
Kontextmodellierung für das Ambient Assisted Living

Dissertation

zur Erlangung des Grades eines

D o k t o r s d e r N a t u r w i s s e n s c h a f t e n

der Universität Dortmund
am Fachbereich Informatik
von

Manfred Wojciechowski

Dortmund

2010

Tag der mündlichen Prüfung:	05. Mai 2011
Dekan/Dekanin:	**Prof. Dr. Peter Buchholz**
Gutachter/Gutachterinnen:	Prof. Dr. Jakob Rehof
	Prof. Dr. Dietmar Jannach

Fraunhofer ISST

Kontextmodellierung für das Ambient Assisted Living

Dissertation

Manfred Wojciechowski

Dortmund, den 23. August 2010
Version 1.1

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Der demographische Wandel in den entwickelten Industriestaaten ist eine der großen gesellschaftlichen Herausforderungen der Gegenwart. Schätzungen des statistischen Bundesamtes der Bundesrepublik Deutschland besagen, dass im Jahr 2050 die Anzahl der in Deutschland lebenden 60jährigen Personen doppelt so hoch sein wird wie die der Neugeborenen [1]. Die dadurch steigenden Kosten zur Betreuung und medizinischen Versorgung der älter werdenden Bevölkerung müssen von einer kleiner werdenden Basis getragen werden. Dem sich abzeichnenden Kostendruck und der damit verbundenen Belastung der Gesellschaft muss entgegengewirkt werden. Ein Ansatz hierzu ist das Bestreben, dem Menschen mit Hilfe technischer und medizinischer Mittel ein möglichst langes selbständiges Leben in seiner vertrauten Umgebung zu ermöglichen.

Als ein wichtiges europäisches Forschungsthema wurde „Ambient Assisted Living“ (AAL) im Rahmen des FP6-Förderprogramms definiert. Der Fokus hier ist die Nutzung moderner Informations- und Kommunikationstechnologien zur Unterstützung der älteren Personen und ihrer Familien. Basierend auf den Ergebnissen wurde ein Folgeprogramm definiert, in welchem ab dem Jahr 2008 europäische Forschung auf breiter Basis im Rahmen des Artikels 169 des europäischen Abkommens gefördert werden soll [2].

Ein wesentliches Merkmal von Lösungen im AAL ist das proaktive Verhalten. Die Hilfestellung der eingesetzten Technologien muss den Menschen unterstützen, ohne ihn hinsichtlich der notwendigen technologischen Kompetenz zu überfordern. Ein Baustein hierfür ist die Nutzung von Sensorik zur Erkennung von Situationen, auf die eine technische Lösung unterstützend reagieren muss. Die Erkennung von Ereignissen in der natürlichen Umwelt und die proaktive Reaktion von Anwendungen diesbezüglich wird im Forschungsfeld „Context Awareness“ betrachtet. Die Erkennung und Modellierung von Kontextinformationen ist dabei ein wichtiger Bestandteil der in diesem Feld durchgeführten Forschungsaktivitäten. Die Ergebnisse erlauben eine Integration der Technologie in die natürliche Umgebung des Menschen und ermöglichen dadurch eine möglichst hürdenfreie Unterstützung im Sinne des AAL.

Diese Dissertation beschäftigt sich mit dem Aspekt der Kontextmodellierung unter Berücksichtigung der besonderen Bedingungen die sich aus der Realisierung von technischen Lösungen im AAL ergeben. Ihre Zielsetzung und die sie tragende Motivation werden zunächst detailliert erläutert. Im Anschluss darin wird die wissenschaftliche Herangehensweise beschrieben und ihre Gliederung begründet.

1.1 Zielsetzung

Zielsetzung der Dissertation ist die Schaffung einer Methodik zur Modellierung und Nutzung von Kontextinformationen im AAL. Damit soll ein Baustein für die Realisierung kontextadaptiver AAL-Dienste zur Unterstützung des Menschen im Sinne des AAL geschaffen werden.

Zu diesem Zweck sollen zunächst die spezifischen Anforderungen an die Kontextmodellierung aus Sicht des AAL sowie geeignete Ansätze und Konzepte erarbeitet werden. Ein besonderer Anspruch an die Dissertation ist hierbei die umfassende Betrachtung der Kontextmodellierung. Im Gegensatz zu bisherigen Ansätzen soll die Arbeit nicht auf Einzelaspekte, wie z.B. der Implementierung einer einzelnen kontextadaptiven Anwendung, beschränkt sein. Vielmehr soll eine umfassende Modellierungsmethodik verfügbar gemacht werden, welche für den Aufbau einer entsprechenden Infrastruktur in der häuslichen Umgebung sowie für die Realisierung und die Nutzung kontextadaptiver Dienste notwendig ist. Konkret sollen aus Sicht der Kontextmodellierung folgende Fragen beantwortet werden:

- Welche Bestandteile der häuslichen Infrastruktur sind für die Realisierung kontextadaptiver Lösungen im AAL relevant und wie werden sie als Kontextinformationen beschrieben sowie zur Verfügung gestellt?

- Wie kann die Sensorik als Bestandteil einer solchen Infrastruktur beschrieben werden und wie ist es möglich, diese entsprechend zu erweitern?

- Wie können kontextadaptive Lösungen im AAL realisiert werden? Wie erfolgt hierbei insbesondere die Realisierung der kontextspezifischen Logik und welche Methodik sollte dabei angewendet werden?

- Wie erfolgen die Aufnahme und der Betrieb einer entwickelten kontextadaptiven Lösung in der spezifischen häuslichen Umgebung eines Bewohners und wie kann hierbei die jeweils unterschiedliche Ausstattung einer solchen Umgebung berücksichtigt werden?

- Was bedeutet die Berücksichtigung des Bewohners für die Umsetzung des kontextadaptiven Verhaltens einer technischen Lösung?

Diese und weitere im Rahmen des AAL relevante Fragestellungen sollen in dieser Arbeit beantwortet werden. Sie beschreiben die notwendigen Konzepte zur Kontextmodellierung und bilden die Grundlage für Werkzeuge, mit denen ein Entwickler entsprechende kontextadaptive Lösungen realisieren sowie dem Bewohner innerhalb seiner intelligenten häuslichen Umgebung bereitstellen kann. Auch bilden sie die Grundlage für die Entwicklung und Instantiierung von Komponenten einer solchen Infrastruktur. Weiterhin sollen die herbei an-

zuwendenden Methoden definiert sowie die dafür benötigten Werkzeuge und Infrastrukturkomponenten identifiziert werden. Die Entwicklung und Bereitstellung der Softwareartefakte ist dagegen nicht im Fokus der Dissertation.

Die Anwendbarkeit der Ergebnisse soll durch eine prototypische Umsetzung wesentlicher Konzepte überprüft werden, wobei die im notwendigen Umfang realisierten Infrastrukturkomponenten zum Einsatz kommen sollen.

1.2 Einordnung in das wissenschaftliche Umfeld

Die Dissertation umfasst zwei Forschungsbereiche: „Ambient Assisted Living“ und „Context Awareness“. Obwohl beide inhaltliche Berührungspunkte haben, werden derzeit die jeweiligen Lösungsansätze kaum bereichsübergreifend gesehen.

Der Forschungsbereich „Ambient Assisted Living“ wird bisher mehr anwendungsbezogen betrachtet und multidisziplinär unter verschiedenen technischen, sozialen und wirtschaftlichen Fragestellungen zum Aufbau und zur Bereitstellung einer assistiven technischen Infrastruktur erörtert. Es werden konkrete technische Lösungen entwickelt und beispielsweise unter dem Aspekt der Usability in entsprechenden Laboren getestet. So lassen sich verschiedene Technologien und allgemeine Forschungsthemen in dieser Anwendungsdomäne facettenartig betrachten. Die Nutzung von Kontextinformationen gehört zum Aufbau notwendiger Lösungsbausteine, wozu diese Dissertation beitragen möchte.

Der Forschungsbereich „Context Awareness“ ist weitgehend theoriebezogen und wird deshalb vielfach unabhängig von einer konkreten Anwendungsdomäne betrachtet. Hier geht es um Erkennung, Ableitung und Abbildung von Informationen in ein Kontextmodell. Weitere Fragestellungen sind die effiziente Realisierung von kontextadaptiven Anwendungen und die Bereitstellung von Kontextinformationen beispielsweise unter dem Gesichtspunkt der Privatsphäre. Diese Arbeit ergänzt die Betrachtungen um spezielle Anforderungen und Lösungsansätze, die sich aus der Anwendungsdomäne „Ambient Assisted Living“ ergeben.

1.3 Motivation

Die Motivation für die Dissertation lässt sich sowohl aus der Anwendungsdomäne „AAL“ als auch aus dem Forschungsbereich „Context Awareness“ ableiten. Der Bedarf an technischen Lösungen zur Bewältigung des Kostendrucks, der aus der demographischen Entwicklung erwächst, wird in Zukunft weiter anwachsen und kann einen immensen Markt eröffnen [3]. Solange jedoch keine standardisierte Infrastruktur existiert, über die ein arbeitsteiliger Aufbau von

intelligenten Lösungen möglich ist, werden lediglich teure und proprietäre Insellösungen für kleine Zielgruppen realisierbar sein. Ein Schritt in eine solche Standardisierung ist die Definition von Vorgaben und Methoden zum Aufbau einer Kontextinfrastruktur und zur Umsetzung des kontextadaptiven Verhaltens einer technischen Lösung. Dadurch soll die Realisierung von konkreten technischen Lösungen vereinfacht werden, um so einen solchen Markt entstehen zu lassen, in dem spezielle Produkte von einer Vielzahl von Anbietern gehandelt werden, die die aufgebaute Infrastruktur der häuslichen Umgebung eines Bewohners nutzen können. Hierfür soll diese Arbeit ebenfalls Grundlagen schaffen.

Im Forschungsbereich „Context Awareness“ ist die Diskussion um die Spezifikation von Kontextmodellen und deren technische Realisierung sowie die Anwendung dieser Modelle noch im Fluss. Die Kontextmodellierung ist ein wichtiger Baustein zur Realisierung kontextadaptiver Anwendungen. Kontextmodelle ermöglichen eine formale Beschreibung der abzubildenden Umwelt. Es gibt bereits eine Reihe von Ansätzen zur Modellierung von Kontextinformationen, von denen sich jedoch noch keiner durchsetzen konnte. Einige davon sind auf bestimmte Anwendungsdomänen oder Einsatzzwecke spezialisiert. In der Forschung werden derzeit noch vielfältige Diskussionen über geeignete Modellierungsansätze geführt [4] [5]. Es gibt noch keine anwendungsübergreifenden Standards für Kontextmodelle, die auch unmittelbar für das AAL genutzt werden können. Aus der Anwendungsdomäne AAL können Anforderungen an Kontextmodelle erhoben werden, die in der aktuellen Diskussion über Kontextmodelle noch nicht im Fokus sind – insbesondere solche aus dem Ubiquitous Computing und dem End User Modeling. Die Dissertation will Ergebnisse aus dem Forschungsbereich „Context Awareness“ in die Anwendungsdomäne „AAL“ überführen und konkretisieren. In der Konkretisierung lassen sich auch Rückschlüsse auf die generellen Konzepte und Verfahren ziehen, die dann in die weiterführende Diskussion einfließen können.

Das Fraunhofer ISST erarbeitet und erprobt mit dem Anwendungsfeld „Smart Living“ technische Lösungen im AAL. Auf der Basis einer informationslogistischen Diensteplattform werden intelligente Dienste im Bereich Gesundheit, Sicherheit, Komfort und Entertainment sowie Facility Management realisiert. Teilaufgaben dieser informationslogistischen Plattform sind die Erkennung und Bereitstellung von Kontextinformationen über ein Kontext-Subsystem das noch sehr einfach und auf bestimmte Anwendungsszenarien hin ausgerichtet ist. Für die Entwicklung dieser kontextadaptiven Dienste existiert derzeit noch keine Methodik, woraus praktische Probleme resultieren, die mit dieser Arbeit konzeptuell überwunden werden sollen. Ein solches Problem ist beispielsweise die fehlende Dokumentation der kontextadaptiven Funktionen bereits realisierter AAL-Dienste. Das Wissen über das kontextadaptive Verhalten und dessen Realisierung ist ausschließlich in den Köpfen der Entwickler vorhanden. Bei der Realisierung von Diensten mit ähnlichem kontextadaptiven Verhalten erfolgt die Umsetzung der Kontextlogik mehrfach, was zur eingeschränkter

Wiederverwendbarkeit von Logik führt. Diese und weitere Probleme zwingen zu methodischem Vorgehen. Wenn sich die Konzepte dieser Arbeit bewähren, sollen sie in das Kontext-Subsystem des Fraunhofer ISST einfließen und damit die informationslogistische Diensteplattform erweitern und sie im Rahmen von AAL, aber auch anderen Anwendungsdomänen flexibler einsetzbar machen.

1.4 Methodik

Die Erarbeitung der Ergebnisse erfolgte in 4 aufeinander aufbauenden Schritten:

1. Identifikation und Analyse der Anforderungen an eine Kontextmodellierung im AAL

   Zunächst wurden die relevanten Forschungsbereiche identifiziert, die mit der Aufgabenstellung verbunden waren. In ihnen wurden anschließend die bestehenden Arbeiten sowie die ableitbaren Anforderungen mit Hilfe einer Literaturrecherche identifiziert. Ihre Ergebnisse sind in der Einführung der Grundlagen (Kap. 2) und der Identifikation der Anforderungen (Kap. 3) dokumentiert.

2. Bewertung bestehender Lösungsansätze

   Die identifizierten Anforderungen müssen durch geeignete Ansätze der Kontextmodellierung umgesetzt werden können. Eine Reihe davon existiert bereits in der Forschung im Bereich „Context Awareness“. Diese werden analysiert und unter Berücksichtigung der identifizierten Anforderungen auf ihre Nutzbarkeit hin bewertet. Auch sind die Lücken festzustellen, die von den bestehenden Ansätzen nicht abgedeckt werden können.

   Die Identifikation und Analyse bestehender Ansätze erfolgt mit Hilfe der Literaturrecherche, deren Auswertung im Kapitel 4.1 erfolgt.

3. Erarbeitung von Lösungsbausteinen und ihre Zusammenführung zu einer Gesamtlösung der Kontextmodellierung

   Sind Lücken identifiziert, so werden dafür eigene Ansätze der Kontextmodellierung erarbeitet, die den bestehenden Anforderungen entsprechen. Diese können durch Erweiterung vorhandener Lösungen oder durch Entwicklung neuer erfolgen. Zunächst wird ein Grobkonzept für die Kontextmodellierung erarbeitet, welches die Grundlagen dafür identifiziert. Es ist in Kapitel 4 dargestellt. Anschließend erfolgt eine Verfeinerung dieses Grundgerüsts in den Kapiteln 5, 6 und 7. Die Anwendung der Lösungsbausteine wird anschließend in einer Methodik zusammengefasst, die im Kapitel 8 beschrieben wird.

4. Evaluierung der Gesamtlösung

   Die Bewertung der Gesamtlösung ist gestuft. Auf der Ebene der Lösungsbausteine wurde beispielsweise ein Anwendertest durchgeführt, bei welchem die Einsetzbarkeit des Interaktionskonzepts zur Konfiguration des kontextadaptiven Verhaltens durch den Benutzer getestet wurde (Kap. 7.3.3.3). Diese kann aufgrund der Rahmenbedingungen der Testdurchführung nur initiale Indikatoren für dessen Anwendbarkeit liefern. Eine weitere Evaluation erfolgte durch die Umsetzung wesentlicher Konzepte der Kontextmodellierung in Werkzeugen und Komponenten eines Context Servers (Kap. 8.1). Die Methodik der Kontextmodellierung wurde angewandt, um initiale Kontextmodelle zu definieren und im Context Server umzusetzen. Diese Methodik wurde zusätzlich zwei unterschiedlichen Praxistests unterzogen. Zum einen wurde das kontextadaptive Verhalten bereits am ISST bestehender kontextadaptive AAL-Dienste in einem Reverse Engineering mit Hilfe der Methodik dokumentiert. Dieses konnte von einem Entwickler, der kurz in seine Aufgabe eingeführt wurde, selbständig durchgeführt werden. Zum anderen wurde für einen einfachen AAL-Dienst das Kontextmodell methodisch definiert und im Context Server umgesetzt, wobei reale Sensoren eingebunden wurden. Das gewünschte kontextadaptive Verhalten des AAL-Dienstes konnte auf dieser Basis umgesetzt werden. Schließlich wurden Teilergebnisse wissenschaftlich publiziert [6] [7] [8] [9].

1.5 Überblick

Die Arbeit ist aus folgenden Kapiteln aufgebaut:

Kapitel 2 gibt einen Überblick über die wesentlichen Themengebiete der Dissertation, definiert wesentliche Begriffe und stellt den aktuellen Stand der Forschung dar. Damit wird die Grundlage geschaffen, um den Zusammenhang und die inhaltlichen Aussagen der Dissertation zu verstehen. Wer sich in den Themengebieten AAL und Context Awareness auskennt, kann dieses Kapitel überspringen.

Kapitel 3 identifiziert die wesentlichen Anforderungen an die Kontextmodellierung aus Sicht der AAL.

Kapitel 4 bewertet die aktuellen Ansätze der Kontextmodellierung anhand der Anforderungen. Anschließend erfolgt die Erarbeitung eines Grundgerüsts für die Kontextmodellierung im AAL. Dabei werden zwei Dimensionen identifiziert, die für die gezielte Betrachtung der jeweiligen Anforderungen und Erarbeitung der Modellierungskonzepte notwendig sind und in einer Übersicht als zweidimensionales Kontextmodell vorgestellt.

Kapitel 5 vertieft das zweidimensionale Kontextmodell auf der Ebene der Infrastruktur. Es werden die Anforderungen, Konzepte, das Metamodell, sowie die unterschiedlichen Kontextmodelle definiert.

Kapitel 6 erörtert das zweidimensionale Kontextmodell auf der Ebene der Dienste. Es werden die Anforderungen, Konzepte, das Metamodell, die Übergänge zur Ebene der Infrastruktur, sowie die unterschiedlichen Kontextmodelle definiert.

Kapitel 7 erarbeitet das zweidimensionale Kontextmodell auf der Ebene der Nutzerinteraktion. Es werden die Anforderungen, Konzepte, das Metamodell, die Übergänge zur Ebene der Dienste, sowie die unterschiedlichen Kontextmodelle definiert.

Kapitel 8 stellt die Methodik zur Realisierung von AAL-Diensten vor. Diese basiert auf dem zweidimensionalen Kontextmodell, sowie den darin zu verwendenden Werkzeugen, die ebenfalls beschrieben werden. Zusätzlich werden Einsatzphasen der Methodik definiert sowie die Anwendung der Methodik am Beispiel eines konkreten AAL-Dienstes demonstriert.

Kapitel 9 fasst die erreichten Ergebnisse zusammen und unterzieht sie einer abschließenden Betrachtung und Bewertung. Es endet mit einem Ausblick auf weitere Arbeiten, die auf diesen Ergebnissen aufbauen können.

2 Grundlagen

Mit diesem Kapitel sollen die Grundlagen geschaffen werden, um den Zusammenhang und die Inhalte der Arbeit besser zu verstehen. Deshalb werden nachfolgend die wichtigsten Begriffe erklärt sowie die in diesem Zusammenhang relevanten Arbeiten und der aktuelle Stand der Entwicklung erläutert. Die Klärung erfolgt für die beiden Forschungsbereiche „Ambient Assisted Living“ und „Context Awareness“.

2.1 Ambient Assisted Living

Der Begriff „Ambient Assisted Living“ wurde ursprünglich von der Europäischen Union im Rahmen des FP6 Forschungsförderprogramms zur Vorbereitung der Art. 169-Initiative zur Unterstützung von älteren Menschen geprägt. Mittlerweile nutzt ihn eine Reihe von europäischen Forschungsinstitutionen zur Beschreibung von Technologien, die es vor allem älteren Menschen ermöglichen, ein lange selbst bestimmtes Leben in den eigenen vier Wänden zu führen. Auch außerhalb von Europa findet er in der Forschung verstärkt Verwendung. So gibt es beispielsweise in den Vereinigten Staaten von Amerika einen Arbeitskreis „Smart Homes and Ambient Assisted Living“ der „International Medical Informatics Association“ (IMIA) [10].

Die Unterstützung der älteren Menschen im Rahmen ihrer eigenen vier Wände basiert im AAL auf dem Aufbau einer intelligenten Umgebung, welche sich in den menschlichen Wohnraum integriert und in der Lage ist, mit diesen zu interagieren. Somit verschmelzen die reale und virtuelle Umgebung und ermöglichen den Aufbau von intelligenten Umgebungen. Die dafür notwendigen Technologien lassen sich dem Forschungsgebiet „Home Automation“ und „Ambient Intelligence“ zuordnen. Im ersten wird die Basis für die Vernetzung von Aktorik und Sensorik geschaffen. Über Kommunikationsmedien und Protokolle können unterschiedlichste Geräte miteinander vernetzt werden, z.B. Haushaltsgeräte, Unterhaltungsmedien oder die Haustechnik. Über eine intelligente Steuerung kann das Zusammenspiel dieser Komponenten geregelt werden. Technologien aus dem zweiten, dem Ambient Intelligence, ermöglichen eine möglichst natürliche Interaktion des Nutzers mit der intelligenten Umgebung. Hierzu gehören beispielsweise multimodale Nutzerinteraktionen oder die Kontextadaptivität. Auf dieser technischen Basis können im Rahmen des AAL intelligente Dienste entwickelt und in die Wohnumgebung des Nutzers integriert werden. Diese können ihm dann Hilfestellung anbieten, die mental kaum oder garnicht mehr wahrzunehmen sind. Solche Dienste sind in unterschiedliche Anwendungsgebiete unterteilt, wie „Gesundheit“, „Sicherheit“, „Komfort“, „Soziales Umfeld“ und „Wirtschaftlichkeit“. Neben diesen techni-

schen Themen werden noch weitere Aspekte betrachtet. Dazu gehört beispielsweise die Wirtschaftlichkeit von Diensten im AAL. Hier müssen tragfähige Geschäftsmodelle sowie mögliche Rollen und Dienstleister in einem Markt identifiziert werden. Eine Umsetzung durch entsprechende Geschäftsprozesse ist ebenfalls Gegenstand der Betrachtung. Diese Bausteine lassen sich durch weitere Lösungen aus anderen technischen Domänen ergänzen, zu denen beispielsweise das „Universal Design“ gehört. Dieses befasst sich mit der Ausformung von Alltagsgegenständen in einer Weise, wie sie von älteren oder behinderten Personen einfacher genutzt werden kann. Ein Beispiel hierfür wäre die Badewanne mit Einstiegstür. Ein weiterer technischer Aspekt ist die Gestaltung und Ausstattung einer Wohnung selbst, die möglichst barrierefrei zugänglich und auf die Bedürfnisse der älteren Menschen hin abgestimmt sein sollte.

Die Vielfalt der notwendigen Lösungsbausteine im AAL stellt die folgende Abbildung dar.

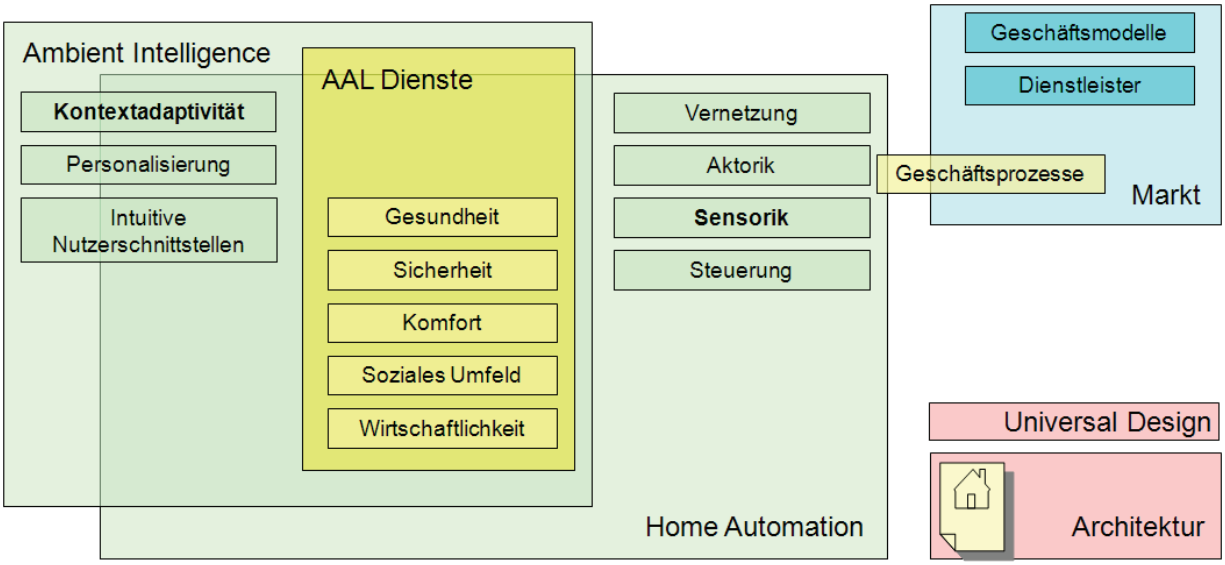


Abbildung 1: Technologien im Ambient Assisted Living

Da das Thema AAL als angewandtes Forschungsthema auf bestehenden Forschungsfeldern aufsetzt, sind dort bereits grundsätzliche Ergebnisse vorhanden. Die spezielle Ausrichtung der Technologien auf die Bedürfnisse einer alternden Gesellschaft sowie der Gesichtspunkt der Kostenoptimierung ist jedoch neu und deshalb noch wenig wissenschaftlich bearbeitet. Einige Besonderheiten und Herausforderungen dieser Thematik werden in [11] aufgeführt. Ein erster wissenschaftlicher Workshop, der auf das Thema AAL spezialisiert war, wurde im November 2007 im Rahmen der Europäischen Konferenz für Ambient Intelligence in Darmstadt veranstaltet [12].

Nachfolgend werden einige der Themen im AAL detaillierter beschrieben.

2.1.1 Ubiquitous Computing und Ambient Intelligence

Bereits in den späten 80er Jahren wurde am Xerox PARC daran geforscht den Computer in die natürliche Umwelt des Menschen zu integrieren und den Menschen direkt mit den computerisierten Objekten in seiner Umgebung interagieren zu lassen. Diese Forschungsarbeiten wurden von Mark Weiser initiiert und folgten der Idee, den Computer aus dem Sichtfeld des Nutzers verschwinden oder für sie unsichtbar werden zu lassen [13]. Der Nutzer sollte nicht durch eine ihn mental beanspruchende Mensch-Maschine-Schnittstelle abgelenkt werden, sondern vielmehr in die Lage versetzt werden, auf natürliche oder entspannte Weise mit einer Umgebung zu kommunizieren, die ihn in seinen Zielen unterstützt. Mark Weiser hat mit diesem Ziel auch den Begriff des Ubiquitous Computing geprägt. Die Interaktion des Nutzers mit dieser technisch ausgestatteten Umgebung sollte sich an seine natürliche Sprache, Bewegung und Gestik anpassen [14].

Nach den Ideen und Ergebnissen des Ubiquitous Computing beschreibt das Ambient Intelligence den Aufbau einer intelligenten Umgebung, die Personen in ihrem Umfeld erkennt und sie in ihren täglichen Aktivitäten, Aufgaben und Ritualen unterstützt. Der Mensch soll mit dieser Umgebung auf eine möglichst natürliche Weise interagieren. Die Intelligenz zur Erkennung der Ziele und Wünsche der Nutzer sowie zur Durchführung der sie unterstützenden Dienste erfolgt durch das koordinierte Zusammenspiel von unterschiedlichsten Geräten mit spezialisierten Aufgabenstellungen. Diese Geräte sind über ein Netzwerk miteinander verbunden über das sie kommunizieren. Die Intelligenz der Umgebung ist in diesem Netzwerk verborgen (Internet of Things). Mit der Miniaturisierung der Geräte verschwinden diese immer mehr aus dem menschlichen Blickfeld, bis nur noch die Kommunikation mit dieser Intelligenz für den Nutzer fassbar ist. Auch sind einzelne Funktionen nicht mehr einem festen Gerät innerhalb des Netzwerks zuzuordnen, sondern bewegen sich flexibel innerhalb des dynamischen Netzwerks.

Nach Oppermann [15] zeichnen sich Ambient Intelligence Systeme durch folgende Eigenschaften aus:

- Sie sind unsichtbar, z.B. eingebettet in Kleidung, Uhren oder Brillen.
- Sie sind mobil und können vom Nutzer mit sich getragen werden.
- Die Kommunikation zwischen Knoten in einem Netzwerk ist spontan.
- Die Knoten in einem Netzwerk sind heterogen und hierarchisch. So können verschiedene Arten von Knoten im Netzwerk existieren, welche

sich in ihrer Leistungsfähigkeit, Funktionalität und Verfügbarkeit unterscheiden.

- Sie sind kontextsensitiv. So können sie auf Grund des Wissens über die lokale Umgebung spontan Informationen mit ähnlichen Knoten in ihrer Umgebung austauschen.

- Sie sind antizipativ, d.h. sie agieren selbständig ohne explizite externe Anfragen.

- Sie ermöglichen eine natürliche Kommunikation mit dem Nutzer mittels Sprache und Gestik, anstatt einer Tastatur, Maus oder Text auf einem Bildschirm.

- Sie ermöglichen eine natürliche Interaktion mit dem Nutzer über Geräte, an die dieser bereits gewöhnt ist, z.B. Kleidung, Uhren, Fernsehen, Telefon und Haushaltsgeräte. Diese Geräte werden mit eigener Intelligenz ausgestattet.

- Sie sind adaptiv, d.h. sie sind in der Lage, auf abnormale und außergewöhnliche Situationen in einer flexiblen Weise zu reagieren.

Vorreiter der Begriffsbildung des „Ambient Intelligence“ in den späten 90er Jahren war Philips. In einer Reihe von internen Workshops wurden Wege und Visionen zur Entwicklung von nutzerfreundlichen Geräten erarbeitet, die dem Anwender auf natürliche Weise Informationen, Kommunikation und Unterhaltung bereitstellen sollen. Im April 2002 wurde durch Philips das HomeLab [16] errichtet, in dem eine Reihe von Technologien erarbeitet und ausprobiert werden können. Das Thema „Ambient Intelligence“ wurde als ein Themengebiet im FP6 Förderprogramm der Europäischen Union aufgenommen. Auf dieser Grundlage wurden Forschungsinitiativen innerhalb der Europäischen Union, aber bald auch weltweit gestartet. Mittlerweile werden die Begriffe „Ubiquitous Computing“ und „Ambient Intelligence“ weitestgehend synonym verwendet, wobei ersterer mehr im amerikanischen und asiatischen Sprachraum und letzterer vorwiegend in Europa verwendet wird.

An unterschiedlichen Fraunhofer Instituten wurden im Rahmen des FP6 AAL-Aktivitäten z.B. im Bereich Multimedia, Design von Mikrosystemen und Augmented Spaces gestartet. Weitere Beispiele für Forschungsaktivitäten sind die Carnegie Mellon Universität mit dem Ambient Intelligence Lab [17], die Kingston University London mit der Ambient Intelligence Research Group [18], das MIT Media Lab [19], NTT Research [20] oder die University of Reading [21].

Zusammenfassend ist festzustellen, dass es aufgrund der technologischen Vielfalt im Bereich des Ambient Intelligence eine Vielzahl unterschiedlichster Forschungsaktivitäten gibt. Ein Schwerpunkt der Forschung liegt sicher auf der

Entwicklung einer flexiblen und dynamischen Diensteinfrastruktur. Die Realisierung von Funktionen auf unterschiedlichen Netzknoten und der Übergang von festen Rechnersystemen zu einem Netz von unterschiedlichen Rechnersystemen mit diversen Eigenschaften und Verfügbarkeiten macht die Entwicklung von Anwendungen sehr aufwändig. Hier gibt es erste Ansätze zur Vereinfachung mittels einer Middleware [22]. Wie aus der Aufzählung von Oppermann ersichtlich, ist „Context Awareness“ ein Baustein in diesem Themenfeld.

2.1.2 Home Automation

Home Automation ist ein Spezialgebiet der Building Automation und auf die spezifischen Anforderungen des privaten Umfelds ausgerichtet. Eine technologische Basis ist die Vernetzung von Komponenten über ein Bussystem. Bereits aus der industriellen Automatisierung existieren Standards, die auch im privaten Bereich eingesetzt werden können. Diese basieren zumeist auf kabelgebundenen Protokollen, welche für einen nachträglichen Einbau in bestehenden Wohngebäuden mit nicht unerheblichen Kosten verbunden sind. Daher werden hier Bussysteme angeboten, die kabellos über Funkmedien eine Vernetzung ermöglichen oder bestehende Leitungen der Stromversorgung nutzen. Beispiele für solche Standards sind EIB [23], X10 [24], LonWorks [25], Universal Powerline Bus (UPB) [26], UPnP [27], ZigBee [28] und Z-Wave [29]. Auf diesen Bussystemen können die Komponenten der Home Automation aufgesetzt werden. In der Regel lassen sich folgende Komponenten unterscheiden:

- Sensoren: Diese Komponenten sind in der Lage, Zustände und Ereignisse innerhalb der Wohnumgebung zu erkennen und zu melden. Es können beispielsweise Sensoren zur Erkennung von Einbruch, Rauch oder Leckagen sein von denen bereits vielfältige Systeme existieren.

- Aktoren: Sie sind in der Lage, reale Gegenstände innerhalb der Wohnumgebung zu steuern und entsprechend einzugreifen. Es können beispielsweise Aktoren zum Öffnen und Schließen von Türen und Fenstern sein oder zum Steuern des Lichtes sowie der Heizung. Auch hier existiert bereits eine Vielzahl von Komponenten.

- Controller: Er ist in der Lage, auf gemeldete Ereignisse der Sensoren hin zu reagieren und beispielsweise geeignete Aktoren zu aktivieren. Der Controller ist hier die zentrale Einheit, die im Zusammenspiel mit den Sensoren und Aktoren die Automatisierung realisiert. Es existieren eine Reihe proprietärer Controller unterschiedlichster Anbieter. Viele von ihnen sind auf spezielle Sensoren und Aktoren hin beschränkt. Eine bedeutende Technologie im Home Automation ist die in der OSGi Alliance [30] (früher „Open Services Gateway initiative“) spezifizierte OSGi-Plattform. Diese bildet eine hardwareunabhängige Softwareplattform

auf der Basis von JAVA, welche zur Vernetzung und Steuerung aller Art von Geräten genutzt werden kann. Ein Einsatzgebiet ist die Steuerung von Komponenten und Bereitstellung von Diensten in Automobilen. Ein weiteres ist die Vernetzung von Aktorik und Sensorik im Home Automation.

Die im Home Automation verfügbaren Funktionen sind im Wesentlichen auch im industriellen Building Automation zu finden. Diese sind zumeist klar definierte Aufgabenstellungen im Bereich der Licht- und Klimasteuerung sowie der Sicherheit. Hinzu kommen aber auch Funktionen, die weitestgehend dem privaten Wohnerlebnis zuzuordnen sind, wie zum Beispiel die Steuerung eines multimedialen Home Entertainment Sytems. Konkrete im Home Automation verfügbare Funktionen sind [31]:

- Steuerung der Heizung, Luftzufuhr und Klimatechnik. Dieses beinhaltet auch die Überwachung und Steuerung der Temperatur sowie der Luftfeuchtigkeit und kann auch unter dem Gesichtspunkt der Ökonomie erfolgen.

- Steuerung der Beleuchtung zu unterschiedlichen Zwecken. Beispielsweise kann zur Einbruchsprävention ein Lichtszenario geschaltet werden, welches bei einbrechender Dunkelheit die Anwesenheit von Personen simuliert. Weiterhin kann das Abschalten aller Lampen bei Verlassen des Hauses zentral veranlasst und die Beleuchtung der Helligkeit der Umgebung angepasst werden.

- Steuerung des natürlichen Lichteinfalls durch Steuerung der Rolläden, der Gardinen oder der Lichtdurchlässigkeit des Fensters.

- Zentrale Steuerung der Audio- und Videoquelle. Auf diese Weise können Audio und Video in unterschiedlichen Räumen konsumiert werden. Diese Funktion wird gelegentlich als „multi-zone audio“ bzw. „multi-zone video“ bezeichnet.

- Integration und Kontrolle von Sicherheitssystemen. Dazu gehört die Erkennung und Meldung von Einbrüchen, Leckagen oder Brand. Ebenso können die Simulation von Anwesenheit erfolgen und medizinische Notfälle gemeldet werden.

- Hausweite Kommunikation ermöglicht entweder einer Raum-zu-Raum Kommunikation oder auch die Schaltung einer hausweiten Durchsage.

Weitere Möglichkeiten sind die Steuerung der Garage, der Sprinkleranlage für die Bewässerung des Gartens oder der Kaffeemaschine. Es existieren Aktoren, die das Ein- und Ausschalten von normalen Hausgeräten über die Stromzufuhr ermöglichen. Somit sind der Kreativität wenig Schranken gesetzt. Die

bestehenden Lösungen im Home Automation zeichnen sich jedoch gegenwärtig dadurch aus, dass sie einzelne Insellösungen darstellen und jeweils klar abgegrenzte Funktionalitäten realisieren. Im privaten Bereich können bereits ambitionierte und technikerfahrene Personen komplexe Lösungen aufbauen oder selbst entwickeln. Anleitungen und Tipps lassen sich im Internet finden [32]. Auch existieren Foren zur Diskussion von Lösungen und Technologien [33].

Derzeitige Forschungsaktivitäten befassen sich mit dem Definieren von Standards sowohl in der Vernetzung als auch in der Bereitstellung einer Middleware für die Integration und Ausführung von Diensten. Einige Forschungsinstitute haben Demonstrationsumgebungen aufgebaut, in denen die Entwicklung und auch der Einsatz von neuen Technologien und Diensten erprobt werden kann. Beispiele finden sich im InHaus [34] der Fraunhofer Gesellschaft.

2.1.2.1 Sensorik

Die Sensorik ist ein wesentlicher Bestandteil der Infrastruktur einer intelligenten Umgebung. Sensoren liefern Kontextinformationen, die für die Kontextadaptivität notwendig sind. Es gibt bereits eine Vielzahl von ihnen zur Erfassung unterschiedlicher Informationen. In [35] wird eine Kategorisierung von Kontextsensoren vorgenommen, die im häuslichen Umfeld von Bedeutung sein können:

- Licht und Sicht: Über einfache optische Sensoren können Informationen zur Intensität, Dunkelheit, Reflektion und Farbtemperatur des Umgebungslichtes gesammelt werden. Spezielle Arten können die Wellenlänge oder Ausschnitte im Lichtspektrum messen. Aus dem Muster der Lichtänderung können Hinweise auf die Umgebung gewonnen werden, z.B. das 50Hz Flackern eines Fernsehers. Kombiniert man mehrere Lichtsensoren an einem Sensorknoten mit unterschiedlich räumlicher Ausrichtung, so kann die Lichtverteilung als Ausgangspunkt für die Ableitung weiterer Kontextinformationen, wie z.B. Bewegung genutzt werden. Verwendet man komplexere Sensoren, z.B. eine CMOS-Kamera in Verbindung mit einem Signalprozessor, so können je nach Komplexität der verwendeten Algorithmen weitere Kontextinformationen zur Umgebung abgeleitet werden, beispielsweise Bewegung, Farbverteilung, Erkennung von Objekten, Landschaften und Personen.

- Audio: Mit Hilfe von Mikrophonen können akustische Informationen der Umgebung erhoben werden wozu die Lautstärke und die Frequenz gehören. Unterschiedliche akustische Typen können registriert werden, z.B. Geräusch, Musik und Sprache. Mit Hilfe von komplexen Algorithmen kann aus dem Geräusch auf die Situation in der Umgebung geschlossen werden, wie z.B. Verkehrsgeräusche und Schreien von Kin-

dern. Auch die Art der Musik kann erkannt werden. Ebenfalls kann die Spracherkennung als Grundlage für die Ableitung von Kontextinformationen genutzt werden. Mit Hilfe von mehreren Mikrophonen kann zudem eine räumliche Ortung der akustischen Quelle erfolgen.

- Bewegung und Beschleunigung: Artefakte können mit Sensoren zum Messen der Bewegung und der Beschleunigung ausgestattet werden. Solche können beispielsweise Kleidungsstücke, Gehhilfen oder das Mobiltelefon sein. Hierüber können Kontextinformationen zum Nutzer dieser Artefakte abgeleitet werden, z.B. sportliche Aktivität oder Sturz.

- Ortung: Es gibt eine Vielzahl von Sensoren zur Ortung von Personen und Gegenständen. In [36] wird ein Überblick über unterschiedliche Ortungssysteme und deren Eigenschaften gegeben. Neben lokal in der häuslichen Umgebung vorhandenen Ortungstechnologien, z.B. mittels Radio Frequency Identification (RFID) [37] oder Infrarot-Ortung, sind auch von externen Dienstleistern angebotene Ortungsdienste wie der Mobilfunkprovider zu berücksichtigen. Die Ortung ist eine häufig verwendete Kontextinformation.

- Magnetfeld und Orientierung: Mit Hilfe von Sensoren, die auf das Magnetfeld der Erde reagieren, kann eine Richtungsbestimmung erfolgen, die dann die Ausrichtung eines Gerätes oder die Richtung einer Bewegung ermittelt.

- Nähe, Berührung und Nutzerinteraktion: Es gibt eine Reihe von Sensoren, mit denen die Entfernung von Gegenständen oder Personen zu einem Bezugspunkt gemessen werden kann. Diese kann beispielsweise mit Hilfe eines Lasers geschehen. Durch eine druckempfindliche Oberfläche kann die Berührung eines Gegenstands durch den Bewohner ermittelt werden. Diese kann auch durch die Verwendung eines spannungsempfindlichen Sensors erfolgen, der auch die Leitfähigkeit der Haut und die Muskelspannung ermitteln kann.

- Temperatur, Feuchtigkeit und Luftdruck: Zur Messung der Temperatur der Umgebung, eines Gegenstands oder einer Person existieren umfangreiche kostengünstige Sensoren. Feuchtigkeit und Luftdruck einer Umgebung können ebenfalls durch kostengünstige Sensoren festgestellt werden. Neben der aktiven Ermittlung dieser Umgebungsinformationen mittels Sensoren können auch im Internet vorhandene Wetterdienste als Informationsquellen für diese Kontextinformation berücksichtigt werden.

- Gewicht: Das Gewicht eines Gegenstands oder einer Person kann mit einfacher Sensorik gemessen werden. Neben dem absoluten Gewicht kann die Varianz der Messwerte interessante Kontextinformationen lie-

fern. So kann beispielsweise mit einer räumlich verteilten Anordnung von Gewichtssensoren die Bewegung von Personen verfolgt werden.

- Aktivität: Die Erkennung einer generellen Aktivität innerhalb eines Raums ist mit einfacher Sensorik möglich. Dieses geschieht beispielsweise mit Hilfe von „Passive Infrared Sensors“ (PIR) [38] die prüfen, ob sich innerhalb der Reichweite Personen bewegen.

- Gassensor und Geruch: Es gibt eine Reihe von spezialisierten Sensoren, mit denen bestimmte Gase, Alkohol oder Lebensmittel ermittelt werden können. Zur Erkennung von Gerüchen gibt es spezialisierte Geräte, die eine Reihe von Sensoren miteinander kombinieren. Diese werden derzeit hauptsächlich in der Lebensmittelindustrie verwendet.

- Bio-Sensor: Mit diesen können Vitalwerte von Personen ermittelt werden, wozu beispielsweise der Blutdruck, der Puls oder der Blutzucker gehören. Viele von ihnen sind bereits für den häuslichen Gebrauch zu erwerben.

Die beschriebenen Arten von Kontextsensoren sind in ihrer momentanen Anwendbarkeit für das häusliche Umfeld im „Ambient Assisted Living“ in unterschiedlichen Stadien zu finden. Einige wenige von ihnen sind bereits heute kostengünstig verfügbar, wie es am Beispiel der angebotenen Funksensoren von der Firma Siegenia [39] der Fall ist.

2.1.3 Dienste im AAL

Aufsetzend auf den technischen Grundlagen des „Home Automation“ und des „Ambient Intelligence“ sollen Dienste definiert und umgesetzt werden, die den älteren Menschen im Rahmen des AAL eine Unterstützung des alltäglichen Lebens gewähren. Die Entwicklung und Nutzung solcher Dienste ist noch in der Erforschung und Erprobung. Im Rahmen des Projekts „SmarterWohnen“ [40] des Fraunhofer ISST ließen sich die hier realisierten Dienste den folgenden Kategorien zuordnen: „Gesundheit“, „Sicherheit“, „Komfort“, „Soziales Umfeld“ und „Wirtschaftlichkeit“.

Dienste im Bereich „Gesundheit“ dienen dazu die Gesundheit des alternden Menschen möglichst lange zu erhalten und seine medizinische Betreuung zu unterstützen. Beispiele hierfür sind die in [41] angedeutete Ernährungsberatung und die Überwachung der Vitalwerte in „SmarterWohnen“.

Dienste im Bereich „Sicherheit“ sollen die Umgebung an die Sicherheitsbedürfnisse der Bewohner anpassen. Beispielsweise könnte sich die Beleuchtung in der Wohnung nachts beim Gang zur Toilette automatisch einschalten.

In „SmarterWohnen“ realisierte Dienste reagieren bei Gefahren in der Wohnung, wie bei Brand, Leckage oder Einbruch.

Dienste im Bereich „Komfort“ sind nicht ausschließlich auf die Bedürfnisse älterer Menschen zugeschnitten. Diese Dienste sollen den Wohn- und Erlebniswert einer Immobilie steigern. Beispiele hierfür sind Licht- und Akustikszenarien in Abhängigkeit von der Situation eines Benutzers. In „SmarterWohnen“ realisierte Dienste ermöglichen das Beziehen von Waren und Dienstleistungen über den heimischen Fernseher.

Dienste im Bereich „Soziales Umfeld“ dienen der Erhaltung und dem Aufbau von sozialen Kontakten des Bewohners. Sie können die Nutzung von Videokonferenzen oder der Einsatz eines Awareness Systems [42] sein. In „SmarterWohnen“ wird ein Dienst angeboten, über den die Bewohner gemeinsame Veranstaltungen planen und bekanntmachen können. So lassen sich Spieleabende oder gemeinsame Ausflüge organisieren.

Dienste im Bereich „Wirtschaftlichkeit“ sind dem Bereich „Home Automation“ zuzuordnen und nicht allein für die Bedürfnisse älterer Personen denkbar. In „SmarterWohnen“ können die Verbrauchswerte von Gas, Wasser oder Strom über den heimischen Fernseher angezeigt werden.

Wie wir sehen, sind viele der im Rahmen des AAL einsetzbaren Dienste nicht ausschließlich auf die Bedürfnisse älterer Personen fokussiert. Vielfach entstammen diese direkt aus dem Umfeld des Ambient Intelligence. Eine exemplarische Aufzählung von Diensten ist in [43] zu finden. Hier werden eine Reihe von Szenarien und Diensten aufgeführt, die an unterschiedlichen Stellen umgesetzt werden oder in Diskussion sind. Dazu gehören beispielsweise die Realisierung kontextabhängiger Lichtszenarien oder die Wechsel von Bildern in Abhängigkeit von der im Raum befindlichen Person in adaptiven Bilderrahmen.

In [44] wird eine verfeinerte Klassifikation von Anwendungen im AAL in den Kategorien „Sicherheit“ und „Komfort“ erarbeitet. Hier werden auch Dienste zur Verlängerung der Selbständigkeit der Bewohner aufgeführt. Sie werden in der Kategorie „Autonomy Enhancement Services“ zusammengefasst. Dienste werden in dieser Klassifikation auch noch danach unterschieden, ob sie im Haus oder außerhalb genutzt werden. Es ergibt sich hier eine sechsteilige Matrix, in der konkrete Dienstetypen identifiziert werden können, wie sie in der folgenden Tabelle dargestellt ist.

Tabelle 1: Klassifikation von Diensten im AAL nach J. Nehmer, M. Becker, A. Karshmer und R. Lamm

	Emergency Treatment Services	Autonomy Enhancement Services	Comfort Services
Indoor Assistance	Emergency prediction Emergency detection Emergency prevention	Cooking assistance Eating assistance Drinking assistance Cleaning assistance Dressing assistance Medication assistance	Logistic services Services for finding things Infotainment services
Outdoor Assistance	Emergency prediction Emergency detection Emergency prevention	Shopping assistance Travel assistance Banking assistance	Transportation services Orientation services

Hier erscheinen bereits wertvolle Hinweise, welche Arten von Diensten sinnvoll im AAL eingesetzt werden können. Es gibt eine Vielzahl von Beispielen für solche Dienste, von denen einige bereits vorgestellt wurden. Jedoch gibt es noch keine definierte Dienstemenge, die aus der Fachliteratur als Standard für das AAL vorausgesetzt werden kann. Die Identifikation, Entwicklung und Erprobung von sinnvollen Diensten ist noch ein aktuelles Forschungsfeld. Es ist zu erwarten, dass mit der Verfügbarkeit neuer Sensorik, aber auch mit der praktischen Erfahrung der Bewohner mit kontextadaptiven Diensten, neue Anwendungen identifiziert werden.

2.2 Context Awareness

Unter dem Themengebiet „Context Awareness“ verbirgt sich das Ziel Anwendungen zu realisieren, die sich selbständig an die natürliche Umgebung des Menschen anpassen. Zu diesem Zweck sollen Informationen aus dieser Umgebung gewonnen und in geeigneter Form zur weiteren Nutzung bereitgestellt werden. Dadurch sind Anwendungen nicht mehr von expliziten Nutzereingaben abhängig, sondern können direkt auf Ereignisse in der Umgebung reagieren.

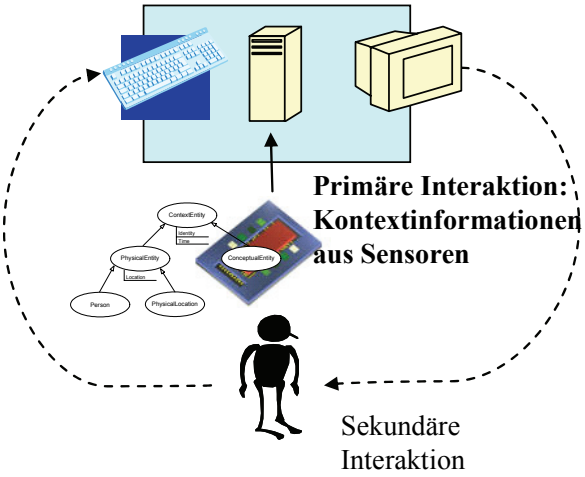


Abbildung 2: Interaktionszyklen kontextadaptiver Anwendungen

Die Erforschung kontextadaptiver Anwendungen ist heute noch in der Anfangsphase in der es viele Fragestellungen gibt, die derzeit noch unbeantwortet sind. Ein Problem stellt die Identifikation von Anwendungsfeldern dar, in denen die Kontextadaptivität sinnvoll genutzt werden kann. Hier gibt es eine Reihe von Anwendungen, die jedoch aus dem Stadium einer prototypischen Umsetzung kaum herausgekommen sind. Ein Grund ist hierfür sicherlich auch die Frage nach der Wirtschaftlichkeit einer Realisierung sowie des Betriebs kontextadaptiver Anwendungen. Kosten sind beispielsweise mit der Bereitstellung von Kontextinformationen über Sensorik verbunden. Fragen nach einer Kosten-Nutzung-Rechnung von konkreten kontextadaptiven Anwendungen sind genauso wenig beantwortet wie die Identifikation von möglichen Geschäftsmodellen. Im Rahmen eines EU-Projektes wurde versucht, einen Ansatz für Geschäftsmodelle in aktiven Netzen auf der Basis fester und mobiler Kommunikationsinfrastrukturen zu erarbeiten [45]. Hinweise auf den ROI kontextadaptiver Anwendungen in einzelnen Anwendungsdomänen sind partiell zu finden [46].

Eine generelle und noch offene Fragestellung betrifft die effiziente Entwicklung von kontextadaptiven Anwendungen. Hier geht es um geeignete Designprinzipien bezogen auf Architektur und Modellierung, aber auch um die Verfügbarkeit von Frameworks oder Werkzeugen. Es gibt zwar unterschiedliche Ansätze, von denen sich jedoch noch keiner als Standard durchsetzen konnte. Gerade in der Erhebung und Nutzung von Kontextinformationen finden sich Problemstellungen im Detail, die ebenfalls noch in der Diskussion sind. Dazu gehört beispielsweise die Unsicherheit bzw. Ungenauigkeit bei der Erkennung von Kontextinformationen [47] [48], sowie die daraus resultierenden Konsequenzen für die Mensch-Maschine-Interaktion. Bereits in der Frühzeit der Dis-

kussion um die Nutzung kontextadaptiver Anwendungen wurde auf die Gefahren einer Kontextadaptivität auf der Grundlage unsicherer Kontextinformationen hingewiesen [49]. In aktuellen Lösungsansätzen wird vorgeschlagen, den Nutzer stärker in die Interaktion einzubindenden [50]. Empirische Untersuchungen zeigen jedoch, dass dieser Ansatz nicht immer tragfähig ist [51]. Ein weiteres Problemfeld ist die Wahrung der Privatsphäre der Nutzer. Kontextinformationen spiegeln den Zustand der Umgebung in einer maschinell interpretierbaren Form wieder. Dazu können auch personenbezogene Informationen gehören. Es gibt unterschiedliche Lösungsansätze um die Privatsphäre des Nutzers zu wahren [52] [53] [54], aber auch hier ist weiterer Diskussionsbedarf vorhanden.

Auf der Ebene der Sensorik zeichnen sich interessante Entwicklungen ab, die jedoch nicht innerhalb des Forschungsbereichs „Context Awareness“ betrachtet werden. Dazu gehören unterschiedlichste Sensor-Hardware [55], die Gewinnung von Kontextinformationen mit Hilfe von Bild- oder Audio-Analyseverfahren [41] oder die Kommunikationsinfrastruktur zur Einbindung von Kontextsensorik [56].

Nach dieser groben Übersicht über das Themengebiet werden nachfolgend einige Definitionen zu wichtigen Begriffen gegeben. Anschließend werden wichtige Fragestellungen und Konzepte beschrieben, die im Zusammenhang mit dieser Arbeit stehen.

2.2.1 Kontext

Der Begriff „Kontext“ ist im Sprachgebrauch mehrdeutig belegt. Je nach dem Zusammenhang wird hierunter ein jeweils anderes Konzept verstanden. Dieses gilt auch für die Verwendung des Begriffs in der Informatik. Daher ist es notwendig, nachfolgend diesen Begriff so zu definieren, wie er in diesem Rahmen zu verstehen ist.

Die Nutzung des Begriffs „Kontext“ entspricht nachfolgend der Definition von A. K. Dey [57]: “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 application themselves”. Mit Kontext ist hier also jegliche charakteristische Information gemeint, die genutzt werden kann, um die Situation einer Entität zu beschreiben. Diese kann dabei eine Person, ein Ort oder ein beliebiges Objekt sein, welches als relevant für die Kommunikation zwischen einem Anwender und einer Anwendung betrachtet wird. Auch die Anwendung selbst kann als Entität betrachtet werden.

Der Kontextbegriff ist hier ein technischer. Mit ihm wird eine technische Beschreibung relevanter Aspekte einer Situation in der realen Welt verstanden.

Eine solche kann durch beobachtbare Zustände von Objekten gekennzeichnet werden. So kann beispielsweise die Situation „Brand“ dadurch beobachtet werden, dass in einem Raum Rauch gemessen wurde. Die Relevanz einer solchen Information bezieht sich auf deren Nutzen für die Adaptivität einer Anwendung. Somit impliziert die Definition eine bewusste Auswahl von Entitäten und der auf ihnen zu beobachtenden Zustände in Bezug auf das erforderliche adaptive Verhalten einer Anwendung.

2.2.2 Kontextadaptivität

Nach A. K. Dey gilt für eine kontextadaptive Anwendung [57]: „A system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user’s task”. Allgemeiner ausgedrückt ist eine Anwendung kontextadaptiv, wenn ihr Verhalten primär vom Kontext abhängig und nicht auf explizite Nutzerinteraktion angewiesen ist.

Das Verhalten einer Anwendung in Bezug auf einen Kontext, bzw. deren Adaptivität kann sich in unterschiedlicher Weise ausprägen. In [58] wird eine Klassifikation der Adaptivität von Anwendungen aus Sicht der Nutzung der Kontextinformationen vorgestellt:

- Kontextbasierte Auswahl: Informationen und Dienste werden von einer Anwendung in Abhängigkeit des Kontextes ausgewählt. Ein Beispiel hierfür sind Standortbezogene Dienste bzw. Location Based Services (LBS) [59].

- Kontextbasierte Präsentation: In Abhängigkeit vom Kontext erfolgt eine Adaption der Nutzerschnittstelle. Beispielsweise kann ein Mobiltelefon auf stumm geschaltet werden, sobald es erkennt, dass der Nutzer sich in einer Sitzung befindet.

- Kontextgetriggerte Aktion: Die Anwendung ist in der Lage, automatisch auf Ereignisse in der Umgebung zu reagieren und Aktionen auszulösen. Bei Erkennung eines Eibruchs in ein Gebäude kann beispielsweise eine Einbruchsmeldung an einen Sicherheitsdienst geschickt werden.

- Annotation von Kontextinformationen: An Artefakten werden die zum Zeitpunkt der Entstehung erhobenen Kontextinformationen annotiert. Diese können zu einem späteren Zeitpunkt genutzt werden, z.B. für kontextbasiertes Suchen nach Dokumenten.

Ergänzend können Kontextinformationen auch zur reinen Visualisierung und Information eines Nutzers verwendet werden, z.B. zur Darstellung seiner momentanen Position.

Schilit [60] gibt eine alternative Klassifikation von kontextadaptiven Anwendungen vor. In einer Matrix mit zwei orthogonalen Dimensionen mit je zwei Punkten werden vier Anwendungsklassen identifiziert. Die Dimensionen sind: Bereitstellen von Informationen vs. Ausführen von Kommandos und manuelle vs. automatische Aktionen. In der Matrix sind die folgenden Klassen zu finden:

- Proximate selection: In diese Klasse werden Anwendungen eingeordnet, die auf explizite Anforderung des Nutzers hin Informationen kontextbezogen bereitstellen. Dieses kann beispielsweise eine ortsbezogene Suche nach Restaurants in der Umgebung sein.

- Automatic Reconfiguration: Hier erfolgt die Bereitstellung der jeweiligen Informationen automatisch ohne explizite Nutzerinteraktion. So kann der Nutzer beispielsweise automatisch mittels SMS informiert werden, wenn eine Postlieferung im Briefkasten liegt.

- Contextual Commands: In dieser Klasse werden Anwendungen zusammengefasst, die ein vom Nutzer initiiertes Kommando an den jeweiligen Kontext anpassen. Beispielsweise kann ein kontextadaptiver Mailserver eine Mail an alle einer Sitzung beteiligten Personen zustellen, ohne dass der Absender den Personenkreis selbst ermitteln muss.

- Context-triggered Actions: Hier werden Aktionen direkt von der Anwendung entsprechend dem Kontext ausgelöst. Der Nutzer ist jetzt nicht Bestandteil einer Interaktion. Beispielsweise kann bei Erkennung eines Notfalls automatisch eine geeignete Gegenmaßnahme eingeleitet werden.

Gleichgültig welcher dieser Klassifizierungen man folgen möchte, so zeigen sie doch die vielfältigen Arten und Möglichkeiten der Adaptivität von Anwendungen auf. Entscheidend ist, dass die Kontextadaptivität in den seltensten Fällen die eigentliche Funktionalität einer Anwendung ausmacht. Vielmehr ist sie eine Querschnittsfunktion einer Anwendung an der Mensch-Maschine-Schnittstelle.

2.2.3 Kontextadaptive Anwendungen

In [61] wird ein Überblick über kontextadaptive Anwendungen zum Jahr 2000 gegeben. Diese Liste ist nicht vollständig, zeigt jedoch, dass diese weitestgehend akademische Prototypen sind, die keine kommerzielle Umsetzung bekommen haben. Folgende Anwendungen werden dort aufgeführt:

- Active Badge System [62]: Dieses wurde in den frühen 90er Jahren im Olivetti Research Lab entwickelt. Mitarbeiter wurden über das System geortet und Telefonanrufe konnten an das dem Mitarbeiter am nächs-

ten liegende Telefon weitergeleitet werden. Die Ortung erfolgt mit Hilfe von IR-Barken.

- ParcTab System [63]: Dieses System wurde etwa zur gleichen Zeit am Xerox Palo Alto Research Center entwickelt. Es basiert auf einem „Personal Digital Assistant“ (PDA) [64] und einer Kommunikation mittels Infrarot (IR). Raumweise wurden Infrarot-Zugänge geschaffen, über die zum einen eine Ortung stattfinden konnte, aber auch die Kommunikation mit einem zentralen Server. Auf dieser Basis wurden eine Reihe von kontextadaptiven Anwendungen realisiert und ausgetestet, z.B. die Darstellung der Ortungsinformationen eines Nutzers, Informationen zur nächstgelegenen lokalen Ressource (z.B. Drucker) oder als Fernbedienung für Geräte in den einzelnen Räumen.

- Anwendungen der Georgia Institute of Technology basierend auf dem Context Toolkit [65]: In/Out board zur Darstellung der Anwesenheit von Personen, Information Display zur Darstellung ortsbezogener Informationen, DUMMBO als kontextadaptive Erweiterung eines digitalen Whiteboards, der Conference Assistant als Informationssystem für Besucher von Konferenzen [66].

- Cyberguide [67]: Ein mobiles kontextadaptives Touristeninformationssystem entwickelt in den 90ern im Rahmen des Georgia Tech Cyberguide Projekts.

- GUIDE [68]: Ein Touristeninformationssystem der Universität Lancaster. Basierend auf der Ortung und von Nutzerpräferenzen werden dem Touristen relevanten Informationen zur Umgebung bereitgestellt.

- Smart Sight Tourist Assistant [69]: Ein Touristeninformationssystem der Carnegie Mellon Universität. Das Global Positioning System (GPS) [70] wird als Sensor zur Ortung des Nutzers verwendet. Neben der Bereitstellung von interessanten ortsbezogenen Informationen kann der Tourist auch Informationen über für ihn selbst wichtige Orte speichern.

- Forget-Me-Not [71]: Anfang der 90er wurde am Rank Xerox Research Center ein System entwickelt, welches Kontextinformationen des Nutzers speichert und für eine spätere Recherche zur Verfügung stellt. Erhobene Kontextinformationen sind beispielsweise der Ort des Nutzers, die Personen in seiner Umgebung und mit welchen Personen Telefonate geführt werden.

- Remembrance Agent [72]: 1996 wurde am MIT Media Lab eine tragbare Erinnerungshilfe entwickelt. Entsprechend dem aktuellen Kontext stellt ein tragbarer Computer Informationen, die in der jeweiligen Situation

nützlich sein könnten, zur Verfügung, wie Notizen, alte EMails, Papiere und andere textbasierte Informationen.

- StartleCam [73]: Am MIT Media Lab wurden eine tragbare Videokamera, ein Computer und eine Reihe von Sensoren miteinander vernetzt. Die Videokamera wurde gestartet, sobald das System ein für den Nutzer interessantes Ereignis erkannt hat. Ein Sensor zur Messung der Leitfähigkeit der Haut wird verwendet um daraus die Erregtheit des Nutzers abzuleiten.

Ab dem Jahr 2000 bis heute gibt es eine Vielzahl von prototypischen Entwicklungen kontextadaptiver Anwendungen, die in ihrer Vollständigkeit nicht aufgeführt werden können. Interessante Beispiele neuerer Realisierungen sind ein kontextadaptives Chat-Programm [74], ein mobiler digitaler Berater [75] oder ein adaptives Mobiltelefon [76]. Ein Beispiel für eine Entwicklung am Fraunhofer ISST ist der digitale Touristenbegleiter für die Olympischen Spiele in Peking [77]. Dieser realisiert eine situationsabhängige Versorgung des Touristen mit Informationen und Diensten. Situationen werden aus den Positionsinformationen des Nutzers und der Zeitdimension abgeleitet sowie aus Profilinformationen und einer Ortsontologie. Alle diese Entwicklungen haben jedoch ihren Weg zur kommerziellen Nutzung noch nicht gefunden.

Erste kommerzielle Ansätze beschränken sich auf die Verwendung des Ortes als einzige Kontextdimension in den Standortbezogenen Diensten. Weit verbreitet sind hier die Navigationssysteme. Darüber hinaus bieten viele Mobilfunkanbieter noch Dienste im Bereich Unterhaltung, Information, Sicherheit und Notfall, sowie Community an. Die anfangs hohen Erwartungen von Analysten an den kommerziellen Erfolg dieser Lösungen konnten aber bei weitem noch nicht erreicht werden. Dafür gibt es vielfältige Gründe. In einer Studie [78] wurden dazu potentielle Anwender zu aktuellen Hinderungsgründen befragt. Gut 43,3% gaben an, dass sie keinen Bedarf für die verfügbaren Dienste hätten. 26% gaben an, dass diese zu teuer seien. Weitere Gründe sind die mangelnde Transparenz bei der Preisgestaltung, sowie Datenschutzbedenken.

2.2.4 Kontextmodell

Ein Kontextmodell ist eine formale Beschreibung der relevanten Aspekte der realen Welt. Ein solches Modell ist notwendig, um die reale Welt mit der technischen Sicht kontextadaptiver Anwendungen zu verbinden [58]. Ein Kontextmodell ermöglicht die Trennung einer Anwendung von der Sensorik. Sensorinformationen können in ein Kontextmodell überführt werden und beschreiben in der Gesamtheit den Zustand des definierten Ausschnitts. Das Kontextmodell abstrahiert von der Technologie der Erhebung der Kontextinformationen. Auf dieser Abstraktion können kontextadaptive Anwendungen aufsetzen und auf

die bereitgestellten Kontextinformationen zugreifen. Wie bereits in der Definition von Dey [57] ersichtlich, beschreibt ein Kontextmodell nur einen Ausschnitt aus der Menge der möglichen beobachtbaren Entitäten und deren Zustände. Somit ist ein Kontextmodell ein Ausschnitt aus einem allgemeinen Weltmodell. Die konkrete Ausprägung ist abhängig vom erwarteten kontextadpativen Verhalten einer Anwendung.

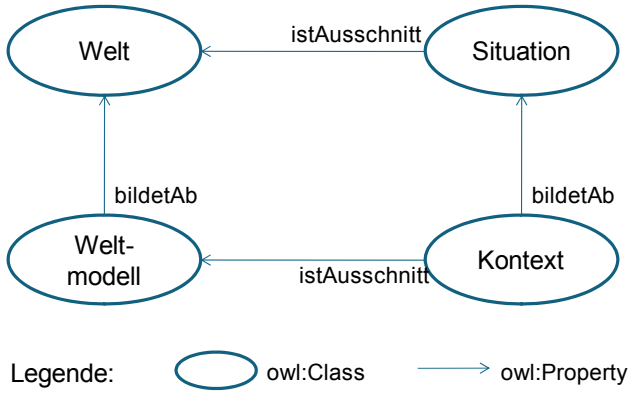


Abbildung 3: Kontext als technische Abbildung eines Ausschnitts der realen Welt

Über ein formales Modell muss ein Ansatz gefunden werden, um Kontextinformationen in einer strukturierten Form zu beschreiben. Es gibt heute bereits unterschiedliche Ansätze zur Kontextmodellierung. In [4] wird eine selektive Übersicht über aktuelle Technologien und Ansätze zur Kontextmodellierung gegeben. Sie können wie folgt klassifiziert werden:

- Abbildung von Kontextinformationen durch **Name-Wert-Paare**: Dieser einfache Ansatz wird häufig verwendet, um Dienste mit Kontextinformationen zu annotieren, z.B. [79] [80]. Das Filtern geeigneter Dienste erfolgt dann mit Hilfe eines exakten Abgleichs auf Strings.

- Beschreibung von Kontextinformationen mittels **XML**: Als erweiterter Ansatz wird XML zur Beschreibung von Kontextinformationen verwendet. Mit Hilfe von XML-Schemata kann eine verbindliche Vorgabe an Struktur und Bedeutung von Kontextinformationen gegeben werden. Beispiele hierfür sind CSCP (Comprehensive Structured Context Profiles) [81], PPDL (Pervasive Profile Description Language) [82] oder ConteXtML [83]. XML-basierte Kontextmodelle werden meist verwendet um kontextbezogene Profilinformationen zu verwalten und bereitzustellen.

- **Graphische Modelle**: Die Erstellung von Kontextmodellen auf der Basis graphischer Modellrepräsentationen ist gerade für die Nutzung durch den Menschen geeignet. Eine solche Repräsentationsform unterstützt

beispielsweise den Erstellungsprozess eines konzeptuellen Kontextmodells – analog zur Erstellung von konzeptuellen Datenmodellen in Form von Entity Relationship Diagrammen. Meistens werden diese Repräsentationsformen in eine ausführbare Form des Kontextmodells überführt [82]. Beispiele für die Modellierung von Kontextaspekten mit Hilfe von UML finden sich in [84]. Ein weiterer Ansatz für eine graphische Modellierung wird in [85] vorgestellt. Hier wird der Ansatz des Object-Role Modeling erweitert um Aspekte der Kontextadaptivität.

- **Objektorientierte Modelle:** Mit Hilfe objektorientierter Kontextmodelle wird versucht, die Realisierung kontextadaptiver Anwendungen zu vereinfachen, indem die Komplexität der Gewinnung von Kontextinformationen in der Implementierung verkapselt wird. Über definierte Schnittstellen hat der Entwickler Zugriff auf die bereitgestellten Kontextinformationen. Diese Implementierungen können in unterschiedlichen Anwendungen wiederverwendet werden. Beispiele für diesen Ansatz finden sich in [86] [87] [88].

- **Logikbasierte Modelle:** Logikbasierte Ansätze eignen sich besonders zur Beschreibung formaler Systeme und stellen Mechanismen zur Verfügung, um aufgrund von als „wahr“ befundenen Aussagen auf weitere Aussagen zu schließen [89]. In diesen Systemen kann man Regeln für die Ableitung von Ausdrücken oder Fakten auf der Basis bereits bekannter Ausdrücke oder Fakten definieren. Werden beispielsweise Kontextinformationen durch Ausdrücke oder Fakten repräsentiert, so können auf der Basis der Regeln zusätzliche Kontextinformationen abgeleitet werden. Beispiele finden sich in [90] [91] [92].

- **Ontologie-basierte Modelle:** Ontologien ermöglichen die Wissensrepräsentation eines formal definierten Systems mit Hilfe von Begriffen und Relationen. Zusätzlich enthalten Ontologien Inferenz- und Integritätsregeln, mit denen die Gültigkeit von Zuständen überprüft, aber auch neues Wissen abgeleitet werden kann. Eine Vielzahl von Beispielen existieren für diesen Ansatz [93] [94] [95] [96].

Aktuell werden vermehrt ontologie-basierte Kontextmodelle diskutiert. Trotz der vielfältigen Diskussionen gibt es aber noch keinen Standard, weder in der Art der Repräsentation in einer der oben beschriebenen Ansätze, noch in der konkreten Ausprägung einer Anwendungsdomäne. Hier ist ebenfalls weiterer Diskussionsbedarf vorhanden.

Gegenstand dieser Dissertation ist die Kontextmodellierung unter den besonderen Anforderungen des Ambient Assisted Living. Es zeigt sich, dass es unterschiedliche Einsatzzwecke für ein Kontextmodell gibt und dass somit auch unterschiedliche Technologien benötigt werden.

2.2.5 Metamodelle

Ein wichtiger Bestandteil dieser Dissertation ist die Definition eines Metamodells für Kontextmodelle im Ambient Assisted Living. Metamodelle sind Modelle von Modellen. Ein Metamodell definiert dabei den grundsätzlichen Aufbau eines Modells und die Semantik der darin verwendeten Modellelemente. Eine solche Definition erfolgt dabei jeweils in Hinblick auf die jeweilige Anwendungsdomäne, für welche konkrete Modelle erstellt werden sollen. Dabei werden die zur Beschreibung der Anwendungsdomäne benötigten Modellelemente, deren Bedeutung und Einsatzbedingungen definiert. Auf dieser Basis können konkrete Modelle für die jeweilige Anwendungsdomäne erstellt werden.

Wichtige Modellierungssprachen basieren auf einem Metamodell. Ein bekanntes Beispiel für die Metamodellierung ist das Meta Object Facility (MOF) [97]. Es wurde entwickelt um Metamodelle spezifizieren zu können. Ein Beispiel hierfür ist die Unified Modeling Language (UML) [98]. Diese bildet sogar eine 4-schichtige Architektur, deren oberste Schicht ein Meta-Metamodell ist. Es ermöglicht eine Erweiterung des Metamodells und damit auch der Modellierungssprache.

Tabelle 2: 4-Schichtenarchitektur von UML

Schicht	Beschreibung	Beispiel
Meta-Metamodell	Infrastruktur für eine Architektur zur Metamodellierung. Definiert die Sprache zur Spezifikation des Metamodells, z.B. MOF.	Meta-Klasse, Meta-Attribut, Meta-Operation, Meta-Beziehung
Metamodell	Instanz eines Meta-Metamodells. Definiert die Sprache zur Definition eines Modells.	Klasse, Attribut, Operation, Assoziation
Modell	Instanz eines Metamodells. Spezifiziert Konzepte zur Beschreibung eines Anwendungsbereichs.	Teilnehmer, Datum, Tagung, Vortrag
Benutzer-Objekte	Instanz eines Modells. Definiert einen speziellen Fall des Anwendungsbereichs.	Dr. Maier, 12.02.2008, CAPS 2008, „Vom Verstehen der Metaebene“

Die im Kapitel zuvor beschriebenen Ansätze zur Kontextmodellierung basieren auf einem implizit oder explizit definierten Metamodell. Im Falle der Verwendung von Name-Wert-Paaren ist dieses Metamodell sehr einfach und generell. Die dort definierten Elemente können in unterschiedlichen Domänen eingesetzt werden und haben keinen expliziten Bezug zur Kontextmodellierung. Die in [85] vorgestellte graphische Kontextmodellierung hat dagegen einen expliziten Bezug der Modellelemente auf Konzepte im Context Awareness.

In Hinblick auf die Realisierung eines Kontextmodells für das AAL hat die explizite Definition eines Metamodells folgende Vorteile:

- Die relevanten Konzepte der Kontextmodellierung sind explizit in den Modellelementen definiert. Darüber lassen sich konkrete Kontextmodelle näher an der Domäne definieren. Dieses führt zu einer domänenspezifischen Modellierungssprache. Damit wird sichergestellt, dass die Anforderungen an die Ausdrucksmächtigkeit eines Kontextmodells erfüllt werden. Zudem sind diese leichter in dieser Domäne zu lesen und zu verstehen.

- Auf der Basis eines domänenspezifischen Metamodells lassen sich Kontextmodelle in unterschiedlichen Technologien (teilweise) ineinander überführen, z.B. aus einer XML-Beschreibung in ein graphisches Modell. Dieses ist eine wichtige Eigenschaft der hier vorgestellten Kontextmodellierung.

Ein wesentlicher Bestandteil eines Metamodells ist die Definition der darin enthaltenen Modellelemente. Diese beschreiben die Einzelteile, aus denen ein konkretes Kontextmodell zusammengestellt werden kann. Nachfolgend sollen die Modellelemente beschrieben werden, die aktuellen Ansätzen der Kontextmodellierung implizit oder explizit zugrunde liegen:

- Context Entity: Dieses Modellelement kann verwendet werden, um eine Person, einen Ort oder einen Gegenstand aus der realen Welt zu repräsentieren.

- Context Relation: Dieses kann verwendet werden, um Beziehungen zwischen Entitäten zu repräsentieren. So kann damit zum Beispiel ausgedrückt werden, dass sich eine Person an einem Ort befindet.

- Context Attribute: Sie kann man verwenden, um die relevanten Informationen zu einer Entität oder einer Relation zu repräsentieren.

Diese Modellelemente werden nachfolgend noch detaillierter beschrieben. Sie bilden ebenfalls eine Grundlage dieser Arbeit, aber werden aufgrund erweiterter Anforderungen um zusätzliche Modellelemente ergänzt. Ein Sonderfall in der Kontextmodellierung ist der 5W1H Ansatz (Who, What, Where, When,

How, Why) [99]. Hier steht die Beschreibung des Benutzers und seines Bezugs zur Umgebung im Mittelpunkt. Die oben beschriebenen Modellelemente stecken hier implizit in den sechs Beschreibungsaspekten einer Kontextinformation. Der Ansatz bildet eine weitere Spezialisierung der Kontextmodellierung in Bezug auf eine personenzentrierte Betrachtung. Für die Dissertation hat sich diese Spezialisierung als nicht ausreichend erwiesen, da auch Ereignisse unabhängig von einer konkreten Person betrachtet werden müssen, wie ein Brand in der Wohnung.

2.2.5.1 Context Entity

Eine Context Entity bzw. Kontextentität definiert das Objekt über das eine Beobachtung gemacht wird bzw. eine Kontextinformation gegeben wird. Diese steht somit im Mittelpunkt der Beobachtung.

Bereits in der Definition von Dey [57] wird aufgeführt, was Kontextentitäten sein können: Personen, Orte und Gegenstände, die für die Kontextadaptivität von Anwendungen relevant sind. Somit können potentiell jegliche Art von Objekten aus der realen Welt als Entität in einem Kontextmodell aufgenommen werden. Es gibt mehrere Arbeiten in denen solche Objekte identifiziert und ontologisch geordnet wurden, z.B. [100]. Diese werden nach verschiedenen Kriterien in einer Generalisierungs-/Spezialisierungs-Hierarchie gegliedert. In CONON (CONtext ONtology) [95] wird beispielsweise eine Upper Ontology definiert. In dieser ist die Wurzel die abstrakte Klasse „Context Entity“. Diese unterteilt sich dann in „Computing Entity“, „Person“, „Location“ und „Activity“. Diese Upper Ontology beschreibt generelle Eigenschaften von grundsätzlichen Kontextentitäten. Dazu können dann domänenspezifische Ontologien definiert werden, die die generellen Konzepte erweitern und im Detail beschreiben. In [101] wird eine Ontologie definiert, an deren Wurzel die Entitäten „Person“, „Kommunikationsgerät“ und „Kommunikationskanal“ beschrieben sind. Diese werden dort noch weitergehend detailliert.

Die Identifikation der relevanten Kontextentitäten ist je nach Einsatzzweck und –domäne unterschiedlich. Es gilt daher, eine entsprechende Ausprägung für die Domäne „Ambient Assisted Living“ zu finden. Unabhängig von der Nutzung von Ontologien zur Implementierung eines Kontextmodells, sind grundsätzliche Anforderungen zu Beschreibung von Kontextentitäten aus den bestehenden Ansätzen zu erkennen. Zum einen müssen Entitäten in einer Generalisierungs-/Spezialisierungshierarchie eingeordnet werden können. Zum anderen muss es möglich sein, abstrakte Kontextentitäten zu definieren und die gemeinsamen Merkmale von konkreten Kontextentitäten, die jedoch selbst nicht instanziiert werden können, zu bündeln. Eine weitere Anforderung ist das Modellierungselement der Aggregation. Eine Aggregationsbeziehung zwischen Kontextentitäten beschreibt den Aufbau einer durch eine andere Entität, die

der ersteren zugehörig ist. Ein Beispiel hierfür ist der Aufbau einer Wohnung durch darin enthaltene Räume.

2.2.5.2 Context Relation

Eine Context Relation bzw. Kontextbeziehung beschreibt explizit eine Beziehung zwischen einer oder mehrerer Context Entities auf der Modellebene. Beziehungen auf der Meta-Modellebene, z.B. die Generalisierungs-/Spezialisierungsbeziehung zwischen Entitäten werden nicht durch dieses Modellelement ausgedrückt. Das Modellelement „Context Relation“ stellt noch nicht die Information über eine konkrete Beziehung zur Verfügung, sondern rückt diese in den Mittelpunkt der Beobachtung. Wird beispielsweise eine Kontextbeziehung zwischen einer Person und einem Raum definiert, so bedeutet es nicht, dass die Person im Raum enthalten ist. Vielmehr drückt es aus, dass eine solche Beziehung zwischen der Person und einem Raum beobachtet werden kann.

Nachfolgend werden einige Beispiele von Kontextbeziehungen aufgeführt, um hier mögliche Eigenschaften zu identifizieren und zu beschreiben, die dieses Modellelement besitzen kann. Solche Beziehungen lassen sich auch unter Verwendung der Graphentheorie durch eine Menge von Knoten und Kanten beschreiben. Die möglichen Eigenschaften von Graphen dienen als Grundlage für die Identifizierung von Eigenschaften des Kontextelements „Context Relation“.

Ein Beispiel für eine Kontextbeziehung zwischen Instanzen nur einer Kontextentität ist „<Person> ist befreundet mit <Person>“. Die Beziehung in diesem Beispiel ist ungerichtet. Ist Person p1 mit Person p2 befreundet, so gilt dieses auch umgekehrt. Die Kardinalität der Beziehung bzw. die Valenz eines diese Beziehung beschreibenden ungerichteten Graphen ist unbestimmt. Eine Person p1 kann mit mehreren Personen befreundet sein. Es kann aber auch Personen ohne eine solche Beziehung geben.

Ein Beispiel für eine Kontextbeziehung zwischen Instanzen zweier Kontextentitäten ist “<Person> befindet sich in <Raum>“. Die Beziehung in diesem Beispiel ist gerichtet. Sie geht von der Entität „Person“ aus und endet in der Entität „Raum“. Es befindet sich eine Person in einem Raum und nicht umgekehrt. Die Kardinalität der Beziehung bzw. die Valenz ist dadurch bestimmt, dass sich eine Person immer nur in einem Raum befinden kann. Der Eingangsgrad des gerichteten Graphen ist hier mit 1 festgelegt. Der Ausgangsgrad ist unbestimmt, da sich auch mehr als nur eine Person in einem Raum befinden kann. Dieser Graph ist bipartit und verbindet nur Knoten vom Typ „Person“ mit Knoten vom Typ „Raum“.

Theoretisch können Kontextbeziehungen auch zwischen Instanzen von mindestens drei Kontextentitäten definiert werden. Der Bedarf für einen sogenannten Hypergraphen ist jedoch in praktischen Beispielen und in der Literatur nicht bekannt und wird daher nicht weiter verfolgt.

In vielen Ansätzen werden Kontextbeziehungen nicht in den Modellen berücksichtigt. Dieses gilt insbesondere für ältere Ansätze, z.B. auf der Basis von Name-Wert-Paaren oder XML-Beschreibungen. Werden Informationen zu Beziehungen im Kontextmodell benötigt, so werden diese nicht explizit modelliert, sondern als Kontextinformation zu einer Kontextentität zugeordnet. Eine Beziehung „<Person> befindet sich in <Raum>“ wird dann so modelliert, dass die Raumbezeichnung als Kontextinformation der Person zugeordnet wird. Es gibt jedoch auch neuere Ansätze, in denen das Konzept der Kontextbeziehung expliziter Bestandteil des entsprechenden Metamodells ist. Beispiele dafür sind Kontextmodelle, die durch Ontologien beschrieben werden.

2.2.5.3 Context Attribute

Ein „Context Attribute“ bzw. Kontextattribut definiert auf der Modellebene die Information, die beobachtbar und für die Beschreibung einer Kontextentität oder Kontextbeziehung relevant ist. Der Wert in einem Kontextattribut beschreibt den Zustand einer Entität oder Beziehung in dem definierten Aspekt. In der Definition von Dey [57] sind das jegliche Informationen, die zur Charakterisierung der Situation einer Entität verwendet werden können.

Mit der Zuordnung eines Kontextattributs zu einer Entität oder einer Beziehung ergibt sich die Semantik der darin enthaltenen Information. Ein Kontextattribut kann verschiedenen Entitäten und Beziehungen zugeordnet werden. So kann beispielsweise ein Attribut „ID“ für alle Entitäten relevant sein. Nach Möglichkeit sollte eine über Entitäten hinweg zutreffende Eigenschaft einer gemeinsamen eventuell abstrakten Generalisierung zugeordnet werden. Die ID könnte so beispielsweise einer Entität „Context Entity“ zugeordnet werden, von welcher alle anderen Kontextentitäten abgeleitet sind. Eine Menge von Kontextattributen zu einer Entität kann einen gemeinsamen Aspekt dieser Entität beschreiben. Hier sollte es möglich sein, diese Attribute als solche zu gruppieren. Ein Beispiel hierfür sind der Blutdruck und der Blutzucker zu einer Person. Diese Kontextattribute können als die Vitalwerte der Person gruppiert werden.

Die Zuordnung eines Kontextattributs zu einer Beziehung wird nur in wenigen Kontextmodellen eingesetzt. Jedoch kann es Sinn machen neben der Erfüllung einer auch weitere Zustände einer Beziehung zwischen Entitäten zu beschreiben. Ein Beispiel hierfür ist die Relation „ist in der Nähe“ zwischen einer Entität „Person“ und einer Entität „Raum“. Die Entfernung kann als Attribut der Relation die Beziehung zwischen den Entitäten genauer beschreiben.

Kontextattribute können sich in ihrer Wertigkeit unterscheiden. Eine Vielzahl von Attributen ist einwertig. Diesen ist jeweils immer genau ein Wert zugewiesen, z.B. „ID“ oder „Alter“. Daneben kann es aber auch mehrwertige Attribute geben, die keinen, einen oder mehrere Werte beinhalten. Ein Beispiel hierfür ist die Definition von „befreundet mit“ als Kontextattribut, in dem die Namen der Freunde direkt einer Entität „Person“ zugewiesen werden. Ein weiteres Beispiel für ein mehrwertiges Attribut ist die Definition eines Werteabschnitts. So kann beispielsweise ein Zeitraum durch eine Anfangszeit und eine Endzeit beschrieben werden. Deshalb kann eine weitere Eigenschaft eines Kontextattributs die Beschreibung eines Werteabschnitts sein.

Eine weitere Unterscheidung kann in der Komplexität der Kontextattribute gefunden werden. Die Mehrzahl der Kontextinformationen ist durch einfache Kontextattribute beschrieben. Es gibt jedoch auch zusammengesetzte, die aus einer Menge von einzelnen Kontextattributen bestehen. Als Beispiel kann hier der „Name“ einer Person beschrieben werden, der sich aus den konkreten Attributen „Vorname“ und „Familienname“ zusammensetzt.

Neben diesen generellen Eigenschaften von Kontextattributen können noch weitere Eigenschaften definiert werden, die sich auf die konkreten Werte in der Instanziierung eines Kontextmodells beziehen. Diese Eigenschaften können genutzt werden, um unabhängig vom Modellwissen einer Anwendungsdomäne konkrete Attributwerte abzuleiten oder zu validieren. Eine solche Eigenschaft ist die Eindeutigkeit. Besitzt ein Attribut diese Eigenschaft, so darf es keine unterschiedlichen Instanzen einer Kontextentität geben, die den gleichen Wert in dem jeweiligen Attribut besitzen, wie beispielsweise für das Attribut „ID“. Eine weitere Eigenschaft bezieht sich auf die Ordnung der in der zugehörigen Kontextdimension definierten Wertemenge. Eine Bedingung kann definiert werden, dass Werte nur in ab- bzw. aufsteigender Reihenfolge in das Kontextattribut übernommen werden dürfen. Diese trifft beispielsweise bei der Zeitdimension zu.

2.2.6 Zeitbezug von Kontextinformationen

Aktuelle kontextadaptive Anwendungen und somit auch die zugrundeliegenden Kontextmodelle sind auf die Bereitstellung und den Zugriff auf gültige Kontextinformationen fokussiert. Es wird auf aktuell erhobenen Kontextinformationen reagiert. Hier ist der Zeitbezug der benötigten Kontextinformationen die augenblickliche Zeit. Es sind aber kontextadaptive Systeme denkbar, deren Zeitbezug auch in die Vergangenheit oder in die Zukunft reichen kann. Ein Beispiel hierfür ist eine kontextadaptive Anwendung, die auf geplante Termine reagieren kann. Sensoren für zukünftige Ereignisse können beispielsweise den möglichen Zugriff auf einen Terminkalender voraussetzen. Aktuell sind Ansätze in der Forschung zu finden [102] [103], die unter dem Schlagwort „Context Prediction“ die Ableitung zukünftiger Ereignisse unter Verwendung

der Historie untersuchen. Die Verwendung einer Kontexthistorie ist interessant unter dem Gesichtspunkt der Verwaltung und Bereitstellung von Kontextinformationen. Die entstehende Datenmenge kann dabei zu hohem Ressourcenbedarf und entsprechenden Kosten führen. Ansätze zur effizienten Historisierung von Kontextinformationen sind aus Veröffentlichungen nicht bekannt. Für uns spielt der Zeitbezug ebenfalls eine Rolle, da in der Anwendungsdomäne „Ambient Assisted Living“ beispielsweise die Erinnerung an einen Arzttermin die Bereitstellung zukünftiger Ereignisse als Kontextinformation notwendig macht.

2.2.7 Privacy und Security

Ein gegenwärtig heftig diskutiertes Problem ist die Wahrung der Privatsphäre des Benutzers und der Schutz der Kontextinformationen vor unbefugtem Zugriff. Es werden eine Vielzahl von Informationen über den Nutzer und seine Umgebung erhoben und den vom Anwender gebrauchten kontextadaptiven Systemen zur Auswertung überlassen. Damit ergibt sich aber auch das Potential des missbräuchlichen Zugriffs auf die Kontextinformationen durch Dritte. Aus der Informationstechnologie können Verfahren zur Sicherung der Zugangswege, z.B. mittels Verschlüsselung oder die Verwendung einer Firewall genutzt werden. Der Besitzer der Kontextinformationen muss die Kontrolle darüber behalten, wer und zu welchem Zweck Zugriff auf Teile des Kontextmodells bekommen kann. In der Forschung sind dazu einige Ansätze veröffentlicht worden [52] [53] [54], die jedoch noch nicht ausreichend sind. Hier ist noch weiterer Forschungsbedarf erkennbar. Diese werden jedoch nachfolgend nicht weiter betrachtet.

2.2.8 Unsicherheit der Kontexterkennung

Das Verhalten eines kontextadaptiven Dienstes basiert auf der Kontexterkennung mit Hilfe von Sensorik und Ableitung von Kontextinformationen auf der Basis von anwendungsspezifischem Regelwissen. Eine solche Erkennung und Ableitung ist aber nicht immer zutreffend und kann zu unerwünschtem oder sogar schädigendem Verhalten eines Dienstes führen. Ursachen für fehlerhafte Kontextinformationen werden in [48] beschrieben:

- Die Kontextinformation ist nicht bekannt, da entsprechende Sensoren fehlen oder aktuell nicht verfügbar sind.
- Sensoren liefern zu einem Kontextattribut widersprüchliche Kontextinformationen. Dies kann in der Fehlfunktion eines Sensors begründet sein.

- Eine Kontextinformation kann von der Sensorik nicht in der geforderten Genauigkeit geliefert werden. Dieses betrifft die Abtastrate, aber auch die Auflösung.

- Die von einem Sensor gemeldete Kontextinformation ist fehlerhaft.

Darüber hinaus können Fehler bei der Ableitung von weiteren Kontextinformationen entstehen, wenn die zugrundeliegenden Kontextinformationen bereits fehlerhaft sind beziehungsweise das Regelwissen unvollständig oder ebenfalls fehlerhaft ist.

Die Handhabung der Unsicherheit der Kontexterkennung ist noch ein offenes Problem. In [104] wird ein Ansatz beschrieben, in welchem eine Kontextontologie mit einem Bayesschen Netzwerk kombiniert wird. Das probabilistische Modell repräsentiert dabei die Unsicherheit in dem Kontextmodell. Mit diesem Ansatz lassen sich die theoretischen Unsicherheiten modellieren und bei der Entscheidungsfindung einsetzen. Jedoch können Probleme mit Sensoren auftreten, die in dem Modell nicht berücksichtigt sind. In [105] wird experimentell nachgewiesen, dass die Kommunikation über den Grad der Unsicherheit der Kontexterkennung zum Nutzer hin zur Verbesserung der Dienstqualität und der Nutzung von kontextadaptiven Anwendungen führen kann. Die Einbindung Betroffener kann aber nicht immer die Lösung für das Problem sein. Explizites Feedback und Bestätigung durch den Anwender führt zu hohem Aufwand und kann die Nutzbarkeit eines Dienstes beeinträchtigen. Vielfach ist er auch nicht in der Lage, die Auswirkungen der Unsicherheit korrekt einzuschätzen.

In dieser Arbeit findet sich dieser Aspekt in der Beschreibung von Qualitätsinformationen zu einzelnen Kontextinformationen wieder. Es werden sowohl Anforderungen an die Qualität aus Sicht der Anwendung beschrieben als auch die verfügbare Qualität aus Sicht der Kontextinfrastruktur. Hierüber wird ein Abgleich durchgeführt, ob die Anforderungen erfüllt werden können. Die tatsächliche Bewertung der Qualität einer vom Sensor gelieferten Kontextinformation wird hier jedoch nicht behandelt.

2.3 Zusammenfassung

Ambient Assisted Living ist ein anwendungsgetriebenes Forschungsfeld, welches eine Vielzahl von einzelnen Lösungsbausteinen beinhaltet. Die Forschungsfelder „Ambient Intelligence“ und „Home Automation“ sind Teile dieser Lösungsbausteine. Im „Home Automation“ spielt die Vernetzung von Sensorik und Aktorik sowie die intelligente Steuerung dieser Komponenten für definierte Einsatzszenarien eine wichtige Rolle. Es ist bereits eine Vielzahl von Sensoren verfügbar und es ist abzusehen, dass sich ihre Menge und Komplexität noch steigern werden. Das „Ambient Intelligence“ befasst sich mit der Bereitstellung einer intelligenten Umgebung, in der Computer als steuernde Intelligenz in den

Hintergrund treten und für den Benutzer ganz verschwinden. Hierfür müssen möglichst intuitive Schnittstellen geschaffen werden, durch die der Benutzer mit der intelligenten Umgebung in Kontakt tritt. „Context Awareness“ ist eine der Technologien, die in diesem Zusammenhang betrachtet wird. Die Nutzung von Sensoren zur Erfassung von Situationen im Umfeld des Bewohners kann dazu verwendet werden, um die intelligente Umgebung selbständig reagieren zu lassen. Eine Grundlage zur formalen Erfassung von Situationsinformationen ist das Kontextmodell. Es gibt bereits eine Reihe von unterschiedlichen Ansätzen zur Kontextmodellierung. Implizite oder explizite Grundlage dieser Kontextmodelle sind Metamodelle. Viele von ihnen sind einfach und generischer Natur. Daneben sind aber auch komplexe und domänenspezifische Metamodelle für die Kontextmodellierung zu finden. Ihre gemeinsamen Modellelemente sind „Context Entity“, „Context Relation“ und „Context Attribute“. Wichtige Fragestellungen bei der Kontextmodellierung sind der Zeitbezug, die Sicherheit von Kontextinformationen und die Wahrung der Privatsphäre sowie die Unzuverlässigkeit der Kontexterkennung. Auf dieser technischen Basis werden kontextadaptive Dienste entwickelt, die das selbstbestimmte Leben einer älteren Person in seiner vertrauten Umgebung unterstützen sollen. Hier können unterschiedliche Kategorien von AAL-Diensten identifiziert werden, z.B. „Sicherheit“, „Gesundheit“ oder „Comfort“. Es beschäftigen sich eine Reihe von Forschungsgruppen mit der Realisierung und Erprobung solcher Dienste in entsprechenden Labors.

