

1.3 Ziel dieser Ausarbeitung

In dieser Arbeit soll der „Wecker 2.0“ skizziert werden. Hierzu ist es notwendig, zu analysieren, für welche Aufgaben ein Wecker in der Regel eingesetzt wird. Aus dieser Analyse werden dann die Probleme gefolgert, die mit einer handelsüblichen Version eines Weckers verbunden sind und die mit dem „Wecker 2.0“ gelöst werden sollen. Anschließend werden Lösungsansätze erarbeitet und Voraussetzungen aufgeführt, die erfüllt sein müssen damit der Wecker in der Wohnung der Zukunft verwirklicht werden kann.

2 Analyse

2.1 Was ist ein Wecker?

Im ersten Schritt wird analysiert, welche Erwartungen an die Funktionen eines Weckers gestellt werden, damit diese Erwartungen beim „Wecker 2.0“ umgesetzt werden können. Neben den Funktionen wird außerdem herausgearbeitet, wie und wann ein Wecker seine Funktion erfüllt.

2.1.1 Was macht ein Wecker?

Ein Wecker wird, wie es der Name schon sagt, in der Regel für das Wecken nach der nächtlichen Schlafperiode eingesetzt. Dieses ist aber nicht die einzige Funktion, die er erfüllen kann. Man kann den Wecker auch dazu einsetzen, anzuzeigen, wann man die Wohnung verlassen will oder wann beispielsweise 60 Minuten vergangen sind, die ein Kuchen im Backofen benötigt. Beispiele für den Einsatz eines Weckers sind:

- An Termine erinnern (Arbeitsbeginn; Party; Geburtstagsfeier)
- Schlaf beenden (Für die Arbeit; Um den Tag nicht zu verschlafen; Um einen Mittagsschlaf zu beenden)
- Anzeigen, wann eine Zeitspanne abgelaufen ist (Beim Kuchen backen; Die Arbeitszeit)

Zusammenfassend kann man aus den oben genannten Punkten zwei Einsatzarten unterscheiden.

Anfang eines Zeitintervalls: In diesem Fall ist nicht der tatsächliche Zeitpunkt des Zeitintervalls interessant, sondern ein bestimmter Zeitpunkt vor dem Anfang, der dem Bewohner genügend Zeit lässt, sich auf das Zeitintervall vorzubereiten. Beispielsweise muss man vor dem Arbeitsbeginn in der Regel noch den Weg zur Arbeit zurücklegen.

Ende eines Zeitintervalls: Hier ist der genaue Zeitpunkt ohne Vorbereitungszeit relevant. Es soll der Zeitpunkt angezeigt werden, der das Ende des Intervalls bestimmt.

2.1.2 Wie macht der Wecker etwas?

Der Wecker erregt die Aufmerksamkeit des Bewohners. Dieses kann durch alle Sinne geschehen. Ein herkömmlicher Wecker erreicht das über akustische Signale. Der Wecker in einem Handy nutzt unter anderem auch den Vibrationsalarm, er erregt also die Aufmerksamkeit des Besitzers über dessen Tastsinn. Visuelle Mittel sind außerdem denkbar. Beispielsweise könnte in einem Fernsehbild ein Symbol eingeblendet werden, das die Aufmerksamkeit des Zuschauers erregt.

Wenn die Aufmerksamkeit des Bewohners sichergestellt ist gibt es in der Regel zwei Möglichkeiten, wie der Wecker sich weiter verhalten kann.

- Er beendet seine Aktionen, den Bewohner zu erinnern
- Er wiederholt seine Aktionen, um die Erinnerung sicher zu stellen

2.1.3 Wann macht er Wecker etwas?

Wenn der Wecker zum Anzeigen des Anfangs eingesetzt wird, dann ist für den Bewohner nicht der tatsächliche Anfang interessant, sondern es muss auch die Zeit zur Vorbereitung mit eingeplant werden. Wenn man beispielsweise um sieben Uhr Arbeitsbeginn hat, dann klingelt der Wecker nicht um sieben Uhr. Er klingelt so früh, dass man sich noch zumindest anziehen kann und auch Zeit verbleibt, den Weg zur Arbeit zurückzulegt. Wenn man abends zu einer Party eingeladen ist, möchte man sich in der Regel noch für die Party zurecht machen und auch hier muss man Zeit für die Wegstrecke einplanen. Dieser Vorlauf hängt von persönlichen und äußeren Einflüssen ab.