Es sind somit eine Vielzahl von Konzepten und Arbeiten bereits vorhanden, die als Ausgangspunkt der Betrachtung dienen.

3 Anforderungen an die Kontextmodellierung aus dem Ambient Assisted Living

Der Dissertation liegt die Annahme zugrunde, dass in der Anwendungsdomäne „Ambient Assisted Living“ Anforderungen an die Kontextmodellierung existieren, die in aktuellen Modellierungsansätzen nicht berücksichtigt sind. In diesem Kapitel werden daher die spezifischen Anforderungen erarbeitet, indem die Entwicklung und Nutzung kontextadaptiver AAL-Dienste aus vier unterschiedlichen Blickwinkeln betrachtet wird.

Zunächst wird die notwendige Infrastruktur der häuslichen Umgebung des Bewohners betrachtet. Sie ist die Grundlage für den Betrieb und die Nutzung eines kontextadaptiven AAL-Dienstes und besteht aus der Vernetzung, der Hardware sowie Software. Dabei spielt insbesondere die Ausstattung der Infrastruktur mit Sensorik eine wesentliche Rolle.

Anschließend betrachten wir die unterschiedlichen AAL-Dienste und deren Eigenschaften im Detail. Es wird untersucht, was die Kontextadaptivität eines AAL-Dienstes ausmacht, ob sich standardisierte AAL-Dienste identifizieren lassen und welche Auswirkungen sie auf die Kontextmodellierug haben.

Dann ist die Nutzung von AAL-Diensten durch den Bewohner Gegenstand der Betrachtung. Es werden Nutzungsfälle in denen der Bewohner mit seiner häuslichen Umgebung interagiert, unter dem Aspekt der Kontextadaptivität untersucht. Hieraus ergeben sich Anforderungen an die Kontextmodellierung aus Sicht der Benutzbarkeit und Akzeptanz.

Abschließend betrachten wir die Phasen der Entwicklung und des Betriebs eines kontextadaptiven AAL-Dienstes. Dabei wird die Schaffung einer Methodik zur Entwicklung solcher Dienste angestrebt. Deshalb werden die einzelnen Phasen und die darin zu identifizierenden Anforderungen genauer betrachtet.

Diese Erhebungsmethode wird in nachfolgender Abbildung visualisiert. Zunächst wird anhand eines Szenarios beispielhaft die zugrundeliegenden Annahmen über die Nutzung und Bereitstellung von kontextadaptiven AAL-Diensten vorgestellt. Die detaillierte Erhebung der Anforderungen geschieht in den dann folgenden Unterkapiteln. Abschließend werden dann die relevanten Anforderungen zusammengefasst, die sich aus der Betrachtung der Anwendungsdomäne ergeben haben.

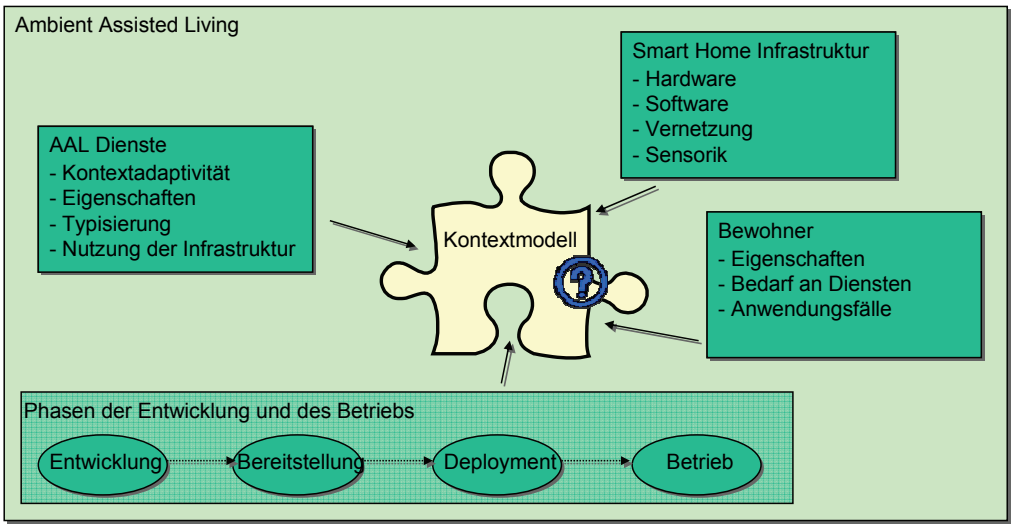


Abbildung 4: Blickwinkel der Anforderungsanalyse

3.1 Szenario

Anhand dieses Szenarios wird exemplarisch die Entwicklung und Nutzung von AAL-Diensten beschrieben und die ihr zugrunde liegende Zielstellung verdeutlicht. Teil des Szenarios ist die Vorstellung, dass Diensteentwicklern neue kontextadaptive AAL-Dienste entwickeln und in einem offenen Dienstemarkt potentiellen Anwendern zur Verfügung stellen. Der Bewohner einer intelligenten Umgebung besitzt über einen solchen Dienstemarkt die Möglichkeit, sich eine an seine Lebenssituation angepasste Menge an Diensten zusammenzustellen und zu erwerben. Die zugrundeliegende Infrastruktur stellt die Übernahme und den Betrieb dieser Dienste sicher. Diese Vorstellung soll nachfolgend an einem Beispiel verdeutlicht werden.

Die Firma „AAL Vision“ ist spezialisiert auf die Realisierung kontextadaptiver AAL-Dienste. Durch eine Markterhebung der Firma wurde erkannt, dass es einen Bedarf für einen Dienst gibt, der die Bewohner an die Einnahme von Medikamenten erinnert. Das Produkt „Medikamentendienst“ wird definiert und entlang eines Software-Entwicklungsprozesses in die Realisierung überführt. Dieser umfasst das Design, die Implementierung und den Test. Nach einer erfolgreichen Testphase wird das Produkt für den Vertrieb freigegeben. Für ihn spielt der AAL-Dienstemarkt eine wichtige Rolle. Er enthält die Möglichkeit, seine Dienste zu veröffentlichen und damit den potentiellen Nutzern bekannt zu machen. Der Medikamentendienst der Firma wird auf diese Plattform geladen und über eine detaillierte Funktionsbeschreibung sowie der Darstellung der Nutzungsvoraussetzungen bekannt gemacht.

Mr. Bond ist ein solcher Kunde, der nun älter geworden und in eine Wohnung mit einer intelligenten, d.h. mit Kontextsensoren ausgerüsteten Umgebung eingezogen ist. Es sind folgende Sensoren vorhanden:

- Zeit
- Ortung innerhalb der Wohnung
- Blutdruck
- Allgemeine Erkennung von Bewegung innerhalb der Wohnung
- Zustand verschiedener Gegenstände, z.B. Heizung, Fenster, Tür, Ofen

Die initiale Menge kontextadaptiver Dienste besteht aus einem Wohnungs-, einem Audio Entertainment- und einem Vitalwertedienst. Diese sind in einem speziellen Wohnungsportal für Mr. Bond zugreifbar.

- Der Wohnungsdienst ermöglicht es, die Ausstattung der Wohnung auf der Basis verschiedene Szenarien anzusteuern. Beispielsweise kann entsprechend der Tageszeit die Beleuchtung und die Heizungsanlage automatisch eingeschaltet werden.

- Der Audio Entertainment Dienst ermöglicht die Beschallung der Wohnung mit Musik in Abhängigkeit von Zeit und Ort. Einfache aktivitätsbezogene Bedingungen werden ebenfalls zur Auswahl angeboten. Sie können von existierenden Kontextinformationen abgeleitet werden, wie z.B. „Bewohner kocht“. Mr. Bond entscheidet sich für klassische Musik beim Kochen. Weiterhin stellt er eine entspannende Geräuschkulisse für das Badezimmer ein.

- Der Dienst zur Überwachung seiner Vitalwerte ist bereits mit den Kontaktdaten seines Hausarztes parametrisiert. Eine Anzahl unterschiedlicher Werte könnte durch den Dienst überwacht werden, jedoch ist lediglich die Überwachung des Blutdrucks aufgrund der beschränkten Infrastruktur aktivierbar.

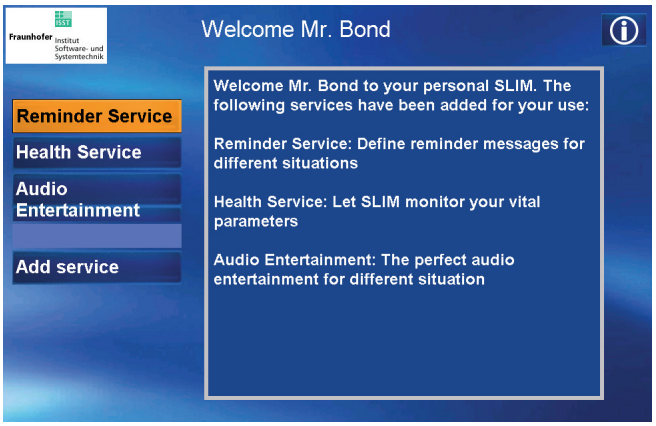


Abbildung 5: Übersicht der initialen Dienstemenge für Mr. Bond

Mr. Bond staunt über die vielfältigen Möglichkeiten, die ihm die AAL-Dienste bieten, möchte aber gerne einen Dienst hinzufügen, der ihn an die Einnahme seiner Medikamente erinnert. Über die zentrale Steuereinheit der Wohnung nutzt er erfolgreich die Möglichkeit, nach weiteren Diensten zu suchen und diese in die intelligente Umgebung aufzunehmen. Mr. Bond werden drei AAL-Dienste angeboten, die ihn an die Einnahme erinnern können. Das Angebot von Firma „AAL Vision“ fällt noch mit einer Zusatzfunktion zur Überwachung der Medikamente und automatischen Nachbestellung auf. Das Portal teilt ihm aber mit, dass die Erinnerung an die Medikamenteneinnahme leider nur eingeschränkt erfolgen kann, da einige notwendige Sensoren zur Erkennung der tatsächlichen Einnahmenotwendigkeit fehlen. Aus diesem Grund kann nur eine zeitbasierte Erinnerung unterstützt werden. Mr. Bond wählt trotz dieser Einschränkung den Dienst aus und übernimmt diesen in seine Dienstemenge. Anschließend setzt er den Zeitpunkt der Erinnerung auf 20:00 Uhr fest. Er würde gerne seine Infrastruktur um die notwendige Sensorik zur Überwachung der Einnahme erweitern. Die zentrale Steuereinheit schlägt daher die Erweiterung um folgenden Sensor vor: „Überwachung der Aktivitäten am Medizinschrank“.

Mr. Bond beauftragt nun einen Dienstleister mit der Erweiterung seiner intelligenten Umgebung um den identifizierten Sensor. Der Provider integriert ihn in die Infrastruktur der intelligenten häuslichen Umgebung. Nach erfolgter Erweiterung macht Mr. Bond sich erneut daran, seine Dienste zu konfigurieren. Nun sind die von ihm gewünschten Kontextbedingungen verfügbar. Er ist zufrieden und genießt die elektronische Unterstützung seiner intelligenten Wohnung.

3.2 Anforderungen aus Sicht einer Smart Home Infrastruktur

Eines der Blickwinkel dieser Anforderungsanalyse ist die Betrachtung der Smart Home Infrastruktur. Die Infrastruktur einer intelligenten Umgebung im AAL zeichnet sich durch eine hohe Komplexität auf verschiedenen Ebenen aus. Auf der unteren Ebene ist die Vernetzung ein wichtiger Bestandteil der Infrastruktur. Auf diesem kann dann die Hardwarelandschaft, bestehend aus Sensorik und Aktorik sowie Rechnerknoten und weiteren Geräten, aufgesetzt werden. In der obersten Ebene erfolgt dann die Verwaltung und Bereitstellung von Software bzw. Basisdiensten, die von den kontextadaptiven Diensten genutzt werden können. In diesen drei aufeinander aufgebauten Bereichen lassen sich Anforderungen an die Kontextmodellierung identifizieren. Die Sensorik als Teil der Infrastruktur wird anschließend separat betrachtet, da diese eine besondere Bedeutung für die Kontextadaptivität der Dienste besitzt.

3.2.1 Vernetzung

Eine grundlegende Problematik im „Home Automation“ ist die Bereitstellung einer Netzinfrastruktur zur Einbindung von Sensorik und Aktorik diverser Art. Hier müssen eventuell unterschiedlichste Netzwerke und Protokolle miteinander verbunden werden. Die Verwendung eines Mediators auf der Ebene des Netzwerks kann hier ein Lösungsansatz sein. Ein aktueller Ansatz besteht hier in der Verwendung der OSGi Technologie [30] in einem Residential Gateway. Ein OSGi-Gateway stellt die Verbindung der Wohnung an die Außenwelt bereit, aber auch die Integration unterschiedlicher Netze und Bussysteme innerhalb der Wohnung. Auf diese Weise können Softwarekomponenten auf dem Gateway mit den verschiedenen Komponenten innerhalb der Wohnung ungeachtet der unterschiedlichen Netzwerke und Protokolle kommunizieren.

Aus Sicht des „Ambient Intelligence“ ist ein solches Netzwerk hoch dynamisch. Sowohl die Verbindung im Netzwerk, z.B. mittels WLAN [106], Bluetooth [107] und IR als auch darüber erreichbare Rechnerknoten können dynamisch verfügbar sein. In [108] wird die Diversität und Skalierung von Netzwerken im „Ambient Intelligence“ untersucht. Dort werden die Trends in 3 unterschiedlichen Netzwerktypen analysiert: Kabelbasierte Netzwerke, Funknetze und Sensornetze. Diese Netztypen unterscheiden sich in folgenden Aspekten: Bandbreite, Verfügbarkeit, unterstützte Protokolle und Sicherheit. Alle diese Aspekte können für die Kontextadaptivität von Diensten im AAL eine Rolle spielen, wie nachfolgende konkrete Beispiele zeigen werden.

Für einen AAL-Dienst kann die verfügbare Bandbreite des Netzwerks eine wichtige Kontextinformation sein. Beispielsweise gilt dieses für einen Infotainmentdienst, der zur Darstellung multimedialer Informationen die einsetzbare Bandbreite zum Präsentationsgerät berücksichtigen muss, um die Form der Darstellung entsprechend anzupassen. Die Auswahl eines geeigneten Endge-

räts zur Präsentation kann auch von der zuverlässigen Verfügbarkeit der Netzwerkverbindung abhängen. Ebenso können die auf dem Netzwerk möglichen Protokolle und die Datensicherheit über die Auswahl geeigneter Netzknoten entscheiden.

Hieraus folgt, dass die Abbildung der Netztopologie und der Eigenschaften von Teilstrecken auf dem Netz, bezogen auf die Bandbreite, Verfügbarkeit, unterstützte Protokolle sowie Sicherheit Bestandteil eines konkreten Kontextmodells sein muss. Zudem muss die Abbildung der Netztopologie in ein Ortsmodell Bestandteil des Kontextmodells sein, um beispielsweise daraus ableiten zu können, welche Bandbreite zum Display im Badezimmer zur Verfügung steht. Eine Übersicht über die Anforderungen an das konkrete Kontextmodell ist in nachfolgender Tabelle gegeben.

Tabelle 3: Anforderungen ein konkretes Kontextmodell aus Sicht der Vernetzung

AK1	Abbildung der Netztopologie
AK1.1	Bandbreite
AK1.2	Verfügbarkeit
AK1.3	Unterstützte Protokolle
AK1.4	Sicherheit
AK1.5	Verknüpfung mit Ortsmodell

Eine Netztopologie kann als Aufzählung von Teilnetzen und deren Verbindung beschrieben werden. Für die Modellierung ergibt sich daraus, dass das Modellelement „Kontextentität“ benötigt wird, um ein Teilnetz im Kontextmodell abbilden zu können. Zudem wird das Modellelement „Kontextrelation“ benötigt, um die Verbindung zwischen Teilnetzen ausdrücken zu können. Letztendlich wird das Modellelement „Kontextattribut“ benötigt, um die Eigenschaften eines Teilnetzes in Bezug auf Bandbreite, Verfügbarkeit, unterstützte Protokolle und Sicherheit zu definieren. Ein Ortsmodell kann wahlweise explizit als eigene Kontextentität und die Verknüpfung über eine Kontextrelation erfolgen, oder auch implizit durch ein Kontextattribut dessen Wertemenge das Ortsmodell repräsentiert. In nachfolgender Tabelle wird eine Übersicht über die Anforderungen an die Kontextmodellierung aus Sicht der Vernetzung gegeben.

Tabelle 4: Anforderungen an die Kontextmodellierung aus Sicht der Vernetzung

AM1	Benötigte Modellelemente zur Beschreibung der Vernetzung	
AM1.1		Verwendung des Modellelements „Kontextentität“ zur Definition von Teilnetzen
AM1.2		Verwendung des Modellelements „Kontextrelation“ zur Definition von Beziehungen zwischen Teilnetzen
AM1.3		Verwendung des Modellelements „Kontextattribut“ zur Definition von Eigenschaften von Teilnetzen

3.2.2 Hardware

In [22] wird eine Hardwaretopologie für das AAL vorgeschlagen, welche Knoten auf den Netzwerken in folgende 4 Gruppen unterteilt: „Fixed Nodes Group“, „Portable Nodes Group“, „Sensor / Actuator Nodes Group“ und „Device Group“. Diese Gruppen unterscheiden sich in folgenden Eigenschaften: Rechenkapazität, Kommunikationsbandbreite und Mobilität.

„Fixed Nodes“ bilden die feste Infrastruktur einer Umgebung im Ambient Intelligence. Diese Knoten besitzen eine hohe Rechenkapazität und Kommunikationsbandbreite, sind aber aufgrund der notwendigen Stromversorgung oder ihrer Anbindung an das Netzwerk nicht mobil. Beispiele hierfür sind lokale Rechner oder Server, auf denen Dienste bereitgestellt werden können, die performant und zuverlässig in einer solchen Umgebung verfügbar sein müssen. Dazu kann beispielsweise ein Kontextserver im Rahmen einer Infrastruktur für kontextadaptive Anwendungen gehören.

„Portable Nodes“ sind mobile und nicht immer verfügbare Knoten innerhalb der Umgebung im „Ambient Intelligence“. Diese können ebenfalls eine hohe Rechenkapazität und Kommunikationsbandbreite besitzen, die aber zumeist nicht so hoch sein wird wie die der „Fixed Nodes“. Aufgrund ihrer Mobilität können diese Knoten während der Laufzeit innerhalb der Infrastruktur verschwinden und nach einer Zeit an anderer Stelle wieder auftauchen. Somit ist die Verfügbarkeit dieser Resource nicht immer garantiert, was schon an einer leeren Batterie liegen kann. Beispiele für diese Knoten sind Laptops oder PDAs (Personal Digital Assistant).

Im „Ambient Intelligence“ kann ein AAL-Dienst zwischen fixen und portablen Knoten migrieren und dadurch beispielsweise dem Nutzer folgen. Das kontextadaptive Verhalten des Dienstes basiert dann auf der Auswahl der geeigneten Knoten. Hierfür ist eine Beschreibung der Eigenschaften des Knotens in

Bezug auf den Prozessortyp, die Rechenkapazität, die Anbindung an das Netzwerk und die daraus resultierende Bandbreite, die Speicherkapazität und die Stromversorgung bzw. Batteriestand interessant. Auch die räumliche Zuordnung ist für die Auswahl von Knoten relevant.

„Sensor / Actuator Nodes“ sind Knoten mit sehr eingeschränkter Rechenkapazität. Diese haben fest definierte Aufgaben und können nur in diesem Rahmen eingesetzt werden. Beispielsweise dienen sie zur Analyse von Sensordaten oder zur Steuerung einer Aktorik. Sie sind sowohl stationär als auch mobil mit entsprechenden Bandbreiten einsetzbar. Für die Kontextmodellierung kann eine räumliche Zuordnung der Knoten wichtig sein, um zu ermitteln, für welchen Raum eine entsprechende Sensorik oder Aktorik verfügbar ist. Daher ist neben einer Georeferenzierung auch ein symbolisches Ortsmodell notwendig. Sensoren werden in einem nachfolgenden Kapitel detaillierter betrachtet.

„Device Nodes“ sind Knoten ohne eigene Rechenkapazität. Sie können einfache Kontextsensoren sein, die in Sensorknoten eingebunden sind. Es können aber auch Nutzerschnittstellen sein, z.B. für Monitore oder mittels Bluetooth angebundene Kopfhörer. „Device Nodes“ existieren nicht für sich alleine, sondern werden über eine der oben angeführten Knoten in die Infrastruktur eingebunden. Die Beziehung zwischen diesen Knoten kann dynamisch, also ad hoc veränderbar sein. Nutzerschnittstellen differieren in Bezug auf ihre Eingabe- und Ausgabemöglichkeiten. Ein Unterscheidungsmerkmal ist dabei die unterstützte Modalität der Interaktion. Beispiele hierfür sind Sprachein- und -ausgabe, Gestik oder Haptik. Die Unterstützung der Interaktionsmodalität erfolgt auf der Basis der multimedialen Eigenschaften der Knoten, die sich ebenfalls unterscheiden können, wie es sich z. B. in der farblichen und räumlichen Auflösungsmöglichkeit visueller Medien zeigt.

Die Auswahl einer geeigneten Nutzerschnittstelle in der Nähe des Bewohners kann eine kontextadaptive Eigenschaft eines AAL-Dienstes sein. Die Beschreibung der Beziehung sowie der Eigenschaften der Device Nodes ist möglicherweise relevant für die Auswahl eines geeigneten Knotens für einen AAL-Dienst und sollte deshalb in einem Kontextmodell dargestellt werden können. Nachfolgend werden in einer Übersicht die Anforderungen an ein konkretes Kontextmodell aus Sicht der Hardware gezeigt.

Tabelle 5: Anforderungen an ein konkretes Kontextmodell aus Sicht der Hardware

AK2	Abbildung der Knotentopologie
AK2.1	Abbildung von „Fixed Nodes“

AK2.1.1			Rechenkapazität
AK2.1.2			Art der Vernetzung
AK2.1.3			Speicherkapazität
AK2.1.4			Verknüpfung mit Ortsmodell
AK2.2		Abbildung von „Portable Nodes“	
AK2.2.1			Rechenkapazität
AK2.2.2			Art der Vernetzung
AK2.2.3			Speicherkapazität
AK2.2.4			Verknüpfung mit Ortsmodell
AK2.2.5			Verfügbarkeit
AK2.2.6			Stromversorgung
AK2.3		Abbildung von „Sensor / Actuator Nodes“	
AK2.3.1			Verknüpfung mit Ortsmodell, Georeferenzierung und symbolisches Raummodell
AK2.4		Abbildung von „Device Nodes“	
AK2.4.1			Zuordnung zu aktiven Knoten
AK2.4.2			Eigenschaften der Interaktionsschnittstelle

Für die Modellierung ergibt sich hieraus, dass das Modellelement „Kontextentität“ benötigt wird, um „Fixed Nodes“, „Portable Nodes“, „Sensor / Actuator Nodes“ und „Device Nodes“ zu definieren. Da „Portable Nodes“ sich von „Fixed Nodes“ lediglich durch die benötigte Stromversorgung und die sich ändernde Verfügbarkeit unterscheiden, kann es sinnvoll sein die gemeinsamen Eigenschaften durch eine Generalisierungsbeziehung zwischen den Kontextentitäten zu beschreiben. Eine Eigenschaft eines Knotens kann durch das Modellelement „Kontextattribut“ definiert werden. Die Vernetzung kann durch eine Beziehung zu dem jeweiligen Teilnetz beschrieben werden. Hierfür wird das Modellelement „Kontextrelation“ benötigt. In AK2.3.1 wird die Verknüpfung eines Sensors mit dem Ortsmodell als eine Anforderung beschrieben. Ein Orts-

modell sollte sowohl georeferenziert als auch in Form eines symbolischen Raummodells vorhanden sein. Daraus ergibt sich die Anforderung an die Unterstützung von verschiedenen Repräsentationsformen zu beispielsweise einem Ortsmodell. In nachfolgender Tabelle wird eine Übersicht über die Anforderungen an die Kontextmodellierung aus Sicht der Hardware gegeben.

Tabelle 6: Anforderungen an die Kontextmodellierung aus Sicht der Hardware

AM2	Benötigte Modellelemente zur Beschreibung der Knoten
AM2.1	Verwendung des Modellelements „Kontextentität“ zur Definition von „Fixed Nodes“, „Portable Nodes“, „Sensor / Actuator Nodes“ und „Device Nodes“
AM2.2	Definition einer Generalisierungsbeziehung eines Modellelements „Kontextentität“ zur Beschreibung von gemeinsamen Eigenschaften eines „Fixed Nodes“ und eines „Portable Nodes“
AM2.3	Verwendung des Modellelements „Kontextrelation“ zur Definition von Beziehungen von Knoten zu Teilnetzen
AM2.4	Verwendung des Modellelements „Kontextattribut“ zur Definition von Eigenschaften von Knoten
AM2.5	Definition von Repräsentationsformen

3.2.3 Software

Sowohl die „Fixed Nodes“ als auch die „Portable Nodes“ sind potentielle Laufzeitumgebungen für die Bereitstellung von Diensten im „Ambient Intelligence“. Sie sollen sich in dieser Umgebung bewegen können und beispielsweise dem Nutzer folgen. Dieses ist eine kontextadaptive Funktionalität solcher AAL-Dienste. Eine Vorraussetzung hierfür sind die Anforderungen an die Beschreibung der Hardware, die bereits zuvor identifiziert wurden. Daneben sind aber auch Eigenschaften der darauf verfügbaren Software zu beschreiben, z.B. Betriebssystem, Laufzeitumgebung, Bibliotheken sowie lokale Dienste. So kann entschieden werden, ob ein Dienst auf dem Knoten lauffähig ist, bzw. für welche Umgebungsparameter der Dienst generiert werden muss [109].

Ein Betriebssystem abstrahiert von der zugrundeliegenden Hardware und stellt Basisfunktionen zur Nutzung von Speicher, Ein- und Ausgabegeräten sowie zur Steuerung der Software bereit. Der Zugriff auf notwendige Ressourcen wird vom Betriebssystem über entsprechende Bibliotheken zur Verfügung ge-

stellt. Dienste können die bereitgestellten Schnittstellen nutzen, und auf diese Funktionen zuzugreifen. Im Ambient Intelligence ist von einer Heterogenität der Knoten und den zugehörigen Betriebssystemen auszugehen. Für Fixed Nodes mit einer hohen Rechenleistung können Serverbetriebssysteme, z.B. auf Basis von LINUX oder Windows zum Einsatz kommen. Für leistungsschwächere Fixed und Portable Nodes können spezialisierte Betriebssysteme gewählt werden, z.B. TinyOS [110]. Spezielle Knoten, die zur Interaktion mit dem Bewohner fähig sind, z.B. PDA, Smartphone, Tablet PC oder Set-Top-Box, können ebenfalls eigene Betriebssysteme besitzen, z.B. „Symbian OS“, „Palm OS“ oder „Windows CE“. Sollen Dienste auf diesen Nodes bewegt bzw. generiert werden, so sind Informationen über das bereitgestellte Betriebssystem, dessen Version sowie die darin enthaltenen Treiber notwendig. Daneben werden auch Informationen zu Bibliotheken gebraucht, die auf dem Node vorhanden sind.

Eine virtuelle Maschine als Laufzeitumgebung für einen Dienst kann von der konkreten Hardware und Betriebssystem abstrahieren. Er wird dabei in einer Programmiersprache erstellt, die nicht direkt in die Maschinensprache der CPU übersetzt wird. Vielmehr wird diese in einem Zwischencode bereitgestellt, welcher dann auf dem Zielsystem durch einen Interpreter ausgeführt wird. Dadurch ist dieser plattformunabhängig und kann auf Nodes mit unterschiedlichen Eigenschaften ausgeführt werden. Beispiele für diesen Ansatz sind in der .NET-Architektur von Microsoft oder der virtuellen Java-Maschine von SUN zu finden. Sowohl das .NET- als auch das Java-Framework sind mittlerweile in unterschiedlichen Ausprägungen und Versionen zu finden. In Java wird beispielsweise zwischen den Varianten ME (Micro Edition), SE (Standard Edition) und EE (Enterprise Edition) unterschieden. Diese Differenzierung liegt im Wesentlichen in den Bibliotheken begründet, die Bestandteil des Frameworks sind. Entsprechende Informationen müssen vorgehalten werden, um zu entscheiden, ob ein Dienst in der virtuellen Maschine auf dem ausgewählten Knoten lauffähig ist bzw. generiert werden kann.

Weiterhin werden Informationen zu Zusatzdiensten der Infrastruktur benötigt, die lokal auf dem Knoten vorhanden oder von diesem erreichbar sind. Eine Middleware ermöglicht beispielsweise die Kommunikation und Koordination mit weiteren Diensten innerhalb der Infrastruktur. Ein Beispiel hierfür ist der Java EE-Container. Weitere Beispiele für Dienste einer AAL-Infrastruktur sind der Naming und Discovery Service [111], der Semantic Match Service [112], der Identification and Authentication Service, das User Management und „Authentication, Authorization, Accounting“ (AAA) [113], sowie eine Rendering Engine [114]. Diese Aufzählung ist nicht vollständig. Es sind Informationen über den Zugang zu diesen Zusatzdiensten als Teil des Kontextmodells vorzuhalten.

Tabelle 7: Anforderungen an ein konkretes Kontextmodell aus Sicht der Software

AK3	Software auf dem Knoten
AK3.1	Betriebssystem
AK3.1.1	Version
AK3.1.2	Treiber
AK3.1.3	Bibliotheken
AK3.2	Virtuelle Maschine
AK3.2.1	Version
AK3.2.2	Bibliotheken
AK3.3	Zusatzdienste
AK3.3.1	Version
AK3.3.2	Zugriffsinformationen

Für die Modellierung ergibt sich hieraus, dass das Modellelement „Kontextentität“ benötigt wird, um „Betriebssystem“, „Virtuelle Maschine“ und „Zusatzdienste“ zu definieren. Diese besitzen gemeinsame Eigenschaften die durch eine Generalisierungsbeziehung zwischen den Kontextentitäten beschrieben werden können. Deren Eigenschaften können durch das Modellelement „Kontextattribut“ definiert werden. Die Zuordnung der Software zu einem Knoten erfolgt durch das Modellelement „Kontextrelation“.

Tabelle 8: Anforderungen an die Kontextmodellierung aus Sicht der Software

AM3	Benötigte Modellelemente zur Beschreibung der Software
AM3.1	Verwendung des Modellelements „Kontextentität“ zur Definition von „Betriebssystem“, „Virtuelle Maschine“, und „Zusatzdienste“
AM3.2	Definition einer Generalisierungsbeziehung eines Modellelements „Kontextentität“ zur Beschreibung von gemeinsamen Eigenschaften von „Betriebssystem“, „Virtuelle Maschine“ und „Zusatzdienste“

AM3.3		Verwendung des Modellelements „Kontextrelation“ zur Definition von Beziehungen von Software zu Knoten
AM3.4		Verwendung des Modellelements „Kontextattribut“ zur Definition von Eigenschaften der Software

3.2.4 Sensorik

Die Sensorik ist ein wesentlicher Bestandteil der Infrastruktur einer intelligenten Umgebung. Sensoren liefern Kontextinformationen, die für die Kontextadaptivität von AAL-Diensten notwendig sind. Somit sind diese eine wesentliche Komponente der Infrastruktur einer intelligenten häuslichen Umgebung für die Realisierung kontextadaptiver AAL-Dienste. Wie bereits in Kap. 2.1.2.1 beschrieben sind aktuell eine Vielzahl von Sensoren verfügbar, mit denen unterschiedliche Informationen zur Situation des Bewohners und seiner intelligenten Umgebung erfasst werden können. Ein Großteil von ihnen ist jedoch für Spezialanwendungen entwickelt und daher noch recht teuer. Viele der sonst noch in der Literatur beschriebenen Sensoren sind derzeit nur in der Forschung vorhanden [41], [115], [116]. Es gibt derzeit noch keinen definierten Standard an Sensoren, die in einer intelligenten häuslichen Umgebung vorhanden sein müssen. Es kann also durch ein Kontextmodell keine feste Menge an Sensorinformationen vorgegeben werden. In der Nomenklatur der Messe „Sensor + Test 2008“ [117] sind insgesamt 160 Kategorien von Kontextsensoren eingetragen. Eine Vielzahl von diesen ist im häuslichen Umfeld nicht relevant. Es muss jedoch davon ausgegangen werden, dass mit der Entwicklung der Sensorik noch weitere Kategorien an Bedeutung gewinnen werden. Als Anforderung daraus ergibt sich an die Kontextmodellierung, dass nicht nur neue Sensoren eines bekannten Typs, sondern auch neue Arten von Kontextsensoren in eine bestehende intelligente häusliche Umgebung eingebunden werden müssen. Daraus folgt die Anforderung zur Erweiterbarkeit des Kontextmodells in Hinblick auf die Integration neuer Sensoren.

Die Auswahl der geeigneten Sensorik zur Bereitstellung der benötigten Kontextinformationen kann als eine kontextadaptive Funktionalität der Infrastruktur der intelligenten häuslichen Umgebung angesehen werden. Sie muss in Abhängigkeit der verfügbaren Sensorik und deren Eigenschaften, sowie den Nutzungsmodalitäten erfolgen. In [35] wird speziell auf die Grundlagen und Eigenschaften von Kontextsensoren eingegangen und grundlegende Anforderungen an ihr Design erhoben. Einige von ihnen können unter dem Gesichtspunkt der Auswahl geeigneter Sensoren auch auf die Kontextmodellierung übertragen werden.

- Robustheit und Verlässlichkeit: Es kann nicht immer davon ausgegangen werden, dass Sensorik dauerhaft verfügbar ist und dass die gelieferten Informationen verlässlich sind. So kann die Stromversorgung eines Sensors abbrechen oder Umwelteinflüsse die Sensorik stören, weshalb die Robustheit und Verlässlichkeit eines Sensors beschrieben werden muss.

- Portabilität, Größe und Gewicht: Aus der Portabilität von Sensoren erfolgt, dass diese nicht immer an Ort oder Person gebunden sind. So kann ein Mobiltelefon als Kontextsensor zur Ortung einer Person verwendet werden. In der Regel wird dieses Telefon einem Besitzer zugeordnet sein, jedoch kann es zeitweise von einer zweiten Person genutzt werden. Diese Zuordnung muss als Information verfügbar sein, um die Sensorinformation der richtigen Entität zuordnen zu können. Aus Größe und Gewicht eines Sensors ergeben sich keine Anforderungen.

- Aufdringlichkeit, Soziale Akzeptanz und Nutzerbedenken: Sensorik kann je nach Beschaffenheit in die Privatsphäre des Nutzers eindringen. Eine Videokamera im Badezimmer zur Analyse der Aktivitäten des Bewohners hat in der Regel keine Akzeptanz. Daher wird an Sensorik geforscht, die über weniger aufdringliche Medien Kontextinformationen ableiten können, z.B. über Analyse akustischer Signale [41]. Die Verlässlichkeit dieser Sensorik kann jedoch geringer sein, sodass in begründeten Ausnahmefällen, z.B Notsituationen, die Verwendung einer weniger akzeptierten Sensorik notwendig sein kann. Die Aufdringlichkeit muss daher Bestandteil der Beschreibung eines Sensors sein.

- Genauigkeit und Offenheit: Die Sensorik kann sich in der Qualität der gelieferten Kontextinformationen unterscheiden, wozu die Abtastrate und die Auflösung gehören. Auch sollte die Abdeckung bzw. der Wertebereich eines Sensors Bestandteil des Kontextmodells werden. Der Aspekt der Offenheit spielt für die Kontextadaptivität von AAL-Diensten keine Rolle.

Daneben können noch weitere Eigenschaften bzw. Nutzungsmodalitäten von Sensoren identifiziert werden, die zu den Auswahlkriterien der geeigneten Sensorik gehören.

- Einheit: Zur korrekten Interpretation der von den Sensoren gelieferten Kontextinformationen muss die zugehörige Einheit bekannt sein. So kann beispielsweise die Temperatur in Fahrenheit oder in Celsius gemessen werden.

- Nutzungsmodalität, Kosten: Die Nutzung eines Kontextdienstes kann mit Kosten verbunden sein. Ein Beispiel hierfür ist die Ortung eines Nut-

zers durch einen Mobilfunkanbieter. Die dabei entstehenden Kosten müssen als Bestandteil der Nutzungsmodalität eines Sensors erkennbar sein.

- Nutzungsmodalität, Privacy: Die Nutzung eines Kontextdienstes kann unter dem Gesichtspunkt der Privacy beschränkt sein. So kann der Zugriff auf die Vitalwerte des Bewohners lediglich für eine eingeschränkte Menge von Diensten erlaubt sein. Die Regeln für den Zugriff auf die Kontextinformationen müssen als Bestandteil der Nutzungsmodalität eines Sensors erkennbar sein.

- Art der Bereitstellung von Kontextinformationen: Eine Eigenschaft von Kontextsensoren kann eine aktive Bereitstellung von Kontextinformationen sein. In diesem Fall ist ein solcher Sensor in der Lage, von sich aus relevante Kontextinformationen zu melden. Dagegen besitzen passive Sensoren diese Eigenschaft nicht. Es muss in geeigneten Zeitabständen auf die Sensorik zugegriffen werden, um die verfügbaren Kontextinformationen abzurufen. Die Art der Bereitstellung muss Teil der Beschreibung der Sensoreigenschaften sein.

Die beschriebenen Anforderungen aus Sicht der Sensorik lassen sich an dieser Stelle noch nicht einem konkreten Kontextmodell oder dem Metamodell zuordnen. Im ersteren Fall kann das Modellelement „Kontextentität“ verwendet werden, um einen Sensor zu beschreiben. Dessen Eigenschaften können mit Hilfe des Modellelements „Kontextattribut“ beschrieben werden. Aufgrund der besonderen Bedeutung der Beschreibung der Sensorik für die Fähigkeit einer Kontextinfrastruktur bekannte und Sensoren einzubinden und die bereitgestellten Kontextinformationen einem konkreten Kontextmodell zuzuordnen wird die Sensorik als Modellelement des Metamodells (siehe Kap. 5.3.2) definiert. Entsprechend werden die identifizierten Anforderungen in der nachfolgenden Tabelle dem Metamodell zugeordnet.

Tabelle 9: Anforderungen an die Kontextmodellierung aus Sicht der Sensorik

AM4	Sensorik
AM4.1	Erweiterbarkeit der Sensorik
AM4.1.1	Einbindung bekannter Sensoren in die Infrastruktur
AM4.1.2	Einbindung neuartiger Sensoren in die Infrastruktur
AM4.2	Auswahl von Sensorik

AM4.2.1			Beschreibung der Robustheit und Verlässlichkeit
AM4.2.2			Beschreibung des Bezugs der Kontextinformationen zu einer konkreten Instanz einer Entität
AM4.2.3			Beschreibung der Intrusivität
AM4.2.4			Beschreibung der Qualität der Kontextinformationen, z.B. Genauigkeit
AM4.2.5			Beschreibung der Einheit
AM4.2.6			Beschreibung der Abdeckung
AM4.2.7			Beschreibung der Nutzungsmodalitäten, Kosten
AM4.2.8			Beschreibung der Nutzungsmodalitäten, Privacy
AM4.2.9			Beschreibung der Art der Bereitstellung von Kontextinformationen: Aktiv/Passiv

3.3 Anforderungen aus Sicht der AAL-Dienste

Ein zweiter Blickwinkel dieser Anforderungsanalyse ist die Betrachtung der AAL-Dienste. Ein AAL-Dienst soll die intelligente häusliche Umgebung des Bewohners um Funktionalität erweitern, die den Bewohner unmittelbar in seinen unterschiedlichen Lebenssituationen unterstützt. Ein AAL-Dienst muss nicht unmittelbare kontextadaptive Eigenschaften besitzen, z.B. ein Pizza-Bestelldienst. Daneben können aber AAL-Dienste identifiziert werden, die unmittelbar auf Ereignisse im Umfeld des Bewohners reagieren. Nachfolgend wird daher zunächst das kontextadaptive Verhalten von AAL-Diensten analysiert. Anschließend werden mögliche Arten und deren gemeinsame Anforderungen an die Kontextmodellierung betrachtet.

3.3.1 Kontextadaptivität von AAL-Diensten

In Kap. 2.2.2 wurden die möglichen Arten der Kontextadaptivität von Anwendungen aufgeführt. Daraus wird ersichtlich, dass diese nie die eigentliche Anwendungslogik eines AAL-Dienstes ausmacht. Vielmehr ist sie eine Eigenschaft des Dienstes, die Anwendungslogik aufgrund von Ereignissen aus der Umgebung selbständig auszulösen, bzw. mit Kontextinformationen zu parametrisieren. Als Beispiel sei hierfür ein Erinnerungsdienst gegeben. Dessen

Anwendungslogik ist relativ einfach und besteht aus der Bereitstellung von Erinnerungsnachrichten, z.B. zur Einnahme eines Medikaments. Die Definition und Umsetzung dieser Anwendungslogik ist keine Aufgabe der Kontextmodellierung. Sie erfolgt von ihr unabhängig.

Das Auslösen der Erinnerungsfunktion aufgrund eines Ereignisses, z.B. dem Vergessen der Einnahme, ist dagegen eine Erweiterung dieser Logik um das kontextadaptive Verhalten, das je nach Dienst unterschiedlich sinnvoll ist. Für den Erinnerungsdienst kann eine Einbruchssituation ohne Bedeutung sein, für einen Notfalldienst jedoch relevant. Die Definition solcher Situationen für die unterschiedlichen Dienste ist deshalb Bestandteil der Kontextmodellierung. Sie sind dabei vielfach komplexer Natur und können nicht nur durch einen einzelnen Sensor erfasst werden, sondern ergeben sich zumeist aus einer Kombination von einzelnen Kontextinformationen und können daraus abgeleitet werden. Ein Einbruch kann beispielsweise daraus geschlossen werden, dass eine Bewegung innerhalb der Wohnung erkannt wird, der Bewohner jedoch nicht in der Wohnung geortet ist oder bereits schläft. Eine Definition solcher Situationen und deren Ableitung aus einer Menge von einzelnen Kontextinformationen ist daher vielfach eine Voraussetzung für die Umsetzung des kontextadaptiven Verhaltens eines AAL-Dienstes. Für die Definition von Bedingungen zum Auslösen eines kontextadaptiven Verhaltens oder der relevanten Situationen werden Vergleichsoperatoren auf dem Kontextmodell benötigt. Hiermit muss ausgedrückt werden können, dass eine Funktion ausgelöst werden soll, wenn ein Zustand einer Kontextentität gegeben ist, z.B. der Bewohner ist gefallen. Aber auch weitere Vergleichsoperationen, die nicht nur auf Gleichheit prüfen, werden benötigt, z.B. das Ereignis findet vor einem Zeitpunkt statt oder der Bewohner befindet sich in der Nähe seiner Wohnung.

Die Auswahl eines geeigneten Ausgabekanals zur Übermittlung der Erinnerungsnachricht ist ebenfalls eine kontextadaptive Erweiterung der eigentlichen Erinnerungsfunktion. Jedoch ist das kontextadaptive Verhalten hier unabhängig von einem konkreten AAL-Dienst. Gleichgültig ob Erinnerungs- oder Notfalldienst, kann die Auswahl des geeigneten Ausgabekanals ein notwendiges kontextadaptives Verhalten eines beliebigen AAL-Dienstes sein. Dieses gilt auch für das in Kap. 3.2 identifizierte kontextadaptive Verhalten, z.B. das Wandern eines AAL-Dienstes auf den verfügbaren Hardwareknoten oder die Auswahl der geeigneten Sensorik. Diese Art der Kontextadaptivität unterscheidet sich daher von dem im vorherigen Absatz beschriebenen. Sie muss durch ein vom Dienst unabhängigen Kontextmodell beschrieben werden. Erstere soll daher als funktionale Kontextadaptivität bezeichnet werden, letztere bezeichnen wir als strukturelle Kontextadaptivität. Diese sollte von der zugrundeliegenden Infrastruktur den AAL-Diensten zur Verfügung gestellt werden, z.B. über eine Methode „getOutputDevice“.

In nachfolgender Abbildung soll der Aufbau eines AAL-Dienstes durch die eigentliche Anwendungslogik und das kontextadaptive Verhalten beschrieben

werden. Die sich an die Kontextmodellierung ergebenden Anforderungen werden in der nachfolgenden Tabelle zusammengefasst.

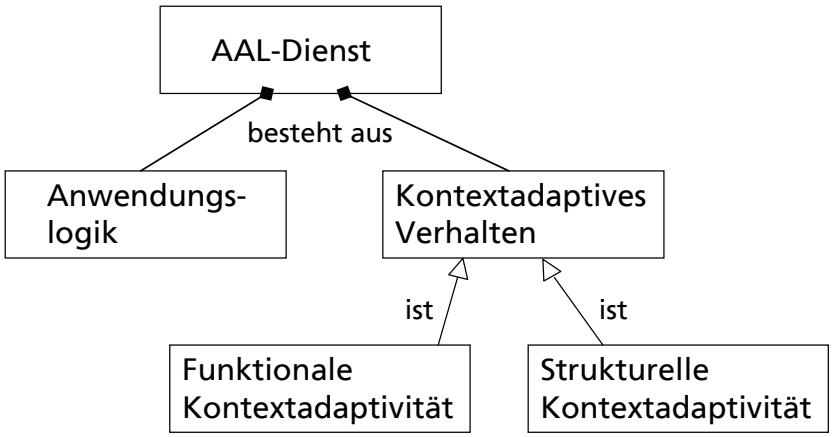


Abbildung 6: Aufbau eines kontextadaptiven AAL-Dienstes

Tabelle 10: Anforderungen an die Kontextmodellierung aus Sicht der Kontextadaptivität von AAL-Diensten

AM5	Kontextadaptivität von AAL-Diensten
AM5.1	Unterstützung der funktionalen Kontextadaptivität
AM5.1.1	Definition und Erkennung relevanter Situationen
AM5.1.2	Umsetzung von Vergleichsoperatoren auf dem Kontextmodell
AM5.2	Unterstützung der strukturellen Kontextadaptivität

3.3.2 Dienstetypen

In Kap. 2.1.3 wurde eine Klassifizierung von AAL-Diensten beschrieben. Das Fraunhofer ISST definiert beispielsweise fünf unterschiedliche Kategorien, in denen AAL-Dienste eingeordnet werden können: „Gesundheit“, „Sicherheit“, „Komfort“, „Soziales Umfeld“ und „Wirtschaftlichkeit“. Nachfolgend werden einige Beispiele für Dienste in diesen Kategorien betrachtet, um daraus mögliche Anforderungen an die Kontextmodellierung abzuleiten. Die betrachtete Menge der AAL-Dienste erhebt keinen Anspruch auf Vollständigkeit und basiert auf derzeitig bekannten Arbeiten, die zum Teil in folgenden Abhandlun-

gen beschrieben sind: [43], [118], [119], [120], [121]. Einige der aufgeführten Arbeiten wurden vom Bundesministerium für Bildung und Forschung der Bundesrepublik Deutschland (BMBF) gefördert. Eine weitere wichtige Quelle für die Förderung von Projekten zur Entwicklung von AAL-Diensten ist das Forschungsrahmenprogramm der EU. Auch Projekte, die im Rahmen von FP6 und FP7 gefördert wurden, sind als Übersicht im Internet zu finden [122]. Einige der betrachteten Dienste sind auch im Rahmen des Projekts „SmarterWohnen“ [40] des Fraunhofer ISST realisiert worden.

Nachfolgend werden einige AAL-Dienste in der Kategorie „Gesundheit“ beschrieben und analysiert.

- Überwachung des Ernährungs- und Wasserhaushalts: Ein solcher Dienst überwacht die Aufnahme von Nahrung und Wasser des Bewohners. Die Daten ermöglichen es, Ratschläge für eine angepasste Ernährung zu geben. Auch kann die Flüssigkeitsaufnahme eines dementen Bewohners mit einem entsprechenden Dienst überwacht werden, um ihn vor dem Austrocknen zu bewahren. Arbeiten in dieser Richtung werden beispielsweise im Rahmen des vom BMBF geförderten Projekts „NutriWear“ getätigt.

- Überwachung von Vitalwerten: Die Vitalwerte des Bewohners werden über entsprechende Sensorik kontrolliert. Auf der Basis konkreter Werte können präventive Maßnahmen angestoßen werden. Dazu gehören beispielsweise die Benachrichtigung des betreuenden Arztes oder die Sensibilisierung des Bewohners. Eine Reihe von Projektaktivitäten zur Entwicklung der Sensorik und Dienste wurde vom BMBF gefördert, z.B. „IMIKRID“, „PRECARE“, „MµGUARD“, „KONMEVIT“, „HYPER-IMS“, „HDSonline“, „PeHeaMon“.

- Überwachung der Medikamenteneinnahme: Ein solcher Dienst überwacht die Einnahme von Medikamenten nach einem Einnahmeplan. Vergisst der Bewohner die Einnahme in einem gegebenen Zeitfenster, so wird er entsprechend daran erinnert. Die Einnahme kann protokolliert und danach dem behandelten Arzt zur Einsicht zur Verfügung gestellt werden. Zusätzlich kann die lokale Verfügbarkeit von Medikamenten überwacht und gegebenenfalls eine Nachbestellung veranlasst werden.

- Überwachung der medizinischen Betreuung: Die Tätigkeiten des Pflegepersonals können aufgezeichnet und für eine spätere Einsichtnahme zur Verfügung gestellt werden.

Diesen Diensten ist gemein, dass sich die notwendigen Kontextinformationen primär auf den Bewohner und seine Vitalwerte beziehen. Daneben werden auch Kontextinformationen benötigt, die die Einnahme von Lebensmitteln,

Flüssigkeit und Medikamenten durch den Bewohner beschreiben. Zudem wird die Zeit als weitere Kontextinformation benötigt, um beispielsweise den Zeitpunkt einer Einnahme zu überwachen. Neben dem Bewohner kann auch der ärztliche Betreuer als weitere Person im Kontextmodell aufgenommen werden. Generell werden die Dienste aktiv, wenn ein Vitalparameter einen kritischen Wert erreicht hat oder die Einnahme einer Substanz erforderlich ist. Konkret kann ein Vitalparameter der Blutdruck oder der Blutzucker sein. Eine eingenommene Substanz kann beispielsweise Wasser, ein Lebensmittel oder ein spezielles Medikament sein. Sowohl Lebensmittel als auch Medikamente können noch weiter spezialisiert werden. Diese Informationen werden als Teil eines Kontextmodells für Dienste in der Kategorie „Gesundheit“ benötigt. Es kann erforderlich sein auf den Vitalwerten oder auf der Zeit Vergleichsoperationen durchzuführen. So können Vergleiche von Kontextinformationen mit gegebenen Vergleichswerten durchgeführt werden, z.B. mit einer vorgegebenen Mindesteinnahmemenge. Interessant ist auch die Notwendigkeit der Unterscheidung von Personen nach ihrer Rolle, z.B. die Unterscheidung zwischen Bewohner, Arzt oder Betreuer. Die Rolle einer Person muss hier als zusätzliche semantische Information die Kontextinformation ergänzen. Die semantische Erweiterung eines Kontextmodells ist daher eine Anforderung aus Sicht der Dienstetypen.

Nachfolgend werden einige AAL-Dienste in der Kategorie „Sicherheit“ beschrieben und analysiert.

- Vermeidung von Unfällen in der Wohnung: Situationen, in denen das häusliche Umfeld ein Potential für Unfälle birgt, sollen vermieden werden. Eine solche ist beispielsweise gegeben, wenn sich ein Gegenstand auf dem Weg des Bewohners in einer dunklen Umgebung befindet. Wird eine entsprechende Situation erkannt, so kann durch ein Warnsignal oder das Einschalten einer Beleuchtung ein Unfall vermieden werden.

- Überwachung von Unfällen in der Wohnung: Mit Hilfe geeigneter Sensorik wird ermittelt, ob ein Bewohner einen Unfall in der häuslichen Umgebung erlitten hat. Ein Beispiel für einen solchen Sensor ist der vom Fraunhofer IESE entwickelte Gehstock, der einen Sturz erkennen kann. Wird ein Unfall ermittelt, so können geeignete Gegenmaßnahmen ausgelöst werden. So kann je nach Schwere des Unfalls der Betreuer oder der Notarzt an den Ort des Geschehens geschickt werden. Eine Entwicklung von Sensorik und eines entsprechenden AAL-Dienstes erfolgt derzeit im Rahmen des von der EU im FP6 geförderten Projekten „EMERGE“ oder auch „CAALYX“. Neben einem Sturz können Schnitte, Verbrennungen oder Vergiftungen weitere Arten von Unfällen sein.

- Überwachung von Gefahrenquellen in der Wohnung: In der Wohnung existieren unterschiedliche Gefahrenquellen für den Bewohner und die Wohnung selbst. Diese entstammen aus der notwendigen Versorgungsinfrastruktur. Gefahrenquellen sind beispielsweise die Strom-, Gas- und Wasserversorgung. Auch Wärmequellen, wie der Heizungskessel auf Basis von Kohle, Holz oder Öl, können als solche potentiell einbezogen werden. Durch Defekte oder Fehlbedienung können beispielsweise gesundheitsgefährdende Gase austreten sowie Brände oder Explosionen entstehen. Durch Sensorik können diese Ereignisse rechtzeitig erkannt und entsprechende Verhütungsmaßnahmen getroffen werden. Dazu gehört beispielsweise die Benachrichtigung des Bewohners und eventueller Anwohner sowie weiterer Dienstleister für die Gefahrenbekämpfung. Beispiele für vom BMBF geförderte Projekte sind „OptoGas“, „Exist“, „IMS“, „Kobra“, „LOCOMED“ oder „PEGAS“.

- Überwachung von Einbrüchen in die Wohnung: Über entsprechende Sensorik kann der Einbruch in die Wohnung eines Bewohners erkannt werden. In Abhängigkeit vom Ort des Einbruchs und der Anwesenheit einer Person können unterschiedliche Aktionen ausgelöst werden. Dazu gehören beispielsweise die Benachrichtigung des Bewohners und der Sicherheitsdienstleister, Maßnahmen zur Abschreckung des Einbrechers sowie das Absperren von Türen zur Einschränkung der Bewegungsfreiheit des Diebs.

Die obigen Dienste benötigen Kontextinformationen, die sich auf die Wohnung, die darin enthaltenen Gegenstände und Personen beziehen. Informationen über die Wohnung müssen ein Ortsmodell beinhalten, um beispielsweise Gefahrenquellen lokalisieren und melden zu können. Ein solches wird auch benötigt, um Gegenstände und den Bewohner in der Wohnung orten und zueinander in Beziehung setzen zu können. Das Ortsmodell muss sowohl Koordinaten beinhalten als auch räumliche Aufteilung darstellen können. Neben dem Bewohner müssen auch weitere Personen, z.B. der Einbrecher oder der Anwohner, identifiziert werden. Auch hier ist eine Unterscheidung zwischen Personen entsprechend ihrer Rolle notwendig. Informationen zur Wohnung bzw. zu Räumen in der Wohnung beinhalten Angaben zur Helligkeit, Rauch oder Gasen. Zu einem Gegenstand kann gehören, den Zeitpunkt seiner Platzierung in der Wohnung als Kontextinformation festzuhalten. Vom Bewohner kann die Information benötigt werden, ob und in welcher Art dieser einen Unfall erlitten hat. Generell werden diese Dienste aktiv, wenn eine potentielle oder tatsächliche Notfallsituation erkannt wurde. Solche Informationen werden als Teil eines Kontextmodells für Dienste in der Kategorie „Sicherheit“ benötigt.

Nachfolgend werden einige AAL-Dienste in der Kategorie „Komfort“ beschrieben und analysiert. Diese können sehr vielschichtig sein und sind nicht nur in das AAL-Umfeld einzuordnen.

- Anpassung der häuslichen Umgebung an die Situation der Bewohner: Kommen die Bewohner zusammen, um gemeinsam Fernsehen zu schauen, so passen sich die Beleuchtung, der Tisch und die Sitzgelegenheit an diese Situation an. Ist der Bewohner gestresst, so wird mit Hilfe von Licht- und Musikszenarien eine angenehme und entspannende Atmosphäre erzeugt. Entsprechende Szenarien lassen sich für weitere Situationen definieren, z.B. Aufwachen, Schlafen gehen, Essen, Besuche von Freunden, außer Haus, Arbeiten, ... Im Rahmen des von der EU im FP6 Rahmenprogramm geförderten Projekts „ALADIN“ werden unterschiedliche Lichtszenarien evaluiert.

- Anpassung der häuslichen Umgebung an externe Rahmenbedingungen: Beleuchtung, Klima, Fenster und sonstige Aktoren passen sich an die externen Rahmenbedingungen wie Wetter und sonstige Ereignisse an. Ein Beispiel hierfür ist das vom BMBF geförderte Projekt „IMIG“.

- Persönlicher Bewohnerbezug: Dienstleistungen und Funktionen innerhalb der Wohnung, die derzeit noch ohne direkten Nutzerbezug sind, werden direkt an dem betreffenden Bewohner ausgerichtet. Beispielsweise werden ankommende Anrufe direkt an das dem Bewohner am nächsten gelegene Interaktionsmedium gerichtet. Die eingestellte Musik folgt dem Bewohner von Raum zu Raum. Gleiches gilt für die präferierte Beleuchtung oder das Raumklima. Weitere Möglichkeiten sind das Befüllen von elektronischen Bilderrahmen oder weiteren flexiblen Dekorationselementen mit personenbezogenen multimedialen Informationen.

- Hinweisdienste: Der Bewohner wird auf bestimmte Ereignisse oder wichtige Termine aufmerksam gemacht. Ereignisse sind beispielsweise interessante Fernsehsendungen, Veranstaltungen oder einkommende elektronische Nachrichten.

- Auffinden von Dingen: Bei der Suche nach Gegenständen kann ein entsprechender Dienst dem Bewohner mitteilen, wo sich dieser Gegenstand befindet.

- Intelligente Hausgeräte: Hausgeräte, insbesondere weiße Ware, wird mit Intelligenz ausgestattet, die es ihnen erlaubt auf die besonderen Bedürfnisse der Bewohner einzugehen und selbständig zu agieren. Bekannte Beispiele sind der intelligente Kühlschrank, der die Ernährung überwacht und Ratschläge gibt sowie automatisch Nahrungsmittel nachbestellt. AAL-Dienste können entsprechende Hausgeräte mit der dafür notwendigen Intelligenz ausstatten. Im Rahmen des von der EU im FP6 Rahmenprogramm geförderten Projekts „EASY LINE+“ wird eine Infrastruktur und ein koordinierender Dienst für weiße Ware entwickelt.

Die obigen Dienste benötigen sehr komplexe und umfangreiche Kontextinformationen. Sie beziehen sich auf die Bewohner, die häusliche Umgebung und die externe Umgebung. Notwendige Informationen über den Bewohner sind seine Position innerhalb der Wohnung, Präferenzen und Interessen sowie die Ableitung der Situation. Eine solche kann in der Regel nicht über einfache Sensoren erfasst werden. Vielfach sind hierfür Verknüpfung von verschiedenen Kontextinformationen und die Anwendung von Heuristiken notwendig [123] [124]. Weiterhin sind geplante Termine als Teil der personenbezogenen Kontextinformationen notwendig. Die häusliche Umgebung muss inklusive der vorhandenen Infrastruktur beschrieben werden. Dazu gehören die möglichen Interaktionsmedien, Geräte und Aktoren. Auch können Gegenstände von Interesse sein, die mit dem Bewohner oder den Geräten interagieren, z.B. Lebensmittel im Kühlschrank. Auch der Zustand von Diensten, z.B. Mail, interessiert als Teil der Kontextinformationen. Außerhalb der häuslichen Umgebung sind Informationen zum Wetter (z.B. Temperatur), Luftqualität oder Lärm von Interesse. Sowohl Informationen zum aktuellen Kontext als auch für künftige Ausprägungen (z.B. Termine oder Wettervorhersage) werden als Teil des Kontextmodells benötigt. Es kann erforderlich sein auf den vorhandenen Informationen Vergleichsoperationen durchzuführen. Es können Vergleiche von Kontextinformationen mit vordefinierten Vergleichswerten durchgeführt werden, aber auch zwischen verschiedenen Kontextinformationen selbst, z.B. der Außentemperatur mit der Temperatur in der Wohnung. Hier muss das Kontextmodell in der Lage sein zu definieren welche Informationen miteinander verglichen werden können.

Nachfolgend werden einige AAL-Dienste in der Kategorie „Soziales Umfeld“ beschrieben und analysiert. Eine Vielzahl dieser Dienste ist ohne Bezug zur Kontextadaptivität.

- Kommunikationssysteme: Es wird eine Kommunikation mit unterschiedlichen Gesprächspartnern sowohl innerhalb der häuslichen Umgebung als auch mit externen Partnern in einer Form ermöglicht, die jener nahe kommt, die einträte, wenn sich alle im gleichen Raum befänden. Hier kann die Ortung der Gesprächspartner eine Rolle spielen, um im Sinne des Bewohnerbezugs die Kommunikation dem Inhaber der Wohnung räumlich folgen zu lassen.

- Awareness Systeme: Sie helfen soziale Beziehungen zu pflegen und aufzubauen. So können Aktivität und Status der Bewohner zusammengefasst, aufgezeichnet und interessierten Kontakten bereitgestellt sowie vermittelt werden, ob der Bewohner gerade beschäftigt ist und nicht gestört werden möchte. Es kann automatisch ein Tagebuch geführt werden, in welchem sich Angehörige über den Tagesablauf des Bewohners informieren können.

- Austausch von Informationen: Es können Dienste angeboten werden, die einen Austausch von Informationen zwischen Bewohnern und Externen ermöglichen. Dazu gehören Terminkalender, Veranstaltungskalender, Bekanntmachungen, das „Schwarze Brett“ oder Mail.

Die obigen Dienste benötigen im Wesentlichen nutzerbezogene Informationen. Dazu gehören die Position, Tätigkeit und Status der Bewohner. Auch externe Personen und deren Beziehungen können Teil des Kontextmodells für Dienste in der Kategorie „Soziales Umfeld“ sein.

Nachfolgend werden einige AAL-Dienste in der Kategorie „Wirtschaftlichkeit“ beschrieben und analysiert. Diese sind generell nicht nur im AAL-Umfeld von Interesse.

- Optimierung des Resourcenverbrauchs: Anhand von Kontextinformationen kann der Verbrauch von Gas, Wasser, Öl, Strom und anderen Resourcen für den Betrieb der häuslichen Umgebung reduziert werden. Dazu können Informationen über das aktuelle und das erwartete Wetter, die Anwesenheit bzw. erwartete Anwesenheit der Bewohner sowie das aktuelle Raumklima verwendet werden. Verlässt der Bewohner für eine längere Zeit die Wohnung, so kann die Raumtemperatur abgesenkt und vor dem Zeitpunkt der Rückkehr wieder angehoben werden. Elektrische Verbraucher können dabei automatisch abgeschaltet werden.

- Dokumentation des Resourcenverbrauchs: Die Verbrauchsinformationen können dokumentiert und dem Bewohner oder auch externen Dienstleistern bzw. Versorgern zur Verfügung gestellt werden.

- Auswahl kostengünstiger Versorger: Entsprechend dem dokumentierten Verbrauch sowie den am Markt vorhandenen Angeboten kann ein kostengünstiger Versorger ausgewählt, dem Bewohner vorgeschlagen und als Dienstleister ausgewählt werden.

- Optimierung der Wartungs- und Instandhaltungskosten von technischen Geräten: Der Status, die Betriebsparameter und Kontextinformationen zur Umgebung können dokumentiert werden, um Rückschlüsse auf Verschleiß und optimale Wartungsintervalle zu gewinnen. Zudem können Schnittstellen für den Zugriff auf Funktionen bereitgestellt werden, mit denen der Servicetechniker auf das Gerät entweder vor Ort oder extern zugreifen kann.

Die obigen Dienste benötigen Kontextinformationen über die Bewohner, die häusliche Umgebung und die externe Umgebung. In Bezug auf den Bewohner kann die aktuelle und geplante Anwesenheit als Information benötigt werden. In der häuslichen Umgebung kann das aktuelle Raumklima von Interesse sein,

z.B. Temperatur, Luftqualität. Zusätzlich werden Informationen über Verbraucher und sonstige Geräte in der Infrastruktur benötigt. Von der externen Umgebung sind das Klima sowie die Helligkeit relevant. Nachfolgend werden die Anforderungen an die konkreten Ausprägungen der jeweiligen dienstetyp-spezifischen Kontextmodelle tabellarisch zusammengefasst.

Tabelle 11: Anforderungen an die konkreten dienstetyp-spezifischen Kontextmodelle

AK6	Dienstetyp-spezifische Kontextmodelle
AK6.1	Dienstetyp „Gesundheit“
AK6.1.1	Informationen über den Bewohner
AK6.1.1.1	Vitalwerte
AK6.1.1.1.1	Blutdruck
AK6.1.1.1.2	Blutzucker
AK6.1.1.1.3	Puls
AK6.1.1.1.4	Gewicht
AK6.1.1.2	Einnahme von
AK6.1.1.2.1	Flüssigkeit
AK6.1.1.2.2	Nahrung
AK6.1.1.2.3	Medikament
AK6.1.1.2.3.1	istVorhanden
AK6.1.2	Informationen über den ärztlichen Betreuer
AK6.1.2.1	Tätigkeit
AK6.2	Dienstetyp „Sicherheit“
AK6.2.1	Informationen über die Wohnung

AK6.2.1.1	Positionsmodell
AK6.2.1.2	Ortsmodell
AK6.2.1.2.1	Räume
AK6.2.1.2.1.1	Verbindungen
AK6.2.1.2.1.2	Bezug zum Positionsmodell
AK6.2.1.2.1.3	Helligkeit
AK6.2.1.2.1.4	Rauch
AK6.2.1.2.1.5	Gase
AK6.2.2	Informationen über die Gegenstände in der Wohnung
AK6.2.2.1	Position innerhalb der Wohnung
AK6.2.2.2	Zeitpunkt der Positionierung
AK6.2.3	Informationen über den Bewohner
AK6.2.3.1	Position innerhalb der Wohnung
AK6.2.3.2	Hat Unfall erlitten?
AK6.2.4	Informationen über weitere Personen
AK6.2.4.1	Art der Person, z.B. Einbrecher, Anwohner, ...
AK6.2.4.2	Ist innerhalb der Wohnung?
AK6.3	Dienstetyp „Komfort“
AK6.3.1	Informationen über den Bewohner
AK6.3.1.1	Position innerhalb der Wohnung
AK6.3.1.2	Situation
AK6.3.1.3	Termine

AK6.3.2	Informationen über die Wohnung
AK6.3.2.1	Vorhandene Infrastruktur
AK6.3.2.1.1	Interaktionsmedien
AK6.3.2.1.2	Geräte
AK6.3.2.1.3	Aktoren
AK6.3.2.1.4	Dienste
AK6.3.3	Informationen über Gegenstände
AK6.3.3.1	Bezug zu Bewohner bzw. Gerät
AK6.3.4	Informationen über die externe Umgebung
AK6.3.4.1	Temperatur
AK6.3.4.2	Luftqualität
AK6.3.4.3	Lärm
AK6.4	Dienstetyp „Soziales Umfeld“
AK6.4.1	Informationen über den Bewohner
AK6.4.1.1	Position innerhalb der Wohnung
AK6.4.1.2	Aktivität
AK6.4.1.3	Status
AK6.4.2	Informationen über weitere Personen
AK6.4.2.1	Beziehung zum Bewohner
AK6.5	Dienstetyp „Wirtschaftlichkeit“
AK6.5.1	Informationen über den Bewohner
AK6.5.1.1	Anwesenheit in der Wohnung

AK6.5.2	Informationen über die Wohnung
AK6.5.2.1	Ortsmodell
AK6.5.2.1.1	Temperatur
AK6.5.2.1.2	Luftqualität
AK6.5.3	Informationen über Geräte
AK6.5.3.1	Betriebsparameter
AK6.5.3.2	Betriebszustand
AK6.5.3.3	Zuordnung zum Ortsmodell
AK6.5.4	Informationen über die externe Umgebung
AK6.5.4.1	Temperatur
AK6.5.4.2	Helligkeit

In Bezug auf die Kontextmodellierung bzw. dem zugrundeliegenden Metamodell lässt sich erkennen, dass die konkreten Kontextmodelle der einzelnen Dienstetypen sich jeweils in den zu beobachtenden Entitäten und der Komplexität der darüber benötigten Informationen unterscheiden. Auch die jeweiligen Situationen, in denen die Dienste aktiviert werden, unterscheiden sich. Daraus folgt, dass es nicht sinnvoll ist ein einheitliches Kontextmodell für alle AAL-Dienste zu definieren. Vielmehr kann als Anforderung abgeleitet werden, dass typspezifische Kontextmodelle zu unterstützen sind, die auf die jeweilig relevanten Kontextinformationen ausgerichtet sind [6]. In den jeweiligen Kontextmodellen wird das Modellelement „Kontextentität“ benötigt um den Bewohner, die Wohnung, Geräte oder weitere Personen zu beschreiben. Eine weitere Anforderung in Bezug auf dieses Modellelement ist die Notwendigkeit der semantischen Erweiterung um Konzepte, wie zum Beispiel der Rolle einer Kontextentität, z.B. „Nachbar“ oder „Betreuer“ (AK6.2.4.1). Weiterhin wird auch die Spezialisierung von Kontextentitäten als Bestandteil des Metamodells benötigt. Das Modellelement „Kontextrelation“ wird benötigt um Beziehungen zwischen verschiedenen Kontextentitäten zu beschreiben, z.B. die Beziehung einer weiteren Person zum Bewohner (AK6.4.2.1). Das Modellelement „Kontextattribut“ wird benötigt um Informationen zu einer Kontextentität zu beschreiben. In einigen der Kontextmodelle wird auch ein Zeitbezug der Kontextinformationen benötigt. Ein Ortsmodell wird benötigt um den Ortsbezug zwischen Kontextentitäten herzustellen. Hier muss sowohl ein georeferenzierter Bezug als auch die

Verwendung eines symbolischen Ortsmodells möglich sein. Weiterhin müssen Situationen als Verknüpfung verschiedener Kontextinformationen definiert werden können. Auch die Definition von Heuristiken zur Beschreibung von Zusammenhängen zwischen Kontextinformationen ist eine ableitbare Anforderung. Diese Anforderungen werden in der nachfolgenden Tabelle zusammengefasst.

Tabelle 12: Anforderungen an die Kontextmodellierung aus Sicht der AAL-Dienste

AM6	Dienstetypen
AM6.1	Unterstützung typspezifischer Kontextmodelle
AM6.2	Modellelemente
AM6.2.1	Verwendung des Modellelements „Kontextentität“ zur Definition von Bewohner, Wohnung, Geräten und sonstigen Personen
AM6.2.1.1	Spezialisierung von Kontextentitäten
AM6.2.2	Verwendung des Modellelements „Kontextrelation“ zur Definition von Beziehungen zwischen Kontextentitäten, z.B. die Beziehung einer Person zu einem Bewohner
AM6.2.3	Verwendung des Modellelements „Kontextattribut“ zur Definition von Eigenschaften von Kontextentitäten, z.B. die Vitalparameter einer Person
AM6.2.3.1	Zeitbezug
AM6.2.3.1.1	Vergangenheit
AM6.2.3.1.2	Aktuelle Information
AM6.2.3.1.3	Zukünftiger Zustand
AM6.2.3.2	Vergleiche von Kontextattributen mit Vorgabewerten
AM6.2.3.3	Zulässige Vergleiche zwischen Kontextattributen
AM6.2.4	Unterschiedliche Repräsentationsformen von z.B. Orts-

			modellen
AM6.2.5			Definition von Situationen, z.B. Notfall
AM6.2.6			Heuristiken zu Abhängigkeiten zwischen Kontextinformationen
AM6.3		Semantische Anreicherung des Kontextmodells	

3.4 Anforderungen aus der Sicht des Bewohners

Ein dritter Blickwinkel dieser Anforderungsanalyse ist die Betrachtung des Bewohners einer intelligenten häuslichen Umgebung. Aus Sicht des Bewohners können zwei unterschiedliche Aspekte beleuchtet werden. Zum einen ist dies der Bedarf eines Bewohners an konkreten AAL-Diensten. Im vorhergehenden Kapitel wurden unterschiedliche Dienstetypen identifiziert. Eine Annahme ist, dass der konkrete Bedarf eines Bewohners an AAL-Diensten von seiner jeweiligen Lebenssituation abhängt. Dieses ist insbesondere bei den Diensten vom Typ „Gesundheit“ ersichtlich. Je nach gesundheitlicher Einschränkung können unterschiedliche AAL-Dienste medizinisch notwendig sein, was bedeutet, dass keine Auswahl von AAL-Diensten vordefiniert werden kann, die als Standardausstattung einer intelligenten häuslichen Umgebung im AAL angenommen werden kann und ausreichend ist. Vielmehr muss es eine Möglichkeit geben, die Dienstemenge an den individuellen Bedarf anzupassen oder zu erweitern. Daraus folgt, dass es möglich sein muss eine häusliche Umgebung durch weitere AAL-Dienste zu erweitern. Das Kontextmodell dient dabei als Schnittstelle zwischen dem AAL-Dienst der Kontextinfrastruktur der häuslichen Umgebung. Dieser Aspekt wird in Kap. 3.5 weiter vertieft.

Ein weiterer Aspekt ist die Interaktion des Bewohners mit seiner intelligenten häuslichen Umgebung und den darin verfügbaren AAL-Diensten. Im Gegensatz zu vielen kontextadaptiven Anwendungen spielt der Anwender eine wichtige Rolle in der kontextadaptiven Ausprägung von AAL-Diensten. Der Bewohner kommt durch die Nutzung der AAL-Dienste in verschiedenen Anwendungsfällen mit den Kontextmodellen in Berührung. Der Aspekt des Endnutzers kann nicht bei der Betrachtung der Anforderungen an die Kontextmodellierung außer Acht gelassen werden. Folgende Anwendungsfälle existieren, in denen Anforderungen aus Sicht des Bewohners identifiziert werden können:

- Explizite Kontexterhebung: Der Bewohner kann manuell Kontextinformationen in die intelligente Umgebung einbringen. Er kann geplante Termine eintragen oder explizit mitteilen, dass er gerade beschäftigt ist und daher nicht gestört werden möchte. Hierbei spielen insbesondere die Kriterien der Ergonomie, aber auch der Wahrung der Privatsphäre

eine Rolle. Für diesen Anwendungsfall wird das Werkzeug, mit dem der Bewohner manuell eine Kontextinformation einbringen kann, als ein besonderer Sensor angesehen. Es ist dann seine Aufgabe, eine für den Bewohner geeignete Benutzerschnittstelle bereitzustellen, was im Rahmen dieser Arbeit nicht weiter diskutiert werden kann. Die Anforderungen an die Bereitstellung von Kontextinformationen über einen Sensor, z.B. die Wahrung der Privatsphäre, sind jedoch in Kap. 3.2.4 beschrieben.

- Kontrolle über die intelligenten Umgebung: Der Bewohner muss in der Lage sein, die Annahmen der intelligenten Umgebung über den Kontext nachzuvollziehen und diese gegebenenfalls zu korrigieren. In [125] wird darauf verwiesen, dass die Übereinstimmung der Erwartungshaltung des Bewohners mit dem Verhalten der intelligenten Umgebung wichtig ist, damit der Bewohner nicht das Gefühl bekommt, die Kontrolle über seine Umgebung zu verlieren und deshalb beginnt, sie zu bekämpfen. Die Kontextadaptivität, die den Benutzer möglichst aus der Interaktion mit dem System fernhält, sowie die Forderung nach der Kontrolle des Nutzers über die intelligente Umgebung stehen scheinbar im Widerspruch zueinander. Diese Forderung resultiert aber aus der Unzuverlässigkeit der Kontexterkennung sowie aus dem nicht immer im Voraus definierbaren gewünschten kontextadaptiven Verhaltens der Umgebung. Die Kontrolle über die Umgebung darf zu keiner Belastung für den Bewohner werden. Dazu müssen die relevanten Kontextinformationen, die Bestandteil der Annahmen der intelligenten Umgebung sind, in einer für den Bewohner geeigneten Art dargestellt werden. Weiterhin muss er die Möglichkeit haben, einfach diese Annahmen zu korrigieren.

- Bestätigung zur Ausführung kritischer Funktionen: Als Strategie für die Handhabung kritischer Funktionen unter Verwendung unsicherer Kontextinformationen kann die explizite Bestätigung durch den Bewohner gefordert werden. Sie muss auf kritische Funktionen und unsichere Kontextinformationen beschränkt bleiben, um den Bewohner nicht über Maßen zu belasten. Hierfür ist es notwendig, dass die Kritikalität einer Funktion eines AAL-Dienstes beschrieben wird. Diese muss Bestandteil einer Dienstebeschreibung sein und ist nicht unmittelbar dem Kontextmodell zuzuordnen. Zudem ist die Unsicherheit einer verfügbaren Kontextinformation zu beschrieben. Letztere ist bereits Bestandteil der Anforderungen zur Beschreibung der Sensorik (Kap. 3.2.4).

- Auswahl von AAL-Diensten: Die Auswahl neuer AAL-Dienste für die häusliche Umgebung setzt voraus, dass der Bewohner über die Art der benötigten Kontextinformationen und eventuelle Einschränkungen oder vorgeschlagene Erweiterungen der Infrastruktur informiert wird. So muss für ihn ersichtlich sein, auf welche Kontextinformationen der AAL-

Dienst zugreifen möchte, um eventuelle Zugriffsregeln zu definieren. Auch hat ersichtlich zu sein, ob diese Kontextinformationen von der verfügbaren Sensorik der häuslichen Umgebung geliefert werden können. Hierbei spielen die Kriterien der Ergonomie eine wesentliche Rolle.

- Definition des kontextadaptiven Verhaltens von Diensten: Der Bewohner muss die Möglichkeit haben, die Kontextadaptivität von Diensten zu konfigurieren. Beispielsweise kann er festlegen, dass ein Komfort-Dienst ihn beim Aufstehen mit seiner Lieblingsmusik beliefert. Hierfür muss er erkennen können, welches kontextadaptive Verhalten festgelegt werden darf. Danach kann er die Konfiguration der Kontextadaptivität vornehmen.

- Darstellung von Kontextinformationen in den Diensten: Es kann notwendig sein, innerhalb der Dienste Informationen über den Kontext geeignet darzustellen. Beispielsweise kann ein Dienst zum Auffinden von Dingen die Position des gesuchten Gegenstands im Raum grafisch darstellen. Hier gelten die Anforderungen aus der Ergonomie. Die geeignete Art der Darstellung einer Kontextinformation ist jedoch abhängig vom jeweiligen Dienst. Die Rauminformation kann beispielsweise über den Namen oder über eine Raumskizze erfolgen. Für beide Formen gibt es entsprechende Einsatzfälle. Daher soll die Darstellung von Kontextinformationen den AAL-Diensten selbst überlassen werden.

Aus der Beschreibung dieser Anwendungsfälle wird ersichtlich, dass der Bewohner unmittelbar mit Kontextinformationen in Berührung kommt. Ein Kontextmodell muss für ihn verständlich sein und ihn in der Interaktion mit der intelligenten häuslichen Umgebung in den beschriebenen Anwendungsfällen unterstützen. Aus Sicht des Bewohners muss ein Kontextmodell eine für ihn geeignete Sicht auf die vorhandenen und verfügbaren Kontextinformationen zur Verfügung stellen, die ihn in der Interaktion mit der intelligenten Umgebung und den AAL-Diensten unterstützt. Dieses impliziert die Art und Weise wie Kontextinformationen auf der Benutzeroberfläche dem Bewohner darzustellen sind. Daraus ergeben sich Anforderungen an die Abstraktion von technischen Details eines operativen Kontextmodells, die für den Bewohner nicht wichtig sind. Diese Anforderungen müssen zusätzlich für die Kontextmodellierung berücksichtigt werden.

Diese Annahmen werden durch die Ergebnisse des ElderTech-Projekts [126] bestätigt. Im Rahmen dieses Projekts wurden bei 13 älteren Personen zwischen 70 und 80 Jahren die Einführung und die Nutzung von AAL-Diensten untersucht. Ein Fazit der Untersuchung ist, dass die Technologie sich mit den Erfahrungen und Bedürfnissen der Nutzer entwickeln muss. Diese Beobachtung unterstützt die Forderung nach einer dynamischen Bereitstellung kontextadaptiver Dienste. Als weitere Kernforderung wird beschrieben, dass es viel

einfacher werden muss, die Technologie in die Umgebung einzuführen, anzuwenden und zu verstehen. Daraus ergeben sich Anforderungen an die Handhabung und Darstellung der Kontextadaptivität eines AAL-Dienstes. Anforderungen an die Interaktion im Rahmen des AAL aus Sicht der Nutzer werden auch an anderer Stelle beschrieben, z.B. [11], [43], [125]. Diese Forderungen können in folgende Kriterien zusammengefasst werden: Ergonomie und soziale Akzeptanz.

Im Sinne der Ergonomie, deren Kriterien in den Standards ISO 9241 und ISO 14915 zusammengefasst sind, muss die Form der Interaktion auf die Bedürfnisse der älteren Bewohner abgestimmt sein. Diese unterscheiden sich beispielsweise von denen eines Entwicklers eines AAL-Dienstes. Daraus sind Anforderungen an die Kommunikation von Kontextinformationen innerhalb der intelligenten Umgebung und den AAL-Diensten in den folgenden Aspekten zu identifizieren:

- Aufgabenangemessenheit: Die Interaktion mit der intelligenten Umgebung und mit den AAL-Diensten erfolgt in einer der jeweiligen Aufgabenstellung angemessenen Form. Der Bewohner soll nicht von seiner eigentlichen Zielstellung abgelenkt werden. Hierbei muss insbesondere sein mögliches fehlendes technisches Verständnis berücksichtigt werden. Für ihn muss eine solche Interaktion möglichst einfach und intuitiv sein. Die komplexen Möglichkeiten zur Definition des kontextadaptiven Verhaltens von AAL-Diensten müssen soweit reduziert werden, bis der Bewohner in der Lage ist, diese zu verstehen und anzuwenden. Die in [125] beschriebene Nutzeroberfläche (Kap. 4.1.5), mit der ein Bewohner in die Lage versetzt werden soll, die Kontextbedingungen für AAL-Dienste selbst zu definieren, ist unter diesem Gesichtspunkt nicht geeignet, da das Verständnis über die Boolsche Verknüpfung von Kontextereignissen vorausgesetzt wird.

- Selbstbeschreibungsfähigkeit: Der Bewohner muss in der Lage sein, den Einsatzzweck und das kontextadaptive Verhalten eines AAL-Dienstes zu verstehen und mit diesem zu interagieren, ohne dass Hilfe durch technisch geschultes Personal notwendig wird. Auch hier ist zu fordern, dass in der Interaktion mit dem Bewohner von der Technologie abstrahiert wird, damit dieser in der ihm bekannten Terminologie kommunizieren kann.

- Steuerbarkeit: Der Bewohner muss das Gefühl haben, dass er das System steuert und weiterhin Herr in seiner häuslichen Umgebung ist [125]. Dazu muss er in der Lage sein, das kontextadaptive Verhalten der AAL-Dienste in seinem Sinne zu bedienen und an seine Bedürfnisse und die momentane Situation anzupassen. Dieser Aspekt ist auch deswegen wichtig, weil durch das auf der Basis der verfügbaren Kontextinformationen proaktive Verhalten der AAL-Dienste die intelligente

Umgebung ein Eigenleben entwickelt, welches vom Bewohner als bedrohlich empfunden werden könnte. Eine negative Eigenschaft von Kontextinformationen ist ja die, dass sie insbesondere bei komplexen Situationsableitungen fehlerhaft sein können. Der Bewohner muss deshalb in der Lage sein, diese Fehler zu erkennen und zu korrigieren. So ist es notwendig, die Annahmen der intelligenten Umgebung über den Kontext in geeigneter Form darzustellen und dem Bewohner Möglichkeiten zur Korrektur zu geben.

- Erwartungskonformität: Der Bewohner möchte möglichst einfach und schnell einen AAL-Dienst nutzen können. Die Nutzerführung zur Definition des kontextadaptiven Verhaltens sollte den Erwartungen der Bewohner entsprechen, die in etwa durch das Handhaben von Fernbedienungen geprägt ist.

- Fehlertolerant: Die Nutzerführung ermöglicht es, Fehler in der Bedienung und in den Eingaben möglichst früh zu erkennen und darauf zu reagieren, sodass ein beabsichtigtes kontextadaptives Verhalten gegenüber den AAL-Diensten mit minimalem Aufwand erreicht werden kann.

- Individualisierbarkeit: Der Bewohner hat die Möglichkeit, die Interaktion bezüglich der Kontextaspekte an seine jeweiligen Bedürfnisse und Arbeitsweise anzupassen. Mit fortschreitender Erfahrung in der Definition des kontextadaptiven Verhaltens können dann sicher auch komplexere Dialoge angeboten werden.

- Lernförderlichkeit: Der Bewohner ist in die Lage zu versetzen, schnell und einfach die Interaktion mit der intelligenten Umgebung und den AAL-Diensten zu erlernen oder anzuwenden. Entsprechend gestaltete Erklärungen und Hilfestellungen sind deshalb vorzuhalten.

- Eignung für Wahrnehmung und Verständnis: Die Informationen bezüglich der Kontextaspekte eines AAL-Dienstes und der Annahmen in der intelligenten Umgebung werden für den Bewohner leicht verständlich und korrekt vermittelt. Er kann die relevanten Inhalte schnell erfassen und verarbeiten.

In nachfolgender Tabelle werden die Anforderungen an die Kontextmodellierung aus Sicht des Bewohners anhand der relevanten Anwendungsszenarien und den entsprechend zu berücksichtigen Kriterien der Ergonomie zusammengefasst.

Tabelle 13: Anforderungen an die Kontextmodellierung aus Sicht des Bewohners

AM7	Nutzung der AAL-Dienste durch den Bewohner
AM7.1	Erweiterbarkeit der Menge der AAL-Dienste der intelligenten häuslichen Umgebung
AM7.2	Interaktion des Bewohners mit der intelligenten Umgebung und den AAL-Diensten
AM7.2.1	Kontrolle über die intelligente Umgebung
AM7.2.1.1	Nutzergerechte Darstellung der Annahmen der intelligenten Umgebung bezüglich des Kontext: erwartungskonform, selbstbeschreibend, geeignet für die Wahrnehmung und das Verständnis.
AM7.2.1.2	Nutzergerechte Korrektur der Annahmen der intelligenten Umgebung bezüglich des Kontext: erwartungskonform, selbstbeschreibend, fehlertolerant, lernförderlich und geeignet für die Wahrnehmung und das Verständnis.
AM7.2.2	Auswahl von AAL-Diensten
AM7.2.2.1	Darstellung der vom AAL-Dienst benötigten Kontextinformationen: erwartungskonform, selbstbeschreibend, geeignet für die Wahrnehmung und das Verständnis.
AM7.2.2.2	Darstellung der von der häuslichen Umgebung bereitgestellten Kontextinformationen: erwartungskonform, selbstbeschreibend, geeignet für die Wahrnehmung und das Verständnis.
AM7.2.3	Definition des kontextadaptiven Verhaltens von Diensten
AM7.2.3.1	Darstellung des möglichen kontextadaptiven Verhaltens: erwartungskonform, selbstbeschreibend, lernförderlich und geeignet für die Wahrnehmung und das Verständnis.
AM7.2.3.2	Festlegung des kontextadaptiven Verhaltens: aufgabenangemessen, erwartungskonform, selbstbeschreibend, fehlertolerant, lernförderlich und geeignet für die Wahrnehmung und das Verständnis.

3.5 Anforderungen aus Sicht der Entwicklungs- und Nutzungsphasen von AAL-Diensten

Der vierte Blickwinkel dieser Anforderungsanalyse ist die Betrachtung der Entwicklung und Nutzung von AAL-Diensten. Dieser Blickwinkel geht einher mit der Zielstellung der Dissertation der Schaffung eines methodischen Ansatzes zur Realisierung des kontextadaptiven Verhaltens eines AAL-Dienstes.

Eine besondere Herausforderung für die Realisierung von AAL-Diensten ist das dynamische technische Umfeld. Dieses gilt für die wachsende Menge verfügbarer Sensorik (Kap. 3.2.4) aber auch für den Bedarf und die Verfügbarkeit von AAL-Diensten (Kap. 3.4). Somit ist der Aufbau einer intelligenten häuslichen Infrastruktur mitsamt den darin verfügbaren AAL-Diensten kein einmaliger Akt, sondern setzt sich sukzessive fort. Daraus folgt, dass die konkreten Ausprägungen der intelligenten häuslichen Umgebungen in Bezug auf Sensorik und AAL-Diensten je nach individuellem Bedarf der jeweiligen Bewohner unterschiedlich sein werden. Unsere Grundannahme ist, dass es aufgrund der Komplexität einer solchen technischen Lösung eine Trennung zwischen dem Aufbau der Kontextinfrastruktur als Bestandteil einer häuslichen intelligenten Umgebung mitsamt der darin enthaltenen Sensorik und der Realisierung kontextadaptiver AAL-Dienste geben wird. Ein AAL-Dienst wird nicht gemeinsam mit der zugehörigen Sensorik gemeinsam als Paket entwickelt und vertrieben, so wie es aktuell der Fall ist. Vielmehr wird eine Kontextinfrastruktur aufgebaut und unterschiedlichen AAL-Diensten zur Verfügung gestellt werden. Dieses impliziert, dass der Entwickler eines AAL-Dienstes die seinem Produkt zugrundeliegende Sensorik nicht selber entwickeln bzw. festlegen kann. Vielmehr muss der AAL-Dienst in der Lage sein die in der Kontextinfrastruktur verfügbare Sensorik zu nutzen. Das der Kontextinfrastruktur zugrundeliegende Kontextmodell muss somit unabhängig von den Vorgaben eines konkreten Dienstes sein. Die Unterstützung der Trennung zwischen Sensorik und dem Kontextmodell eines AAL-Dienstes ist daher eine Anforderung an die Kontextmodellierung. Diese Trennung ermöglicht zudem den Aufbau eines Dienstemarktes. Unterschiedliche Diensteentwickler können für den Markt AAL-Dienste entwickeln und dort bereitstellen. Gleiches gilt für die Entwickler von Sensoren. Der einzelne Bewohner besitzt dann die Möglichkeit, sich auf dem Markt zu bedienen und seine häusliche intelligente Umgebung entsprechend seinem Bedarf an neuer Sensorik und AAL-Diensten zu erweitern.

Diese Grundannahme gilt nachfolgend bei der Analyse der weiteren Anforderungen in den einzelnen Entwicklungs- und Nutzungsphasen von AAL-Diensten. Wie in Kap. 3.3.1 beschrieben können wir den Aufbau eines AAL-Dienstes in die eigentliche Anwendungslogik und das kontextadaptive Verhalten unterteilen. Für die methodische Realisierung der Anwendungslogik sind vielfältige Vorgehensmodelle bekannt, z.B. das Wasserfall-, Spiral-, RUP- oder V-Modell. Vorgehensmodelle spalten einzelne Aktivitäten zu verschiedenen Phasen im Entwicklungsprozess auf. Jedes dieser Phasen hat eine klar abge-

grenzte Zielstellung und führt zu Ergebnissen, auf denen die Aktivitäten in der Folgephase aufbauen können. Auf diese Weise wird beispielsweise sichergestellt, dass Anforderungen erhoben und abgestimmt wurden und vollständig in die Konzeption einfliessen. Dadurch kann die Komplexität der Erstellung von Anwendungen in Teilaspekte aufgeteilt und Spezialisten zugeordnet werden. Entwicklungswerkzeuge können eingesetzt werden, mit denen die den jeweiligen Phasen zugeordneten Konzepte und Modellierungssichten unterstützt werden. Eine Annahme für die weitere Betrachtung von Anforderungen ist, dass ein solches methodisches Vorgehen auch für die Realisierung des kontextadaptiven Verhaltens von Vorteil ist. Auch für die Kontextmodellierung sollten in den jeweiligen Phasen spezialisierte Konzepte und Modellierungssichten identifiziert werden können. Darauf aufbauend können dann Entwicklungswerkzeuge bereitgestellt werden, die die Entwicklung der kontextadaptiven Logik eines AAL-Dienstes unterstützen. Eine Analogie dieser Vorgehensweise ist in der Datenmodellierung zu finden. Mit dem Datenbank-Entwurfsprozess ([127], S. 45) sind verschiedene Phasen definiert, welche den Vorgehensmodellen der Softwareentwicklung weitgehend entsprechen.

Für die Definition der jeweiligen Phasen der Kontextmodellierung betrachten wir nachfolgend zunächst die definierten Phasen im Wasserfallmodell. Dieses ist sehr einfach gehalten und definiert in seiner erweiterten Ausprägung folgende Phasen mit entsprechenden Übergängen und Rücksprüngen:

- Planung: Die Planungsphase dient dazu, grob festzulegen, was zu realisieren ist, um auf den hier erhobenen Informationen eine Planung des Aufwands, der Machbarkeit und der Form der Umsetzung vornehmen zu können. Zu den Aktionen innerhalb der Planungsphase gehören die Marktbeobachtung, die Erstellung eines Lastenhefts, die Projektkalkulation und die Projektplanung im engeren Sinn.

- Definition: In der Definitionsphase müssen die Anforderungen an das umzusetzende Produkt im Detail erhoben, analysiert und dokumentiert werden. Zu ihren Aktionen gehören die Definition des umzusetzenden Produkts in Form eines Pflichtenhefts, einem Produktmodell und einem Modell der Nutzerschnittstelle. Damit können die geforderten Produkteigenschaften verbindlich festgelegt werden.

- Entwurf: In dieser Phase erfolgt ein formalisierter Entwurf der Umsetzung der geforderten Produkteigenschaften. Zumeist ist er unabhängig von der für die Implementierung verwendeten Technologie. Für unterschiedliche Aspekte der Realisierung werden Entwurfsmethoden verwendet, um die grundlegenden Konzepte der Umsetzung festzulegen. Zu den Aktionen innerhalb dieser Phase gehören die Festlegung der Architektur, das Design der Klassen und Datenstrukturen.

- Implementierung: In dieser Phase wird auf der Basis der im Entwurf festgelegten grundlegenden Konzepte das Produkt mit Hilfe unterschiedlicher Technologien umgesetzt. So können Datenmodelle in ein Datenbankmanagementsystem überführt und die Anwendungslogik mit Hilfe einer Programmiersprache realisiert werden.

- Test: Hier soll sichergestellt werden, dass die Umsetzung des Produkts den in der Definitionsphase festgelegten funktionalen und nichtfunktionalen Anforderungen entspricht. Dazu werden Testpläne erstellt und spezifiziert. Auf dieser Basis werden dann Tests durchgeführt und dokumentiert.Dabei ist zu kontrollieren, ob die in der Definitionsphase festgelegten Eigenschaften erfüllt sind.

- Einsatz und Wartung: Das fertige Produkt wird dem Kunden ausgeliefert. Dieser wird in das Produkt eingeführt und verwendet es. Die Wartung stellt sicher, dass das Produkt betriebsbereit bleibt.

Das Wasserfallmodell hat einige Nachteile. Dazu gehört die mangelnde Flexibilität des Vorgehensmodells gegenüber Änderungen beispielsweise in den Anforderungen. Fehler können erst spät erkannt werden und führen zu erheblichen Kosten. Diese und weitere Kritikpunkte führten zu alternativen oder ergänzenden Vorgehensmodellen. Das V-Modell definiert z.B. lediglich Aktivitäten und Ergebnisse und legt keine zeitliche Abfolge fest. Zudem bestimmt es unterschiedliche Detaillierungsebenen von der Systemsicht bis zur Softwaresicht. Unabhängig von dieser Kritik sind diese Phasen eine gute Grundlage für die Betrachtung des Vorgehens zur Kontextmodellierung. Die Festlegung dieser Phasen impliziert nicht die Verwendung des V-Modells für die Realisierung der Anwendungslogik eines AAL-Dienstes. Analog zur Datenmodellierung können die Phasen der Kontextmodellierung den Phasen des jeweils gewählten Vorgehensmodells zugeordnet und angepasst werden.

Neben den Phasen der Entwicklung des kontextadaptiven Verhaltens sollen noch die Phasen betrachtet werden, die für die Bereitstellung eines AAL-Dienstes in einem Dienstemarkt, sowie für die Übernahme und Betrieb des AAL-Dienstes in der intelligenten häuslichen Umgebung benötigt werden.

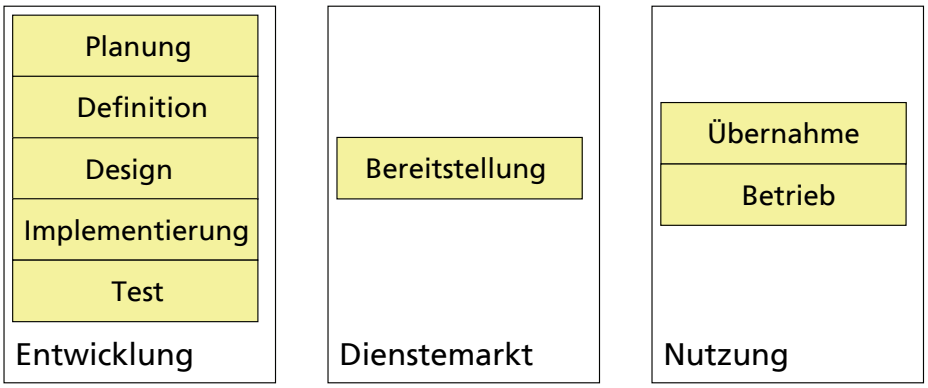


Abbildung 7: Betrachtete Phasen der Entwicklung und Nutzung von AAL-Diensten

Diese Phasen müssen nachfolgend getrennt für die häusliche Kontextinfrastruktur und die kontextadaptiven AAL-Dienste betrachtet werden. Zunächst werden der Aufbau und die Nutzung der Kontextinfrastruktur der häuslichen Umgebung entlang der Phasen betrachtet. Anschließend erfolgt eine solche Betrachtung für die Realisierung und die Nutzung der kontextadaptiven AAL-Dienste. Diese getrennte Betrachtung ist notwendig aufgrund der Annahme der Trennung zwischen Kontextinfrastruktur und AAL-Diensten.

3.5.1 Kontextinfrastruktur

Die Kontextinfrastruktur muss die Grundlagen für den Betrieb eines AAL-Dienstes schaffen. Sie hat die notwendigen Kontextinformationen zur Umsetzung des kontextadaptiven Verhaltens des Dienstes bereitzustellen. Weiterhin muss eine Kontextinfrastruktur notwendige Mechanismen zur Nutzung von Sensorik vorhalten. Dabei muss die Sensorik unabhängig von den zu unterstützenden Diensten betrachtet werden. Diese muss ebenfalls unabhängig von einer konkreten Instanz einer intelligenten häuslichen Umgebung sein. Diese Voraussetzungen müssen durch ein Kontextmodell geschaffen werden. In ihr muss beschrieben werden, welche Arten von Kontextinformationen in einer häuslichen Umgebung potentiell bereitgestellt werden können. Die Realisierung einer solchen Standardsicht auf Sensoren und den von ihnen gelieferten Kontextinformationen ist Bestandteil eines Domänenmodells für das AAL. Dieses ist Gegenstand der Betrachtung der Phasen der Entwicklung der Kontextinfrastruktur.

In der Planungsphase muss zunächst das Einsatzgebiet der Kontextinfrastruktur festgelegt werden. Hierzu ist eine Betrachtung der im AAL einzusetzenden Dienstetypen notwendig. Dabei ergeben sich noch keine grundlegenden Anforderungen an die Kontextmodellierung.

In der Definitionsphase wird aus der Betrachtung des Einsatzgebiets eine Erhebung der dort vorhandenen Sensorik sowie der benötigten Kontextinformationen. Hieraus resultiert eine Übersicht der potentiell in der Kontextinfrastruktur zu nutzenden Kontextinformationen.

In der Designphase wird dann die Übersicht der potentiellen Kontextinformationen konzeptuell in ein formales Kontextmodell überführt. Dieses dient als Grundlage für die Implementierung und kann zur Diskussion und zur Dokumentation verwendet werden.

In der Implementierungsphase muss das konzeptuelle Kontextmodell in eine Form überführt werden, die von der Kontextinfrastruktur zur Verwaltung und Einbindung der Sensorik sowie zur Bereitstellung der verfügbaren Kontextinformationen genutzt werden kann.

Eine Testphase entfällt für das entwickelte Kontextmodell. Jedoch muss es regelmäßig darauf hin überprüft werden, ob es aufgrund neuer Anforderungen durch die Sensorik oder neuer AAL-Dienste ausreichend ist oder erweitert werden muss. Im letzteren Fall durchläuft die Entwicklung des Kontextmodells für die Kontextinfrastruktur wieder die Phasen der Entwicklung.

Die Instanziierung einer Kontextinfrastruktur kann je nach Bewohner unterschiedlich sein, weil unterschiedliche Sensoren in die Kontextinfrastruktur einer häuslichen intelligenten Umgebung aufgenommen werden können. Wir gehen davon aus, dass sie von Herstellern im Dienstemarkt bereitgestellt werden. In dieser Phase ist daher eine Beschreibung der Sensoren notwendig, um das Auffinden geeigneter Produkte im Dienstemarkt zu ermöglichen. Aus ihr müssen die Art und die Qualität der lieferbaren Kontextinformationen ersichtlich werden.

Bei der Übernahme des Sensors in die Kontextinfrastruktur muss eine Zuordnung der gelieferten Kontextinformationen in das der Infrastruktur zugrundeliegende Kontextmodell erfolgen. Diese muss auf der Basis der Sensorbeschreibung sowie des implementierten Kontextmodells der Infrastruktur erfolgen. Dabei können zusätzliche Informationen zur Nutzungsmodalität eingebracht werden, z.B. Regeln für den Zugriff auf die Kontextinformationen.

In der Phase des Betriebs müssen die von den Sensoren gelieferten Kontextinformationen in das implementierte Kontextmodell überführt und dort den AAL-Diensten zur Verfügung gestellt werden.

Aus dieser Beschreibung der Verwendung von Kontextmodellen in den identifizierten Phasen für die Entwicklung und die Nutzung einer Kontextinfrastruktur wird ersichtlich, dass sie in den einzelnen Phasen einen unterschiedlichen Einsatzzweck besitzen und sich somit in der Ausprägung und im Abstraktions-

grad voneinander unterscheiden werden. Nachfolgend sind diese Anforderungen tabellarisch zusammengefasst.

Tabelle 14: Anforderungen an die Kontextmodellierung aus Sicht der Entwicklung und Nutzung der Kontextinfrastruktur

AM8	Entwicklung und Nutzung der Kontextinfrastruktur
AM8.1	Trennung der Sicht auf Kontextinformationen zwischen AAL-Diensten und Kontextinfrastruktur
AM8.2	Unterstützung der unterschiedlichen Phasen der Entwicklung und Nutzung der Kontextinfrastruktur
AM8.2.1	Bereitstellung einer Übersicht der potentiell nutzbaren Kontextinformationen in der Definitionsphase.
AM8.2.2	Konzeptuelle Beschreibung der in der Kontextinfrastruktur zu implementierenden Kontextinformationen in der Designphase.
AM8.2.3	Lauffähige Umsetzung des Kontextmodells zur Verwaltung von Sensoren und Bereitstellung von Kontextinformationen in der Implementierungsphase, der Übernahmephase und der Phase des Betriebs.
AM8.2.4	Beschreibung der Eigenschaften der Sensorik und der von ihnen gelieferten Kontextinformationen für die Bereitstellungs- und Übernahmephase.

3.5.2 AAL-Dienste

Ein AAL-Dienst wird von einem Dienstleister entwickelt und über einen Dienstemarkt für die Nutzung innerhalb einer intelligenten häuslichen Umgebung verfügbar gemacht. Aufgrund der Trennung zwischen Kontextinfrastruktur und den AAL-Diensten kann der Diensteentwickler die Ausprägung der Sensorik nicht selber bestimmen. Vielmehr muss er ein Kontextmodell unabhängig von der Sensorik definieren, welches dem kontextadaptiven Verhalten seines Dienstes zugrunde liegen muss. Dieses beschreibt sodann die Anforderungen des AAL-Dienstes an die Kontextinfrastruktur des Bewohners. Bei der Übernahme eines solchen Dienstes muss ein Abgleich der Anforderungen mit den Fähigkeiten der Kontextinfrastruktur erfolgen. Im Betrieb sind dann die vorhandenen Kontextinformationen auf das dem AAL-Dienst zugrundeliegende

Kontextmodell abzubilden, das Gegenstand der Betrachtung von Phasen der Entwicklung und Nutzung der AAL-Dienste ist.

In der Planungsphase wird zunächst das Einsatzgebiet des AAL-Dienstes festgelegt. Hierzu erfolgt eine Einordnung zu den identifizierten Dienstetypen. Zusätzlich müssen die umzusetzenden Funktionen des AAL-Dienstes in Form eines Lastenheftes beschrieben werden. Es umfasst auch eine Beschreibung des kontextadaptiven Verhaltens des Dienstes.

In der Definitionsphase erfolgt eine Beschreibung der Realisierung der Funktionalität. Hierbei wird es notwendig sein, die Beschreibung der Kontextadaptivität um die Anforderungen an die Art und Qualität der benötigten Kontextinformationen anzureichern. Dieses kann in Form eines Pflichtenheftes geschehen.

In der Designphase sind die benötigten Kontextinformationen konzeptuell in ein formales Kontextmodell zu überführen. Es dient als Grundlage für die Implementierung des dienstespezifischen Kontextmodells und kann zur Diskussion und Dokumentation verwendet werden.

In der Implementierungsphase muss das konzeptuelle Kontextmodell in eine Form überführt werden, die zur Realisierung des kontextadaptiven AAL-Dienstes genutzt werden kann. Hierfür muss der Diensteentwickler eine eigene Kontextinfrastruktur besitzen, in der das Kontextmodell umgesetzt wird. Dieses wird durch entsprechende Sensorik mit Kontextinformationen befüllt und dem Dienst zur Verfügung gestellt.

In der Testphase wird der realisierte AAL-Dienst in der Kontextinfrastruktur des Diensteentwicklers getestet. Dabei ist auch das korrekte kontextadaptive Verhalten zu prüfen. Der Test erfolgt auf der Basis des implementierten Kontextmodells und den dort vorhandenen Kontextinformationen. Hierbei kann es sinnvoll sein, das Verhalten auch unter unterschiedlichen Ausstattungen der Kontextinfrastruktur in Bezug auf die Sensorik zu testen.

In der Phase der Bereitstellung bietet der Diensteentwickler den AAL-Dienst in einem Dienstemarkt den potentiellen Nutzern an. Zu diesem Zweck muss der AAL-Dienst so beschrieben werden, dass er nachgefragt wird. Dazu gehört die Beschreibung der Funktionalität und des kontextadaptiven Verhaltens sowie der Anforderungen an die Kontextinfrastruktur.

Bei der Übernahme des AAL-Dienstes in die häusliche intelligente Umgebung des Bewohners muss eine Abbildung des dienstespezifischen Kontextmodells in das durch die Infrastruktur bereitgestellte Kontextmodell erfolgen. Hierfür sind die Beschreibung des Dienstes, sowie die Implementierung der Kontextinfrastruktur notwendig. Bei diesem Abgleich ist es möglich, dass nur ein Teil des kontextadaptiven Verhaltens des AAL-Dienstes durch die Kontextinfrastruktur unterstützt wird. Dieses kann dazu führen, dass der Bewohner die

Kontextinfrastruktur um zusätzliche Sensorik erweitert (Übernahmephase der Kontextinfrastruktur).

In der Phase des Betriebs müssen die vom AAL-Dienst benötigten Kontextinformationen in dem dienstespezifischen Kontextmodell durch die Kontextinfrastruktur zur Verfügung gestellt werden.

Aus dieser Beschreibung der Verwendung von Kontextmodellen in den identifizierten Phasen für die Entwicklung und die Nutzung eines AAL-Dienstes wird ersichtlich, dass diese in den einzelnen Phasen einen unterschiedlichen Einsatzzweck haben und sich somit in der Ausprägung und im Abstraktionsgrad voneinander unterscheiden werden. Nachfolgend werden diese Anforderungen tabellarisch zusammengefasst.

Tabelle 15: Anforderungen an die Kontextmodellierung aus Sicht der Entwicklung und Nutzung von AAL-Diensten

AM9	Entwicklung und Nutzung von AAL-Diensten
AM9.1	Trennung der Sicht auf Kontextinformationen zwischen AAL-Diensten und Kontextinfrastruktur
AM9.1.1	Abgleich der Anforderungen eines AAL-Dienstes mit den Fähigkeiten der Kontextinfrastruktur
AM9.2	Unterstützung der unterschiedlichen Phasen der Entwicklung und Nutzung der AAL-Dienste
AM9.2.1	Beschreibung des geforderten kontextadaptiven Verhaltens des AAL-Dienstes in der Planungsphase
AM9.2.2	Detaillierte Beschreibung der Art und Qualität der benötigten Kontextinformationen für die Umsetzung der Kontextadaptivität.
AM9.2.3	Konzeptuelle Beschreibung der vom AAL-Dienst benötigten Kontextinformationen in der Designphase.
AM9.2.4	Lauffähige Umsetzung des dienstespezifischen Kontextmodells in der Implementierungs-, Test-, Übernahme- und Betriebsphase.
AM9.2.5	Beschreibung der kontextadaptiven Funktionalität und der Anforderungen an die Kontextinfrastruktur eines AAL-Dienstes für die Bereitstellungs- und Übernahmephase.

3.6 Zusammenfassung

In diesem Kapitel wurde eine Erhebung der Anforderungen an die Kontextmodellierung im AAL aus vier Blickwinkeln durchgeführt. Es wurden die notwendige Infrastruktur der häuslichen Umgebung des Bewohners und die daraus erkennbaren Anforderungen betrachtet. Danach wurden die Kontextadaptivität von AAL-Diensten und mögliche Ausprägungen abgehandelt. Im Weiteren war die Nutzung von AAL-Diensten durch den Bewohner zu bearbeiten. Zuletzt wurden die Phasen der Entwicklung und der Nutzung von kontextadaptiven AAL-Diensten betrachtet. Hieraus konnten eine Reihe konkreter Anforderungen an die Kontextmodellierung identifiziert werden, die zwei Kategorien zugeordnet werden können.

Die Anforderungen der ersten Kategorie betreffen die konkrete Ausprägung eines Kontextmodells. Es wird beschrieben, welche Kontextinformationen in ihm enthalten sein müssen. Dazu gehören die Anforderungen, die sich aus der Betrachtung der Infrastruktur, aber auch der Untersuchung der Dienstetypen ergeben. Sie lassen sich mit vielen der bereits bekannten Ansätze zur Kontextmodellierung umsetzen. Diese Anforderungen sind nicht unmittelbar relevant für die Zielstellung der Dissertation einer Modellierungsmethodik für das AAL. Sie können als Vorgaben für die Realisierung der unterschiedlichen Kontextmodelle auf Basis der Modellierungsmethodik angesehen werden. Im Anhang wird die Umsetzung der konkreten Kontextmodelle beschrieben.

Die Anforderungen der zweiten Kategorie beschreiben die generellen Eigenschaften, die ein Kontextmodell, unabhängig von dessen konkreter Ausprägung, im AAL erfüllen muss. Daraus ergeben sich die benötigten Modellelemente und deren Semantik. Anhand dieser Anforderungen lässt sich die Eignung der aktuell bekannten oder noch zu entwickelnden Ansätze zur Kontextmodellierung bewerten. Daher bilden sie die Basis für die in den nächsten Kapiteln dargestellte Konzeption. Diese Anforderungen können wie folgt zusammengefasst werden:

- Bereitstellung von Modellelementen zur Erstellung konkreter Kontextmodelle, z.B. Kontextentitäten, Kontextrelationen, Kontextattribute, zulässige Vergleiche zwischen Kontextattribute, Situationen und Repräsentationsformen (AM1, AM1.2, AM1.3, AM2.1, AM2.2, AM2.3, AM2.4, AM2.5, AM3.1, AM3.2, AM3.3, AM3.4, AM6.2, AM6.3).

- Nutzung des Kontextmodells zur Auswahl von Sensorik (AM4.2)

- Erweiterbarkeit des konkreten Kontextmodells, beispielsweise zur Erfassung neuartiger Sensoren (AM4.1) oder der Erweiterung der Menge von AAL-Diensten in der häuslichen Umgebung (AM7.1).

- Trennung zwischen Diensten und Kontextinfrastruktur (AM8.1, AM9.1). Dazu gehört die Unterscheidung zwischen der strukturellen und der funktionalen Kontextadaptivität eines AAL-Dienstes (AM5.1, AM5.2).

- Unterstützung spezifischer Kontextmodelle (AM6.1) für einzelne Dienstetypen.

- Endnutzergerechte Darstellung, Korrektur und Konfiguration von Kontextinformationen (AM7.2.1, AM7.2.2, AM7.2.3).

- Unterstützung unterschiedlicher Entwicklungs- und Nutzungsphasen der Kontextinfrastruktur und der AAL-Dienste. Bereitstellung geeigneter Abstraktionsebenen und Repräsentationsformen. Unterstützung der Überführung zwischen diesen Phasen (AM8.2, AM9.2).

4 Kontextmodellierung im Ambient Assisted Living

Nachfolgend werden auf der Basis der erhobenen Anforderungen an die Kontextmodellierung die grundlegenden Konzepte für die Kontextmodellierung im AAL erarbeitet und in den dann darauf folgenden Kapiteln ausdifferenziert. Dazu gehören die mehrdimensionale Kontextmodellierung, sowie die Definition eines Metamodells. Zuvor werden jedoch die bereits bekannten Kontextmodelle anhand der Anforderungen in Bezug auf diese Anforderungen bewertet.

4.1 Bewertung aktueller Kontextmodelle

In Kap. 2.2.4 ist bereits eine Übersicht der relevanten Kontextmodelle gegeben worden. Die Auswahl der zu betrachtenden Modellierungsansätze erfolgte auf der Basis der in [4] gegebenen Übersicht aktueller Ansätze zur Kontextmodellierung, sowie der Betrachtung weiterer interessanter Veröffentlichungen. Diese werden nachfolgend im Detail beschrieben und in Bezug auf die Anforderungen bewertet.

4.1.1 CAPEUS

Ein Beispiel für Kontextmodelle auf der Basis von Name-Wert-Paaren ist CAPEUS (Context-Aware Packets Enabling Ubiquitous Services) [79][128]. Zielstellung von CAPEUS ist es die kontextadaptive Selektion und Ausführung von Diensten zu unterstützen. Bei der Auswahl von Diensten sollen neben den gewünschten Dienstattributen auch Kontextinformationen berücksichtigt werden. Kern der Lösung ist ein Datenzentrisches Protokoll, welches die Weiterleitung von Anwendungsdaten anhand kontextueller Einschränkungen ermöglicht. Ein Kernkonzept sind dabei die Context-Aware Packets (CAPs). Dieses sind Datenstrukturen, welche zum Zwecke der Dienstvermittlung und –nutzung ausgetauscht werden. Ein Bestandteil hiervon sind die Context Constraints. Diese spielen eine zentrale Rolle, denn sie dienen der Vermittlung von Dienstanforderungen. Dieser Teil bildet das der Lösung zugrundeliegende Kontextmodell ab. Innerhalb der Context Constraints sind die folgenden Modellelemente definiert.

- Entitäten: Die Begriffsdefinition einer Entität unterscheidet sich von der von uns verwendeten Definition nach Dey. Eine Entität wird wie folgt definiert: „Eine Entität ist eine Einheit, welche ein CAP sendet, empfängt, verarbeitet oder Effekt hat auf die Evaluierung eines CAPs“. Eine Entität wird weiter spezialisiert in „Item“, und „Service Endpoint“. In dieser Form ist die Entität bereits spezifisch für den Einsatzzweck der Lö-

sung definiert und nicht so universal wie die Definition nach Dey gehalten.

- Relationen: Auch die Definition dieses Modellelement ist eine spezifische Ausprägung und besitzt keine universelle Einsetzbarkeit: „Eine Relation deklariert räumliche oder temporale Abhängigkeiten zwischen individuellen Entitäten. Eine Relation wird aus einer Menge von Entitäten gebildet.“

Entitäten werden hier in Form von Name-Wert-Paaren repräsentiert. Mit Hilfe dieser einfachen Datenstruktur ist es möglich, Kontextinformationen zu benennen und mit einem Wert zu hinterlegen. Auf diese Weise können verschieden Modellelemente abgebildet werden, die als Anforderungen identifiziert wurden: Kontextentitäten, Kontextattribute, sowie einfache Kontextrelationen. Weitergehende Modellelemente wie beispielsweise die Definition zulässiger Vergleiche zwischen Kontextattributen, die Definition von Situationen und Repräsentationsformen sind auf dieser Datenstruktur nicht möglich.

Die Nutzung des Kontextmodells zur Auswahl von Sensorik wird in CAPEUS nicht betrachtet. Dennoch sollte die Verwendung von Name-Wert-Paaren ausreichend sein, um die Eigenschaften von Sensoren zu beschreiben und für die Auswahl zu verwenden.

Diese einfache Datenstruktur ist sehr generisch und erlaubt daher eine einfache Erweiterung des Kontextmodells im Rahmen der gegebenen Modellelementen „Kontextentität“ und „Kontextrelation“.

Die Trennung zwischen Kontextinfrastruktur und Diensten ist in dem Ansatz nicht vorgesehen. Die einfache Datenstruktur macht es auch nicht möglich. Diese Feststellung gilt auch für die Unterstützung von unterschiedlichen Dienstetypen.

Eine Endnutzer-gerechte Darstellung von Kontextinformationen ist nicht vorgesehen. Das Kontextmodell dient der Selektion und Ausführung im Hintergrund. Das kontextadaptive Verhalten wird programmatisch in Form einer XML-Notation festgelegt. Diese Notation erlaubt die Definition von Events, die zu einer Aktion führen sollen. In nachfolgender Abbildung wird die Definition eines solchen Events für einen Erinnerungsdienst gezeigt. Diese Darstellung ist für den Endanwender ohne technischem Verständnis oder Schulung nicht unmittelbar zu verstehen und beispielsweise für die Konfiguration des kontextadaptiven Verhaltens zu nutzen.

```
<AND>
  <condition subject = "day" object = "monday"  type = "="/>
  <condition subject = "user\_location"
    object = "at_work" type = " ="/>
</AND>
```

Abbildung 8: Event Beispiel: Erinnerung

Der Ansatz von CAPEUS ist auf die Entwicklung der kontextadaptiven Selektion und Ausführung von Diensten beschränkt. Es werden die relevanten Implementations-spezifischen Details beschrieben. Die Fragestellung eines Vorgehensmodells zur Entwicklung von kontextadaptiven Diensten auf dem Lösungsansatz ist nicht Bestandteil der vorgestellten Arbeiten zu CAPEUS. Die sukzessive Bereitstellung von Diensten ist ebenfalls nicht Gegenstand der Betrachtung zu CAPEUS. Entsprechend werden auch nicht mögliche Phasen der Bereitstellung und Nutzung der Dienste betrachtet.

CAPEUS erfüllt die identifizierten Anforderungen an die Kontextmodellierung aus dem AAL nicht. Ein wesentlicher Grund hierfür ist die Beschränkung auf die Modellelemente „Kontextentität“ und „Kontextrelation“ und dem implizit vorhandenen Modellelement „Kontextattribut“, sowie der daraus resultierenden einfachen Repräsentation in Form von Name-Wert-Paaren.

4.1.2 ConteXtML

Die Zielstellung von ConteXtML [83] ist die Bereitstellung eines XML-basierten Protokolls zum Austausch von Kontextinformationen beispielsweise zwischen einem mobilen Client und einem Kontextserver. Bereits 1999 entwickelt wurde dieses Protokoll ständig weiter entwickelt und wird auch noch in vielfältigen Projekten verwendet, z.B. im MobiComp-Projekt [129] im Rahmen der EPOCH FP6 Network of Excellence.

Ein Bestandteil der Schemadefinition von ConteXtML besteht in der Beschreibung des Aufbaus einer Session zur Kommunikation zwischen dem Client und dem Server, sowie dem Austausch von Nachrichten. So besitzt der Client die Möglichkeit den aktuellen Kontext dem Server zu übermitteln und Bedingungen für die Bereitstellung von Informationen durch den Server zu setzen. Zudem kann ein Client den Server nach vorhandenen Kontextinformationen fragen. Ein Teil der Schemadefinition ermöglicht die Beschreibung von Kontextinformationen zu einer Person. Weitere wichtige Modellelemente sind „spatial“

zur Beschreibung von ortsbezogenen Informationen, sowie „temporal“ zur Beschreibung von zeitbezogenen Informationen. Davon abstrahierend kann festgestellt werden, dass die Modellelemente des zugrundeliegenden Kontextmodells auf die Grundkonzepte „Kontextentität“ und „Kontextattribut“ beschränkt sind. Nachfolgend wird ein Beispiel für das Setzen einer Ortsinformation zu einem Benutzer mit Hilfe von ConteXtML gezeigt.

```
<context session="new">
  <person role="user" first="Nick" last="Ryan" />
  <spatial proj="UTM" zone="33" datum="Euro 1950 (mean)">
    <point x="281993" y="4686790" z="205" />
  </spatial>
</context>
```

ConteXtML ist nicht für die Auswahl der geeigneten Sensorik vorgesehen. Entsprechend enthält es keine Elemente mit denen vorhandene Sensorik beschrieben werden kann. Das Konzept „Kontextentität“ findet sich nicht explizit in der Schemadefinition wieder. Eine Erweiterung des Kontextmodells funktioniert nur durch eine Erweiterung der Schemadefinition. Eine Unterscheidung zwischen unterschiedlichen Kontextmodellen, z.B. der Infrastruktur oder den Dienstetypen ist nicht möglich. Eine Endnutzer-gerechte Darstellung ist aufgrund der Zielstellung nicht vorgesehen. ConteXtML ist als fertiges Protokoll mitsamt definiertem Kontextmodell im Betreib von kontextadaptiven mobilen Anwendungen einzusetzen. Eine Betrachtung der Entwicklungs- und Nutzungsphasen entfällt bei diesem Ansatz.

ConteXtML erfüllt die identifizierten Anforderungen an die Kontextmodellierung aus dem AAL nicht. Dieses resultiert aus der speziellen Zielstellung des Protokolls und der Beschränkung auf wenige spezielle Kontextentitäten und Kontextattribute des definierten Kontextmodells.

4.1.3 NEXUS

Ein weiteres Beispiel für XML-basierte Kontextmodelle ist die Beschreibungssprache für Kontextinformationen in NEXUS [130]. NEXUS wurde im Sonderforschungsbereich 627 der Universität Stuttgart entwickelt. Ziel dieses Sonderforschungsbereichs ist die Erforschung von Methoden und Verfahren für die Definition, die Verwaltung und die Nutzung von umfangreichen und detaillierten räumlichen Modellen unserer physischen Umgebung. Die Architektur von NEXUS besteht aus drei Ebenen. In einer obersten Ebene sind die Anwendungen angesiedelt, die auf der Plattform aufsetzen. NEXUS stellt in seiner mittleren Ebene einen Föderierungsmechanismus zur Verfügung, welcher die Bereitstellung von Ortsinformationen in unterschiedlichen Diensten erlaubt. Diese werden in einer zentralen Instanz registriert und vermittelt. Die eigentli-

chen ortsbezogenen Informationen sind in der unteren Diensteschicht enthalten. Dort ermöglicht NEXUS die Ad-hoc-Integration von Diensten.

Der wesentliche Fokus des Kontextmodells von NEXUS liegt auf einem komplexen Ortsmodell. Dieses kann sowohl georeferenziert als auch symbolisch sein [58]. Das Kontextmodell baut auf den primären Dimensionen „Identität“, „Zeit“ und „Ort“ auf. Eine Kontextentität kann anhand dieser primären Dimensionen beschrieben werden, z.B. Herr Meyer befindet sich am NEXUS-Stand der Messe Stuttgart. Der Zeitbezug kann sowohl auf die aktuelle, als auch auf die vergangene und zukünftige Zeit definiert werden. Somit kann beispielsweise ermittelt werden welche Personen sich zu einem bestimmten Zeitpunkt an einem bestimmten Ort befinden. Diese Grundlagen des Kontextmodells werden in nachfolgender Abbildung verdeutlicht [131].

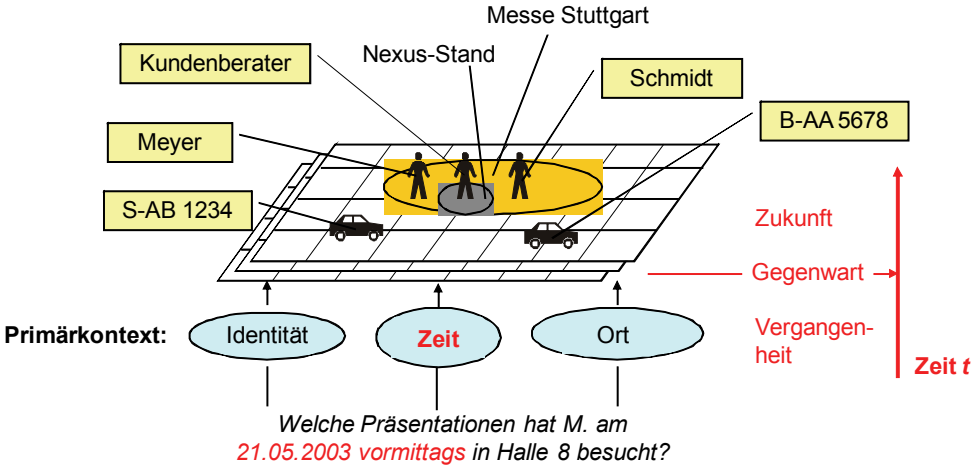


Abbildung 9: Ortsmodell und Primärkontext in NEXUS

Neben dem Primärkontext können noch weitere Kontextattribute definiert werden, die als Sekundärkontext bezeichnet werden. Oberhalb eines solchen Kontextmodells ist ein „Reasoning Layer“ vorgesehen, in welchem die im Kontextmodell vorhandenen Kontextinformationen verwendet werden können, um neue komplexe Kontextinformationen abzuleiten. Diese werden in das Kontextmodell wieder zurückgespielt. Dieser Layer kann beispielsweise durch eine Inference Engine realisiert werden, in denen Ontologien zur Beschreibung der Zusammenhänge verwendet werden können. Die Verwendung des „Reasoning Layer“ wird in nachfolgender Abbildung dargestellt [58].

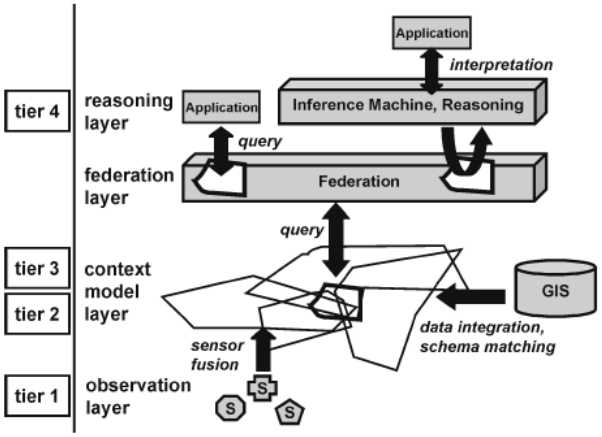


Abbildung 10: Verwendung des „Reasoning Layer“ in NEXUS

Das Kontextmodell enthält die Modellelemente „Kontextentität“ und „Kontextattribut“. Die Verbindung zwischen Kontextentitäten erfolgt über die Primärdimensionen, über welche diese zueinander in Bezug gesetzt werden können. Das komplexe Ortsmodell von Nexus ermöglicht die Unterscheidung zwischen unterschiedlichen Repräsentationsformen eines Ortes. Explizite Kontextrelationen, die über eine Verbindung über die Primärdimensionen hinausgehen, sind im Kontextmodell nicht vorgesehen. Vergleichsmöglichkeiten über die Primärdimensionen hinausgehend sind ebenfalls nicht vorgesehen. Auch die Beschreibung und Ableitung von Situationen ist kein Bestandteil des Kontextmodells. Letztere könnten aber über den „Reasoning Layer“ durchgeführt werden, was aber in der verfügbaren Literatur nicht beschrieben ist. Gleiches gilt für die semantische Anreicherung des Kontextmodells um beispielsweise das Rollenkonzept.

Die Nutzung des Kontextmodells zur dynamischen Bereitstellung und Auswahl von Sensorik ist eine definierte Zielstellung von NEXUS. Die Erweiterbarkeit des konkreten Kontextmodells ist ebenfalls gegeben. Eine Trennung zwischen Kontextinfrastruktur und Diensten und die Unterstützung spezifischer Kontextmodelle für verschiedene Dienstetypen ist nicht gegeben.

Die Endnutzer-gerechte Darstellung von Kontextinformationen ist in NEXUS nicht gegeben. Aufgrund des Fokus auf Ortsmodelle ist es denkbar, dass ortsbezogene Kontextinformationen sehr wohl Endnutzer-gerecht aufbereitet und dargestellt werden können, beispielsweise durch die Darstellung von Positionen verschiedener Entitäten auf einer Karte. Für generelle Kontextinformationen ist eine solche Darstellung jedoch nicht ausreichend.

Veröffentlichungen in Bezug auf die Realisierungsmethodik von kontextadaptiven Diensten und der Kontextinfrastruktur auf Basis von NEXUS existieren nicht. Auch die Bereitstellung und Nutzung solcher Dienste wird nicht beschrieben.

Ein Bestandteil von NEXUS ist die Beschreibungssprache für konkrete Kontextinformationen. Sie basiert auf XML und definiert im Schema die Elemente des Modells und kann erweitert werden. In ihm sind die möglichen Typen von Sensoren und Kontextinformationen definiert sowie die zulässigen Tags bzw. Kontextattribute festgelegt. Sowohl zur Bereitstellung der Kontextinformationen als auch zur Qualität eines Messwerts werden Metainformationen bereitgestellt.

NEXUS unterstützt bereits eine Vielzahl von notwendigen Modellelementen für die Erstellung konkreter Kontextmodelle für das AAL. Ein Vorteil ist hier insbesondere das komplexe Ortsmodell, welches sowohl die Georeferenzierung als auch symbolische Ortsmodelle ermöglicht. Kein Bestandteil des Kontextmodells sind aber Kontextrelationen und Situationen, sowie die Ermöglichung von unterschiedlichen Repräsentationsformen auf nicht ortsbezogenen Kontextdimensionen. Die Erkennung von Situationen könnte auf Ebene des „Reasoning Layer“ ergänzt werden. Die Trennung zwischen Kontextinfrastruktur und Diensten, die Unterstützung spezifischer Kontextmodelle für einzelne Dienstetypen, die Endnutzer-gerechte Darstellung von Kontextinformationen, sowie die Unterstützung unterschiedlicher Entwicklungs- und Nutzungsphasen sind weitere nicht unterstützte Anforderungen aus dem AAL.

4.1.4 Erweitertes Object-Role Model

Als Beispiel für graphische Modelle wird das erweiterte Object-Role Model [101] von Henricksen, Indulska und Rakotonirainy betrachtet. Es erweitert die im originalen Object-Role Model vorhandenen Modellelemente um zusätzliche Konzepte zur Kontextmodellierung, die eine graphische Repräsentation der Modellelemente beinhaltet. Mit Hilfe der in der graphischen Repräsentation definierten Symbole können Kontextinformationen und deren Beziehungen zueinander konzeptuell und implementierungsunabhängig beschrieben werden.

Das Kontextmodell hat einen sehr starken Bezug zur eingesetzten Sensorik und deren Eigenschaften. Es kann beschrieben werden welche Kontextinformationen ein Kontextmodell bereitstellt, aus welcher Sensorik diese verfügbar gemacht werden und welche Eigenschaften diese besitzen. Das Kontextmodell ist daher eng an die Sensorik gekoppelt und unterstützt daher nicht die Trennung zwischen Kontextinfrastruktur und Diensten. Das auf diese Weise definierte Kontextmodell dient dann als Ausgangspunkt für die konkrete Implementierung durch einen Entwickler, beispielsweise auf der Basis eines relationalen Datenbankmanagentsystems. In [132] beschreiben die Autoren eine

Überführung des konzeptuellen Kontextmodells in eine Ontologie. Somit werden die Phasen der Entwicklung und Nutzung zum Teil berücksichtigt.

Das Kontextmodell von Henricksen, Indulska und Rakotonirainy definiert eine Reihe von Kernkonzepten, die für die Erstellung eines solchen Kontextmodells relevant sind. Eines davon ist die Darstellung von Entitäten und der zugehörigen Attribute. Dieses entspricht der „Kontextentität“ und dem „Kontextattribut“. Eine Entität wird grafisch durch ein doppelt gerahmtes Rechteck repräsentiert, während das Attribut durch ein einfaches Rechteck dargestellt wird. Ein Attribut wird einer Entität durch eine Assoziation zugeordnet, die durch einen Pfeil darzustellen ist.

Beziehungen zwischen Entitäten können ebenfalls modelliert werden, was durch die Definition von Assoziationen zwischen Entitäten erfolgt. Eine solche Assoziation entspricht einer „Kontextrelation“. Eine Assoziation kann somit sowohl Attribute ihren Entitäten zuordnen als auch die Beziehung zwischen Kontextentitäten selbst beschreiben.

Die Klassifikation von Assoziationen ist ein wesentliches Konzeptelement dieses Ansatzes. Über eine solche kann eine Aussage zur Eigenschaft der Beziehung gemacht werden. Diese Eigenschaften werden aus unterschiedlichen Blinkwinkeln beschrieben. Ein Betrachtungspunkt ist hierbei die Stabilität einer solchen Assoziation. Es wird unterschieden zwischen statischen und dynamischen Assoziationen. Erstere bleiben über die gesamte Lebenspanne des Kontextmodells stabil, wobei letztere sich verändern können. Statische und dynamische Assoziationen werden grafisch über unterschiedliche Ausprägungen des Pfeils repräsentiert. Ein weiterer Betrachtungswinkel ist die Erhebung der Kontextinformation, die der Assoziation zugrunde liegt. Es wird zwischen der Erhebung mittels Sensorik, der Übernahme aus Profilinformationen und der Ableitung aus vorhandenem Regelwissen unterschieden. Dieser Betrachtungswinkel ist nach Ansicht der Autoren deshalb interessant, weil sich darüber Aussagen zur Qualität der Kontextinformationen ableiten lassen. Auch diese Merkmale werden durch eine unterschiedliche Ausprägung des Pfeils grafisch repräsentiert. Ein letzter Betrachtungswinkel sind die strukturellen Bedingungen auf den Assoziationen. Hier kann zwischen einfachen, mehrfachen, alternativen und temporalen Assoziationen unterschieden werden. Sie ist einfach, wenn sie nur einmal der zugehörigen Kontextentität zugeordnet werden kann. So kann beispielsweise das Kontextattribut „Körpertemperatur“ die Kontextentität „Bewohner“ nur einmal kennzeichnen. Eine mehrfache Assoziation beschreibt dagegen eine Beziehung, die der zugehörigen Kontextentität mehrfach zugeordnet werden kann. Dies ist beispielsweise bei einer Assoziation „besitzt“ zwischen den Kontextentitäten „Person“ und „Handy“ der Fall. Eine alternative Assoziation beschreibt mehrere mögliche Beziehungen zwischen Kontextentitäten, bei der nur eine wirklich benötigt wird. Sie ist dann sinnvoll, wenn beispielsweise Kontextinformationen von unterschiedlichen Quellen mit unterschiedlicher Qualität erhoben werden. In diesem Fall kann die Kontextin-

formation aus der Quelle mit der höchsten Qualität übernommen, während die anderen verfügbaren Quellen verworfen werden. Eine temporale Assoziation beschreibt ebenfalls mehrfache Beziehungen, jedoch mit einem zeitlichen Gültigkeitswert für jede der Assoziation. Auch diese Merkmale werden über eine unterschiedliche Ausprägung des Pfeils grafisch repräsentiert.

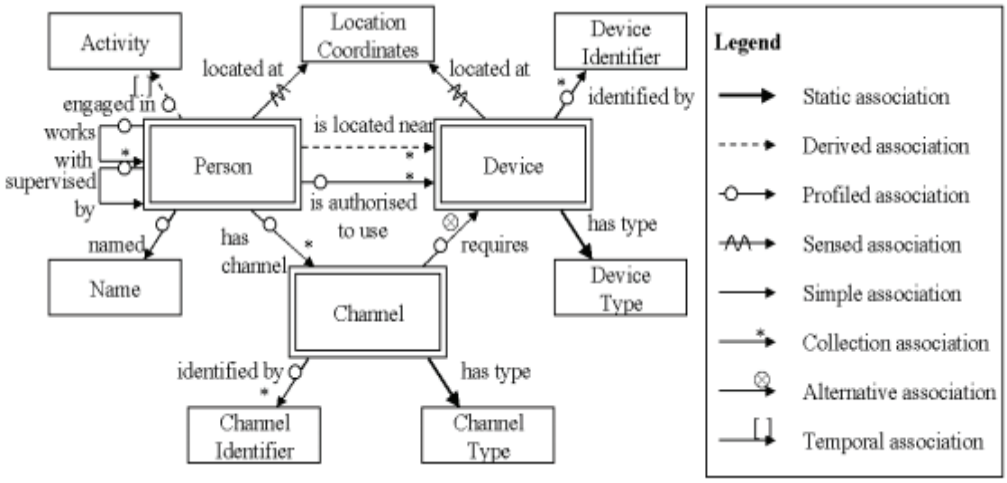


Abbildung 11: Beispiel eines konzeptuellen Kontextmodells nach Henricksen, Indulska und Rakotonirainy

Ein weiteres Konzeptelement ist die Definition von Abhängigkeiten zwischen Assoziationen. Hiermit ist gemeint, dass mit der Änderung in einer Assoziation die von ihr abhängige ebenfalls verändert sein kann. Ein Beispiel hierfür ist die Abhängigkeit der Lebenszeit einer Batterie von der verfügbaren Bandbreite. Sie wird durch einen gestrichelten Pfeil dargestellt, der zwei Assoziationen verbindet. Solche Assoziationen können als Regelwissen verwendet werden.

Die Modellierung der Qualität von Kontextinformationen ist ein weiteres Konzeptelement. So können Qualitätsparameter und die zugehörigen Metriken beschrieben werden. Sie werden durch ein Oval repräsentiert und durch Verbindungslinien den entsprechenden Assoziationen zugeordnet.

Das erweiterte Object-Role Model unterstützt vielfältige Modellelemente, die für die Kontextmodellierung im AAL benötigt werden. Nicht unterstützte Modellelemente sind jedoch „Situation“ und „Repräsentationsform“. Das Kontextmodell ist konzeptueller Art und kann daher so nicht operativ für die Auswahl von Sensorik oder von AAL-Diensten genutzt werden. Die Beschreibung der Sensorik ist jedoch expliziter Bestandteil des Kontextmodells. Aufgrund der expliziten Definition der benötigten Modellelemente kann ein konkretes Kontextmodell erweitert werden. Jedoch ist eine Trennung zwischen Kontextinfrastruktur und Diensten nicht möglich. Die Unterstützung spezifischer Kontext-

modelle für einzelne Dienstetypen ist nicht vorgesehen. Die Darstellung der Kontextmodelle richtet sich an die Domänenexperten für die konzeptuelle Erstellung eines solchen Modells. Der Endnutzer ist nicht Zielpunkt der Darstellung. Es erfolgt keine durchgängige Unterstützung der unterschiedlichen Entwicklungs- und Nutzungsphasen der Kontextinfrastruktur und der AAL-Dienste. Die Betrachtung ist beschränkt auf die Konzeption und den Übergang in die Implementierung des Kontextmodells.

4.1.5 ECA-Modell

Ein weiteres Beispiel für ein graphisches Modell ist das ECA-Modell [125]. Die Zielstellung dieses Kontextmodells ist die Konfiguration des kontextadaptiven Verhaltens einer Anwendung durch den Endanwender. Die Grundidee hierbei ist es die Elemente des Kontextmodells durch grafische Symbole zu repräsentieren, die der Endanwender mittels Drag & Drop zu komplexen Regeln miteinander verknüpfen kann.

In diesem Ansatz wird die kontextadaptive Anwendungslogik eines Dienstes über eine Reihe von Event-Condition-Action-Regeln (ECA) definiert. Events entsprechen dabei Signalen, die von einzelnen Kontextsensoren geliefert werden. Sie können im Condition-Teil mittels boolscher Algebra zu komplexen Bedingungen miteinander verknüpft werden. Im Action-Teil wird das Verhalten der Anwendung bei Erfüllung der Bedingung beschrieben. Folgendes Beispiel für eine ECA-Regel wird von den Autoren für ein Frühstücksszenario gegeben:

Event part:	*When the wheelchair with RFID-tag is detected in the kitchen,*
Condition part:	*(and) it is the first time between 7 am and 8 am,*
Action part:	*(then) turn on the kitchen light,*
	(and) turn on the kitchen coffee maker,
	(and) displays the morning news on the kitchen video screen.

Das Metamodell dieses Ansatzes sieht ein Modellelement „Event“ zur Beschreibung von Kontextereignissen vor. Hierfür gibt es eine grafische Repräsentation. Mit diesem Modellelement können Ereignisse beschrieben werden, die sich Kontextentitäten beziehen. Dennoch sind Kontextentitäten und Kontextattribute kein expliziter Bestandteil des Kontextmodells. Gleiches gilt für die sonstigen Modellelemente, die für die Erstellung konkreter Kontextmodelle im AAL benötigt werden, z.B. Kontextrelationen, Situationen und Repräsentationsformen.

Über einen graphischen Regeleditor kann der Benutzer bestehende Regeln zu einer graphischen Repräsentation der einzelnen Regelbestandteile modifizieren oder neue definieren. So existieren Symbole für einzelne Kontexttypen, z.B. Zeit oder die Ortung von Gegenständen. Zusätzlich stehen Symbole für die boolschen Operatoren zur Verfügung. Auf der Basis eines Puzzle-

Paradigmas können diese Symbole miteinander verknüpft werden. An die darüber definierten komplexen Bedingungen können ebenfalls mittels grafischer Symbole die möglichen Aktionen an das Puzzle angehängt werden. Eine das Frühstücksszenario umsetzende grafische Repräsentation des Regelwerks auf der Benutzerschnittstelle wird in der folgenden Abbildung gezeigt. Die Beschreibung des kontextadaptiven Verhaltens durch den Nutzer erfolgt in der Benutzeroberfläche durch Drag & Drop der vordefinierten graphischen Repräsentationen von Modellelementen.

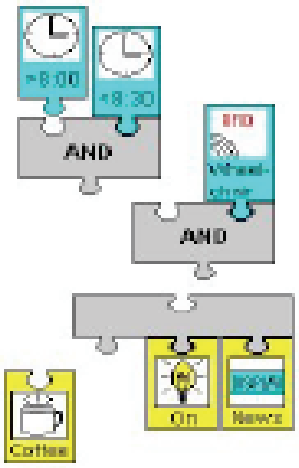


Abbildung 12: Grafische Repräsentation eines ECA-Regelwerks [125]

Dieses Kontextmodell ist sehr einfach gehalten und unterstützt die in den Anforderungen identifizierten Modellelemente nicht. Dieses ist aber der Tatsache geschuldet, dass die Zielstellung hier die Endnutzer-gerechte Darstellung von Kontextinformationen ist, in der eine möglichst einfache Sicht und Abstraktion von technischen Details gefordert ist. Das Kontextmodell kann nicht zur Auswahl der Sensorik verwendet werden. Die Erweiterbarkeit ist durch das Modellelement „Event“ gegeben, welches beliebig instanziiert werden kann. Eine Trennung zwischen Kontextinfrastruktur und Diensten ist nicht gegeben. Vielmehr ist in der Darstellung der ECA-Regelwerke durch die Autoren eine enge Kopplung zwischen Sensoren und Events zu geben. Eine Unterscheidung zwischen verschiedenen Dienstetypen ist nicht gegeben. Der Modellierungsansatz betrachtet nicht die unterschiedlichen Entwicklungs- und Nutzungsphasen einer Kontextinfrastruktur und der AAL-Dienste.

4.1.6 Active Object Model

Das Active Object Model ist ein Beispiel für ein objektorientiertes Kontextmodell. Dieses wurde in Zusammenhang mit dem kontextadaptiven Touristeninformationssystem GUIDE [68] der Universität Lancaster entwickelt. Kontextinformationen werden hier benötigt, um dem Touristen Informationen zu Sehenswürdigkeiten in seiner Umgebung zu übermitteln. Die Kontextinformationen die zur Auswahl benötigt werden, lassen sich zwischen persönlichen und Umgebungsinformationen unterscheiden [87]. Personönliche Informationen sind beispielsweise die Interessen des Besuchers, seine aktuelle Position, die Zeitspanne die für einen Besuch vorgesehen, das Budget oder die Präferenzen für Erfrischungen. Umgebungsinformationen beinhalten die aktuelle Zeit, das Wetter, die Jahreszeit und der aktuelle Zustand im öffentlichen Nahverkehr. Diese Kontextinformationen werden durch Java-Klassen repräsentiert. Es ist ein Framework definiert, in denen das Zusammenspiel der verschiedenen Klassen zur Realisierung der GUIDE-Anwendung vorgegeben ist. Dieses wird in der folgenden Abbildung dargestellt.

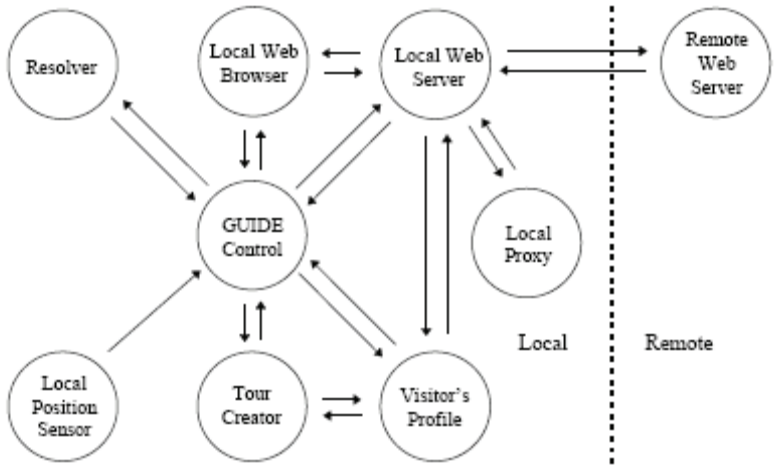


Abbildung 13: Das GUIDE Objektmodell

Das Kontextmodell ist hier in den verschiedenen Objekten „Local Position Sensor“ bzw. „Vistor’s Profile“ implementiert. Werden weitere Kontextinformationen benötigt, so können diese über zu implementierende Objekte bereitgestellt werden. Diese verkapseln den Zugriff auf die konkrete Sensorik und stellen den Zugriff auf die Kontextinformationen über definierte Schnittstellen bereit. So ist beispielsweise für das Objekt „Local Position Sensor“ die Funktion „receiveNewPosition“ definiert, über welches das zentrale „GUIDE Control“-Objekt benachrichtigt werden kann, wenn den Besucher an einer neuen Position erkannt wurde. Möchte das „GUIDE Control“-Objekt auf die präferierte Sprache des Besuchers zugreifen, so erfolgt dieses über die Methode „return-

PreferredLanguage“ des Objekts „Visitor’s Profile“. In diesen Objekten ist die Gewinnung von Kontextinformationen über Sensorik, sowie deren Verdichtung verkapselt. Weiterhin kann mittels der Vererbung gemeinsame Eigenschaften von Kontextentitäten beschrieben werden.

Das Kontextmodell definiert nicht explizit die Modellelemente „Kontextentität“ und „Kontextattribut“. Die Objekte des „Active Object Model“ liefern verschiedenste Kontextinformationen. Der Bezug zu einer Kontextentität oder einer Kontextrelation ist implizit in der Implementierung der Objekte verknüpft. Zulässige Vergleiche zwischen Kontextattributen, Situationen und Repräsentationsformen sind ebenfalls kein Bestandteil des Kontextmodells.

Das Kontextmodell dient nicht der Auswahl der verfügbaren Sensorik. Die Sensorik ist eng mit den Objekten verknüpft. Entsprechend ist keine technische Trennung zwischen Kontextinfrastruktur und Diensten vorhanden. Die Sensorik ist unmittelbarer Bestandteil der GUIDE Anwendung. Eine Unterstützung spezifischer Kontextmodelle für einzelne Dienstetypen ist nicht vorgesehen. Die Darstellung von Ortsinformationen durch Fotos ist Bestandteil des Kontextmodells. Dadurch soll es dem Anwender erleichtert werden sich in der Umgebung zurechtzufinden und der Anwendung mitzuteilen wo genau er sich befindet. Die Endnutzer-gerechte Darstellung ist allerdings auf den Ortsaspekt beschränkt und daher nicht generell einsetzbar. Eine Betrachtung der unterschiedlichen Entwicklungs- und Nutzungsphasen ist im „Active Object Model“ nicht gegeben.

4.1.7 CoDaMoS

Logikbasierte Modelle spielen aktuell in der Kontextmodellierung keine Rolle. Diese werden daher nachfolgend auch nicht weiter betrachtet. Dafür gibt es eine Vielzahl von Kontextmodellen, die auf der Verwendung von Ontologien basieren. Ein Beispiel hierfür ist das ontologiebasierte Kontextmodell [114], welches im CoDAMoS-Projekt [133] eingesetzt wurde. Die Zielstellung des Projekts ist die Unterstützung der kontextabhängigen Adaption mobiler Dienste im Ambient Intelligence. Es soll im Rahmen des Projekts eine Kontextinfrastruktur für das „Ambient Intelligence“ aufgebaut werden, welche die folgenden Anforderungen erfüllt:

- Adaptivität der Anwendung: Auf Basis der Informationen über den Anwender, die verfügbaren Dienste, die zugrundeliegende Hardware, die verfügbare Netzwerkverbindung, Zeit, Ort und weitere Kontextinformationen soll das Kontextmodell die geeignete Adaption der Anwendung unterstützen.

- Berücksichtigung der Resourcen: Informationen über verfügbare Resourcen und ihrer Eigenschaften, z.B. Rechenkraft, Speicher, Batterie-

stand und Bandbreite, sollen Dienste adaptiert bzw. dynamisch geeigneten Rechenknoten zugeordnet werden.

- Mobile Dienste: Dienste müssen in der Lage sein gemeinsam mit dem Anwender sich zu bewegen und entsprechend auf den Knoten der verfügbaren Infrastruktur zu migrieren.

- Semantische Diensteerkennung: Auf Basis von Kontextinformationen sollen automatisch Dienste ermittelt werden, die für den Anwender relevant sind.

- Codegenerierung: Aufgrund unterschiedlicher Ausstattung einzelner Rechnerknoten in Bezug auf Hard- und Software soll es möglich sein spezifischen Code zu generieren, welcher auf dem ausgewählten Knoten lauffähig ist. Hierfür muss dass Kontextmodell Informationen über den entsprechenden Knoten bereitstellen.

- Kontextadaptive Nutzeroberfläche: Die Nutzeroberfläche muss sich an den jeweiligen technischen Einschränkungen mobiler oder sonstiger Endgeräte anpassen. Mobile Endgeräte mit kleinem Bildschirm benötigen eine andere Oberfläche als stationäre Endgeräte mit einem entsprechend größerem Bildschirm. Hierfür sind entsprechende Kontextinformationen bereitzustellen.

Das zugrundeliegende Kontextmodell basiert auf einer Ontologie. In dieser sind viel Hauptentitäten definiert: Person, Umgebung, Plattform und Dienst und unterstützt so direkt die Anforderungen, die sich bei der Beschreibung der Infrastruktur ergeben. Eine Übersicht der Ontologie wird in nachfolgender Abbildung gegeben.

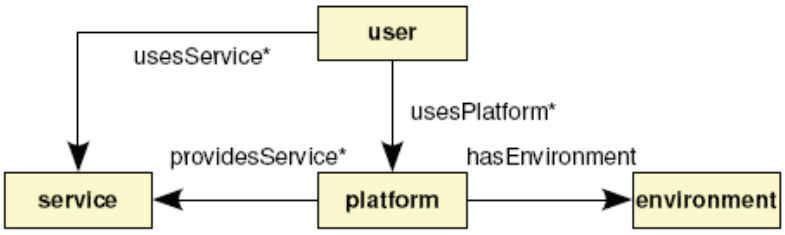


Abbildung 14: Überblick der Kontextontologie

Jede dieser Entitäten wird noch weiter verfeinert. Die Abhängigkeiten zwischen diesen Entitäten werden in einer Ontologie beschrieben, wobei einer Person unterschiedliche Eigenschaften und Aufgaben zuweisbar sind. Weiterhin ist dem Benutzer ein Profil zugeordnet und er kann ein mobiles Endgerät verwenden. In der Umgebung sind die Aspekte Ort, Zeit und Umgebungsbedingungen, wie Temperatur oder Helligkeit vorgehalten. Die Plattform beschreibt die Umgebung für die Ausführung von Diensten, bezogen auf Soft-

ware und Hardware. Zur Beschreibung der Software gehören Betriebssystem, Middleware oder virtuelle Maschine, zu der von Hardware die CPU-Leistung, Speichergröße oder auch die verfügbaren Endgeräte. Die Dienstebeschreibung ist aufgegliedert in „Service Profil“, „Service Model“ und „Service Grounding“. In nachfolgender Abbildung wird die Verfeinerung der Entität „Environment“ dargestellt.

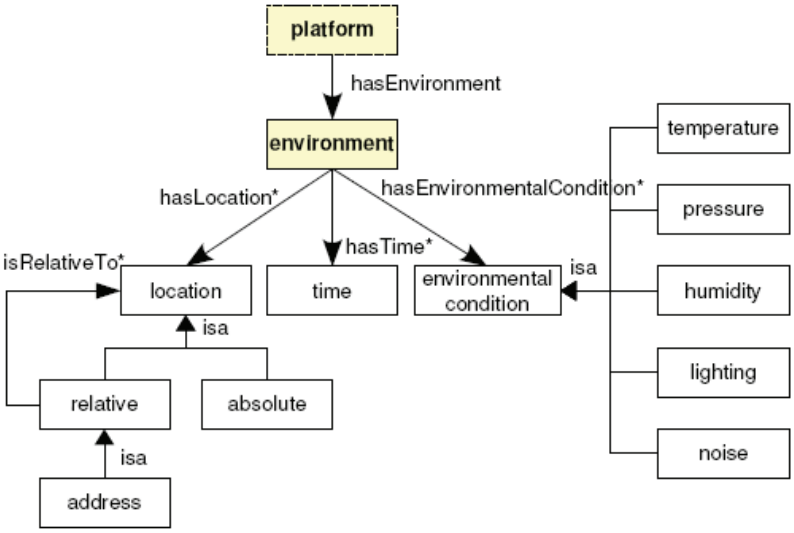


Abbildung 15: Verfeinerung der Entität „Environment“

Die Ontologien sind mit Hilfe der Web Ontology Language (OWL) [134] definiert. OWL definiert Schema Features bzw. Sprachelemente, mit denen Ontologien beschrieben werden können. Ein wesentliches Feature ist das Sprachelement „Class“. Dieses definiert eine Gruppe von Individuen, die aufgrund gemeinsamer Eigenschaften zusammengehörend sind. Über dieses Sprachelement lässt sich das Modellelement „Kontextentität“ abbilden. Das OWL Sprachelement „rdfs:subClassOf“ erlaubt die Definition von Generalisierungs- und Spezialisierungsbeziehungen zwischen Klassen und erfüllt damit ebenfalls die Anforderungen an die Kontextmodellierung aus dem AAL. Das OWL Sprachelement „rdfs:Property“ kann verwendet werden, um Beziehungen zwischen einzelnen Individuen oder zwischen Individuen und Datenwerten zu beschreiben. So kann dieses verwendet werden für die Modellelemente „Kontextrelation“ und „Kontextattribut“. Diese Sprachelemente werden in dem CoDaMoS-Kontextmodell verwendet. Zulässige Vergleiche zwischen Kontextattributen können nicht ausgedrückt werden. Die Definition von Situationen und Repräsentationsformen von Kontextattributen ist ebenfalls nicht auf Ebene der OWL Sprachelemente möglich.

Laut Zielstellung des CoDaMoS-Projekt soll das Kontextmodell zur Auswahl der geeigneten Sensorik genutzt werden können. Die Erweiterbarkeit des Kontextmodells ist durch die Verwendung der entsprechenden OWL Sprachele-

mente möglich. Eine Trennung zwischen Kontextinfrastruktur und Diensten ist in dem Ansatz nicht vorgesehen. Auch eine Unterstützung spezifischer Kontextmodelle für einzelne Dienstetypen ist nicht vorgesehen. Man könnte dieses umgehen, indem ein Dienstetyp als Konzept in das Kontextmodell aufgenommen wird und die Kontextentitäten dem jeweiligen Dienstetyp zugeordnet werden. Ungünstig bei diesem Vorgehen ist jedoch die Vermischung der Modellebenen. Eine Endnutzer-gerechte Darstellung des Kontextmodells wird nicht betrachtet. Die Unterstützung der unterschiedlichen Entwicklungs- und Nutzungsphasen der Kontextinfrastruktur und der AAL-Dienste wird bei dem Ansatz ebenfalls nicht explizit betrachtet. Da die CoDaMoS Kontextmodellierung auf Ontologien basiert, können eventuell vorhandenen Vorgehensmodelle für die Entwicklung und Bereitstellung semantischer Anwendungen verwendet werden.

4.1.8 CONON

Als ein weiteres Beispiel soll die CONtext ONtology (CONON) [95] betrachtet werden. Sie beschreibt eine erweiterbare Ontologie zur Kontextmodellierung in einer pervasiven Umgebung. Als Beschreibungssprache für die Ontologien verwenden die Autoren die Web Ontology Language (OWL). Als Motivation dafür geben sie folgende Gründe an:

- Das Teilen von Wissen zwischen unterschiedlichen technischen Komponenten einer pervasiven Umgebung, z.B. Agenten und Dienste auf Basis der definierten Begrifflichkeiten.
- Die Möglichkeit mit Hilfe des semantischen Wissens mit Hilfe der Inferenz aus einfachen Kontextinformationen höherwertige Kontextinformationen abzuleiten.
- Existierendes Wissen in unterschiedlichen Domänen wiederzuverwerten.

CONON definiert eine „Upper Ontology“ in welcher die grundlegenden Kontextentitäten und deren Eigenschaften definiert sind. Spezifische Kontextmodelle können darauf aufsetzen und in einer „Domain-Specific Ontology“ um eigene Konzepte ergänzen. Dieses Vorgehen ist in nachfolgender Abbildung dargestellt.

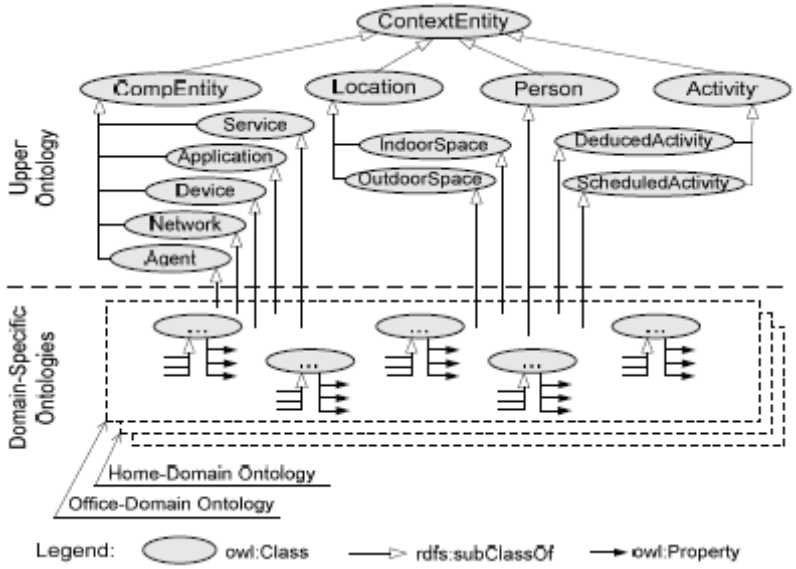


Abbildung 16: Aufteilung in Upper Ontology und Domain-Specific Ontology

In der folgenden Abbildung ist ein Beispiel für eine domänenspezfische Ontologie gegeben.

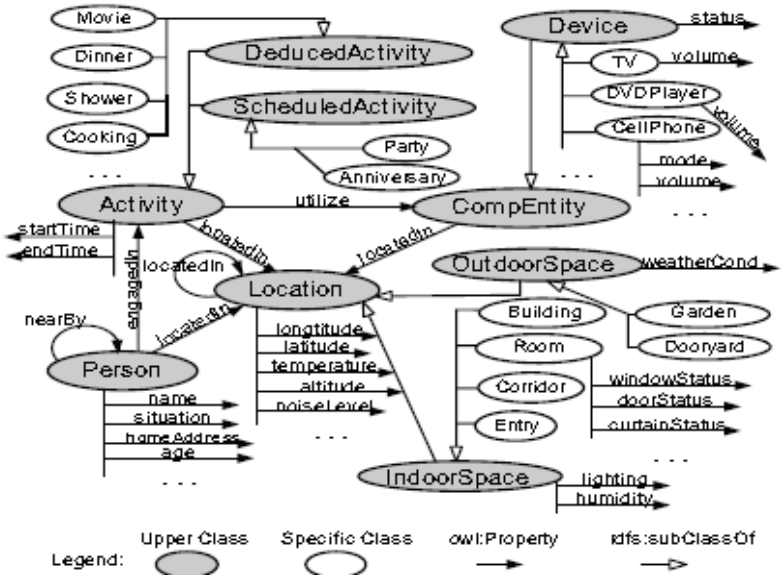


Abbildung 17: Beispiel eines CONON Kontextmodells

Die Bewertung des CONON Kontextmodells entspricht weitgehend der dem des CoDaMoS Kontextmodells, da beide auf der Definition von Kontextmodellen mit Hilfe von OWL basieren. Dieses gilt insbesondere für die Abbildung von Modellelementen auf die Sprachelemente von OWL. Im Gegensatz zu CoDaMoS kann jedoch in CONON das Konzept der domänenspezifischen Kontextmodelle verwendet werden, um spezifische Kontextmodelle für die jeweiligen Dienstetypen zu realisieren. Die Beschreibung der Sensorik für die Auswahl geeigneter Sensoren ist jedoch explizite Zielstellung von CONON und findet sich daher auch nicht unmittelbar im Kontextmodell wieder.

4.1.9 Zusammenfassung

Keine der betrachteten Kontextmodelle erfüllt die Anforderungen an die Kontextmodellierung aus der Anwendungsdomäne AAL ganz. Es gibt Unterschiede in der Ausdrucksmächtigkeit der Modellierungskonzepte. Die Ontologiebasierten Kontextmodelle kommen den Anforderungen am nächsten. Repräsentationsformen und Situationen lassen sich in diesen nicht auf gleicher Ebene wie die Modellelemente „Kontextentität“ oder „Kontextrelation“ modellieren. Die Nutzung des Kontextmodells zur Auswahl von Sensorik unterstützen Kontextmodelle, die auf den Bereich „Ambient Intelligence“ spezialisiert sind. Die meisten Ansätze unterstützen die Erweiterbarkeit eines Kontextmodells. Keine der betrachteten Kontextmodelle unterstützt die Trennung zwischen Kontextinfrastruktur und Diensten. Die Unterstützung spezifischer Kontextmodelle für einzelne Dienstetypen kann lediglich durch CONON realisiert werden. Die Endnutzer-gerechte Darstellung ist explizite Zielstellung des ECA-Modells. Keine der bestehenden Modelle ermöglicht eine durchgehende Unterstützung der Entwicklungsphasen, der Nutzung der Kontextinfrastruktur sowie der AAL-Dienste. Die meisten der Modelle sind beschränkt auf die Implementierung und den Betrieb. Das erweiterte Object-Role Model unterstützt die Designphase und den Übergang in die Implementierung. Dieses Ergebnis wird in nachfolgender Tabelle zusammengefasst. Zusätzlich wird beschrieben ob das jeweilige Kontextmodell in der konkreten Ausprägung neben den generellen Eigenschaften für die Beschreibung der Kontextinfrastruktur bzw. der dienstetypspezifsichen Kontextmodelle verwendet werden kann. Dazu muss es die Anforderungen an die konkreten Kontextmodelle AK1 – AK3 bzw. AK6 erfüllen.

Tabelle 16: Bewertung aktueller Kontextmodelle

	CAPEUS	ConteXtML	NEXUS	Erweitertes ORM	ECA-Modell	Active Object Model	CoDAMos	CONON
Konkrete Ausprägung								
Infrastruktur	N	N	N	N	N	N	J	N
Dienste	N	N	T	N	N	N	N	T
Generelle Eigenschaften								
Bereitstellung von Modellelementen für die Kontextmodellierung	T	T	T	T	T	T	T	T
Auswahl von Sensorik	J	N	J	N	N	N	J	N
Erweiterbarkeit	J	N	J	J	J	J	J	J
Trennung Infrastruktur und Dienst	N	N	N	N	N	N	N	N
Dienstetyp-spezifische Kontextmodelle	N	N	N	N	N	N	N	J
Endnutzergerechte Interaktion	N	N	N	N	J	N	N	N
Unterstützung unterschiedlicher Phasen der Entwicklung und Nutzung	N	N	N	T	N	N	N	N

J = Anforderung erfüllt, N = Anforderung nicht erfüllt, T = Anforderung teilweise erfüllt

4.2 Mehrdimensionale Kontextmodellierung für das AAL

Aus der Bewertung der aktuellen Kontextmodelle wird ersichtlich, dass keines von ihnen die Anforderungen an die konkrete Ausprägung aus Sicht der Infrastruktur und der Dienste ganz erfüllt. Abgesehen von der konkreten Ausprägung der Kontextmodelle in Bezug auf die Beschreibung der Infrastruktur oder die dienstespezifischen Kontextmodelle gilt dieses insbesondere für die generellen Eigenschaften der Kontextmodellierung. Keines der vorgestellten Kontextmodelle kommt den Anforderungen nahe genug, um diese durch Ergänzungen durch die benötigten Modellelemente erfüllen zu können.

Die Anforderungen an die generellen Eigenschaften eines Kontextmodells im AAL unterscheiden sich von denen der aktuellen Ansätze und sind zudem teilweise auch zueinander konträr. So soll es beispielsweise auf der einen Seite möglich sein, die technischen Eigenschaften der Sensorik zu beschreiben (AM4.2). Hier werden eine Reihe technischer Informationen zur Qualität und zur Verwendung der Sensorik benötigt. Auf der anderen Seite muss dem Bewohner eine Sicht auf die für ihn relevanten Teile des Kontextmodells bereitgestellt werden, die von der Komplexität und technischen Details abstrahiert (AM7.2). Die Anwendungsfälle, in denen der Bewohner eine geeignete Sicht auf das Kontextmodell benötigt, sind in Kap. 3.4 beschrieben. Beide Sichten auf das Kontextmodell werden beispielsweise benötigt, um dem Bewohner die Konfiguration eines kontextadaptiven Dienstes zu ermöglichen, welches durch die verfügbare Sensorik unterstützt wird. Dieser Anwendungsfall wird auch als „End-User Programming“ [125] bezeichnet.

Diese Notwendigkeit ist auch in den unterschiedlichen Phasen der Entwicklung und Nutzung der kontextadaptiven AAL-Dienste zu sehen. Die Sicht auf die benötigten Kontextinformationen und die zugrundeliegenden Kontextmodelle ist in der Phase der Planung und Definition eine andere als für deren Entwicklung oder Betrieb. Eine Analogie findet man hier beispielsweise in den Phasen des Entwurfsprozesses der Datenmodellierung ([127], S. 48).

Die der Dissertation zugrundeliegende Lösung besteht darin, die unterschiedlichen Anforderungen inhaltlich zu bündeln und verschiedene Sichten auf die Kontextmodellierung zu definieren. Jede dieser Sichten ist spezialisiert auf ein Teilproblem und realisiert eine spezielle Ausprägung des Kontextmodells mit dem entsprechenden inhaltlichen Fokus, Abstraktionsgrad und Repräsentationsform. Die in den Sichten definierten Modellelemente werden aufeinander abgestimmt, um Übergänge oder Verknüpfungen zwischen diesen zu ermöglichen. Auf diese Weise entsteht ein integriertes Kontextmodell für das AAL, welches die unterschiedlichen Anforderungen unterstützt und diese zueinander in Beziehung bringt.

Die Sichten entstehen nachfolgend durch die Definition von Modellierungsdimensionen, denen Anforderungen zugeordnet werden, die unabhängig vonei-

nander zu betrachten sind. Dabei können zwei Dimensionen identifiziert werden. In einer Dimension werden die Anforderungen gebündelt, die auf den Einsatzzweck des Kontextmodells ausgerichtet sind. Ein Beispiel dafür ist die Einbindung der Sensorik (AM4.1) oder die Konfiguration eines Dienstes durch den Bewohner (AM7.2.3). In der zweiten Dimension werden die Anforderungen gebündelt, die sich aus den Phasen der Entwicklung und Nutzung des Kontextmodells ergeben (AM8.2, AM9.2). Innerhalb dieser beiden Dimensionen werden zusammengehörige Anforderungen nochmals gebündelt und ergeben die möglichen Ebenen. Diese Dimensionen werden in den folgenden Unterkapiteln im Detail beschrieben.

In der so aufgespannten Matrix ist in den einzelnen Zellen die Ausrichtung des jeweiligen Kontextmodells bezogen auf seinen Einsatzzweck sowie der Phase seiner Entwicklung und Nutzung definiert. Diese bilden die entsprechenden Sichten auf das Gesamtmodell. Hierfür werden in den folgenden Kapiteln (Kap. 5, 6 und 7) entsprechende inhaltliche Zielsetzungen, der Abstraktionsgrad und die Repräsentationsform definiert. Innerhalb dieser Sichten können die aktuellen Ansätze zur Kontextmodellierung erneut betrachtet und zur Lösungsfindung herangezogen werden.

Die Basis für die Definition der Übergänge und Verknüpfungen bildet ein Metamodell, welches die gemeinsamen Modellelemente und deren Semantik definiert (AM1, AM1.2, AM1.3, AM2.1, AM2.2, AM2.3, AM2.4, AM2.5, AM3.1, AM3.2, AM3.3, AM3.4, AM6.2, AM6.3). Zudem erlaubt die Definition eines Metamodells die Erweiterung der konkreten Kontextmodelle mit Hilfe der definierten Modellelemente (AM4.1, AM7.1). Dieser zweidimensionale Modellierungsansatz wird in der nachfolgenden Abbildung visualisiert.

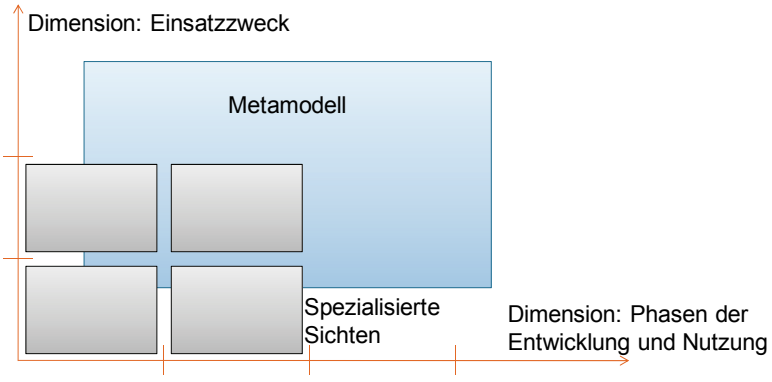


Abbildung 18: Zweidimensionale Kontextmodellierung

Nachfolgend werden die einzelnen Dimensionen im Detail beschrieben.

4.2.1 Dimension „Einsatzzweck“

Die Dimension „Einsatzzweck“ umfasst die Anforderungen, die den konkreten Einsatzzweck des Kontextmodells beschreiben:

- Nutzung des Kontextmodells zur Auswahl von Sensorik (AM4.2).
- Trennung zwischen Diensten und Kontextinfrastruktur (AM8.1, AM9.1).
- Unterscheidung zwischen der strukturellen und der funktionalen Kontextadaptivität eines AAL-Dienstes (AM5.1, AM5.2).
- Unterstützung dienstetyp-spezifischer Kontextmodelle (AM6.1).
- Endnutzergerechte Darstellung, Korrektur und Konfiguration von Kontextinformationen (AM7.2.1, AM7.2.2, AM7.2.3).

Für das AAL wird nachfolgend in der Dimension „Einsatzzweck“ eine Aufteilung der unterschiedlichen Anforderungen in drei Ebenen vorgestellt [8]. Aufbauend auf einer sehr technischen Sicht von Kontextinformationen auf der Ebene der Infrastruktur, abstrahieren die darauf aufsetzenden Ebenen immer mehr von der technologischen weg zu einer anwendungsorientierten Sicht hin.

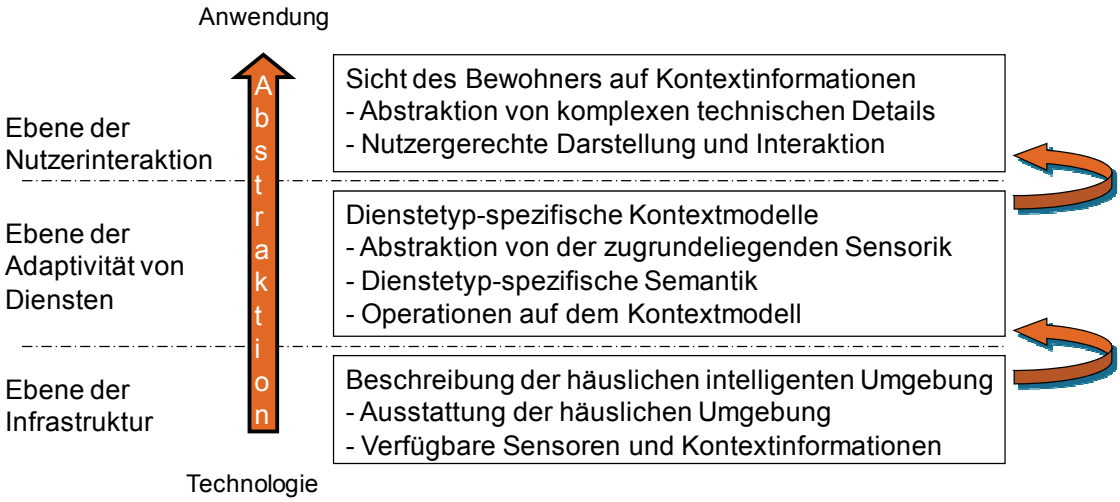


Abbildung 19: 3 Ebenen der Kontextmodellierung im AAL

Auf der Ebene der Infrastruktur werden die Anforderungen zusammengefasst, die generell mit der Bereitstellung und Erkennung von Kontextinformationen zu tun haben. Sie ist unabhängig von der Nutzung der Kontextinformation in den AAL-Diensten. Der Fokus des Kontextmodells auf dieser Ebene ist die Beschreibung verfügbarer Kontextinformationen und derer Eigenschaften und

wird benötigt für die Beschreibung von Kontextsensoren sowie die konkreten Entitäten, auf die sich die Informationen beziehen. Auf dieser Basis kann eine dynamische Integration und Auswahl der Sensorik in und aus einer Kontextinfrastruktur erfolgen. Zudem geschieht hier eine generelle Beschreibung der häuslichen Umgebung unabhängig von der Nutzung dieser Kontextinformationen für einen konkreten AAL-Dienst, die für die Unterstützung der strukturellen Kontextadaptivität benötigt wird.

Auf der zweiten Ebene werden die Anforderungen zusammengefasst, die sich auf die Adaptivität der AAL-Dienste beziehen. Das Kontextmodell auf ihr abstrahiert von den technischen Details der Kontexterkennung in der Infrastruktur. Sie dient der Bereitstellung von dienstetyp-spezifischen Kontextinformationen. Dazu gehört auch die Anreicherung der durch die Infrastruktur gelieferten Kontextinformationen um zusätzliche Semantik. Ein Beispiel hierfür ist die Unterscheidung zwischen dem Bewohner und seinem Betreuer für einen AAL-Dienst im Dienstetyp „Gesundheit“. Auf der Ebene der Infrastruktur wird dagegen eine solche Unterscheidung zur Einordnung von Kontextinformationen und Beschreibung der Sensorik nicht benötigt. Das Kontextmodell der zweiten Ebene konzentriert sich dabei auf die Konzepte und Methoden, die aus operativer Sicht von den Diensten benötigt werden. Es erfolgt so die Bereitstellung von Kontextinformationen für die Auswahl und den Betrieb der AAL-Dienste, die für die funktionale Kontextadaptivität der Dienste benötigt wird.

Auf der dritten Ebene der Nutzerinteraktion werden die Anforderungen zusammengefasst, die sich auf die Kommunikation mit dem Bewohner beziehen. Das Kontextmodell auf dieser Ebene abstrahiert von der Komplexität des operativen Kontextmodells auf Ebene der Dienste. Sie fokussiert sich auf die Bereitstellung und Festlegung von Kontextinformationen für und durch den Endanwender unter dem Aspekt der Ergonomie. Dazu gehört beispielsweise die Konfiguration des kontextadaptiven Verhaltens der AAL-Dienste durch den Bewohner.

Zwischen diesen Ebenen existieren Übergänge, die zu modellieren sind. Die Verfügbarkeit von Kontextinformationen auf der Ebene der Infrastruktur hat Auswirkungen auf die Qualität der Kontextmodelle in der Ebene der Dienste. Werden Kontextinformationen in einer geforderten Qualität von einem AAL-Dienst benötigt, so muss ein Abgleich mit den Eigenschaften der Sensorik in der Infrastruktur erfolgen können. Ein weiterer Übergang ist zwischen den Kontextmodellen auf der Ebene der Nutzerinteraktion und der Dienste zu finden. Der Bewohner soll in der Lage sein, das kontextadaptive Verhalten eines AAL-Dienstes zu verstehen und zu steuern. Dazu müssen die relevanten Informationen aus dem Kontextmodell der Ebene der Dienste in ein vereinfachtes Kontextmodell der Ebene der Nutzerinteraktion überführt werden. Vom Nutzer getätigte Eingaben zum gewünschten kontextadaptiven Verhalten müssen wieder in eine operative Form überführt werden. So existieren Bezie-

hungen zwischen den drei Ebenen, die modelliert werden müssen. Dieses erfolgt auf der Basis des gemeinsamen Metamodells.

4.2.2 Dimension „Phasen der Entwicklung und Nutzung“

Die Dimension „Phasen der Entwicklung und Nutzung“ umfasst die Anforderunge, die die dynamische Sicht auf die Kontextmodellierung in den Phasen der Entwicklung und der Nutzung von kontextadaptiven AAL-Diensten beschreiben:

- Unterstützung unterschiedlicher Entwicklungs- und Nutzungsphasen der Kontextinfrastruktur und der AAL-Dienste. Bereitstellung geeigneter Abstraktionsebenen und Repräsentationsformen. Unterstützung der Überführung zwischen diesen Phasen (AM8.2, AM9.2).

In Kap. 3.5 wurden bereits die unterschiedlichen Phasen identifiziert, in denen die Kontextmodellierung eine Rolle spielt. In den entsprechenden Phasen dienen Kontextmodelle einem jeweils unterschiedlichen Zweck. Aus dieser Betrachtungsweise lassen sich vier unterschiedliche Modelltypen identifizieren: das informale, konzeptuelle, operative und deskriptive Kontextmodell. Eine solche Unterscheidung zwischen diesen Modelltypen lässt sich auch analog in anderen Domänen der Modellierung beispielsweise in der Datenmodellierung wiederfinden. Die vier identifizierten Modelltypen werden nachfolgend entlang der unterschiedlichen Phasen detailliert beschrieben.

In der Phase der „Planung“ und der „Definition“ werden die benötigten Produkteigenschaften erhoben und dokumentiert. Sie erfolgt beispielsweise in Form eines Lasten- und eines Pflichtenheftes. Zielstellung hierbei ist die Schaffung einer abgestimmten Dokumentation geforderter Produkteigenschaften. Ihre meist in Textform strukturierte Beschreibung kann vom Auftraggeber verstanden und nachvollzogen werden. Gleiches muss hierbei auch für den Aspekt der Kontextadaptivität gelten, weil auch hier die geforderte Ausprägung der Kontextinfrastruktur und das kontextadaptive Verhalten eines AAL-Dienstes so zu beschrieben ist, dass sie vom Auftraggeber verstanden und nachvollzogen werden kann. Analog zu den sonstigen Produkteigenschaften wird dafür ebenfalls eine textuelle und informelle Form der Kontextbeschreibung benötigt. Sie ist aber aufgrund der fehlenden formalen Festlegungen kein Modell im eigentlichen Sinne.

In der Designphase werden die geforderten Produkteigenschaften konzeptuell auf der Basis formaler Modelle umgesetzt. Für verschiedene Aspekte existieren spezialisierte formale Modelle, z.B. das Entity Relationship Diagramm für die Datenmodellierung oder UML für die Festlegung von Klassen und deren Verhalten. Diese Modelle haben eine spezifische Ausrichtung auf einen Aspekt und definieren eindeutig die darin relevanten Konzepte. Zumeist besitzen sie

auch eine grafische Repräsentation der einzelnen Modellelemente. Auf diese Weise können wesentliche Konzepte für die Umsetzung der geforderten Produkteigenschaften übersichtlich und inhaltlich eindeutig dargestellt werden. Gleiches muss auch für die Konzeption der kontextadaptiven Eigenschaften der Kontextinfrastruktur und der AAL-Dienste gelten, für die deshalb ein konzeptuelles Kontextmodell mit einer grafischen Repräsentation benötigt wird.

In der Implementierungsphase muss eine Umsetzung der Anforderungen und Konzepte erfolgen. Diese geschieht auf der Basis ausführbarer Modelle, für die verschiedene Möglichkeiten verfügbar sind. Zur Umsetzung der Anwendungslogik lassen sich Programmiersprachen beispielsweise auf dem objektorientierten Paradigma oder spezialisierte domänenspezifische Sprachen einsetzen. Für die Umsetzung der Verwaltung und Bereitstellung von Daten stehen relationale Datenbankmanagementsysteme zur Verfügung und für den Einsatz von Nutzeroberflächen lassen sich Oberflächenbeschreibungssprachen wie HTML gebrauchen. Ihnen allen liegt ein Modell zugrunde, in welchem die konkrete Ausprägung für eine Anwendung beschrieben und durch eine Entwicklungs- oder eine Laufzeitumgebung verwendet werden kann. Gleichermaßen wird auch für die Umsetzung der kontextadaptiven Eigenschaften ein operatives Kontextmodell benötigt. Dieses muss ebenfalls ausführbar sein und durch eine geeignete Laufzeitumgebung, beispielsweise einem Kontext Managementsystem unterstützt werden. Es benötigt alle technischen Details, die für die Ausführung notwendig sind.

In der Testphase wird die korrekte Umsetzung des implementierten Produkts getestet. Dieses erfolgt gegen die in der Designphase festgelegten Anforderungen. Hierfür werden Testpläne erstellt und spezifiziert. Bezugnehmend auf die in den Pflichtenheften definierten Produkteigenschaften werden hierbei unterschiedliche Testfälle identifiziert und zumeist formal strukturiert schriftlich festgehalten. Die Umsetzung der Testfälle erfolgt entlang dem implementierten Produkt in der jeweiligen Programmier- oder Beschreibungssprache. Gleiches muss auch beim Testen des kontextadaptiven Verhaltens geschehen. Die dabei umzusetzenden Testfälle werden informell beschrieben. Für die Umsetzung der Testfälle wird also ein operatives Kontextmodell mitsamt der zugrundeliegenden Laufzeitumgebung benötigt.

In der Phase der Bereitstellung sollen Sensoren oder AAL-Dienste zur Übernahme in die häusliche Umgebung auf dem Dienstemarkt angeboten werden. Zur Registrierung und zum Auffinden dieser Artefakte wird eine Beschreibung ihrer Eigenschaften benötigt. Im Umfeld der serviceorientierten Architekturen existieren dazu beispielsweise mit UDDI [135] eine Registrierungskomponente und eine zugehörige Dienstebeschreibungssprache, mit der verschiedenste Aspekte eines Dienstes notiert werden können. Die Funktionalität wird beispielsweise durch die Zuordnung zu einer Dienstetaxonomie beschrieben und fungiert als Grundlage für die manuelle oder automatische Auswahl einer geeigneten Diensteinstanz. Gleiches muss wiederum für die kontextadaptiven

Eigenschaften eines AAL-Dienstes gelten. Sie sind in einer Form zu beschreiben, die einem potentiellen Anwender das kontextadaptive Verhalten des AAL-Dienstes erkennen lassen und ihn in die Lage versetzen, sich den gewünschten AAL-Dienst sowie dessen kontextadaptives Verhalten bzw. die gewünschte Sensorik vorzustellen. Hierfür wird ein deskriptives Kontextmodell benötigt.

In der Phase der Übernahme sind die ausgewählten Sensoren oder AAL-Dienste in die intelligente häusliche Umgebung des Bewohners aufzunehmen. Hierfür werden das operative Kontextmodell und die zugehörige Laufzeitumgebung benötigt, um das kontextadaptive Verhalten der AAL-Dienste zu initialisieren oder die Sensoren einzubinden. Für die Zuordnung in das operative Kontextmodell benötigt man auch die zu den Sensoren oder AAL-Diensten gehörenden Beschreibungen, womit das deskriptive Kontextmodell in das operative Kontextmodell überführt werden kann.

In der Phase des Betriebs werden Kontextinformationen von den Sensoren bereitgestellt und das kontextadaptive Verhalten der AAL-Dienste innerhalb der intelligenten häuslichen Umgebung des Bewohners umgesetzt. Hierfür wird das operative Kontextmodell mitsamt der zugehörigen Laufzeitumgebung benötigt.

Die Zuordnung der Phasen zu den einzelnen Umgebungen und den identifizierten Modelltypen wird in nachfolgender Abbildung dargestellt. Entsprechend dem gewählten Vorgehensmodell der Softwareentwicklung sowie der Umsetzung der Bereitstellung, Übernahme und des Betriebs finden Überführungen zwischen den einzelnen Modelltypen statt.

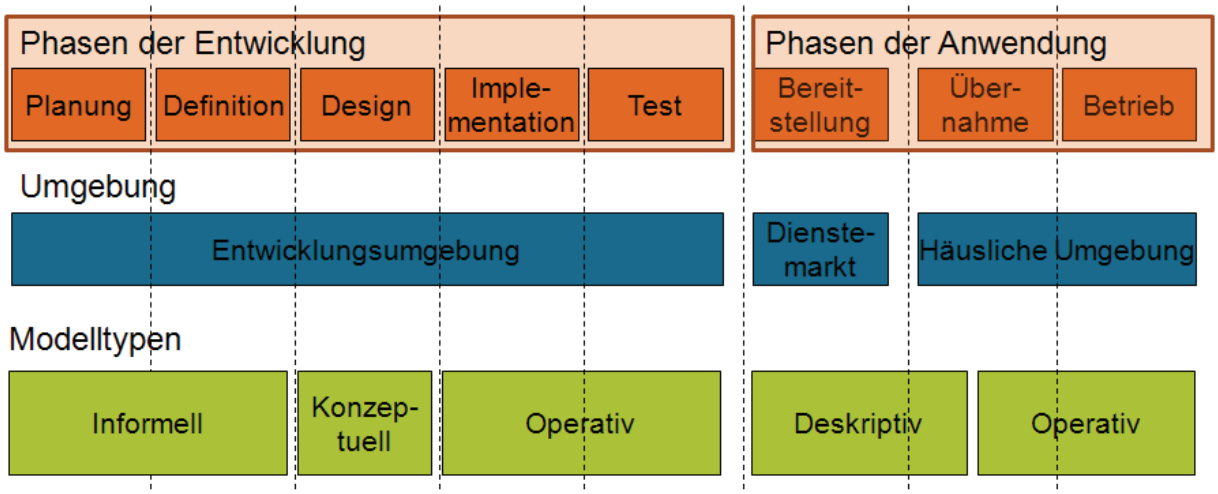


Abbildung 20: Modelltypen der Kontextmodellierung im AAL

4.3 Metamodell als Grundlage der Kontextmodellierung

Die Definition eines Metamodells ist ein wesentlicher Bestandteil der Lösung für die Kontextmodellierung im AAL. Folgende Anforderungen bilden die Grundlage hierfür:

- Bereitstellung von Modellelementen zur Erstellung konkreter Kontextmodelle, z.B. Kontextentitäten, Kontextrelationen, Kontextattribute, zulässige Vergleiche zwischen Kontextattribute, Situationen und Repräsentationsformen (AM1, AM1.2, AM1.3, AM2.1, AM2.2, AM2.3, AM2.4, AM2.5, AM3.1, AM3.2, AM3.3, AM3.4, AM6.2, AM6.3)

- Erweiterbarkeit des konkreten Kontextmodells, beispielsweise zur Erfassung neuartiger Sensoren (AM4.1) oder der Erweiterung der Menge von AAL-Diensten in der häuslichen Umgebung (AM7.1)

- Die mehrdimensionale Kontextmodellierung erfordert ein Metamodell als Basis für die Definition von Übergängen und Verknüpfungen zwischen den einzelnen Sichten des integrierten Kontextmodells. In jeder Ebene des Kontextmodells sind die darin benötigten Konzepte und Modellelemente zu definieren. In den unterschiedlichen Modelltypen werden diese Elemente in jeweils spezifischen Abstraktions- und Formalisierungsstufen konkretisiert. Die Übergänge zwischen den Phasen erfolgen auf der Basis der gemeinsamen Modellelemente, wobei sich diese auf den einzelnen Ebenen aufgrund ihrer unterschiedlichen inhaltlichen Ausrichtung im Detail unterscheiden können. Jedoch sind auch hier gemeinsame Modellelemente zu identifizieren, die eine Grundlage für die Übergänge zwischen den einzelnen Ebenen bilden.