Ein Beispiel für persönliche Einflüsse ist das Frühstücken vor der Arbeit. Manche Bewohner frühstücken nicht, andere wiederum trinken eine Tasse Kaffee und wieder andere Bewohner bevorzugen ein ausgelassenes Frühstück. Ein Beispiel für äußere Einflüsse ist die Wegstrecke bis zur Arbeit. Es kann schneien oder die Sonne scheint oder die öffentlichen Verkehrsmittel können einen veränderten Streckenplan nutzen. Diese und andere Bedingungen können den Zeitpunkt verschieben, an dem der Bewohner geweckt werden soll.

2.2 Anforderungen

Im Folgenden wird der „Wecker 2.0“ konkretisiert. Es werden die Funktionen und Rahmenbedingungen beschrieben, die bei der Umsetzung des Projektes wichtig sind.

2.2.1 Funktionale Anforderungen

In der Motivation (Kap.: 1.1) wurden verschiedene Szenarien beschrieben, in denen ein handelsüblicher Wecker nicht so funktioniert, wie es vom Bewohner gewünscht wird. Um die Funktionen des Weckers zu verbessern, sollten fehlerhafte Einstellungen durch den Bewohner vermieden werden und ebenso sollten Änderungen der Termine direkt angepasst werden. Außerdem sollte der Wecker zum einen auf den aktuellen Zustand des Bewohners reagieren aber auch nur das tun, was der Bewohner tatsächlich wünscht. Der Bewohner sollte auch da auf die Termine aufmerksam gemacht werden, wo er sich in der Wohnung gerade befindet.

Damit die oben genannten Ideen vom „Wecker 2.0“ erfüllt werden können, ergeben sich die folgenden Anforderungen:

- Dem Wecker müssen die Termine und deren Änderungen bekannt sein.
- Der Bewohner muss neue Termine setzen können.
- Der Wecker muss Zugang zu Informationen außerhalb der Wohnung haben (z.B. Wetter).
- Der Wecker muss wissen, was der Bewohner macht und gemacht hat.
- Der Bewohner muss die Funktionen des Weckers kontrollieren können.
- Der Wecker muss den Alarm an verschiedene Geräte im Haus übermitteln können.

2.2.2 Nichtfunktionale Anforderungen

Die Frage, „Welche Änderungen würden Sie gerne beim Entwurf eines Weckers sehen“, ergab in einer Umfrage, dass sich fast alle Befragten die folgenden Punkte angaben [6]: Interaktion; Personalisierung und Methode des Weckens.

Weitere Befragungen haben ergeben, dass ein Wecker nur dann in das tägliche Ritual einbezogen wird, wenn er Sicherheit und Komfort vermittelt. Das Gefühl von Sicherheit und Komfort entsteht aus einfacher Handhabung und wenn der Wecker einen zeitgemäßen Eindruck vermittelt, dann wird dadurch das Gefühl von Sicherheit und Vertrauen in den Wecker verstärkt. Flexibilität in der Handhabbarkeit ist auch ein wichtiges Merkmal, dass dazu beiträgt, dass der Wecker in das tägliche Ritual eingebunden wird. Eine wichtige Erkenntnis ist, dass die Sicherheit wichtiger ist, als die Sinne für Tradition, Neuheit oder die persönlichen Neigungen [6].

3 Lösungsansätze

Um die in der Analyse beschriebenen Funktionen zu erfüllen benötigt der Wecker verschiedene Kenntnisse. Diese Kenntnisse lassen sich in

- Wissen über die Zeitplanung
- Wissen über den Zustand des Bewohners
- Wissen aus externen Quellen

einteilen. Im Folgenden wird beschrieben, wie der Wecker zu diesem Wissen gelangen kann.