Nachfolgend werden verschiedene Ansätze für die Metamodellierung beschrieben und bewertet. Daraus resultierend wird der Ansatz erarbeitet, der für die Definition des Metamodells im Rahmen der Kontextmodellierung im AAL verwendet wird. Zu den betrachteten Ansätzen gehören die Extensible Markup Language (XML) als eine einfache textbasierte Beschreibungssprache, sowie die Verwendung der Unified Modeling Language (UML). Eine weitere Alternative ist die Verwendung einer Ontologiesprache, z.B. OWL. Abschließend werden die Konzepte der Model Driven Architecture (MDA) betrachtet.

4.3.1 XML

Die Extensibile Markup Language ist eine textuelle Auszeichnungssprache zur Darstellung hierarchisch strukturierter Daten. In der XML-Spezifikation [136] des World Wide Web Konsortiums (W3C) wird festgelegt wie XML-Dokumente aufgebaut sein müssen. Die Spezifikation gibt dabei den physischen und den logischen Aufbau eines XML-Dokuments vor. Dies erfolgt durch die Definition

von Sprachelementen auf deren Basis konkrete XML-Dokumente erstellt werden können. Beispiele für diese Sprachelemente sind „Document“, „Character“, „Start-Tag“ und „End-Tag“. Die Namen der Strukturelemente, z.B. die Namen der Tags und der Attribute, lassen sich frei wählen. Dadurch wird auch nicht die Semantik dieser Elemente durch die XML-Spezifikation vorgegeben. Entspricht ein Dokument den Vorgaben der XML-Spezifikation, so ist dieses „wohlgeformt“.

Mit Hilfe einer Schemasprache können anwendungsspezifische Vorgaben an die Erstellung von XML-Dokumenten gemacht werden. Das W3C empfiehlt dafür XML-Schema (XSD) [137]. Es können Vorgaben an die Struktur von XML-Dokumenten gemacht werden, aber auch den Inhalt von Elementen und Attributen beispielsweise mittels regulärer Ausdrücke zu beschränken. Mit der Definition der Struktur der XML-Dokumente werden die Benennungen der Strukturelemente vorgegeben und somit auch deren Semantik. Diese Vorgaben werden selbst wieder in einem XML-Dokument beschrieben. Ein XML-Dokument, welches „wohlgeformt“ ist und den Vorgaben einer XML-Schemadefinition entspricht, wird als „gültig“ bezeichnet. Es gibt eine Vielzahl von spezialisierten XML-Sprachen. Die bekannteste davon ist die Hypertext Markup Language (HTML). Weitere Beispiele sind SVG, GML, SMIL, SAML, SyncML oder MathML. Zudem basieren Standards für Webservices vielfach auf XML, z.B. SOAP, WSDL oder WS-*. Auch semantische Sprachen basieren vielfach auf der textuellen Repräsentation durch XML, z.B. RDF oder OWL. Beispiele für XML-basierende Kontextmodelle wurden in Kap. 2.2.4 aufgeführt.

Das Metamodell als Basis zur Erstellung konkreter Kontextmodelle im AAL kann mit Hilfe einer Schemasprache für XML-Dokumente umgesetzt werden. So kann ein XML-Schema definiert werden, welches die Strukturelemente für die Repräsentation der benötigten Modellelemente „Kontextentität“, „Kontextrelation“, „Kontextattribut“, „Zulässige Vergleiche zwischen Attributen“, „Situation“ und „Repräsentationform“ vorgibt. Für diese Modellelemente können entsprechende Attribute vorgegeben werden, z.B. für deren Benennung oder der Beschreibung weiterer Eigenschaften (AM4.2.1 - AM4.2.9, AM6.2.3.1). XML-Schema ermöglicht die Ableitung von neuen Elementtypen über „Extensions“ bzw. die Definition weiterer eingeschränkter Elementtypen über „Restrictions“. Dadurch können Generalisierungs- und Spezialisierungsbeziehungen zwischen Modellelementen des Kontextmodells ausgedrückt werden (AM2.2, AM3.2, AM6.2.1.1). Mit Hilfe eines solchen XML-Schemas können dann konkrete Kontextmodelle in Form von XML-Dokumenten erstellt werden.

XML ist für den plattform- und implementationsunabhängigen Austausch von Daten zwischen Anwendungen geeignet. Ein XML-Dokument kann durch einen validierenden XML-Parser anhand der Schemadefinition auf seine Gültigkeit hin überprüft werden. Die Überführung eines XML-Dokuments in eine alternative Schemadefinition kann mittels einer Transformationssprache wie XSL Transformation (XSLT) [138] erfolgen. Das XML-Dokument kann anschließend

von einer Anwendung mittels des XML-Parsers ausgelesen und dort dargestellt oder in seine interne Repräsentation überführt werden.

Durch die textuelle Darstellung sowohl der Strukturen als auch der konkreten Inhalte können XML-Dokumente unmittelbar durch den Menschen gelesen werden. Allerdings ist eine rein textuelle Darstellung eines Kontextmodells für den Entwickler oder den Bewohner insbesondere bei großen Modellen schwer zu überblicken und zu lesen. Insbesondere für den technischen Laien ist eine solche Darstellung in Bezug auf die Wahrnehmung und das Verständnis nicht geeignet (AM7.2).

4.3.2 UML

Die Unified Modeling Language (UML) [98] ist eine standardisierte Sprache für die Modellierung von Software und anderen Systemen. Einsatzgebiet von UML ist der gesamte Softwareentwicklungsprozess von der Anforderungsanalyse bis hin zur Generierung von Softwareartefakten und der abschließenden Dokumentation. Somit entspricht UML der generellen Anforderung an die Kontextmodellierung bezüglich der Unterstützung der unterschiedlichen Phasen der Entwicklung der Kontextinfrastruktur (AM8.2), sowie der AAL-Dienste (AM9.2). Unterschiedliche Aspekte eines Softwaresystems können mit Hilfe von UML ausgedrückt werden, beispielsweise der statische Aufbau in Form von Komponenten und Klassen, das dynamische Verhalten des Systems in Form von Interaktionen zwischen den Komponenten oder die Verteilung auf die zugrundeliegende Hardware. Aufgrund der Vielzahl und der Komplexität dieser universellen Sprache wurden verschiedene Konzepte bei der Definition von UML angewendet, die diese beherrschbar machen. Nach ([139], S.10) können folgende der UML Spezifikation zugrundeliegende Paradigmen identifiziert werden:

- Hierarchischer Aufbau der Sprache: Die UML Sprachdefinition besteht aus einer Reihe aufeinander aufbauender oder einander ergänzender Sprachdefinitionspakete. Um die Sprache für einen gewissen Zweck einzusetzen, sind dann Kenntnisse über den Sprachkern und die für den Zweck geeigneten Sprachpakete notwendig.

- Objektorientierung als Grundlage der gesamten Sprache: UML baut auf einer objektorientierten Begriffswelt auf und unterstützt direkt die dazugehörigen Grundmechanismen der Modellierung. Dazu gehören die Klassifikation bzw. Exemplifikation, die Generalisierung bzw. Spezialisierung, die Strukturierung, die Gruppierung von Konzepten zu Modulen, die Abkapselung und das Information Hiding, sowie die deskriptive oder preskriptive Beschreibung des Verhaltens.

- Trennung von Begriffswelt und Notation: Im Mittelpunkt des Sprachstandards steht die Definition der Modellierungskonzepte, d.h. der Menge an Begriffen, die benutzt werden können, um die in einem Modell enthaltenen Informationen zu reflektieren, sowie die Menge der Regeln zur Konstruktion eines Modells auf der Grundlage dieser Begriffswelt. Die Notation wird dagegen der Begriffswelt zugeordnet und dient allein der physischen Repräsentation eines Modells in Form von Diagrammen. Hier erfolgt eine strikte Trennung zwischen Inhalt (Konzepte) und Form (Notation) eines Modells. Dieses ermöglicht die Nutzung einer dem jeweiligen Zweck angepassten Notation und die Erweiterung bzw. Einschränkung der Sprache für konkrete Anwendungsgebiete.

- Metamodellierung als Basisprinzip der Sprachdefinition: Wie bereits in Kap. 2.2.5 beschrieben baut UML auf einer 4-schichtigen Architektur auf, deren oberste Schicht ein Meta-Metamodell ist. Dieses ermöglicht die Erweiterung des Metamodells und damit auch der Modellierungssprache.

- Unterscheidung zwischen Modell und Diagramm: Ein UML-Modell ist eine organisierte Sammlung von Konzepten aus der UML-Begriffswelt mit dem Ziel einer Abstraktion eines existierenden oder zukünftigen Systems. Repräsentiert wird das Modell oder auch nur ein Teil des Modells durch eine Gruppe von Diagrammen. Ein Diagramm ist dabei stets ein Teil einer Sicht auf das Modell. Es muss niemals alle in dem Modell enthaltene Elemente beinhalten. Ebenso kann ein Modellelement mehrfach in einem oder mehreren Diagrammen erscheinen. Beispiele für diese Diagramme sind das Klassendiagramm oder das Sequenzdiagramm.

Die 4-schichtige Architektur von UML ist konform zur Metadaten-Architektur „Meta Object Facility“ (MOF) [97]. In der hier definierten Schicht M3, dem Meta-Metamodell, werden die Konzepte festgelegt, mit denen das Metamodell von UML beschrieben werden kann. Diese fassen die Grundkonzepte der Objektorientierung zusammen. Dieses wird auch als „UML Infrastructure“ bezeichnet. Die „UML Infrastructure“ [140] definiert ein Paket mit dem Namen „Core“. Dieses enthält die Unterpakete „PrimitiveTypes“ zur Definition der primitiven Datentypen, sowie „Basic“ zur Erklärung der Grundbegriffe „Klasse“, „Attribut“, „Operation“ und „Paket“. Weiterhin enthält es das Unterpaket „Abstractions“ für weiterführende Konzepte wie die Generalisierung, Instanz, Ausdruck, Einschränkung, Vielfachheit, Namensraum, Re-Definition, Sichtbarkeit und den allgemeinen Begriff der Relation. Letztendlich erweitert das Unterpaket „Constructs“ die Begriffswelt um das Grundkonzept „Assoziation“. Auf dieser Ebene sind bereits wesentliche Konzepte festgelegt, die für die Definition der Elemente eines Kontextmodells benötigt werden. Das objektorientierte Paradigma erlaubt die Definition von Klassen, die die Modellelemente „Kontextentität“, „Kontextrelation“, „Kontextattribut“, „Zulässige Vergleiche zwischen

Attributen“, „Situation“ und „Repräsentationform“ repräsentieren. Die Eigenschaften dieser Modellelemente können mit Hilfe der Attribute beschrieben werden. Die Generalisierungs- und Spezialisierungsbeziehung kann mit Hilfe der Grundkonzepte auf dieser Ebene ausgedrückt werden. Weiterhin können Beziehungen zwischen den Modellelementen mit Hilfe der Assoziationen beschrieben werden.

Auf Ebene M2 erfolgt die Definition der UML-Sprachelemente mittels Instantiierung der Modellelemente auf Ebene des Meta-Metamodells M3. Hier werden die Modellelemente definiert, die für die Beschreibung von Software und anderen Systemen benötigt werden. Dieser Teil von UML wird als „UML Superstructure“ [141] bezeichnet. Alle Elemente des UML-Metamodells sind Instanzen der Elemente der „UML Infrastructure“, also des Meta-Metamodells. Das UML-Metamodell importiert alle Elemente der UML Infrastructure. Dieses geschieht durch das Paket „UML::Classes::Kernel“, das das Paket „Constructs“ aus der „UML Infrastructure“ wiederverwendet. „Kernel“ erklärt durch „PaketMerge“ die Begriffe aus „Constructs“ zu Begriffen der Sprache UML. Somit sind alle bekannten Konzepte aus dem Meta-Metamodell auch auf Ebene M2 zu finden und können hier ebenfalls für die Definition des Metamodells für die Kontextmodellierung verwendet werden. Diese werden um weitere Aspekte erweitert bzw. bei Bedarf eingeschränkt. Zudem werden zusätzliche Konzepte für die Strukturbeschreibung und Klassifikation eingeführt. Dazu gehören beispielsweise die Definition von „Assoziationsklassen“, „Komponenten“ oder „Interne Strukturen“. Weiterhin werden Begriffe für die Installations- und Konfigurationsbeschreibung, sowie die Beschreibung der dynamischen Aspekte hinzugefügt. Ein Beispiel hierfür ist die Definition von Modellelementen zur Beschreibung eines Zustandsautomaten.

Auf Ebene M1 erfolgt die Definition von konkreten Modellen zur Beschreibung von Software und sonstigen Systemen durch die Instanziierung der Elemente des UML Metamodells. Hier ist beispielsweise der konkrete Aufbau einer Anwendung durch Beschreibung der in ihr enthaltenen Klassen und deren Beziehungen in Form eines Klassendiagramms beschrieben. Die Nutzung der Anwendung kann in Form von Anwendungsfalldiagrammen beschrieben werden. Das dynamische Verhalten der Anwendung kann durch Sequenzdiagramme beschrieben werden. Die Ebene M0 beschreibt Instanzen der auf Ebene M1 definierten Modellelemente, z.B. die Instanziierung einer Klasse „User“.

Die Konformität der Architektur von UML zur Metadaten-Architektur MOF hat den Vorteil, dass die in der Norm MOF 1.x und ihrer Nachfolgeversionen definierten Technologieabbildungen ebenfalls verwendet werden können. Die bekannteste dieser Abbildungen ist „XML Metadata Interchange“ (XMI) [142]. Diese erklärt, wie Instanzen der Elemente des MOF-Modells auf XML-Schemadefinitionen abgebildet werden. Wendet man die Abbildungsregeln für Elemente eines Metamodells auf das UML-Metamodell an, so erhält man eine Schemadefinition, die ein Format zum Austausch von UML-Modellen erklärt.

Die Technologieabbildung von MOF auf XML bildet damit die Grundlage, um UML-Modelle in standardisierter Form auszutauschen. Diese Abbildung zwischen der Metamodellierung und der XML-Technologie ist in der folgenden Abbildung dargestellt ([139], S. 80).

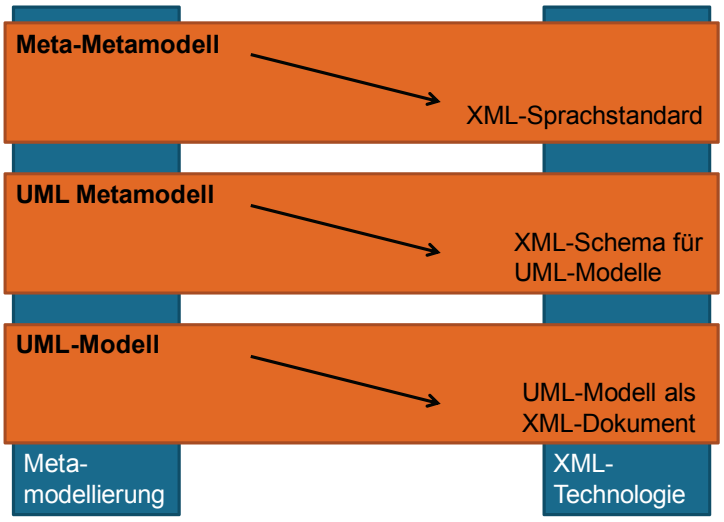


Abbildung 21: Metamodellierung und XML Technologie

Die Definition des Metamodells zur Erstellung konkreter Kontextmodelle kann mit Hilfe von UML erfolgen. Die Erweiterbarkeit des konkreten Kontextmodells ist aufgrund der Metadaten-Architektur gegeben. Die Instantiierung des Kontextmodells, z.B. die Beschreibung eines Bewohners „Anna Mustermann“, erfolgt dann auf Ebene M0 bzw. darunter. Die Frage nach der geeigneten Ebene wird im gewählten Ansatz (Kap. 4.3.6) beantwortet.

Die Notwendigkeit der verschiedenen Sichten mit jeweils spezifischen Abstraktionsstufen für die Kontextmodellierung wird durch die Unterscheidung zwischen Diagramm und Modell unterstützt. So kann ein spezialisiertes Diagramm für die Entwicklungsphase „Design“ definiert werden, welches nur den jeweils benötigten Ausschnitt aus dem Modell umfasst.

Die Verwendung der „XML Metadata Interchange“ ermöglicht die Abbildung des Kontext-Metamodells in ein entsprechendes XML-Schema. Aus einem konkreten Kontextmodell kann ein XML-Dokument generiert werden, welches für die plattform- und implementationsunabhängige Beschreibung und beispielsweise die Instantiierung eines Kontextservers verwendet werden kann.

Durch die grafische Notation, die der Begriffswelt des Metamodells zugeordnet wird, können Kontextmodelle in visueller Form dargestellt werden. Für die im UML-Metamodell definierten Konzepte existieren bereits vorgegebene grafische Repräsentationen. UML ermöglicht jedoch auch die Verwendung einer

eigenen Notation, falls diese für den jeweiligen Anwendungszweck geeigneter ist. Eine solche Notation wird im Entwicklungsprozess von Software durch entsprechende Entwicklungswerkzeuge angeboten und durch den geschulten Entwickler verstanden. Für den technischen Laien ist die Verwendung von UML und einer entsprechenden Notation beispielsweise für die Definition des kontextadaptiven Verhaltens eines AAL-Dienstes (AM7.2.3) dennoch nicht geeignet, da die zugrundeliegenden Konzepte und Begriffe verstanden werden müssen. Zu diesem Zweck erfolgt eine Vereinfachung der Konzepte des Kontext-Metamodells auf Ebene der Nutzerinteraktion (Kap. 4.2.1).

4.3.3 Model Driven Architecture

Die „Model Driven Architecture“ (MDA) baut auf dem MOF-Architekturansatz auf und hat die modellgetriebene Softwareentwicklung als Ziel. Eingesetzte Basistechnologien sind hier die Nutzung von UML zur Beschreibung der Modelle, sowie von XMI für deren Austausch. Ein Grundsatz hierbei ist es die technologieunabhängigen und technologiebezogenen Modellierungskonzepte sowohl von der eingesetzten Notation als auch von im Entwicklungsprozess eingesetzten Codegeneratoren und Modelltransformatoren zu trennen. Die notationsunabhängige Darstellung des Modells ermöglicht beispielsweise dir EInsatz verschiedener Werkzeuge. Die Trennung vom Einsatz verschidener nachgeordneter Komponenten ermöglicht beispielsweise die Verwendung der Modellelemente desselben Modells sowohl als Eingabe für die Modelltransformatoren als auch für einen Dokumentationsgenerator.

Die Grundlage der modellgetriebenen Softwareentwicklung ist die Transformation eines Modells höherer Abstraktionsstufe in ein oder mehrere Modelle niedrigerer Abstraktionsstufe. Mittels Regeln der Modelltransformation können Elemente der Quell-Metamodells in Instanzen des Ziel-Metamodells überführt und mit benötigten zusätzlichen Details angereichert werden. Auf diese Weise soll eine Trennung der Spezifikation der Funktionalität und der Benutzung eines Systems von den Details seiner Realisierung auf der Grundlage einer konkreten Hardware- und Softwareplattform ermöglicht werden. Folgende Aktivitäten sind daher Bestandteil der modellgetriebenen Softwareentwicklung der MDA:

- Spezifikation eines Systems unabhängig von der zur Realisierung einzusetzenden Plattform.
- Spezifikation der einsetzbaren Plattformen.
- Auswahl einer Plattform für die Realisierung des Systems.
- Transformation der Systemspezifikation in eine plattformspezifische Spezifikation.

Mit UML werden in einem MDA-konformen Vorgehen während der Softwareentwicklung unterschiedliche Modelle für verschiedene Viewpoints erstellt:

- Computation Independent Model (CIM): Das System wird in seiner Einsatzumgebung modelliert. Es beschreibt die Nutzung des Systems, die Anforderungen an seine Umgebung, sowie den Nutzen. Die hier enthaltenen Modelle bestehen aus dem Anwendungsfalldiagramm, dem Interaktions- uind Aktivitätsdiagramm für die Geschäftsmodelle, sowie Klassendiagramme für die Informationsmodelle.

- Platform Independent Model (PIM): In diesem Viewpoint werden die Strukturen und das Verhalten des System unabhängig von den später zur Implementation einzusetzenden technischen Grundlagen beschrieben. Dieses umfasst die Geschäfts- und Informationsmodelle aus dem CIM, die aber durch die so genannten Computational-Modelle ergänzt werden. Hier können alle Diagramme von UML zur Struktur- und Verhaltensbeschreibung genutzt werden. Lediglich die Deployment-Modelle bleiben hier auf abstrakte Modelle beschränkt.

- Platform Model (PM): Dieses ist eine technische Spezifikation einer Plattform, beispielsweise einer Java Enterprise Edition-Plattform.

- Platform Specific Model (PSM): Das PSM wird die Spezifikation eines Systems im PIM mit den konkreten Details über die Realisierung auf der ausgewählten Plattform (PM) ergänzt. Dieses erfolgt mittels der Modelltransformation. Das PSM als Ergebnis der Transformation kann wieder alle in UML verfügbaren Modell- und Diagrammformen beinhalten. Zudem können diese durch weitere Informations- und Beschreibungsmittel wie Quellcode oder Schnittstellenbeschreibungen ergänzt werden.

Die MDA ermöglicht somit eine durchgängige Unterstützung der einzelnen Phasen der Entwicklung einer Kontextinfrastruktur und der AAL-Dienste (AM8.2, AM9.2) basierend auf einem gemeinsamen UML-basierenden Kontext-Metamodell (Kap. 4.3.2).

4.3.4 Ecore / EMF

Ecore ist ein auf EMOF (Essential MOF) basierendes Metamodell und wird nachfolgend als Alternative zu UML betrachtet. EMOF ist eine Untermenge von MOF und ermöglicht eine vereinfachte Erstellung von Metamodellen, die nicht die vollständige Ausdrucksmächtigkeit benötigen. Ecore ist eine Grundlage für das Eclipse Modeling Framework (EMF) [143], welches zur automatisierten Erzeugung von Java-Quelltext anhand von strukturierten Modellen dient.

Ecore ist im Vergleich zu UML sehr einfach. Die wesentlichen Konzepte sind „Klassen“, „Datentypen“, „Klassifizierung“, „Attribute“, „Referenzen“, „Pakete“ und „Annotationen“. So repräsentiert „EClass“ eine Klasse mit möglichen Operationen, Attributen und Referenzen. Über die „SuperTypes“-Beziehung zwischen Klassen können Generalisierungs-/Spezialisierungsbeziehungen definiert werden. Die Klasse „EOperation“ ist ein getyptes Element mit Namen. Es hat „eParameters“ als Referenzen. Diese sind getypte Elemente mit Namen. Die Klasse „EAttribute“ repräsentiert ein Attribut mit einem Namen und einem Typen. Die Klasse „EReference“ repräsentiert ein Ende einer Beziehung zwischen zwei Klassen. Die Klasse „EDataType“ repräsentiert den Typ eines Attributs.

Dieses Metamodell kann genutzt werden, um die identifizierten Modellelemente des Kontext-Metamodells abzubilden. Es erlaubt die Definition von Klassen, die die Modellelemente „Kontextentität“, „Kontextrelation“, „Kontextattribut“, „Zulässige Vergleiche zwischen Attributen“, „Situation“ und „Repräsentationform“ repräsentieren. Die Eigenschaften dieser Modellelemente können mit Hilfe der Attribute beschrieben werden. Die Generalisierungs- und Spezialisierungsbeziehung kann ebenfalls ausgedrückt werden. Weiterhin können Beziehungen zwischen den Modellelementen mit Hilfe der Referenzen beschrieben werden. Die Erweiterbarkeit des konkreten Kontextmodells ist aufgrund der Metadaten-Architektur gegeben.

Ein Vorteil bei der Verwendung von Ecore ist die enge Eclipse-Integration. Eclipse [144] ist eine freie integrierte Java-Entwicklungsumgebung. Gemeinsam mit EMF ermöglicht Eclipse die grafische Modellierung und die automatisierte Generierung von Java-Quelltext. Auf Basis von Ecore kann ein eigenes Metamodell erstellt und mit einer grafischen Notation verknüpft werden. Diese Beschreibung kann von Eclipse unmittelbar genutzt werden um die entsprechende Nutzeroberfläche für die Modellierung bereitzustellen. EMF nutzt XML, bzw. XMI, zur persistenten Speicherung eines Modells. Mit Hilfe des JET (Java Emitter Templates) erfolgt die Generierung des entsprechenden Java-Codes. Die Komponente „JMerge“ wird von Eclipse für die Synchronisierung von Änderungen am Modell und den generierten Klassen verwendet.

4.3.5 OWL

Die Web Ontology Language (OWL) [134] ist ein Beispiel für eine Sprache, mit der eine Ontologie beschrieben werden kann. Eine Ontologie ist eine formale Darstellung von Begrifflichkeiten, der zwischen ihnen bestehenden Beziehungen und ihrer beabsichtigten Bedeutung. Zusätzlich kann eine Ontologie Inferenz- und Integritätsregeln beinhalten, die für die Ableitung zusätzlichen Wissens und zur Überprüfung der Wahrheit einer Aussage verwendet werden können. Typische Bestandteile einer Ontologie können wie folgt beschrieben werden:

- Konzepte: Definition der relevanten Begriffe bzw. der in der Ontologie enthaltenen Objekte. Dieses kann beispielsweise der „Bewohner“ (AM6.2.1) sein. Konzepte werden auch als Klassen bezeichnet. Diese können in einer Generalisierungs-/Spezialisierungsbeziehung organisiert sein.

- Instanzen: Instanzen repräsentieren das zugehörige Konzept. Eine Instanz zum Konzept „Bewohner“ kann beispielsweise „Anna Mustermann“ sein.

- Relationen: Relationen beschreiben die möglichen Beziehungen zwischen Konzepten bzw. Instanzen. Als Beispiel kann eine Beziehung „wohnt in“ zwischen den Konzepten „Bewohner“ und „Wohnung“ definiert werden. Relationen werden häufig auch als Eigenschaften bezeichnet.

- Axiome: Dieses sind Aussagen in einer Ontologie, die immer wahr sind. Diese werden dazu verwendet um Wissen abzubilden, die nicht aus anderen Begriffen abgeleitet werden können.

Ontologien werden für die Wissensrepräsentation beispielsweise im semantischen Web verwendet. Dort werden diese für die semantische Suche nach Informationen oder Diensten genutzt. Im Gegensatz zur syntaktischen Suche kann dabei das in der Ontologie vorhandene Wissen berücksichtigt werden. Beispiele für formale Sprachen zur Beschreibung von Ontologien sind beispielsweise RDF-Schema [145], DAML+OIL [146], F-Logic [147] oder OWL.

Die Sprachkomponenten von OWL können auch als dessen Metamodell bezeichnet werden, mit denen konkrete Ontologien erstellt werden können. Die Beschreibung der Ontologie erfolgt mittels der definierten Sprachelemente mit Hilfe von XML. Ein wesentlicher Bestandteil ist das Element „class“. Dieses erlaubt die Definition von Konzepten bzw. Klassen. Das OWL Sprachelement „rdfs:subClassOf“ erlaubt die Definition von Generalisierungs- und Spezialisierungsbeziehungen zwischen Klassen. Mit Hilfe von „rdf:type“ könen Instanzen einer Klasse zugeordnet werden. Relationen zwischen Instanzen von zwei Klassen werden mit Hilfe des Sprachelements „ObjectProperty“ definiert. Mit Hilfe von „DatatypeProperty“ können Relationen zwischen Instanzen von Klassen und RDF-Literalen sowie XML-Schema Datentypen definiert werden. Mit Hilfe von „rdfs:domain“ und „rdfs:range“ können die Grundmenge bzw. der Wertebereich einer Relation angegeben werden. Zudem kann mit Hilfe von „rdfs:subPropertyOf“ eine Relation als Spezialisierung einer bereits bestehenden Relation definiert werden. Zu einer Relation können dessen Eigenschaften beschrieben werden, beispielsweise die Transivität oder Symmetrie. Zudem können Einschränkungen auf dem Wertebereich definiert werden. Diese Aufzählung stellt nur einen kleinen Ausschnitt auf den vorhandenen Sprachkonzepten dar. In Abhängigkeit der benötigten Ausdrucksstärke stehen drei unter-

schiedliche Ausprägungen von OWL zur Verfügung: OWL Lite, OWL DL und OWL Full.

Mit CoDaMos und CONON existieren zwei Beispiele für die Verwendung von OWL für die Kontextmodellierung. Diese bilden die im Kontext-Metamodell benötigten Modellelemente auf die Sprachelemente von OWL ab. So wird beispielsweise eine „Kontextentität“ durch das Element „class“ repräsentiert (AM1.1, AM2.1, AM3.1, AM6.2.1). Die Generalisierungs- und Spezialisierungsbeziehungen zwischen Klassen können für die Definition von solchen Beziehungen zwischen Kontextentitäten genutzt werden (AM2.2, AM3.2, AM6.2.1.1). Die Definition einer „Kontextrelation“ (AM1.2, AM2.3, AM3.3, AM6.2.2) kann mit Hilfe der „ObjectProperty“ erfolgen. Das Modellelement „Kontextattribut“ (AM1.3, AM2.4, AM3.4, AM6.2.3) kann mit Hilfe der „DatatypeProperty“ in eine Ontologie abgebildet werden. Die Beschreibung der zulässigen Vergleiche zwischen Kontextattributen (AM5.1.1, AM6.2.3.3) lassen sich nicht unmittelbar auf die OWL Sprachelemente abbilden. Das Modellelement „Situation“ (AM5.1.2, AM6.2.5) könnte gleich wie „Kontextentität“ auf das Element „class“ abgebildet werden. Die Abhängigkeiten zwischen dem Zustand der Kontextentitäten und einer Situation kann mit Hilfe der Inferenzregeln in Form der in OWL DL vorhandenen „Description Logic“ beschrieben werden. Problematisch ist hierbei jedoch, dass die Konzepte „Kontextentität“ und „Situation“ auf Ebene der Ontologie verschwinden. Gleiches gilt beispielsweise für das Konzept der „Repräsentationsform“ (AM6.2.4). Die Konzepte eines Kontext-Metamodells sind nicht mehr explizit sichtbar. Eine Möglichkeit dieses Problem zu umgehen besteht darin diese Konzepte in die Ontologie aufzunehmen. So kann eine Klasse „Kontextentität“ definiert werden. Eine Kontextentität „Bewohner“ kann dieser durch eine Relation „is a“ zugeordnet werden. Daraus folgt jedoch eine Durchmischung der einzelnen Modellebenen. Die einzelnen Ebenen der Metadaten-Architektur sind hierbei nicht mehr eindeutig zu trennen. Aus diesem Grund heraus wird die Verwendung einer Ontologie als Basis für die Umsetzung des Kontext-Metamodells nicht weiter betrachtet.

Mit der Begriffsdefinition von OWL gibt es auch eine zugehörige grafische Notation. Dadurch können Kontextmodelle basierend auf OWL in visueller Form dargestellt werden. Die Notation kann durch den Entwickler verstanden werden. Für den technischen Laien ist die Verwendung von OWL und der entsprechenden Notation beispielsweise für die Definition des kontextadaptiven Verhaltens eines AAL-Dienstes (AM7.2.3) dennoch nicht geeignet, da die zugrundeliegenden Konzepte und Begriffe verstanden werden müssen.

Ein weiterer Grund gegen die Verwendung von Ontologien zur Umsetzung der Kontextmodelle ist die schlechte Performanz derzeit existierender Inferenzmaschinen [148]. In [58] wird daher vorgeschlagen das eigentliche Kontextmodell mit einer Ontologie zu überlagern, worin lediglich das Inferenz- und Integritätswissen beschrieben wird. In [149] beschreiben die Autoren den Einsatz von

UML als Basis für eine Ontologie-Modellierungssprache. Die Eigenschaften und die existierenden Werkzeuge von UML können hier auch für die Modellierung von Ontologien genutzt werden. In einem weiteren Ansatz wurde basierend auf MOF mit dem Ontology Definition Metamodel (ODM) [150] ein Metamodell zur Definition von Ontologien geschaffen. Dieses besteht aus drei separaten Metamodellen zur Definition von Ontologien mit Hilfe von RDFS, OWL oder Topic Maps. In ODM sind auch drei UML Profile definiert, mit denen die Verwendung der UML Notation und vorhandener Werkzeuge für die Modellierung von Ontologien und die Generierung von XML-basierten Repräsentationen dieser Sprachen ermöglicht wird. Dieser Ansatz würde es ermöglichen das Inferenz- und Integritätswissen in einer Ontologiesprache in der bekannten UML-Notation zu beschreiben und dann in Form von XML zur weiteren Verwendung vorzuhalten.

4.3.6 Gewählter Ansatz

Für die Definition des Kontext-Metamodells wird in den nachfolgenden Kapiteln UML verwendet. Folgende Eigenschaften von UML sind die Grundlage dieser Entscheidung:

- Das UML-Metamodell ermöglicht die Definition der benötigten Modellelemente des Kontext-Metamodells.

- Explizite Trennung zwischen den unterschiedlichen Modellebenen aufbauend auf der MOF Metadaten-Architektur. Dadurch wird eine Vermischung der unterschiedlichen Ebenen zugehörigen Begrifflichkeiten vermieden. Dieses wäre beispielsweise bei der Verwendung von OWL der Fall.

- Unterscheidung zwischen Modell und Diagramm. Durch die Trennung zwischen Modell und Diagrammen ist es möglich unterschiedliche Sichten auf das Modell zu definieren. Ein Diagramm stellt einen Teil einer Sicht auf das Modell dar und muss also niemals alle in dem Modell enthaltenen Informationen reflektieren. Genauso kann ein Modellelement in einem oder mehreren Diagrammen enthalten sein. In Hinblick auf die Unterstützung der unterschiedlichen Entwicklungsphasen können Sichten definiert werden, die den jeweils benötigten Detaillierungsgrad auf das Modell repräsentieren. Dieses ist bei der Verwendung von XML oder auch EMF nicht gegeben.

- Möglichkeit der Integration unterschiedlicher UML-Profiles in die Modellierung, z.B. der ODM zur Beschreibung von Inferenz- und Integritätsregeln.

- Vielzahl der vorhandenen Technologien basierend auf UML zur Modellierung und Modelltransformation, insbesondere in Zusammenhang mit der Model Driven Architecture. Diese können für die Methodik der Kontextmodellierung und der Realisierung des kontextadaptiven Verhaltens der AAL-Dienste und der Kontextinfrastruktur eingesetzt werden.

Ein Vorteil von Ecore / EMF ist das im Gegensatz zu UML einfachere Metamodell. Dieses erleichtert die Definition des Kontext-Metamodells. Ebenso unterstützt die Eclipse Entwicklungsumgebung auf einfache Weise die Definition eigener Metamodelle auf Basis von Ecore und die Definition einer zugehörigen grafischen Notation. Nachteilig ist die explizite Ausrichtung von Eclipse auf Java als Zielplattform.

Mit der Entscheidung für UML stellt sich die Frage nach der geeigneten Umsetzung des Kontext-Metamodells entlang der Metadaten-Architektur. Folgende Lösungsvarianten stehen zur Verfügung:

- Variante 1: Erstellung eines eigenständigen Kontext-Metamodells auf Ebene M2 der MOF Metadaten-Architektur. Mit Hilfe der Modellelemente auf Ebene M3 wird das Kontext-Metamodell definiert. Diese ist unabhängig vom UML-Metamodell.

- Variante 2: Erweiterung des UML-Metamodells um das Kontext-Metamodell mit Hilfe der Modellelemente auf Ebene M3. Mit Hilfe der Modellelemente auf Ebene M3 wird das bestehende UML-Metamodell um das Kontext-Metamodell erweitert. Dieses erfolgt in einem eigenen Paket.

- Variante 3: Erweiterung des UML-Metamodells um das Kontext-Metamodell mit Hilfe von Profilen und Stereotypen. Mit dem UML-Profiling ist ein leichtgewichtiger Mechanismus zur anwendungsspezifischen Anpassung von UML vorhanden. Ein Stereotyp beschreibt, wie eine vorhandene Klasse des UML-Metamodells oder ein bereits definierter Stereotyp verwendet werden können, um ein anwendungsspezifisches Konzept zu beschreiben. Hier erfolgt die Erweiterung des UML-Metamodells mit Modellelementen der Ebene M2.

- Variante 4: Erstellung des Kontext-Metamodell auf Ebene M1 mit Hilfe von Klassendiagrammen. Die Instanzen des Kontext-Metamodells sind die konkreten Kontextmodelle auf Ebene M0. Die Instanzen innerhalb der Kontextmodelle befinden sich dann in einer untergeordneten Ebene. Somit würde die Metadaten-Architektur eine zusätzliche Ebene bekommen.

Variante 4 entspricht nicht der MOF Metadaten-Architektur, da hier eine weitere Ebene eingeschoben wird. Dieses führt dazu, dass die auf der MOF Meta-

daten-Architektur aufbauenden Werkzeuge nicht verwendet werden können. Zudem ist dieser Ansatz nicht sauber, da mit dem Metamodell nicht die Umsetzung des Kontextmodells in Form von Klassen ausgedrückt werden soll. Variante 1 und 2 haben den Vorteil, dass die abstrakte Syntax des Kontext-Metamodells explizit zu erkennen ist, wobei sich diese in der Variante 3 nur indirekt ausdrückt. Ein Nachteil dieser Varianten dagegen ist, dass viele existierende Werkzeuge nicht unmittelbar eingesetzt werden können und Anpassungen insbesondere im Repository und in der Visualisierung vorgenommen werden müssen. In [151] vergleichen die Autoren die beiden Varianten und identifizieren im Detail die jeweiligen Vor- und Nachteile. Sie schlagen eine konkrete Architektur für die Umsetzung domänen-spezifischer Erweiterungen von UML für entsprechende Werkzeuge vor, mit der die nativen Umsetzung eines Metamodells entsprechend Variante 1 und 2 mit den Vorteilen der direkten Werkzeugunterstützung erreicht werden soll. Exemplarisch umgesetzt haben die Autoren dieses anhand eines frei verfügbaren Modellierungswerkzeugs.

Viele Autoren von domänenspezifischen Metamodellen gehen aufgrund der bekannten Problematik zweigleisig vor. Sie definieren die abstrakte Syntax des Metamodells mit Hilfe von Variante 1 bzw. 2 mittels der Modellelemente auf Ebene M3. Die konkrete Syntax, d.h. die Umsetzung dieses Metamodells, beschreiben sie durch die Verwendung der UML-Profile entsprechend Variante 3. Ein Beispiel für ein solches Vorgehen ist beispielsweise in der Definition von SecureUML [152] zu finden. Die Definition des Kontext-Metamodells erfolgt daher nachfolgend unter Verwendung von Variante 2 durch eine Erweiterung des UML-Metamodells durch Verwendung der Modellelemente auf Ebene M3. Die konkrete Umsetzung kann dann bei Bedarf mit Hilfe der UML-Profiles erfolgen. Diese ist aber nicht Gegenstand der weiteren Ausführungen in den nachfolgenden Kapiteln.

In Verbindung mit UML kann die Entwicklung kontextadaptiver AAL-Dienste entsprechend dem Grundgedanken der MDA erfolgen. Die Grundlage hierfür ist die Definition eines Kontext-Metamodells für die 3 Ebenen der Kontextmodellierung „Infrastruktur“, „Dienste“ und „Nutzerinteraktion“. Für die Entwicklungs- und Nutzungsphasen werden dann auf dem Kontext-Metamodell die jeweils relevanten Ausschnitte bzw. Sichten für die Modelltypen „Informell“, „Konzeptuell“, „Operativ“ und „Deskriptiv“ definiert. Diese befinden sich auf Ebene M2 der MOF Metadaten-Architektur. Beim Übergang zwischen den Modelltypen erfolgt eine zunehmende Anreicherung der Sichten um weitere Details. Im Gegensatz zur Anwendungslogik eines AAL-Dienstes erfolgt am Ende der Transformation keine Generierung eines Quellcode für eine Zielplattform. Der Grund dafür ist, dass ein generelles Verhalten einer Kontextlogik identifiziert werden kann, die unabhängig von der konkreten Ausprägung eines Kontextmodells ist. Dazu gehört beispielsweise die Einbindung von Sensorik, die Aggregation und Interpretation von Sensorinformationen und die Bereitstellung von Kontextinformationen für die eigentliche Anwendungslogik [153]. Dieses generelle Verhalten kann durch einen Kontextserver umgesetzt werden.

Die dienstespezifische Ausprägung dieses Verhaltens wird durch das konkrete Kontextmodell beschrieben. Dieses befindet sich auf Ebene M1 der MOF Metadaten-Architektur. Am Ende der Transformation wird daher eine Beschreibung des Kontextmodells erzeugt, welche vom Kontextserver für die Umsetzung der spezifischen Kontextlogik verwendet werden kann. In der nachfolgenden Abbildung werden die Transformation zwischen den Sichten und die Umsetzung des Kontextmodells im Kontextserver dargestellt.

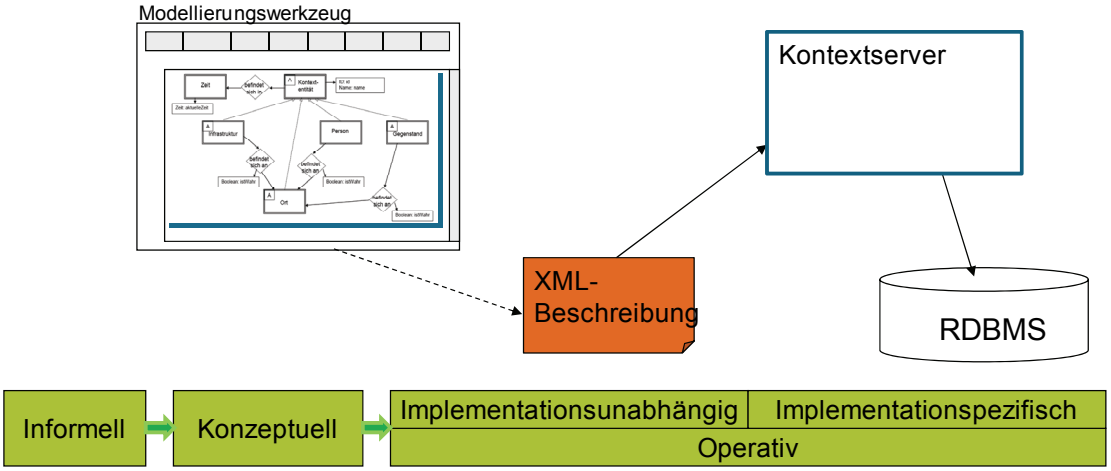


Abbildung 22: Modellbasierte Umsetzung der Kontextlogik

4.4 Zusammenfassung

In diesem Kapitel wurden zunächst aktuelle Ansätze zur Kontextmodellierung beschrieben und anhand der früher identifizierten Anforderungen bewertet. Dabei wurde ersichtlich, dass keiner dieser Ansätze ganz ausreicht. Insbesondere die Anforderungen bezüglich der generellen Eigenschaften, wie die Trennung von Diensten und Infrastruktur, die Unterstützung dienstetyp-spezifischer Kontextmodelle, die endnutzergerechte Interaktion sowie die Unterstützung unterschiedlicher Phasen der Entwicklung und der Nutzung, spielen für die Umsetzung kontextadaptiver Dienste im AAL eine wesentliche Rolle. Während die Anforderungen an die konkrete Ausprägung eines Kontextmodells aufgrund der zugrundeliegenden Metamodelle in den aktuellen Kontextmodellen umsetzbar sind, ist dieses bei den generellen Eigenschaften nicht der Fall, sodass ein geeigneter Ansatz zur Kontextmodellierung gefunden werden muss. Die Grundlage dafür ist eine zweidimensionale Kontextmodellierung. In der ersten Dimension wird der Einsatzzweck eines Kontextmodells betrachtet. Hierin sind drei Ebenen der Kontextmodellierung mit einem jeweils unterschiedlichen Fokus auf die Beschreibung der Kontextinfrastruktur, des kontextadaptiven Verhaltens der Dienste und der Interaktion mit dem Bewohner definiert. In der zweiten Dimension wird die unterschiedliche Verwendung eines Kontext-

modells in den Phasen der Entwicklung und der Nutzung dargestellt. Aufgrund dieser Differenzierung können vier unterschiedliche Modelltypen identifiziert werden, die in den jeweiligen Phasen Verwendung finden. Auch ergeben sich unterschiedliche Sichten, in denen die verschiedenen Anforderungen zuzuordnen sind. Unter Berücksichtigung dieser Anforderungen werden dann die jeweils notwendigen Konzepte, die geeigneten inhaltlichen Ausrichtungen, Abstraktionsgrade und Präsentationsformen definiert, was in den nachfolgenden Kapiteln geschehen soll. Zwischen diesen Sichten lassen sich Übergänge identifizieren. Ihre Umsetzung muss auf der Basis eines gemeinsamen Metamodells erfolgen. Die Definition des Kontext-Metamodells erfolgt durch Erweiterung des UML-Metamodells mit Hilfe der Modellelemente auf Ebene M3. Die konkrete Umsetzung des Kontext-Metamodells kann dann mit Hilfe von UML-Profiling erfolgen. Diese Grundlagen der zweidimensionalen Kontextmodellierung im AAL sind in folgender Abbildung dargestellt.

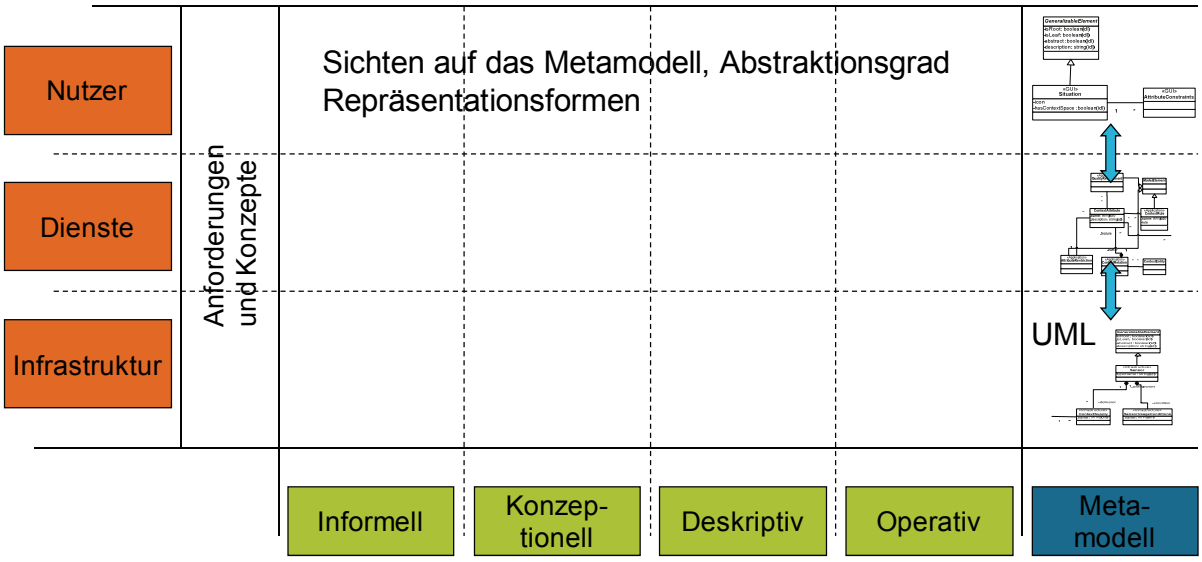


Abbildung 23: Zweidimensionale Kontextmodellierung im AAL

5 Kontextmodellierung auf der Ebene der Infrastruktur

Im vorhergehenden Kapitel wurde die zweidimensionale Betrachtung der Kontextmodellierung im AAL motiviert und eingeführt. Auf dieser Basis wird nun die Ebene der Infrastruktur im Detail betrachtet und konzipiert. Zunächst wird ihr Einsatzzweck beschrieben und anschließend werden die Anforderungen, die dieser Ebene zuzuordnen sind, zusammengefasst. Daraufhin sind die notwendigen Konzepte zur Umsetzung dieser Anforderungen darzustellen. Auf ihnen basierend wird das in dieser Ebene benötigte Metamodell definiert um dadurch abschließend die vier Modelltypen bzw. Sichten für die jeweiligen Phasen herausarbeiten zu können.

5.1 Einsatzzweck

Diese Ebene dient dem Aufbau und dem Betrieb der Kontextinfrastruktur des Bewohners unabhängig von den darauf aufsetzenden AAL-Diensten. Hierzu gehören eine Beschreibung der in der häuslichen Umgebung vorhandenen Komponenten, aber auch der Sensorik und der durch sie gelieferten Kontextinformationen.

Wie bereits in Kap. 3.3.1 beschrieben, besitzt die häusliche intelligente Infrastruktur eine eigene kontextadaptive Funktionalität (strukturelle Kontextadaptivität), welche von den AAL-Diensten genutzt werden kann. Ein Beispiel hierfür ist die Kommunikation mit dem Bewohner über ein in seiner Nähe befindliches und geeignetes Endgerät oder das Folgen eines AAL-Dienstes auf den Hardwareknoten, die sich am Bewohner befinden. Beides sind Beispiele aus dem Ubiquitous Computing. Daneben besitzt diese Ebene die Aufgabe, Kontextinformationen in Abhängigkeit von der in der Wohnung verfügbaren Sensorik bereitzustellen. Sie impliziert die Möglichkeit der Ad-hoc-Integration von neuer Sensorik.

5.2 Anforderungen

Im Kapitel 3 wurden bereits, unabhängig von der zweidimensionalen Betrachtung der Kontextmodellierung, die bestehenden Anforderungen aus Sicht des AAL identifiziert. Nachfolgend werden auf der Grundlage des Einsatzzwecks die daraus folgenden Anwendungsfälle identifiziert. Anhand dieser Anwendungsfälle werden die Anforderungen zusammengefasst, die dieser Ebene zuzuordnen sind.

Folgende Anwendungsfälle lassen sich aus dem Einsatzzweck ableiten:

- Bereitstellung von Kontextinformationen über die verfügbaren Bestandteile der häuslichen Infrastruktur zur Realisierung der strukturellen Kontextadaptivität.

- Bereitstellung von generellen Kontextinformationen in Abhängigkeit der verfügbaren Sensorik.

- Ad-hoc-Integration neuer Sensorik.

Die Anforderung der Bereitstellung von Kontextinformationen über die verfügbaren Bestandteile der häuslichen Infrastruktur ist in Kap. 3.2 beschrieben. Zusammengefasst muss hierfür das Kontextmodell in der Lage sein, die Eigenschaften einer Infrastruktur in Bezug auf die Vernetzung (AK1), die Hardwaretopologie (AK2) und der darauf vorhandenen Software (AK3) zu beschreiben. Die sich daraus ergebenden Anforderungen an das Metamodell sind in AM1, AM2 und AM3 zusammengefasst.

Für die Erhebung dieser Kontextinformationen werden Sensoren verwendet. Das Kontextmodell soll auf dieser Ebene Informationen über die verfügbaren Sensoren und der von ihnen gelieferten Kontextinformationen bereitstellen. Bestandteile der Beschreibung eines Sensors sind dessen Eigenschaften, z.B. in Bezug auf die Robustheit und Zuverlässigkeit (AM4.2.1), der Bezug zu einem realen Objekt der häuslichen Umgebung (AM4.2.2), der Grad der Intrusivität (AM4.2.3), die Qualität der Informationserkennung (AM4.2.4), die Einheit (AM4.2.5), die Abdeckung (AM4.2.6), die Nutzungsmodalitäten (AM4.2.7, AM4.2.8) oder die Art der Bereitstellung (AM4.2.9). Der Bezug zu einem realen Objekt der häuslichen Umgebung impliziert, dass diese ebenfalls Bestandteil des Kontextmodells sein müssen. Als Beispiel sei hier ein Blutdrucksensor genannt. Der Bezug muss hier zum Blutdruck einer konkreten Person hergestellt werden. Eine Entität „Person“ und das zugehörige Attribut „Blutdruck“ müssen somit ebenfalls im Kontextmodell enthalten sein. Für die Zuordnung des Blutdrucksensors zu einer konkreten Person, z.B. „Frau Meier“ muss diese Entität über ein entsprechendes Attribut identifizierbar sein.

Die Erhebung von Kontextinformationen über Sensorik ist mit einer generellen Unsicherheit verbunden (Kap. 2.2.8). Daher kann es hilfreich sein, von ihr gelieferte Kontextinformationen auf der Basis von bekanntem Regelwissen zu validieren. Ist solches vorhanden, so sollte es in Form einer Integritätsregel das Kontextmodell anreichern und ermöglichen, valide Zustände und Übergänge im Kontextmodell zu beschreiben, sodass sie von der Kontextinfrastruktur zur Überprüfung der Sensorinformationen genutzt werden kann. Ein Beispiel für eine solche Regel ist, dass eine Person an einem Ort o_1 nicht zu einem Zeitpunkt t_{n+1} an einem Ort o_2 geortet werden kann, der von o_1 mit einer Distanz > $d_{max}(t_{n+1}-t_n)$ entfernt ist.

Neben der Validierung kann Regelwissen genutzt werden, um Kontextinformationen abzuleiten, für die keine Sensorik vorhanden ist. Wenn solches Wissen vorhanden ist, sollte dieses in Form einer Inferenzregel das Kontextmodell anreichern. Darüber hinaus ist zu fordern, dass Abhängigkeiten zwischen Elementen des Kontextmodells, die für die Bereitstellung zusätzlicher Kontextinformationen genutzt werden können, zu beschreiben sind. Ein Beispiel hierfür ist die Abhängigkeit zwischen Tageszeit und Helligkeit. Ist kein Helligkeitssensor vorhanden, so kann von der aktuellen Tageszeit auf die Helligkeit geschlossen werden. Auch für diese Art der Gewinnung von Kontextinformationen gelten die Anforderungen zur Beschreibung von Sensoreigenschaften.

Die Ad-hoc-Integration von Sensoren unterscheidet zwischen der Einbindung bereits bekannter Sensorik (AM4.1.1) und neuer Sensorarten (AM4.1.2). Da Sensoren einen Bezug zum realen Objekt der häuslichen Umgebung benötigen, führt diese Feststellung zur Anforderung von Erweiterbarkeit des Kontextmodells. Es müssen neue Entitäten oder neue Attribute zu schon bestehenden aufgenommen werden können.

5.3 Konzepte

Ausgehend von den identifizierten Anforderungen werden nun die auf dieser Ebene relevanten Konzepte vorgestellt. Sie gehen unmittelbar auf das nachfolgend vorgestellte Metamodell über. Folgende Konzepte liegen der Kontextmodellierung auf der Ebene der Infrastruktur zugrunde:

- Modellelemente zur Beschreibung der häuslichen Infrastruktur und der verfügbaren generellen Kontextinformationen
- Explizite Beschreibung der Sensorik im Metamodell
- Identifikation gemeinsamer Eigenschaften von Entitäten

Sie werden nachfolgend im Detail beschrieben.

5.3.1 Modellelemente zur Beschreibung von Kontextinformationen

Aus der Betrachtung der Infrastruktur sind bereits eine Reihe von Entitäten identifiziert worden, die Bestandteil eines Kontextmodells sein müssen. Diese sind beispielsweise Vernetzung, Hardware und Software. Ein Hinweis auf eine geeignete Modellierung dieser Entitäten ist in CoDAMoS [133] zu finden. Weitere können aus der Betrachtung der einzelnen Dienstetypen abgeleitet werden. Ein Beispiel für eine Modellierung dieser Art von Kontextinformationen ist in der CONtext ONtology [95] beschrieben. Aus der Bewertung bestehender Ansätze (Kap. 4.1) wurde ersichtlich, dass es kein Kontextmodell gibt, welches

alle Anforderungen in Hinsicht auf die konkrete Ausprägung erfüllt, besonders die, neuartige Sensortypen einzubinden. Daraus ergibt sich ja die Anforderung nach der Erweiterbarkeit eines solchen Kontextmodells.

Eine Lösung zur Umsetzung dieser Anforderungen stellt die Definition von Modellelementen auf der Meta-Ebene dar, mit denen das konkrete Kontextmodell definiert werden kann. Bereits bekannte Modellelemente sind in Kap. 2.2.5 vorgestellt und werden auf Ebene der Infrastruktur ebenfalls benötigt (AM1, AM2, AM3):

- Kontextentität: Sie definiert das zu beobachtende Objekt der realen Umgebung.
- Kontextrelation: Sie definiert zu beobachtende Beziehungen zwischen Kontextentitäten.
- Kontextattribut: Sie definiert die beobachtbaren Eigenschaften einer Kontextentität oder einer Kontextrelation.

Diese Modellelemente sind aber für die Umsetzung der Anforderungen noch nicht ausreichend. Eine besonders zu beachtende Anforderung ist die Beschreibung der Einheit einer Kontextinformation (AM4.2.5). Die möglichen Einheiten hängen von der Art der von einem Sensor gelieferten Kontextinformation ab. Werden Temperaturinformationen geliefert, so sind die möglichen Einheiten „Fahrenheit“ oder „Grad“. Die mögliche Zuordnung einer Kontextinformation zu möglichen Einheiten soll durch folgende zusätzliche Modellelemente beschrieben werden:

- Kontextinformation: Dieses Modellelement beschreibt die von einem Sensor gelieferte Kontextinformation.
- Kontextdimension: Sie beschreibt die möglichen Arten von Kontextinformationen.
- Repräsentationstyp: Er beschreibt die möglichen Repräsentationsformen oder Einheiten in denen eine Kontextinformation bereitgestellt werden kann. Diese ist abhängig von der zugehörigen Kontextdimension.

Diese Modellelemente werden in Kap. 5.4 im Detail beschrieben. Mit ihrer Hilfe wird ein konkretes Kontextmodell auf der Ebene der Infrastruktur definiert.

5.3.2 Explizite Beschreibung der Sensorik im Metamodell

Ein weiteres Konzept ist die explizite Definition eines Modellelements „Sensorik“ auf der Ebene des Metamodells. Dieses ist, im Vergleich zu den sonstigen Infrastrukturkomponenten (Kap. 3.2) eine Sonderbehandlung der Sensorik. Auf der Ebene des Metamodells soll durch dieses Element die Rolle der Sensorik als Quelle der verfügbaren Kontextinformationen beschrieben und der Bezug der Sensorik zu einem Kontextattribut einer Entität oder einer Relation ausgedrückt werden. Zudem sollen die möglichen Eigenschaften der Sensorik (AM4.2.1 – AM4.2.9) beschrieben werden.

5.3.3 Identifikation gemeinsamer Eigenschaften von Entitäten

Eine Aufgabe des Kontextmodells auf dieser Ebene ist die Zuordnung der von den Sensoren gelieferten Kontextinformationen zu einem realen Objekt der häuslichen Umgebung (AM4.2.2). Mit dieser Aufgabenstellung werden die Kontextentitäten und deren beobachtbaren Eigenschaften bzw. Kontextattribute im Kontextmodell modelliert. Verschiedene Arten von Kontextinformationen sind nicht ausschließlich einer Entität vorbehalten. So kann beispielsweise die Position sowohl die beobachtbare Eigenschaft einer Person als auch die eines jedes anderen beweglichen Objektes sein. Entsprechend kann ein Ortungssensor nicht nur einer Person, sondern beispielweise auch einem Buch zugeordnet werden. Daher sollte es möglich sein, im Kontextmodell die gemeinsamen beobachtbaren Eigenschaften von Kontextentitäten auszudrücken. Sie können in Form einer Vererbungshierarchie formuliert sein. Zu ihrer Beschreibung soll das Metamodell die Definition von Generalisierungs- oder Spezialisierungsbeziehungen zwischen Kontextentitäten benutzen. Sie soll auch eine Mehrfachvererbung berücksichtigen, um verschiedene Gruppen gemeinsamer Eigenschaften modellieren zu können und abstrakte Kontextentitäten zuzulassen, welche nur die gemeinsamen Eigenschaften der spezialisierenden Entitäten beschreiben. So kann beispielsweise eine abstrakte Kontextentität „Bewegliches Objekt“ definiert werden, welches die Position als Kontextattribut besitzt. Von dieser können dann die Entitäten „Person“ und „Buch“ spezialisiert werden, die dessen Attribute erben. Die Zuordnung eines Ortungssensors kann dann im Kontextmodell zur Kontextentität „Bewegliches Objekt“ erfolgen.

5.4 Metamodell

Nachfolgend werden die Elemente des Metamodells auf Ebene der Infrastruktur beschrieben. Dieses Metamodell ist die Grundlage für alle Modelltypen bzw. Sichten auf dieser Ebene. Je nach Modelltyp besitzen die entsprechenden Sichten einen mehr oder weniger eingeschränkten Umfang des Metamodells. Die Definition des Metamodells erfolgt nachfolgend durch Erweiterung des UML-Metamodells in einem Paket „Context.Infrastructure“. Für die Defini-

tion grundlegender Eigenschaften von Modellelementen des Kontext-Metamodells wird auf die grundlegenden Modellierungskonzepte von UML zugegriffen, die dort im Paket „Kernel“ definiert sind. Dieses wird in der nachfolgenden Abbildung dargestellt.

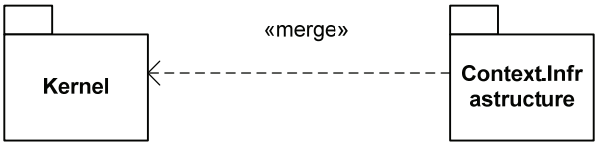


Abbildung 24: Aufteilung des Metamodells in Pakete

In dem Paket „Context.Infrastructure“ werden die Elemente des Metamodells definiert, die auf der Ebene der Infrastruktur für die Kontextmodellierung benötigt werden. Die wesentlichen Modellelemente wurden bereits in der Konzeption (Kap. 5.3) identifiziert. In englischer Bezeichnung sind das die Modellelemente „ContextEntitity“, „ContextRelation“, „ContextAttribute“, „ContextDimension“, „ContextInformation“ und „ContextRule“. Hierüber werden die Entitäten und Relationen sowie deren Zustände und Abhängigkeiten repräsentiert. Ein weiteres Modellelement ist „Sensor“. Es repräsentiert die in der häuslichen Umgebung vorhandene Sensorik mitsamt seinen Eigenschaften, z.B. die Nutzungsmodalitäten, sowie die Einschränkungen und Fähigkeiten (AM4.2.1 – AM4.2.9). Hierfür werden weitere Modellelemente benötigt. Eine vertiefte Betrachtung dieser Eigenschaften und deren Modellierung erfolgt in einer ergänzenden Diplomarbeit [154]. Die Zielsetzung der Diplomarbeit ist die Konzeption und Realisierung einer „Sensor Registry“, welche Bestandteil einer intelligenten häuslichen Umgebung ist. Die darin benötigte Sensorbeschreibungssprache definiert Modellelemente, die zur vollständigen Beschreibung der Sensorik benötigt werden. Das Modellelement „Sensor“ bildet die Verknüpfung zu den Ergebnissen der Diplomarbeit, weshalb eine vertiefende Betrachtung der Sensorik und deren Beschreibung an dieser Stelle nicht weiter erfolgt.

Die identifizierten Modellelemente und deren Beziehungen zueinander werden in dem nachfolgenden Klassendiagramm gezeigt und anschließend in alphabetischer Reihenfolge detailliert beschrieben.

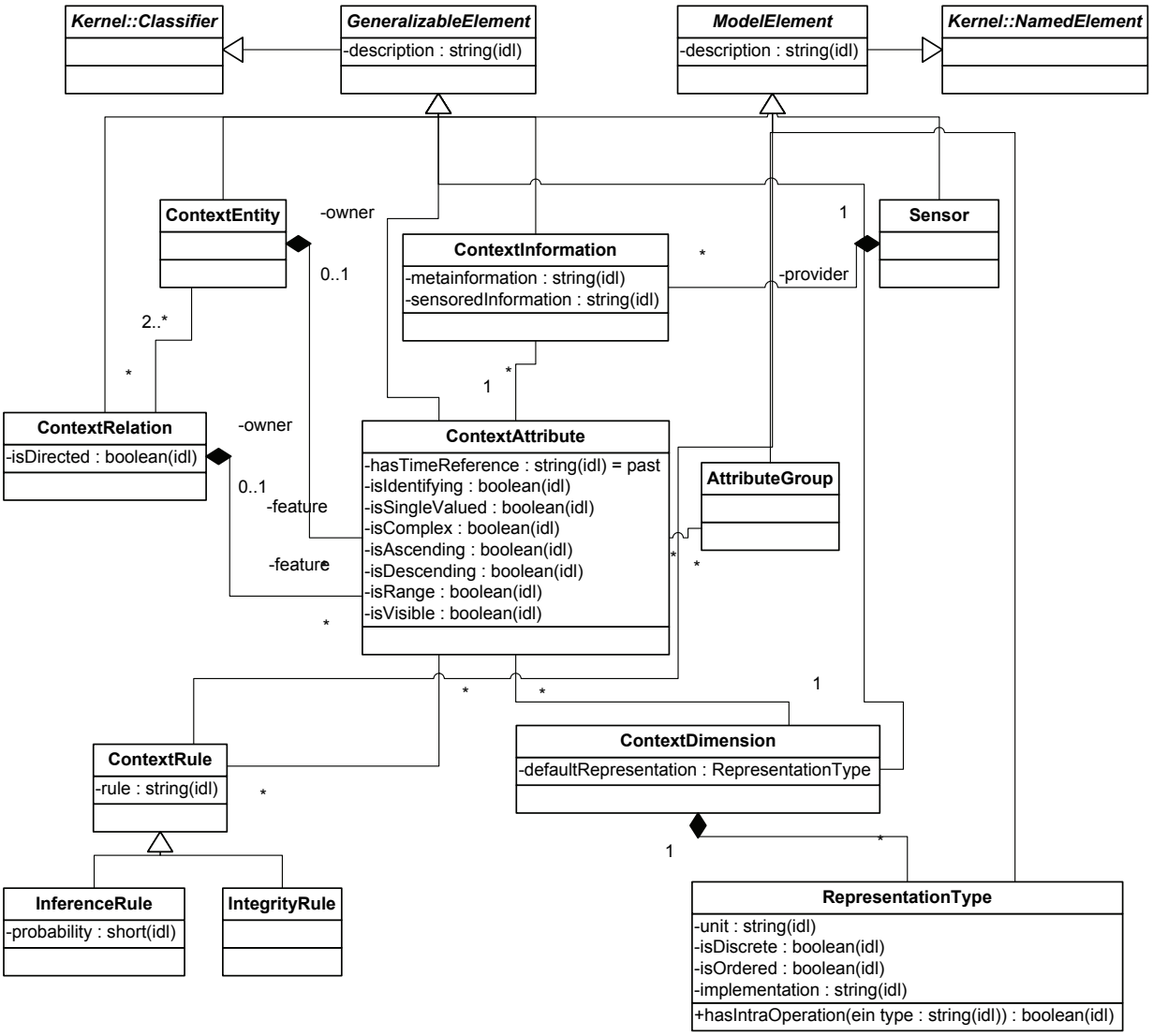


Abbildung 25: Modellelemente im Package „Context.Infrastructure“

5.4.1 AttributeGroup

Generalisierungen

- „ModelElement“

Beschreibung

Das Modellelement „AttributeGroup“ ermöglicht die Gruppierung von Kontextattributen zu zusammengehörigen Einheiten (Kap. 2.2.5.3). Dadurch können beispielsweise der Blutdruck und der Blutzucker zu einer Gruppe „Vitalparameter“ zusammengefasst werden.

Attribute

Dieses Modellelement besitzt keine eigenen Attribute.

Beziehungen

- Eine „AttributeGroup“ kann mehrere Kontextattribute zusammenfassen.

5.4.2 ContextAttribute

Generalisierungen

- „GeneralizableElement“

Beschreibung

Das Modellelement „ContextAttribute“ beschreibt die beobachtbare Eigenschaft einer Kontextentität oder einer Kontextrelation. Der Name des Attributs muss darin eindeutig sein. Ein Attribut kann unterschiedliche Eigenschaften besitzen, die in Kap. 2.2.5.3 beschrieben sind.

Attribute

- hasTimeReference : string

  „past“: Das Kontextattribut hat einen expliziten Zeitbezug in die Vergangenheit. Die Historie ist Bestandteil des Kontextattributs.

  „present“: Der Zeitbezug des Kontextattributs bezieht sich auf die Gegenwart.

  „future“: Das Kontextattribut hat einen expliziten Zeitbezug in die Zukunft.

Ist kein Argument angegeben, so bezieht sich der Zeitbezug lediglich auf die Gegenwart. Es können kommasepariert mehrere Argumente angegeben werden.

- isIdentifying : boolean

  true: Durch den Wert des Kontextattributs wird die Kontextentität eindeutig identifiziert.

  false: Der Wert des Kontextattributs ist nicht eindeutig und kann daher nicht für die Identifizierung einer Kontextentität genutzt werden.

- isSingleValued : boolean

  true: Das Kontextattribut kann nur einen Wert annehmen.

  false: Das Kontextattribut kann mehrere Werte annehmen.

- isComplex : boolean

  true: Das Kontextattribut kann einen zusammengesetzten komplexen Wert annehmen.

  false: Das Kontextattribut kann nur einen einfachen Wert annehmen.

- isAscending : boolean

  true: Die fortlaufenden Werte des Kontextattributs sind aufsteigend.

  false: Die fortlaufenden Werte des Kontextattributs sind nicht zwingend aufsteigend.

  Die Beschreibung dieser Eigenschaft ermöglicht es die Gültigkeit des Werts des Kontextattributs zu überprüfen. Sind die fortlaufenden Werte aufsteigend, so darf eine vom Sensor gemeldete neue Kontextinformation keinen niedrigeren Wert besitzen.

- isDescending : boolean

  true: Die fortlaufenden Werte des Kontextattributs sind absteigend.

  false: Die fortlaufenden Werte des Kontextattributs sind nicht zwingend absteigend.

  Die Beschreibung dieser Eigenschaft ermöglicht es die Gültigkeit des Werts des Kontextattributs zu überprüfen. Sind die fortlaufenden Werte absteigend, so darf eine vom Sensor gemeldete neue Kontextinformation keinen höheren Wert besitzen.

- isRange : boolean

  true: Das Kontextattribut beschreibt einen Bereich innerhalb der Wertemenge der zugrundeliegenden Dimension, z.B. ein Zeitabschnitt.

  false: Das Kontextattribut beschreibt einen Punkt innerhalb der Wertemenge der zugrundeliegenden Dimension, z.B. einen Zeitpunkt.

- isVisible : boolean

  true: Das Kontextattribut ist für die übergeordnete Ebene „Dienste“ sichtbar.

  false: Das Kontextattribut ist nicht in der übergeordneten Ebene „Dienste“ sichtbar. Es wird für interne Aufgaben des Kontextserver benötigt, z.B. für die Verwaltung von Kontextentitäten in einer „Entity Registry“ (Kap. 5.8.2, 8.1)

Beziehungen

- Das Modellelement „ContextAttribute“ steht in einer Aggregationsbeziehung zu den Modellelementen „ContextEntity“ oder „ContextRelation“. Sie beschreibt die Eigenschaften „feature“ einer Kontextentität bzw. einer Kontextrelation. Diese wiederum sind die Besitzer „owner“ eines Kontextattributs.

- Das Modellelement „ContextAttribute“ bezieht sich auf ein oder mehrere Kontextinformationen, die von den Sensoren geliefert werden. Diese Kontextinformationen sind im Modellelement „ContextInformation“ beschrieben.

- Ein Kontextattribut ist einer Kontextdimension zugeordnet und bezieht sich auf diese, z.B. einer Zeit- oder einer Ortsdimension.

- Ein Kontextattribut kann einer „AttributeGroup“ zugeordnet sein.

5.4.3 ContextDimension

Generalisierungen

- „GeneralizableElement“

Beschreibung

„ContextDimension“ ist ein generalisierbares Modellelement, welches die möglichen Kontextdimensionen beschreibt, denen ein Kontextattribut zugeordnet werden kann.

Eine Kontextdimension ordnet auf der Modellebene einem Kontextattribut einen Werteraum zu. Damit wird festgelegt, welche konkreten Werte ein Attribut annehmen kann. Hierüber sind die Eigenschaften der Werte, aber auch die möglichen Operationen auf den Attributen festgelegt. Ein Kontextattribut muss genau einer Kontextdimension zugeordnet werden. Eine Kontextdimension kann dagegen einem oder mehreren Kontextattributen zugeordnet werden. Als Beispiel kann die Zeit als eine Kontextdimension festgelegt werden. Diese kann den Attributen „Öffnungszeit“ und „Schließzeit“ einer Entität „Einkaufszentrum“ zugeordnet werden. Unterschiedliche Kontextattribute der gleichen Kontextdimension können zueinander in Beziehung gesetzt bzw. miteinander verglichen werden.

In den meisten Kontextmodellen gibt es keine explizite Unterscheidung zwischen Kontextattributen und zugehörigen Kontextdimensionen. Vielfach werden Entitäten mit Attributen beschrieben, deren Interpretation und die Zuordnung zu einem Wertebereich nicht formal festgelegt sind. Dieses wird bei diesen Ansätzen in die Zuständigkeit der Anwendungslogik verlagert. Eine Ausnahme bildet das Aspect-Scale-Context Modell [155]. Hier erfolgt eine explizite Betrachtung der Kontextdimension. Das Modellelement „Aspect“ entspricht dabei unserer Beschreibung von Kontextattributen. Im Modellelement „Scale“ wird eine Repräsentationsform zu einer Kontextdimension definiert. Das Modellelement „Context“ repräsentiert einen konkreten Kontextwert und ist einer Kontextentität und einer zugehörigen „Scale“ zugeordnet.

Attribute

- defaultRepresentation : RepresentationType

  Die bevorzugte Repräsentationsform dieser Kontextdimension

Beziehungen

- Eine Kontextdimension kann mehreren Kontextattributen zugeordnet werden.

- Eine Kontextdimension besitzt ein oder mehrere Representationsformen.

5.4.4 ContextEntity

Generalisierungen

- „GeneralizableElement“

Beschreibung

Das Modellelement „ContextEntity“ ermöglicht die Modellierung der Kontextentitäten. Instanzen dieses Modellelements können in einer Generalisierungs- oder Spezialisierungsrelation zueinander in Beziehung stehen. Zudem können abstrakte Kontextentitäten definiert werden. Aggregationsbeziehungen können ebenfalls zwischen Kontextentitäten definiert werden.

Attribute

Dieses Modellelement besitzt keine eigenen Attribute.

Beziehungen

- Eine Kontextentität kann kein, ein oder mehrere Kontextattribute besitzen.
- Eine Kontextentität kann einer Kontextrelation zugeordnet sein.

5.4.5 ContextInformation

Generalisierungen

- „ModelElement“

Beschreibung

Das Modellelement „ContextInformation“ repräsentiert die Kontextinformationen, die von einem Sensor zu einem Kontextattribut geliefert werden. Hierüber können die verschiedenen Arten von Kontextinformationen und die möglichen Metainformationen, z.B. der Zeitpunkt der Informationserhebung modelliert werden.

Attribute

- metaInformation : string

  Die zu einer Kontextinformation zugehörige Metainformation

- sensoredInformation : string

  Beschreibung der vom Sensor gelieferten Kontextinformation

Beziehungen

- Eine Kontextinformation wird von einem Sensor geliefert.
- Eine Kontextinformation wird einem Kontextattribut zugeordnet.

5.4.6 ContextRelation

Generalisierungen

- „GeneralizableElement“

Beschreibung

Das Modellelement „ContextRelation“ ermöglicht die Modellierung der Kontextrelationen zwischen Kontextentitäten, z.B. die Beziehung zwischen Personen und Geräten. Eine solche Beziehung kann gerichtet sein. Die beobachtbaren Eigenschaften dieser Beziehungen werden in den zugehörigen Kontextattributen beschrieben. Instanzen dieses Modellelements können in einer Generialisierungs- oder Spezialisierungsrelation zueinander in Beziehung stehen. Zudem können abstrakte Kontextrelationen definiert werden.

Attribute

- isDirected : boolean

  true: Die Beziehung zwischen den Kontextentitäten ist gerichtet.

  false: Die Beziehung zwischen den Kontextentitäten ist ungerichtet.

Beziehungen

- Eine Kontextrelation kann kein, ein oder mehrere Kontextattribute besitzen.
- Einer Kontextrelation können zwei oder mehr Kontextentitäten zugeordnet sein.

5.4.7 ContextRule

Generalisierungen

- „ModelElement“

Beschreibung

Das Modellelement „ContextRule“ ermöglicht die Definition von Regelwissen in einem konkreten Kontextmodell. Mit ihm werden besondere Beziehungen zwischen Kontextattribute beschrieben. Spezialisierte Kontextregeln sind Inferenz- und Integritätsregeln.

Die Verwendung von Regelwissen ist bereits Bestandteil von einigen Ansätzen der Kontextmodellierung. Dazu gehören insbesondere Ontologie-basierte Kontextmodelle. So lassen sich beispielsweise in OWL DL Regeln mit Hilfe der „Description Logic“ beschreiben [156]. Diese basiert auf der Prädikatenlogik. Nachfolgend wird ein gekürztes Beispiel für die Definition einer „Frau“ als „Person“ mit dem Wert „female“ in der Eigenschaft „Geschlecht“ gegeben [157].

```
<owl:Ontology rdf:about=""/>
  <owl:Class rdf:ID="Gender"/>
  <owl:Class rdf:ID="Person"/>
  <owl:Class rdf:ID="Woman">
    <rdfs:subClassOf rdf:resource="#Person"/>
    <owl:equivalentClass>
      <owl:Restriction>
        <owl:onProperty rdf:resource="#Gender"/>
        <owl:hasValue rdf:resource="#female" rdf:type="#Gender"/>
      </owl:Restriction>
    </owl:equivalentClass>
  </owl:Class>

  <owl:ObjectProperty rdf:ID="gender"
   rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
    <rdfs:range rdf:resource="#Gender"/>
    <rdfs:domain rdf:resource="#Person"/>
  </owl:ObjectProperty>
</rdf:RDF>
```

Dieses Regelwissen kann von einer Inferenzmaschine verwendet werden, um auf dieser Basis neues Wissen abzuleiten bzw. die Gültigkeit von Wissen zu validieren.

Eine weitere Möglichkeit zur Beschreibung von Regelwissen insbesondere im Umfeld der UML ist die Verwendung der Object Constraint Language (OCL) [158]. OCL basiert in einer abgespeckten Variante auf dem gemeinsamen Kern von UML und MOF. Dieses ermöglicht die Nutzung dieser Variante für jegliche auf MOF basierende Metamodelle. Die volle Variante der OCL dagegen kann nur gemeinsam mit der UML verwendet werden. Dort dient OCL der textuellen Spezifikation von Invarianten in Klassendiagrammen, von Bedingungen in Sequenzdiagrammen oder der Formulierung von Vor- und Nachbedingungen für Methoden. OCL basiert ebenfalls auf der Prädikatenlogik und ermöglicht das Regelwissen zu Abhängigkeiten und Restriktionen formalisiert zu beschreiben. Die Auswertung eines OCL-Ausdrucks führt zur Ausgabe eines Ergebnisses und kann für die Überprüfung der Konsistenz eines Modells genutzt werden. Im Gegensatz zu OWL kann mit Hilfe von OCL nichts selbst im Modell verändert werden. So erlaubt es OCL beispielsweise nicht den Wert eines Kontextattributs in Abhängigkeit von anderen Attributwerten setzen, um so Regelwissen zur Ableitung neuen Wissens einzusetzen.

Eine weitere häufig gebrauchte Methode in der Kontextmodellierung zur Beschreibung von Regelwissen ist die Formulierung einfacher IF-THEN-ELSE Regeln. Hiermit wird der Zustand eines Kontextattributs in Abhängigkeit sonstiger Kontextattribute beschrieben, z.B. in [159].

Attribute

- rule : string

  Die Beschreibung des Regelwissens. Die konkrete Ausprägung ist abhängig von der jeweils verwendeten Beschreibungssprache.

Beziehungen

- Mit dem Regelwissen werden zwei oder mehr Kontextattribute zueinander in Beziehung gesetzt. Das Modellelement „ContextRule“ verbindet diese miteinander.

5.4.8 GeneralizableElement

Generalisierungen

- „Classifier (from Kernel)“

Beschreibung

Ein „GeneralizableElement“ beschreibt alle Modellelemente deren Instanzen in einer Generalisierungs- oder Spezialisierungshierarchie zueinander in Bezie-

hung stehen können. Zudem darf eine Instanz dieses Modellelements abstrakt sein, d.h. selbst nicht instantiiert werden. Dieses gilt für alle Modellelemente des Metamodells, die von „GeneralizableElement“ abgeleitet sind und somit dessen Eigenschaften erben. Auf der Ebene der Infrastruktur ist dieses beispielsweise das Modellelement „ContextEntity“. „GeneralizableElement“ ist eine abstrakte Metaklasse. Diese Eigenschaft erbt es von der Klasse „Classifier“ im Package „Kernel“ des UML-Metamodells.

Attribute

- description : string

  Beschreibung des Modellelements

Beziehungen

Dieses Modellelement besitzt keine weiteren Beziehungen.

5.4.9 InferenceRule

Generalisierungen

- „ContextRule“

Beschreibung

Das Modellelement „InferenceRule“ ist eine Spezialisierung von „ContextRule“. Es ermöglicht die Definition von Regeln zur Ableitung von zusätzlichen Kontextinformationen. Als Beispiel kann eine Regel definiert werden, dass die Helligkeit draußen nach 20.00 Uhr gering ist. Natürlich ist die Richtigkeit abhängig von der Jahreszeit, dem Breitengrad und sonstigen Faktoren. Aufgrund der potentiellen Unsicherheit solcher Regeln wird ein Wahrscheinlichkeitsfaktor benötigt.

Attribute

- probability : short

  Die Wahrscheinlichkeit für die Gültigkeit dieser Regel

Beziehungen

Dieses Modellelement besitzt keine weiteren Beziehungen.

5.4.10 IntegrityRule

Generalisierungen

- „ContextRule“

Beschreibung

Das Modellelement „IntegrityRule“ ist auch eine Spezialisierung von „ContextRule“. Es ermöglicht die Definition von Regeln zur Validierung der Gültigkeit von Kontextinformationen.

Attribute

Dieses Modellelement besitzt keine eigenen Attribute.

Beziehungen

Dieses Modellelement besitzt keine weiteren Beziehungen.

5.4.11 ModelElement

Generalisierungen

- „NamedElement (from Kernel)“

Beschreibung

Das „ModelElement“ bildet die Basis aller nicht generalisierbaren Modellelemente des Metamodells. Jedes weitere ist von dieser abgeleitet und kann eine Beschreibung besitzen. Sie ist abgeleitet von der Klasse „NamedElement“ aus dem Package „Kernel“ des UML-Metamodells und erbt von dort die Zuordnung eines im Namensraum eindeutigen Namens.

Attribute

- description : string

  Eine textuelle Beschreibung des Modellelements.

Beziehungen

Dieses Modellelement besitzt keine weiteren Beziehungen.

5.4.12 RepresentationType

Generalisierungen

- „ModelElement“

Beschreibung

Das Modellelement „RepresentationType“ ermöglicht die Beschreibung der existierenden Repräsentationsformen einer Kontextdimension, die mehrere Formen der Repräsentation besitzen kann. Beispielsweise ist eine Ortsdimension durch mehrere geographische Koordinatensysteme repräsentierbar, z.B. Gauss-Krüger oder WGS84. Mit einer Repräsentationsform ist zumeist auch die entsprechende Einheit festgelegt.

Repräsentationsformen von Kontextdimension können unterschiedliche Eigenschaften haben, wobei zwischen diskreten und kontinuierlichen Werten unterschieden werden kann. Wertemengen können aufzählbar oder unendlich sein, müssen aber nicht immer eine Ordnung besitzen. Auch in einer geordneten Menge von diskreten Werten können einzelne Werte verschiedenen Abstand zueinander besitzen, weil es möglich ist, unterschiedliche Abstände zu definieren, um beispielsweise Ähnlichkeitsabstufungen zu modellieren. Die notwendige Sicht auf die möglichen Eigenschaften ist abhängig vom jeweiligen Modelltyp und wird daher an dieser Stelle nicht weiter detailliert.

Attribute

- unit : string

  Einheit der Repräsentation einer Kontextinformation in der jeweiligen Kontextdimension.

- isDescrete : boolean

  true - Der Wertebereich der Repräsentationsform der Kontextdimension besteht aus diskreten aufzählbaren Werten.

  false - Der Wertebereich der Repräsentationsform der Kontextdimension besteht aus kontinuierlichen Werten.

- isOrdered : boolean

  true - Der Wertebereich der Repräsentationsform der Kontextdimension besitzt eine Ordnung.

  false - Der Wertebereich der Repräsentationsform der Kontextdimension ist ohne Ordnung.

- implementation : string

  Die Implementierung der Repräsentationsform durch vorgegebene Datentypen, z.B. „integer“. Diese ist abhängig von der jeweiligen Realisierung des Kontextservers.

- hasIntraOperation : boolean

  true – Die Repräsentationsform besitzt eine Überführungsfunktion zu einer alternativen Repräsentationsform.

  false – Die Repräsentationsform besitzt keine Überführungsfunktion zu einer alternativen Repräsentationsform.

Beziehungen

Dieses Modellelement besitzt keine weiteren Beziehungen.

5.4.13 Sensor

Generalisierungen

- „GeneralizableElement“

Beschreibung

Das Modellelement „Sensor“ ermöglicht die Definition von unterschiedlichen Sensortypen als Lieferant von Kontextinformationen. Die zugehörigen Kontextinformationen sind in einer Aggregationsbeziehung zum Sensor definiert. Die genaue Ausprägung dieses Modellelements ist abhängig vom jeweiligen Modelltyp.

Attribute

Dieses Modellelement besitzt keine eigenen Attribute.

Beziehungen

- Ein Sensor liefert ein oder mehrere Kontextinformationen.

5.5 Informelle Kontextbeschreibung

Eine informelle Kontextbeschreibung wird in den Phasen der Planung und der Definition sowie in der Testphase benötigt. Die konkrete Form der Kontextbeschreibung ist in diesen Phasen unterschiedlich. Daher werden diese nachfolgend getrennt betrachtet. In dieser Sicht auf das Kontext-Metamodell sind die dort definierten konkreten Modellelemente implizit in der Kontextbeschreibung enthalten.

5.5.1 Planungsphase

Die Zielstellung der Planungsphase ist die Festlegung der wichtigsten Eigenschaften der häuslichen intelligenten Infrastruktur für die Abschätzung der Machbarkeit und Planung der weiteren Aktivitäten. Die Erstellung eines Lastenhefts ist ein allgemein angewandtes Instrument zur Sammlung und Dokumentation der funktionalen und nicht-funktionalen Anforderungen an eine zu erstellende Software. Dieses soll in dieser Phase ebenfalls eingesetzt werden.

Die wichtigsten Funktionen aus Sicht der Kontextinfrastruktur sind hierbei die Realisierung der strukturellen Kontextadaptivität und die Bereitstellung von Kontextinformationen für die Nutzung durch die AAL-Dienste. Im Lastenheft werden diese mit einem Kennzeichen und einem Titel versehen und kurz beschrieben. Als Erweiterung dieser Struktur soll eine Trennung der Beschreibung von kontextadaptiven und nicht-kontextadaptiven Funktionen erfolgen. Erstere können in der Kennzeichnung durch ein vorangestelltes „K“ kenntlich gemacht werden. Nachfolgend werden zwei Beispiele für einen Eintrag im Lastenheft gegeben.

K.01 Automatische Auswahl eines geeigneten Endgeräts

Funktion: In Abhängigkeit von den in den Nähe des Bewohners vorhandenen Endgeräts wird ein geeignetes ausgewählt und für die Nutzerinteraktion verwendet. Diese Funktion kann den AAL-Diensten als Bestandteil der häuslichen intelligenten Infrastruktur zur Verfügung gestellt werden.

K.02 Bereitstellung von Kontextinformationen zum Bewohner

Funktion: Über entsprechende Sensorik werden Kontextinformationen zum Bewohner der AAL-Diensten zur Verfügung gestellt. Beispiele dafür sind seine Vitalwerte, die Position innerhalb der Wohnung oder seine Tätigkeit.

Wie aus dem Beispiel ersichtlich werden in der textuellen Beschreibung bereits die wesentlichen Inhalte des Kontextmodells festgelegt. Diese erfolgt hier jedoch informell und dient als erster Anhaltspunkt für die weitere Modellierung.

5.5.2 Definitionsphase

Die Zielstellung in dieser Phase ist die weitere Konkretisierung der Anforderungen und Rahmenbedingungen der häuslichen intelligenten Infrastruktur. Mit ihr muss das Produkt soweit eindeutig definiert sein, dass diese Dokumentation als Grundlage für die Abnahme durch einen Kunden verwendet werden kann. Das hier verwendete Instrument ist das Pflichtenheft, welches auf dem Lastenheft aufbaut und es erweitert.

In der Konkretisierung der kontextadaptiven Funktionen der Infrastruktur erfolgt nun eine Aufzählung der dafür benötigten Kontextinformationen. Hier soll bereits zwischen der Kontextentität bzw. Kontextrelation und den zugehörigen Kontextattributen unterschieden werden. Weiterhin muss beschrieben werden, in welcher Qualität und Form diese benötigt werden, was auch in tabellarischer Form angefügt werden kann. Nachfolgend sind die Beispiele aus dem Lastenheft entsprechend zu erweitern.

K.01 Automatische Auswahl eines geeigneten Endgeräts

Funktion: In Abhängigkeit von den in den Nähe des Bewohners vorhandenen Endgeräten wird ein geeignetes ausgewählt und für die Nutzerinteraktion verwendet. Diese Funktion kann den AAL-Diensten als Bestandteil der häuslichen intelligenten Infrastruktur zur Verfügung gestellt werden.

Kontextinformationen:

Bewohner			
	Position	*Die Position des Bewohners innerhalb der Wohnung*	*Auf 1 Meter genau*
Endgerät			
	Position	*Die Position des Endgeräts innerhalb der Wohnung*	*Auf 1 Meter genau*
	Auflösung	*Die Auflösung des Displays*	*Pixel*

K.02 Bereitstellung von Kontextinformationen zum Bewohner

Funktion: Über entsprechende Sensorik werden Kontextinformationen zum Bewohner den AAL-Diensten zur Verfügung gestellt. Beispiele dafür sind seine Vitalwerte, die Position innerhalb der Wohnung oder seine Tätigkeit.

Kontextinformationen:

Bewohner			
	Position	*Die Position des Bewohners innerhalb der Wohnung*	
	Blutdruck	*Der Blutdruck des Bewohners*	*Systolischer Druck in kPa*
	Tätigkeit	*Die Tätigkeit des Bewohners*	*„Schlafen“, „Lesen“, „Kochen“*

Wie aus dem Beispiel ersichtlich, werden nun die Kontextinformationen separat in strukturierter Form beschrieben. In dieser Struktur sind bereits einige Elemente des Metamodells enthalten.

5.5.3 Testphase

Die Zielstellung in dieser Phase ist die Sicherstellung der im Pflichtenheft festgelegten Eigenschaften des Produkts. Auf dessen Grundlage werden daher in der Regel ein Testplan sowie eine Testspezifikation erstellt. Hier sind die Testfälle zu identifiziern und im Rahmen der Spezifikation weiter zu konkretisieren. Dazu gehören die Festlegung der Rahmenbedingungen, die Beschreibung des Ausgangszustands, der durchzuführenden Testschritte sowie des erwarteten Ergebnisses. Testplan, Testspezifikation sowie die Testdurchführung werden nach einer vorgegebenen Struktur dokumentiert.

Das Testen der kontextadaptiven Funktionen der intelligenten häuslichen Infrastruktur folgt demselben Muster. Eine Grundlage für die Spezifikation der funktionalen Tests ist die Beschreibung der Kontextinformationen des Pflichtenhefts. Diese werden sowohl für die Beschreibung des Ausgangszustands als auch für die durchzuführenden Testschritte und des erwarteten Ergebnis benötigt. Nachfolgend wird eine beispielhafte Testspezifikationen erstellt.

T.K.01a Korrekte Auswahl des dem Bewohner am Nächsten befindlichen Endgeräts

Beschreibung: Mit diesem Test soll sichergestellt werden, dass die Auswahl eines Endgeräts in Abhängigkeit der Position zum Bewohner funktioniert.

Ausgangszustand:

Endgerät 1 (LCD-Display)		
	Position	*Haustür*
	Auflösung	*320 x 200 Pixel*
Endgerät 2 (PAL-Fernseher)		
	Position	*Wohnzimmer*
	Auflösung	*704 x 625 Pixel*

Durchzuführende Schritte: Der Bewohner bewegt sich von der Haustür zum Wohnzimmer.

Erwartetes Ergebnis: Zunächst wird Endgerät 1 ausgewählt. Nach Betreten des Wohnzimmers wird Endgerät 2 ausgewählt.

Wie aus dem Beispiel ersichtlich, sind die Kontextinformationen aus dem Pflichtenheft Bestandteil der Testspezifikation. Sie werden ebenfalls in strukturierter Form beschrieben, in der einige Elemente des Metamodells explizit enthalten sind.

5.6 Konzeptuelles Kontextmodell

Auf der Basis des Pflichtenhefts erfolgt die konzeptuelle Umsetzung der Kontextinfrastruktur. Hierzu gehören die in der Softwareentwicklung bekannten Methoden des Softwareentwurfs und der Datenmodellierung. Hinzu kommt nun die Konzeption der Kontextadaptivität durch die Erstellung eines konzeptuellen Kontextmodells. Dabei wird ein von der konkreten Implementierung unabhängiges Kontextmodell definiert. Dieses dient der Erarbeitung und Diskussion der konktextadaptiven Aspekte durch die Entwickler. Die informellen Kontextbeschreibungen der einzelnen Funktionen aus dem Pflichtenheft werden dabei über das konzeptuelle Kontextmodell formalisiert und zusammengefasst.

Anschließend werden zunächst die in der Literatur bekannten Ansätze zur konzeptuellen Kontextmodellierung betrachtet. Hier ist lediglich das erweiterte „Object-Role Model“ von Henricksen, Indulska und Rakotonirainy bekannt. Dieser ist jedoch für die spezifischen Anforderungen aus dem AAL nicht ausgelegt und kann in dieser Form nicht genutzt werden. Danach wird die Nutzung der in UML definierten grafischen Notation für die konzeptuelle Modellierung diskutiert. Auch diese ist nicht unmittelbar geeignet. Daher wird anschlie-

ßend aufbauend auf den betrachteten Konzepten ein erweiterter Ansatz zur konzeptuellen Kontextmodellierung auf der Ebene der Infrastruktur vorgestellt.

5.6.1 Nutzung des erweiterten Object-Role Models

Bereits in Kap. 4.1.4 wurde mit dem erweiterten „Object-Role Model“ die Kontextmodellierung nach Henricksen, Indulska und Raktotonirainy beschrieben. Diese definiert für die einzelnen Modellelemente eine grafische Notation und erlaubt die Beschreibung von Kontextinformationen und deren Beziehung zueinander konzeptuell und implementierungsunabhängig zu beschreiben.