3.1 Wissen über die Zeitplanung

In dem Kapitel über die Anforderungen (Kap.:2.2) wurde gezeigt, dass dem Wecker Wissen über die Zeitpunkte zur Verfügung gestellt werden muss. Es gibt verschiedene Zeitpunkt-Arten, an denen ein Wecker aktiv werden kann. Neben der Einteilung in Anfangs- und Endzeitpunkt, können noch weitere Unterscheidungen gemacht werden. Die folgenden Beispiele stehen für die Kategorien: explizit geplante Zeitpunkte, implizit geplante Zeitpunkte und ungeplante Zeitpunkte.

Termine sind Zeitpunkte in der Zukunft, die geplant sind. Hierbei kann es sich um berufliche oder private Termine handeln.

Gewohnheiten spiegeln Muster im Verhalten des Bewohners wider. Wenn der Bewohner beispielsweise immer um 20:00 die Tagesschau im Fernsehen sieht, könnte man davon ausgehen, dass er diese auch in der Zukunft nicht verpassen will.

Backzeitraum wäre hier ein ungeplanter Zeitpunkt in der nahen Zukunft. Diese Zeitpunkte ergeben sich aus dem täglichen Leben und sind nicht planbar und damit auch nicht vorhersehbar.

Grundsätzlich gibt es zwei Möglichkeiten, dem Wecker Zeitpunkte zugänglich zu machen: automatisch und manuell. Explizit und implizit geplante Zeitpunkte können dem Wecker automatisch übergeben werden, da sie für die Zukunft feststehen. Bei der automatischen Erfassung muss besonders darauf geachtet werden, dass der Wecker nur die Aktionen ausführt, die der Bewohner auch wünscht. Jeder Fehler senkt das Vertrauen in den Wecker und damit auch die Wahrscheinlichkeit, dass der Wecker in das tägliche Ritual integriert wird [6]. Ungeplante Zeitpunkte müssen in jedem Fall dem Wecker manuell übergeben werden, da sie in der Regel nicht vorhersehbar sind.

3.1.1 Automatische Erfassung

Ein elektronischer Terminkalender ist ein logischer Schritt, wenn man die Termine des Bewohners dem Wecker automatisch zugänglich machen möchte. Hierfür muss der Terminkalender angepasst werden, so dass er alle Termine dem Wecker zur Verfügung stellt. Dabei muss entschieden werden, wann die Termine vermittelt werden. Wenn sie zu spät bekannt gemacht werden, dann könnte ein möglicher Alarm nicht mehr rechtzeitig ausgelöst werden. Je früher die Termine übermittelt werden, desto mehr Termine muss der Wecker speichern und es müssen auch alle Änderungen der Termine kommuniziert werden. Daher sollte hier ein Mittelweg gefunden werden. Gewohnheiten könnten erfasst und anschließend ausgewertet werden. Dabei sind jedoch Fehler in der Auswertung problematisch. Wie oben schon erwähnt, führt jeder Fehler zu mehr Misstrauen gegenüber dem Wecker. Eine mögliche Lösung wäre das Ergebnis einer Auswertung erst nach einer Rücksprache mit dem Bewohner in die tägliche Routine zu übernehmen.

3.1.2 Manuelle Erfassung

Für die manuelle Erfassung muss eine Möglichkeit geschaffen werden, mit der der Bewohner dem Wecker die Zeitpunkte eingibt. Hierbei kann es sich um geplante- oder ungeplante Zeitpunkte handeln. Damit die Eingabe überall in der Wohnung geschehen kann, sollten die Eingabe vom Wecker getrennt werden. Für diese Trennung bietet sich die Blackboard Architektur des Eventheap an. Über das Blackboard werden die Zeitpunkte von der Eingabe an den Wecker übergeben. So hat jedes Modul in der Wohnung die Möglichkeit mit dem Wecker zu kommunizieren, wenn es mit dem Backboard interagieren kann.

3.2 Wissen über den Zustand des Bewohners

In dem Kapitel über die Anforderungen (Kap.: 2.2) wurde unter anderem gezeigt, dass der Wecker Wissen über den Zustand des Bewohners braucht, damit die Verbesserungen umgesetzt werden können. Wenn man sich überlegt, was „Wissen über den Bewohner“ bedeutet, kommt man zu den folgenden Fragen:

- Wo ist er? (Ort)
- Was macht er? (Aktion)
- Wer ist er? (Identität)
- Wann macht er etwas? (Zeit)

Diese Fragen entsprechen der Beschreibung von Context Aware Computing aus dem Paper [1]. Allgemein wird in dem Paper „Kontext“ wie folgt definiert:

> „Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves.“

Das bedeutet, dass der „Wecker 2.0“ ein Programm sein muss, welches sich dem Kontext seiner Umgebung bewusst ist. Aus dem Kontext kann dann das Verhalten des Weckers angepasst werden, wie es auch in [5] beschrieben ist. Im Folgenden wird beschrieben, was beim „Context Aware Computing“ beachtet werden muss.

3.2.1 Grundlagen des Context Aware Computing

In [2] wird gesagt, dass es drei relevante Theorien zum Thema Kontext gibt. Diese Theorien zeigen, dass „Kontext“ ein sich ständig änderndes und situationsabhängiges Konstrukt ist.

Situationsbedingte Aktion: In dieser Theorie wird davon ausgegangen, dass die ausgeführten Aktionen durch die vorherrschende Aktion bestimmt wird. Pläne existieren, aber sie werden laufend der Situation angepasst und legen daher keinen bindenden Ablaufplan fest.

Aktivitätstheorie: Hier definieren die Aktionen selbst den Kontext. Es müssen bei der Erstellung des Kontextes nicht nur die beteiligten Personen und Artefakte betrachtet werden, sondern auch die Wünsche und Ziele der beteiligten Personen, da diese sich ändern können.

Soziale Netzwerke: Je nachdem, in welchem sozialen Netzwerk sich eine Person befindet, definiert das den aktuellen Kontext. Dabei muss beachtet werden, dass sich eine Person in vielen verschiedenen Netzwerken gleichzeitig befinden kann und ja nach Kombination ein anderer Kontext entsteht.

Um diese verschiedenen Arten von Kontext zu erkennen müssen Informationen gewonnen werden. Hierbei kann man zwei Arten von Informationen unterscheiden:

externe Informationen: Dieses sind physikalische Werte und Eigenschaften der Einrichtung und Personen des Hauses. Externe Informationen können mit verschieden Sensoren relativ einfach erfasst werden. Um diese Informationen anschließend mit bestimmten Kontexten in Verbindung zu bringen, könnte hingegen eine komplexe Aufgabe sein.

interne Informationen: Dieses sind abstrakte Informationen über die Interessen und Ziele einer Person. Hierzu gehört auch, was die Person bisher getan hat und welche Fortschritte ihre aktuellen Aktivitäten aufweisen. Diese Informationen kann man sehr viel schwieriger gewinnen, als die externen Informationen. In [2] geht man davon aus, dass man diese Informationen abschätzen muss oder „informiertes Raten“ angewandt werden könnte, um sich den internen Informationen anzunähren.

Bei einem bestimmten Kontext kann es sich um einen einfachen oder komplexen Kontext handeln. Ein einfacher Kontext ist etwas, das direkt von Sensoren abgeleitet werden kann. Beispielsweise könnte ein RFID Chip, den der Bewohner bei sich trägt, direkt den Schluss zulassen, dass „der Bewohner in der Wohnung anwesend ist“. Für einen komplexeren Kontext („der Bewohner Schläft in seinem Bett“) reicht es nicht aus, festzustellen, dass der Bewohner sich gerade im Bett befindet. Hierfür sind mehrere Daten notwendig, die sich vielleicht auch erst aus anderen Kontexten ergeben. Es muss zum Beispiel der Kontext „der Bewohner ist in der Wohnung“ gegeben sein, damit er auch in seinem Bett schlafen kann.