Wesentliche Modellelemente der konzeptuellen Kontextmodellierung nach Henricksen, Indulska und Rakotonirainy entsprechen unserem Metamodell auf Ebene der Infrastruktur. Dazu gehören beispielsweise die Definition von Kontextentitäten und der zugehörigen Kontextattribute. Auch Assoziationen zwischen Kontextentitäten in Form von Kontextrelationen können modelliert werden. Der Ansatz ermöglicht die Beschreibung von Sensoren und deren Eigenschaften und deren Zuordnung zu den Kontextinformationen. Weiterhin können Abhängigkeiten zwischen Assoziationen in Form von Regelwissen beschrieben werden. Jeden dieser Modellelemente ist eine grafische Repräsentation zugeordnet. Diese ermöglicht eine Visualisierung der Bestandteile und Zusammenhänge des Kontextmodells und ist eine Grundlage für die konzeptuelle Kontextmodellierung. Trotz dieser Eigenschaften ist dieser Ansatz in der vorliegenden Ausprägung nicht ausreichend. Die Gründe dafür werden nachfolgend beschrieben.

Die Grundannahme der Kontextmodellierung nach Henricksen, Indulska und Rakotonirainy ist, dass der Aufbau der zugrundeliegenden Kontextinfrastruktur den Entwicklern bekannt bzw. in deren Kontrolle ist. Dieses ist bei aktuellen Entwicklungen kontextadaptiver Anwendungen derzeit durchaus der Fall. Daraus resultiert in der Kontextmodellierung in dem beschriebenen Ansatz die Unterscheidung der Assoziationen in ihrer Art der Erhebung. Im AAL muss jedoch eine Trennung zwischen der Infrastruktur und den Diensten vorgenommen werden (AM8.1, AM9.1). Die konkret verwendete Sensorik ist bei der Konzeption der Infrastruktur noch nicht zu berücksichtigen, da diese je nach Ausstattung einer Wohnung unterschiedlich sein kann. Daher spielt die Art der Erhebung der Kontextinformationen für das konzeptuelle Kontextmodell im AAL keine Rolle und wird nicht in die Modellierung aufgenommen. Weitere für die Konzeption notwendige, aber nicht berücksichtigte Elemente des Kontextmodells sind die Kontextdimensionen und deren Repräsentationsformen. Eine ebenfalls unberücksichtige Anforderung ist die Definition von Attributen zu Relationen.

5.6.2 Nutzung der grafischen Notation von UML

Die Definition unseres Kontext-Metamodells basiert auf der Nutzung von UML (Kap. 4.3.6). Diese definiert für eine Menge von Begriffen eine entsprechende grafische Repräsentation. Nachfolgend soll die Verwendung dieser Notation für die konzeptuelle Kontextmodellierung betrachtet werden.

Aufgrund der bestehenden Problematik der Definition von domänenspezifischen Metamodellen mit Hilfe von UML wurde in Kap. 5.4 die abstrakte Syntax des Metamodells mit Hilfe der Modellelemente auf Ebene M3 definiert. Die konkrete Syntax, d.h. die Umsetzung dieses Metamodells, erfolgt durch die Verwendung der UML-Profile. In einem UML-Profile sind die Anpassungen des UML-Metamodells an eine definierte Domäne zusammengefasst. Eines der Modellelemente in dem Paket „Profiles“ ist die Klasse „Stereotype“. Diese erlaubt die Erweiterung einer bestehenden Klasse des UML-Metamodells im eingeschränkten Rahmen. Stereotypes werden bei der Erstellung der konkreten Syntax verwendet, um das UML-Metamodell um die Modellelemente des Kontext-Metamodells zu erweitern bzw. es dahin anzupassen. Nachfolgend wird ein Beispiel für die Umsetzung des Modellelements „GeneralizableElement“ gegeben.

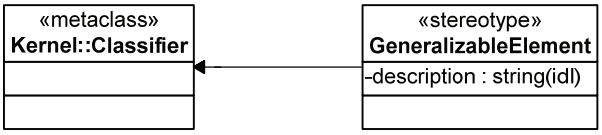


Abbildung 26: Umsetzung des Modellelements „GeneralizableElement“

Auf diese Weise können sämtliche Modellelemente des Kontext-Metamodells mit Hilfe von Stereotypen durch die Spezialisierung der Modellelemente „Kernel::Classifier“ und „Kernel::NamedElement“ umgesetzt werden.

UML sieht für die Repräsentation der mit Hilfe von Stereotypen definierten Sprachelemente die Verwendung der grafischen Notation von Klassen vor. Diese werden durch Rechtecke repräsentiert. Werden diese in der Modellierung eingesetzt, so werden die Namen der Elemente des Metamodells dem Namen des Modellelements in doppelten eckigen Klammern vorangestellt. In der nachfolgenden Abbildung wird ein Beispiel für die Verwendung dieser Notation gegeben.

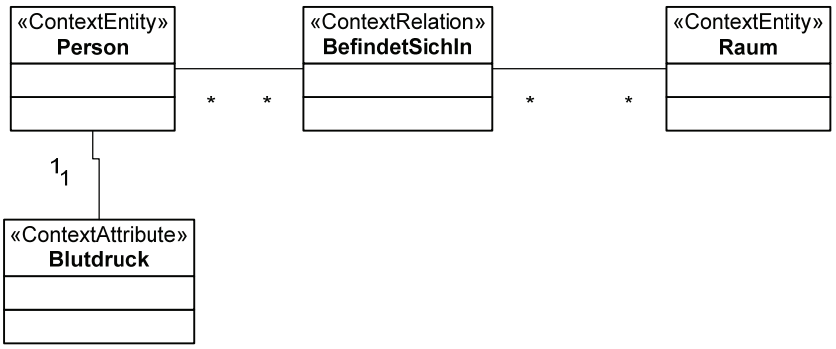


Abbildung 27: Verwendung der grafischen Notation für die Kontextmodellierung

Die Unterscheidung der einzelnen Modellelemente des Kontext-Metamodells ist in dieser Notation in der „Überschrift“ über dem Modellnamen gegeben. Die grafische Repräsentation der unterschiedlichen Modellelemente ist jedoch identisch. Dadurch geht die unterschiedliche Semantik der einzelnen Modellelemente in der grafischen Notation verloren. Das Metamodell von UML-Profiles sieht für diesen Zweck die Klasse „Image“ vor, die die Erstellung einer eigenen Notation zu einem Stereotypen ermöglicht. Aufgrund der Notwendigkeit die Unterschiedlichkeit der Modellelemente des Kontext-Metamodells auch in der Notation zu repräsentieren sollte eine eigene grafische Repräsentation jedes Modellelements realisiert werden. Die Definition einer solchen Notation ist Gegenstand des folgenden Kapitels.

5.6.3 Konzeptuelle Kontextmodellierung für das AAL

In den vorhergehenden Kapiteln wurden zwei mögliche Ansätze für die konzeptuelle Kontextmodellierung beschrieben. Der Ansatz von Henricksen, Indulska und Rakotonirainy erfüllt nicht alle Anforderungen an das Metamodell. Der Vorteil dieses Ansatzes ist jedoch die Repräsentation eines jeden Modellelements durch ein separates Symbol und dadurch die Verdeutlichung der unterschiedlichen Konzepte. Zudem sind dieser Ansatz der Kontextmodellierung und damit die Symbolik der Notation bereits bekannt. Die Verwendung der Mechanismen von UML zur Definition einer angepassten grafischen Notation hat den Vorteil, dass die verfügbaren Werkzeuge auf der MOF Metadaten-Architektur unmittelbar eingesetzt werden können (Kap. 4.3.2). Die vorgegebene Notation zur Repräsentation der mit Hilfe von Stereotypen definierten Modellelemente ist für die konzeptuelle Kontextmodellierung jedoch nicht ausreichend. Daher wird nachfolgend eine nach Möglichkeit in UML verwendbare Notation basierend auf der grafischen Notation des erweiterten „Object-Role Model“ von Henricksen, Indulska und Rakotonirainy definiert. Die Modellelemente, die sich nicht mit Hilfe einer eigenen grafischen Notation für Stereoty-

pen ausdrücken lassen, werden explizit gekennzeichnet. Die Regeln zur Gestaltung einer grafischen Repräsentation von Stereotypen sind in der Definition der UML Superstructure beschrieben ([141], Kap. 18.3.8). Nachfolgend wird die grafische Notation der für die konzeptuelle Kontextmodellierung relevanten Teile des Kontext-Metamodells beschrieben.

Eine Kontextentität wird übereinstimmend mit dem erweiterten „Object-Role Model“ durch ein doppelt gerahmtes Rechteck repräsentiert. Die Eigenschaft einer Kontextentität in einer Generalisierungs-/Spezialisierungsbeziehung zu einer anderen Kontextentität zu stehen wird entsprechend der UML-Notation durch einen nicht ausgefüllten Pfeil dargestellt. Dieses gilt ebenso für die weiteren Modellelemente, die dieselbe Eigenschaft besitzen. In der UML-Notation erfolgt die Darstellung des Namens der Kontextentität unter dem jeweiligen Symbol.

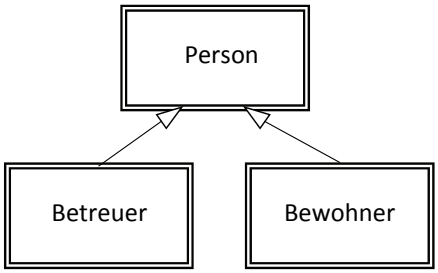


Abbildung 28: Generalisierungs-/Spezialisierungsbeziehung zwischen Kontextentitäten

Abstrakte Kontextentitäten können im erweiterten „Object-Role Model“ nicht dargestellt werden. Im Zusammenhang mit der Generalisierungs- oder Spezialisierungsbeziehung können abstrakte Kontextentitäten genutzt werden, um gemeinsame Eigenschaften von konkreten Kontextentitäten konzeptuell zu bündeln. In UML werden abstrakte Klassen durch die Verwendung einer kursiven Schriftart im Namen gekennzeichnet. Dieses wird in gleicher Form für die Darstellung abstrakter Kontextentitäten bzw. Elemente des Metamodells in der angepassten Notation verwendet.

Abbildung 29: Abstrakte Kontextentität

Eine weitere Eigenschaft von Kontextentitäten, welche im erweiterten „Object-Role Model“ nicht erfasst wird, ist die Definition von Aggregationsbeziehungen. Basierend auf UML wird eine solche Aggregation über einen Pfeil mit einer ausgefüllten Raute an der Spitze ausgedrückt.

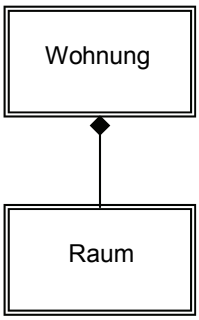


Abbildung 30: Aggregationsbeziehung zwischen Kontextentitäten

Kontextattribute werden im erweiterten „Object-Role Model“ separat von der zugehörigen Kontextentität grafisch durch ein Rechteck repräsentiert. Dieses ermöglicht die Zuordnung weiterer Informationen, beispielsweise von Qualitätsinformationen an jedem Kontextattribut. In UML sind Attribute dagegen Bestandteil einer Klasse und werden in der grafischen Notation innerhalb der Klasse aufgeführt. Für die Kontextmodellierung wird nachfolgend die Notation des erweiterten „Object-Role Model“ übernommen. Somit wird ein Kontextattribut durch ein einfaches Rechteck repräsentiert.

Im Metamodell ist zu einem Kontextattribut die Eigenschaft des Zeitbezugs „hasTimeReference“ definiert. Diese ist bei der konzeptuellen Kontextmodellierung bereits zu beachten. In der Regel besitzt eine Kontextinformation einen aktuellen Zeitbezug. Es kann jedoch vorkommen, dass sich Kontextinformationen auf vergangene oder absehbare zukünftige Zustände von Kontextentitäten oder –relationen beziehen. Um dieses in der Modellierung anzudeuten, soll der Zeitbezug in Klammern hinter dem Namen des Attributs dargestellt werden. Hier kann entweder generell „Zeit“ oder auch konkret „Historie“, „Aktuell“ oder „Zukunft“ angegeben werden. Bei einem aktuellen Zeitbezug kann dieser Zusatz entfallen.

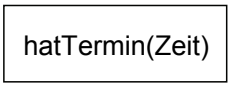


Abbildung 31: Zeitbezug eines Attributs

Die Modellierung des Zeitbezugs nach obiger Notation entspricht nicht den Regeln von UML in Bezug auf Stereotypen. Alternativ dazu kann diese Eigenschaft als Attribut zum Modellelement entsprechend der Klassennotation angegeben werden.

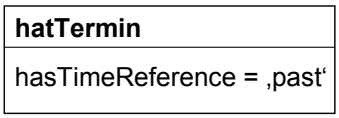


Abbildung 32: Alternative Notation für den Zeitbezug eines Attributs

Die Gruppierung von Attributen kann insbesondere für die konzeptuelle Kontextmodellierung relevant sein, um zusammengehörige Attribute zu visualisieren. Ihre Zusammengehörigkeit soll dadurch visualisiert werden, dass sie durch ein Rechteck umfasst werden, wobei das Rechteck den Gruppennamen trägt.

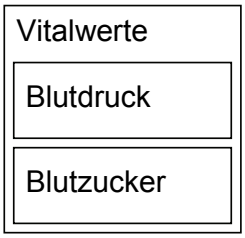


Abbildung 33: Gruppierung von Attributen

Die Modellelemente „ContextDimension“ und „RepresentationType“ sind im erweiterten „Object-Role Model“ nicht vorgesehen. Die verfügbaren Kontextdimensionen und ihre Repräsentationsformen sollen in der konzeptuellen Kontextmodellierung auch nicht definiert werden. Die Zugehörigkeit eines Kontextattributs zu einer Kontextdimension erfolgt durch das Voranstellen der Dimensionsbezeichnung vor dem Namen des Attributs. Kontextattribute derselben Kontextdimension besitzen eine identische Dimensionsbezeichnung vor ihrem Namen. Ist es notwendig, die Repräsentationsform in der konzeptuellen Kontextmodellierung ebenfalls zu berücksichtigen, so kann diese als Parameter in Klammern hinter dem Namen der Kontextdimension angegeben werden.

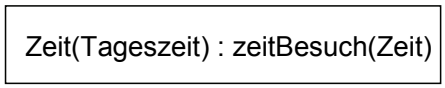


Abbildung 34: Kontextdimension

Die obige Notation entspricht ebenfalls nicht den Regeln für die Darstellung von Stereotypen. Die Modellelemente „ContextDimension“ und „RepresentationType“ spielen für die konzeptuelle Kontextmodellierung eine untergeordnete Rolle. Daher kann hierfür bei Bedarf die Standardnotation für Stereotypen zur Repräsentation dieser Modellelemente verwendet werden. In der nachfolgenden Abbildung wird die Verwendung dieser Notation dargestellt.

Abbildung 35: Alternative Darstellung der Kontextdimension

Eine Kontextrelation wird im erweiterten „Object-Role Model“ durch eine Assoziation zwischen Kontextentitäten beschrieben. Diese ist dort gleichwertig mit einer Assoziation zwischen einer Kontextentität und einem Kontextattribut. Im Kontext-Metamodell besitzt das Konzept „ContextRelation“ jedoch einen höheren Stellenwert und soll daher durch ein eigenes Symbol repräsentiert werden. In Anlehnung an die Modellierung eines Entity-Relationship-Diagramms soll eine „ContextRelation“ durch eine Raute repräsentiert werden.

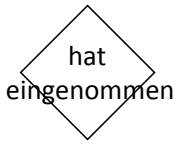


Abbildung 36: Kontextrelation

Die Kontextrelation wird mit den zugehörigen Kontextentitäten durch die UML-Assoziationen mit der entsprechenden Notation verbunden. Eine Linie verbindet die entsprechenden Symbole. Gleichermaßen werden auch die Kontextattribute mit den zugehörigen Kontextentitäten und Kontextrelationen verbunden.

Eine Kontextrelation kann gerichtet sein, d.h. von einer Kontextentität ausgehend in die andere führen. In der Notation soll eine solche Ausrichtung durch einen Pfeil an der Assoziation ausgedrückt werden. Diese zeigt von der ausgehenden Kontextentität in die Kontextrelation und von dort in die zugehörige Kontextentität. Dieses wird in der nachfolgenden Abbildung dargestellt.

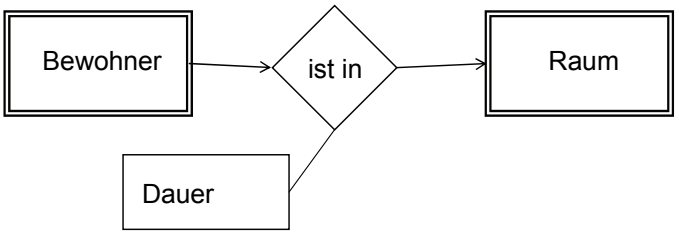


Abbildung 37: Gerichtete Kontextrelation zwischen Kontextentitäten

Eine solche Notation ist ebenfalls nicht mit Hilfe der Stereotypen in UML umsetzbar. In UML beschreibt ein solcher Pfeil das Konzept der Navigierbarkeit zwischen Modellelementen. Dieses ist entspricht semantisch nicht der gerichteten Relation. Eine Alternative besteht darin die ausgehende Kontextentität an der Assoziation als „owner“ zu bezeichnen. Dieses wird in der nachfolgenden Abbildung gezeigt.

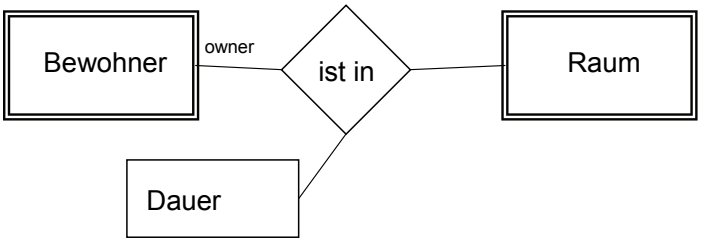


Abbildung 38: Alternative Darstellung einer gerichteten Kontextrelation

Regelwissen zu Abhängigkeiten zwischen Kontextinformationen kann im erweiterten „Object-Role“ Model beschrieben werden. Die voneinander abhängigen Kontextinformationen werden dabei durch einen gestrichelten Pfeil zwischen den einzelnen Assiziationen beschrieben. Das zugehörige Regelwissen wird dabei dem Pfeil als Text ergänzt. Diese Notation soll in abgewandelter Form übernommen werden. Dabei verbindet der gestrichelte Pfeil die zugehörigen Kontextattribute. Das Wissen wird ohne weitere Formalisierung im Klartext beschrieben und mit dem Pfehl verbunden. An dieser Stelle erfolgt noch keine Unterscheidung zwischen Inferenz- und Integritätsregeln. Dieses wird in nachfolgender Abbildung verdeutlicht.

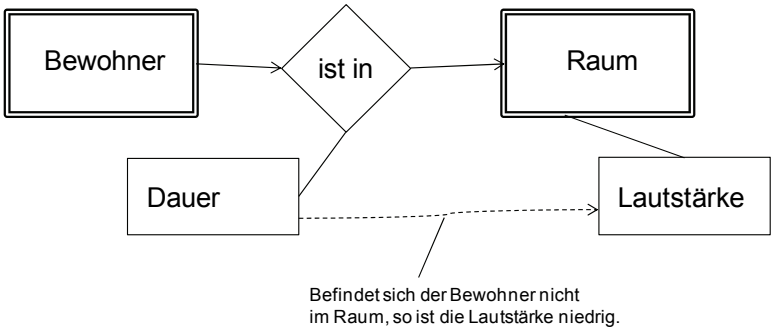


Abbildung 39: Beschreibung von Regelwissen

Diese Notation ist ebenfalls nicht mit Hilfe der UML-Stereotypen umsetzbar. Alternativ dazu wird das Regelwissen in der Standardnotation für Stereotypen beschrieben und mit den Kontextattributen verbunden. Dieses wird in der nachfolgenden Abbildung verdeutlicht.

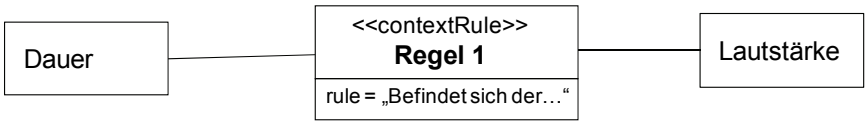


Abbildung 40: Alternative Darstellung des Regelwissens

Diese graphische Notation ermöglicht eine detaillierte Dokumentation und Diskussion des konzeptuellen Kontextmodells. Nachteilig ist sie jedoch bei der Darstellung eines großen Kontextmodells, das aus einer Vielzahl von Entitäten und Relationen mitsamt derer Attribute besteht. Die separate Darstellung der Attribute in einem eigenen Symbol benötigt dabei viel Platz. Aus diesem Grund kann bei Übersichten eine Kurznotation verwendet werden, in der alle Kontextattribute einer Kontextentität bzw. –relation in einem Attributsymbol zusammengefasst werden. Gruppen von Attributen werden durch Einrücken unter einer Gruppenüberschrift dargestellt. In dieser Notation kann ein einzelnes Kontextattribut nicht mehr einzeln adressiert werden. Zudem ist diese nicht mit der Notation für die UML-Stereotypen umsetzbar.

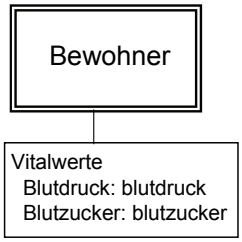


Abbildung 41: Kurznotation des konzeptuellen Kontextmodells

Mit der definierten grafischen Notation lassen sich nun die Anforderungen an das Kontextmodell auf der Ebene der Infrastruktur konzeptuell erfassen und dokumentieren, was eine Grundlage für dessen Realisierung ist.

5.7 Operatives Kontextmodell

Das operative Kontextmodell ist die lauffähige Umsetzung des konzeptuellen Kontextmodells. Zu dessen Realisierung müssen die im konzeptuellen Kontextmodell beschriebenen Modellelemente verfeinert und vervollständigt werden. Diese werden nachfolgend beschrieben. Abschließend wird eine implementationsneutrale Repräsentation des operativen Kontextmodells beschrieben.

5.7.1 Ergänzungen zum konzeptuellen Kontextmodell

Folgende Ergänzungen und Verfeinerungen des konzeptuellen Kontextmodells müssen im operativen Kontextmodell vorgenommen werden.

ContextAttribute: Hier sind die weiteren Eigenschaften des Kontextattributs zu ergänzen. Das sind die Attribute „isIdentifying“, „isSingleValued“, „isComplex“, „isAscending“, „isDescending“, „isRange“ und „isVisible“ des Modellelements „ContextAttribute“.

ContextDimension: Hier ist die Standardrepräsentation einer Kontextdimension festzulegen. Dieses erfolgt im Attribut „defaultRepresentation“.

RepresentationType: Die Einheit eines Repräsentationstyps zu einer Kontextdimension ist zu ergänzen. Zudem müssen die Eigenschaften des Wertebereichs in der Repräsentation festgelegt werden. Dieses erfolgt in den Attributen „isDiscrete“ und „isOrdered“ des Modellelements „RepresentationType“. Zudem muss die Implementierung der Repräsentationsform eines Werts angegeben werden. Dieses erfolgt im Attribut „implementation“. Dieses können die Datentypen sein, mit denen die Werte einer Kontextdimension im Kontextserver implementiert werden, z.B. „integer“.

ContextRule: Die Beschreibung des Regelwissens im konzeptuellen Kontextmodells ist nun auf die beiden Spezialisierungen „InferenceRule“ und „IntegrityRule“ zuzuordnen. Das informell beschriebene Regelwissen ist in eine formale Form zu überführen. Die konkrete Form hängt dabei von der jeweiligen Implementierung der Inferenzmaschine ab. In Kap. 5.4.7 sind einige Möglichkeiten zur Beschreibung von Regelwissen aufgeführt. Die Auswahl eines dieser Beschreibungssprachen führt zu Einschränkungen in der Möglichkeit der Implementierung des Kontextmodells bzw. eines entsprechenden Kontextservers. Ein Ausweg ist hier die Verwendung des Ontology Definition MetaModel (ODM), welches in Kap. 4.3.5 beschrieben wurde. Dieses definiert ein Metamodell für Ontologien auf Basis der MOF, so dass hierfür entsprechend dem Ansatz der „Model Driven Architecture“ (Kap. 4.3.3) zunächst ein implementationsunabhängiges Modell erstellt werden kann, welches in Folgeschritten in die konkrete Implementierung beispielsweise auf Basis von OWL überführt werden kann. Entsprechende Profile sind bereits definiert. Vorteilhaft dabei sind zudem die Nutzbarkeit der verfügbaren MOF-Werkzeuge, beispielsweise die Modelltransformation und die Überführung in eine XML-Darstellung. Zudem ist die Wahrscheinlichkeit für das Ableitungswissen im Attribut „probability“ im Modellelement „InferenceRule“ nachzutragen. Zu einer konkreten Ausprägung dieses Attributs wird auf weitere Arbeiten im Bereich der Handhabung von Unsicherheiten in der Kontexterkennung verwiesen (Kap. 2.2.8).

5.7.2 Implementationsneutrale Repräsentation

Die konkrete Form der Umsetzung der Modellelemente und deren Instanziierung sind abhängig von der zugrundeliegenden Technologie. So kann das Kontextmodell beispielsweise auf der Basis eines objektorientierten Frameworks oder mit Hilfe einer Ontologie-Inferenzengine aufgebaut werden. Erfahrungen haben gezeigt, dass der Einsatz einer Inferenzengine für Kontextmodelle aus Sicht der Performanz nicht immer zu empfehlen ist [160]. Diese deckt sich mit solchen des Fraunhofer ISST [148]. Auch die Abbildung auf ein relationales Datenbankmodell ist eine Möglichkeit der Realisierung. Das zugrundeliegende Entity-Relationship-Paradigma ist jedoch in der semantischen Ausdrucksfähigkeit beschränkt. Das Fraunhofer ISST hat das relationale Modell daher um verschiedene semantische Konstrukte erweitert, z.B. die Generalisierungs- oder Spezialisierungsbeziehung, „instanceOf“, „partOf“ und „geoContains“. Im Rahmen dieser Arbeit wurde ein Kontextserver implementiert, der eine solche Abbildung des Metamodells in das relationale Datenbankmodell realisiert. Er besteht aus einer Reihe von Tabellen und Sichten für das Infrastruktur-Kontextmodell sowie einer Menge von Tabellen für die Verwaltung der zugehörigen Metadaten. Da ein solcher Kontextserver nur eine mögliche Form der Realisierung ist, soll an dieser Stelle nicht weiter darauf eingegangen werden. Gibt es verschiedene Möglichkeiten der Realisierung, so ist es sinnvoll, eine implementierungsneutrale Repräsentation des operativen Kontextmodells zu entwickeln.

Für die implementierungsneutrale Repräsentation von Modellen empfiehlt sich die Verwendung von XML. Auf diese Weise können die Elemente des Metamodells mit Hilfe von XML-Schemata eine Schemadefinition erhalten. Das konzeptuelle Kontextmodell kann dann auf dieser Basis in eine XML-basierte Repräsentation des operativen Kontextmodells überführt werden. Da das Kontext-Metamodell auf UML basiert, bietet es sich hierfür die Verwendung bereits bestehender Werkzeuge an. XMI definiert dabei die Überführung eines MOF-basierenden Metamodells nach XML (Kap. 4.3.2).

Die XMI-konforme Beschreibung des operativen Kontextmodells kann mit Hilfe eines Importmechanismus in den Kontextserver übernommen und in die implementationsspezifische Form überführt werden. Der grundsätzliche Import-Mechanismus ist Bestandteil des im Rahmen der Dissertation implementierten Kontextservers.

5.8 Deskriptives Kontextmodell

Das deskriptive Kontextmodell wird in der Phase der Bereitstellung und Übernahme von Bestandteilen der Infrastruktur in die häusliche intelligente Umgebung benötigt (Kap. 4.2.2). Folgende Bestandteile müssen übernommen werden:

- Sensorik: Sensoren werden in die Infrastruktur eingebunden und zur Ermittlung von Kontextinformationen genutzt.

- Kontextentitäten: Konkrete Entitäten werden der Infrastruktur bekannt gemacht, z.B. wird „Anna Mustermann“ als „Bewohner“ registriert.

Die Beschreibung der Sensorik und der Kontextentitäten einer Infrastruktur basiert ebenfalls auf dem Metamodell auf Ebene der Infrastruktur. Die entsprechenden Modelle dienen dazu konkrete Sensoren und Entitäten zu beschreiben und für die Kontextinfrastruktur verfügbar zu machen. In der MOF Metadaten-Architektur ist eine konkrete Beschreibung auf Ebene M0 zu finden.

5.8.1 Sensorbeschreibung

Mit Hilfe einer Sensorbeschreibung können Sensoren und deren Eigenschaften beschrieben und auf einem Dienstemarkt zur Verfügung gestellt werden. Auf Basis der Eigenschaftsbeschreibung können diese gefunden und in die konkrete häusliche Infrastruktur eingebunden werden. Bestandteil der Kontextinfrastruktur kann eine „Sensor Registry“ für die Einbindung der Sensorik sein. Eine solche Registry ist Bestandteil einer Vielzahl existierender Kontextserver oder Frameworks. Ein Überblick dazu ist in [4] zu finden.

Die Sensorbeschreibung umfasst die Modellelemente „Sensor“ und „ContextAttribute“ des Metamodells. Zudem werden hier noch weitere Modellelemente benötigt, welche das Metamodell erweitern und die spezifischen Eigenschaften der Sensorik beschreiben. Die Betrachtung bestehender Ansätze, sowie die Konzeption einer solchen Sensor Registry und der zugehörigen Sensorbeschreibung ist Bestandteil einer Diplomarbeit [151].

5.8.2 Entitätsbeschreibung

Mit Hilfe einer Entitätsbeschreibung können Kontextentitäten und deren Eigenschaften beschrieben und zur Registrierung in der Kontextinfrastruktur genutzt werden. Bestandteil der Kontextinfrastruktur kann eine „Entity Registry“ (Kap. 8.1) sein, welche die dynamische Registrierung und Einbindung von Kontextentitäten ermöglicht. Nur bei wenigen existierenden Kontextservern oder Frameworks ist die Bereitstellung einer solchen Funktionalität bekannt, z.B. [161].

Teile des operativen Kontextmodells werden für die Festlegung der registrierbaren Kontextentitäten in der „Entity Registry“ benötigt. Diese umfassen die Modellelemente „ContextEntity“ und „ContextAttribute“ des Metamodells. Im Modellelement „ContextAttribute“ werden die spezifischen Eigenschaften der Kontextentität beschrieben, die für die Verwaltung dieser Entitäten innerhalb

des „Entity Server“ benötigt werden. Diese sind abhängig vom jeweiligen Entitätstyp. Für die Beschreibung einer „Person“ können das beispielsweise das „Alter“ oder das „Geschlecht“ sein. Ein Kontextattribut kann bei Bedarf auch auf Ebene der Dienste sichtbar sein und somit in den dienstespezifischen Kontextmodellen genutzt werden. Im Kontext-Metamodell ist hierfür das Attribut „isVisible“ im Modellelement „ContextAttribut“ vorgesehen. Wird dieses lediglich für die Verwaltung im „Entity Server“ benötigt, so wird dieses auf „false“ gesetzt, andernfalls auf „true“.

Der benötigte Ausschnitt aus dem operativen Kontextmodell kann mit Hilfe der XML-Repräsentation auf Basis von XMI beschrieben und für die Umsetzung einer implementationsspezifischen Repräsentation im „Entity Server“ verwendet werden.

Die Beschreibung einer konkreten Entität befindet sich in der MOF Metadaten-Architektur auf Ebene M0. Eine Abbildung der Metamodel-Architektur in eine XML-Repräsentation mit Hilfe der XMI ist auf dieser Ebene nicht vorgesehen. Daher werden nachfolgend alternative Möglichkeiten betrachtet und eine Lösung vorgeschlagen.

Zu den bekannten Kontextservern oder Frameworks sind keine Spezifikationen für eine Entitätsbeschreibung veröffentlicht. Eine ähnliche Funktionalität bieten die Verzeichnisdienste basierend auf dem „Lightweight Directory Access Protocol“ (LDAP) [162]. Verzeichnisdienste werden in der IT-Infrastruktur eines Unternehmens für die Verwaltung und Bereitstellung von Informationen zu Benutzern und Bestandteilen der Infrastruktur verwendet. Grundlage hierfür ist eine hierarchische Datenbank, in denen die Daten gespeichert, verglichen, gesucht, modifiziert und gelöscht werden können. Ein solcher Verzeichnisdienst könnte eine Grundlage für die Realisierung eines „Entity Server“ sein. Das logische Modell von LDAP basiert auf den wesentlichen Modellelementen „Entry“ und „Attribut“. Ein „Entry“ repräsentiert eine Menge von Informationen über ein einzelnes Objekt aus der Realität. Es entspricht der Kontextentität der Kontextmodellierung. Ein „Entry“ besitzt eine Menge von Attributen, welches ein Attributtyp mit einem oder mehren Werten beinhaltet. Somit erfüllt das logische Modell die wesentlichen Anforderungen zur Beschreibung von Kontextentitäten. Ein LDAP-Server ermöglicht den Zugriff auf die vorhandenen Informationen über verschiedene Programmierschnittstellen. Mit der Directory Service Markup Language (DSML) [163] existiert eine Spezifikation für die XML-basierte Kommunikation mit einem LDAP-Server und zur Repräsentation des logischen Modells. In DSML wird ein Ausschnitt aus dem logischen Modell immer im Zusammenhang mit den entsprechenden Methodenaufrufen im LDAP-Server beschrieben. Im nachfolgenden Beispiel wird das Hinzufügen einer Person in die Datenbank dargestellt.

```
<addRequest dn="CN=Anna,OU=HR,DC=Example,DC=COM">
  <attr name="objectclass"><value>top</value></attr>
```

```
  <attr name="objectclass"><value>person</value></attr>
  <attr name="sn"><value>Mustermann</value></attr>
  <attr name="givenName"><value>Anna</value></attr>
  <attr name="age"><value>38</value></attr>
</addRequest>
```

Die Verwendung der DSML ist ein geeigneter Ausgangspunkt für die Beschreibung von Kontextentitäten. Folgende Gründe sprechen jedoch gegen dessen Verwendung in der vorliegenden Form:

- Die Beschreibung einer Kontextentität erfolgt immer im Zusammenhang mit dem Methodenaufruf auf dem LDAP Server.

- Das Konzept der Kontextentität ist nicht explizit in der Schemadefinition der DSML sichtbar. Diese ist im „Distinguished Name“ der „Entry“ bzw. im Attribut „objectclass“ implizit enthalten.

Aufbauend auf DSML wird eine eigene Entitätsbeschreibung festgelegt. In dieser ist kein Methodenaufruf definiert. Zudem werden die Modellelemente des Kontext-Metamodells explizit in der Schemadefinition verwendet. Nachfolgend wird das zugehörige XML-Schema beschrieben.

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema id="entitySchema">
  <xs:element name="contextEntities">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="contextEntity" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="contextAttribute" minOccurs="1" max-
Occurs="unbounded">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="value" minOccurs="1" max-
Occurs="unbounded"/>
                  </xs:sequence>
                  <xs:attribute name="name" type="xs:string"/>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
            <xs:attribute name="name" type="xs:string"/>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
```

```
  </xs:element>
</xs:schema>
```

Der im Kontextmodell definierte Name der Kontextentität und der zugehörigen Kontextattribute sind im Attribut „name“ des XML-Elements „contextEntity“ bzw. „contextAttribute“ anzugeben. Der zugehörige Wert der Kontextattribute zu einer konkreten Instanz einer Kontextentität sind im Inhalt des XML-Elements „value“ zu beschreiben. Im nachfolgenden Beispiel wird eine Person „Anna Mustermann“ mit Hilfe der Entitätsbeschreibung im „Entity Server“ registriert.

```
<contextEntities>
  <contextEntity name=“Person“>
    <contextAttribute name=“sn“>
      <value>Mustermann</value>
    </contextAttribute>
    <contextAttribute name=”givenName”>
      <value>Anna</value>
    </contextAttribute>
    <contextAttribute name=”age”>
      <value>38</value>
    </contextAttribute>
  </contextEntity>
</contextEntities>
```

5.9 Zusammenfassung

In diesem Kapitel wurde zunächst der Einsatzzweck der Kontextmodellierung auf der Ebene der Infrastruktur beschrieben. Darauf aufbauend wurden die Anforderungen identifiziert, die dieser Ebene zuzuordnen sind. Die zur Umsetzung der Anforderungen benötigten Konzepte wurden anschließend erarbeitet. Darauf aufsetzend wurde ein Metamodell definiert, welches die für die Erstellung konkreter Kontextmodelle auf dieser Ebene benötigten Modellelemente festlegt. Die Modellelemente, die zur Beschreibung der Sensoreigenschaften benötigt werden, sind im Rahmen der Dissertation nicht weiter vertieft worden, sondern in eine Diplomarbeit ausgelagert worden. Anhand des Metamodells wurden danach die für die einzelnen Phasen der Entwicklung und Nutzung der Kontextinfrastruktur benötigten Modelltypen definiert und zum Teil anhand von Beispielen verdeutlicht.

6 Kontextmodellierung auf der Ebene der Dienste

Auf der Basis zweidimensionaler Kontextmodellierung wird nachfolgend die Ebene der Dienste im Detail betrachtet und konzipiert. Zunächst wird der Einsatzzweck dieser Ebene beschrieben und die Anforderungen, die ihr zuzuordnen sind, zusammengefasst. Daraufhin werden die notwendigen Konzepte zur Umsetzung der Anforderungen beschrieben sowie das benötigte Metamodell und dessen Abbildung in die Ebene der Infrastruktur definiert. Darauf aufbauend sind abschließend die vier Modelltypen für die jeweiligen Phasen zu beschreiben.

6.1 Einsatzzweck

Diese Ebene dient der Definition und Bereitstellung von dienstetyp-spezifischen Kontextmodellen. Auf der unteren Ebene der Infrastruktur verfügbare Kontextinformationen werden in die dienstetyp-spezifischen Kontextmodelle überführt und bei Bedarf um zusätzliche Semantik angereichert. Sie abstrahiert auch von technischen Details der Kontexterkennung in der Infrastruktur. Auf diese Weise werden Kontextinformationen in einer Form zur Verfügung gestellt, die für die Umsetzung der funktionalen Kontextadaptivität (Kap. 3.3.1) eines AAL-Dienstes und ihre Einbindung in die Kontextinfrastruktur benötigt werden.

6.2 Anforderungen

Im Kapitel 3 wurden bereits unabhängig von der zweidimensionalen Betrachtung der Kontextmodellierung die bestehenden Anforderungen aus Sicht des AAL identifiziert. Jetzt werden auf der Grundlage des Einsatzzwecks die Anwendungsfälle identifiziert, die auf dieser Ebene zuzuordnen sind und dann deren spezifische Anforderungen zusammengefasst.

Folgende Anwendungsfälle lassen sich aus dem Einsatzzweck ableiten:

- Einbinden eines AAL-Dienstes in die Kontextinfrastruktur
- Bereitstellen von dienstetyp-spezifischen Kontextinformationen zur Realisierung der funktionalen Kontextadaptivität eines AAL-Dienstes

Die Einbindung eines AAL-Dienstes in die Kontextinfrastruktur erfordert die Verknüpfung der kontextadaptiven Funktionen des Dienstes mit der Funktionalität der Erkennung und Bereitstellung von Kontextinformationen durch die Inf-

rastruktur. Sie bedingt zunächst einen Abgleich der Anforderungen eines Dienstes an die bereitzustellenden Kontextinformationen mit ihrer Verfügbarkeit in der Kontextinfrastruktur (AM9.1.1), denn der Dienst kann nicht voraussetzen, dass die Informationen in der erforderlichen Form verfügbar sind. Daher müssen diese Anforderungen so beschrieben werden, dass sie für den Abgleich mit der Kontextinfrastruktur genutzt werden können (AM9.2.5).

Die Bereitstellung dienstetyp-spezifischer Kontextinformationen bedingt die Definition von Dienstetypen, wofür spezielle Kontextmodelle erstellt werden können (AM6.1). Wie aus den Anforderungen an die konkrete Ausprägung der einzelnen dienstetyp-spezifischen Kontextmodelle (AK6.1 - AK6.5) ersichtlich ist, haben sie einen unterschiedlichen Fokus und unterscheiden sich daher im Detail. Diese unterschiedlichen Kontextmodelle müssen jedoch aus einer gemeinsamen Menge von Sensoren gespeist werden. Zusätzlich werden Metainformationen benötigt, die Auskunft über die Qualität der verfügbaren Kontextinformationen geben und für den Abgleich mit den Anforderungen aus den Diensten genutzt werden können (AM9.2.5). Weiter sind auf dieser Ebene Kontextinformationen mittels dienstetyp-spezifischer Semantik anzureichern (AM6.1), sowie relevante Situationen zu definieren und zu erkennen, die für die Umsetzung der funktionalen Kontextadaptivität eines Dienstes benötigt werden (AM5.1.1). Für die Definition von relevanten Situationen oder Ereignissen auf dem Kontextmodell wird zudem die Umsetzung von Vergleichsoperatoren benötigt (AM5.1.2).

6.3 Konzepte

Ausgehend von den identifizierten Anforderungen werden nun die auf dieser Ebene relevanten Konzepte vorgestellt. Sie gehen unmittelbar auf das nachfolgend vorgestellte Metamodell, aber auch in die Abbildung auf der Ebene der Infrastruktur über. Folgende Konzepte liegen der Kontextmodellierung auf der Ebene der Dienste zugrunde:

- Dienstetyp-spezifische Kontextmodelle
- Dienstetyp-spezifische Semantik
- Beschreibung der Qualität der Kontextinformationen
- Definition von Situationen
- Vergleichsoperatoren auf Kontextattributen
- Anforderungen eines AAL-Dienstes

Diese werden nachfolgend im Detail beschrieben.

6.3.1 Dienstetyp-spezifische Kontextmodelle

Auf Ebene der Dienste lassen sich unterschiedliche Dienstetypen mit unterschiedlichen Anforderungen an ein konkretes Kontextmodell unterscheiden (siehe Kap. 3.3.2). Daher muss es möglich sein auf Ebene der Dienste unterschiedliche Kontextmodelle für die einzelnen Dienstetypen zu definieren und bereitzustellen. Mit dem Modellelement „ApplicationDomain“ wird die Definition eines Dienstetyps ermöglicht. Durch die Referenz des Kontextmodells zu einer Instanz dieses Modellelements kann dessen Zugehörigkeit zu einem Dienstetyp beschrieben werden.

6.3.2 Dienstetyp-spezifische Semantik

Mit der Anreicherung eines Kontextmodells durch dienstetyp-spezifische Semantik wird dieses um Konzepte erweitert, die auf der Ebene der Infrastruktur nicht verfügbar sind.

Ein einfaches Konzept der semantischen Erweiterung ist die Belegung einer Kontextentität mit einem sprechenden Namen. Auf der Ebene der Infrastruktur werden Kontextentitäten dienstetyp-neutral und entsprechend ihrer Eigenschaften benannt, z.B. „Person“. Auf Ebene des Dienstes kann es notwendig sein, hier von „Patient“, „Bewohner“ oder „Schutzbedürftiger“ zu sprechen. Die Umbenennung einer Kontextentität auf der Ebene der Dienste für jeden Dienstetyp ist daher Bestandteil des Konzepts.

In der Analyse der einzelnen Dienstetypen (Kap. 3.3.2) wurde das Rollenkonzept als notwendig identifiziert. Auf der Ebene der Infrastruktur wird eine solche rollenspezifische Unterscheidung von Entitäten nicht benötigt. In ihr werden Kontextentitäten vielmehr anhand der gemeinsamen Eigenschaften in einer Generalisierungs- oder Spezialisierungshierarchie eingeordnet (Kap. 5.3.3). Auf der Ebene der Dienste kann diese Hierarchie dagegen genutzt werden um Rollenkonzepte auszudrücken. Als Beispiel kann im Dienstetyp „Gesundheit“ (Kap. 3.3.2) eine generelle Kontextentität „Person“ definiert werden. Spezialisierungen dieser Entität können der „Bewohner“ und der „ärztliche Betreuer“ sein. Der Verweis auf eine Generalisierung oder eine Spezialisierung einer Kontextentität kann beispielsweise in der Definition von Kontextrelationen oder Situationen verwendet werden.

Für die unterschiedlichen Spezialisierungen können verschiedene Kontextattribute relevant sein. So müssen nicht alle für eine Kontextentität auf Ebene der Infrastruktur verfügbaren Kontextinformationen auch auf der Ebene der Dienste benötigt werden. Für den Bewohner sind beispielsweise seine Vitalwerte relevante Kontextinformationen, für den ärztlichen Betreuer dagegen lediglich seine Tätigkeit. Eine Auswahl der für eine Kontextentität relevanten Attribute ist daher Bestandteil des Konzepts.

Die obigen Konzepte gelten gleichermaßen für die Definition von Kontextrelationen und -attributen. Diese können ebenfalls umbenannt, spezialisiert sowie selektiert werden.

6.3.3 Beschreibung der Qualität von Kontextinformationen

Die Beschreibung der Qualität der Kontextinformationen wird für den Abgleich der Anforderungen durch die Dienste mit den Fähigkeiten der Infrastruktur benötigt. Hier kann eine Reihe von Qualitätsinformationen identifiziert werden, z.B. die generelle Verfügbarkeit, die Zuverlässigkeit und Genauigkeit der Information oder der Zeitpunkt ihrer Erhebung. Die Qualität eine Kontextinformation wird durch ein Modellelement „QualityMetaInformation“ beschrieben. Es lässt sich zwischen der verfügbaren und der geforderten Qualität von Kontextinformationen unterscheiden. Diese werden in den spezialisierten Modellelementen „QualityFeature“ bzw. „QualityRequirement“ definiert. Die Maßeinheit, mit der die Qualität angegeben wird, ist abhängig von der jeweiligen Kontextdimension, z.B. Ortungsgenauigkeit in „m“.

6.3.4 Anforderungen eines AAL-Dienstes

Zur Umsetzung des funktionalen kontextadaptiven Verhaltens eines AAL-Dienstes benötigt dieses Teile des dienstetyp-spezifischen Kontextmodells. Die Anforderungen an den Ausschnitt aus Sicht des AAL-Dienstes müssen im Kontextmodell beschrieben werden. Dazu gehören der Zugriff auf den Kontextraum, sowie die Verfügbarkeit von Kontextinformationen in der benötigten Qualität. Zu diesem Zweck wird ein Modellelement „AALService“ benötigt, welches die Abhängigkeit eines AAL-Dienstes von den zugehörigen Kontextinformationen bzw. Kontextraum beschreibt. Das Modellelement „QualityRequirement“ beschreibt die spezifischen Qualitätsanforderungen zu einer Kontextinformation.

Auf Ebene M2 der Metadaten-Architektur beschreibt das Modellelement „AALService“ die generelle Abhängigkeit eines AAL-Dienstes vom Kontextmodell. Dieses Modellelement wird bei der Instanziierung des Metamodells auf Ebene M1 übernommen. Dabei beschreibt „AALService“ die mögliche Abhängigkeit von den unterschiedlichen verfügbaren Kontexträumen und Kontextinformationen. Die konkrete Abhängigkeit eines AAL-Dienstes wird auf Ebene M0 beschrieben. Dieses erfolgt im deskriptiven Kontextmodell in der Dienstebeschreibung (Kap. 6.9.1).

Aus Sicht des Kontextmodells auf Ebene der Infrastruktur ist ein AAL-Dienst eine bestehende Software innerhalb der intelligenten häuslichen Umgebung. Die Information zur Verfügbarkeit eines AAL-Dienstes kann zur Realisierung des strukturellen kontextadaptiven Verhaltens genutzt werden. Das Mo-

dellelement „AALService“ entspricht daher auf Ebene M2 der Metadaten-Architektur einer „ContextEntity“ der Infrastruktur. Soll die Beschreibung eines AAL-Dienstes in der Infrastruktur aufgenommen werden, so kann dieses durch Verknüpfung dieser Modellelemente auf Ebene M1 erfolgen.

6.3.5 Definition von Situationen

Nach Dey [57] beschreiben Kontextinformationen die Situation einer Entität. Eine Situation ist charakterisiert durch typische Zustände von Kontextinformationen. Beispielsweise kann eine Situation „Personen befinden sich in einer Besprechung“ durch die Anzahl der Personen in einem Raum, die Tageszeit und die Art des Raumes beschrieben werden. In einer Situation kann also zwischen diese charakterisierenden, sowie irrelevanten Kontextinformationen unterschieden werden. So kann in diesem Beispiel die Temperatur innerhalb des Raums ohne Bedeutung für diese Situation sein. In einer formalen Definition einer Situation müssen also die diese charakterisierenden Kontextinformationen beschrieben werden. Diese beinhalten die relevanten Kontextattribute und deren Ausprägung, sowie die zugehörigen Kontextentitäten und Kontextrelationen. Formal soll ein „ContextSpace“ bzw. Kontextraum CS als Quadrupel wie folgt beschrieben werden.

CS = (CE, CR, CA, CAC)

CE ist die Menge aller Kontextentitäten, die für die Beschreibung des Kontextraums relevant sind bzw. in ihr enthalten sind. CE ist dabei eine Untermenge der Kontextentitäten, die im Kontextmodell definiert sind.

CR ist die Menge aller Kontextrelationen, die für die Beschreibung des Kontextraums relevant sind bzw. in ihr enthalten sind. CR ist dabei eine Untermenge der Kontextrelationen, die im Kontextmodell definiert sind.

CA ist die Menge aller Kontextattribute, die für die Beschreibung einer Kontextentität bzw. einer Kontextrelation in einem Kontextraum relevant sind. CA ist dabei eine Untermenge der Kontextattribute, die im Kontextmodell definiert sind.

CAC ist die Menge der Bedingungen, die auf den Kontextattributen in CA definiert sind. Erfüllt eine Instanz des Kontextmodells alle Bedingungen in CAC, so ist diese im Kontextraum enthalten. Zur Definition der Bedingungen können Vergleichsoperationen auf den Kontextattributen definiert werden.

Ein Kontextraum definiert einen Ausschnitt auf dem Kontextmodell und repräsentiert die dadurch charakterisierte Situation. Für die Definition eines solchen Raums wird das Modellelement „ContextSpace“ verwendet. Es besitzt auch einen Namen, welcher die entsprechende Situation bezeichnet, und kann

durch eine Menge alternativer Kontexträume gekennzeichnet werden, die ebenfalls modelliert sowie dieser zugeordnet werden müssen. Eine Instanz einer Kontextentität befindet sich innerhalb eines Kontextraums, wenn die Werte seiner Kontextattribute sowie der verknüpften Kontextrelationen in dem durch die Bedingungen definierten Wertebereich liegen.

In der folgenden Abbildung wird beispielhaft die Definition einer Situation „Emergency“ über einen Kontextraum verdeutlicht. Diese Situation soll vereinfacht dadurch gekennzeichnet sein, dass sich die Person innerhalb einer Wohnung befindet und dort über einen längeren Zeitraum keine Aktivität zeigt. Somit ist der Kontextraum durch Bedingungen an den Kontextattributen „time“, „activity“ und „position“ der Kontextentität „Person“ definiert. Weitere im Kontextmodell bekannte Kontextentitäten oder sonstige Kontextattribute der Kontextentität „Person“ spielen für die Beschreibung dieser Situation keine Rolle, ebenso wie die Relationen zwischen Kontextentitäten. In einem Beispiel gelten für eine Instanz mit der ID 3 der Kontextentität „Person“ folgende Werte der Kontextattribute: „time“ > „5 min“, „activity“ = „keine Aktivität“ und „position“ = „innerhalb der Wohnung“. Somit befindet sich diese Instanz innerhalb des Kontextraums. Die Instanzen mit der ID 1 und 2 befinden sich entweder außerhalb der Wohnung oder zeigen Aktivität und befinden sich somit nicht innerhalb des Kontextraums.

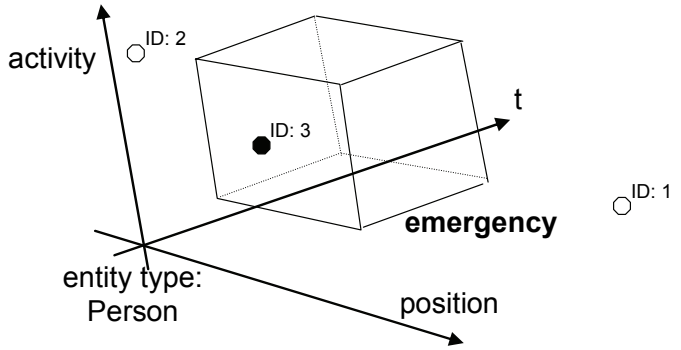


Abbildung 42: Kontextraum

Das Konstrukt eines Kontextraums ist in den meisten publizierten Ansätzen zur Kontextmodellierung nicht explizit gegeben. Allerdings können in verschiedenen ontologiebasierten Modellen Inferenzregeln dafür verwendet werden, um hochwertige Kontextinformationen bzw. Situationen abzuleiten. Solche Regeln können beispielsweise in F-Logic [124] oder OWL [95] beschrieben werden.

6.3.6 Vergleichsoperatoren auf Kontextattributen

Zur Definition der Einschränkungen auf den Kontextattributen innerhalb eines Kontextraums sowie für Abfragen auf dem Kontextmodell werden Vergleichsoperationen auf einzelnen Kontextattributen benötigt. Ihre Umsetzung muss durch das Kontextmodell unterstützt werden. Die nachfolgend beschriebenen Vergleichsoperationen können identifiziert werden deren Umsetzung von den Eigenschaften der Werte in den Kontextattributen bzw. entsprechenden Kontextdimensionen abhängt.

- Gleichheit: Der Wert eines Kontextattributs entspricht einem Vergleichswert. Ein Beispiel hierfür ist die Einschränkung des Kontextraums für die Entität „Person“ mit dem Kontextattribut „Ort“ = „Schulungsraum“. Diese Vergleichsoperation sollte uneingeschränkt für jegliche Kontextdimensionen möglich sein.

- größer oder kleiner: Der Wert eines Kontextattributs ist größer oder kleiner als ein angegebener Vergleichswert. Ein Beispiel hierfür ist die Einschränkung des Kontextraums für die Entität „Person“ mit dem Kontextattribut „Blutdruck“ > „140“. Die Verfügbarkeit einer solchen Vergleichsoperation hängt davon ab, ob die Werte innerhalb der Kontextdimension eine Ordnung besitzen.

- Entfernung: Der Wert eines Kontextattributs ist in der Nähe eines angegebenen Vergleichswerts. Hierfür soll als Voraussetzung eine Kontextdimension gelten, für deren Werte eine Distanzfunktion zur Berechnung von Entfernungen zwischen Einzelwerten vorhanden ist, zum Beispiel die Einschränkung des Kontextraums für die Entität „Raum“ ist in der Nähe von (x,y). Ein Ortsmodell, das auf einem geographischen Koordinatensystem basiert, kann als Distanzfunktion zum Beispiel die euklidische Metrik verwenden, während in einem symbolischen Ortsmodell Entfernungen zwischen benachbarten Orten explizit in einer zugehörigen Datenstruktur angegeben werden [164].

- Punkt ist enthalten in Raum: Der Wert dieses Kontextattributs ist in einem Wertebereich enthalten wie die Einschränkung des Kontextraums für die Entität „Person“ mit dem Kontextattribut „position“ gegeben über einen Punkt P(x,y), enthalten in einem Zimmer, oder mit dem Kontextattribut „position“ gegeben über einen Raum, definiert über zwei diagonale Eckpunkte R(x1,y1,x2,y2). Die Voraussetzung hierfür ist also die Beschreibung, ob es sich bei dem Attribut um einen Punkt oder einen Raum handelt. Mit dieser Unterscheidung können noch weitere Vergleichsoperatoren definiert werden, z.B. Raum ist enthalten in Raum, Raum schneidet Raum, Raum umfasst Raum, Raum ist entfernt von Raum.

Zur Umsetzung dieser Vergleichsoperatoren muss das Kontextmodell deshalb Informationen darüber enthalten, ob ein Kontextattribut eine Ordnung besitzt und es sich hierbei um einen Punkt oder einen Raum in der entsprechenden Kontextdimension handelt. Solche Informationen enthält das Metamodell.

Die physische Repräsentation von Kontextattributen erfolgt in der Umsetzung des Kontextmodells über verschiedene Datentypen wie String oder Integer. Die Vergleichsoperatoren auf dem Kontextmodell lassen sich über die in den Datentypen definierten Operationen realisieren. Auf ihrer Ebene sind so beliebige Kontextattribute des gleichen Datentyps miteinander vergleichbar, z.B. die Temperatur mit der Helligkeit. Semantisch macht jedoch ein solcher Vergleich keinen Sinn und sollte durch das Kontextmodell nicht zugelassen werden. Zu diesem Zweck definiert das Modellelement „ContextDimension“ die semantisch zusammengehörenden Kontextattribute, sodass nur die Kontextattribute innerhalb derselben Kontextdimension miteinander verglichen werden können.

6.4 Abbildungen von Modellelementen auf die Ebene der Infrastruktur

In Kap. 4.2.1 wird die Trennung zwischen den drei unterschiedlichen Ebenen „Infrastruktur“, „Dienst“ und „Nutzerinteraktion“ in der Dimension „Einsatzzweck“ als wesentliches Konzept der Kontextmodellierung beschrieben. Auf Ebene der Infrastruktur werden die intelligente häusliche Umgebung und die durch die Sensoren bereitgestellten Kontextinformationen beschrieben. Auf Ebene der Dienste werden die jeweils dienstetyp-spezifischen Kontextmodelle definiert. Die auf Ebene der Infrastruktur vorhandenen Kontextinformationen müssen in die Ebene der Dienste überführt werden. Die entsprechenden Möglichkeiten der Abbildung werden nachfolgend auf Basis der Modellelemente des Metamodells beschrieben. Generell gilt, dass bei einer Übernahme von Kontextattributen in die Ebene der Dienste die Informationen zu deren Eigenschaften (z.B. isIdentifying) sowie der zugehörigen Repräsentationsformen (z.B. isDiscrete) automatisch in das Kontextmodell übernommen werden.

- **Selektion einer Kontextentität (F1)**: Eine Kontextentität aus dem Infrastruktur-Kontextmodell kann ausgewählt und in das Kontextmodell auf der Ebene der Dienste übernommen werden. Dazu können auch Kontextattribute gehören. Sowohl die Kontextentität als auch die übernommenen Kontextattribute sind dabei umbenennbar. Alle identifizierenden Kontextattribute werden automatisch übernommen.

  Beispiel: Auf der Ebene der Infrastruktur ist eine Kontextentität „Person“ definiert. Zu dieser Entität sind unter anderem die Kontextattribute „Position“ und „Körpergewicht“ definiert. In einem dienstetyp-spezifischen Kontextmodell kann nun die Kontextentität „Patient“ definiert und auf die zugrundeliegende Kontextentität „Person“ abgebildet

werden. In der Definition wird als nicht-identifizierendes Kontextattribut lediglich „Körpergewicht“ übernommen.

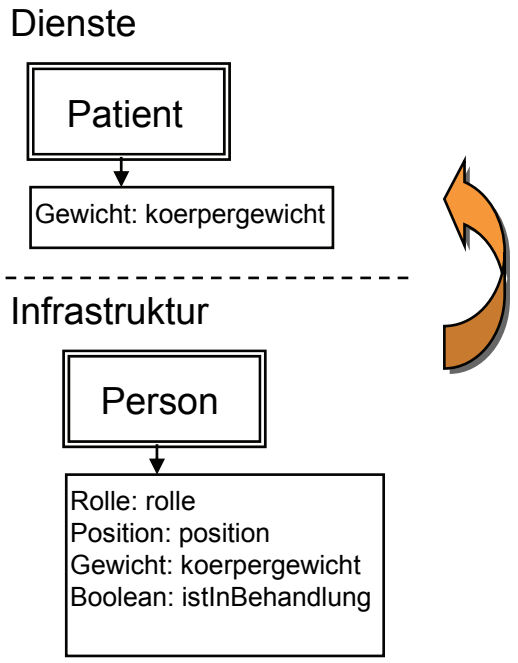


Abbildung 43: Selektion einer Kontextentität

- **Spezialisierende Selektion einer Kontextentität (F2)**: Diese Überführungsfunktion entspricht der Selektionsfunktion F1. Zusätzlich können Bedingungen auf den zugehörigen Kontextattributen definiert werden, die für diese Spezialisierung der Kontextentität gelten müssen. Je nach der Änderungshäufigkeit des ausgewählten Attributs kann die Zuordnung einer konkreten Instanz der Kontextentität zu einer Spezialisierung permanent oder temporär sein.

  Beispiel: In dem vorhergehenden Beispiel ist die Zuordnung einer Person zur Entität „Patient“ nur für die Instanzen gegeben, deren Attribut „istInBehandlung“ gleich „true“ ist.

- **Selektion einer Kontextrelation (F3)**: Eine Kontextrelation aus dem Infrastruktur-Kontextmodell kann ausgewählt und in das Kontextmodell auf der Ebene der Dienste übernommen werden, wenn die zugehörigen Kontextentitäten bereits selektiert wurden (F1 bzw. F2). Mit der Übernahme kann auch jene der zugehörigen Kontextattribute erfolgen. Sowohl die Kontextrelation als auch die übernommenen Kontextattribute können dabei umbenannt werden.

Beispiel: Auf der Ebene der Infrastruktur ist eine Kontextrelation „(Person) ist in Beziehung zu (Person)“ definiert. Zu dieser Relation bestehen die Kontextattribute „istWahr“ und „artBeziehung“. In einem Dienste-Kontextmodell wurden die Kontextentitäten „Betreuer“ und „Patient“ mittels spezialisierender Selektion (F2) übernommen. In einem Dienste-Kontextmodell kann nun die Kontextrelation „(Betreuer) betreut (Patient)“ definiert und auf die zugrundeliegende Kontextrelation abgebildet werden. Dabei wird das Kontextattribut „istWahr“ übernommen.

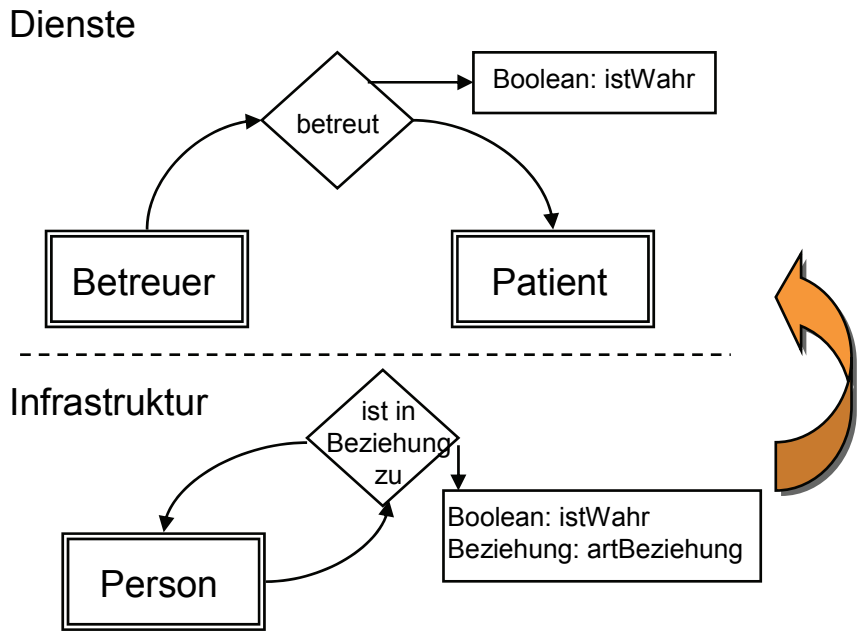


Abbildung 44: Selektion einer Kontextrelation

- **Spezialisierende Selektion einer Kontextrelation (F4)**: Diese Überführungsfunktion entspricht der Selektionsfunktion F3. Zusätzlich können Bedingungen auf den zugehörigen Kontextattributen definiert werden, die für diese Spezialisierung der Kontextrelation gelten müssen. Je nach der Dynamik des ausgewählten Attributs kann die Zuordnung der konkreten Instanz einer Kontextrelation zur Spezialisierung permanent oder temporär sein.

  Beispiel: Im vorhergehenden Beispiel ist die definierte Beziehung nur gegeben, wenn das Attribut „artBeziehung“ der Relation gleich „betreuend“ ist.

- **Ableitung einer Kontextrelation (F5)**: Es kann eine neue Kontextrelation im Dienste-Kontextmodell definiert werden, die aus dem Kontextattribut von zwei Kontextentitäten im Infrastruktur-Kontextmodell

abgeleitet wird. Die beiden Attribute müssen derselben Kontextdimension zugeordnet und daher miteinander vergleichbar sein. Die abgeleitete Kontextrelation kann dann über den Vergleichsoperator definiert werden. Das zugehörige Kontextattribut trägt das Ergebnis der Vergleichsoperation.

Beispiel: Auf der Ebene der Infrastruktur sind die zwei Kontextentitäten „Person“ und „Ort“ definiert. Beide besitzen ein Attribut „position“, welches der Kontextdimension „Position“ zugeordnet ist. Beide Entitäten wurden in das Dienste-Kontextmodell übernommen. Eine neue Kontextrelation „(Bewohner) befindet sich in (Raum)“ wird definiert. Das Kontextattribut „istWahr“ ergibt sich aus einer „istIn“-Vergleichsoperation (siehe „Punkt ist enthalten in Raum“, Kap. 6.3.6) zwischen dem Kontextattribut „position“ der Kontextentität „Person“ mit dem Kontextattribut „position“ der Kontextentität „Raum“.

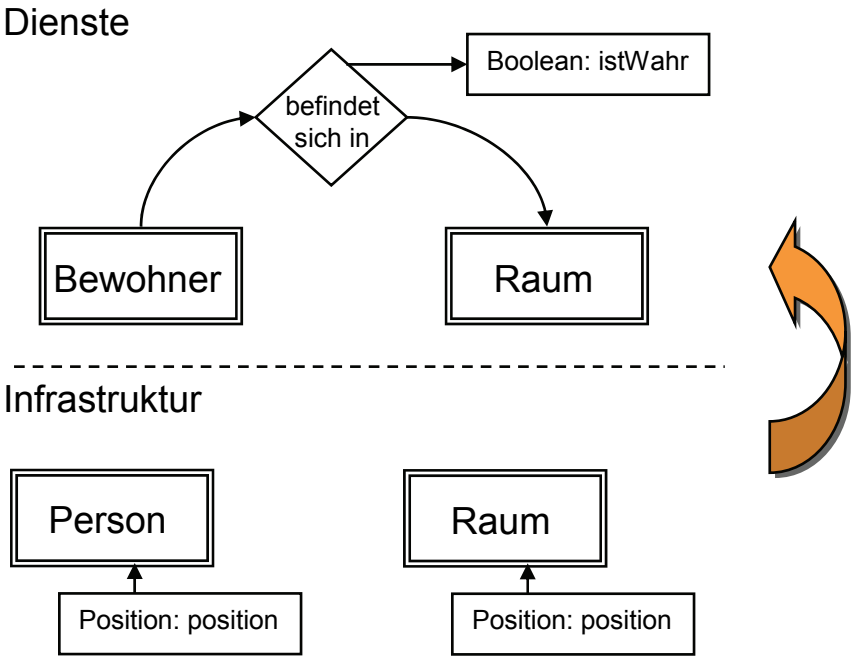


Abbildung 45: Ableitung einer Kontextrelation

- **Ableitung eines Kontextattributs (F6)**: Zu einer im Dienste-Kontextmodell bestehenden Kontextentität kann ein neues Kontextattribut definiert werden, wenn zur entsprechenden Kontextentität auf der Ebene der Infrastruktur eine Kontextrelation definiert ist. Ein Kontextattribut der dieser Kontextrelation oder einer darüber verbundenen Kontextentität kann in das neue Kontextattribut auf der Ebene der Dienste übernommen werden. Je nach Eindeutigkeit der Kontextrelati-

on ist das neue Kontextattribut ein- oder mehrwertig. Diese Möglichkeit der Ableitung folgt aus der grundsätzlichen Freiheit der Kontextmodellierung eine Beziehung zwischen Kontextentitäten explizit als Kontextrelation zu modellieren oder implizit als Eigenschaft einer Kontextentität zuzuordnen.

Beispiel: Auf der Ebene der Infrastruktur sind die Kontextentitäten „Person“ und „Ort“ modelliert worden. Zwischen beiden Entitäten existiert eine Kontextrelation „(Person) befindet sich an (Ort)“. Beide Kontextentitäten werden in das dienstetyp-spezifische Kontextmodell übernommen. Nun wird in der Kontextentität „Person“ das Kontextattribut „ort“ hinzugefügt. Dieses ist über die entsprechende Kontextrelation an das Kontextattribut „name“ der Kontextentität „Ort“ gebunden.

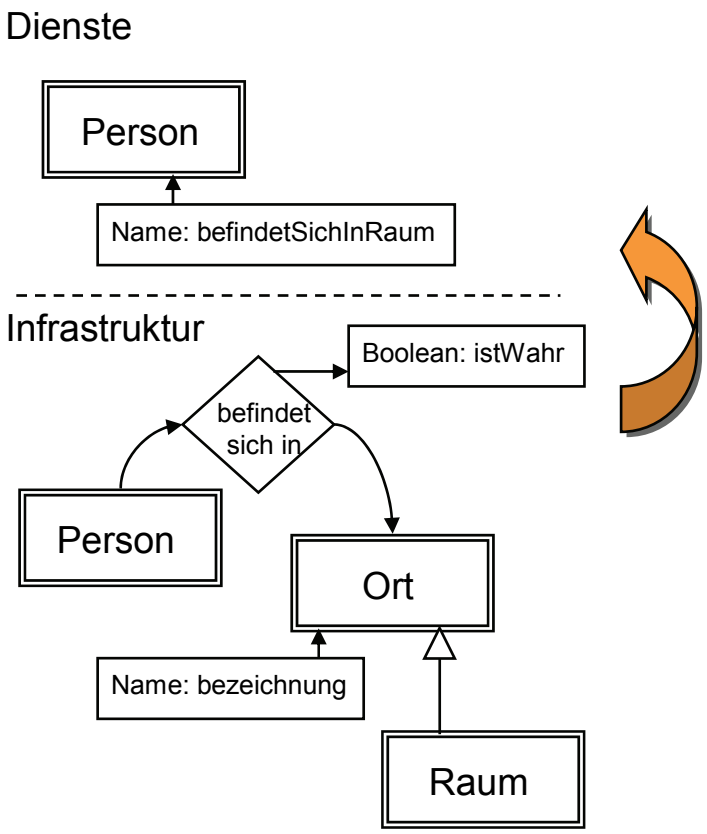


Abbildung 46: Ableitung eines Kontextattributs

- **Definition einer generellen Kontextentität (F7)**: Auf der Ebene des Dienste-Kontextmodells kann eine Reihe bereits definierter Kontextentitäten zu einer generellen zusammengefasst werden. Besitzen alle

spezialisierten Kontextentitäten dieselben -attribute, so können diese in einer generellen Kontextentität zusammengefasst werden.

Beispiel: Auf der Ebene des Dienste-Kontextmodells wurden der „Arzt“, der „Pfleger“ und die „Aushilfe“ als spezialisierende Entitäten aus dem Infrastruktur-Kontextmodell übernommen (F2). Diese werden durch die generelle Kontextentität „Personal“ zusammengefasst.

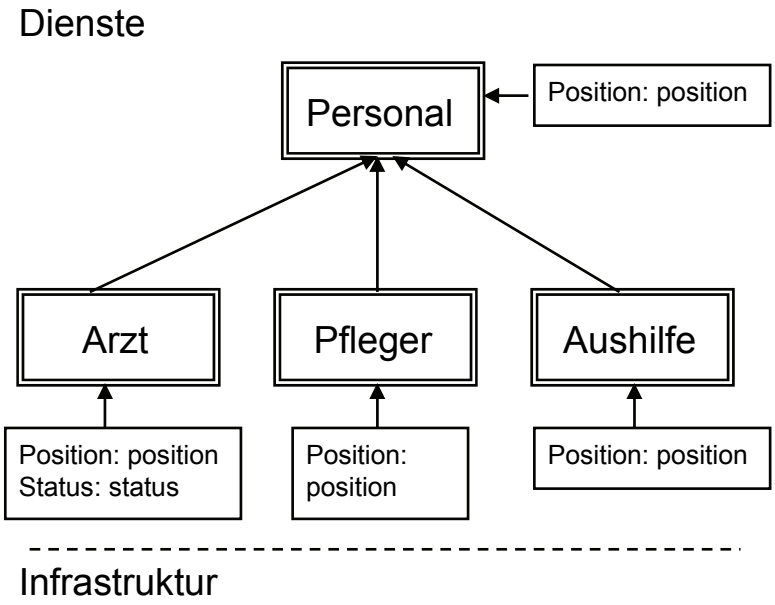


Abbildung 47: Definition einer generellen Kontextentität

- **Auswahl der Dimensionsrepräsentation (F8)**: Sind unterschiedliche Repräsentationsformen einer Kontextdimension definiert und existieren Überführungsfunktionen zwischen diesen, so kann ein Kontextattribut beim Übergang von der Ebene der Infrastruktur in die Ebene der Dienste in eine alternative Repräsentationsform überführt werden.

  Beispiel: Ein Temperatursensor liefert die Temperatur in Grad Fahrenheit. Auf der Ebene der Dienste wird diese Kontextinformation in Grad Fahrenheit benötigt.

- **Auswahl spezialisierter Kontextdimension (F9)**: Ist eine mehr spezialisierte Kontextdimension definiert, so kann ein Kontextattribut beim Übergang in die Ebene der Dienste aus der generellen in die spezialisierte Kontextdimension überführt werden.

Beispiel: Auf der Ebene der Infrastruktur sind die Tätigkeiten einer Person als Kontextdimension definiert, welche die für die Erstellung konkreter Kontextmodelle auf dieser Ebene benötigten Modellelemente festlegt. Ein Kontextsensor erfasst alle Tätigkeiten der Person. Auf der Ebene der Dienste werden nun zu der Person nur die pflegerischen Tätigkeiten als Kontextinformationen benötigt.

6.5 Metamodell

Nachfolgend werden die Elemente des Metamodells auf Ebene der Dienste beschrieben. Dieses Metamodell ist die Grundlage für alle Modelltypen bzw. Sichten auf dieser Ebene. Je nach Modelltyp besitzen die entsprechenden Sichten einen mehr oder weniger eingeschränkten Umfang des Metamodells. Die Definition des Metamodells erfolgt nachfolgend durch Erweiterung des UML-Metamodells in einem Paket „Context.Application“. Für die Definition grundlegender Eigenschaften von Modellelementen des Kontext-Metamodells wird auf die grundlegenden Modellierungskonzepte von UML zugegriffen, die dort im Paket „Kernel“ definiert sind. Für die Beschreibung der Abbildung des Kontextmodells auf Ebene der Dienste auf das Kontextmodell auf Ebene der Infrastruktur wird zudem eine Referenzierung auf das Paket „Context.Infrastructure“ benötigt. Dieses wird in der nachfolgenden Abbildung dargestellt.

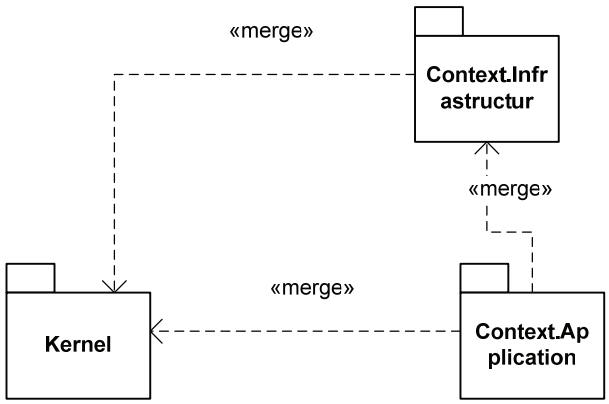


Abbildung 48: Aufteilung des Metamodells in Pakete

In dem Paket „Context.Application“ werden die Elemente des Metamodells definiert, die auf der Ebene der Dienste für die Kontextmodellierung benötigt werden und im Wesentlichen identisch sind mit denen auf der Ebene der Infrastruktur: „Context-Entity“, „ContextRelation“, „ContextAttribute“, „ContextDimension“ und „RepresentationType“. Auf Ebene der Infrastruktur werden diese verwendet um die intelligente häusliche Umgebung inklusive der vorhandenen Sensorik zu beschreiben. Auf Ebene der Dienste werden diese Modellelemen-

te benötigt, um die dienstetyp-spezifische Sicht auf die benötigten Kontextinformationen zu beschreiben. Die weiteren auf dieser Ebene spezifischen Modellelemente wurden bereits in der Konzeption (Kap. 6.3) beschrieben: „ApplicationDomain“, „QualityMetaInformation“ und „ContextSpace“. Zudem sind die Modellelemente vorhanden, mit denen die Abbildung des Kontextmodells auf die Ebene der Infrastruktur beschrieben werden kann (Kap. 6.4).

Die identifizierten Modellelemente und deren Beziehungen zueinander werden in den nachfolgenden Klassendiagrammen gezeigt und anschließend detailliert in alphabetischer Reihenfolge beschrieben. Zur besseren Übersicht werden zunächst die Modellelemente zur Definition des Kontextmodells dargestellt. Danach werden die Modellelemente für die Abbildung aus der Ebene der Infrastruktur dargestellt.

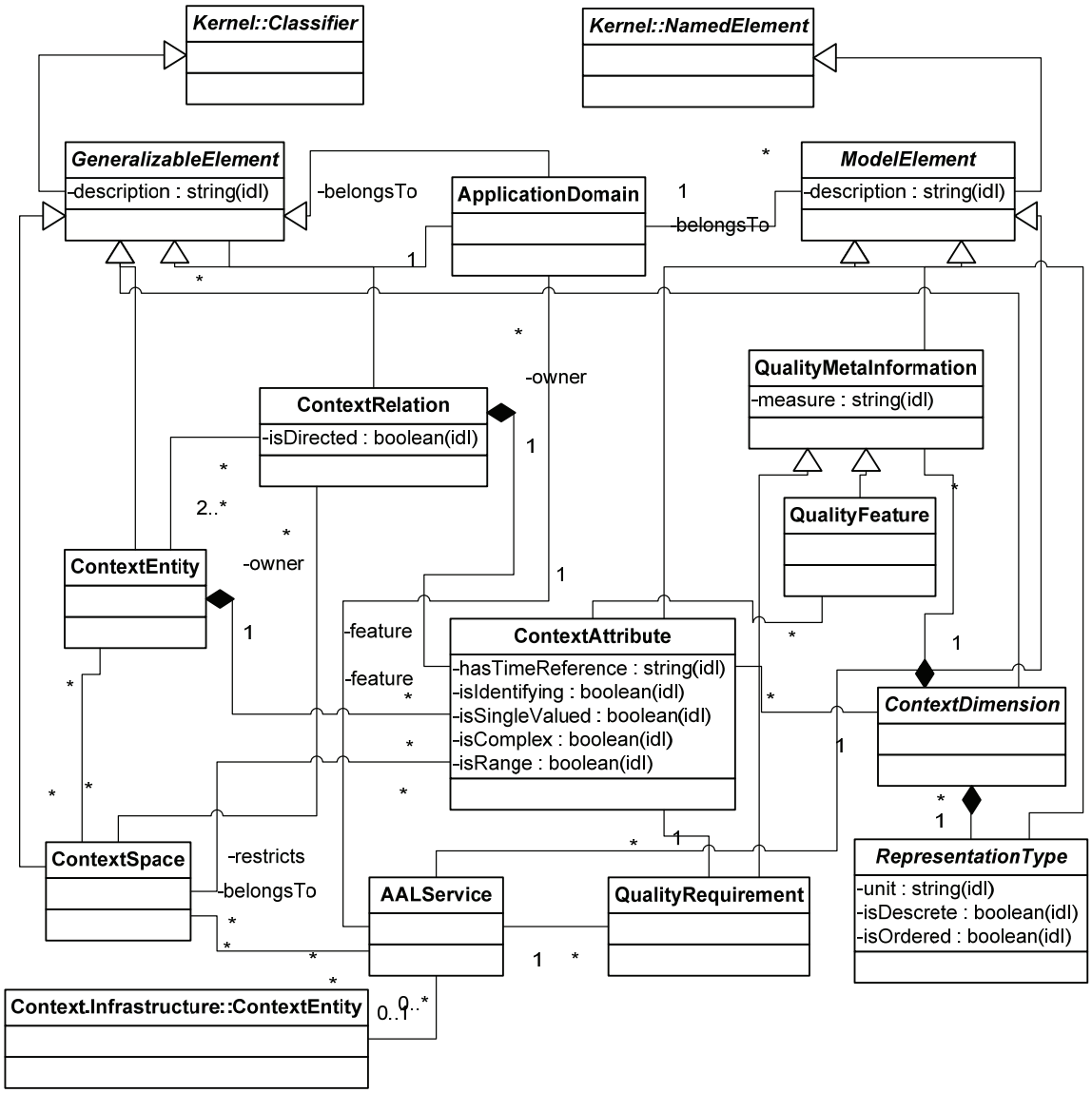


Abbildung 49: Modellelemente „Kontextmodell“ im Package „Context.Application“

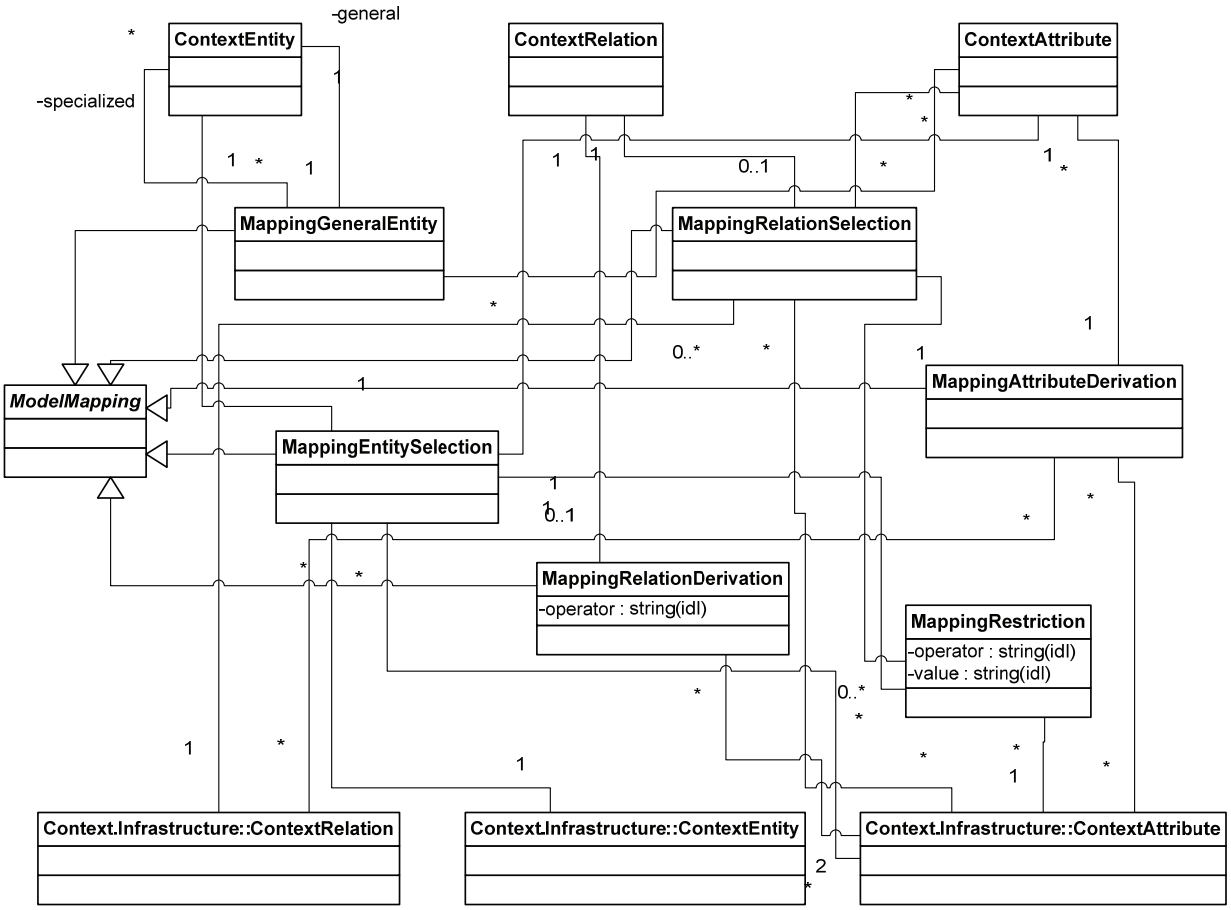


Abbildung 50: Modellelemente “Mapping” im Package “Context.Application”

6.5.1 AALService

Generalisierungen

- „ModelElement“

Beschreibung

Dieses Modellelement beschreibt die Zugehörigkeit eines AAL-Dienstes zu einem Dienstetyp und die Anforderungen an die Kontextinformationen und die benötigten Kontexträume.

Attribute

Dieses Modellelement besitzt keine eigenen Attribute.

Beziehungen

- Ein AAL-Dienst ist einem AAL-Dienstetyp zugeordnet.
- Ein AAL-Dienst kann kein, ein oder mehrere Kontexträume benötigen.
- Ein AAL-Dienst kann kein, ein oder mehrere Kontextinformationen in einer erforderten Qualität benötigen.
- Ein AAL-Dienst kann einer Kontextentität im Kontextmodell auf Ebene der Infrastruktur zugeordnet werden.

6.5.2 ApplicationDomain

Generalisierungen

- „GeneralizableElement“

Beschreibung

Dieses Modellelement beschreibt die Anwendungsdomäne bzw. den AAL-Dienstetyp, dem das Kontextmodell zugeordnet wird, z.B. „health“.

Attribute

Dieses Modellelement besitzt keine eigenen Attribute.

Beziehungen

- Es können kein, ein oder mehrere Modellelemente („ModelElement“) einer Anwendungsdomäne zugeordnet werden.
- Es können kein, ein oder mehrere generalisierbare Modellelemente („GeneralizableElement“) einer Anwendungsdomäne zugeordnet werden.

6.5.3 ContextAttribute

Generalisierungen

- „ModelElement“

Beschreibung

Das Modellelement „ContextAttribute“ beschreibt die beobachtbare Eigenschaft einer Kontextentität oder einer Kontextrelation. Ein Attribut kann für die Umsetzung der Vergleichsoperationen die Unterscheidung zwischen einem Punkt und einem Raum innerhalb der Kontextdimension benötigen. Zusätzlich sind weitere Eigenschaftsbeschreibungen für die Bereitstellung der Kontextinformationen notwendig, die bereits Bestandteil der Beschreibung auf Ebene der Infrastruktur sind.

Attribute

- hasTimeReference : string

  „past“: Das Kontextattribut hat einen expliziten Zeitbezug in die Vergangenheit. Die Historie ist Bestandteil des Kontextattributs.

  „present“: Der Zeitbezug des Kontextattributs bezieht sich auf die Gegenwart.

  „future“: Das Kontextattribut hat einen expliziten Zeitbezug in die Zukunft.

  Ist kein Argument angegeben, so bezieht sich der Zeitbezug lediglich auf die Gegenwart. Es können kommasepariert mehrere Argumente angegeben werden.

- isIdentifying : boolean

  true: Durch den Wert des Kontextattributs wird die Kontextentität eindeutig identifiziert.

  false: Der Wert des Kontextattributs ist nicht eindeutig und kann daher nicht für die Identifizierung einer Kontextentität genutzt werden.

- isSingleValued : boolean

  true: Das Kontextattribut kann nur einen Wert annehmen.

  false: Das Kontextattribut kann mehrere Werte annehmen.

- isComplex : boolean

  true: Das Kontextattribut kann einen zusammengesetzten komplexen Wert annehmen.

  false: Das Kontextattribut kann nur einen einfachen Wert annehmen.

- isRange : boolean

  true: Das Kontextattribut beschreibt einen Bereich innerhalb der Wertemenge der zugrundeliegenden Dimension, z.B. ein Zeitabschnitt.

  false: Das Kontextattribut beschreibt einen Punkt innerhalb der Wertemenge der zugrundeliegenden Dimension, z.B. einen Zeitpunkt.

Beziehungen

- Das Modellelement „ContextAttribute“ steht in einer Aggregationsbeziehung zu den Modellelementen „ContextEntity“ oder „ContextRelation“. Sie beschreibt die Eigenschaften „feature“ einer Kontextentität bzw. einer Kontextrelation. Diese wiederum sind die Besitzer „owner“ eines Kontextattributs.

- Ein Kontextattribut ist einer Kontextdimension zugeordnet und bezieht sich auf diese, z.B. einer Zeit- oder einer Ortsdimension.

- Einem Kontextattribut sind Metainformationen bezüglich der Qualität der enthaltenen Kontextinformationen zugeordnet.

- Es können Qualitätsanforderungen an ein Kontextattribut aus Sicht eines AAL-Dienstes definiert sein.

- Das Modellelement „ContextAttribute“ ist einem Modellelement „AttributeMapping“ zugeordnet, in welchem die Abbildung des Kontextattributs auf die Ebene der Infrastruktur beschrieben ist.

- Ein Kontextattribut kann zur Einschränkung eines Kontextraums verwendet werden.

6.5.4 ContextDimension

Generalisierungen

- „GeneralizableElement“

Beschreibung

„ContextDimension“ beschreibt die möglichen Kontextdimensionen, denen ein Kontextattribut zugeordnet werden kann. Eine Kontextdimension ordnet auf der Modellebene einem Kontextattribut einen Werteraum zu. Damit wird festgelegt, welche konkreten Werte ein Attribut annehmen kann. Hierüber sind ihre Eigenschaften, aber auch die möglichen Operationen auf den Attributen bestimmt. Ein Kontextattribut muss genau einer Kontextdimension zugeordnet werden, die ihrerseits aber mehreren Kontextattributen zugeordnet werden kann. Als Beispiel ist die Zeit als eine Kontextdimension festzulegen, der die Attribute „Öffnungszeit“ und „Schließzeit“ einer Entität „Einkaufszentrum“ zugeordnet werden können. Es ist möglich, unterschiedliche Kontextattribute der gleichen Kontextdimension zueinander in Beziehung zu setzen bzw. miteinander zu vergleichen.

Attribute

Dieses Modellelement besitzt keine eigenen Attribute

Beziehungen

- Eine Kontextdimension kann mehreren Kontextattributen zugeordnet werden.

- Eine Kontextdimension besitzt eine Representationsform.

6.5.5 ContextEntity

Generalisierungen

- „GeneralizableElement“

Beschreibung

Das Modellelement „ContextEntity“ ermöglicht die Modellierung der Kontextentitäten. Seine Instanzen können in einer Generalisierungs- oder Spezialisierungsrelation zueinander in Beziehung stehen. Es können auch Aggregationsbeziehungen zwischen Kontextentitäten definiert werden.

Attribute

Dieses Modellelement besitzt keine eigenen Attribute.

Beziehungen

- Eine Kontextentität kann kein, ein oder mehrere Kontextattribute besitzen.
- Eine Kontextentität kann einer Kontextrelation zugeordnet sein.
- Eine Kontextentität kann in kein, ein oder mehreren Kontexträume enthalten sein.
- Das Modellelement „ContextEntity“ ist einem Modellelement „EntityMapping“ zugeordnet, in welchem die Abbildung der Kontextentität auf die Ebene der Infrastruktur beschrieben ist.

6.5.6 ContextRelation

Generalisierungen

- „GeneralizableElement“

Beschreibung

Das Modellelement „ContextRelation“ ermöglicht die Modellierung der Kontextrelationen zwischen Kontextentitäten, wie die Beziehung zwischen Personen und Geräten, die gerichtet sein kann. Die beobachtbaren Eigenschaften dieser Beziehungen werden in den zugehörigen Kontextattributen beschrieben. Instanzen dieses Modellelements können in einer Generialisierungs- oder Spezialisierungsrelation zueinander bestehen.

Attribute

- isDirected : boolean

  true: Die Beziehung zwischen den Kontextentitäten ist gerichtet.

  false: Die Beziehung zwischen den Kontextentitäten ist ungerichtet.

Beziehungen

- Eine Kontextrelation kann kein, ein oder mehrere Kontextattribute besitzen.
- Einer Kontextrelation können zwei oder mehr Kontextentitäten zugeordnet sein.
- Eine Kontextrelation kann in kein, ein oder mehreren Kontexträume enthalten sein.
- Das Modellelement „ContextRelation“ ist einem Modellelement „RelationMapping“ zugeordnet, in welchem die Abbildung der Kontextrelation auf die Ebene der Infrastruktur beschrieben ist.

6.5.7 ContextSpace

Generalisierungen

- „GeneralizableElement“

Beschreibung

Das Modellelement „ContextSpace“ ermöglicht die Modellierung von Kontexträumen, um relevante Ausschnitte aus dem Zustandsraum des Kontextmodells zu definieren und ihnen Situationen sowie relevante Kontextentitäten und –relationen zuzuordnen. Außerdem werden die charakteristischen Kontextattribute in einschränkender Funktion dem Modellelement beigefügt. Instanzen dieses Modellelements können zueinander in einer Generalisierungs- oder Spezialisierungsrelation stehen.

Attribute

Das Modellelement besitzt keine eigenen Attribute.

Beziehungen

- Ein Kontextraum kann ein oder mehr Kontextentitäten beinhalten.

- Ein Kontextraum kann kein, ein oder mehr Kontextrelationen beinhalten.
- Ein Kontextraum kann durch kein, ein oder mehr Kontextattribute eingeschränkt werden.
- Ein Kontextraum kann von AAL-Diensten zur Umsetzung des kontextadaptiven Verhaltens benötigt werden.

6.5.8 GeneralizableElement

Generalisierungen

- „Classifier (from Kernel)“

Beschreibung

Ein „GeneralizableElement“ beschreibt alle Modellelemente deren Instanzen in einer Generalisierungs- oder Spezialisierungshierarchie zueinander in Beziehung stehen können. Zudem darf eine Instanz dieses Modellelements abstrakt sein, d.h. selbst nicht instantiiert werden. Dieses gilt für alle Modellelemente des Metamodells, die von „GeneralizableElement“ abgeleitet sind und somit dessen Eigenschaften erben. Auf der Ebene der Dienste ist dieses beispielsweise das Modellelement „ContextEntity“. „GeneralizableElement“ ist eine abstrakte Metaklasse. Diese Eigenschaft erbt es von der Klasse „Classifier“ im Package „Kernel“ des UML-Metamodells.

Attribute

- description : string

  Beschreibung des Modellelements

Beziehungen

- Ein generalisierbares Modellelement auf Ebene der Dienste ist einem Dienstetyp („ApplicationDomain“) zugeordnet.

6.5.9 MappingAttributeDerivation

Generalisierungen

- „ModelMapping“

Beschreibung

Dieses Modellelement wird verwendet um die Ableitung eines Modellelements „ContextAttribute“ aus einer Kontextrelation aus der Ebene der Infrastruktur zu beschreiben (F6).

Attribute

Dieses Modellelement besitzt keine eigenen Attribute.

Beziehungen

- Zuordnung zum abgeleiteten Kontextattribut auf Ebene der Dienste.
- Zuordnung zur Kontextrelation auf Ebene der Infrastruktur („Context.Infrastructure::ContextRelation“), aus dessen Wert das neue Kontextattribut abgeleitet werden kann.
- Zuordnung zum Kontextattribut auf Ebene der Infrastruktur („Context.Infrastructure::ContextAttribut“), dessen Wert bei Gültigkeit der Kontextrelation in das abgeleitete Kontextattribut übernommen werden soll.

6.5.10 MappingEntitySelection

Generalisierungen

- „ModelMapping“

Beschreibung

Dieses Modellelement wird verwendet um die Abbildung eines Modellelements „ContextEntity“ von der Ebene der Infrastruktur zu beschreiben. Dabei können Kontextattribute aus der Ebene der Infrastruktur übernommen werden. Die Abbildung kann im Rahmen der Selektion bzw. spezialisierenden Selektion einer Kontextentität (F1, F2) erfolgen.

Attribute

Dieses Modellelement besitzt keine eigenen Attribute.

Beziehungen

- Zuordnung zur Kontextentität auf Ebene der Dienste.
- Zuordnung zu den definierten Kontextattributen der Kontextentität auf Ebene der Dienste.
- Zuordnung zur entsprechenden Kontextentität auf Ebene der Infrastruktur („Context.Infrastructure::ContextEntity“).
- Zuordnung zu den übernommenen Kontextattributen der Kontextentität auf Ebene der Infrastruktur („Context.Infrastructure::ContextAttribute“).
- Es kann keine, eine oder mehrere Restriktionen („MappingRestriction“) für eine spezialisierende Selektion definiert werden.

6.5.11 MappingGeneralEntity

Generalisierungen

- „ModelMapping“

Beschreibung

Dieses Modellelement wird verwendet um eine generelle Kontextentität zu beschreiben (F7).

Attribute

Dieses Modellelement besitzt keine eigenen Attribute.

Beziehungen

- Zuordnung zur generellen Kontextentität auf Ebene der Dienste.
- Zuordnung zur den speziellen Kontextentitäten auf Ebene der Dienste.
- Zuordnung der Kontextattribute der speziellen Kontextentitäten auf die Kontextattribute der generellen Kontextentität.

6.5.12 MappingRelationDerivation

Generalisierungen

- „ModelMapping“

Beschreibung

Dieses Modellelement wird verwendet um die Ableitung eines Modellelements „ContextRelation“ aus dem Vergleich von zwei Kontextattributen der Ebene der Infrastruktur zu beschreiben (F5).

Attribute

- operator : string

  Der zu verwendende Vergleichsoperator.

Beziehungen

- Zuordnung zur abgeleiteten Kontextrelation auf Ebene der Dienste.
- Zuordnung zum zugehörigen Kontextattribut auf Ebene der Dienste.
- Zuordnung zur den beiden Kontextattributen auf Ebene der Infrastruktur („Context.Infrastructure::ContextAttribute“) deren Werte miteinander verglichen werden sollen.

6.5.13 MappingRelationSelection

Generalisierungen

- „ModelMapping“

Beschreibung

Dieses Modellelement wird verwendet um die Abbildung eines Modellelements „ContextRelation“ von der Ebene der Infrastruktur zu beschreiben. Dabei können Kontextattribute aus der Ebene der Infrastruktur übernommen werden. Die Abbildung kann im Rahmen der Selektion bzw. spezialisierenden Selektion einer Kontextrelation (F3, F4) erfolgen.

Attribute

Dieses Modellelement besitzt keine eigenen Attribute.

Beziehungen

- Zuordnung zur Kontextrelation auf Ebene der Dienste.
- Zuordnung zu den definierten Kontextattributen der Kontextrelation auf Ebene der Dienste.
- Zuordnung zur entsprechenden Kontextrelation auf Ebene der Infrastruktur („Context.Infrastructure::ContextRelation“).
- Zuordnung zu den übernommenen Kontextattributen der Kontextrelation auf Ebene der Infrastruktur („Context.Infrastructure::ContextAttribute“).
- Es kann keine, eine oder mehrere Restriktionen („MappingRestriction“) für eine spezialisierende Selektion definiert werden.

6.5.14 MappingRestriction

Generalisierungen

Es sind keine Generalisierungen vorhanden.

Beschreibung

Dieses Modellelement beschreibt Restriktionen einer Abbildung eines Modellelements aus der Ebene der Infrastruktur in die Ebene der Dienste aufgrund eines einschränkenden Vergleichs auf einem Kontextattribut. Diese ist die Basis für die spezialisierende Selektion einer Kontextentität (F2) bzw. einer Kontextrelation (F4).

Attribute

- operator : string

  Der zu verwendende Vergleichsoperator.

- value : string

  Der Vergleichswert.

Beziehungen

- Zuordnung zu dem Kontextattribut auf Ebene der Infrastruktur („Context.Infrastructure::ContextAttribute“), welches dem einschränkenden Vergleich zugrundeliegt.

6.5.15 ModelElement

Generalisierungen

- „NamedElement (from Kernel)“

Beschreibung

Das „ModelElement“ bildet die Basis aller nicht generalisierbaren Modellelemente des Metamodells auf Ebene der Dienste. Jedes weitere ist von dieser abgeleitet und kann eine Beschreibung besitzen. Sie ist abgeleitet von der Klasse „NamedElement“ aus dem Package „Kernel“ des UML-Metamodells und erbt von dort die Zuordnung eines im Namensraum eindeutigen Namens.

Attribute

- description : string

  Eine textuelle Beschreibung des Modellelements.

Beziehungen

- Ein nicht generalisierbares Modellelement auf Ebene der Dienste ist einem Dienstetyp („ApplicationDomain“) zugeordnet.

6.5.16 ModelMapping

Generalisierungen

Keine Generalisierung vorhanden.

Beschreibung

Das abstrakte Modellelement „ModelMapping“ wird verwendet um Abbildungen des Kontextmodells auf Ebene der Infrastruktur auf die Ebene der Dienste zu beschreiben. Die möglichen Abbildungen sind in Kap. 6.4 definiert.

Attribute

Dieses Modellelement besitzt keine eigenen Attribute.

Beziehungen

Dieses Modellelement besitzt keine weiteren Beziehungen.

6.5.17 QualityMetaInformation

Generalisierungen

- „ModelElement“

Beschreibung

„QualityMetaInformation“ ist ein Modellelement, das die möglichen Qualitätseigenschaften zu einem Kontextattribut definiert. Die möglichen Qualitätseigenschaften sind von der Kontextdimension des Attributs abhängig.

Attribute

- measure : string

  Die Maßeinheit, mit der die Qualitätseigenschaft gemessen wird.

Beziehungen

- Die Definition einer Qualitätseigenschaft ist einer Kontextdimension zugehörig.

6.5.18 QualityFeature

Generalisierungen

- „QualityMetaInformation“

Beschreibung

„QualityFeature“ ist ein Modellelement, das die Qualitätseigenschaften einer Kontextinformation in einem Kontextattribut beschreibt.

Attribute

Dieses Modellelement besitzt keine eigenen Attribute.

Beziehungen

- Zuordnung zum Kontextattribut, dessen Qualitätseigenschaften beschrieben werden.

6.5.19 QualityRequirement

Generalisierungen

- „QualityMetaInformation“

Beschreibung

„QualityRequirement“ ist ein Modellelement, das die Qualitätsanforderungen einer Kontextinformation in einem Kontextattribut für einen AAL-Dienst beschreibt.

Attribute

Dieses Modellelement besitzt keine eigenen Attribute.

Beziehungen

- Zuordnung zum zugehörigen AAL-Dienst.
- Zuordnung zum Kontextattribut, dessen Qualitätsanforderungen beschrieben werden.

6.5.20 RepresentationType

Generalisierungen

- „ModelElement“

Beschreibung

Das Modellelement „RepresentationType“ ermöglicht die Beschreibung der existierenden Repräsentationsformen einer Kontextdimension, die mehrere Formen der Repräsentation besitzen kann. Beispielsweise ist eine Ortsdimension durch mehrere geographische Koordinatensysteme, wie Gauss-Krüger oder WGS84, repräsentierbar. Mit einer Repräsentationsform ist zumeist auch die entsprechende Einheit festgelegt. Auf der Ebene der Dienste kann die Repräsentationsform festgelegt werden, welche von den jeweiligen Diensten benötigt wird.

Repräsentationsformen von Kontextdimension können unterschiedliche Eigenschaften haben und deshalb zwischen diskreten und kontinuierlichen Werten unterscheiden. Wertemengen können aufzählbar oder unendlich sein und eine Ordnung besitzen, die zur Umsetzung der Vergleichsoperationen notwendig ist.

Attribute

- unit : string

  Einheit der Repräsentation einer Kontextinformation in der jeweiligen Kontextdimension.

- isDescrete : boolean

  true - Der Wertebereich der Repräsentationsform der Kontextdimension besteht aus diskreten aufzählbaren Werten.

  false - Der Wertebereich der Repräsentationsform der Kontextdimension besteht aus kontinuierlichen Werten.

- isOrdered : boolean

  true - Der Wertebereich der Repräsentationsform der Kontextdimension besitzt eine Ordnung.

  false - Der Wertebereich der Repräsentationsform der Kontextdimension ist ohne Ordnung.

Beziehungen

Dieses Modellelement besitzt keine weiteren Beziehungen.

6.6 Informelle Kontextbeschreibung

Eine informelle Kontextbeschreibung wird in den Phasen der Planung und der Definition sowie in der Testphase benötigt. Die konkrete Form der Kontextbeschreibung ist in diesen Phasen unterschiedlich. Daher werden diese nachfolgend getrennt betrachtet.

6.6.1 Planungsphase

In der Planungsphase sollen die wichtigsten Eigenschaften des AAL-Dienstes festgelegt, die Machbarkeit abgeschätzt und die Möglichkeit für weitere Aktivitäten eingeräumt werden. Allgemein anerkanntes Instrument zur Sammlung und Dokumentation der funktionalen und nicht-funktionalen Anforderungen an eine zu erstellende Software ist das Lastenheft, das in dieser Phase auch hier eingesetzt werden soll. Derzeit existiert jedoch keine Entwicklungsmethodik, die die Definition kontextadaptiver Anwendungen mit Hilfe eines Lastenheftes betrachtet. Daher wird nachfolgend ein erweiterter Ansatz erarbeitet.

Bei der Erstellung eines Lastenheftes für einen AAL-Dienste muss zwischen kontextabhängiger und davon unabhängiger Funktionalität unterschieden werden. Die kontextabhängige Funktionalität soll in einem eigenständigen Kapitel zusammengefasst werden, damit sie als Grundlage für die Erstellung des konzeptuellen Kontextmodells verwendet werden kann. Hier soll kurz die eigentliche Funktion und ihr kontextadaptives Verhalten im Klartext beschrieben werden. Das Lastenheft soll letztlich eine gesammelte Darstellung der benötigten kontextadaptiven Funktionen des AAL-Dienstes ergeben.

Als Beispiel wird nachfolgend ein AAL-Medikamentendienst beschrieben, der zwei Funktionen besitzen soll. Die eine führt die Überwachung des Bestands eines Medikaments im Medizinschrank durch und löst bei Bedarf die Nachbestellung über den Hausarzt und weiteren Dienstleistern aus. Die andere überwacht die Einnahme des Medikaments durch den Bewohner und erzeugt eine Erinnerung, falls sie nicht innerhalb einer definierten Zeitspanne erfolgt ist und über das dem Nutzer am nächsten gelegene Endgerät ausgegeben wird. Es kann je nach Situation das Handy des Nutzers oder aber auch ein Display in der Wohnung sein. Die Beschreibung einer kontextadaptiven Funktionalität im Lastenheft soll durch ein vorangehendes „K“ in der Funktionsaufzählung gekennzeichnet werden.

K.01 Überwachung des Medikamentenvorrats

Funktion: Die Funktion überwacht den Füllstand eines Medikaments im Medizinschrank und löst sobald eine Mindestmenge unterschritten ist eine Bestellung über einen Arzt und weiteren Dienstleistern aus.

K.02 Medikamentenerinnerung

Funktion: Die Funktion übermittelt dem Nutzer über das nächstgelegene Endgerät eine Nachricht, welche ihn an die Einnahme seines Medikaments erinnern soll.

In der Funktion „K.02“ wird die Übermittlung einer Erinnerung an das nächstgelegene Endgerät beschrieben, welche für sich betrachtet, bereits eine kontextadaptive Funktion darstellt. Diese beschreibt jedoch ein strukturelles kontextadaptives Verhalten (Kap. 3.3.1) und wird daher im Lastenheft nicht als eigene dem Dienst zugehörige Funktion erfasst. Wie aus dem Beispiel ersichtlich, werden im Text der Beschreibung bereits die wesentlichen Inhalte des Kontextmodells festgelegt, hier jedoch nur informell, um als erster Anhaltspunkt für die weitere Modellierung zu dienen.

6.6.2 Definitionsphase

Ziel dieser Phase ist die weitere Konkretisierung der Anforderungen an die Kontextadaptivität eines AAL-Dienstes. Mit ihr muss das Produkt soweit eindeutig definiert sein, dass diese Dokumentation Grundlage für die Abnahme durch einen Kunden sein kann. Das hier verwendete Instrument ist das Pflichtenheft, welches auf dem Lastenheft aufbaut und es erweitert.

In der Konkretisierung der kontextadaptiven Funktionen des zu erstellenden AAL-Dienstes müssen folgende Punkte ergänzt werden:

- Beschreibung der Kontextbedingungen für das Auslösen der Funktion, z.B. beim Vergessen der Einnahme eines Medikaments. Hier soll bereits zwischen der Kontextentität bzw. Kontextrelation und den zugehörigen Kontextattributen unterschieden und die relevanten Situationen sowie deren Charakteristika beschrieben werden.

- Beschreibung der Anforderungen an die Qualität und die Form der Bereitstellung der Kontextinformationen, z.B. die konkrete Repräsentationsform.

- Beschreibung der Anforderungen an die Zuverlässigkeit der Kontexterkennung und der möglichen Auswirkungen einer Fehlerkennung. Wie in Kap. 2.2.8 beschrieben, sind Kontextinformationen generell unsicherer Natur. Deshalb muss bereits bei der Festlegung der Funktionalität beschrieben werden, welche Auswirkung ein Fehlverhalten des Dienstes in dieser Funktionalität für den Bewohner hat. Folgende Kategorien werden nun eingeführt, denen die aufgeführten kontextadaptiven Funktionen zugeordnet werden können. Als „Nicht akzeptabel“ wird eine Fehlerkennung bewertet, wenn die darauf basierende Funktion zu unmittelbarem Schaden für den Nutzer oder seiner Umgebung führt. Als „Kritisch“ wird sie bewertet, wenn diese nicht unmittelbar zu einem Schaden führt, jedoch für den Nutzer direkt erkennbar und belästigend wirkt und so zum Schwinden der Akzeptanz führen kann. Als „Akzeptabel“ kann eine Fehlerkennung bewertet werden, wenn sie zu keinem Schaden führt und auch für den Nutzer nicht sofort als störend empfunden wird.

- Unterscheidung, ob die Kontextbedingung von der Anwendung vorgegeben ist oder ob diese in einem vorzugebenden Rahmen vom Nutzer definiert werden kann. Das ist beispielsweise der Fall, wenn der Anwender angeben kann, welches konkrete Medikament bei der Einnahme überwacht werden soll.

Nachfolgend wird beispielhaft die erweiterte Beschreibung der Funktion K.01 aus dem Lasteheft beschrieben.

K.01 Überwachung des Medikamentenvorrats

Funktion: Die Funktion überwacht den Füllstand eines Medikaments im Medizinschrank und löst bei Bedarf eine Bestellung über einen Arzt und weiteren Dienstleistern aus.

Kontextbedingung: Die Funktion soll angestoßen werden, sobald eine Mindestmenge des Medikaments unterschritten ist (Medikamentenvorrat knapp).

Anforderung an Qualität und Repräsentation: Der Füllstand soll für Tabletten in Stück ermittelt werden. Für flüssige Medikamente erfolgt dieses in ml.

Zuverlässigkeit: Die Erkennung einer Unterschreitung muss zuverlässig erfolgen. Eine Fehlerkennung wird als „Nicht akzeptabel" eingestuft. Das Nichtvorhandensein eines Medikaments kann schwerwiegende Auswirkungen auf den Nutzer haben.

Definition durch Anwender: Der Anwender hat die Möglichkeit die Medikamente festzulegen, deren Einnahme überwacht werden soll. Außerdem kann auch die Mindestmenge angegeben werden.

Die Beschreibung der Kontextbedingungen und der Anforderungen an die Qualität und Repräsentation bilden die Grundlage für das in der Designphase zu erstellende konzeptuelle Kontextmodell.

Die Auswirkungen einer Fehlerkennung müssen in der Konzeption der Anwendungslogik des AAL-Dienstes beachtet werden. Grundsätzlich unterscheidet sich diese nicht von der sonstiger Anwendungen. Daher können die gängigen Methoden zur Spezifikation der Anwendungslogik verwendet werden, auf die hier nicht weiter eingegangen wird. Ein Unterschied besteht jedoch darin, dass bei der Konzeption die Auswirkungen einer unsicheren Kontexterkennung beachtet werden müssen. Im Pflichtenheft sind daher die Funktionen bereits den einzelnen Kategorien „Nicht-akzeptabel", „Kritisch" und „Akzeptabel" zugeordnet worden. Durch die Festlegung der Anforderungen an die Qualität der Kontexterkennung soll verhindert werden, dass ein Fehlverhalten ausgelöst wird. Mit der Erhöhung der Anforderungen an die Zuverlässigkeit besteht aber die Gefahr, dass eine zugrundeliegende Kontextinfrastruktur diese nicht erfüllt und somit die Funktion nicht unterstützt. Die Erhöhung der Qualitätsanforderungen kann somit nicht das alleinige Mittel sein, um ein unerwünschtes Fehlverhalten zu vermeiden. Eine weitere Möglichkeit besteht in der Definition alternativer Funktionen, deren Fehlverhalten weniger kritisch einzustufen sind und die ersatzweise verwendet werden können, falls die hohen Qualitätsanforderungen nicht umzusetzen sind. Für den Umgang mit unsicheren Kontextinformationen bietet sich auch die Einbindung des Nutzers in Form einer Sicherheitsabfrage oder die Visualisierung von unsicheren Kontextinformationen an. Eine solche Einbindung ist aber auch kritisch zu sehen, da der Nutzer mög-

lichst nicht über eine unerwartete Mensch-Maschine-Interaktion belastet werden sollte. Der Umgang mit unsicheren Kontextinformationen ist eine noch offene Fragestellung in der Erforschung kontextadaptiver Anwendungen und ist deshalb noch in der Diskussion, z.B. in [165].

Die Beschreibung der Definierbarkeit kontextadaptiven Dienste-Verhaltens durch den Anwender ist eine Grundlage für das Kontextmodell auf der Ebene der Nutzerinteraktion.

6.6.3 Testphase

In dieser Phase sollen die im Pflichtenheft festgelegten Eigenschaften des Produkts abgesichert werden. Mit Hilfe des Pflichtenhefts werden daher in der Regel ein Testplan sowie eine Testspezifikation erstellt. Hier werden die Testfälle identifiziert und im Rahmen der Spezifikation weiter konkretisiert, wozu die Festlegung der Rahmenbedingungen, die Beschreibung des Ausgangszustands, die durchzuführenden Testschritte sowie die erwarteten Ergebnisse gehören. Testplan, Testspezifikation sowie die Testdurchführung werden nach einer vorgegebenen Struktur dokumentiert.

Das Testen der kontextadaptiven Funktionen des AAL-Dienstes folgt demselben Muster. Eine Grundlage für die Spezifikation der funktionalen Tests ist die Beschreibung der Kontextbedingungen im Pflichtenheft, die sowohl für die Beschreibung des Ausgangszustands als auch für die durchzuführenden Testschritte und des erwarteten Ergebnis benötigt werden. Nachfolgend wird eine beispielhafte Testspezifikationen erstellt.

T.K.01a Korrektes Auslösen der Benachrichtigungsfunktion

Beschreibung: Mit diesem Test soll sichergestellt werden, dass die Benachrichtigungsfunktion korrekt ausgelöst wird.

Kontextbedingung: Füllstand von Medikament 1 ist kleiner als 5 Stück

Ausgangszustand:

Medikament 1		
	Füllstand	*5 Stück*

Durchzuführende Schritte: Es wird ein Stück Medikament 1 aus dem Medizinschrank entnommen.

Erwartetes Ergebnis: Die Benachrichtigungsfunktion wird ausgelöst.

6.7 Konzeptuelles Kontextmodell

Auf der Basis des Pflichtenhefts erfolgt die konzeptuelle Umsetzung des kontextadaptiven Verhaltens des AAL-Dienstes. Hierzu gehören die in der Softwareentwicklung bekannten Methoden des Softwareentwurfs und der Datenmodellierung. Hinzu kommt nun die Konzeption der Kontextadaptivität durch die Erstellung eines konzeptuellen Kontextmodells. Dabei wird ein von der konkreten Implementierung unabhängiges Modell definiert. Es dient der Erarbeitung und Diskussion kontextadaptiver Aspekte durch die Entwickler. Die informellen Kontextbeschreibungen der einzelnen Funktionen aus dem Pflichtenheft werden dabei über das konzeptuelle Kontextmodell formalisiert und zusammengefasst.

Die hierfür benötigte konzeptuelle Modellierung setzt auf dem in der Ebene der Infrastruktur vorgestellten Notation (Kap. 5.6.3) auf und ergänzt sie um zusätzliche auf der Ebene der Dienste benötigte Modellelemente. Diese werden auf der Ebene der Dienste ebenfalls im vollen Umfang benötigt und werden zusätzlich um die Definition von Kontexträumen und den Qualitätsanforderungen ergänzt.

Das Modellelement „ContextSpace“ ist im erweiterten „Object-Role Model“ nicht vorgesehen. Als Erweiterung soll ein Kontextraum durch eine gestrichelte Umrandung visualisiert werden, die alle relevanten Kontextentitäten, Relationen und Kontextattribute umfasst. Außerdem sind in der Umrandung sein Name sowie eine textmäßig verfasste Beschreibung der attributiven Einschränkungen enthalten.

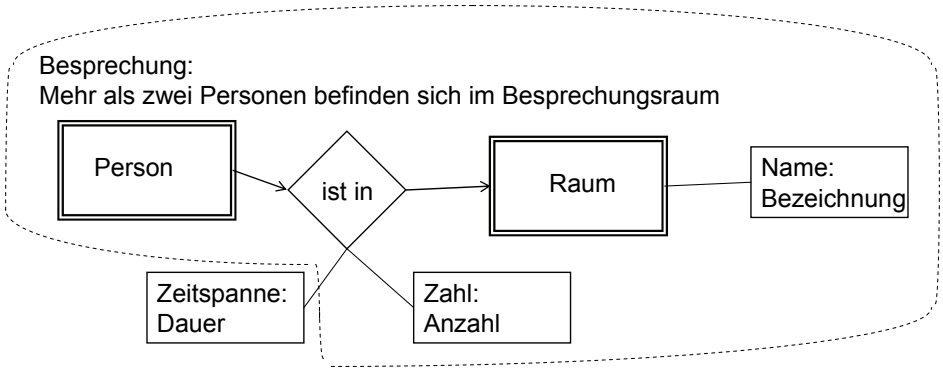


Abbildung 51: Kontextraum

Diese Notation ist nicht konform zu den Visualisierungsregeln für die UML-Stereotypes. Eine konforme Notation hierzu kann die Enthalten-Beziehung zwischen dem Kontextraum und den weiteren Modellelementen nicht geeignet darstellen.

Die Beschreibung von Qualitätsanforderungen an die Kontextinformationen ist bereits ein Bestandteil des Ansatzes von Henricksen, Indulska und Rakotonirainy. Diese Qualitätsanforderungen werden dort durch ein Oval repräsentiert, welches mit den entsprechenden Assoziationen verbunden wird. Hierdurch wird das Modellelement „QualityMetaInformation“ repräsentiert. Die Anforderungen an die Qualität sind im Klartext in das Oval einzutragen.

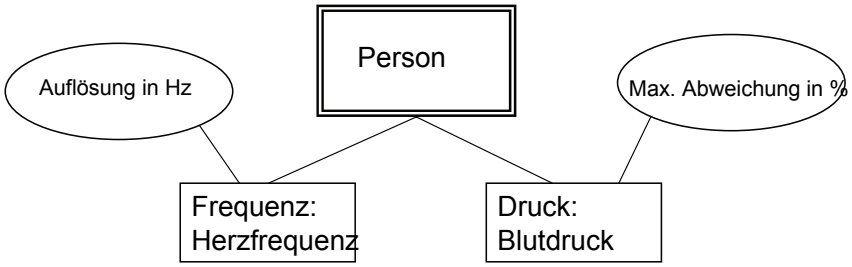


Abbildung 52: Qualitätsanforderungen

Mit der definierten grafischen Notation lassen sich nun die Anforderungen an das Kontextmodell eines AAL-Dienstes konzeptuell erfassen, dokumentieren und als Grundlage für dessen Realisierung nutzen.

6.8 Operatives Kontextmodell

Das operative Kontextmodell ist die lauffähige Umsetzung des konzeptuellen Kontextmodells. Zu dessen Realisierung müssen die im konzeptuellen Kontextmodell beschriebenen Modellelemente verfeinert und vervollständigt werden. Diese werden nachfolgend beschrieben. Abschließend wird eine implementationsneutrale Repräsentation des operativen Kontextmodells beschrieben.

6.8.1 Ergänzungen zum konzeptuellen Kontextmodell

Die Ergänzungen zum konzeptuellen Kontextmodell bestehen in der Definition der Überführungsfunktionen (Kap. 6.4), mit denen die Modellelemente in der Ebene Infrastruktur auf die definierten Modellelemente auf Ebene der Dienste abgebildet werden. Hierzu werden die folgenden Modellelemente zum operativen Kontextmodell ergänzt.

MappingAttributeDerivation: Dieses Modellelement wird verwendet um die Ableitung eines Modellelements „ContextAttribute“ aus einer Kontextrelation aus der Ebene der Infrastruktur zu beschreiben (F6).

MappingEntitySelection: Dieses Modellelement wird verwendet um die Abbildung eines Modellelements „ContextEntity“ von der Ebene der Infrastruktur zu beschreiben. Dabei können Kontextattribute aus der Ebene der Infrastruktur übernommen werden. Die Abbildung kann im Rahmen der Selektion bzw. spezialisierenden Selektion einer Kontextentität (F1, F2) erfolgen.

MappingGeneralEntity: Dieses Modellelement wird verwendet um eine generelle Kontextentität zu beschreiben (F7).

MappingRelationDerivation: Dieses Modellelement wird verwendet um die Ableitung eines Modellelements „ContextRelation“ aus dem Vergleich von zwei Kontextattributen der Ebene der Infrastruktur zu beschreiben (F5).

MappingRelationSelection: Dieses Modellelement wird verwendet um die Abbildung eines Modellelements „ContextRelation“ von der Ebene der Infrastruktur zu beschreiben. Dabei können Kontextattribute aus der Ebene der Infrastruktur übernommen werden. Die Abbildung kann im Rahmen der Selektion bzw. spezialisierenden Selektion einer Kontextrelation (F3, F4) erfolgen.

MappingRestriction: Dieses Modellelement beschreibt Restriktionen einer Abbildung eines Modellelements aus der Ebene der Infrastruktur in die Ebene der Dienste aufgrund eines einschränkenden Vergleichs auf einem Kontextattribut. Diese ist die Basis für die spezialisierende Selektion einer Kontextentität (F2) bzw. einer Kontextrelation (F4).

6.8.2 Implementationsneutrale Repräsentation

Für die Definition der implementationsneutralen Repräsentation des operativen Kontextmodells auf Ebene der Dienste gilt dieselbe Motivation wie für das operative Kontextmodell auf Ebene der Infrastruktur (Kap. 5.7.2). Auch hier wird die Überführung des Kontextmodells mit Hilfe von XMI in ein XML-Dokument vorgenommen. Die XMI-konforme Beschreibung des operativen Kontextmodells kann mit Hilfe eines Importmechanismus in den Kontextserver übernommen und in die implementationsspezifische Form überführt werden. Der grundsätzliche Import-Mechanismus ist Bestandteil des im Rahmen der Dissertation implementierten Kontextservers.

6.9 Deskriptives Kontextmodell

Das deskriptive Kontextmodell wird in der Phase der Bereitstellung und Übernahme von Bestandteilen der Infrastruktur in die häusliche intelligente Umgebung benötigt (Kap. 4.2.2). Folgende Bestandteile müssen auf Ebene der Dienste übernommen werden:

- AAL-Dienste: Kontextsensitive AAL-Dienste werden in die intelligente häusliche Umgebung eingebunden und können auf das dienstetyp-spezifische Kontextmodell zugreifen.

Die Beschreibung des kontextadaptiven Verhaltens der AAL-Dienste basiert auf dem Kontextmodell auf Ebene der Dienste. In der MOF Metadaten-Architektur ist eine konkrete Beschreibung auf Ebene M0 zu finden.

6.9.1 Dienstebeschreibung

Mit Hilfe einer Dienstebeschreibung können kontextadaptive AAL-Dienste in der Phase der Bereitstellung auf dem Dienstemarkt zur Verfügung gestellt und bei Bedarf in die intelligente häusliche Umgebung übernommen werden. Wichtiger Bestandteil eines solchen Dienstemarktes ist ein Verzeichnisdienst. Über diesen werden AAL-Dienste der Anbieter registriert und bereitgestellt. Die Registrierung ist eine wesentliche Funktionalität in einem solchen Dienstemarkt. Sie geht mit einer Beschreibung des zu registrierenden Dienstes einher und dient der Einordnung sowie der Auswahl von Diensten durch den Bewohner. Eine solche Dienstebeschreibung ermöglicht eine dynamische Integration eines kontextadaptiven AAL-Dienstes in die intelligente häusliche Umgebung des Bewohners. Fragen zur Architektur und Realisierung eines solchen Dienstemarktes sind nicht Gegenstand der weiteren Ausarbeitung. Nachfolgend wird lediglich die Form der Dienstebeschreibung betrachtet.

Die Registrierung von Diensten ist nicht nur ein Thema im AAL. In offenen Diensteinfrastrukturen, beispielsweise einer Serviceorientierten Architektur (SOA), sind Verzeichnisdienste ebenfalls vorhanden. Ein bekannter Ansatz ist UDDI (Universal Description, Discovery and Integration) [135]. Dieser Verzeichnisdienst ermöglicht die Registrierung von Diensten über eine SOAP-Schnittstelle. Die „White Pages“ enthalten hier die Informationen über die Identität des Serviceanbieters, z.B. das Geschäftsfeld, die Kontaktdaten eines Ansprechpartners und eine Unternehmensidentität. Die „Green Pages“ enthalten die Schnittstellenbeschreibung eines Dienstes. Hier werden die technischen Informationen in Form eines Servicetyps beschrieben, beispielsweise mit Hilfe der Web Service Description Language (WSDL) [166] und auch Informationen zu den verwendeten Protokollen sowie zur Netzadresse des Dienstes gegeben, die zur technischen Einbindung des Dienstes notwendig sind. In den „Yellow Pages“ erfolgt die Taxierung eines Dienstes hinsichtlich ihres Zweckes in ein Klassifikationsschema. Hierfür existieren Standard-Taxonomien, mit denen eine Einordnung zu einem bestimmten Industriezweig, einer Serviceeinrichtung oder einem bestimmten Gebiet erfolgen kann. Neben UDDI existieren noch weitere Ansätze innerhalb gängiger JEE Anwendungsserver (z.B. IBM WebSphere) oder des OSGi Frameworks [30] für die Realisierung eines Verzeichnisdienstes.

Aktueller Gegenstand der Forschung ist die Erweiterung der rein syntaktischen Dienstebeschreibung um eine semantische Beschreibung hin zu „Semantic Web Services“ [167]. Dadurch soll das Auffinden, Auswählen und Ausführen der Dienste, sowie die Komposition von Diensten erleichtert werden. Der Einsatz einer Ontologie ermöglicht die Maschinen-verständliche Interpretation der Dienstebeschreibung bis hin zur automatisierten Orchestrierung von Diensten auf Basis einer Zielbeschreibung. Beispiele für semantische Dienstebeschreibungen sind die „Web Service Modeling Ontology“ (WSMO) [168], die „Ontology Web Language for Services“ (OWL-S) [169] oder „METEOR for Semantic Web Services (METEOR-S) [170].

OWL-S basiert auf OWL (Kap. 4.3.5) und ermöglicht die Beschreibung von Eigenschaften und Fähigkeiten von Diensten im Web. Es soll das automatische Auffinden, Ausführen, Komposition und Überwachen dieser Dienste ermöglichen. OWL-S ergänzt bestehende Standards für Webdienste wie WSDL und UDDI um eine semantische Schicht. Es besteht aus drei unterschiedlichen Teilen bzw. Profilen. Das „ServiceProfile“ dient dem Auffinden eines Dienstes und enthält Informationen über die Organisation die den Dienst anbietet, über die Vorbedingungen, Eingabe- und Ausgabewerte, sowie die Eigenschaften und den Nutzen des Dienstes. Das Modellelement „ServiceProfile“ ermöglicht die Zuordnung eines Dienstes zur Profilbeschreibung. Das Modellelement „Profile“ beinhaltet Informationen zum Dienstenamen, zu Kontaktinformationen und eine Beschreibung der Funktionalität. Die zugehörigen Modellelemente sind „ServiceCategory“ zur Einordnung des Dienstes in eine Taxonomie und „ServiceParameter“ zur Beschreibung der Schnittstellen. Zudem wird in „Profile“ die Vorbedingung für die Ausführung des Dienstes, sowie das Resultat beschrieben. In der nachfolgenden Abbildung wird eine Übersicht der Profilontologie aus der zugehörigen Spezifikation gegeben.

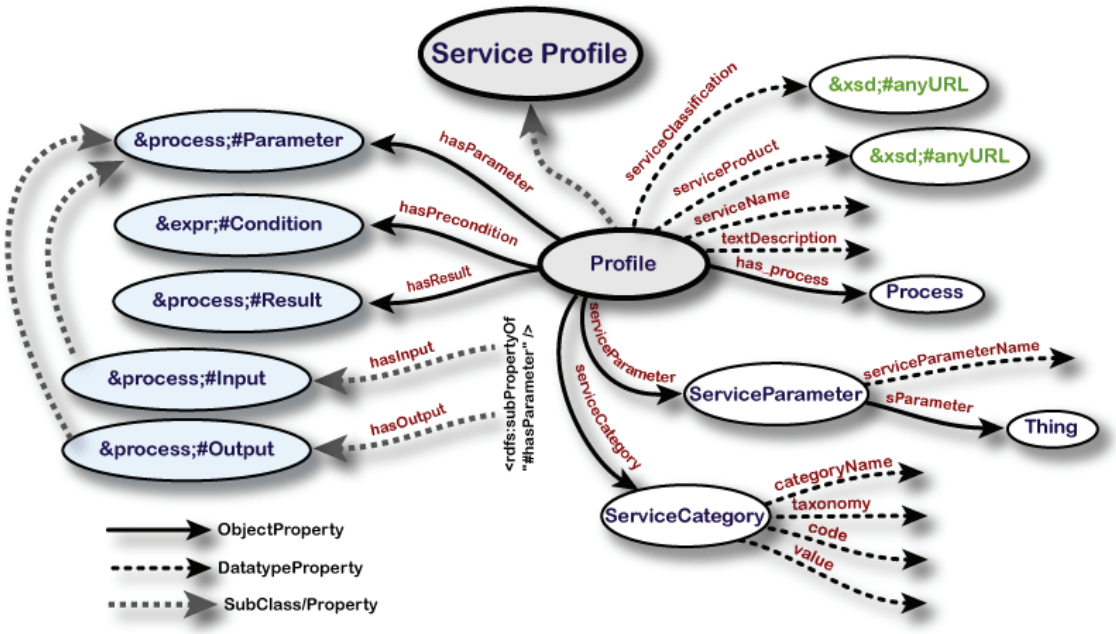


Abbildung 53: OWL-S Profilontologie [169]

Das „ServiceModel“ beschreibt die Funktionalität des Dienstes in Form einer Prozessbeschreibung. Die Grundlage hierfür ist eine Prozessontologie. Diese beinhaltet Konzepte zur Beschreibung des Aufbaus eines Prozesses aus atomaren, einfachen und zusammengesetzten Prozessteilen. Zudem können die Kontrollflüsse definiert werden. Die dazu einsetzbaren Modellelemente sind „Sequence“, „Split“, „Slit+Join“, „Unordered“, „Choice“, „If-Then-Else“, „Iterate“ und „Repeat“. Zudem kann der Datenfluss und die Parameterbindung an die einzelnen Prozessschritte beschrieben werden. Dieses kann für die automatisierte Komposition und Ausführung von Diensten genutzt werden. In der nachfolgenden Abbildung wird eine Übersicht dieser Prozessontologie aus der zugehörigen Spezifikation gegeben.

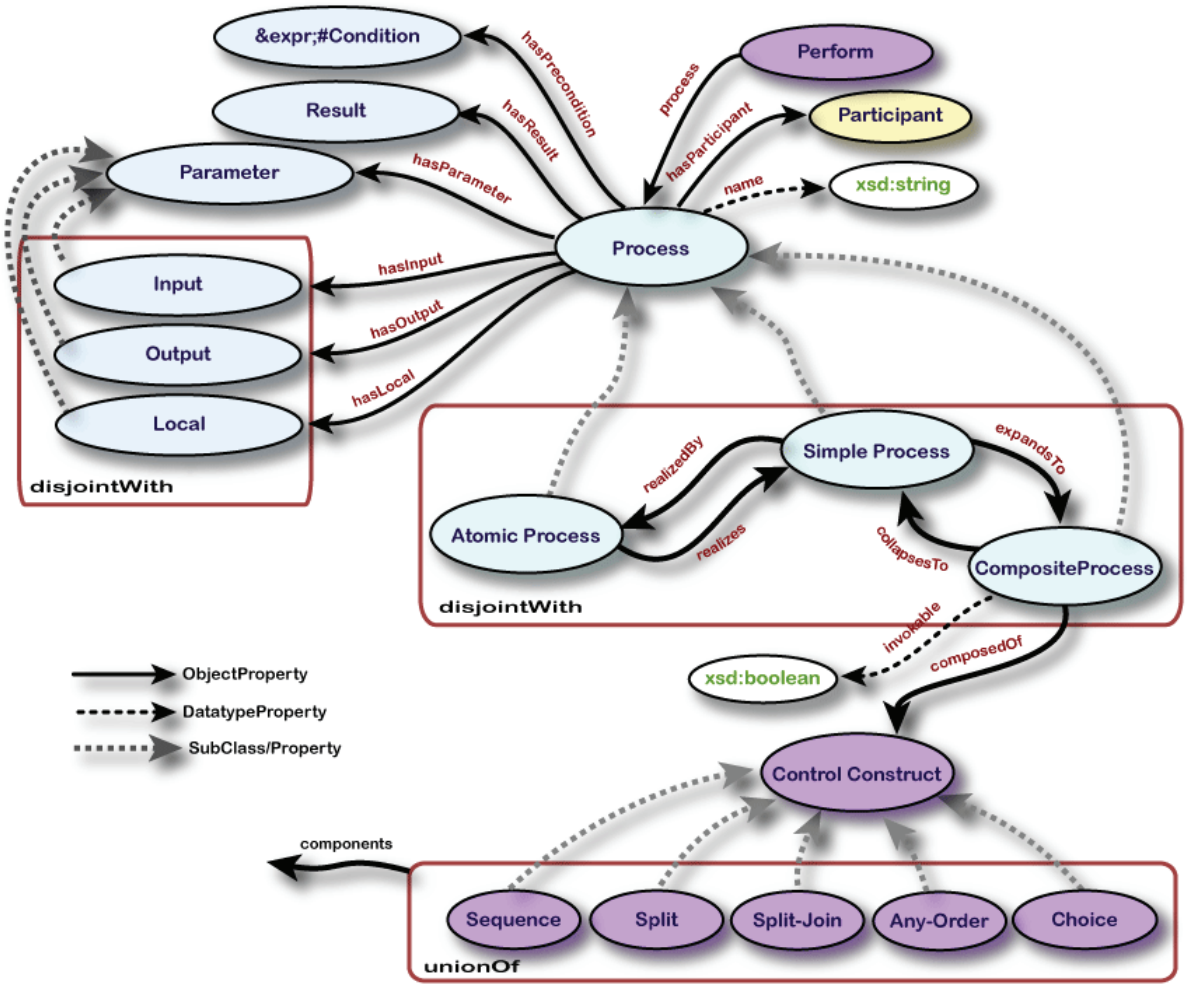


Abbildung 54: OWL-S Prozessontologie [169]

Das „ServiceGrounding“ beinhaltet technische Details bezüglich der verwendeten Protokolle, Formate und zur Adressierung. Diese Informationen werden für den Aufruf der Dienste benötigt. Hierbei wird WSDL verwendet. Jede dieser Profile wird in Form einer Ontologie definiert, welche zur semantischen Beschreibung eines Dienstes genutzt werden kann.

Sowohl mit der syntaktischen als auch mit der semantischen Dienstebeschreibung existiert bereits eine Reihe von Technologien die für die Beschreibung, Bereitstellung und Integration von AAL-Diensten genutzt werden kann. Die Kontextadaptivität ist eine zusätzliche Eigenschaft der eigentlichen Funktionalität eines AAL-Dienstes (Kap. 3.3.1). Somit muss eine bestehende Dienstebeschreibung den Aspekt der Kontextadaptivität beinhalten. Folgende Elemente

des Kontext-Metamodells sind dabei in geeigneter Form in der Dienstebeschreibung abzubilden:

- ApplicationDomain: Der AAL-Dienst muss einem Dienstetyp zugeordnet werden, beispielsweise „health“, „security“, „comfort“, „social“ oder „economy“ (Kap. 3.3.2).

- AALService, ContextAttribute, ContextEntity, ContextRelation, ContextSpace, QualityRequirement: Die Einzelfunktion eines Dienstes kann ein kontextadaptives Verhalten besitzen. Zum Abgleich der Anforderungen an die intelligente Umgebung ist eine explizite Beschreibung der benötigten Kontextunterstützung notwendig. Es muss zu einer Funktion des Dienstes beschrieben werden, welche Kontextinformationen in welcher Qualität benötigt werden.

Mit Hilfe von UDDI ist eine Zuordnung eines AAL-Dienstes zu einem Dienstetyp mittels der „Yellow Pages“ möglich. Für die Klassifizierung existieren aus unterschiedlichen Anwendungsdomänen eine Reihe von Taxonomien, die zur Einordnung eines Dienstes genutzt werden können, z.B. UN/SPSC [171]. Für die Beschreibung von AAL-Dienstetypen existiert noch keine solche Taxonomie und müsste ergänzt werden. Die Beschreibung der Anforderungen an die Kontextunterstützung ist mit Hilfe von UDDI nicht möglich.

In OWL-S kann sowohl das „ServiceProfile“ als auch das „ServiceModel“ zur Beschreibung der Funktionalität eines Dienstes genutzt werden. Das „ServiceProfile“ dient dem Auffinden eines Dienstes. Das „ServiceModel“ beschreibt die Funktionalität im Detail und kann beispielsweise für das Monitoring oder die Komposition genutzt werden. Die Einordnung eines AAL-Dienstes zu einem Dienstetyp kann hier im Modellelement „ServiceCategory“ erfolgen. Ebenso wie bei UDDI muss hier eine Taxonomie der AAL-Dienstetypen erstellt werden, dem der Dienst zugeordnet werden kann. Die Beschreibung der Anforderungen an die Kontextunterstützung kann in der Eigenschaft „hasPrecondition“ des Modellelements „Profile“ erfolgen. Diese sind genereller Art und beziehen sich nicht explizit auf Kontextbedingungen. Die relevanten Teile des Kontextmodells müssen hierfür in geeigneter Art in die Beschreibung der Vorbedingungen überführt werden. Vorbedingungen können in unterschiedlichen Sprachen beschrieben werden, z.B. SWRL [172], RDF [145], KIF [173] oder PDDL [174]. Die konkrete Abbildung der Kontextbedingungen in eine der Sprachen wird nicht weiter betrachtet.

Es gibt eine Reihe von Forschungsarbeiten zu kontextadaptiven Webdiensten und deren Beschreibung, z.B. [175] [176]. Die Kontextadaptivität ist hierbei jedoch auf den Aspekt des dynamischen Auffindens von Diensten in Abhängigkeit des jeweiligen Kontexts des Nutzers beschränkt. So soll beispielsweise in Abhängigkeit vom aktuellen Ort und dem verfügbaren Endgerät des Nutzers ein geeigneter und für den Ort gültiger Touristeninformationsdienst gefunden

werden. Die Kontextadaptivität bezieht sich somit auf die Verfügbarkeit des Dienstes in Abhängigkeit vom Kontext und nicht auf das kontextadaptive Verhalten einer Funktion innerhalb des Dienstes.

Es sind somit verschiedene Ansätze zur Beschreibung von Diensten verfügbar. Keiner dieser Ansätze unterstützt unmittelbar die Beschreibung der Anforderungen an die verfügbaren Kontextinformationen und Kontexträume. Es können jedoch bestehenden Ansätze um den Aspekt der Kontextadaptivität ergänzt werden. Die konkrete Umsetzung einer Dienstebeschreibung wird nachfolgend nicht weiter vertieft. Es wird jedoch eine abstrakte Beschreibung der notwendigen Ergänzung nachfolgend beschrieben, die in eine konkrete Ergänzung der jeweiligen Dienstebeschreibung überführt werden muss.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" element-
FormDefault="qualified">
  <xs:element name="contextAttribute">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="qualityRequirement"/>
      </xs:sequence>
      <xs:attribute name="name" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="contextEntity">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="contextAttribute" minOccurs="0" max-
Occurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="name" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="contextModel">
    <xs:complexType>
      <xs:choice maxOccurs="unbounded">
        <xs:element ref="contextSpace"/>
        <xs:element ref="contextRelation"/>
        <xs:element ref="contextEntity"/>
      </xs:choice>
    </xs:complexType>
  </xs:element>
  <xs:element name="contextRelation">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="relatedEntities"/>
```

```
      <xs:element ref="contextAttribute" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="name" type="xs:string" use="required"/>
  </xs:complexType>
 </xs:element>
 <xs:element name="features">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="isContextAdaptive"/>
    </xs:sequence>
  </xs:complexType>
 </xs:element>
 <xs:element name="function">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="features"/>
      <xs:element ref="contextModel"/>
    </xs:sequence>
    <xs:attribute name="name" type="xs:string" use="required"/>
    <xs:attribute name="alternative" type="xs:string"/>
  </xs:complexType>
 </xs:element>
 <xs:element name="isContextAdaptive" type="xs:boolean"/>
 <xs:element name="qualityRequirement">
 <xs:complexType>
    <xs:sequence>
      <xs:element name="value"/>
    </xs:sequence>
    <xs:attribute name="name" type="xs:string" use="required"/>
  </xs:complexType>
 </xs:element>
 <xs:element name="relatedEntities">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="contextEntity" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
 </xs:element>
 <xs:element name="contextSpace">
  <xs:complexType>
    <xs:attribute name="name" type="xs:string" use="required"/>
  </xs:complexType>
 </xs:element>
</xs:schema>
```