3.2.2 Einschränkungen im Context Aware Computing

Bei der Definition von Kontexten und deren Erkennung gibt es auch verschiedene Probleme. Zum einen ist die Menge der Kontexte, die ein Programm erkennen kann, endlich. Das liegt daran, dass ein Kontext in der Regel erst definiert werden muss, damit ein Programm diesen Kontext auch erkennen kann. Da sich die Anforderungen an ein Programm im Laufe der Zeit ändern, sollte hier darauf geachtet werden, dass es möglich ist, die bekannten

Kontexte zu erweitern. Andererseits gibt es auch Kontexte, die sich nicht definieren lassen, weil für den Kontext keine Vorschriften existieren. Ein weiterer Punkt ist die Zeit. Ein Kontext kann eine zeitlich begrenzte Lebensdauer haben. Diese Lebensdauer hängt von den internen und externen Umständen ab, unter denen der Kontext definiert ist [2]. Zum Beispiel könnte das Essen in der Küche um 13:00 das Mittagsessen sein, aber am Wochenende ist es das Frühstück.

Ein weiteres Problem bei der Kontext-Erkennung sind technische Einschränkungen. Ein Sensor, der eingesetzt wird, um Kontext zu erkennen, hat in der Regel produktbedingte Fehlergrenzen innerhalb denen er funktioniert. Eine weitere technische Einschränkung ist das Ausfallen der Sensoren. Neben den technischen Einschränkungen gibt es auch noch Störungen aus der Umgebung. Diese Störungen können die eigentlichen Messungen verfälschen oder unterdrücken. Diese Fehler können auch durch eine falsche Verwendung der Einheiten durch den Nutzer hervorgerufen werden. Daher ist es bei der Definition von Kontexten notwendig sich um die Fehlertoleranz Gedanken zu machen um ein unerwünschtes Verhalten des Programms zu vermeiden [10].

3.3 Wissen aus externen Quellen

Um den Bewohner möglichst gut zu unterstützen, reicht das Wissen, das innerhalb der Wohnung gewonnen werden kann, nicht aus. Der Weg von der Wohnung zum Termin, der im Terminkalender angegeben ist, führt in der Regel aus der Wohnung hinaus. Die Zeit, die es dauert diese Strecke zu überbrücken, wird durch verschiedene externe Gegebenheiten beeinflusst. Beispielsweise sollte im Winter für einen vereisten Fußweg mehr Zeit eingeplant werden als im Sommer, wenn keine Gefahr besteht auszurutschen. Des Weiteren beeinflussen Baustellen die Wegezeit. Baustellen können auf der Straße, wie auch bei den öffentlichen Verkehrsmitteln für Behinderungen sorgen.

Um dieses Wissen zu erlangen, bietet sich das Internet als Quelle an. Über das Internet können Staus, Baustellen oder das Wetter abgefragt werden. Mit diesen Daten ist es anschließend möglich, die Zeit besser zu berechnen, die notwendig ist, um von der Wohnung zum Ort der Termins zu gelangen. Hierbei können, wie auch bei anderen Quellen, Ausfälle auftreten oder falsche Daten werden an den Wecker übermittelt. Daher ist es auch bei diesen Daten notwendig, jene Probleme mit einzuplanen.

4 Zusammenfassung und Ausblick

4.1 Zusammenfassung

Es wurde gezeigt, dass ein handelsüblicher Wecker verschiedene Probleme aufweist, die durch Automatisierung und Context Aware Computing gelöst werden können. Das erste Problem ist das Stellen des Weckers auf die richtige Zeit. Diese Aufgabe kann der Wecker selbstständig erfüllen, wenn dem Wecker Wissen über die Zeitplanung des Bewohners mitgegeben wird. Das zweite Problem ist zu bestimmen, wann der genaue Zeitpunkt ist. Dieser Zeitpunkt kann dynamisch angepasst werden, wenn dem Wecker Wissen über den Kontext mitgeteilt wird. Dieses Wissen sollte dem Wecker durch eine lose Kopplung übertragen werden, damit die einzelnen Komponenten erweiterbar und austauschbar sind. Für diese Aufgabe eignet sich die Blackboard-Architektur.