Das Schemaelement „function“ bildet die Klammer um die Beschreibung der Einzelfunktion. Das Attribut „name“ referenziert den Namen der Einzelfunktion und das Attribut „alternative“ kennzeichnet, dass es sich um eine alternative Umsetzung der Einzelfunktion handelt. Die enthaltenen Schemaelemente sind „features“ und „contextModel“.

Das Schemaelement „features“ ermöglicht eine Beschreibung der Eigenschaften der Einzelfunktion. Dieses kann das Schemaelement „isContextAdaptive“ beinhalten. „isContextAdaptive“ zeigt an, ob die Funktion kontextadaptiv ist.

Das Schemaelement „contextModel“ umschließt die Beschreibung des für die Einzelfunktion benötigten Kontextmodells. Es beinhaltet optional die Schemaelement „contextSpace“, „contextEntity“ oder „contextRelation“.

Das Schemaelement „contextSpace“ definiert den Raum, welcher als Bestandteil des Kontextmodells der Einzelfunktionen benötigt wird. Das Attribut „name“ referenziert hier den Namen des Kontextraums, der in dem Kontext-Referenzmodell definiert ist.

Das Schemaelement „contextEntity“ beschreibt die Kontextentität, dessen Attribut im Kontextmodell der Einzelfunktion benötigt wird. Das Attribut „name“ referenziert an dieser Stelle den Namen der Kontextentität, welche in dem Kontext-Referenzmodell definiert ist. Es beinhaltet „contextAttribute“.

Das Schemaelement „contextRelation“ beschreibt die Relation, deren Kontextattribut im Kontextmodell der Einzelfunktion benötigt wird. Das Attribut „name“ referenziert dabei den Namen der Kontextrelation, die in dem Kontext-Referenzmodell definiert ist. Es beinhaltet die Schemaelemente „relatedEntities“ und „contextAttribute“, wobei „relatedEntities“ eine Aufzählung der für eine Relation relevanten Kontextentitäten ermöglicht.

Das Schemaelement „contextAttribute“ beschreibt das Attribut, welches als Bestandteil des Kontextmodells für die Einzelfunktion benötigt wird. Das Attribut „name“ referenziert jetzt den Namen des Kontextattributs, welches in dem Kontext-Referenzmodell definiert ist und beinhaltet das Schemaelement „qualityRequirement“, welches die Qualitätsanforderungen an die Kontextinformation beschreibt. Das „name“ referenziert auch den zugehörigen Qualitätsparameter, der im Kontext-Referenzmodell zu dem Attribut definiert ist. Im Schemaelement „value“ wird der entsprechende Messwert der Qualitätsanforderung eingetragen.

Für die Erstellung einer solchen Dienstebeschreibung kann die Aufzählung der Funktionen im Pflichtenheft des AAL-Dienstes herangezogen werden. Ein Beispiel für die Beschreibung der Kontextadaptivität der Einzelfunktionen „Überwachung des Medikamentenvorrats“ wird nachfolgend gegeben.

```
<function name="ueberwacheMedikamentenVorrat">
  <features>
    <isContextAdaptive>True</isContextAdaptive>
  </features>
  <contextModel>
    <contextEntity name="Medikament">
      <contextAttribute name="istVorhanden">
        <qualityRequirement name="Abweichung">
          <value>+-2</value>
        </qualityRequirement>
      </contextAttribute>
    </contextEntity>
  </contextModel>
</function>
```

6.10 Zusammenfassung

In diesem Kapitel wurde zunächst der Einsatzzweck der Kontextmodellierung auf der Ebene der Dienste beschrieben. Darauf aufbauend wurden die Anforderungen identifiziert, die dieser Ebene zuzuordnen sind und die entsprechenden Konzepte erarbeitet, die zur Umsetzung der Anforderungen benötigt werden. Darauf aufsetzend wurde ein Metamodell definiert, das die für die Erstellung konkreter Kontextmodelle auf dieser Ebene brauchbaren Modellelemente festlegt. Weiterhin wurden die Überführungsfunktionen identifiziert, mit denen sich die Modellelemente auf die Ebene der Infrastruktur abbilden lassen. Anhand des Metamodells wurden danach die für die einzelnen Phasen der Entwicklung und Nutzung der Kontextinfrastruktur benötigten Modelltypen definiert und zum Teil anhand von Beispielen verdeutlicht.

7 Kontextmodellierung auf der Ebene der Nutzerinteraktion

Auf der Basis der zweidimensionalen Kontextmodellierung wird nachfolgend die Ebene der Nutzerinteraktion im Detail betrachtet und konzipiert. Zunächst wird der Einsatzzweck dieser Ebene beschrieben. Anschließend werden die Anforderungen, die dieser Ebene zuzuordnen sind, zusammengefasst und dann die notwendigen Konzepte zur Umsetzung dieser Anforderungen dargestellt. Basierend auf diesen Konzepten wird das in dieser Ebene benötigte Metamodell und dessen Abbildung in die Ebene der Dienste definiert sowie abschließend die vier Modelltypen für die jeweiligen Phasen beschrieben.

7.1 Einsatzzweck

Diese Ebene dient der Kommunikation mit dem Bewohner. In unterschiedlichen Anwendungsfällen müssen ihm einerseits Kontextinformationen dargestellt und andererseits auch von ihm beschrieben werden (Kap. 3.4). Sie muss in einfacher und verständlicher Form geschehen. Dieser Aspekt der Kontextinfrastruktur und der darauf aufsetzenden AAL-Dienste ist spezieller Gegenstand dieser Betrachtung.

7.2 Anforderungen

Im Kapitel 3 wurden bereits, unabhängig von der zweidimensionalen Betrachtung der Kontextmodellierung, die bestehenden Anforderungen aus Sicht des AAL identifiziert. Zunächst werden auf der Grundlage des Einsatzzwecks die dieser Ebene zuzuordnenden Anwendungsfälle identifiziert und dann die entsprechenden Anforderungen zusammengefasst.

Folgende Anwendungsfälle lassen sich aus dem Einsatzzweck ableiten:

- Darstellung von Kontextinformationen. Sie werden zur Kontrolle über die intelligente Umgebung und der Auswahl von AAL-Diensten benötigt (siehe Kap. 3.4).

- Festlegung des kontextadaptiven Verhaltens eines AAL-Dienstes.

Der Bewohner muss in der Lage sein, die Annahmen der intelligenten Umgebung über den Kontext nachzuvollziehen und sie gegebenenfalls zu korrigieren. Hierbei spielen die Kriterien der Ergonomie eine wesentliche Rolle. Dazu müssen die relevanten Kontextinformationen, die Bestandteil der Annahmen der intelligenten Umgebung sind, in einer für den Bewohner geeigneten Art

dargestellt werden (A7.2.1.1). Weiterhin muss er die Möglichkeit haben, diese Annahmen in einfacher Art und Weise zu korrigieren (A7.2.1.2).

Die Auswahl neuer AAL-Dienste für die häusliche Umgebung setzt voraus, dass der Bewohner über die Art der benötigten Kontextinformationen und eventuelle Einschränkungen oder vorgeschlagene Erweiterungen der Infrastruktur informiert wird. So sollte er erkennen können, an welchen Kontextinformationen der AAL-Dienst interessiert ist, um eventuelle Zugriffsregeln zu definieren (A7.2.2.1). Auch muss ersichtlich sein, ob diese Kontextinformationen von der verfügbaren häuslichen Umgebungssensorik geliefert werden können (A7.2.2.2). Der Bewohner muss die Möglichkeit haben, die Kontextadaptivität von Diensten zu konfigurieren und beispielsweise zu bestimmen, dass ein Komfort-Dienst ihn beim Aufstehen mit seiner Lieblingsmusik beliefert. Hierfür muss er wissen, welches kontextadaptive Verhalten variabel ist (A7.2.3.1), um bei Bedarf die Konfiguration der Kontextadaptivität verändern zu können (A7.2.3.2). Diese Art der Darstellung und der Modellierung von Kontextinformationen muss für einen Bewohner ohne spezielle Voraussetzungen verständlich und praktikabel sein.

7.3 Konzepte

Ausgehend von den identifizierten Anforderungen werden nun die auf dieser Ebene relevanten Konzepte vorgestellt. Sie gehen unmittelbar auf das nachfolgend vorgestellte Metamodell über. Folgende Konzepte liegen der Kontextmodellierung auf der Ebene der Nutzerinteraktion zugrunde:

- Nutzerinteraktion zur Kontrolle der intelligenten Umgebung
- Nutzerinteraktion zur Auswahl von AAL-Diensten
- Konfiguration des kontextadaptiven Verhaltens eines AAL-Dienstes

Diese Konzepte werden nachfolgend im Detail beschrieben.

7.3.1 Nutzerinteraktion zur Kontrolle der intelligenten Umgebung

Die Kontrolle über die Umgebung im AAL kann für den Bewohner folgendes bedeuten:

- Kontrolle über die in der intelligenten Umgebung verfügbaren AAL-Dienste: Der Bewohner muss die Möglichkeit besitzen, selbst zu bestimmen, welche Dienste in seiner Umgebung verfügbar sind, um die Dienstemenge bei Bedarf erweitern oder einschränken zu können. Hierzu ist in der intelligenten Umgebung dem Nutzer eine Übersicht

über die verfügbaren Dienste zur Verfügung zu stellen und bei Bedarf die Möglichkeit zu geben, Dienste aus dem Portfolio zu entfernen.

- Kontrolle über einen einzelnen AAL-Dienst: Der Bewohner muss einen einzelnen AAL-Dienst kontrollieren können. Dazu gehört es, ihn gegebenenfalls zu deaktivieren bzw. wieder zu aktivieren.

- Kontrolle über die Kontextadaptivität der intelligenten Umgebung: Aufgrund der Unzuverlässigkeit der Kontexterkennung ist es notwendig, die Annahmen des Systems über den Bewohner und seine Umgebung sowie das daraus resultierende Verhalten darzustellen. Der Bewohner soll erkennen können, welches kontextadaptive Verhalten ausgelöst wurde und welche Annahmen dazu führten. Bei Bedarf soll das Verhalten korrigiert werden können.

Es gibt keine bekannten Arbeiten die sich mit dem Aspekt der Kontrolle über die intelligente Umgebung befassen. Somit sind keine Konzepte zur Umsetzung dieser Anforderungen bekannt, auf denen das Kontext-Metamodell aufsetzen kann. Daher wird nachfolgend ein initiales Konzept dargestellt. Dieses basiert auf Annahmen über vom Bewohner benötigten Informationen aus dem Kontextmodell und der geeignete Darstellung. Zudem basiert das Konzept auf den grundlegenden Kriterien der Ergonomie. Eine Validierung dieses Konzepts steht jedoch noch aus.

Bezüglich der Kontextadaptivität der intelligenten Umgebung sollte der Bewohner Antworten zu folgenden Fragen bekommen können:

- Was ist der Grund dafür, dass eine Einzelfunktion in einem AAL-Dienst ausgelöst wurde, obwohl aus Sicht des Bewohners dieses nicht erfolgen dürfte?

- Was ist der Grund dafür, dass eine Einzelfunktion in einem AAL-Dienst nicht ausgelöst wurde, obwohl aus Sicht des Bewohners dieses hätte erfolgen sollen?

Zu diesem Zweck sollten dem Betroffenen zwei unterschiedliche Sichten auf die AAL-Dienste und deren Einzelfunktionen zur Verfügung gestellt werden. In einer ersten Sicht sollte eine Auflistung der Einzelfunktionen erfolgen, die aufgrund von Kontextereignissen ausgelöst werden. Ein Eintrag soll dabei folgende Inhalte erfassen:

- Ausgelöste Einzelfunktion

- AAL-Dienst, der die Einzelfunktion zugehörig ist

- Zeitpunkt der Auslösung

- Grund für die Auslösung, d.h. die erfüllten Kontextbedingungen, die zur Auslösung geführt haben

Der Grund für die Auslösung kann durch den Ausschnitt aus dem Kontextmodell, der zu dem kontextadaptiven Verhalten geführt hat, visualisiert werden. Für die Visualisierung gelten die Anforderungen aus der Ergonomie in Bezug auf die Selbstbeschreibung, Erwartungskonformität und Eignung für Wahrnehmung und Verständnis. Eine graphische Repräsentation des Kontextmodells ist hierfür eine geeignete Präsentationsform. Das konzeptuelle Kontextmodell aus Kap. 6.7 ist hierfür jedoch in der Form nicht geeignet, da es zu viele Modellelemente besitzt, die für das Verständnis der auslösenden Kontextbedingungen durch den Bewohner nicht notwendig sind. Die wesentlichen für den Bewohner zu verstehenden Konzepte sind:

- Welches Objekt ist Gegenstand der Beobachtung?
- Welche Beziehung zwischen Objekten ist Gegenstand der Beobachtung?
- Welche Eigenschaften des Objekts oder der Beziehung sind relevant für das Auslösen der kontextadaptiven Eigenschaft des AAL-Dienstes?
- Welche Situation führt zum Auslösen der kontextadaptiven Eigenschaft des AAL-Dienstes?

Daher werden in einer Vereinfachung nur die Modellelemente „ContextEntity“, „ContextRelation“, „ContextAttribute“ und „ContextSpace“, sowie einfache Assoziationen zwischen diesen benötigt. Auf diese Weise kann ein Ausschnitt aus dem Kontextmodell visualisiert werden, der zur Auslösung der Einzelfunktion geführt hat. Die Darstellung der konkreten Kontextentitäten, -relationen und -räume kann durch geeignete Icons im Metamodell ergänzt werden, z.B. Symbol für eine Person. Die folgende Abbildung zeigt ein Beispiel für eine solche Darstellung.

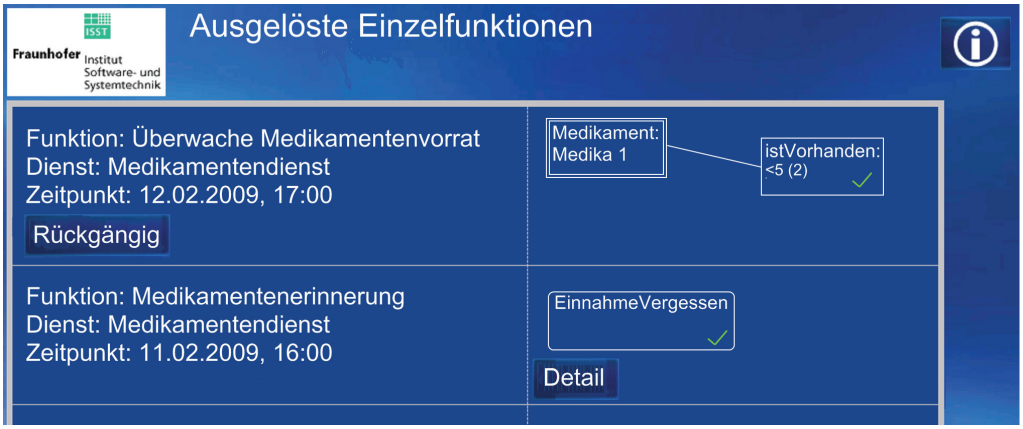


Abbildung 55: Darstellung der ausgelösten Einzelfunktionen

In einer zweiten Sicht soll dem Bewohner ermöglicht werden, eine Einzelfunktion auszuwählen und zu erkennen, warum diese nicht ausgelöst wird. Deshalb sollen zu einer Einzelfunktion die Kontextbedingungen angezeigt werden, die zum Auslösen der Funktion erfüllt sein müssen. Die Darstellung des relevanten Ausschnitts aus dem Kontextmodell entspricht dabei der oben beschriebenen graphischen Repräsentation. Hierbei sollen nicht erfüllte Kontextbedingungen durch ein entsprechendes Symbol gekennzeichnet werden. Die folgende Abbildung zeigt ein Beispiel einer solchen Darstellung.

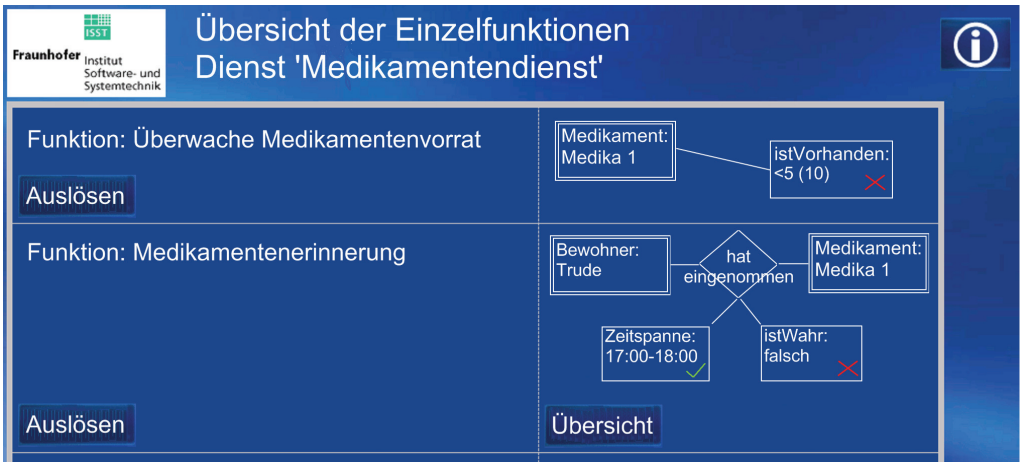


Abbildung 56: Darstellung der Kontextbedingungen zu den Einzelfunktionen

7.3.2 Nutzerinteraktion zur Auswahl von AAL-Diensten

Die Darstellung der kontextadaptiven Fähigkeiten der intelligenten Umgebung ist für den Bewohner insbesondere bei der Auswahl von AAL-Diensten notwendig. Aufgrund fehlender Sensorik kann es möglich sein, dass die benötigten Ausschnitte aus den jeweiligen dienstetyp-spezifischen Kontextmodellen nicht unterstützt werden und daher Einzelfunktionen eines AAL-Dienstes für den Bewohner nicht verfügbar sind. Dieses sollte den Nutzern deutlich gemacht werden oder zumindest erkennbar sein, wie die Kontextinfrastruktur erweitert werden kann, um die fehlende Unterstützung zu erlangen.

Es gibt keine bekannten Arbeiten die sich mit der Visualisierung der kontextadaptiven Fähigkeiten der intelligenten Umgebung befassen. Somit sind keine Konzepte zur Umsetzung dieser Anforderungen bekannt, auf denen das Kontext-Metamodell aufsetzen kann. Daher wird nachfolgend ein initiales Konzept dargestellt. Dieses basiert auf Annahmen über vom Bewohner benötigten Informationen aus dem Kontextmodell und der geeignete Darstellung. Zudem basiert das Konzept auf den grundlegenden Kriterien der Ergonomie. Eine Validierung dieses Konzepts steht jedoch noch aus.

Das Konzept zur Darstellung der Fähigkeiten baut auf der im vorherigen Kapitel beschriebene Vereinfachung des konzeptuellen Kontextmodells auf. Sie muss noch um folgende Aussagen zur Unterstützung des geforderten Kontextmodells angereichert werden:

- Kontextinformationen werden in der geforderten Qualität geliefert.
- Kontextinformationen werden geliefert, jedoch nicht in der geforderten Qualität.
- Kontextinformationen werden nicht geliefert.

Diese Aussagen können durch entsprechende Symbole, beispielsweise durch eine Ampel, in der graphischen Repräsentation der Modellelemente „ContextAttribute“ und „ContextSpace“ dargestellt werden.

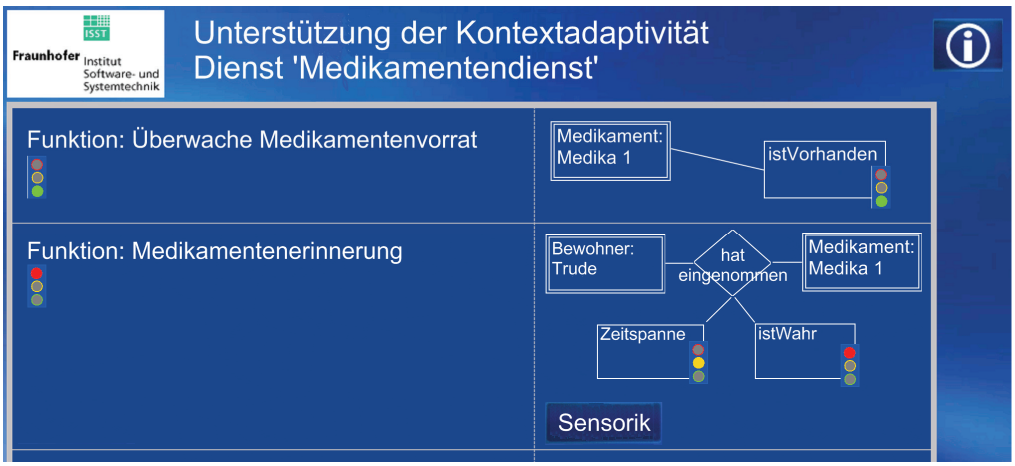


Abbildung 57: Darstellung der kontextadaptiven Unterstützung für eine Einzelfunktion

7.3.3 Konfiguration des kontextadaptiven Verhaltens eines AAL-Dienstes

Die vorhergehenden Konzepte waren beschränkt auf die bewohnerbezogene Visualisierung eines Kontextmodells. Der jetzt zu behandelnde Anwendungsfall geht über die Visualisierung hinaus und betrachtet die Interaktion des Bewohners auf dem Kontextmodell. Der Nutzer muss in der Lage sein, das kontextadaptive Verhalten seiner AAL-Dienste zu definieren, wenn es in einer Form erfolgt, die den oben erwähnten ergonomischen Anforderungen entspricht.

Für diesen Anwendungsfall gibt es einige wenige in der Literatur veröffentlichte Ansätze, die nachfolgend beschrieben und bewertet werden. In [125] wird ein Ansatz vorgestellt, in welchem der Bewohner über eine grafische Oberfläche ECA-Regeln zu Kontextereignissen definieren kann. Anind K. Dey beschreibt in [177] mit „a CAPpella“ einen Ansatz des „Programming by Demonstration“. Hierbei soll der Nutzer das gewünschte kontextadaptive Verhalten der intelligenten Umgebung durch eine Reihe von Nutzungsbeispielen antrainieren können. Beide Ansätze sind aus Sicht der Ergonomie unbefriedigend. Daher wurde ein eigener Ansatz erarbeitet [7] [9], der es dem Bewohner ermöglicht, das kontextadaptive Verhalten durch Auswahl und Spezialisierung vordefinierter Situationsbeschreibungen zu definieren. Diese drei Ansätze werden nachfolgend dargestellt und bewertet.

7.3.3.1 Event-Condition-Action Programmierung

Bei diesem Ansatz wird die kontextadaptive Anwendungslogik eines Dienstes über eine Reihe von Event-Condition-Action-Regeln (ECA) definiert. Dieser ist in Kap. 4.1.5 im Detail beschrieben. Der Bewohner hat hierbei die Möglichkeit über einen graphischen Regeleditor das kontextadaptive Verhalten über die Verknüpfung von Sensorsymbolen mit Symbolen für boolsche Operatoren zu definieren.

Die Autoren des Ansatzes ließen als Beweis für die Brauchbarkeit der grafischen Repräsentation eines ECA-basierten Metamodells ein definiertes Szenario durch 15 Informatikstudenten umsetzen. Die Studenten waren in der Lage, dieses weitestgehend fehlerfrei und ohne Unterstützung anzuwenden. Ob das auch durch technikfremde Benutzer gelingt, wollten die Autoren als nächstes prüfen, haben darüber aber noch nicht berichtet. Die Anwendbarkeit des Ansatzes für einen technikfremden Bewohner ist somit noch nicht bewiesen. Sicherlich ist die Puzzle-Metapher nach einer kurzen Einweisung gut zu erlernen, weil sie intuitives Vorgehen ermöglicht und damit die Erlernbarkeit, die Wahrnehmung und das Verständnis fördert. Allerdings kann die Erstellung von boolschen Regeln für einen unbedarften Bewohner trotz graphischer Nutzeroberfläche eine hohe Hürde darstellen. Solche Regeln können sehr komplex und dann, trotz einer grafischen Repräsentation, unübersichtlich und schwer verständlich sein. Auch ist die fehlende Abstraktion der Kontextbeschreibung von der zugrundeliegenden Sensorik problematisch. In dem Beispiel ist die Ortung einer Person mittels RFID explizit in der grafischen Repräsentation der Events sichtbar. Aufgrund der fehlenden technischen Abstraktion von der Sensorik und der Notwendigkeit der Definition eines komplexen Regelwerks durch den Bewohner erfüllt dieser Ansatz die Anforderungen an die Kontextmodellierung aus dem AAL nicht.

7.3.3.2 A CAPpella: Programming by Demonstration

Mit „a CAPpella“ [177] wird ein Ansatz vorgestellt, welcher auf dem Paradigma Programming by Demonstration basiert. Die Autoren begründen ihren Ansatz so, dass ein Endnutzer, der bereits Probleme hat, ein Videogerät zu programmieren, auch nicht in der Lage sein wird, ein Regelwerk für das kontextadaptive Verhalten von Diensten aufzustellen. Daher ist ihrer Meinung nach der zuvor beschriebene ECA-Ansatz nicht für den Endnutzer geeignet. Sie schlagen vor, dass der Nutzer kontextadaptives Verhalten nicht explizit über ein Regelwerk definieren muss, sondern die Anwendung dieses über eine Reihe von Beispielen erlernen kann. Die konzeptuelle Grundlage hierfür sind Mechanismen maschinellen Lernens, das zum Beispiel über die Verwendung von Hidden-Markov-Modellen erfolgt und das ein Nutzer antrainieren kann, um relevante Situationen zu erkennen.

Das Training bedient sich einer grafischen Benutzerschnittstelle. Beispiele werden in der Trainingsphase mit Hilfe einer Kamera aufgezeichnet, wobei Kontextinformationen auch von weiteren Sensoren, wie Mikrofon oder RFID, stammen und anschließend dem Benutzer vorgelegt. Dieser kann die Aufzeichnung beliebig oft abspielen und dem System mitteilen, welche der aufgezeichneten Kontextinformationen in der betrachteten Situation relevant sind und welche Anwendungsreaktion in dieser Situation gewünscht ist. Mit der Anzahl der Trainingseinheiten steigt auch die Genauigkeit des erlernten Regelwerks.

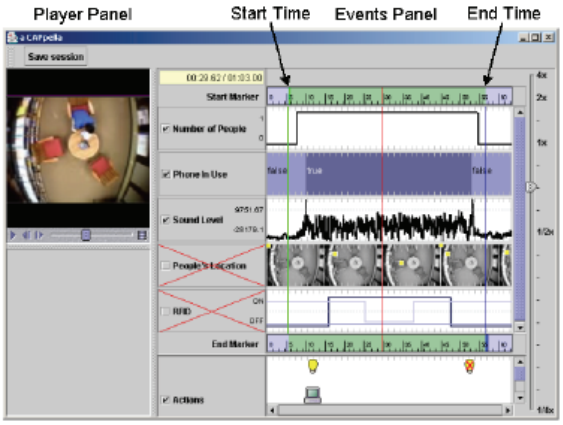


Abbildung 58: Nutzerschnittstelle zum Antrainieren des kontextadaptiven Verhaltens

Das Metamodell für diesen Ansatz definiert das Modellelement „Event“ zur Beschreibung von Kontextereignissen. Es gibt allerdings keine Zuordnung dieses Elements zu Kontextentitäten und deren Attributen, die allein in die zugrundeliegenden Modellelemente des maschinellen Lernens eingehen.

In einer Studie haben die Autoren die Eignung dieses Ansatzes getestet, indem sie 14 Versuchspersonen ohne informationstechnischem Hintergrund beauftragten, ein vorgegebenes Szenario umzusetzen. Diese wurden nur kurz in die Grundfunktionen des Systems eingewiesen und dann ohne weiteres Feedback bei der Bewältigung der Aufgabenstellung beobachtet. Die Versuchspersonen gaben an, dass das Antrainieren sehr einfach zu handhaben war. Damit glauben die Autoren, die Eignung des Ansatzes bewiesen zu haben. Die Versuchsergebnisse zeigten jedoch gleichzeitig, dass das antrainierte kontextadaptive Verhalten nur eine geringe Zuverlässigkeit besaß. Die Genauigkeit der Situationserkennung schwankte zwischen 50 und 78,6 Prozent, was nach den Autoren daran lag, dass die Anzahl der Trainingsdurchläufe nicht ausreichend war.

Dieser Ansatz unterstützt die Definition des kontextadaptiven Verhaltens durch den Bewohner, wobei sonstige Anwendungsfälle unberücksichtigt bleiben. Die

Verwendbarkeit dieses Ansatzes für einen technikfremden Nutzer ist trotz der Studie nicht erwiesen. Sie hat gezeigt, dass die Form der Nutzerinteraktion der Zielgruppe angemessen d.h. lernförderlich und geeignet für die Wahrnehmung und das Verständnis war. Problematisch hingegen ist der Ansatz in Hinblick auf die Fehlertoleranz. Falsche Trainingsinformationen führen zu einer fehlerhaften Ausprägung des gelernten Wissens, welches erst durch eine Vielzahl korrekter Trainingseinheiten wieder kompensiert werden kann. Ein weiteres Problem besteht hier in Richtung der Erwartungskonformität. Der Bewohner muss das gewünschte kontextadaptive Verhalten durch eine Vielzahl von Trainingseinheiten in das System überführen. Dabei kann er nicht sicher sein, ob denn nach der Trainingseinheit das gewünschte Verhalten tatsächlich zuverlässig erkannt wird. Neben der Anzahl der Trainingseinheiten ist hierfür auch das zugrundeliegende Kontextmodell für die Zuverlässigkeit entscheidend. Hängt das Verhalten von einem Kontextaspekt ab, welches nicht Teil des Kontextmodells ist, so wird auch ein wiederholtes Training nicht zum gewünschten Verhalten führen. Dieser Zusammenhang ist dem normalen Benutzer jedoch nicht bewusst. So ist die Brauchbarkeit dieses Ansatzes im AAL stark davon abhängig, ob ein universelles Kontextmodell samt Sensorik definiert werden kann, welches jedes erdenkliche kontextadaptive Nutzerverhalten abbilden kann, was jedoch ohne eine Einschränkung auf fest definierte Szenarien nicht möglich ist.

7.3.3.3 Auswahl und Spezialisierung vordefinierter Situationsbeschreibungen

Der im Rahmen dieser Dissertation erarbeitete Ansatz abstrahiert von den technischen Details der Kontexterkennung und der Definition komplexer Regelwerke und vereinfacht auf diese Weise die Kontextmodellierung für einen Endnutzer.

Diese Kontextmodellierung durch den Endanwender basiert auf der Auswahl und Spezialisierung vordefinierter Situationsbeschreibungen, indem eine Situationstaxonomie Bestandteil des Kontextmodells auf der Ebene der Nutzerinteraktion ist. Jedes ihrer Elemente besitzt eine grafische Repräsentation und eine textuelle Beschreibung. Dabei kann zwischen abstrakten und konkreten Elementen der Situationstaxonomie unterschieden werden. Die abstrakten werden nur zur thematischen Einordnung von Situationsbeschreibungen innerhalb der Taxonomie benötigt. Die konkreten können dagegen auch zur Konfiguration des kontextadaptiven Verhaltens eines AAL-Dienstes genutzt werden. Die Beschreibung einer solchen Situationstaxonomie muss zusätzlich Bestandteil des Metamodells auf Ebene der Nutzerinteraktion sein. Nachfolgend wird dafür ein Beispiel gegeben.

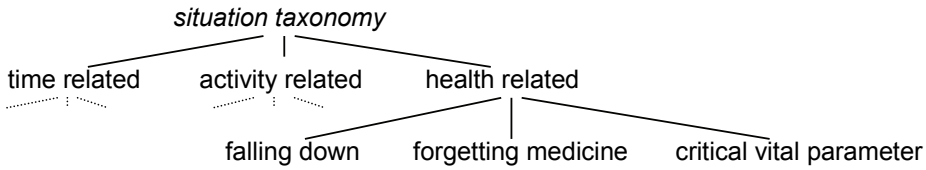


Abbildung 59: Beispiel einer Situationstaxonomie

Die Situationstaxonomie kann vom Bewohner verwendet werden, um das kontextadaptive Verhalten eines AAL-Dienstes zu konfigurieren. Es erfolgt in drei Teilschritten. Zunächst wählt der Bewohner eine Einzelfunktion des Dienstes, deren Kontextadaptivität er festlegen möchte, wobei auf der Nutzeroberfläche die obersten Elemente der Situationstaxonomie angezeigt werden, die von der Kontextinfrastruktur unterstützt und zur Konfiguration verwendet werden können.

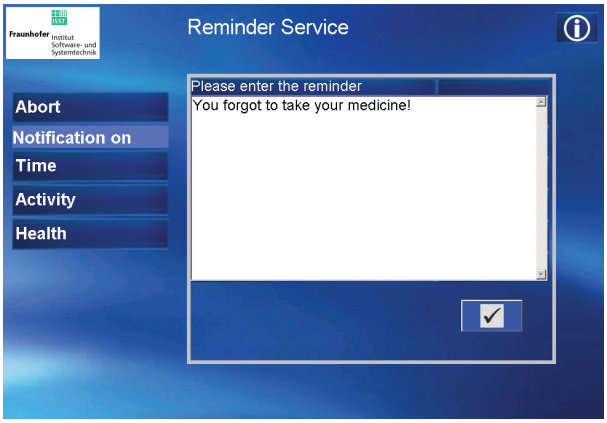


Abbildung 60: Schritt 1 - Einstieg in die Situationstaxonomie

In einem zweiten Schritt hat der Bewohner die Möglichkeit, innerhalb der darunter befindlichen Taxonomieelemente zu navigieren. Dabei werden ihm eine textuelle Beschreibung der gewählten Situation und die zugehörige grafische Repräsentation angezeigt. Über die Auswahl eines abstrakten Elements der Situationstaxonomie werden dem Bewohner die darin eingeordneten weiteren vordefinierten Situationen dargestellt.

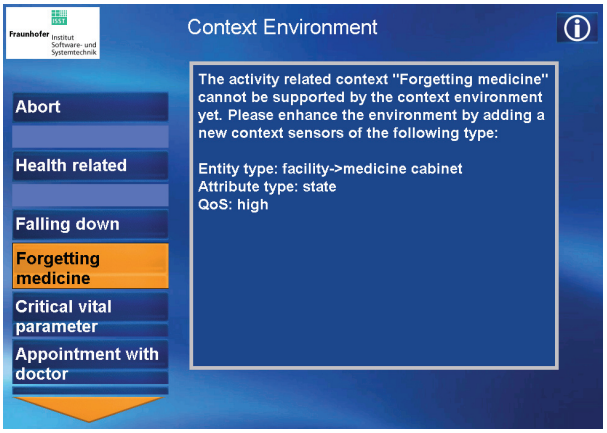


Abbildung 61: Schritt 2 - Navigation durch die Taxonomie und Auswahl

In einem dritten Schritt hat der Nutzer nun die Möglichkeit, eine Verfeinerung der vordefinierten Situationsbeschreibung vorzunehmen. Die Verfeinerung erfolgt auf den in der Definition der Situation enthaltenen Modellelementen „ContextEntity“, „ContextRelation“ und „ContextAttribut“. Dazu kann der Bewohner eine der vorgegebenen Verfeinerungsmöglichkeiten auswählen und in einem entsprechenden Dialog die Einschränkung vornehmen. Es können vier grundsätzliche Möglichkeiten der Verfeinerung angeboten werden. Diese werden nachfolgend anhand eines Beispiels erläutert. Für dieses Beispiel sei eine Situation „Treffen mit Personen“ definiert, welche durch die Kontextentitäten „Person“ und „Raum“, sowie die Kontextrelation „<Person> istIn <Raum>“ gekennzeichnet ist. Ist mehr als eine Person im selben Raum enthalten, so gilt diese Situation für alle im Raum befindlichen Personen.

- Selektion einer konkreten Kontextentität: Zu einer in der Situationsbeschreibung enthaltenen Kontextentität kann der Nutzer eine konkrete Instanz als weitere Einschränkung der Situation auswählen. Eine Liste aller bekannten Instanzen wird dem Bewohner zur Auswahl zur Verfügung gestellt. Ein Beispiel hierfür ist die Auswahl von „Franz Meier“ in der Kontextentität „Person“.

- Selektion eines spezialisierten Entitätstyps: Sind im Kontextmodell zu einer Kontextentität Spezialisierungen definiert, so kann der Nutzer eine solche als weitere Einschränkung der Situation auswählen. Eine Liste der spezialisierenden Entitätsdefinitionen wird dem Bewohner zur Auswahl zur Verfügung gestellt. Ein Beispiel hierfür ist die Auswahl der Kontextentität „Betreuer“ als Spezialisierung der Kontextentität „Person“.

- Einschränkung der Anzahl von Kontextentitäten: Ist in einer Situationsbeschreibung eine Kontextrelation enthalten, so kann der Bewohner auf den damit verbundenen Kontextentitäten eine Einschränkung bezogen auf die Anzahl der konkreten Instanzen festlegen. Ein Beispiel hierfür ist die Einschränkung der Anzahl der Personen auf „>3“ in der Kontextrelation „Person ist in Raum“.

- Einschränkung auf einem Kontextattribut: Auf den in der Situationsbeschreibung enthaltenen Kontextattributen kann der Bewohner Einschränkungen im Wertebereich vornehmen. Ihre Art ist abhängig von der zugrundeliegenden Kontextdimension, aber auch davon ob es sich hierbei um einen Punkt oder einen Bereich innerhalb der Dimension handelt. Ein Beispiel hierfür ist die Begrenzung der Zeit auf „>18:00:00“ Uhr.

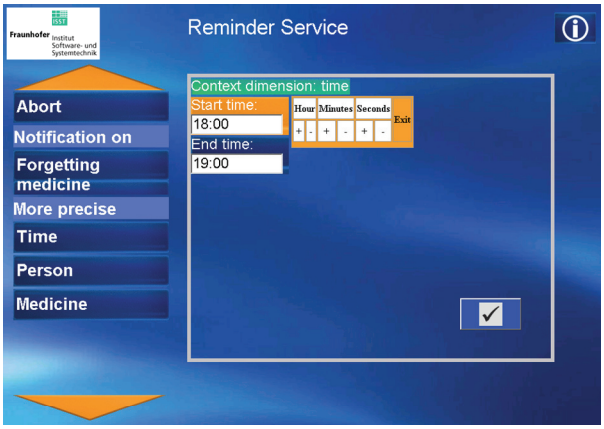


Abbildung 62: Schritt 3 - Weitere Verfeinerung der Situationsbeschreibung

Dieser Ansatz ermöglicht dem Bewohner, das kontextadaptive Verhalten eines Dienstes auf einfache Weise festzulegen. Dabei ist die Ausdrucksmächtigkeit der Kontextmodellierung für den Bewohner auf die durch die Situationstaxonomie und der Spezialisierungsmöglichkeiten gegebenen Vorgaben beschränkt. Diese Beschränkung wurde bewusst eingesetzt, um den Umgang mit der Kontextmodellierung für den Bewohner zu vereinfachen.

Zur Umsetzung dieser Art der Kontextmodellierung muss das Kontextmodell auf der Ebene der Nutzerinteraktion Informationen über die Situationstaxonomie sowie die dem Bewohner anzubietende Spezialisierungsmöglichkeiten beinhalten. Weiterhin müssen Informationen über die Darstellung der Modellelemente enthalten sein, z.B. für die grafische Darstellung.

Dieser Ansatz erfüllt wesentliche Anforderungen aus Sicht der Ergonomie. Durch die vordefinierte Taxonomie wird dem Bewohner unmittelbar das poten-

tielle kontextadaptive Verhalten eines Dienstes ersichtlich. Eine fehlerhafte Konfiguration kann unmittelbar durch eine neue Beschreibung der Kontextadaptivität korrigiert werden. Es sind keine die Fehldefinition überdeckende Trainingsläufe wie in „a CAPpella“ notwendig. Eine Individualisierung ist in diesem Ansatz nicht vorgesehen. Die Möglichkeit der Exploration der Situationstaxonomie ist in Hinblick auf die Lernförderung gegeben. Die vereinfachende und abstrahierende Form der Kontextmodellierung in diesem Ansatz ist für die Verbesserung der Wahrnehmung und des Verständnisses geeignet.

Zur Bewertung der Einsetzbarkeit dieses Ansatzes und zur Validierung der obigen Aussagen wurde am Fraunhofer ISST ein Test mit 12 wissenschaftlichen Mitarbeitern durchgeführt, denen eine webbasierte Nutzeroberfläche für einen AAL-Erinnerungsdienst zur Verfügung gestellt wurde. In ihr wurde der oben beschriebene Ansatz zur Interaktion auf dem Kontextmodell umgesetzt. Es wurde ein Versuchsaufbau definiert, in welchem in vorgegebener Reihenfolge vier Aufgaben durch die Tester zu lösen waren. Jede dieser Aufgaben hatte eine besondere Zielstellung, die nachfolgend beschrieben wird.

- Es sollte die Erinnerung an einen Arzttermin festgelegt werden. Die Situation „Arzttermin“ war Bestandteil der Situationstaxonomie. Die Aufgabe konnte dementsprechend einfach durch die Auswahl des Eintrags in der Situationstaxonomie gelöst werden. Anhand dieser Aufgabe sollte festgestellt werden, ob das Konzept der Konfiguration mittels der Situationstaxonomie intuitiv verstanden wurde und zur Definition des gewünschten kontextadaptiven Verhaltens verwendet werden konnte.

- Es sollte eine Erinnerung an die Einnahme eines bestimmten Medikaments festgelegt werden. Die Situation „Einnahme Medikament vergessen“ war Bestandteil der Situationstaxonomie. Nach Auswahl des Eintrags in der Taxonomie musste noch in einem zweiten Schritt eine Verfeinerung durch die Auswahl eines konkreten Medikaments vorgenommen werden (Spezialisierung der Kontextentität „Medikament“). Anhand dieser Aufgabe sollte festgestellt werden, ob das Konzept der Verfeinerung der Situationstaxonomie intuitiv verstanden wurde und zur Definition des gewünschten kontextadaptiven Verhaltens verwendet werden konnte.

- Es sollte eine Erinnerung an ein Treffen mit einem Freund festgelegt werden. Eine solche Situation war jedoch nicht Bestandteil der Situationstaxonomie und konnte vom Benutzer nicht gefunden werden. Das Ziel der Aufgabe war herauszufinden, ob der Anwender intuitiv verstand, warum das geforderte kontextadaptive Verhalten nicht definiert werden konnte.

- In einer vierten Aufgabe sollte eine Situation als Auslöser für eine Erinnerung festgelegt werden, die Bestandteil der Situationstaxonomie war.

Jedoch wurde diese nicht von der zugrundeliegenden Sensorik unterstützt. So war die Situationsbeschreibung Bestandteil der Taxonomie. Diese war jedoch nicht auswählbar und mit einer entsprechenden Hilfestellung versehen. Das Ziel der Aufgabe war es herauszufinden, ob der Anwender verstand, warum das geforderte kontextadaptive Verhalten nicht definiert werden konnte.

Zu jedem dieser Aufgaben musste der Fragebogen ausgefüllt werden, der aus den nachfolgend beschriebenen Einzelfragen bestand. Der Aufbau der Fragen diente dazu die Eignung des Konzepts entsprechend den folgenden Anforderungen zur Definition des kontextadaptiven Verhaltens eines AAL-Dienstes zu validieren. Diese Anforderungen und die zugrundeliegenden Kriterien der Ergonomie sind in Kap. 3.4 beschrieben.

- AM7.2.3.1: Darstellung des möglichen kontextadaptiven Verhaltens: erwartungskonform, selbstbeschreibend, lernförderlich und geeignet für die Wahrnehmung und das Verständnis.

- AM7.2.3.2: Festlegung des kontextadaptiven Verhaltens: aufgabenangemessen, erwartungskonform, selbstbeschreibend, fehlertolerant, lernförderlich und geeignet für die Wahrnehmung und das Verständnis.

Folgende Fragen mit der entsprechenden Zielstellung sind Bestandteil des Fragebogens.

- „Das Prinzip der Kontextmodellierung ist einfach zu verstehen“: Diese Fragestellung zielt auf die Bewertung der „Selbstbeschreibungsfähigkeit“ und „Eignung für die Wahrnehmung und das Verständnis“ des Konzepts durch den Endanwender. Konkret sind damit die ausreichende Abstraktion des Kontextmodells und die Verständlichkeit der darin enthaltenen Modellelemente verbunden.

- „Es ist einfach das kontextadaptive Verhalten des Dienstes wie gewünscht zu konfigurieren“: Diese Fragestellung zielt auf das Kriterium der „Aufgabenangemessenheit“ für die Festlegung des kontextadaptiven Verhaltens. Konkret ist damit die Eignung der Situationstaxonomie und der Verfeinerungen zur Gestaltung einfacher Dialoge und kurzer Interaktionswege verbunden.

- „Die Art der Kontextmodellierung entspricht meinen Erwartungen“: Diese Fragestellung zielt auf das Kriterium der „Erwartungskonformität“ für die Festlegung des kontextadaptiven Verhaltens. Diese ist verbunden mit den bereits bestehenden Erfahrungen des Endanwenders mit vergleichbaren Aufgabenstellungen oder Technologien und den ihm bekannten Konzepten.

- „Ich brauchte Hilfestellung durch eine weitere Person, um die Konfiguration durchzuführen“: Diese Fragestellung zielt auf das Kriterium der „Selbstbeschreibungsfähigkeit“ für die Festlegung des kontextadaptiven Verhaltens. Es sollte herausgefunden werden ob das Konzept intuitiv und ohne Schulung oder externe Hilfestellung verstanden wird.

- „Ich überblicke, welche Einstellungen ich bezüglich der Kontextadaptivität des Dienstes vornehmen kann“: Diese Fragestellung zielt auf das Kriterium „Eignung für Wahrnehmung und Verständnis“ für die Festlegung des kontextadaptiven Verhaltens. Ein Teilaspekt hierbei ist die Erkennung der Konfigurationsmöglichkeiten die das System dem Endanwender vorgibt. Konkret ist damit die Eignung der Situationstaxonomie und der Verfeinerungen zur Beschreibung der verfügbaren Konfigurationsmöglichkeiten verbunden.

- „Ich verstehe, welches kontextadaptive Verhalten nicht eingestellt werden kann“: Diese Fragestellung zielt auf das Kriterium „Eignung für Wahrnehmung und Verständnis“ für die Festlegung des kontextadaptiven Verhaltens. Ein Teilaspekt hierbei ist die Erkennung der vom System vorgegebenen Grenzen für die Konfiguration durch den Endanwender. Konkret ist damit die Eignung der Situationstaxonomie und der Verfeinerungen zur Beschreibung von nicht enthaltenen Konfigurationsmöglichkeiten verbunden.

- „Die Art der Kontextmodellierung ist geeignet, um das mögliche kontextadaptive Verhalten zu erforschen“: Diese Fragestellung zielt auf das Kriterium „Lernförderlichkeit“ für die Darstellung und Festlegung des kontextadaptiven Verhaltens. Konkret ist damit die Eignung der Aufteilung des möglichen kontextadaptiven Verhaltens in die vordefinierte Situationstaxonomie und die darauf anschließenden Verfeinerungsmöglichkeiten zur Exploration verbunden.

- „Die Art der Kontextmodellierung ist geeignet, um fehlerhafte Eingaben zu vermeiden“: Diese Fragestellung zielt auf das Kriterium der „Fehlertoleranz“ für die Festlegung des kontextadaptiven Verhaltens. Konkret ist damit die Eignung der Situationstaxonomie und der Verfeinerungen zur Vermeidung von ungewollten Konfigurationen verbunden. Diese können entweder aus dem Mißverständnis der zugrundeliegenden Konzepte oder der Art der Dialogführung entstehen.

- „Ich habe das Gefühl, dass ich über diesen Ansatz das kontextadaptive Verhalten des Dienstes kontrollieren kann“: Wie in [125] festgestellt, ist eine wesentliche Voraussetzung für die Akzeptanz kontextadaptiver AAL-Lösungen die Beibehaltung der Kontrolle beim Bewohner über die technische Umgebung. Neben den technischen Möglichkeiten spielt

hier auch die gefühlte Einschätzung durch den Bewohner eine Rolle, welche mit dieser Fragestellung bewertet werden soll.

Es wurde zu Beginn keine Schulung oder Einarbeitung gegeben, sondern die Testpersonen wurden direkt gebeten, die Aufgaben auszuführen und anschließend einen Fragebogen auszufüllen. Diese mussten mit einer Einschätzung zwischen 1 (volle Zustimmung) bis 5 (volle Ablehnung) beantwortet werden. Die nachfolgende Tabelle gibt die durchschnittliche Wertung der Fragen durch die Testteilnehmer wider.

Tabelle 17: Ergebnis des Labortests zur Einsetzbarkeit

Frage	∅ Wertung
Ist diese Art der Kontextmodellierung einfach zu verstehen?	1,5
Ist es einfach, das kontextadaptive Verhalten des Erinnerungsdienstes zu definieren?	2,5
Entspricht die Art der Kontextmodellierung Ihrer Erwartung?	2,2
Brauchten Sie Hilfe, um die Konfiguration durchzuführen?	5
War es für Sie ersichtlich, welches kontextadaptive Verhalten konfiguriert werden kann?	2
War es für Sie ersichtlich, welches kontextadaptive Verhalten nicht konfiguriert werden kann?	2,5
Ist diese Art der Modellierung geeignet, um das mögliche kontextadaptive Verhalten eines Dienstes zu erkunden?	2,2
Hatten Sie das Gefühl, dass Sie mit diesem Ansatz das kontextadaptive Verhalten des Dienstes kontrollieren können?	1,8
Ist diese Art der Kontextmodellierung geeignet, um Fehler zu vermeiden?	2,3

Als Ergebnis haben alle Testpersonen bestätigt, dass dieser Ansatz zur Kontextmodellierung einfach zu verstehen ist. Dennoch sind die weiteren Bewertungen nicht so befriedigend wie erwartet. Beispielsweise waren einige der Testpersonen nicht in der Lage, die Beschränkungen in der Modellierung der

Kontextadaptivität einzuschätzen. Folgende Gründe können für das nicht optimale Ergebnis gefunden werden:

- Ein Großteil der Testpersonen hatte noch keine Erfahrung mit dem Thema „Kontextadaptivität“ und mit der speziellen Modellierung von Kontexten.

- Es gab keinerlei Vorbereitung, Schulung oder Einweisung in den Umgang mit der Nutzeroberfläche.

- Teilweise haben technische Probleme (JavaScript) mit unterschiedlichen Browsern den Umgang mit der Nutzeroberfläche erschwert. Diese Probleme sind in die Bewertung des generellen Konzepts eingegangen.

- Die Situationstaxonomie war nicht sorgfältig genug ausgewählt. Einige Testpersonen haben angemerkt, dass sie einen Eintrag an einer anderen Stelle erwartet hätten.

Trotz dieser Schwierigkeiten wurden alle Aufgaben ohne Hilfestellung gelöst.

Das Ergebnis des Labortests kann als lediglich als initialer Indikator für die Eignung des Konzepts angesehen werden. Belastbare Aussagen können daraus nicht abgeleitet werden. Die Gründe hierfür betreffen die Auswahl und Vorbereitung der Testpersonen, die Definition des Kontextmodells, sowie die Effekte aus den technischen Rahmenbedingungen. Folgende Maßnahmen müssen für die Erstellung eines aussagekräftigen Versuchsaufbaus getroffen werden.

- Die Anzahl von 12 Testpersonen kann generell als ausreichend angesehen werden, wenn die objektive Bewertung durch die Probanden störende Einflüsse ausgeschlossen werden können. Dazu gehört die Bereitstellung einer einheitlichen technischen Basis für alle Probanden, sowie deren Einführung in den Versuchsablauf und die Erläuterung des Fragebogens. Dadurch können beispielsweise Missverständnisse beim Verständnis der Aufgabenstellung und der Fragen vermieden werden.

- Die Auswahl der Testpersonen muss die Zielgruppe der AAL-Dienste umfassen. Dazu gehören ältere Personen, die bereits heute solche Dienste verwenden können und bei denen kein technisches Vorauswissen zu Modellierungstechniken und dem Umgang mit dem Internet angenommen werden kann.

- Die Situationstaxonomie, die Verfeinerungsmöglichkeiten, sowie die grafische Repräsentation der Modellelemente muss sorgfältig erarbeitet werden, um inhaltliche Missverständnisse zu vermeiden. Der Umfang

und die konkrete Ausgestaltung der Situationstaxonomie sollten dabei bereits einem Dienstetyp, z.B. „health“ entsprechen.

- Es muss ausgeschlossen werden, dass technische Probleme den Versuchsablauf beeinträchtigen. Dieses ist durch die Bereitstellung einer definierten Systemumgebung und der Durchführung von Tests auf dieser Umgebung sichzustellen.

- Es muss ausgeschlossen werden, dass die Art der Umsetzung des Konzepts in Form der Nutzungsoberfläche den Versuchsablauf beeinträchtigt. Hier sind die generellen Prinzipien der Softwareergonomie beim Design zu beachten, die auf die Zielgruppe auszurichten sind. Dazu gehören beispielsweise die Verwendung geeigneter Schriftgrößen und Farben, sowie die geeignete Strukturierung der Oberfläche. Zudem sollte ein geeignetes Interaktionskonzept erstellt und umgesetzt werden. Dazu gehören beispielsweise Rückmeldungen auf getätigte Eingaben oder Hilfestellungen.

- Es muss ausgeschlossen werden, dass die verwendete Technologie eine Barriere zur Nutzung des Konzepts darstellt. Das Konzept der Konfiguration des kontextadaptiven Verhaltens ist unabhängig davon, ob dieses in einer Browser-basierten Anwendung umgesetzt wird oder einer anderen Technologie. Für die Zielgruppe bietet sich aktuell die Umsetzung des Versuchaufbaus für eine Settop-Box an. Der Auswahl dieser Technologie liegt die Annahme zugrunde, dass für die Zielgruppe aktuell der Zugang über einen Fernseher und die Konfiguration mit Hilfe einer Fernbedienung am geeignetesten für die Nutzung von AAL-Diensten ist.

Der Versuchsaufbau bestehend aus den vier Aufgabenstellungen repräsentiert wesentliche Nutzungssituationen der Kontextmodellierung für den Endanwneder und kann daher in der bestehenden Form übernommen werden. Es können sicherlich noch weitere Situationen identifiziert werden, z.B. die Änderung einer bestehenden Konfiguration des kontextadaptiven Verhaltens. Jedoch sollte der Versuchsaufbau auf möglichst wenige relevante Situationen beschränkt werden, um nicht den Umfang und damit die Dauer unnötig zu verlängern. Der Fragenkatalog kann ebenfalls in der bestehenden Form übernommen werden. Dieser kann eventuell um ein Freifeld zur Rückmeldung beliebiger Kommentare erweitert werden. Eine erneute Durchführung der Tests in der überarbeiteten Form ist jedoch im Rahmen dieser Dissertation nicht vorgesehen.

7.4 Abbildungen von Modellelementen zur Ebene der Dienste

In Kap. 4.2.1 wird die Trennung zwischen den drei unterschiedlichen Ebenen „Infrastruktur“, „Dienst“ und ,“Nutzerinteraktion“ in der Dimension „Einsatzzweck“ als wesentliches Konzept der Kontextmodellierung beschrieben. Auf Ebene der Dienste werden die dienstetyp-spezifischen Kontextmodelle für die Umsetzung des kontextadaptiven Verhaltens eines AAL-Dienstes bereitgestellt. Auf Ebene der Nutzerinteraktion werden die für den Bewohner geeigneten Sichten auf das Kontextmodell bereitgestellt. Eine Abbildung zwischen den Modellelementen der Ebene der Interaktion und der Ebene der Dienste ist in beide Richtungen notwendig.

Für die Darstellung der Kontextinformationen aus der Ebene der Dienste müssen diese entsprechend den Darstellungskonzepten (Kap. 7.3) in die Ebene der Interaktion überführt werden. Die hier darzustellenden Elemente des Metamodells sind „ContextEntity“, „ContextRelation“, „ContextAttribute“ und „ContextSpace“. Auf Ebene der Nutzerinteraktion sind diese durch ein Icon und eine textuelle Beschreibung zu ergänzen. Das Modellelement „ContextSpace“ entspricht hier einer Situationsbeschreibung.

Nach der erfolgten Konfiguration des kontextadaptiven Verhaltens eines AAL-Dienstes durch den Bewohner müssen diese Modellinformationen zur Umsetzung des Verhaltens in die Ebene der Dienste überführt werden. Die Konfiguration erfolgt dabei durch Auswahl einer Situation und die Beschreibung einer Verfeinerung. Dazu werden die Modellelemente „Situation“ und die Spezialisierungen von „SituationConstraint“, sowie die zugehörigen Modellelemente „ContextEntity“, „ContextRelation“ und „ContextAttribute“ verwendet. Bei der Überführung in die Ebene der Dienste führt dieses zu einer spezialisierten Instanz des Modellelements „ContextSpace“.

7.5 Metamodell

Nachfolgend werden die Elemente des Metamodells auf Ebene der Nutzerinteraktion beschrieben. Dieses Metamodell ist die Grundlage für alle Modelltypen bzw. Sichten auf dieser Ebene. Je nach Modelltyp besitzen die entsprechenden Sichten einen mehr oder weniger eingeschränkten Umfang des Metamodells. Die Definition des Metamodells erfolgt nachfolgend durch Erweiterung des UML-Metamodells in einem Paket „Context.Interaction“. Für die Definition grundlegender Eigenschaften von Modellelementen des Kontext-Metamodells wird auf die grundlegenden Modellierungskonzepte von UML zugegriffen, die dort im Paket „Kernel“ definiert sind. Für die Beschreibung der Abbildung des Kontextmodells auf Ebene der Nutzerinteraktion auf das Kontextmodell auf Ebene der Dienste wird zudem eine Referenzierung auf das Paket „Context.Application“ benötigt. Dieses wird in der nachfolgenden Abbildung dargestellt.

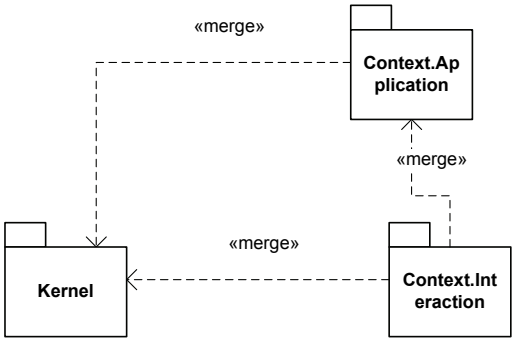


Abbildung 63: Aufteilung des Metamodells in Pakete

In dem Paket „Context.Interaction“ werden die Elemente des Metamodells definiert, die auf der Ebene der Nutzerinteraktion für die Kontextmodellierung benötigt werden. Die notwendigen Modellelemente wurden in der Konzeption identifiziert: „ContextEntity“, „ContextRelation“, „ContextAttribute“, „Situation“. Weiterhin werden Modellelemente zur Beschreibung der Spezialisierungsmöglichkeiten benötigt: „SelectionConstraint“, „SpecializationConstraint“, „CardinalityConstraint“ und „AttributeConstraint“. Die hier aufgezählten Modellelemente müssen für die Überführung auf die Ebene der Dienste dem entsprechenden Dienstetyp zugeordnet werden. Hierfür wird das Modellelement „AssociatedModel“ definiert.

Die identifizierten Modellelemente und deren Beziehungen zueinander werden in dem nachfolgenden Klassendiagramm gezeigt und anschließend in alphabetischer Reihenfolge detailliert beschrieben.

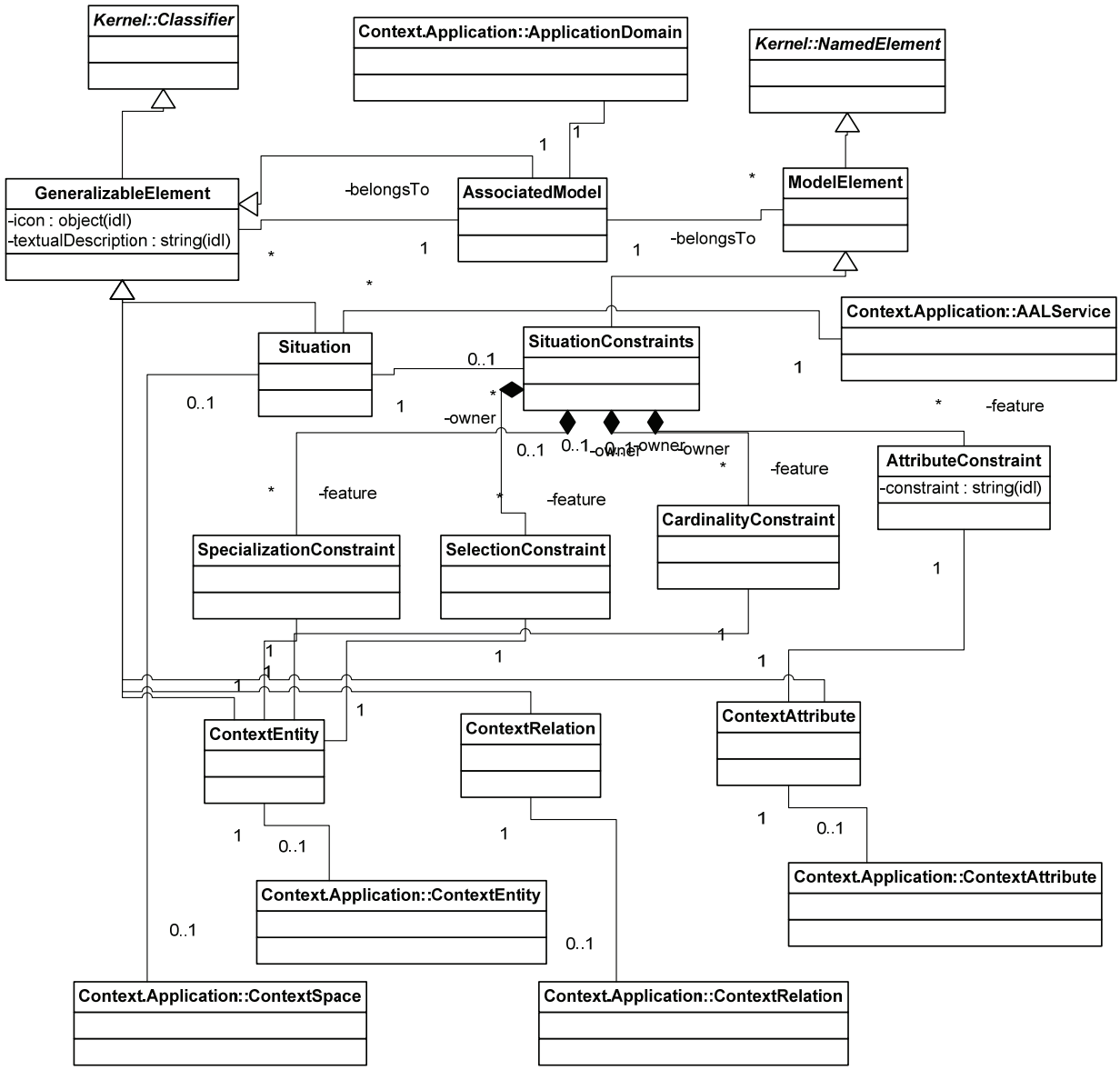


Abbildung 64: Modellelemente im Package „Context.Interaction“

7.5.1 AssociatedModel

Generalisierungen

- „GeneralizableElement“

Beschreibung

„AssociatedModel“ ist ein Modellelement, welches die Zuordnung des Kontextmodells auf Ebene der Interaktion zu einem Kontextmodell auf der Ebene der Dienste beschreibt.

Attribute

Dieses Modellelement besitzt keine eigenen Attribute.

Beziehungen

- Ein „AssociatedModel“ ist einem dienstetyp-spezifischen Kontextmodell zugeordnet („Context.Application::ApplicationDomain“).
- Die generalisierbaren Modellelemente sind dem „AssociatedModel“ zugeordnet.
- Die nicht generalisierbaren Modellelemente sind dem „AssociatedModel“ zugeordnet.

7.5.2 AttributeConstraint

Generalisierungen

Dieses Modellelement besitzt keine weitere Generalisierung.

Beschreibung

„AttributeConstraint“ ist ein Modellelement, welches die Spezialisierung einer Situationsbeschreibung durch den Bewohner ermöglicht. Diese Spezialisierung besteht in der Definition einer Einschränkung auf einem Kontextattribut. Mit ihm muss die Zuordnung zu einem Kontextattribut beschrieben werden.

Attribute

- constraint : string

  Beschreibung der Einschränkung auf dem Kontextattribut.

Beziehungen

- Ein „AttributeConstraint“ ist einer „SituationConstraints“ zugehörig.
- Ein „AttributeConstraint“ ist einem Kontextattribut zugeordnet.

7.5.3 CardinalityConstraint

Generalisierungen

Dieses Modellelement besitzt keine weitere Generalisierung.

Beschreibung

„CardinalityConstraint“ ist ein Modellelement, welches die Spezialisierung einer Situationsbeschreibung durch den Bewohner ermöglicht. Diese Spezialisierung besteht in der Definition einer Einschränkung der Kardinalität von Instanzen einer Kontextentität innerhalb einer Kontextrelation. Mit ihm muss eine Zuordnung zu einer Kontextentität innerhalb einer Kontextrelation beschrieben werden.

Attribute

Dieses Modellelement besitzt keine eigenen Attribute.

Beziehungen

- Ein „CardinalityConstraint“ ist einer „SituationConstraints“ zugehörig.
- Ein „CardinalityConstraint“ ist einer Kontextentity zugeordnet.

7.5.4 ContextAttribute

Generalisierungen

- „GeneralizableElement“

Beschreibung

Das Modellelement „ContextAttribute“ beschreibt den beobachtbaren Zustand und die Eigenschaft einer Kontextentität oder –relation, die dem Benutzer dargestellt und zur Interaktion zur Verfügung gestellt werden können. Es kann je nach Anwendungsfall für unterschiedliche Zwecke genutzt werden, beispielsweise zur Beschreibung der weiteren Einschränkungen auf einer Situationsbe-

schreibung oder zur Darstellung der Möglichkeiten und Annahmen der Kontextinfrastruktur.

Attribute

Dieses Modellelement besitzt keine eigenen Attribute.

Beziehungen

- Dieses Modellelement ist einem Kontextattribut auf Ebene der Dienste („Context.Application::ContextAttribute“) zugeordnet.
- Dieses Modellelement kann von einem „AttributeConstraint“ referenziert werden.

7.5.5 ContextEntity

Generalisierungen

- „GeneralizableElement“

Beschreibung

Das Modellelement „ContextEntity“ repräsentiert die Kontextentitäten, die im Kontextmodell definiert werden können, z.B. Personen, Räume oder Gegenstände. Es dient der Visualisierung einer Kontextentität beim Nutzer und kann zur weiteren Einschränkung einer Situationsdefinition verwendet werden.

Attribute

Dieses Modellelement besitzt keine eigenen Attribute.

Beziehungen

- Dieses Modellelement ist einer Kontextentität auf Ebene der Dienste („Context.Application::ContextEntity“) zugeordnet.
- Dieses Modellelement kann von einem „CardinalityConstraint“ verwendet werden.
- Dieses Modellelement kann von einem „SelectionConstraint“ verwendet werden.
- Dieses Modellelement kann von einem „SpecializationConstraint“ verwendet werden.

7.5.6 ContextRelation

Generalisierungen

- „GeneralizableElement“

Beschreibung

Das Modellelement „ContextRelation“ repräsentiert die Kontextrelationen, die im Kontextmodell definiert werden können. Es dient der Visualisierung einer Kontextrelation beim Bewohner.

Attribute

Dieses Modellelement besitzt keine eigenen Attribute.

Beziehungen

- Dieses Modellelement ist einer Kontextrelation auf Ebene der Dienste („Context.Application::ContextRelation“) zugeordnet.

7.5.7 GeneralizableElement

Generalisierungen

- „Classifier (from Kernel)“

Beschreibung

Ein „GeneralizableElement“ beschreibt alle Modellelemente deren Instanzen in einer Generalisierungs- oder Spezialisierungshierarchie zueinander in Beziehung stehen können. Zudem darf eine Instanz dieses Modellelements abstrakt sein, d.h. selbst nicht instantiiert werden. Dieses gilt für alle Modellelemente des Metamodells, die von „GeneralizableElement“ abgeleitet sind und somit dessen Eigenschaften erben. Auf der Ebene der Interaktion ist dieses beispielsweise das Modellelement „ContextEntity“. „GeneralizableElement“ ist eine abstrakte Metaklasse. Diese Eigenschaft erbt es von der Klasse „Classifier“ im Package „Kernel“ des UML-Metamodells. Die abgeleiteten Modellelemente besitzen zusätzlich eine grafische und eine textuelle Repräsentation und sind einem „AssociatedModel“ zugeordnet.

Attribute

- icon : object

  Eine grafische Repräsentation des Modellelements

- textualDescription : string

  Eine textuelle Beschreibung des Modellelements

Beziehungen

- Ein generalisierbares Modellelement ist einem „AssociatedModel“ zugeordnet.

7.5.8 ModelElement

Generalisierungen

- „NamedElement (from Kernel)“

Beschreibung

Das „ModelElement“ bildet die Basis aller nicht generalisierbaren Modellelemente des Metamodells auf Ebene der Interaktion. Jedes weitere ist von dieser abgeleitet und kann eine Beschreibung besitzen. Sie ist abgeleitet von der Klasse „NamedElement“ aus dem Package „Kernel“ des UML-Metamodells und erbt von dort die Zuordnung eines im Namensraum eindeutigen Namens.

Attribute

Dieses Modellelement besitzt keine eigenen Attribute.

Beziehungen

- Ein nicht generalisierbares Modellelement ist einem „AssociatedModel“ zugeordnet.

7.5.9 SelectionConstraint

Generalisierungen

Dieses Modellelement besitzt keine weitere Generalisierung.

Beschreibung

„SelectionConstraint“ ist ein Modellelement, welches die Spezialisierung einer Situationsbeschreibung durch den Bewohner ermöglicht. Sie besteht in der Auswahl der konkreten Instanz einer Kontextentität. Mit dem Modellelement muss eine Zuordnung zu einer Kontextentität beschrieben werden.

Attribute

Dieses Modellelement besitzt keine eigenen Attribute.

Beziehungen

- Ein „SelectionConstraint“ ist einer „SituationConstraints“ zugehörig.
- Ein „SelectionConstraint“ ist einer Kontextentity zugeordnet.

7.5.10 Situation

Generalisierungen

- „GeneralizableElement“

Beschreibung

Das generalisierbare Modellelement „Situation“ dient der Definition einer Situationstaxonomie, die dem Bewohner zur Visualisierung und Konfiguration des möglichen kontextadaptiven Verhaltens einer Einzelfunktion bei einem AAL-Dienst bereitgestellt werden kann.

Attribute

Dieses Modellelement besitzt keine eigenen Attribute.

Beziehungen

- Dieses Modellelement ist einem Kontextraum auf Ebene der Dienste („Context.Application::ContextSpace“) zugeordnet.

- Diesem Modellelement können kein, ein oder mehrere „SituationConstraints“ zugehörig sein.

- Eine „Situation“ kann einem AAL-Dienst als Konfigurationsmöglichkeit zugeordnet werden.

7.5.11 SituationConstraints

Generalisierungen

- „ModelElement“

Beschreibung

„SituationConstraints“ ist ein Modellelement, welches die Spezialisierungsmöglichkeiten einer Situationsbeschreibung durch den Bewohner definiert. Mögliche Einschränkungen können durch die zugehörigen Modellelemente „SpecializationConstraint“, „SelectionConstraint“, „CardinalityConstraint“ und „AttributeConstraint“ definiert werden.

Attribute

Dieses Modellelement besitzt keine eigenen Attribute.

Beziehungen

- Eine „SituationConstraints“ ist einer „Situation“ zugehörig.

- Einer „SituationConstraints“ können folgende Modellelemente zugehörig sein: „SpecializationConstraint“, „SelectionConstraint“, „CardinalityConstraint“ und „AttributeConstraint“.

7.5.12 SpecializationConstraint

Generalisierungen

Dieses Modellelement besitzt keine weitere Generalisierung.

Beschreibung

„SpecializationConstraint“ ist ein Modellelement, das die Spezialisierung einer Situationsbeschreibung durch den Bewohner ermöglicht. Sie besteht in der Auswahl einer spezialisierten Kontextentität. Mit ihm muss eine Zuordnung zu einer Kontextentität beschrieben werden.

Attribute

Dieses Modellelement besitzt keine eigenen Attribute.

Beziehungen

- Ein „SpecializationConstraint“ ist einer „SituationConstraints“ zugehörig.
- Ein „Specialization Constraint“ ist einer Kontextentity zugeordnet.

7.6 Informelle Kontextbeschreibung

Der Aspekt der Nutzerinteraktion ist ein Bestandteil der Kontextinfrastruktur oder eines AAL-Dienstes. Die für die Kontextinfrastruktur notwendigen Konzepte sind in den Kap. 7.3.1 und 0 beschrieben und müssen von dieser umgesetzt werden. Für die AAL-Dienste wird im Rahmen eines Pflichtenhefts beschrieben, welches kontextadaptive Verhalten einer Funktion durch den Bewohner definiert werden kann (Kap. 6.6.2). In dieser Phase werden die Anforderungen an die Kontextmodellierung noch nicht zwischen der Ebene der Dienste und der Nutzerinteraktion getrennt betrachtet. Das Pflichtenheft ist daher auch die Grundlage für die konzeptuelle Kontextmodellierung auf der Ebene der Nutzerinteraktion. In der Testphase muss keine Unterscheidung gemacht werden, ob das kontextadaptive Verhalten durch den Bewohner konfiguriert werden kann oder nicht. Hier sind lediglich die zu untersuchenden Testfälle aus der Situationstaxonomie und den Spezialisierungsmöglichkeiten für eine Funktion abzuleiten und auf der Ebene der Dienste durchzuführen. Tests zur Nutzerinteraktion betreffen nur die Qualität der Situationstaxonomie und brauchen deshalb keine eigene informelle Kontextbeschreibung.

7.7 Konzeptuelles Kontextmodell

Aus der Beschreibung des nutzerkonfigurierbaren kontextadaptiven Verhaltens im Pflichtenheft geht in der Phase des Designs eine Konzeption der Situationstaxonomie und der möglichen Spezialisierungen hervor. Dabei muss die Situationstaxonomie sorgfältig erarbeitet werden, um dem Nutzer ein schnelles und sicheres Auffinden der gewünschten Situationsbeschreibung zu ermöglichen. Hierfür wird nachfolgend eine Erweiterung der konzeptuellen Kontextmodellierung (Kap. 5.6.3, Kap. 6.7) um die hierfür benötigten grafischen Elemente beschrieben.

Auf Ebene der Dienste beschreibt das Modellelement „ContextSpace“ die relevanten Kontextentitäten, -relationen und –attribute. Die dort verwendete grafische Notation mit der Umrandung der darin enthaltenen Modellelemente ist für die Darstellung der Situationstaxonomie nicht geeignet. In der Situationstaxo-

nomie soll nicht der Aufbau eines korrespondieren Kontextraums beschrieben werden. Daher wird das Modellelement „Situation“ durch ein eigenes Symbol dargestellt. Hierfür wird ein abgerundetes Rechteck verwendet. Zur Unterscheidung zwischen abstrakten und konkreten Elementen der Situationstaxonomie wird analog zur Definition von Kontextentitäten eine abstrakte Situation durch eine kursiv dargestellte Bezeichnung der Situation dargestellt.

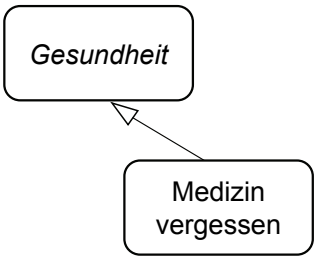


Abbildung 65: Definition einer Situationstaxonomie

Die Zuordnung einer konkreten Situation zu einem Kontextraum muss ebenfalls dargestellt werden, sowie die möglichen Spezialisierungen der Situationsbeschreibung. Zu diesem Zweck wird eine Assoziation zwischen der Situation und dem zugehörigen Kontextraum mit Hilfe der entsprechenden UML-Notation beschrieben. Die Beschreibung des Kontextraums wird dann durch Symbole ergänzt, welche die Möglichkeit der weiteren Einschränkung darstellen. Eine solche wird durch einen Kreis symbolisiert mit einem Buchstaben, welcher auf die Art der Einschränkung hinweist: „A“ttributeConstraint, „S“electionConstraint, „Sp“ecializationConstraint und „C“ardinalityConstraint. Diese Symbole werden den Modellelementen zugeordnet, an denen sie eingesetzt werden können. Die Verwendung dieser Symbole wird in der nachfolgenden Abbildung dargestellt.

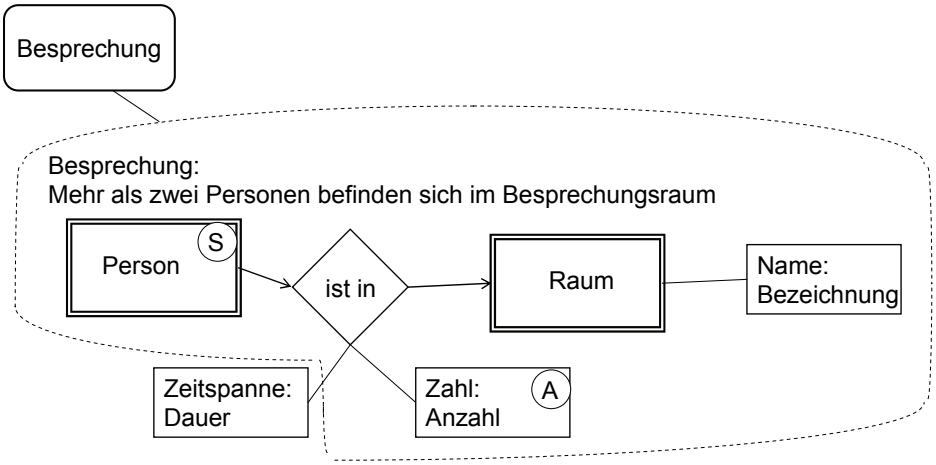


Abbildung 66: Definition der Einschränkungsmöglichkeiten

Diese Notation ist nicht konform zu den Visualisierungsmöglichkeiten der UML-Stereotypes. Analog zur Darstellung der Kontexträume ist auch hierfür keine geeignete alternative Notation zu finden.

7.8 Operatives Kontextmodell

Das operative Kontextmodell ist die lauffähige Umsetzung des konzeptuellen Kontextmodells. Zu dessen Realisierung müssen die im konzeptuellen Kontextmodell beschriebenen Modellelemente verfeinert und vervollständigt werden. Diese werden nachfolgend beschrieben. Abschließend wird eine implementationsneutrale Repräsentation des operativen Kontextmodells beschrieben.

7.8.1 Ergänzungen zum konzeptuellen Kontextmodell

Folgende Ergänzungen und Verfeinerungen des konzeptuellen Kontextmodells müssen im operativen Kontextmodell vorgenommen werden.

AssociatedModel: Es wird eine grafische Repräsentation, sowie eine textuelle Beschreibung des zugehörigen Dienstetyps hinzugefügt. Zudem wird die zugehörige „ApplicationDomain“ auf Ebene der Dienste zugeordnet.

ContextEntity: Es wird eine grafische Repräsentation, sowie eine textuelle Beschreibung der Kontextentität hinzugefügt. Zudem wird diese der zugehörigen „ContextEntity“ auf Ebene der Dienste zugeordnet.

ContextRelation: Es wird eine grafische Repräsentation, sowie eine textuelle Beschreibung der Kontextrelation hinzugefügt. Zudem wird diese der zugehörigen „ContextRelation“ auf Ebene der Dienste zugeordnet.

ContextAttribute: Es wird eine grafische Repräsentation, sowie eine textuelle Beschreibung des Kontextattributs hinzugefügt. Zudem wird diese dem zugehörigen „ContextAttribute“ auf Ebene der Dienste zugeordnet.

Situation: Es wird eine grafische Repräsentation, sowie eine textuelle Beschreibung der Situation hinzugefügt. Ist die Situation nicht abstrakt, so wird diese dem zugehörigen „ContextSpace“ auf Ebene der Dienste zugeordnet.

7.8.2 Implementationsneutrale Repräsentation

Für die Definition der implementationsneutralen Repräsentation des operativen Kontextmodells auf Ebene der Nutzerinteraktion gilt dieselbe Motivation wie für das operative Kontextmodell auf Ebene der Infrastruktur (Kap. 5.7.2). Auch

hier wird die Überführung des Kontextmodells mit Hilfe von XMI in ein XML-Dokument vorgenommen. Die XMI-konforme Beschreibung des operativen Kontextmodells kann mit Hilfe eines Importmechanismus in den Kontextserver übernommen und in die implementationsspezifische Form überführt werden. Der grundsätzliche Import-Mechanismus ist Bestandteil des im Rahmen der Dissertation implementierten Kontextservers.

7.9 Deskriptives Kontextmodell

Das deskriptive Kontextmodell wird in der Phase der Bereitstellung und Übernahme von Bestandteilen der Infrastruktur in die häusliche intelligente Umgebung benötigt (Kap. 4.2.2). Folgende Bestandteile müssen auf Ebene der Nutzerinteraktion übernommen werden:

- AAL-Dienste: AAL-Dienste werden in die intelligente häusliche Umgebung eingebunden und erlauben die Konfiguration des kontextadaptiven Verhaltens durch den Bewohner-

Die Beschreibung des nutzerdefinierbaren kontextadaptiven Verhaltens der AAL-Dienste basiert auf dem Metamodell auf Ebene der Nutzerinteraktion. In der MOF Metadaten-Architektur ist eine konkrete Beschreibung auf Ebene M0 zu finden.

7.9.1 Dienstebeschreibung

Ergänzend zur Dienstebeschreibung in Kap. 6.9.1 müssen die Eigenschaften eines AAL-Dienstes in Bezug auf die Konfigurierbarkeit der Kontextadaptivität beschrieben werden. Folgende Elemente des Kontext-Metamodells sind dabei in geeigneter Form in der Dienstebeschreibung abzubilden:

- Situation: Die Konfigurierbarkeit des kontextadaptiven Verhaltens einer Einzelfunktion eines Dienstes kann durch die Zuordnung der Funktion zu den dabei auswählbaren Situationen beschrieben werden.

Die abstrakte Dienstebeschreibung wird daher wie folgt ergänzt. Zunächst wird in den Eigenschaften einer Einzelfunktion beschrieben, dass dessen kontextadaptives Verhalten durch den Bewohner konfiguriert werden kann. Dieses erfolgt durch ein Schemaelement „isUserDefinable“. Ist der Wert auf „true“ gesetzt, so kann das Verhalten durch den Bewohner konfiguriert werden. Ist dieses der Fall, dann wird die Beschreibung des Kontextmodells durch die Aufzählung der Situationen im Schemaelement „Situation“ ergänzt. Im Attribut „name“ des Schemaelements wird der Name der der Funktion zugeordneten Situation eingetragen. Nachfolgend wird der Ausschnitt aus der Schemadefini-

tion der Dienstebeschreibung mit den Ergänzungen (in fetter Schrift) dargestellt.

```
<xs:element name="features">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="isContextAdaptive"/>
      <xs:element ref="isUserDefinable"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="isUserDefinable" type="xs:boolean"/>

<xs:element name="contextModel">
  <xs:complexType>
    <xs:choice maxOccurs="unbounded">
      <xs:element ref="situation"/>
      <xs:element ref="contextSpace"/>
      <xs:element ref="contextRelation"/>
      <xs:element ref="contextEntity"/>
    </xs:choice>
  </xs:complexType>
</xs:element>
<xs:element name="situation">
  <xs:complexType>
    <xs:attribute name="name" type="xs:string"/>
  </xs:complexType>
</xs:element>
```

Ein Beispiel für die Beschreibung der Kontextadaptivität der Einzelfunktionen „Aufzeichnung Ereignis“ wird nachfolgend gegeben.

```
<FUNCTION name="Aufzeichnung Ereignis">
  <DESCRIPTION>Diese Funktion zeichnet ein Ereignis auf</DESCRIPTION>
  <FEATURES>
    <ISCONTEXTADAPTIVE>true</ISCONTEXTADAPTIVE>
    <ISUSERDEFINABLE>true</ISUSERDEFINABLE>
  </FEATURES>
  <CONTEXTMODEL>
    <SITUATION name="Aktivitäten"/>
    <SITUATION name="Einbruch"/>
  </CONTEXTMODEL>
</FUNCTION>
```

7.10 Zusammenfassung

In diesem Kapitel wurde zunächst der Einsatzzweck der Kontextmodellierung auf der Ebene der Nutzerinteraktion beschrieben. Darauf aufbauend wurden die Anforderungen identifiziert, die dieser Ebene zuzuordnen sind. Die zur Umsetzung der Anforderungen benötigten Konzepte wurden anschließend erarbeitet. Ein wesentliches Konzept ist hierbei die Konfiguration des kontextadaptiven Verhaltens eines AAL-Dienstes mittels einer vordefinierten Situationstaxonomie und möglichen Spezialisierungen. Darauf aufsetzend wurde ein Metamodell definiert, welches die für die Erstellung konkreter Kontextmodelle auf dieser Ebene benötigten Modellelemente festlegt. Weiterhin wurden die Überführungen zwischen den Modellelementen auf der Ebene der Dienste und der Nutzerinteraktion identifiziert. Anhand des Metamodells wurden danach die für die einzelnen Phasen der Entwicklung und Nutzung der Kontextinfrastruktur benötigten Modelltypen definiert und zum Teil anhand von Beispielen verdeutlicht.

8 Methodik der Kontextmodellierung zur Realisierung kontextadaptiver AAL-Dienste

In den Kapiteln 4 bis 7 wurde die zweidimensionale Kontextmodellierung für das AAL beschrieben. Die darin definierten Modelltypen und Modellebenen sowie die zugrundeliegenden Metamodelle bilden die Basis für die Definition und die Umsetzung der Kontextinfrastruktur und des kontextadaptiven Verhaltens von AAL-Diensten. Nachfolgend wird die Methodik der Kontextmodellierung zur Realisierung kontextadaptiver AAL-Dienste beschrieben. Zunächst werden die Werkzeuge identifiziert, die zur Umsetzung dieser Methodik benötigt werden, wobei ein Teil von ihnen schon im Rahmen dieser Arbeit umgesetzt wurde und somit direkt eingesetzt werden kann. Dann wird die Methodik unter Berücksichtigung verschiedener Einsatzszenarien vorgestellt, um abschließend die Anwendung am Beispiel der Realisierung eines AAL-Dienstes zu beschreiben.

8.1 Werkzeuge

Nachfolgend werden die Werkzeuge dargestellt, die zur Umsetzung der Methodik notwendig und in der folgenden Abbildung den beiden Dimensionen der Kontextmodellierung zugeordnet sind.

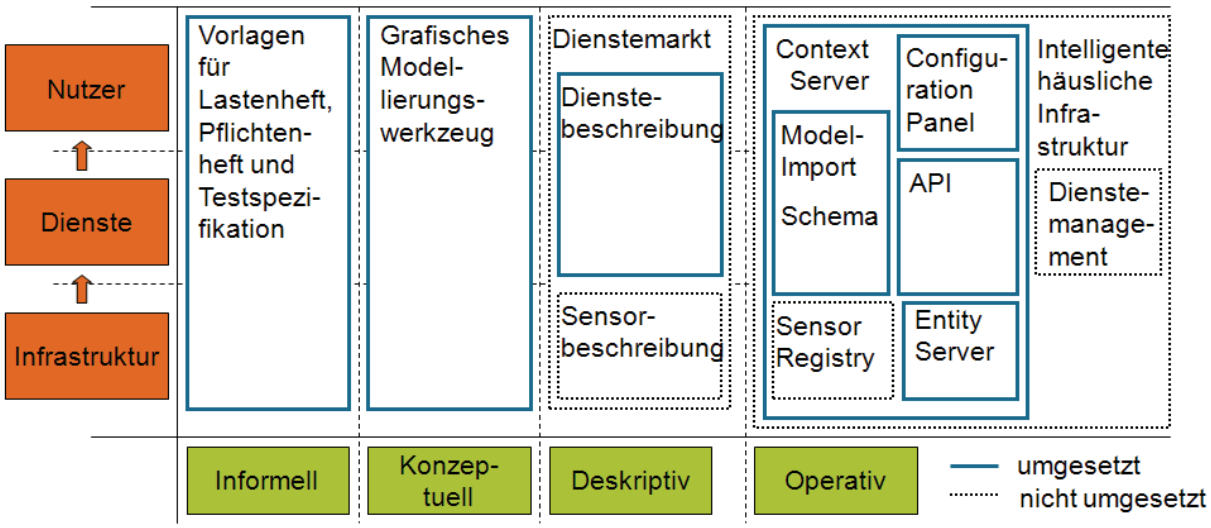


Abbildung 67: Werkzeuge für die Umsetzung der Methodik

Für die Beschreibung des informellen Kontextmodells in den Phasen der Planung, der Definition und des Tests können **Vorlagen** für Textdokumente ver-

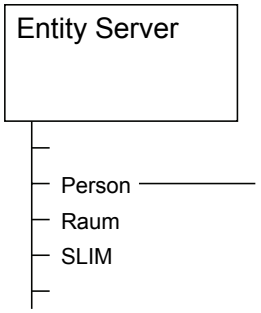
wendet werden, in denen die strukturellen Vorgaben umgesetzt sind. Auf diese Weise können Lasten- und Pflichtenhefte sowie Testspezifikationen mit den für die Kontextmodellierung notwendigen Erweiterungen erstellt und für die folgenden Phasen zur Verfügung gestellt werden.

Mit Hilfe eines **grafischen Modellierungswerkzeugs** kann die definierte grafische Notation des konzeptuellen Kontextmodells umgesetzt und in der Designphase der Modellierung zur Verfügung gestellt werden. Es gibt eine Reihe von Möglichkeiten zur Umsetzung eines solchen Modellierungswerkzeugs, von denen eine einfache auf der Basis eines Zeichenprogramms erfolgen kann. Im Rahmen dieser Dissertation wurden auch Diagrammvorlagen für MS Visio erstellt, die ebenfalls für die konzeptuelle Modellierung genutzt werden können. Eine andere Möglichkeit besteht in der Erweiterung der integrierten Entwicklungsumgebung mit Hilfe einer grafischen Bibliothek, wofür das EMF [178] für Eclipse ein Beispiel ist. Da das Kontext-Metamodell auf Basis der MOF Metadaten-Architektur erstellt wurde, bietet sich hier die Verwendung eines entsprechenden Modellierungswerkzeugs an. Leider ist die graphische Notation des konzeptuellen Kontextmodells nicht vollständig in die Möglichkeiten der Darstellung von UML-Stereotypen abbildbar. Auch die Verwendung eines Modellierungswerkzeugs für domänenspezifische Programmiersprachen (DSL) gehört dazu. Ein Beispiel hierfür ist das in der Softwareentwicklungsumgebung „Visual Studio“ von Microsoft enthaltene „DSL Tool“.

Der **Dienstemarkt**, zu dem die Sensorik und die AAL-Dienste gehören, hat die Aufgabe, Komponenten bereitzustellen, die in die intelligente häusliche Infrastruktur übernommen werden können, um diese zu erweitern. Eine Umsetzung des Dienstemarktes wurde im Rahmen dieser Arbeit nicht vorgenommen. Grundlage für sie könnte die Nutzung einer Service-Registry (UDDI) aus dem Umfeld der service-orientierten Architektur sein. Die notwendige Erweiterung einer **Dienstebeschreibung** für die kontextadaptiven Aspekte eines AAL-Dienstes ist aber Bestandteil der Dissertation, wogegen die **Beschreibung der Sensorik** in eine Diplomarbeit ausgelagert wurde. Der Dienstemarkt kann genutzt werden, um neue Komponenten potentiellen Nutzern bekannt zu machen.

Die **intelligente häusliche Infrastruktur** stellt die Basisfunktionen für die Übernahme und den Betrieb der kontextadaptiven AAL-Dienste bereit. Dazu gehört die Umsetzung der strukturellen Kontextadaptivität (Kap. 3.3.1), die Integration und der Betrieb der AAL-Dienste sowie die Umsetzung der Interaktionskonzepte für die Kontrolle der intelligenten Infrastruktur (Kap. 7.3.1) und den Zugriff auf den Dienstemarkt. Diese Funktionen sind im Rahmen der Dissertation nicht umgesetzt. Ein wichtiger Bestandteil der intelligenten häuslichen Infrastruktur ist der **Context Server**, der die operativen Kontextmodelle zur Verfügung stellt und somit ermöglicht, dass die Erkennnung und Bereitstellung von Kontextinformationen, die zur Umsetzung des kontextadaptiven Verhaltens eines AAL-Dienstes gehören, genutzt werden können. Der Kontextser-

ver besteht aus einer Reihe von Einzelkomponenten, wobei der **Entity Server** als erste der Registrierung konkreter Kontextentitäten, z.B. der Person „Erika Maier“, dient. Zu jedem bekannten Entitätstyp können Schemainformationen definiert werden, die für die Beschreibung und Suche beispielsweise von Name, Alter oder Geschlecht einer Person notwendig sind. Der Entity Server stellt zudem eine global eindeutige ID für jede konkrete Instanz einer Entität zur Verfügung, wie es aus der nachfolgenden Abbildung zu entnehmen ist. Somit ist der Entity Server eine auf die Verwaltung von Kontextentitäten spezialisierte Datenbank. Darüberhinaus muss diese mit dem Kontextmodell synchronisiert werden. Wird eine konkrete Kontextentität im Entity Server registriert, so muss diese im Kontextserver in das Kontextmodell aufgenommen werden. Wird die Kontextentität aus dem Entity Server entfernt, so muss diese ebenfalls aus dem Kontextmodell herausgenommen werden.



Name	Vorname	Geschl.	Alter	GlobalID
Meier	Hans	M	34	Person:1
Meier	Klara	W	32	Person:2
Meier	Erna	W	78	Person:3

Abbildung 68: Verwaltung von Entitäten durch den Entity Server

Eine weitere Komponente ist die **Sensor Registry**. Sie ermöglicht die Verwaltung und Registrierung von Sensoren. Mit Hilfe der Sensorbeschreibung erfolgt hier eine Zuordnung der Sensoren zum Kontextmodell auf der Ebene der Infrastruktur und den Entitätsinstanzen des Entity Servers. Neben der Zuordnung werden die Qualitätseigenschaften, die Nutzungsmodalitäten und die technischen Parameter zum Zugriff auf den Sensor beschrieben.

Die **Import-Komponente** realisiert dann die Übernahme der implementationsneutralen Beschreibung des operativen Kontextmodells in den Context Server und setzt diese in eine entsprechende Implementierung um. Der Context Server besitzt eine **API** (Application Programming Interface), über die ein AAL-Dienst auf die Kontextinformationen zugreifen und synchron Kontextinformationen abfragen kann. Zudem ist aber auch die asynchrone Benachrichtigung bei Eintreten von Ereignissen veranlassbar. Die Beschreibung der API ist im Anhang (Kap. A.1.3) zu finden. Eine weitere Komponente des Context Server ist das **Configuration Panel**, welches der Umsetzung des Interaktionskonzepts zur Konfiguration des kontextadaptiven Verhaltens eines AAL-Dienstes

durch den Bewohner dient (Kap. 7.3.3.3). Dieses kann in den AAL-Dienst eingebunden werden und ergänzt ihn um die beschriebene Funktionalität. Die aufgeführten Komponenten des Context Servers sind im Rahmen dieser Arbeit umgesetzt und können genutzt werden. Eine Ausnahme ist die Sensor Registry, die im Rahmen der erwähnten Diplomarbeit nicht vollständig umgesetzt wurde. Existierende Umsetzungen von Kontextservern und Frameworks unterstützen das vorgestellte Kontext-Metamodell nicht und können daher nicht im Rahmen der Methodik eingesetzt werden.

Bestandteil der intelligenten häuslichen Infrastruktur ist auch eine **Dienste-verwaltung**. Sie versorgt in der häuslichen Umgebung vorhandene Dienste und ermöglicht die Übernahme von AAL-Diensten aus dem Dienstemarkt, ist aber nicht Gegenstand der Dissertation.

8.2 Methodik

Die Methodik der Kontextmodellierung zur Realisierung kontextadaptiver AAL-Dienste besteht aus der Anwendung des zweidimensionalen Vorgehens unter Nutzung der identifizierten Werkzeuge. Es wird dabei kein konkretes Vorgehensmodell der Softwareentwicklung festgelegt. Die in der Dimension ‚Phasen der Entwicklung und Nutzung' identifizierten Phasen müssen in das jeweils ausgewählte Vorgehensmodell abgebildet werden. Dieses kann beispielsweise das V-Modell sein.

Aus der jeweiligen Phase ergibt sich die Verwendung des entsprechenden Modelltyps und Werkzeugs. Die konkrete Ausprägung der Methodik ist je nach Anwendungsfall unterschiedlich, da der Entwicklung von AAL-Diensten unterschiedlich komplexe Einsatzszenarien zugrunde liegen können. Diese werden nachfolgend beschrieben und auf die Unterschiede eingegangen.

- Ein komplexes Einsatzszenario wird in der Zielsetzung dieser Dissertation in Kap. 3.1 beschrieben. In ihm können verschiedene Anbieter AAL-Dienste entwickeln und auf einem Dienstemarkt zur Verfügung stellen. Der Bewohner kann dann seine intelligente häusliche Umgebung individuell mit der Sensorik und den AAL-Diensten ausstatten. Im Szenario erfolgt eine vollständige Trennung zwischen Infrastruktur und AAL-Diensten sowie den Anbietern von AAL-Diensten und deren Nutzern.

- In einem vereinfachten Einsatzszenario erfolgt die Bereitstellung der häuslichen Infrastruktur und der darin verfügbaren AAL-Dienste dagegen aus einer Hand. Hierbei gibt es keine Arbeitsteilung nach dem obigen Prinzip. Die Sensoren und die AAL-Dienste werden von einem Anbieter erstellt. Der Anbieter ermöglicht jedoch seinem Kunden den Aufbau der Kontextinfrastruktur und die sukzessive individuelle Erweiterung der Dienstemenge aus einem geschlossenen Dienstemarkt. Ein

solch vereinfachtes Einsatzszenario existiert bereits. beispielsweise mit der Firma „Moeller“ in Kooperation mit der Firma „Metz“ [179]. Wohnungen können mit funkbasierten Sensoren der Firma „Moeller„ ausgestattet werden. Sensorsignale gehen in ein Steuermodul ein und stehen dort unterschiedlichen Diensten zur Verfügung. Diese können über ein Fernsehgerät der Firma „Metz“ dargestellt und gesteuert werden. Zusätzliche Dienste können ausschließlich von der Firma „Moeller“ erworben werden. Somit können Fremdanbieter keine eigenen Dienste in dieses geschlossene System einbringen.

- Aktuelle Ansätze zur Entwicklung von kontextadaptiven Diensten basieren bisher nur auf einfachen Einsatzszenarien. In ihnen existiert eine enge Kopplung zwischen Kontextinfrastruktur und dem Dienst, was bedeutet, dass eine Kontextinfrastruktur nur für einen konkreten Dienst aufgebaut wird und bezogen auf das Kontextmodell und die Ausstattung mit Sensorik ausgerichtet ist. Eine Mehrfachverwendung der Infrastruktur und die Bereitstellung von Diensten über einen Dienstemarkt existiert hier nicht.

Alle drei Varianten besitzen ihre Berechtigung und können durch die nachfolgend vorgestellte Methodik unterstützt werden. Um die notwendige Flexibilität zu ermöglichen, erfolgt eine Aufteilung der Methodik in fünf Teilschritte. Die konkrete Ausprägung einer jeden ist dabei abhängig vom jeweiligen Einsatzszenario. Die Methodik wird in folgende Teilschritte gegliedert:

- Entwicklung der Kontextinfrastruktur: Die für den Betrieb eines AAL-Dienstes benötigte Kontextinfrastruktur wird definiert und umgesetzt. Diese beinhaltet Funktionen zur Umsetzung des strukturellen kontextadaptiven Verhaltens, sowie zur Beschreibung der in der intelligenten häuslichen Umgebung vorhandenen Entitäten. Ein wesentliches Ergebnis ist hierbei das Kontextmodell auf Ebene der Infrastruktur der Ebene M1.

- Instantiierung der Kontextinfrastruktur: Die Kontextinfrastruktur ist Bestandteil der intelligenten häuslichen Umgebung des Bewohners. Das Kontextmodell auf Ebene der Infrastruktur wird durch die Einbindung der konkret vorhandenen Sensoren und Entitäten instanziiert. Durch die Bereitstellung von Kontextinformationen durch die Sensoren ist der aktuelle Zustand der intelligenten häuslichen Umgebung im Kontextserver verfügbar.

- Entwicklung von AAL-Dienstetypen: Es werden die dienstetypspezifischen Kontextmodelle auf der Ebene der Dienste, sowie der Nutzerinteraktion der Ebene M1 definiert und umgesetzt. Diese werden im Kontextserver bereitgestellt und können zur Umsetzung des funktionellen kontextadaptiven Verhaltens der AAL-Dienste genutzt werden.

- Entwicklung eines AAL-Dienstes: Die Funktionalität eines konkreten AAL-Dienstes wird definiert und umgesetzt. Die Umsetzung des funktionalen kontextadaptiven Verhaltens basiert auf einem Ausschnitt des dienstetypspezifischen Kontextmodells.

- Instantiierung der AAL-Dienstemenge: Ein AAL-Dienst ist im Dienstemarkt verfügbar und kann über die Dienstebeschreibung vom Bewohner gefunden und in die intelligente häusliche Umgebung aufgenommen werden. Der AAL-Dienst setzt auf dem instantiierten Kontextmodell auf Ebene der Dienste und Nutzerinteraktion auf. Der Bewohner kann den Dienst konfigurieren und nutzen.

Für die Umsetzung der Zielvision von Trennung zwischen Infrastruktur und AAL-Diensten in dem komplexen Szenario wird eine herstellerübergreifende standardisierte Definition des Kontextmodells auf Ebene M1 auf den Ebenen der Infrastruktur, der Dienste und der Nutzerinteraktion benötigt. Ein solches Referenz-Kontextmodell bildet die Grundlage für die Bereitstellung und Einbindung von Sensorik, sowie für die Umsetzung des kontextadaptiven Verhaltens der AAL-Dienste unterschiedlicher Hersteller. Die Definition des Referenz-Kontextmodells umfasst die Entwicklung der Kontextinfrastruktur sowie der AAL-Dienstetypen und muss vor den weiteren Teilschritten erfolgen. Ein solches Referenz-Kontextmodell existiert derzeit noch nicht und kann auch nicht im Rahmen dieser Arbeit erstellt werden. Der Grund dafür ist, dass als Grundlage für ein solches Referenz-Kontextmodell ein ausreichend vollständiges Bild über die wichtigen AAL-Dienste und deren Anforderungen an das Kontextmodell gegeben sein muss. Dieses fehlt noch aktuell, da die Identifikation, Entwicklung und Erprobung von sinnvollen Diensten noch ein aktuelles Forschungsfeld ist. Die Analyse der Anforderungen in Kap. 3.3.2 basiert auf aktuell bekannten Diensten. Darauf aufbauend wird im Anhang (A.2) ein initialer Vorschlag für die jeweiligen Kontextmodelle gemacht. Dieses kann dann sukzessive anhand der Anforderungen neuer AAL-Dienste hin zu einem Referenz-Kontextmodell ergänzt werden. Aufbauend auf den Referenz-Kontextmodellen können AAL-Dienste von Anbietern entwickelt und auf dem Dienstemarkt zur Verfügung gestellt werden. Die häusliche intelligente Umgebung eines Bewohners kann durch eine Kontextinfrastruktur ausgestattet und instantiiert werden. Dazu werden die verfügbaren Sensoren eingebunden und die in der Umgebung vorhandenen Kontextentitäten beschrieben. Der Bewohner kann dann die gewünschten Dienste aus dem Dienstemarkt aussuchen, diese in die intelligente häusliche Umgebung übernehmen und nutzen. Aufgrund neu verfügbarer Sensoren und weiteren Anforderungen aus Sicht der AAL-Dienste kann eine Erweiterung der Referenz-Modelle notwendig sein.

Das vereinfachte Einsatzszenario macht ebenso die Definition eines Referenz-Kontextmodells nach dem obigen Schema notwendig. Allerdings muss dieses nicht herstellerübergreifend standardisiert, sondern kann nach Belieben vom Hersteller festgelegt werden. Die Entwicklung von AAL-Diensten erfolgt durch

den Produzenten auf der Basis der von ihm festgelegten Kontextmodelle. Über einen geschlossenen Dienstemarkt kann der Anbieter den Kunden die entwickelten AAL-Dienste zur Verfügung stellen. Die Instantiierung der Kontextinfrastruktur und der Dienstemenge kann wieder nach obigem Muster erfolgen.

Bei einer engen Kopplung zwischen Kontextinfrastruktur und AAL-Dienst kann die Entwicklung von AAL-Dienstetypen entfallen. Statt der dienstetypspezifischen Kontextmodelle können dienste-spezifische Kontextmodelle erstellt und auf Ebene der Dienste umgesetzt werden. Die Entwicklung der Kontextinfrastruktur sowie der AAL-Dienste können fließend ineinander übergreifend umgesetzt werden. Dieses gilt ebenso für die Instantiierung der Kontextinfrastruktur und der Dienstemenge. Entsprechend wird dann ein Dienstemarkt zur Bereitstellung von AAL-Diensten nicht benötigt. Dieses Einsatzszenario kann bereits heute mit den aktuellen Methoden umgesetzt werden. Allerdings bietet die Methodik dieser Arbeit zusätzlich die phasenübergreifende Betrachtung der Kontextmodellierung für die Entwicklung und Nutzung von AAL-Diensten, sowie die Konzepte zur Nutzerinteraktion.

Nachfolgend werden die einzelnen Teilschritte im Detail beschrieben.

8.2.1 Entwicklung der Kontextinfrastruktur

Zielstellung dieses Teilschritts ist die Entwicklung der Kontextinfrastruktur für eine intelligente häusliche Umgebung. Bestandteil der Kontextinfrastruktur ist die Bereitstellung von Informationen über die häusliche Umgebung in Form eines Kontextmodells. Zudem müssen durch die Kontextinfrastruktur Funktionen bereitgestellt werden, mit denen das strukturelle kontextadaptive Verhalten der AAL-Dienste umgesetzt werden kann. In der folgenden Abbildung sind die Elemente der Kontextmodellierung, sowie die benötigten Werkzeuge farblich hervorgehoben.

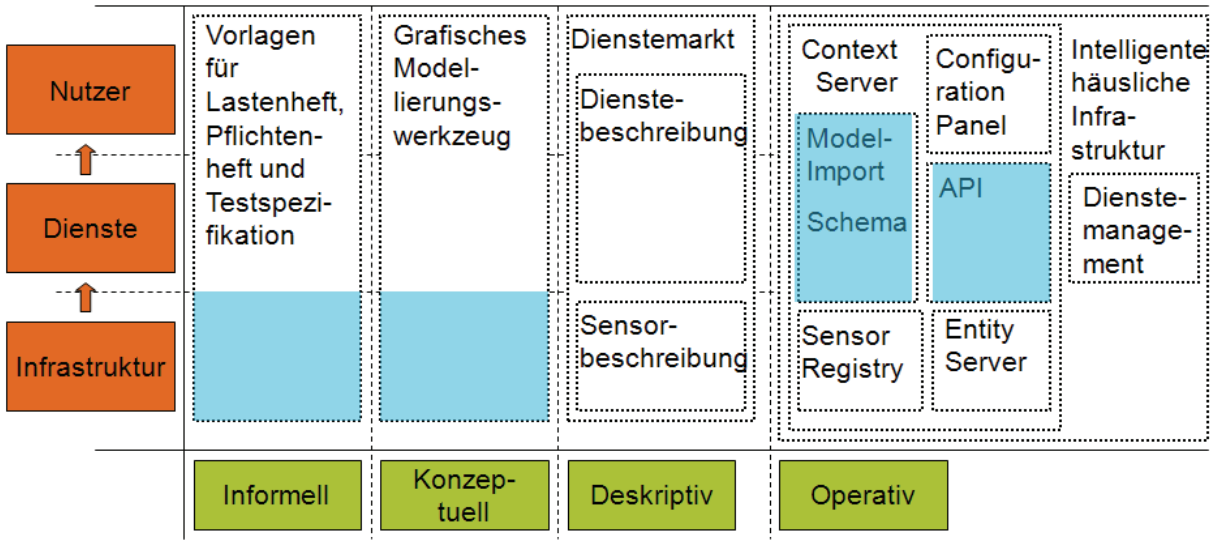


Abbildung 69: Modellelemente und Werkzeuge zur Entwicklung der Kontextinfrastruktur

Nachfolgend werden die einzelnen Schritte der Entwicklung der Kontextinfrastruktur mit Schwerpunkt auf die Kontextadaptivität beschrieben.

8.2.1.1 Planung und Definition der Kontextinfrastruktur

Ebene auf der Metadaten-Architektur: M1

Dimension „Einsatzzweck“: Infrastruktur

Dimension „Phasen der Entwicklung und Nutzung“: Informell

Werkzeug: Lastenheft, Pflichtenheft

Vorbedingung: -

Ergebnis: Schriftlich dokumentierte Anforderungen an das Kontextmodell aus fachlicher Sicht.

Beschreibung: Die Planung und die Definition der Kontextinfrastruktur werden hier zusammengefasst beschrieben. In diesen Phasen müssen zunächst die Anforderungen an das umzusetzende strukturelle kontextadaptive Verhalten der Kontextinfrastruktur, sowie die bereitzustellenden Kontextinformationen über die intelligente häusliche Umgebung erhoben werden. Die Strukturierung und Dokumentation dieser Anforderungen erfolgt unter Verwendung der infor-

mellen Beschreibung des Kontextmodells auf Ebene der Infrastruktur mit Hilfe eines Lastenhefts (Kap. 5.5.1) bzw. Pflichtenhefts (Kap. 5.5.2).

8.2.1.2 Design der Kontextinfrastruktur

Ebene auf der Metadaten-Architektur: M1

Dimension „Einsatzzweck“: Infrastruktur

Dimension „Phasen der Entwicklung und Nutzung“: Konzeptuell

Werkzeug: Modellierungswerkzeug

Vorbedingung: Pflichtenheft

Ergebnis: Konzeptuelles Kontextmodell

Beschreibung: Auf Basis des Pflichtenhefts erfolgt nun das Design der Kontextinfrastruktur. Die informelle Beschreibung der benötigten Kontextinformationen wird mit Hilfe eines Modellierungswerkzeugs auf Basis der grafischen Notation in das konzeptuelle Kontextmodell (Kap. 5.6.3) überführt. Es werden die benötigten Kontextentitäten und –relationen, sowie die zugehörigen Kontextattribute identifiziert. Diese werden so strukturiert, dass gemeinsame Kontextattribute durch Generalisierungs-/Spezialisierungsbeziehungen zwischen Kontextentitäten und –relationen beschrieben sind. Zudem werden die Kontextdimensionen und Repräsentationsformen im konzeptuellen Kontextmodell ergänzt.

8.2.1.3 Implementierung der Kontextinfrastruktur

Ebene auf der Metadaten-Architektur: M1

Dimension „Einsatzzweck“: Infrastruktur

Dimension „Phasen der Entwicklung und Nutzung“: Operativ

Werkzeug: Modellierungswerkzeug, Context Server: Model-Import, API

Vorbedingung: Konzeptuelles Kontextmodell

Ergebnis: Kontextmodell im Kontextserver operativ verfügbar, Umgesetztes strukturelles kontextadaptives Verhalten der Kontextinfrastruktur

Beschreibung: Das konzeptuelle Kontextmodell wird um zusätzliche Modellelemente ergänzt, z.B. Eigenschaften der Kontextattribute (Kap. 5.7.1). Daraus resultiert das operative Kontextmodell. Dieses wird mit Hilfe des Modellierungswerkzeugs in die implementationsneutrale Repräsentationsform (Kap. 7.8.2) überführt. Mit Hilfe der Komponente „Model-Import“ wird dieses in den Kontextserver importiert und dort wird die entsprechende implementationsspezifische Repräsentation generiert. Das Kontextmodell auf Ebene der Infrastruktur ist im Kontextserver operativ verfügbar.

Die Anwendungslogik der Kontextinfrastruktur ist nach den bekannten Vorgehensmodellen umgesetzt. Für die Umsetzung der Kontextlogik kann diese mittels einer API auf das Kontextmodell im Kontextserver zugreifen.

8.2.1.4 Test der Kontextinfrastruktur

Ebene auf der Metadaten-Architektur: M1

Dimension „Einsatzzweck“: Infrastruktur

Dimension „Phasen der Entwicklung und Nutzung“: Informell, Operativ

Werkzeug: Testspezifikation, Context Server: API

Vorbedingung: Pflichtenheft, Kontextserver

Ergebnis: Umgesetztes strukturelles kontextadaptives Verhalten der Kontextinfrastruktur ist getestet

Beschreibung: Basierend auf den im Pflichtenheft beschriebenen Anforderungen wird eine Testspezifikation erstellt. Das zu testende kontextadaptive Verhalten der Kontextinfrastruktur wird mit Hilfe der informellen Kontextbeschreibung (Kap. 5.5.3) in der Testspezifikation festgelegt. Die Umsetzung der Testfälle kann vom Kontextserver unterstützt werden, indem die in der Testspezifikation festgelegten Ausprägungen der Kontextinformationen simuliert werden. Die umgesetzte Kontextinfrastruktur greift dabei auf die Kontextinformationen im Kontextserver über die API zu.

Es gibt einige wenige bekannte Ansätze in der Literatur zum Testen kontextadaptiver Anwendungen. In [180] wird ein Ansatz vorgestellt, in welchem automatisch die kritischen Testpunkte identifiziert und systematisch die dem Test zugrundeliegenden Kontextinformationen manipuliert werden können. Die Voraussetzung für die Nutzung dieses Ansatzes ist das Vorhandensein des Quellcodes und eine Beschreibung der Signaturen der zugrundeliegenden Middleware bzw. Kontextservers, mit welcher die Anwendung mit der Kontextlogik verknüpft ist. Mit Hilfe der Seiteneffekt- und der Escape-Analyse werden

die kontextadaptiven Programmteile identifiziert. Das Ergebnis ist hier ein Kontrollflussgraph in denen die kontextadaptiven Teile markiert sind. Auf dieser Basis werden dann die Kontextänderungen identifiziert, welche zu einer Änderung des Verhaltens der Anwendung führen. Diese dienen dann der automatischen Generierung von Testfällen. Die Umsetzung basiert auf der Verwendung des Context Toolkits und eine direkte programmatische Verknüpfung der Anwendungslogik mit der Middleware. Methoden für das Testen von kontextadaptiven Anwendungen sind nicht Gegenstand dieser Dissertation und werden daher nicht weiter vertieft.

8.2.2 Instantiierung der Kontextinfrastruktur

Mit der Instantiierung erfolgt die Übernahme einer verfügbaren Kontextinfrastruktur in die intelligente häusliche Umgebung des Bewohners. Diese wird dort in der spezifischen Ausprägung bereitgestellt. Dazu gehören die Registrierung der vorhandenen Kontextentitäten und die Übernahme der Sensoren in die Infrastruktur. In der folgenden Abbildung sind die Elemente der Kontextmodellierung, sowie die benötigten Werkzeuge farblich hervorgehoben.

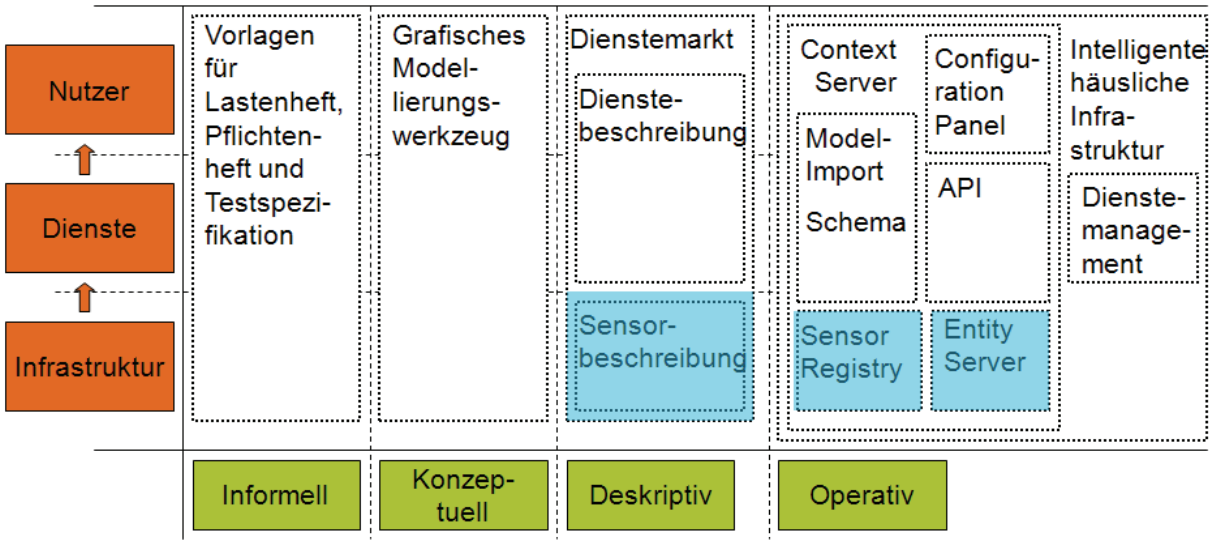


Abbildung 70: Modellelemente und Werkzeuge zur Instantiierung der Kontextinfrastruktur

Nachfolgend werden die einzelnen Schritte der Instantiierung der Kontextinfrastruktur beschrieben.

8.2.2.1 Bereitstellung der Sensorik

Ebene auf der Metadaten-Architektur: M0

Dimension „Einsatzzweck“: Infrastruktur

Dimension „Phasen der Entwicklung und Nutzung“: Deskriptiv

Werkzeug: -

Vorbedingung: Kontextmodell auf Ebene M1

Ergebnis: Sensoren für die Einbindung in die Kontextinfrastruktur sind mittels der Sensorbeschreibung verfügbar.

Beschreibung: Mit Hilfe der Sensorbeschreibung (Kap. 5.8.1) werden die verfügbaren Sensoren und deren Eigenschaften beschrieben.

8.2.2.2 Übernahme in die Kontextinfrastruktur

Ebene auf der Metadaten-Architektur: M0

Dimension „Einsatzzweck“: Infrastruktur

Dimension „Phasen der Entwicklung und Nutzung“: Deskriptiv

Werkzeug: Context Server: Sensor Registry, Entity Server

Vorbedingung: Die Kontextinfrastruktur inklusive der darin enthaltenen Komponenten und des kontextadaptive strukturellen Verhaltens sind in die intelligente häusliche Umgebung des Bewohners übernommen. Die Möglichkeiten der Übernahme und des Betriebs dieser Komponenten werden an dieser Stelle nicht weiter betrachtet. Grundlagen dafür können jedoch im Bereich ‚Home Automation’ gefunden werden (Kap. 2.1.2). Bestandteil der Kontextinfrastruktur ist das definierte Kontextmodell der Infrastruktur auf Ebene M1. Sensoren sind inklusive der Sensorbeschreibungen verfügbar.

Ergebnis: Die Kontextinfrastruktur ist instantiiert. Die Kontextentitäten sind registriert. Die Sensoren beliefern den Kontextserver mit Kontextinformationen zu den registrierten Kontextentitäten und –relationen.

Beschreibung: Die in der intelligenten häuslichen Umgebung vorhandenen Kontextentitäten werden im „Entity Server“ registriert. Dieses erfolgt mit Hilfe der Entitätsbeschreibung (Kap. 5.8.2). Damit wird das Kontextmodell auf Ebe-

ne M1 instantiiert. Die konkreten Kontextentitäten und deren Beziehungen sind auf Ebene M0 hinzugefügt.

Mit Hilfe der Sensorbeschreibung (Kap. 5.8.1) werden die vorhandenen Sensoren in der „Sensor Registry“ eingebunden. Bei der Registrierung erfolgt die Zuordnung der Kontextinformationen zu den Kontextattributen der entsprechenden Kontextentitäten bzw. –relationen. Die von den Sensoren gelieferten Kontextinformationen werden in das Kontextmodell auf Ebene der Infrastruktur auf Ebene M0 übernommen.

8.2.2.3 Betrieb der Kontextinfrastruktur

Ebene auf der Metadaten-Architektur: M0

Dimension „Einsatzzweck“: Infrastruktur

Dimension „Phasen der Entwicklung und Nutzung“: Operativ

Werkzeug: Context Server: API

Vorbedingung: Übernahme der Kontextentitäten und der Sensoren in die Kontextinfrastruktur.

Ergebnis: Die Kontextinfrastruktur stellt das strukturelle kontextadaptive Verhalten für die Nutzung durch die AAL-Dienste bereit.

Beschreibung: Die Kontextinfrastruktur greift auf das Kontextmodell auf Ebene der Infrastruktur mittels der API des Kontextservers zu, um das strukurelle kontextadaptive Verhalten umzusetzen. Der Kontextserver wird durch die Sensorik beliefert und nutzt das vorhandene Regelwissen, um neue Kontextinformationen abzuleiten oder deren Gültigkeit zu überprüfen.

8.2.3 Entwicklung von AAL-Dienstetypen

Bei der Entwicklung von AAL-Dienstetypen werden die dienstetyp-spezifischen Kontextmodelle auf der Ebene der Dienste sowie der Nutzerinteraktion definiert und umgesetzt. In der nachfolgenden Abbildung sind die betroffenen Elemente in den beiden Dimensionen der Kontextmodellierung farblich hervorgehoben.

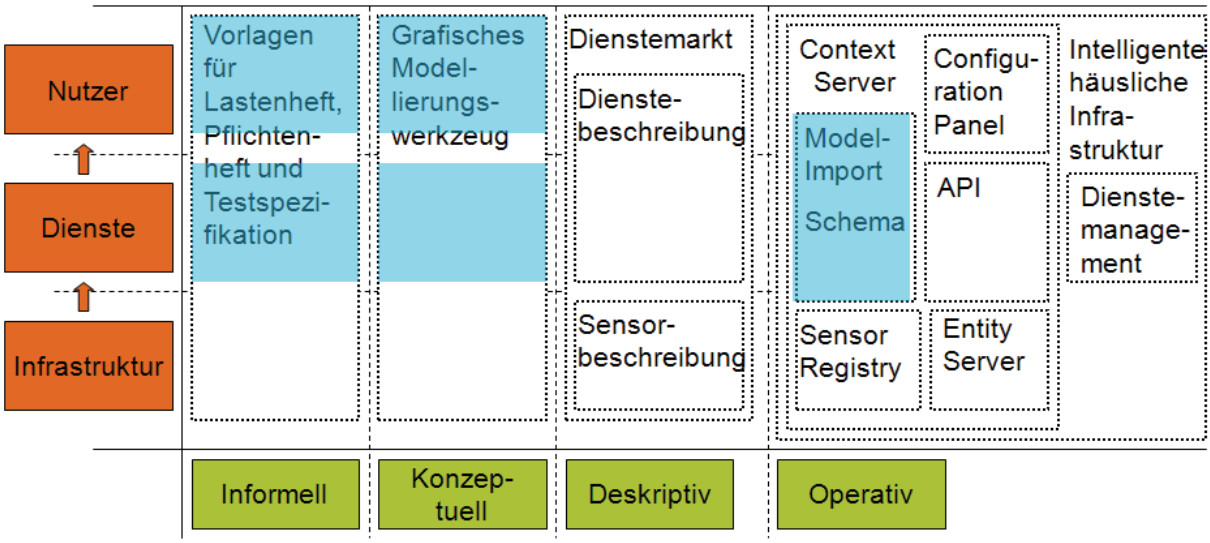


Abbildung 71: Modellelemente und Werkzeuge zur Entwicklung von Dienstetypen

Nachfolgend werden die einzelnen Schritte zur Entwicklung der Dienstetypen beschrieben.

8.2.3.1 Planung und Definition der Dienstetypen

Ebene auf der Metadaten-Architektur: M1

Dimension „Einsatzzweck“: Dienste, Nutzerinteraktion

Dimension „Phasen der Entwicklung und Nutzung“: Informell

Werkzeug: Lastenheft, Pflichtenheft

Vorbedingung: -

Ergebnis: Schriftlich dokumentierte Anforderungen an das Kontextmodell eines Dienstetyps aus fachlicher Sicht.

Beschreibung: Die Entwicklung der AAL-Dienstetypen entspricht der Domänenmodellierung für einen Anwendungsbereich von AAL-Diensten. Hier muss zunächst das generell benötigte kontextadaptive Verhalten innerhalb einer Domäne, welche durch einen Dienstetyp umgesetzt wird, definiert werden. Dieses gilt auch für die Situationstaxonomien und die Verfeinerungsmöglichkeiten. Das erfolgt zunächst anhand des Lastenhefts in der Planungsphase (Kap. 6.6.1, Kap. 7.6). Dieses wird in der Definitionsphase mit Hilfe des Pflich-

tenhefts weiter konkretisiert (Kap. 6.6.2, Kap. 7.6). Die informelle Beschreibung der benötigten Kontextinformationen erfolgt sowohl für die Ebene der Dienste als auch der Nutzerinteraktion.

8.2.3.2 Design der Dienstetypen

Ebene auf der Metadaten-Architektur: M1

Dimension „Einsatzzweck“: Dienste, Nutzerinteraktion

Dimension „Phasen der Entwicklung und Nutzung“: Konzeptuell

Werkzeug: Modellierungswerkzeug

Vorbedingung: Pflichtenheft

Ergebnis: Konzeptuelles Kontextmodell

Beschreibung: Auf Basis des Pflichtenhefts erfolgt nun das Design der Dienstetypen. Die informelle Beschreibung der benötigten Kontextinformationen wird mit Hilfe eines Modellierungswerkzeugs auf Basis der grafischen Notation in das konzeptuelle Kontextmodell auf Ebene der Dienste (Kap. 6.7) und auf Ebene der Nutzerinteraktion (Kap. 7.7) überführt. Es werden die benötigten Kontextentitäten und –relationen, sowie die zugehörigen Kontextattribute identifiziert. Weiterhin werden die Kontexträume, sowie die darin enthaltenen Kontextentitäten und –relationen, sowie die relevanten Kontextattribute beschrieben. Zudem werden auf Ebene der Nutzerinteraktion die zugehörige Situationstaxonomie und die möglichen Verfeinerungsschritte festgelegt. Initiale Kontextmodelle für die bereits identifizierten Dienstetypen sind im Anhang (Kap. A.2.2) zu finden.

8.2.3.3 Implementierung der Dienstetypen

Ebene auf der Metadaten-Architektur: M1

Dimension „Einsatzzweck“: Dienste, Nutzerinteraktion

Dimension „Phasen der Entwicklung und Nutzung“: Operativ

Werkzeug: Modellierungswerkzeug, Context Server: Model-Import, API

Vorbedingung: Konzeptuelles Kontextmodell

Ergebnis: Dienstetyp-spezifische Kontextmodelle im Kontextserver operativ verfügbar.

Beschreibung: Das konzeptuelle Kontextmodell wird um zusätzliche Modellelemente ergänzt, z.B. die Abbildung der Modellelemente auf die Ebene der Infrastruktur (Kap. 6.8.1) bzw. die Verbindung zwischen den Ebenen der Dienste und der Nutzerinteraktion (Kap. 7.8.1). Daraus resultiert das operative Kontextmodell. Dieses wird mit Hilfe des Modellierungswerkzeugs in die implementationsneutrale Repräsentationsform (Kap. 6.8.2, 7.8.2) überführt. Mit Hilfe der Komponente „Model-Import“ wird dieses in den Kontextserver importiert und dort wird die entsprechende implementationsspezifische Repräsentation generiert. Die dienstetyp-spezifischen Kontextmodelle auf Ebene der Dienste und der Nutzerinteraktion sind im Kontextserver operativ verfügbar.

8.2.4 Entwicklung eines AAL-Dienstes

Bei der Entwicklung eines AAL-Dienstes erfolgt die Definition und Umsetzung eines AAL-Dienstes auf der Grundlage eines dienstetyp-spezifischen Kontextmodells. In der nachfolgenden Abbildung sind die betroffenen Elemente in den beiden Dimensionen der Kontextmodellierung farblich hervorgehoben.

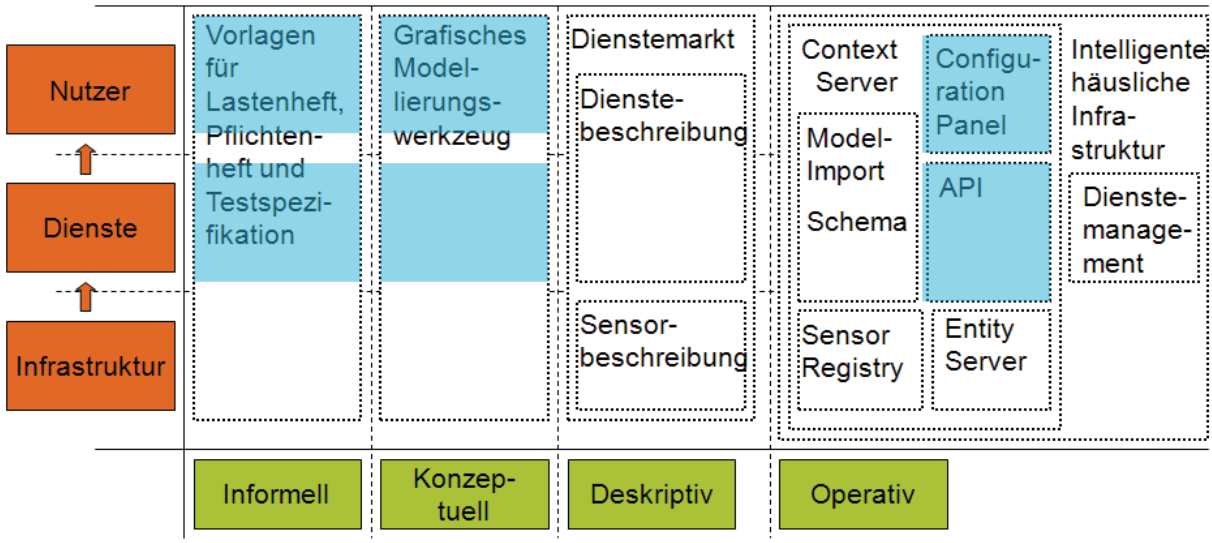


Abbildung 72: Modellelemente und Werkzeuge zur Entwicklung eines AAL-Dienstes

Nachfolgend werden die einzelnen Schritte zur Entwicklung der Dienstetypen beschrieben.

8.2.4.1 Planung und Definition eines AAL-Dienstes

Ebene auf der Metadaten-Architektur: M1

Dimension „Einsatzzweck“: Dienste, Nutzerinteraktion

Dimension „Phasen der Entwicklung und Nutzung“: Informell

Werkzeug: Lastenheft, Pflichtenheft

Vorbedingung: Dienstetyp-spezifisches Kontextmodell

Ergebnis: Schriftlich dokumentierte Anforderungen an das Kontextmodell eines AAL-Dienstes aus fachlicher Sicht.

Beschreibung: Die Entwicklung eines AAL-Dienstes beginnt mit der Beschreibung des funktionalen kontextadaptiven Verhaltens mit Hilfe eines Lastenhefts (Kap. 6.6.1). Es muss sich im Rahmen der durch die dienstetyp-spezifischen Kontextmodelle gegebenen Vorgaben bewegen. Sind zusätzliche Anforderungen vorhanden, so können diese zu einer Erweiterung des Kontextmodells der AAL-Dienstetypen führen (Kap. 8.2.3). Diese Beschreibung eines AAL-Dienstes wird in der Definitionsphase mit Hilfe des Pflichtenhefts (Kap. 6.6.2) weiter konkretisiert.

8.2.4.2 Design eines AAL-Dienstes

Ebene auf der Metadaten-Architektur: M1

Dimension „Einsatzzweck“: Dienste, Nutzerinteraktion

Dimension „Phasen der Entwicklung und Nutzung“: Konzeptuell

Werkzeug: Modellierungswerkzeug

Vorbedingung: Pflichtenheft, Dienstetyp-spezifisches Kontextmodell

Ergebnis: Konzeptuelles Kontextmodell des AAL-Dienstes

Beschreibung: Für die Umsetzung des kontextadaptiven Verhaltens wird nun in der Konzeptionsphase der relevante Ausschnitt aus dem konzeptuellen Kontextmodells eines Dienstetyps ausgewählt und mit den spezifischen Qualitätsanforderungen ergänzt (Kap. 6.7). Dieses erfolgt mit Hilfe des Modellierungswerkzeugs.

8.2.4.3 Implementierung eines AAL-Dienstes

Ebene auf der Metadaten-Architektur: M1

Dimension „Einsatzzweck“: Dienste, Nutzerinteraktion

Dimension „Phasen der Entwicklung und Nutzung“: Operativ

Werkzeug: Context Server: API, Configuration Panel

Vorbedingung: Konzeptuelles Kontextmodell des AAL-Dienst, Dienstetyp-spezifisches Kontextmodell im Kontextserver operativ verfügbar.

Ergebnis: Implementierter kontextadaptiver AAL-Dienst, Einbindung des „Configuration Panel“ in den AAL-Dienst zur Konfiguration des kontextadaptiven Verhaltens durch den Bewohner.

Beschreibung: Es erfolgt eine Umsetzung des AAL-Dienstes auf der Basis der im Context Server verfügbaren API und des Configuration Panel. Die Umsetzung eines solchen AAL-Dienstes wird nachfolgend erläutert.

In Kap. 3.3.1 wurde beschrieben, dass die Kontextadaptivität nie die eigentliche Anwendungslogik eines AAL-Dienstes ausmacht. Vielmehr beschreibt sie eine Eigenschaft des Dienstes, die eigentliche Anwendungslogik aufgrund von Ereignissen aus der Umgebung selbständig auszulösen bzw. diese mit Kontextinformationen zu parametrisieren. Aufgrund dessen besteht die Realisierung des kontextadaptiven Verhaltens aus den folgenden Teilschritten.

- Realisierung der Anwendungslogik.
- Realisierung der Logik zur Erkennung von Ereignissen bzw. der Kontextlogik.
- Verknüpfung der Kontextlogik mit der Anwendungslogik.

Die Realisierung der Anwendungslogik unterscheidet sich nicht von der sonstiger Anwendungen. Daher wird hierauf nicht weiter eingegangen. Die Realisierung der Kontextlogik ist dagegen Gegenstand der in dieser Dissertation erarbeiteten Methodik. Dazu gehören die Kontextmodellierung auf der Ebene der Dienste und die Überführung in einen Kontextserver. Die Kontextinformationen können zur Verknüpfung mit der Anwendungslogik mittels der API des Kontextserver zur Verfügung gestellt werden. Die API des im Rahmen dieser Arbeit entwickelten Kontextserver ist im Anhang beschrieben.

Die Verknüpfung zwischen Anwendungslogik und Kontextereignissen kann unterschiedlich realisiert werden. Ein gängiges Konzept hierfür ist die Verknüp-

fung mittels des Event-Condition-Action-Paradigmas. Die zugrundeliegenden Kontextbedingungen werden in einem Event- und Condition-Teil beschrieben. Die Überwachung der im Event-Teil beschriebenen Kontextinformationen und die Benachrichtigung bei Ereignissen ist dabei Aufgabe des Kontextservers. Die Anwendungslogik wird im Action-Teil innerhalb des Dienstes umgesetzt und bei Erfüllung der Kontextbedingung angestoßen. Ein Beispiel für eine solche Umsetzung ist in [181] zu finden. Diese Art der Realisierung wird in der folgenden Abbildung verdeutlicht. Ein weiterer Ansatz ist die Betrachtung von Kontext als einen separaten Aspekt der Anwendungslogik und die Verwendung einer aspekt-orientierten Programmiersprache (AOP), z.B. AspectJ, zu dessen Implementierung. In [182] wird ein Überblick und eine Bewertung bestehender Ansätze gegeben. Eine Vertiefung in die Umsetzung einer kontextadaptiven Anwendung und die Verknüpfung zwischen Anwendungslogik und Kontextlogik ist hier nicht vorgesehen.

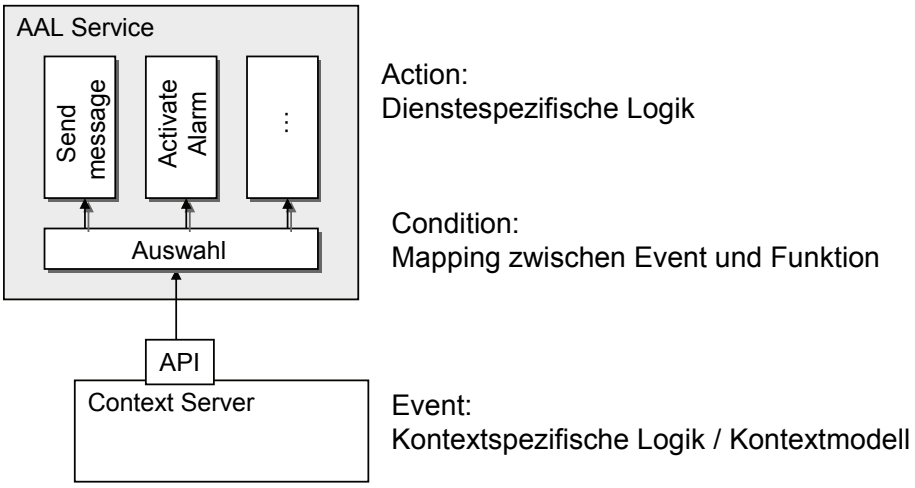


Abbildung 73: Realisierung des kontextadaptiven Verhaltens eines AAL-Dienstes

Generell ist eine Umsetzung eines AAL-Dienstes sowohl mit Hilfe des Event-Condition-Action-Paradigmas als auch über die Verwendung einer Aspekt-orientierten Programmiersprache möglich. Bei beiden Ansätzen erfolgt eine Trennung zwischen Kontextbedingungen, der eigentlichen Anwendungslogik und der Verknüpfung dieser Teile. Ein interessanter Aspekt bei der Verknüpfung der Anwendungslogik mit der Funktionalität des Kontextservers ist die Art der Schnittstelle in Form einer Kontextanfragesprache bzw. „Context Query Language“. Über eine solche Anfragesprache können Anfragen an das Kontextmodell und Kontextbedingungen unabhängig von der konkreten Implementierung beschrieben und an den Kontextserver gestellt werden. Es gibt eine Reihe von Ansätzen zur Definition einer solchen Sprache, jedoch sind diese derzeit auf konkrete Anwendungsszenarien spezialisiert und nicht allgemein durchgesetzt. In Hinblick auf eine allgemeine Context Query Language besteht aktuell noch Forschungsbedarf. Ein Beispiel und ein erster Schritt in Richtung

der Identifikation von Anforderungen an eine solche Sprache ist in [183] zu finden.

Neben dem von der Anwendungslogik vorgegebenen kontextadaptiven Verhalten besteht noch die Möglichkeit der Konfiguration durch den Bewohner. Das Interaktionskonzept zur Konfiguration (Kap. 7.3.3.3) ist unabhängig von dem jeweiligen AAL-Dienst. Spezifisch ist bei jedem AAL-Dienst der Ausschnitt aus der Situationstaxonomie, der für die Konfiguration verwendet werden kann. Daher wird die Umsetzung des Interaktionskonzepts als eine generische Komponente des Context Servers in Form des „Configuration Panel“ zur Einbindung in den AAL-Dienst zur Verfügung gestellt. Das „Configuration Panel“ wird in den AAL-Dienst eingebunden und mit der Dienstebeschreibung (Kap. 7.9.1) parametrisiert. Auf der Grundlage dieser Beschreibung und des operativen Kontextmodells auf der Ebene der Nutzerinteraktion (Kap. 7.8) erfolgt eine automatische Bereitstellung der benötigten Dialogbilder durch das „Configuration Panel“. Diese kann vom Bewohner für die Definition des kontextadaptiven Verhaltens einer Funktion des AAL-Dienstes verwendet werden. Die ausgewählte und verfeinerte Situationsbeschreibung überführt das „Configuration Panel“ in eine neue Definition eines Kontextraums in der Ebene der Dienste (Kap. 7.4). Die im Context Server erkannten Events im Kontextraum werden dann über das „Configuration Panel“ an den AAL-Dienst geleitet und dort zur Auslösung oder Parametrisierung einer Funktion verwendet. Das „Configuration Panel“ ist Bestandteil der Implementierung des Context Servers im Rahmen dieser Arbeit. Die Verwendung des „Configuration Panel“ wird in nachfolgender Abbildung verdeutlicht.

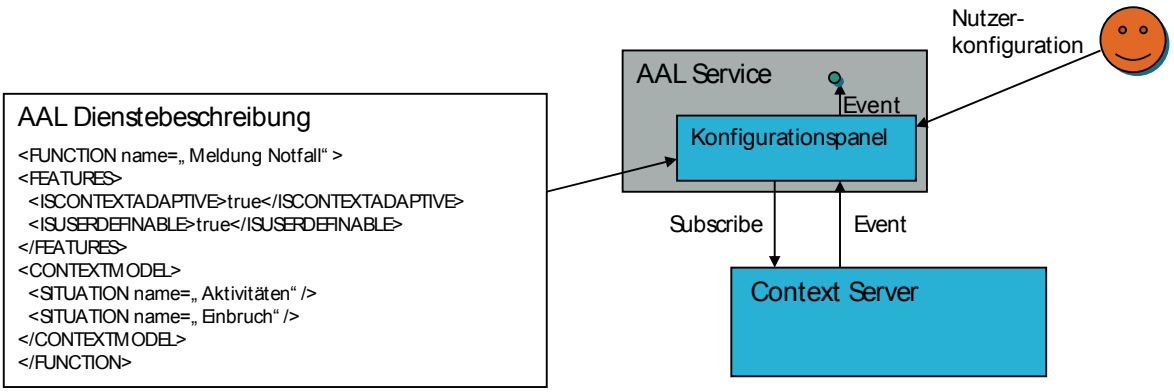


Abbildung 74: Realisierung des konfigurierbaren kontextadaptiven Verhaltens eines AAL-Dienstes

Durch Verwendung des operativen Kontextmodells auf Ebene der Dienste und der Nutzerinteraktion mittels der API und des „Configuration Panel“ wird das kontextadaptive Verhalten eines AAL-Dienstes umgesetzt.

8.2.4.4 Testen eines AAL-Dienstes

Ebene auf der Metadaten-Architektur: M1

Dimension „Einsatzzweck“: Dienste, Nutzerinteraktion

Dimension „Phasen der Entwicklung und Nutzung“: Informell, Operativ

Werkzeug: Testspezifikation, Context Server: API, Configuration Panel

Vorbedingung: Pflichtenheft, Dienstetyp-spezifisches Kontextmodell im Kontextserver operativ verfügbar, AAL-Dienst ist entwickelt und über die API bzw. Configuration Panel in die Kontextinfrastruktur eingebunden.

Ergebnis: Kontextadaptiver AAL-Dienst ist getestet.