4.2 Ausblick

Ich habe vor diesen Wecker umzusetzen. Der Wecker soll über die Termine des Informiert werden und die Alarmzeit berechnen. Bei der Berechnung sollen zum einen die Informationen über den Bewohner einfließen und zum anderen sollen Informationen aus externen Quellen eingeholt werden. Bei der Umsetzung habe ich vor mich an der Arbeit „Agentenbasierte Workflow Engine für eine intelligente Wohnung“ [8] zu orientieren, da meine Ideen für den Wecker einen Anwendung dieser Engine zu sein scheint.

Literaturverzeichnis

[1] ABOWD, G. D. ; DEY, A. K. ; BROWN, P. J. ; DAVIES, N. ; SMITH, M. ; STEGGLES, P. : Towards a Better Understanding of Context and Context-Awareness. In: *HUC '99: Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing*. London, UK : Springer-Verlag, 1999. – ISBN 3–540–66550–1, S. 304–307

[2] GREENBERG, S. : Context as a dynamic construct. In: *Hum.-Comput. Interact.* 16 (2001), Nr. 2, S. 257–268. http://dx.doi.org/http://dx.doi.org/10.1207/S15327051HCI16234_09. – DOI http://dx.doi.org/10.1207/S15327051HCI16234_09. – ISSN 0737–0024

[3] GREGOR ; RAHIMI ; VOGT ; SCHULZ ; V.LUCK: Tangible Computing revisited: Anfassbare Computer in Intelligenten Umgebungen. In: *Ambient Assisted Living Knongress* (2009). – http://users.informatik.haw-hamburg.de/~ubicomp/arbeiten/papers/MMWismar2009.pdf

[4] JOHANSON, B. ; FOX, O. : The Event Heap: A Coordination Infrastructure for Interactive Workspaces, 2002, S. 83–93

[5] JOSEPH, M. ; SERAFINI, L. ; TAMILIN, A. : Context shifting for effective search over large knowledge bases. In: *CIAO '09: Proceedings of the 1st Workshop on Context, Information and Ontologies*. New York, NY, USA : ACM, 2009. – ISBN 978–1–60558–528–4, S. 1–9

[6] KLAUSER, K. ; WALKER, V. : It's about time: an affective and desirable alarm clock. In: *DPPI '07: Proceedings of the 2007 conference on Designing pleasurable products and interfaces*. New York, NY, USA : ACM, 2007. – ISBN 978–1–59593–942–5, S. 407–420

[7] STEGELMEIER, S. : iFlat - Eine dienstorientierte Architektur für intelligente Räume. In: *Ambient Assisted Living Knongress* (2009). – http://users.informatik.haw-hamburg.de/~ubicomp/arbeiten/papers/aal2009.pdf

[8] TENNSTEDT, S. : Agentenbasierte Workflow Engine für eine intelligente Wohnung. (2010)

[9] WEISER, M. : The computer for the 21st century. In: *SIGMOBILE Mob. Comput. Commun. Rev.* 3 (1999), Nr. 3, S. 3–11. http://dx.doi.org/http://doi.acm.org/10.1145/329124.329126. – DOI http://doi.acm.org/10.1145/329124.329126. – ISSN 1559–1662

[10] YE, J. ; MCKEEVER, S. ; COYLE, L. ; NEELY, S. ; DOBSON, S. : Resolving uncertainty in context integration and abstraction: context integration and abstraction. In: *ICPS '08: Proceedings of the 5th international conference on Pervasive services*. New York, NY, USA : ACM, 2008. – ISBN 978–1–60558–135–4, S. 131–140

Glossar

Bewohner
: Der Bewohner wird hier synonym mit dem Nutzer oder Anwender des Weckers verwendet

Blackboard Architektur
: In dieser Architektur gibt es eine zentrale Einheit, die Nachrichten aufnimmt und diese allen zur Verfügung stellt, die sich für diese Nachricht interessieren.

Eventheap
: Hier ist der iRos-Eventheap [4] gemeint, der die Blackboard Architektur umsetzt, indem der alle eingehenden Nachrichten als Events interpretiert und diese an angemeldete Systeme verteilt, die sich für die Nachricht interessieren.

iFlat
: iFlat steht für ein Intelligentes Haus, das den Bewohner unterstützen soll

Bildquellen

Alle Bilder kommen von MS Office Cliparts.
http://office.microsoft.com/de-de/clipart/default.aspx