Beschreibung: Basierend auf den im Pflichtenheft beschriebenen Anforderungen wird eine Testspezifikation erstellt. Das zu testende kontextadaptive Verhalten des AAL-Dienstes wird mit Hilfe der informellen Kontextbeschreibung (Kap. 6.6.3) in der Testspezifikation festgelegt. Die Umsetzung der Testfälle kann vom Kontextserver unterstützt werden, indem die in der Testspezifikation festgelegten Ausprägungen der Kontextinformationen simuliert werden. Die umgesetzte Kontextinfrastruktur greift dabei auf die Kontextinformationen im Kontextserver über die API zu. Zu Möglichkeiten des Testens für kontextadaptive Anwendungen gelten ebenso die Anmerkungen zu bestehenden Ansätzen wie in Kap. 8.2.1.4.

8.2.5 Instantiierung der AAL-Dienstemenge

Bei der Instantiierung der AAL-Dienstemenge werden die im Dienstemarkt verfügbaren AAL-Dienste in die intelligente häusliche Umgebung des Bewohners integriert und können vom Bewohner genutzt werden. In der nachfolgenden Abbildung sind die betroffenen Elemente in den beiden Dimensionen der Kontextmodellierung farblich hervorgehoben.

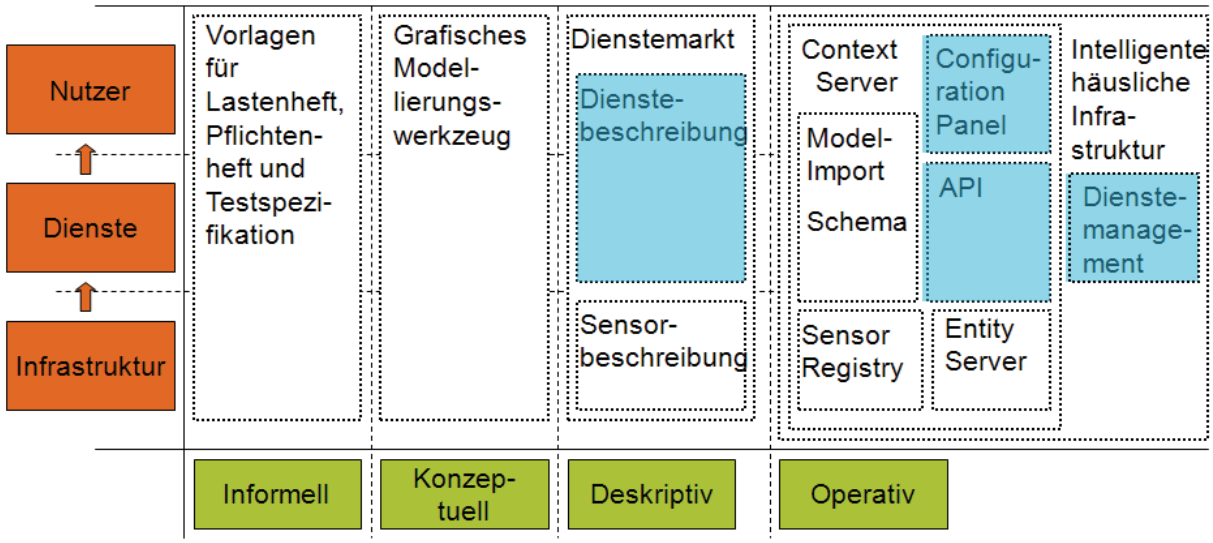


Abbildung 75: Modellelemente und Werkzeuge zur Instantiierung der AAL-Dienstemenge

Nachfolgend werden die einzelnen Schritte der Überführung und der Nutzung eines AAL-Dienstes unter Angabe der notwendigen Werkzeuge sowie des Verweises auf die Art der Kontextmodellierung beschrieben.

8.2.5.1 Bereitstellen eines AAL-Dienstes

Ebene auf der Metadaten-Architektur: M0

Dimension „Einsatzzweck“: Dienste, Nutzerinteraktion

Dimension „Phasen der Entwicklung und Nutzung“: Deskriptiv

Werkzeug: Modellierungswerkzeug, (Dienstemarkt)

Vorbedingung: Konzeptionelles Kontextmodell des AAL-Dienstes, AAL-Dienst ist implementiert.

Ergebnis: AAL-Dienst ist mitsamt zugehöriger Dienstebeschreibung verfügbar.

Beschreibung: Das konzeptionelle Kontextmodell enthält die notwendigen Informationen über die Anforderungen an das dienstetyp-spezifische Kontextmodell. Teile des Kontextmodells können über das Modellierungswerkzeug Teilfunktionen des AAL-Dienstes zugeordnet werden. Daraus kann das deskriptive Kontextmodell erzeugt werden. Dieses muss in die konkrete Dienste-

beschreibung überführt werden (Kap. 6.9.1, 7.9.1). Je nach Anwendungsszenario kann mit der Dienstebeschreibung der AAL-Dienst im Dienstemarkt registriert werden.

8.2.5.2 Übernahme eines AAL-Dienstes in die häusliche intelligente Umgebung

Ebene auf der Metadaten-Architektur: M0

Dimension „Einsatzzweck“: Dienste, Nutzerinteraktion

Dimension „Phasen der Entwicklung und Nutzung“: Deskriptiv, Operativ

Werkzeug: (Dienstemarkt), Dienstemanagement

Vorbedingung: AAL-Dienst ist verfügbar, instantiierte Kontextinfrastruktur ist operativ verfügbar.

Ergebnis: AAL-Dienst ist in die häusliche intelligente Umgebung eingebunden.

Beschreibung: Ein AAL-Dienst ist im Dienstemarkt verfügbar und kann aufgrund der zugehörigen Dienstebeschreibung gefunden werden. Die Funktionalität zum Zugriff auf den Dienstemarkt ist Bestandteil einer intelligenten häuslichen Infrastruktur. Bei der Auswahl eines AAL-Dienstes durch den Bewohner muss diesem das durch seine Kontextinfrastruktur unterstützte kontextadaptive Verhalten des ausgewählten Dienstes (Kap. 7.3.2) dargestellt werden. Dabei ist ein Abgleich zwischen den Anforderungen des AAL-Dienstes und den Möglichkeiten der Kontextinfrastruktur durchzuführen. Dieser ist ebenfalls eine Funktionalität der intelligenten häuslichen Infrastruktur. Die Aufnahme des AAL-Dienstes in die Kontextinfrastruktur erfolgt über die Komponente „Dienstemanagement“, die für die Verwaltung der Dienste und deren Betrieb zuständig ist.

8.2.5.3 Betrieb eines AAL-Dienstes in der häuslichen intelligenten Umgebung

Ebene auf der Metam-Architektur: M0

Dimension „Einsatzzweck“: Dienste, Nutzerinteraktion

Dimension „Phasen der Entwicklung und Nutzung“: Operativ

Werkzeug: Context Server: API, Configuration Panel

Vorbedingung: AAL-Dienst ist über die API bzw. Configuration Panel in die Kontextinfrastruktur eingebunden.

Ergebnis: Kontextadaptiver AAL-Dienst ist in der häuslichen intelligenten Umgebung verfügbar.

Beschreibung: Der AAL-Dienst wird zur Umsetzung des kontextadaptiven Verhaltens über das „Configuration Panel“ und die API an den Kontextserver angebunden. Darüber kann der AAL-Dienst die verfügbare Kontextinformationen abrufen und bei relevanten Ereignissen benachrichtigt werden. Diese Informationen kann der AAL-Dienst zur Anpassung des kontextadaptiven Verhaltens nutzen. Über das „Configuration Panel“ kann der Bewohner das kontextadaptive Verhalten des AAL-Dienstes konfigurieren.

Weiterer Bestandteil der intelligenten häuslichen Infrastruktur ist die Umsetzung der Kontrolle über die intelligente Umgebung (Kap. 7.3.1).

8.3 Beispiel „Umsetzung eines Notfall-Dienstes“

Nachfolgend wird die Methodik am Beispiel der Umsetzung eines Notfall-Dienstes demonstriert. Dieser wurde im Rahmen der Dissertation mit Hilfe der verfügbaren Werkzeuge umgesetzt. Das Beispiel beschränkt sich aber auf die Entwicklung und den Betrieb des AAL-Dienstes. Anhand des Beispiels soll auch die Nutzung des „Configuration Panel“ dargestellt werden.

In der Phase der Entwicklung der Kontextinfrastruktur wurde das im Anhang beschriebene Kontextmodell der Infrastruktur (A.2.1) im Context Server umgesetzt. In der Phase der Instantiierung der Kontextinfrastruktur wurde der Context Server mit einer Reihe von Dummy-Sensoren ausgestattet. Zu diesem Zweck wurde eine einfache Sensor Registry im Context Server realisiert. In der Phase der Entwicklung von AAL-Dienstetypen wurden die im Anhang beschriebenen dienstetyp-spezifischen Kontextmodelle (A.2.2) im Context Server umgesetzt. Auf dieser Basis erfolgt nun die Beschreibung des Notfall-Dienstes in der Phase der Entwicklung eines AAL-Dienstes.

Zunächst wird das kontextadaptive Verhalten des AAL-Dienstes im Rahmen des Lastenheftes beschrieben. Dieser soll folgende Funktionen zur Verfügung stellen.

K.01 Aufzeichnung relevanter Ereignisse

Funktion: Für die Sicherheit des Bewohners wichtige Ereignisse sollen in Form von Kontextinformationen aufgezeichnet werden.

Diese Funktion wird im Rahmen des Pflichtenhefts weiter konkretisiert.

K.01 Aufzeichnung relevanter Ereignisse

Funktion: Für die Sicherheit des Bewohners wichtige Ereignisse sollen in Form von Kontextinformationen aufgezeichnet werden.

Kontextbedingung: Die Aufzeichnungsfunktion soll bei einer Besuchssituation oder einer Einbruchssituation ausgelöst werden.

Anforderung an Qualität und Repräsentation: Sekundengenaue Erkennung der Situationen.

Zuverlässigkeit: Die Erkennung der Situationen muss zuverlässig erfolgen. Eine Fehlerkennung wird als ‚Nicht akzeptabel' eingestuft.

Definition durch Anwender: Der Anwender hat die Möglichkeit, die Art des zu überwachenden Ereignisses auszuwählen und weiter zu spezifizieren.

Aus der informellen Beschreibung der Kontextbedingungen werden die relevanten Ausschnitte aus den definierten konzeptuellen Kontextmodellen auf der Ebene der Dienste mit den Qualitätsanforderungen des Notfall-Dienstes ergänzt. Nachfolgend wird ein Beispiel für den definierten Kontextraum ‚Besuch' gegeben. Auf der Ebene der Nutzerinteraktion werden Elemente der bereits definierten Situationstaxonomie ausgewählt, die zur Konfiguration des Dienstes genutzt werden können: ‚Aktivitäten' und ‚Einbruch'. ‚Aktivitäten' ist dabei ein Vaterelement des Elements ‚Besuch' und kann noch weitere Situationsbeschreibungen beinhalten.

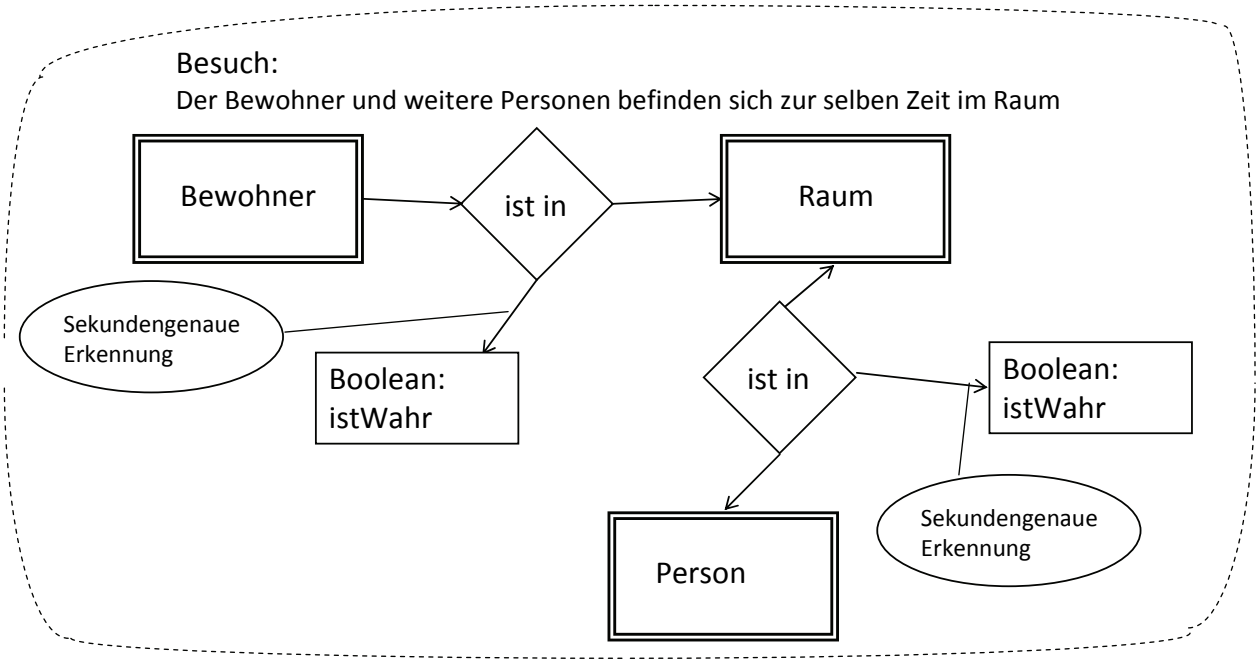


Abbildung 76: Konzeptuelles Kontextmodell des AAL-Dienstes

Die Aufzeichnungsfunktion des AAL-Dienstes wird implementiert. Anschließend wird in der Dienstebeschreibung das kontextadaptive und nutzerdefinierbare Verhalten des Dienstes beschrieben. Über diese Dienstebeschreibung wird der Dienst im Dienstemarkt verfügbar gemacht.

```
<APPLICATION name="Notfall" xmlns="
http://isst.fraunhofer.de/Application.xsd">
  <SERVICETYPE>security</SERVICETYPE>
   <FUNCTION name="Aufzeichnung Ereignis">
    <DESCRIPTION>Diese Funktion zeichnet ein Ereignis
auf</DESCRIPTION>
    <FEATURES>
     <ISCONTEXTADAPTIVE>true</ISCONTEXTADAPTIVE>
     <ISUSERDEFINABLE>true</ISUSERDEFINABLE>
    </FEATURES>
    <CONTEXTMODEL>
     <SITUATION name="Aktivitäten"/>
     <SITUATION name="Einbruch"/>
    </CONTEXTMODEL>
  </FUNCTION>
</APPLICATION>
```

In der Phase der Instantiierung der Dienstemenge wird der AAL-Dienst mit Hilfe der Diensteverwaltung in die intelligente häusliche Infrastruktur übernommen. Der Bewohner kann den Dienst nun mit Hilfe des „Configuration Panel“ konfigurieren. Nachfolgenden Abbildungen zeigen Screenshots vom umgesetzten Dienst. Zunächst wird die oberste Ebene der Situationstaxonomie dargestellt, die dem Bewohner zur Auswahl zur Verfügung gestellt wird.

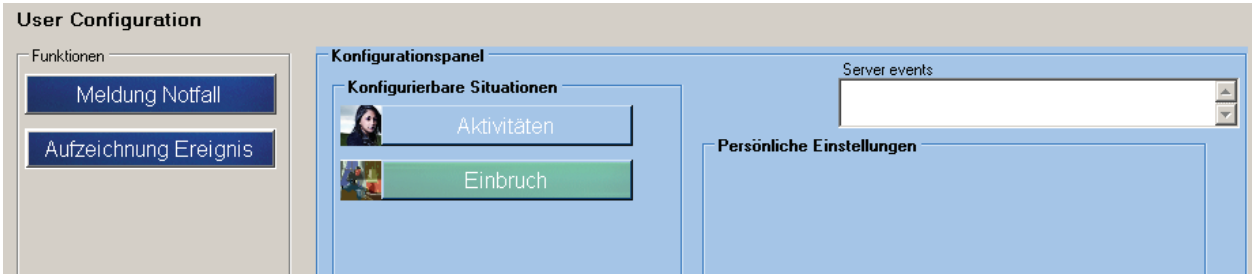


Abbildung 77: Einstieg in die Situationstaxonomie

Der Bewohner wählt nun die abstrakte Situationsbeschreibung „Aktivitäten“ aus. Die spezielle Situation „Besuch“ wird nun dargestellt.

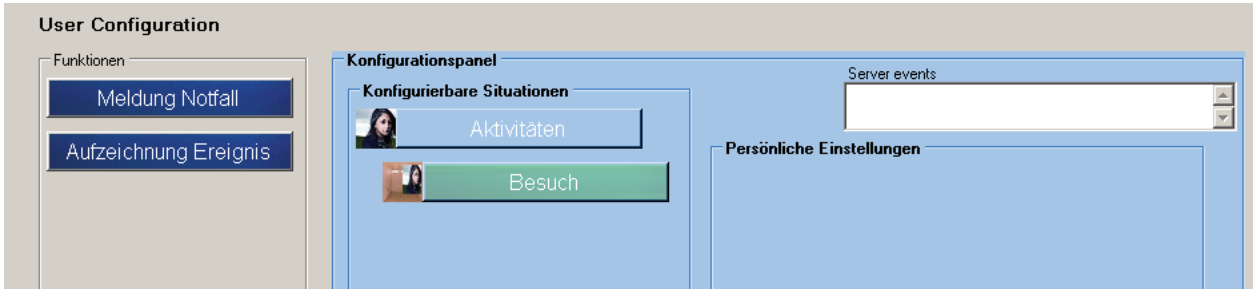


Abbildung 78: Auswahl der Situation „Aktivitäten“

Der Bewohner wählt die Situationsbeschreibung „Besuch“ aus. Es werden die weiteren Spezialisierungsmöglichkeiten sowie ein Button zur Übernahme der Konfiguration angezeigt. Möglichkeiten zur Spezialisierung sind die Auswahl einer spezialisierten Kontextentität zu „Personen“, sowie die Festlegung der Anzahl der Personen.

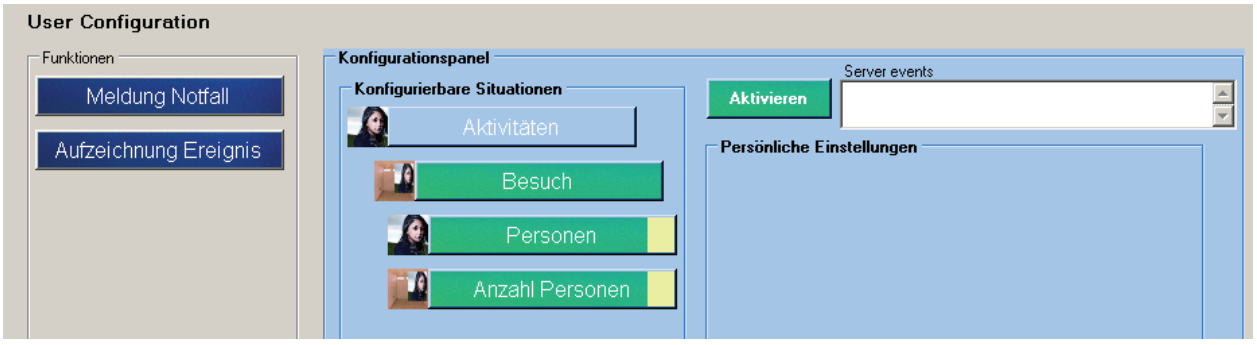


Abbildung 79: Auswahl der Situation „Besuch“

Nachfolgend schränkt der Bewohner die Art der Personen ein, die in die Situationserkennung einbezogen werden sollen.

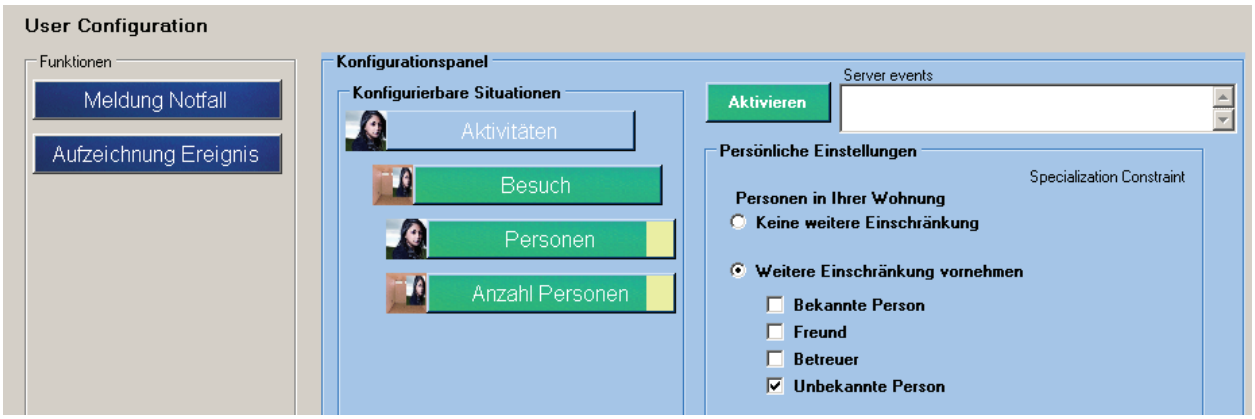


Abbildung 80: Einschränkung der Art der Personen

Die Konfiguration des AAL-Dienstes wird mittels Betätigung des Buttons „Aktivieren“ abgeschlossen.

In dem beschriebenen Beispiel beschränkt sich die Realisierung des Dienstes auf die Umsetzung der Anwendungslogik, die Definition der Qualitätsanforderungen an den relevanten Ausschnitten der Referenz-Kontextmodelle sowie die Anbindung der Anwendungslogik an die Kontextlogik mit Hilfe der Dienstebeschreibung und der Anbindung an den „Configuration Panel“.

8.4 Zusammenfassung

In diesem Kapitel wurden zunächst die Werkzeuge vorgestellt, die zur Umsetzung der Methodik benötigt werden. Eine Vielzahl dieser Werkzeuge wurde im Rahmen der Dissertation umgesetzt und steht für die Entwicklung der Kontextinfrastruktur und der AAL-Dienste zur Verfügung. Danach wurde die Methodik vorgestellt, die zunächst in fünf Schritten gegliedert wurde. Ihre Verwendung wurde für drei unterschiedlich komplexe Szenarien beschrieben. Anschließend wurde der Einsatz der Werkzeuge und der Bezug zur zweidimensionalen Kontextmodellierung in jedem Schritt aufgezeigt und dann die Verwendung der Methodik am Beispiel eines Notfall-Dienstes im Ausschnitt demonstriert.

9 Schlussbetrachtung

9.1 Zusammenfassung

Die Zielstellung der Dissertation ist die Erarbeitung einer Methodik für die Kontextmodellierung im Rahmen des AAL. Zunächst wurden die besonderen Anforderungen aus der Anwendungsdomäne AAL an die Kontextmodellierung erarbeitet, besonders die Trennung zwischen Kontextinfrastruktur und den Diensten, die Bereitstellung dienstetyp-spezifischer Kontextmodelle und ihre Erweiterbarkeit, die Bereitstellung einer nutzerspezifischen Kontextmodellierung sowie die Berücksichtigung der Phasen der Entwicklung und Nutzung von AAL-Diensten. Es wurde eine Reihe aktueller Ansätze zur Kontextmodellierung anhand der Anforderungen bewertet und der Bedarf für einen weitergehenden Ansatz erkannt. Der in dieser Arbeit entwickelte Ansatz basiert auf einer zweidimensionalen Kontextmodellierung von „Einsatzzweck“ sowie „Phasen der Entwicklung und Nutzung“. In der ersten Dimension werden drei Ebenen der Abstraktion festgelegt: „Infrastruktur“, „Dienste“ und „Nutzerinteraktion“ und in der zweiten vier Modelltypen identifiziert und den verschiedenen Phasen „Planung“, „Definition“, „Design“, „Implementation“, „Test“, „Bereitstellung“, „Übernahme“ sowie „Betrieb“ zugeordnet. Jeder dieser Modelltypen besitzt einen eigenen Fokus und eine eigene Darstellung. Beispielsweise definiert das konzeptuelle Kontextmodell eine graphische Repräsentation, der ein bereits bekannter Ansatz zur konzeptuellen Kontextmodellierung zugrunde liegt, welcher hier aber um fehlende Konzepte ergänzt wird. In die durch zwei Dimensionen aufgespannte Matrix werden die jeweils durch sie eingegrenzten Anforderungen eingeordnet sowie entsprechende Konzepte und Ausprägungen des Kontextmodells definiert. Die gemeinsamen Elemente des Kontextmodells sowie die Überführungen zwischen den einzelnen Elementen der Matrix werden durch das sie tragende Metamodell definiert. Die zweidimensionale Kontextmodellierung ist auch Bestandteil der Methodik, bei der sich fünf Schritte unterscheiden lassen: die Definition der Kontextinfrastruktur, die Instanziierung der Kontextinfrastruktur, die Definition der AAL-Dienstetypen, die Definition eines AAL-Dienstes, sowie die Instanziierung der AAL-Dienstemenge. Jede der Schritte beschreibt einen Teilschritt in der Realisierung einer intelligenten häuslichen Umgebung mit integrierten AAL-Diensten und definiert den notwendigen Ausschnitt aus dem zweidimensionalen Kontextmodell sowie die benötigten Werkzeuge, von denen die wesentlichen im Rahmen der Dissertation umgesetzt worden sind und so der Methodik zur Verfügung stehen. Die Definition der Schritte ermöglicht auch eine flexible Nutzung der Methodik für unterschiedlich komplexe Szenarien von kontextadaptiven AAL-Diensten.

9.2 Erreichte Ergebnisse

Folgende Ergebnisse wurden im Rahmen der Dissertation erreicht, die über den aktuellen Stand der Wissenschaft hinausgehen:

- Identifikation der Anforderungen an die Kontextmodellierung im AAL.
- Definition einer zweidimensionalen Kontextmodellierung im AAL.
- Definition einer Methodik zur Realisierung kontextadaptiver Dienste im AAL.
- Konzeption der Nutzerinteraktion zur Konfiguration des kontextadaptiven Verhaltens von AAL-Diensten (End User Context Modeling).
- Definition initialer konkreter Kontextmodelle im AAL auf der Basis der zweidimensionalen Kontextmodellierung (siehe Anhang).
- Realisierung von Werkzeugen bzw. Komponenten zur Umsetzung der definierten Methodik.

Mit Hilfe dieser Ergebnisse können kontextadaptive AAL-Dienste methodisch entwickelt werden.

9.3 Kritische Bewertung

Die Grundlage für die Dissertation bilden theoretische Annahmen über die Anforderungen an die Kontextmodellierung im AAL, die auf der Analyse von Veröffentlichungen basieren. Eine weitergehende Grundannahme ist das Funktionieren eines Geschäftsmodells für die Entwicklung und Bereitstellung von AAL-Diensten über einen Dienstemarkt. Aufgrund der Neuartigkeit des Themengebiets existieren hierfür nur wenige praktische Erfahrungen, die in diese Annahmen eingehen können. Sie sind durch weitere praktische Arbeiten in der Entwicklung von konkreten AAL-Diensten sowie eines AAL-Dienstemarktes zu bestätigen.

Trotz solcher Einschränkungen konnte diese Methodik bereits in wesentlichen Teilen erfolgreich eingesetzt werden. Die konzeptuelle Kontextmodellierung fand beispielsweise Anwendung, um das kontextadaptive Verhalten der bereits am Fraunhofer ISST realisierten AAL-Dienste im Reverse Engineering zu dokumentieren. Die graphische Notation konnte mit Hilfe des Modellierungswerkzeugs von einem Entwickler verwendet werden, um die der Kontextadaptivität zugrundeliegenden Kontextinformationen formalisiert zu beschreiben. Dazu reichte eine einstündige Einführung des Entwicklers in die Methodik und das Werkzeug. Die entstandenen Kontextmodelle sind dann genutzt worden, um

diese im Entwicklungsteam zu diskutieren und neue Ideen zu entwickeln. In einem weiteren Einsatz wurde ein einfaches Kontextmodell für ein konkretes Projekt am Fraunhofer ISST entsprechend der Methodik entwickelt und in ein operatives Kontextmodell im Context Server überführt. Anschließend wurden reale Sensoren mit Hilfe der Sensorbeschreibung in das Kontextmodell im Context Server eingebunden. Das Funktionieren der Erkennung von Situationen konnte mit Hilfe der Sensorik getestet werden.

Auf der Ebene der Nutzerinteraktion wurden Konzepte für die Kontrolle der intelligenten Umgebung, die Auswahl der AAL-Dienste und für die Konfiguration des kontextadaptiven Verhaltens durch den Bewohner erstellt. Erstere sind ein initialer Aufschlag und wurden nicht weiter evaluiert. Diese müssen in weiteren Arbeiten vertieft und evaluiert werden. Das Interaktionskonzept zur Konfiguration wurde durch einen Labortest bewertet. Aufgrund des eingeschränkten Versuchsaufbaus können die Ergebnisse lediglich als Indikator für die tatsächliche Nutzbarkeit durch den Endanwender verwendet werden. Auch hier sind noch weitere Folgearbeiten notwendig, was eventuell zu einer Überarbeitung des Kontext-Metamodells auf dieser Ebene führen kann.

9.4 Ausblick

Auf den Ergebnissen dieser Dissertation können eine Reihe von praktischen und theoretischen Folgearbeiten aufgebaut werden. Ein Aspekt ist beispielsweise die Betrachtung der konkreten Eigenschaften von Sensoren und die Definition einer Sensorbeschreibung sowie deren Umsetzung in einer Sensor Registry. Ein weiterer Aspekt ist die Realisierung der intelligenten häuslichen Infrastruktur, deren Funktionalität in dieser Dissertation zum Teil beschrieben wurde. Die Festlegung des darin umzusetzenden strukturellen kontextadaptiven Verhaltens muss in das Kontextmodell auf der Ebene der Infrastruktur einfließen, welches bereits im Anhang initial definiert wurde. Hinzu kommen Aspekte, die weitere Konkretisierung von Kontextdimensionen betreffen sowie deren mögliche Repräsentationsformen und Qualitätseigenschaften. Auch die Aspekte der Nutzungsbedingungen in Bezug auf die Wahrung der Privatsphäre oder die entstehenden Kosten sollten noch eingehender betrachtet werden. Ein weiterer Aspekt ist die Identifikation von sinnvollen Diensten in den einzelnen Dienstetypen und die Ableitung von Referenz-Kontextmodellen. Weitergehend sollten auch Geschäftsmodelle für die der Dissertation zugrundeliegende Zielvision eines AAL-Dienstemarktes identifiziert werden. Zudem sollte die Bewertung der Interaktionskonzepte einer tragfähigen Versuchsordnung unterzogen werden. Diese Folgearbeiten sind schrittweise nach Möglichkeit anhand konkreter Projekte zu leisten.

A Anhang

Nachfolgend sind im Anhang folgende Zusatzinformationen zu finden:

- Informationen zur Implementierung des Kontextserver
- Initiale Kontextmodelle auf Ebene der Infrastruktur und der Dienste für das Ambient Assisted Living

A.1 Kontextserver

Nachfolgend werden die wesentlichen Konzepte der Umsetzung des Kontextserver beschrieben. Dazu gehören die Architektur, das Design, sowie eine Beschreibung der API.

A.1.1 Architektur

In der nachfolgenden Abbildung wird eine Übersicht der Grobarchitektur der realisierten Kontext-Infrastruktur und dem darin befindlichen Kontextserver gegeben. Die Funktionalität der Komponenten dieser Architektur ist bereits in Kap. 8.1 beschrieben worden. Die Hauptkomponenten „Context Server“, „Sensor Registry“ und „Entity Server“ sind voneinander abhängig. Der „Context Server“ verwaltet die konkreten Kontextmodelle. Diesem sind die definierten Modellelemente bekannt. Der „Entity Server“ greift auf den „Context Server“ zu, um die definierten Modellelemente auf Ebene der Infrastruktur für die Registrierung der konkreten Kontextentitäten zu ermitteln. Wird hier eine neue Kontextentität registriert, so wird diese zusammen mit der ID der Entität an den „Context Server“ übermittelt und dort in das Kontextmodell aufgenommen. Die „Sensor Registry“ greift auf den „Context Server“ zu um die definierten Modellelemente auf Ebene der Infrastruktur für die Registrierung der Sensoren zu ermitteln. Sie greift auch auf den „Entity Server“ zu, um die IDs der Kontextentitäten zu ermitteln, denen die durch die Sensoren gelieferten Kontextinformationen zugeordnet werden. Neue Kontextinformationen werden von der „Sensor Registry“ an den „Context Server“ übermittelt.

Die Hauptkomponenten sind in der Umsetzung als eigenständige Server unter Nutzung der Programmiersprache C# und des Windows Communication Frameworks (WCF) des Windows Betriebssystems realisiert worden. In der Konsequenz ist ein Betrieb dieser Komponenten auf einem alternativen Betriebssystem nicht möglich. Das WCF ermöglicht eine deklarative Festlegung der konkreten Kommunikationskanäle. Auf diese Weise können die Dienste als

Web-Services angeboten werden. Eine alternative Bereitstellung dieser Komponenten ist in Form von Windows Enterprise Services möglich. Zur Verwaltung der Modelle und der Modellinformationen benötigen die Komponenten ein relationales Datenbankmanagementsystem mit erzeugenden, schreibenden und lesenden Zugriffsrechten. In der aktuellen Implementierung wird der Microsoft SQL-Server ab der Version 2005 unterstützt. Die Kommunikation zwischen den einzelnen Komponenten erfolgt asynchron unter Nutzung der Microsoft Message Queue. Diese muss auf dem Betriebsystem installiert sein. Die einzelnen Anwendungen und auch das „Configuration Panel“ greifen synchron über den angebotenen Kommunikationskanal oder auch asynchron mittels Messages auf den „Context Server“ zu.

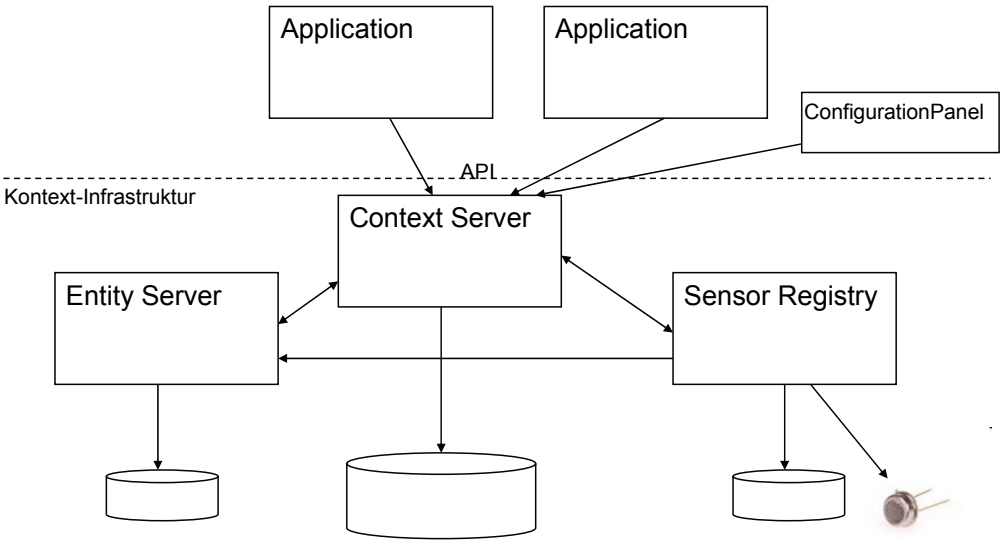


Abbildung: Grobarchitektur des Context Server

A.1.2 Design

Die Implementierung des Kontextservers besteht aus sechs wesentlichen Packages, die nachfolgend beschrieben werden:

- De.Fraunhofer.ISST.ContextServer.ContextInformationProvision: Die Klassen in diesem Package sind für den Empfang und die weitere Verarbeitung von Kontextinformationen aus den Sensoren zuständig. Hier ist das Format definiert, in welchem die Kontextinformationen von den Sensoren geliefert werden müssen (Sub-Package „ContextInformation“). Weiterhin ist hier eine rudimentäre Sensor Registry implementiert worden, die anstelle einer externen Sensor Registry genutzt werden kann (Sub-Package „InternalSensor“).

- De.Fraunhofer.ISST.ContextServer.ContextModelManagement: Die Klassen in diesem Package realisieren datenbankunabhängig die Methoden für die Verwaltung der einzelnen Kontextmodelle und für den Zugriff auf die darin vorhandenen Kontextinformationen. In entsprechenden Sub-Packages „Application“, „Infrastructure“ und „Interaction“ erfolgt dieses separat für die einzelnen Modellebenen.

- De.Fraunhofer.ISST.ContextServer.DatabaseManagement: Die Klassen in diesem Package realisieren die Abbildung der Modellverwaltung auf die zugrundeliegende Datenbank. Im Sub-Package „General“ werden allgemeine Zugriffsmethoden wie die Überprüfung des Vorhandenseins der Datenbank realisiert. Im Sub-Package „DatabaseSchema“ werden die Methoden zur Generierung und Aktualisierung der benötigten Tabellen und Sichten bereitgestellt. Dieses erfolgt separat für die einzelnen Modellebenen in jeweils weiteren Sub-Packages. In den Sub-Packages „Application“, „Infrastructure“ und „Interaction“ werden letztendlich die Methoden für den Zugriff auf die konkreten Kontextinformationen realisiert. In einem Interface (z.B. IDatabaseAccess) werden die zu realisierenden Methoden beschrieben. Die konkrete Realisierung erfolgt dann spezifisch für das jeweilige Datenbankmanagementsystem (z.B. DatabaseIntAccessMsSQL).

- De.Fraunhofer.ISST.ContextServer.ExchangeObject: Die Klassen in diesem Package realisieren die Objekte, die an der Schnittstelle zur Anwendung oder auch den anderen Komponenten der Architektur benötigt werden.

- De.Fraunhofer.ISST.ContextServer.RessourceManagement: Die Klassen in diesem Package verwalten die Ressourcen des Context Server.

- De.Fraunhofer.ISST.ContextServer.SubscriptionManagement: Die Klassen in diesem Package verwalten die Subscriptions und die Zuordnung dieser zu den Message Queues, über welche entsprechende Ereignisse gemeldet werden.

- De.Fraunhofer.ISST.ContextServer.Tools: In diesem Package werden benötigte Tool-Klassen verwaltet.

A.1.3 API

Die API ermöglicht die Anbindung eines AAL-Dienstes an den Kontextserver. Eine solche Anbindung ist nur auf Ebene der Dienste des Kontextmodells möglich. Das „Configuration Panel“ vorgesehen greift ebenfalls mittels der API auf den Kontextserver zu. Für den Austausch von Informationen zwischen „Context Entity“, „Sensor Registry“ und „Context Server“ sind Methoden zum

Zugriff auf das Kontextmodell auf Ebene der Infrastruktur in der API vorhanden. Nachfolgend werden nur die API-Methoden auf Ebene der „Dienste“ beschrieben. Es sind sowohl Methoden für den Zugriff auf das Kontextmodell (Ebene M1) als auch auf die darin enthaltenen Instanzen (Ebene M0) vorhanden. Folgende Methoden ermöglichen den synchronen Zugriff auf die Instanzen:

- Suche nach einer konkreten Kontextentität.
- Suche nach einer konkreten Kontextrelation.
- Suche nach konkreten Kontextentitäten, die in einem Kontextraum vorhanden sind.

Zudem sind folgende Methoden vorhanden, mit denen eine asynchrone Benachrichtigung bei folgenden Ereignissen bestellt und wieder abbestellt werden kann:

- Benachrichtigung bei Änderungen bei einer konkreten Kontextentität.
- Benachrichtigung bei Änderungen bei einer konkreten Kontextrelation.
- Benachrichtigung bei Änderungen in einem Kontextraum.

Nachfolgend werden die einzelnen Methoden im Detail und am Beispiel beschrieben. Die Definition der Suchausdrücke in den Methoden erfolgt objektorientiert mit Hilfe der in der API definierten Klassen. Dieses erschwert die Lesbarkeit der Suchausdrücke. Alternativ dazu könnte eine Kontextanfragesprache (Context Query Language) bzw. eine entsprechende Grammatik umgesetzt werden. Es existiert allerdings derzeit noch keine solche Sprache, die auf dem Kontext-Metamodell angewendet werden kann. Die Entwicklung einer Kontextanfragesprache und die Umsetzung im Kontextserver kann in nachfolgenden Arbeiten durchgeführt werden. Die Beschreibung der Methoden erfolgt anhand der Methodensignatur in C#.

A.1.3.1 Suche nach einer konkreten Kontextentität

Die Suche nach einer konkreten Kontextentität erfolgt durch Aufruf der folgenden Methode:

public ResultContainer ApplicationFindAndGetEntities(string applicationName, EntityQueryObject queryObject);

Die Methode besitzt folgende Eingabeparameter:

- *string applicationName* – Der Name des Dienstetyps zum Kontextmodell.

- *EntityQueryObject queryObject* – Ein Objekt, das den Suchausdruck für die Kontextentität enthält.

Als Ergebnis des Methodenaufrufs wird ein Objekt der Klasse „ResultContainer“ zurückgegeben. Dieser enthält Repräsentationen der gefundenen Instanzen der Kontextentitäten mitsamt den zugehörigen Kontextattributen. Diese Klasse besitzt folgende C#-Eigenschaft:

- *Hashtable results* - Hashtabelle, in welcher die das Ergebnis repräsentierende Instanzen der Klasse „EntityResultObject“ vorhanden sind.

Die Klasse „EntityResultObject“ repräsentiert eine konkrete Instanz einer Kontextentität und besitzt folgende C#-Eigenschaften:

- *string entityType* – Zuordnung der Instanz zum Entitätstyp (Kontextentität auf Ebene M1).

- *string globalID* – Eindeutige ID zur Instanz der Kontextentität.

- *ArrayList attributes* – Liste mit zugehörigen Kontextattributen (Klasse „AttributeResultObject“).

Die Klasse „AttributeResultObject“ repräsentiert eine konkrete Instanz eines Kontextattributs und besitzt folgende C#-Eigenschaften:

- *string attributeName* – Name des Kontextattributs.

- *object attributeValue* – Im Kontextattribut enthaltene Kontextinformation.

- *string implementation* – Name des Datentyps, durch den das Kontextattribut repräsentiert wird.

- *bool isRange* – Besitzt das Kontextattribut die „isRange“-Eigenschaft?

Die Klasse „EntityQueryObject“ repräsentiert den Suchausdruck für die Kontextentität. Über den folgenden Konstruktor kann ein neuer Suchausdruck definiert werden:

public EntityQueryObject(string entityType);

Der Konstruktor besitzt folgenden Parameter:

- *string entityType:* Der Name des Entitätstypen bzw. der Kontextentität auf Ebene M1. Dieser muss angegeben werden.

Über folgende C#-Eigenschaften können weitere Einschränkungen vorgenommen werden:

- *string globalID* – Die eindeutige ID einer Instanz der Kontextentität kann für die Suche verwendet werden.

Zusätzliche Suchausdrücke können über folgende Methode einer Suchinstanz hinzugefügt werden:

public void AddQueryCondition(IQueryConstructObject queryCondition);

Diese besitzt folgenden Eingabeparameter:

- *IQueryConstructObject queryCondition* – Instanz eines Suchausdrucks. Jeder solche Suchausdruck definiert eine Suchbedingung auf einem Kontextattribut der Kontextentität. Das Interface „IQueryConstructObject“ kann selbst nicht instantiiert werden. Die folgenden Klassen implementieren das Interface und können verwendet werden: „QueryAttributeObject“ und „QueryContainerObject“.

Alle über diese Methode hinzugefügten Suchausdrücke werden über einen AND-Operator miteinander verknüpft.

Die Klasse „QueryAttributeObject“ ermöglicht die Definition von Bedingungen auf einem Kontextattribut. Über folgenden Konstruktor kann eine neue Instanz erzeugt werden:

public QueryAttributeObject(string attributeName, int function, string attributeValue);

Der Konstruktor besitzt folgende Parameter:

- *string attributeName* – Name des Kontextattributs.
- *int function* – Verweis auf die zu verwendende Vergleichsfunktion.
- *string attributeValue* – Vergleichswert zum Kontextattribut.

Als Verweis zur Vergleichsfunktion besitzt die Klasse folgende Konstanten, die im Konstruktor verwendet werden können:

- *public const int FUNCTION_EQUAL = 0;*
- *public const int FUNCTION_NOT_EQUAL = 1;*
- *public const int FUNCTION_GREATER_THAN = 2;*

- *public const int FUNCTION_LESS_THAN = 3;*

- *public const int FUNCTION_IS_IN = 4;*

Die Klasse “QueryContainerObject” ermöglicht die boolsche Verknüpfung von einzelnen Suchausdrücken. Über den Konstruktor kann der boolsche Operator zur Instanz festgelegt werden:

public QueryContainerObject(int operation);

Der Konstruktor besitzt folgenden Parameter:

- *int operation* – Verweis auf den boolschen Operator.

Als Verweis zum boolschen Operator sind in der Klasse folgende Konstanten, die im Konstruktor verwendet werden können:

- *public const int OPERATOR_AND = 0;*

- *public const int OPERATOR_OR = 1;*

Ein Suchausdruck kann der Instanz mit folgender Methode hinzugefügt werden:

public void AddQueryCondition(IQueryConstructObject queryCondition);

Diese Methode besitzt den folgenden Eingabeparameter:

- *IQueryConstructObject queryCondition* – Suchausdruck, der dem „QueryContainerObject“ hinzugefügt wird. Dieses kann eine Instanz eines „QueryAttributeObject“ oder eine weitere Instanz eines „QueryContainerObject“ sein.

Alle mit der Methode dem Containerobjekt zugefügten Suchausdrücke werden mit dem definierten boolschen Operator miteinander verknüpft.

Nachfolgend wird ein Beispiel für die Definition einer Suche nach einer Kontextentität gegeben.

```
EntityQueryObject queryObject = new EntityQueryObject("Patient");

QueryAttributeObject attributeObject = new QueryAttributeObject("blutdruck",
QueryAttributeObject.FUNCTION_GREATER_THAN, "120");

queryObject.AddQueryCondition(attributeObject);
```

```
QueryContainerObject containerObject = new QueryContainerOb-
ject(QueryContainerObject.OPERATOR_OR);

queryObject.AddQueryCondition(containerObject);

QueryAttributeObject attributeObject1 = new QueryAttribute-
Object("blutzucker", QueryAttributeObject.FUNCTION_LESS_THAN, "180");

containerObject.AddQueryCondition(attributeObject1);

QueryAttributeObject attributeObject2 = new QueryAttributeObject("puls",
QueryAttributeObject.FUNCTION_NOT_EQUAL, "130");

containerObject.AddQueryCondition(attributeObject2);
```

Diese Definition entspricht dem folgenden Suchausdruck:

(AND (blutdruck>120) (OR (blutzucker < 180) (puls <> 130)))

Das Ergebnis der Suche kann wie folgt ausgelesen werden:

```
ResultContainer resultContainer = contextServerIn-
stance.FindAndGetEntities("health", queryObject);

resultTextbox.AppendText("Number results:" + resultContainer.results.Count +
cr.ToString());

foreach (EntityResultObject resultObject in resultContainer.results.Values)
        {
            resultTextbox.AppendText("EntityID:" + resultObject.globalID +
cr.ToString());
            resultTextbox.AppendText("EntityType:" + resultObject.entityType +
cr.ToString());
            foreach (AttributeResultObject attributeResult in
resultObject.attributes)
            {
                resultTextbox.AppendText("  " + attributeResult.attributeName + "
= " + attributeResult.attributeValue + " (" + attributeResult.implementation +
",isRange=" + attributeResult.isRange.ToString() + ")" + cr.ToString());
            }
        }
```

A.1.3.2 Suche nach einer konkreten Kontextrelation

Die Suche nach einer konkreten Kontextrelation erfolgt durch Aufruf der folgenden Methode:

public ResultContainer ApplicationFindAndGetRelations(string applicationName, RelationQueryObject queryObject)

Die Methode besitzt folgende Eingabeparameter:

- *string applicationName* – Der Name des Dienstetyps zum Kontextmodell.
- *RelationQueryObject queryObject* – Ein Objekt, das den Suchausdruck für die Kontextrelation enthält.

Als Ergebnis des Methodenaufrufs wird ein Objekt der Klasse „ResultContainer“ zurückgegeben. Dieser enthält Repräsentationen der gefundenen Instanzen der Kontextrelationen mitsamt den zugehörigen Kontextattributen. Diese Klasse besitzt folgende C#-Eigenschaft:

- *Hashtable results* - Hashtabelle, in welcher die das Ergebnis repräsentierende Instanzen der Klasse „RelationResultObject“ vorhanden sind.

Die Klasse „RelationResultObject“ repräsentiert eine konkrete Instanz einer Kontextrelation und besitzt folgende C#-Eigenschaften:

- *string fromEntityType* – Entitätstyp der ausgehenden Kontextentität der Kontextrelation (Kontextentität auf Ebene M1).
- *string fromGlobalID* – Eindeutige ID zur Instanz der ausgehenden Kontextentität.
- *string toEntityType* – Entitätstyp der eingehenden Kontextentität der Kontextrelation (Kontextentität auf Ebene M1).
- *string toGlobalID* – Eindeutige ID zur Instanz der eingehenden Kontextentität.
- *ArrayList attributes* – Liste mit zugehörigen Kontextattributen (Klasse „AttributeResultObject“).

Die Klasse „AttributeResultObject“ repräsentiert eine konkrete Instanz eines Kontextattributs und besitzt folgende C#-Eigenschaften:

- *string attributeName* – Name des Kontextattributs.

- *object attributeValue* – Im Kontextattribut enthaltene Kontextinformation.

- *string implementation* – Name des Datentyps, durch den das Kontextattribut repräsentiert wird.

- *bool isRange* – Besitzt das Kontextattribut die „isRange“-Eigenschaft?

Die Klasse „RelationQueryObject“ repräsentiert den Suchausdruck für die Kontextrelation. Über den folgenden Konstruktor kann ein neuer Suchausdruck definiert werden:

public RelationQueryObject(string relationType);

Der Konstruktor besitzt folgenden Parameter:

- *string relationType:* Der Name des Relationstyps bzw. der Kontextrelation auf Ebene M1. Dieser muss angegeben werden.

Über folgende C#-Eigenschaften können weitere Einschränkungen vorgenommen werden:

- *string fromEntityType* – Der Entitätstyp der ausgehenden Kontextentität kann für die Suche vorgegeben werden.

- *string fromGlobalID* – Die eindeutige ID einer Instanz der ausgehenden Kontextentität kann für die Suche vorgegeben werden.

- *string toEntityType* – Der Entitätstyp der eingehenden Kontextentität kann für die Suche vorgegeben werden.

- *string toGlobalID* – Die eindeutige ID einer Instanz der eingehenden Kontextentität kann für die Suche vorgegeben werden.

Zusätzliche Suchausdrücke können über folgende Methode einer Suchinstanz hinzugefügt werden:

public void AddQueryCondition(IQueryConstructObject queryCondition);

Diese besitzt folgenden Eingabeparameter:

- *IQueryConstructObject queryCondition* – Instanz eines Suchausdrucks. Jeder solche Suchausdruck definiert eine Suchbedingung auf einem Kontextattribut der Kontextentität. Das Interface „IQueryConstructObject“ kann selbst nicht instantiiert werden. Die folgenden Klassen implementieren das Interface und können verwendet werden: „QueryAttributeObject“ und „QueryContainerObject“.

Alle über diese Methode hinzugefügten Suchausdrücke werden über einen AND-Operator miteinander verknüpft.

Die Klasse „QueryAttributeObject“ ermöglicht die Definition von Bedingungen auf einem Kontextattribut. Über folgenden Konstruktor kann eine neue Instanz erzeugt werden:

public QueryAttributeObject(string attributeName, int function, string attributeValue);

Der Konstruktor besitzt folgende Parameter:

- *string attributeName* – Name des Kontextattributs.
- *int function* – Verweis auf die zu verwendende Vergleichsfunktion.
- *string attributeValue* – Vergleichswert zum Kontextattribut.

Als Verweis zur Vergleichsfunktion besitzt die Klasse folgende Konstanten, die im Konstruktor verwendet werden können:

- *public const int FUNCTION_EQUAL = 0;*
- *public const int FUNCTION_NOT_EQUAL = 1;*
- *public const int FUNCTION_GREATER_THAN = 2;*
- *public const int FUNCTION_LESS_THAN = 3;*
- *public const int FUNCTION_IS_IN = 4;*

Die Klasse “QueryContainerObject” ermöglicht die boolsche Verknüpfung von einzelnen Suchausdrücken. Über den Konstruktor kann der boolsche Operator zur Instanz festgelegt werden:

public QueryContainerObject(int operation);

Der Konstruktor besitzt folgenden Parameter:

- *int operation* – Verweis auf den boolschen Operator.

Als Verweis zum boolschen Operator sind in der Klasse folgende Konstanten, die im Konstruktor verwendet werden können:

- *public const int OPERATOR_AND = 0;*

- *public const int OPERATOR_OR = 1;*

Ein Suchausdruck kann der Instanz mit folgender Methode hinzugefügt werden:

public void AddQueryCondition(IQueryConstructObject queryCondition);

Diese Methode besitzt den folgenden Eingabeparameter:

- *IQueryConstructObject queryCondition* – Suchausdruck, der dem „QueryContainerObject“ hinzugefügt wird. Dieses kann eine Instanz eines „QueryAttributeObject“ oder eine weitere Instanz eines „QueryContainerObject“ sein.

Alle mit der Methode dem Containerobjekt zugefügten Suchausdrücke werden mit dem definierten boolschen Operator miteinander verknüpft.

Nachfolgend wird ein Beispiel für die Definition einer Suche nach einer Kontextentität gegeben.

```
RelationQueryObject relationQueryObject = new RelationQueryObject("Patient_hatEingenommen_EinnehmbarerGegenstand");

relationQueryObject.toEntityType = "Fluessigkeit";

relationQueryObject.toGlobalID = "Fluessigkeit:1";

QueryAttributeObject attributeObject = new QueryAttributeObject("istWahr", QueryAttributeObject.FUNCTION_EQUAL, "True");

relationQueryObject.AddQueryCondition(attributeObject);
```

In diesem Beispiel wird nach eine einer Instanz der Kontextrelation „Patient_hatEingenommen_EinnehmbarerGegenstand“ gesucht, wobei der einnehmbare Gegenstand eine Flüssigkeit mit der ID „Fluessigkeit:1“ ist. Das Ergebnis der Suche kann wie folgt ausgelesen werden:

```
ResultContainer resultContainer = contextServerInstance.FindAndGetRelations("health", relationQueryObject);

resultTextbox.AppendText("Number results:" + resultContainer.results.Count + cr.ToString());

foreach (RelationResultObject resultObject in resultContainer.results.Values)
        {
```

```
            resultTextbox.AppendText("From-EntityID:" +
resultObject.fromGlobalID + cr.ToString());
            resultTextbox.AppendText("From-EntityType:" +
resultObject.fromEntityType + cr.ToString());
            resultTextbox.AppendText("To-EntityID:" + resultObject.toGlobalID +
cr.ToString());
            resultTextbox.AppendText("To-EntityID:" + resultObject.toEntityType
+ cr.ToString());
            foreach (AttributeResultObject attributeResult in
resultObject.attributes)
            {
               resultTextbox.AppendText("  " + attributeResult.attributeName + "
= " + attributeResult.attributeValue + " (" + attributeResult.implementation +
",isRange=" + attributeResult.isRange.ToString() + ")" + cr.ToString());
            }
        }
```

A.1.3.3 Suche nach konkreten Kontextentitäten in einem Kontextraum

Mit Hilfe dieser Methode kann ermittelt werden welche konkreten Kontextentitäten in einem Kontextraum enthalten sind, z.B. in welcher Wohnung ein Einbruch stattfindet.

Die Suche nach einer konkreten Kontextentität in einem Kontextraum erfolgt durch Aufruf der folgenden Methode:

public ResultContainer ApplicationFindEntitiesInContextSpace(string applicationName, SpaceQueryObject queryObject)

Die Methode besitzt folgende Eingabeparameter:

- *string applicationName* – Der Name des Dienstetyps zum Kontextmodell.
- *SpaceQueryObject queryObject* – Ein Objekt, das den Suchausdruck für den Kontextraum enthält.

Als Ergebnis des Methodenaufrufs wird ein Objekt der Klasse „ResultContainer“ zurückgegeben. Dieser enthält Repräsentationen der im Kontextraum gefundenen Instanzen der Kontextentität mitsamt den zugehörigen Kontextattributen. Diese Klasse besitzt folgende C#-Eigenschaft:

- *Hashtable results* - Hashtabelle, in welcher die das Ergebnis repräsentierende Instanzen der Klasse „SpaceResultObject“ vorhanden sind.

Die Klasse „SpaceResultObject“ repräsentiert eine konkrete Instanz eines Kontextraums und besitzt folgende C#-Eigenschaften:

- *public ArrayList entities* – Liste der enthaltenen Kontextentitäten in Form von Instanzen der Klasse „EntityResultObject“.

Die Klasse „EntityResultObject“ repräsentiert eine konkrete Instanz einer Kontextentität und besitzt folgende C#-Eigenschaften:

- *string entityType* – Zuordnung der Instanz zum Entitätstyp (Kontextentität auf Ebene M1).
- *string globalID* – Eindeutige ID zur Instanz der Kontextentität.
- *ArrayList attributes* – Liste mit zugehörigen Kontextattributen (Klasse „AttributeResultObject“).

Die Klasse „AttributeResultObject“ repräsentiert eine konkrete Instanz eines Kontextattributs und besitzt folgende C#-Eigenschaften:

- *string attributeName* – Name des Kontextattributs.
- *object attributeValue* – Im Kontextattribut enthaltene Kontextinformation.
- *string implementation* – Name des Datentyps, durch den das Kontextattribut repräsentiert wird.
- *bool isRange* – Besitzt das Kontextattribut die „isRange“-Eigenschaft?

Die Klasse „SpaceQueryObject“ repräsentiert den Suchausdruck für den Kontextraum. Über den folgenden Konstruktor kann ein neuer Suchausdruck definiert werden:

public SpaceQueryObject(string spaceName);

Der Konstruktor besitzt folgenden Parameter:

- *string spaceName:* Der Name des Kontextraums auf Ebene M1. Dieser muss angegeben werden.

Zusätzliche Suchausdrücke können über folgende Methode einer Suchinstanz hinzugefügt werden:

public void AddQueryCondition(IQueryConstructObject queryCondition);

Diese besitzt folgenden Eingabeparameter:

- *IQueryConstructObject queryCondition* – Instanz eines Suchausdrucks. Jeder solche Suchausdruck definiert eine Suchbedingung auf einem Kontextattribut der Kontextentität. Das Interface „IQueryConstructObject“ kann selbst nicht instantiiert werden. Die folgenden Klassen implementieren das Interface und können verwendet werden: „QueryEntityObject“ und „QueryContainerObject“.

Alle über diese Methode hinzugefügten Suchausdrücke werden über einen AND-Operator miteinander verknüpft.

Die Klasse „QueryEntityObject“ ermöglicht die weitere Einschränkung des Kontextraums auf konkrete Kontextentitäten, die darin enthalten sein müssen. Über folgenden Konstruktor kann eine neue Instanz erzeugt werden:

public QueryEntityObject(string entityType, string entityGlobalID) Der Konstruktor besitzt folgende Parameter:

- *string entityType* – Typ der Kontextentität.

- *string entityGlobalID* – Globale ID der konkreten Kontextentität.

Die Klasse “QueryContainerObject” ermöglicht die boolsche Verknüpfung von einzelnen Suchausdrücken. Über den Konstruktor kann der boolsche Operator zur Instanz festgelegt werden:

public QueryContainerObject(int operation);

Der Konstruktor besitzt folgenden Parameter:

- *int operation* – Verweis auf den boolschen Operator.

Als Verweis zum boolschen Operator sind in der Klasse folgende Konstanten, die im Konstruktor verwendet werden können:

- *public const int OPERATOR_AND = 0;*

- *public const int OPERATOR_OR = 1;*

Ein Suchausdruck kann der Instanz mit folgender Methode hinzugefügt werden:

public void AddQueryCondition(IQueryConstructObject queryCondition);

Diese Methode besitzt den folgenden Eingabeparameter:

- *IQueryConstructObject queryCondition* – Suchausdruck, der dem „QueryContainerObject“ hinzugefügt wird. Dieses kann eine Instanz eines „QueryEntityObject“ oder eine weitere Instanz eines „QueryContainerObject“ sein.

Alle mit der Methode dem Containerobjekt zugefügten Suchausdrücke werden mit dem definierten boolschen Operator miteinander verknüpft.

Nachfolgend wird ein Beispiel für die Definition einer Suche nach einer Kontextentität in einem Kontextraum gegeben.

```
spaceQueryObject = new SpaceQueryObject("Einbruch");

QueryContainerObject containerObject = new QueryContainerOb-
ject(QueryContainerObject.OPERATOR_OR);

spaceQueryObject.AddQueryCondition(containerObject);

QueryEntityObject entityObject1 = new QueryEntityObject("Raum", "Raum:1");

containerObject.AddQueryCondition(entityObject1);

QueryEntityObject entityObject2 = new QueryEntityObject("Raum", "Raum:2");

containerObject.AddQueryCondition(entityObject2);
```

In diesem Beispiel wird nach einem Kontextraum „Einbruch“ gesucht, in denen die konkreten Kontextentitäten mit der ID „Raum:1“ oder „Raum:2“ vom Typ „Raum“ enthalten sind. Das Ergebnis der Suche kann wie folgt ausgelesen werden:

```
ResultContainer resultContainer = contextServer-
Instance.FindEntitiesInContextSpace("security", spaceQueryObject);

resultTextbox.AppendText("Number results:" + resultContainer.results.Count +
cr.ToString());
foreach (SpaceResultObject resultObject in resultContainer.results.Values)
        {
            foreach (EntityResultObject entityResultObject in
resultObject.entities)
            {
                resultTextbox.AppendText("Entity:" + enti-
tyResultObject.entityGlobalID + " - " + entityResultObject.entityType + " - " +
entityResultObject.entityID + cr.ToString());
            }
            resultTextbox.AppendText(cr.ToString());
```

}

A.1.3.4 Benachrichtigung bei Änderung einer konkreten Kontextentität

Die asynchronen Methoden sind in ihrer Funktionalität ähnlich mit den synchronen Methoden. Es wird hierbei jedoch nicht sofort ein Ergebnis geliefert. Es wird stattdessen ein Ereignis gemeldet, wenn ein Suchkriterium erfüllt ist. Die asynchrone Benachrichtigung erfolgt mittels einer „MessageQueue“.

Das Bestellen einer solchen Benachrichtigung erfolgt durch den Aufruf der folgenden Methode:

public Subscription ApplicationSubscribeEntityEvent(string subscriberName, string applicationName, EntityQueryObject queryObject);

Die Methode besitzt folgende Eingabeparameter:

- *string subscriberName* – Der Name des aufrufenden AAL-Dienstes. Dieser muss eindeutig sein und dient der Zuordnung der Anmeldung zum bestellenden Dienst.
- *string applicationName* – Der Name des Dienstetyps zum Kontextmodell.
- *EntityQueryObject queryObject* – Ein Objekt, das den Suchausdruck für die Kontextentität enthält. Die Beschreibung der Klasse ist in Kap. A.1.3.1 enthalten.

Als Ergebnis des Methodenaufrufs wird ein Objekt der Klasse „Subscription“ zurückgegeben. Dieser enthält die notwendigen Informationen zur Bestellung der Benachrichtigung. Diese Klasse besitzt folgende C#-Eigenschaft:

- *string messageQueueName* – Der Name der „MessageQueue“, über welche eine Benachrichtigung erfolgt.
- *string subscriptionID* – Die ID der Bestellung.

Nachfolgend wird ein Beispiel für den Aufruf der Methode gegeben. Diese meldet ein Event, sobald die Person mit der ID ‚Person:1’ eines der folgenden Räume betritt: ‚Schlafzimmer’, ‚Esszimmer’ oder ‚Toilette’.

EntityQueryObject queryObject = new EntityQueryObject("Person");
queryObject.globalID = "Person:1";

QueryAttributeObject attributeObject = new QueryAttributeObject("istInRaum",
QueryAttributeObject.FUNCTION_EQUAL, "Schlafzimmer");

queryObject.AddQueryCondition(attributeObject);

QueryAttributeObject attributeObject1 = new QueryAttribute-Object(QueryAttributeObject.OPERATOR_OR, "istInRaum", QueryAttribute-Object.FUNCTION_EQUAL, "Garten");

queryObject.AddQueryCondition(attributeObject1);

QueryAttributeObject attributeObject2 = new QueryAttribute-Object(QueryAttributeObject.OPERATOR_OR, "istInRaum", QueryAttribute-Object.FUNCTION_EQUAL, "Esszimmer");

queryObject.AddQueryCondition(attributeObject2);

QueryAttributeObject attributeObject3 = new QueryAttribute-Object(QueryAttributeObject.OPERATOR_OR, "istInRaum", QueryAttribute-Object.FUNCTION_EQUAL, "Toilette");

queryObject.AddQueryCondition(attributeObject3);

subscription = contextServerInstance.SubscribeEntityEvent("testapp", "securi-ty", queryObject);

Die Bestellung kann mit folgendem Methodenaufruf wieder beendet werden:

public bool ApplicationUnsubscribeEntityEvent(Subscription subscription);

Die Methode besitzt folgende Eingabeparameter:

- *Subscription subscription* – Die Beschreibung der zur beendenden Bestellung.

Mit der Benachrichtigung über die „MessageQueue“ wird dem AAL-Dienst eine Instanz der Klasse „EventObject“ übermittelt. Diese besitzt die folgenden C#-Eigenschaften:

- *string subscriptionType* – Typ der Bestellung: „ENTITY“.
- *string subscriptionID* – ID der zugehörigen Bestellung.
- *int eventType* – Typ des gemeldeten Events.

Folgende Eventtypen sind definiert:

- *public const int NO_EVENT = 0;*
- *public const int REACHED_EVENT = 1;*
- *public const int LEAVED_EVENT = 2;*

A.1.3.5 Benachrichtigung bei Änderung einer konkreten Kontextrelation

Die asynchronen Methoden sind in ihrer Funktionalität ähnlich mit den synchronen Methoden. Es wird hierbei jedoch nicht sofort ein Ergebnis geliefert. Es wird stattdessen ein Ereignis gemeldet, wenn ein Suchkriterium erfüllt ist. Die asynchrone Benachrichtigung erfolgt mittels einer „MessageQueue“.

Das Bestellen einer solchen Benachrichtigung erfolgt durch den Aufruf der folgenden Methode:

public Subscription ApplicationSubscribeRelationEvent(string subscriberName, string applicationName, RelationQueryObject queryObject);

Die Methode besitzt folgende Eingabeparameter:

- *string subscriberName* – Der Name des aufrufenden AAL-Dienstes. Dieser muss eindeutig sein und dient der Zuordnung der Anmeldung zum bestellenden Dienst.
- *string applicationName* – Der Name des Dienstetyps zum Kontextmodell.
- *RelationQueryObject queryObject* – Ein Objekt, das den Suchausdruck für die Kontextrelation enthält. Die Beschreibung der Klasse ist in Kap. A.1.3.2 enthalten.

Als Ergebnis des Methodenaufrufs wird ein Objekt der Klasse „Subscription“ zurückgegeben. Dieser enthält die notwendigen Informationen zur Bestellung der Benachrichtigung. Diese Klasse besitzt folgende C#-Eigenschaft:

- *string messageQueueName* – Der Name der „MessageQueue“, über welche eine Benachrichtigung erfolgt.
- *string subscriptionID* – Die ID der Bestellung.

Nachfolgend wird ein Beispiel für den Aufruf der Methode gegeben. Diese meldet ein Ereignis sobald eine Person mit der ID ‚Person:1160’ sich in einem Raum befindet.

RelationQueryObject queryObject = new RelationQueryOb-
ject("Bewohner_BefindetSichIn_Raum");

queryObject.fromGlobalID = "Person:1160";

subscription = contextServerInstance.SubscribeRelationEvent("testapp",
"economy", queryObject);

Die Bestellung kann mit folgendem Methodenaufruf wieder beendet werden:

public bool ApplicationUnsubscribeRelationEvent(Subscription subscription);

Die Methode besitzt folgende Eingabeparameter:

- *Subscription subscription* – Die Beschreibung der zur beendenden Bestellung.

Mit der Benachrichtigung über die „MessageQueue“ wird dem AAL-Dienst eine Instanz der Klasse „EventObject“ übermittelt. Diese besitzt die folgenden C#-Eigenschaften:

- *string subscriptionType* – Typ der Bestellung: „RELATION“.
- *string subscriptionID* – ID der zugehörigen Bestellung.
- *int eventType* – Typ des gemeldeten Events.

Folgende Eventtypen sind definiert:

- *public const int NO_EVENT = 0;*
- *public const int REACHED_EVENT = 1;*
- *public const int LEAVED_EVENT = 2;*

A.1.3.6 Benachrichtigung bei Änderung eines Kontextraums

Die asynchronen Methoden sind in ihrer Funktionalität ähnlich mit den synchronen Methoden. Es wird hierbei jedoch nicht sofort ein Ergebnis geliefert. Es wird stattdessen ein Ereignis gemeldet, wenn ein Suchkriterium erfüllt ist. Die asynchrone Benachrichtigung erfolgt mittels einer „MessageQueue“.

Das Bestellen einer solchen Benachrichtigung erfolgt durch den Aufruf der folgenden Methode:

public Subscription ApplicationSubscribeContextSpaceEvent(string subscriberName, string applicationName, SpaceQueryObject queryObject);

Die Methode besitzt folgende Eingabeparameter:

- *string subscriberName* – Der Name des aufrufenden AAL-Dienstes. Dieser muss eindeutig sein und dient der Zuordnung der Anmeldung zum bestellenden Dienst.

- *string applicationName* – Der Name des Dienstetyps zum Kontextmodell.

- *SpaceQueryObject queryObject* – Ein Objekt, das den Suchausdruck für den Kontextraum enthält. Die Beschreibung der Klasse ist in Kap. A.1.3.3 enthalten.

Als Ergebnis des Methodenaufrufs wird ein Objekt der Klasse „Subscription“ zurückgegeben. Dieser enthält die notwendigen Informationen zur Bestellung der Benachrichtigung. Diese Klasse besitzt folgende C#-Eigenschaft:

- *string messageQueueName* – Der Name der „MessageQueue“, über welche eine Benachrichtigung erfolgt.

- *string subscriptionID* – Die ID der Bestellung.

Nachfolgend wird ein Beispiel für den Aufruf der Methode gegeben. Diese meldet sobald ein Einbruch in Raum 1, Raum 2 oder Raum 3 stattgefunden hat.

```
SpaceQueryObject spaceQueryObject = new SpaceQueryObject("Einbruch");

QueryEntityObject entityObject = new QueryEntityObject("Raum", "Raum:3");

spaceQueryObject.AddQueryCondition(entityObject);

QueryEntityObject entityObject1 = new QueryEntityObject(QueryEntityObject.OPERATOR_OR, "Raum", "Raum:2");

spaceQueryObject.AddQueryCondition(entityObject1);

QueryEntityObject entityObject2 = new QueryEntityObject(QueryEntityObject.OPERATOR_OR, "Raum", "Raum:1");

spaceQueryObject.AddQueryCondition(entityObject2);
```

subscription = contextServerInstance.SubscribeContextSpaceEvent("testapp", "security", spaceQueryObject);

Die Bestellung kann mit folgendem Methodenaufruf wieder beendet werden:

public bool ApplicationUnsubscribeSpaceEvent(Subscription subscription);

Die Methode besitzt folgende Eingabeparameter:

- *Subscription subscription* – Die Beschreibung der zur beendenden Bestellung.

Mit der Benachrichtigung über die „MessageQueue“ wird dem AAL-Dienst eine Instanz der Klasse „EventObject“ übermittelt. Diese besitzt die folgenden C#-Eigenschaften:

- *string subscriptionType* – Typ der Bestellung: „SPACE“.
- *string subscriptionID* – ID der zugehörigen Bestellung.
- *int eventType* – Typ des gemeldeten Events.

Folgende Eventtypen sind definiert:

- *public const int NO_EVENT = 0;*
- *public const int REACHED_EVENT = 1;*
- *public const int LEAVED_EVENT = 2;*
- *public const int CHANGED_EVENT = 3;*

A.2 Kontextmodelle

Aufbauend auf den Anforderungen an das konkrete Kontextmodell aus Kap. 3 werden nachfolgend die daraus resultierenden Kontextmodelle auf Ebene M1 der Metadaten-Architektur konzeptuell beschrieben. Diese können in den weiteren Arbeiten hin zu Referenz-Kontextmodellen ergänzt werden. Die nachfolgend beschriebenen Kontextmodelle sind in der Implementierung des Kontextservers verfügbar.

A.2.1 Kontextmodell auf Ebene der Infrastruktur

Nachfolgend wird ein Überblick über das Kontextmodell auf Ebene der Infrastruktur mit Hilfe der konzeptuellen Modellierung gegeben. Zur Strukturierung der im Kontextmodell zu instanziierenden Modellelemente wird auf dem Ansatz von SOCAM [184] aufgesetzt. Dieses definiert ein Kontextmodell mittels Ontologien auf zwei Ebenen. Auf einer oberen Ebene (Upper Layer Ontology) werden die generellen Konzepte definiert. Auf dieser Basis werden für einzelne Domänen spezifische Ontologien definiert, welche die generellen Konzepte spezialisieren. Dieser Ansatz soll nachfolgend übernommen werden. Auf der oberen Ebene definiert der Ansatz von SOCAM die grundlegenden Kontextentitäten „CompEntity“, „Location“, „Person“ und „Activity“. Diese sind von der generellsten Kontextentität „ContextEntity“ abgeleitet. Die Kontextentität „CompEntity“ definiert die Entitäten, die Bestandteil der intelligenten Umgebung des Benutzers sind. Weitere Spezialisierungen dieser Entität sind beispielsweise „Service“, „Application“, „Device“, „Network“ und „Agent“. Grundlegend ist diese Ontologie daher für die Kontextmodellierung im AAL geeignet. Es fehlt hier jedoch eine generelle Beschreibung von Gegenständen, die Bestandteil der Umgebung des Bewohners sind, die jedoch nicht der eigentlichen intelligenten Infrastruktur zuzuordnen sind. Dieses sind beispielsweise Einrichtungsgegenstände oder Haushaltsgeräte. Diese sollen durch ein eigenes Konzept „object“ als Spezialisierung der Entität „ContextEntity“ modelliert werden. Das Konzept der „Activity“ in SOCAM lässt sich nicht in das AAL übertragen. In SOCAM werden darunter verschiedene Aktivitäten beschrieben, die aus bestehenden Kontextinformationen abgeleitet oder die geplant sein können. Entsprechende Spezialisierungen dieser Entität sind „DeducedActivity“ und „ScheduledActivity“. In unserem Ansatz sind solche Aktivitäten entweder als Attribut zu der entsprechenden Person oder als Situation mit Hilfe eines Kontextraums zu modellieren. Zusätzlich wird die Zeit als eine technische Kontextentität modelliert, um hierüber eine Zuordnung eines Zeitsensors in das Modell zu ermöglichen. Entsprechend besteht die obere Ebene des Infrastruktur-Kontextmodells für das AAL in der nachfolgend verwendeten deutschen Bezeichnung aus folgenden Kontextentitäten: „Zeit“, „Kontextentität“, „Infrastruktur“, „Ort“, „Person“ und „Gegenstand“.

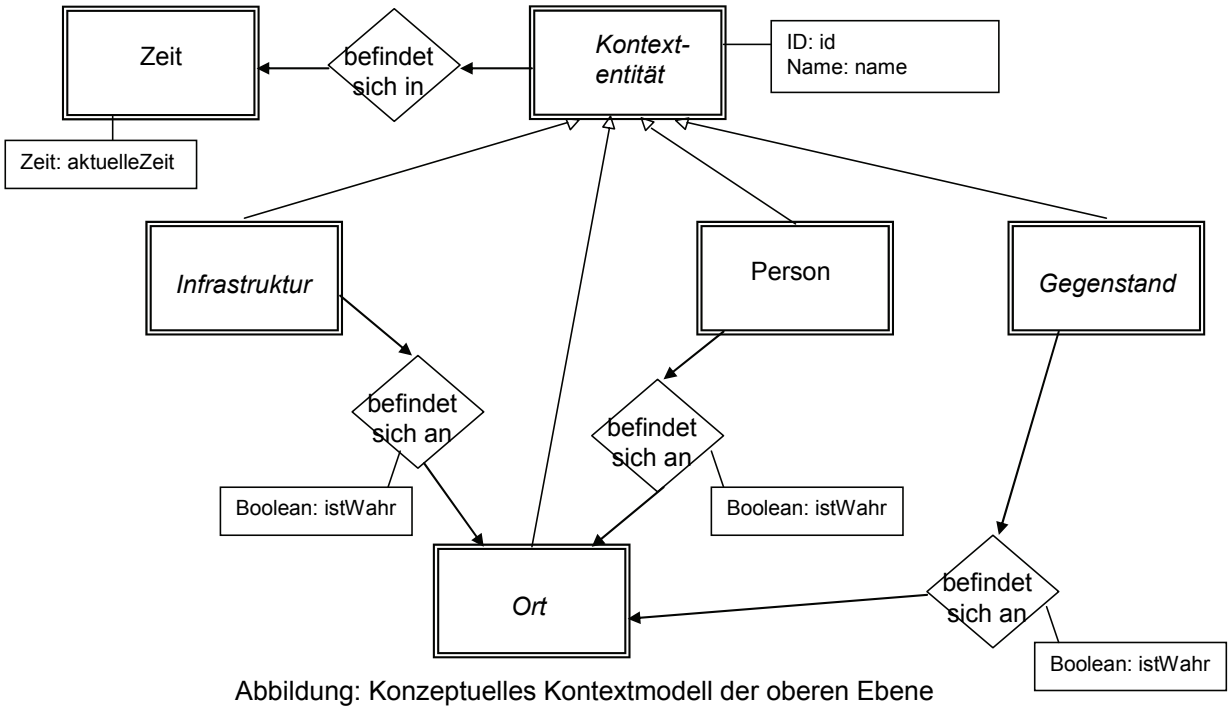


Abbildung: Konzeptuelles Kontextmodell der oberen Ebene

Unterhalb dieser oberen Ebene definiert die Ontologie von SOCAM mehrere domänenspezifische Ontologien. Diesen Ansatz übernehmen wir ebenfalls und definieren die für das AAL relevanten Entitäten der häuslichen Umgebung in einer solchen Spezialisierung. Dieses hat den Vorteil, dass bei einer Ausweitung der intelligenten Umgebung außerhalb des häuslichen Bereichs die darin relevanten Entitäten in einer entsprechenden spezifischen Ontologie beschrieben werden können. Solche Ausweitungen werden derzeit in der Forschung diskutiert und sind teilweise unter dem Begriff „Smart Space“ bekannt.

Nachfolgend wird nun die Entität „Person“ im Detail beschrieben. Auf Ebene der Infrastruktur definiert das Kontextmodell keine weitere Spezialisierung dieser Entität. Es werden die generellen Kontextattribute der Entität definiert, sowie die möglichen Beziehungen zu anderen Entitäten.

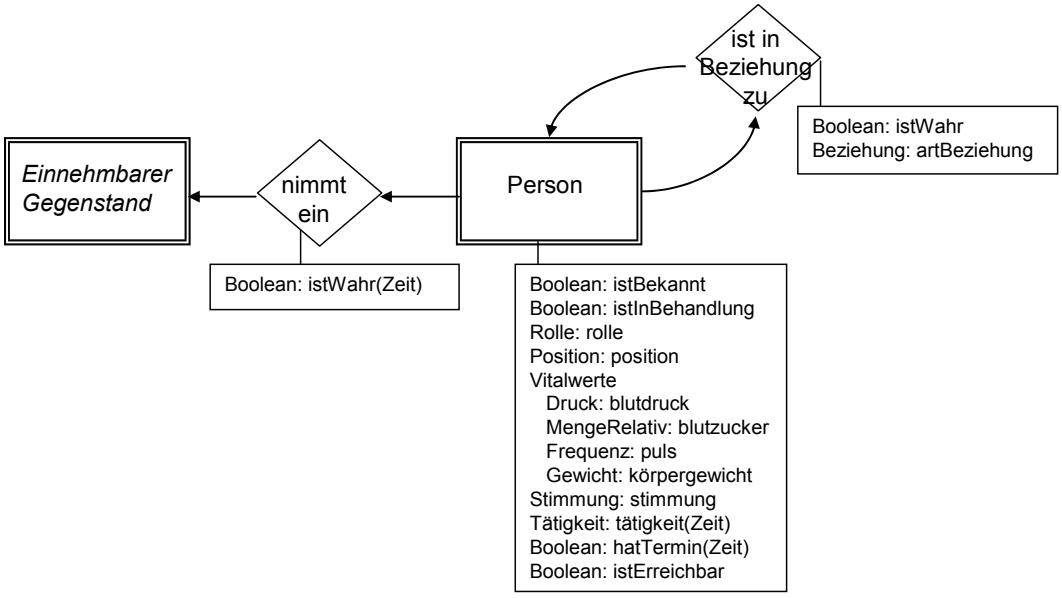


Abbildung: Spezialisierung Entität „Person“

Nachfolgend wird nun die Entität „Ort“ im Detail beschrieben. Es wird nachfolgend zwischen Orten unterschieden, die sich innerhalb und außerhalb von Gebäuden befinden. Dieses entspricht der Unterscheidung zwischen „IndoorSpace“ und „OutdoorSpace“ aus dem SOCAM-Kontextmodell. Innerhalb eines Gebäudes wird nachfolgend zwischen der Wohnung und den darin enthaltenen Räumen unterschieden.

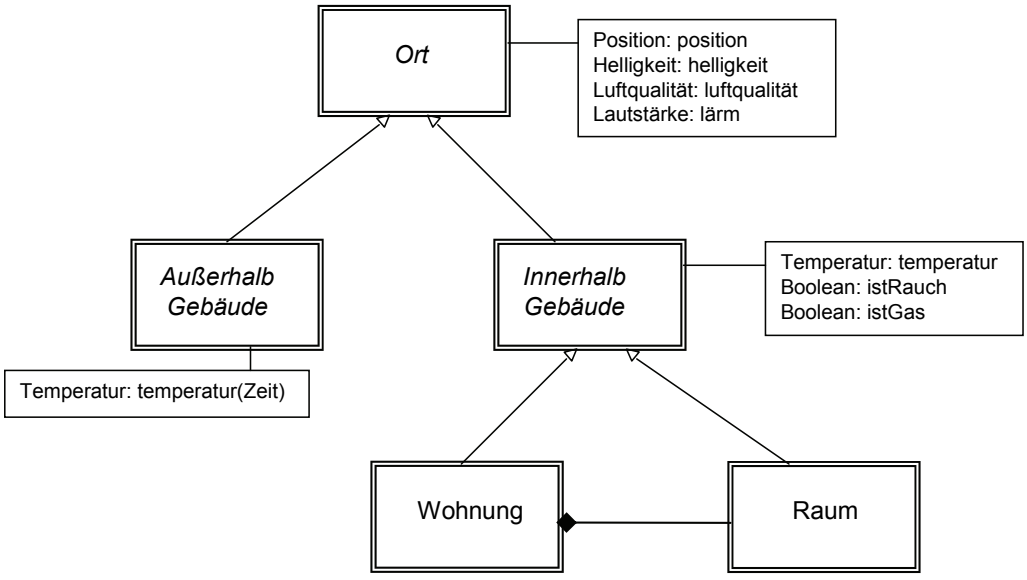


Abbildung: Spezialisierung Entität „Ort“

Nachfolgend wird nun die Entität „Infrastruktur“ im Detail beschrieben. Hier werden alle Kontextentitäten beschrieben, die Bestandteil der intelligenten Umgebung sind. Hier geht es um die Beschreibung von Netzwerk, Hardware und Software. Eine Grundlage für die Definition dieser Kontextentitäten ist das CoDAMoS-Kontextmodell [133]. Eine weitere Verfeinerung dieser Konzepte ist in CoDAMoS enthalten und wird nachfolgend nicht weiter verfolgt. Unter der Infrastruktur-Software sind dort das Betriebssystem, die virtuelle Maschine oder bestehende Middleware definiert. Die Ressourcen einer Hardware lassen sich dort weiter unterscheiden zwischen der Stromversorgung, CPU, Hauptspeicher und Massenspeicher. Bei dem Interaktionsmedium wird weiter unterschieden zwischen einem Eingabe- und einem Ausgabemedium.

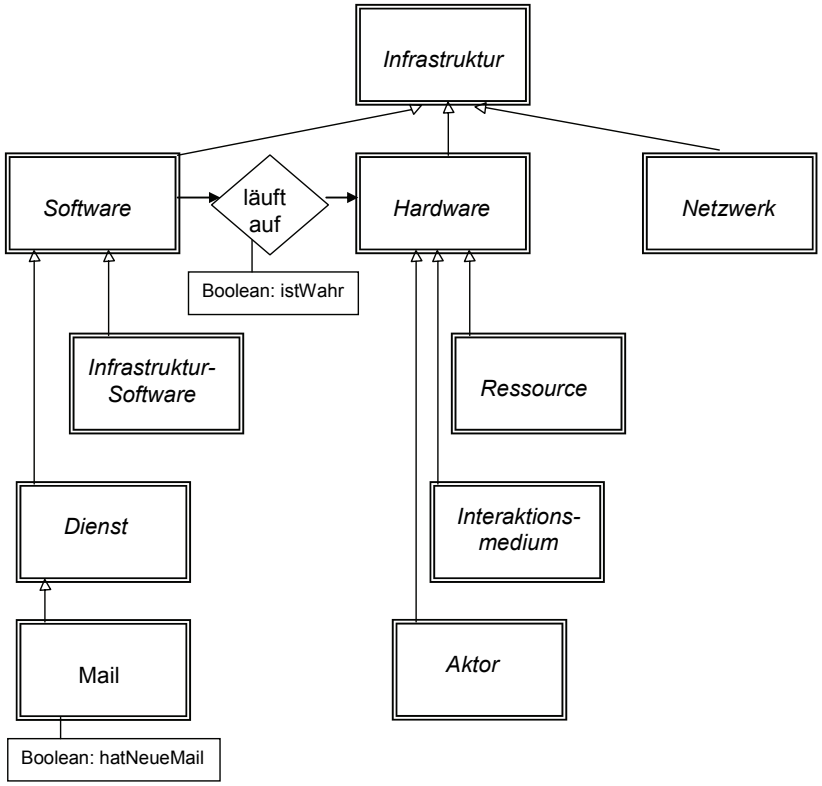


Abbildung: Spezialisierung Entität „Infrastruktur“

Nachfolgend wird nun die Entität „Gegenstand“ im Detail beschrieben. Ein Gegenstand wird nachfolgend zunächst entsprechend seinen Eigenschaften unterschieden. Mögliche Eigenschaften sind, ob ein Gegenstand vom Menschen eingenommen werden kann, ob dieses zur Verwahrung anderer Gegenstände genutzt werden kann, oder ob es sich um ein elektrisches Gerät handelt. Bei einnehmbaren Gegenständen wird unterschieden zwischen Flüssigkeit, Nahrung und Medikamenten. Ein Kühlschrank ist sowohl ein elektrisches Gerät als auch ein Gegenstand zur Verwahrung anderer Gegenstände.

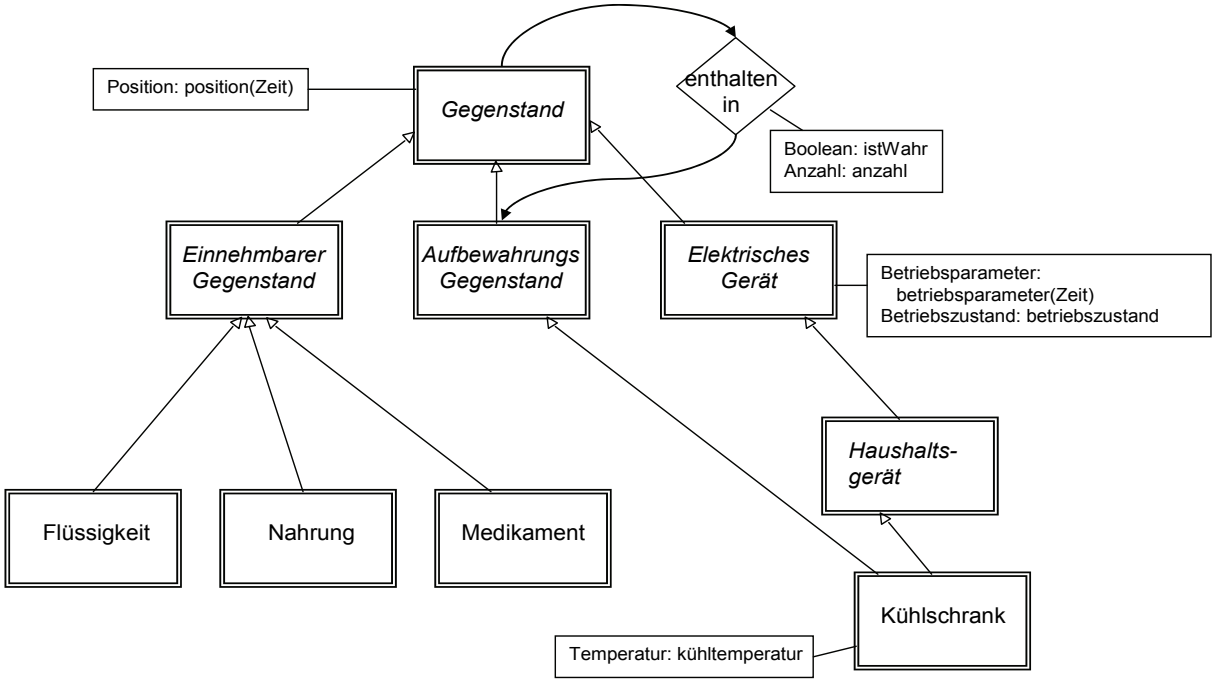


Abbildung: Spezialisierung Entität „Gegenstand“

A.2.2 Kontextmodell auf Ebene der Dienste

Nachfolgend werden die Kontextmodelle auf Ebene der Dienste beschrieben. Auf dieser Ebene ist nicht nur ein Kontextmodell, sondern für jeden Dienstetyp im AAL eines festzulegen. Diese bestimmen die für die jeweiligen Dienstetypen benötigten Modellinstanzen. Hierbei kann es Überschneidungen zwischen den einzelnen dienstetyp-spezifischen Kontextmodellen geben. Die Festlegung der jeweiligen Kontextmodelle basiert auf folgenden Grundlagen:

- Instanziierung der Modellelemente, die im Metamodell auf dieser Ebene vorgegeben sind.
- Anforderungen an ein Kontextmodell auf Ebene der Dienste, die in Kapitel 3.3.2 identifiziert wurden.

In der Literatur sind bereits Ansätze zur Kategorisierung von AAL-Diensten beschrieben. Jedoch gibt es noch keine Publikationen von jeweils zugehörigen Kontextmodellen. Daher kann aus der Literatur kein Hinweis für eine mögliche Gestaltung des Kontextmodells entnommen werden.

Die nachfolgend definierten Kontextmodelle erheben keinen Anspruch auf Vollständigkeit.

A.2.2.1 Kontextmodell „health”

Nachfolgend wird ein Überblick über das Kontextmodell für den Dienstetyp „health“ gegeben. Dieses Kontextmodell ist primär auf den Bewohner und seinen Vitalwerten fokussiert. Daneben werden aber auch Kontextinformationen zur Einnahme von Lebensmitteln, Flüssigkeiten und Medikamente benötigt. Der Zeitpunkt der Einnahme wird für die Überwachung benötigt. Weiterhin muss die Anzahl der Medikamente überwacht in das Modell aufgenommen werden. Weiterhin wird neben dem Bewohner auch der ihm zugehörige ärztliche Betreuer modelliert mitsamt der durchgeführten Tätigkeit.

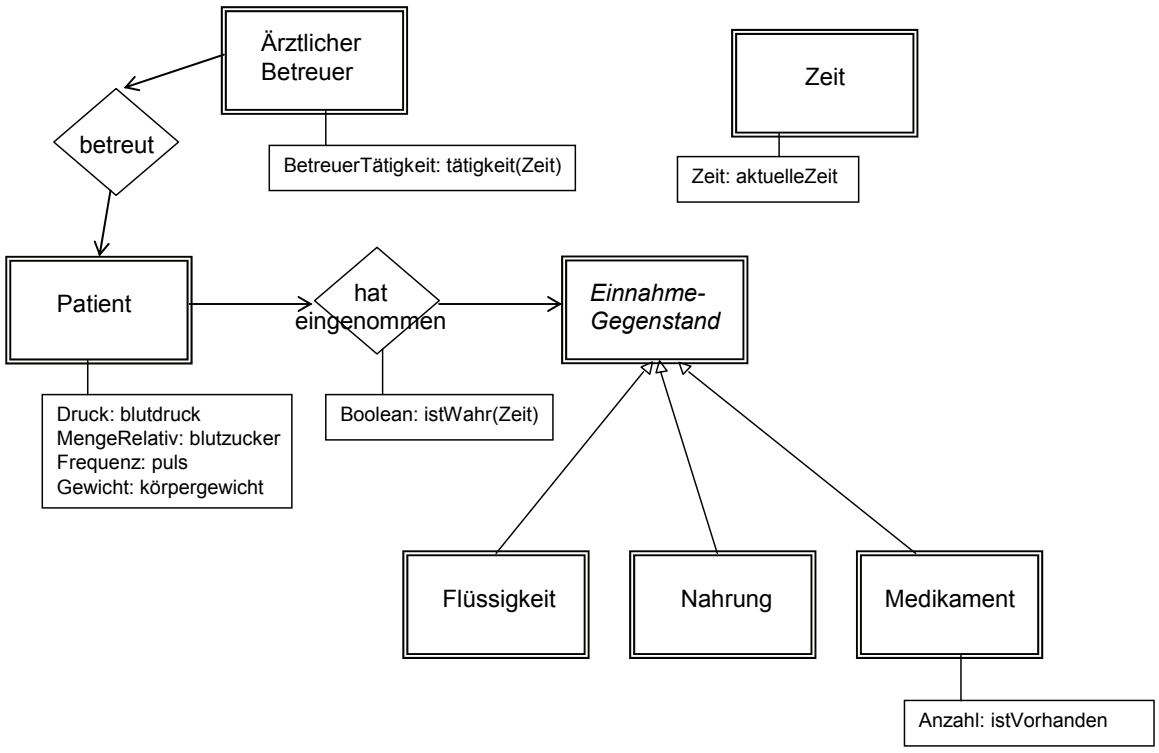


Abbildung: Konzeptuelles Kontextmodell „health“

A.2.2.2 Kontextmodell „security”

Nachfolgend wird ein Überblick über das Kontextmodell für den Dienstetyp „security“ gegeben. Das Kontextmodell ist fokussiert auf die Wohnung und die

darin enthaltenen Gegenstände und Personen. Die Gegenstände und die Personen müssen innerhalb der Wohnung und den darin vorhandenen Räumen verortet werden können. Zudem muss bei Personen zwischen dem Bewohner und sonstigen Personen unterschieden werden können.

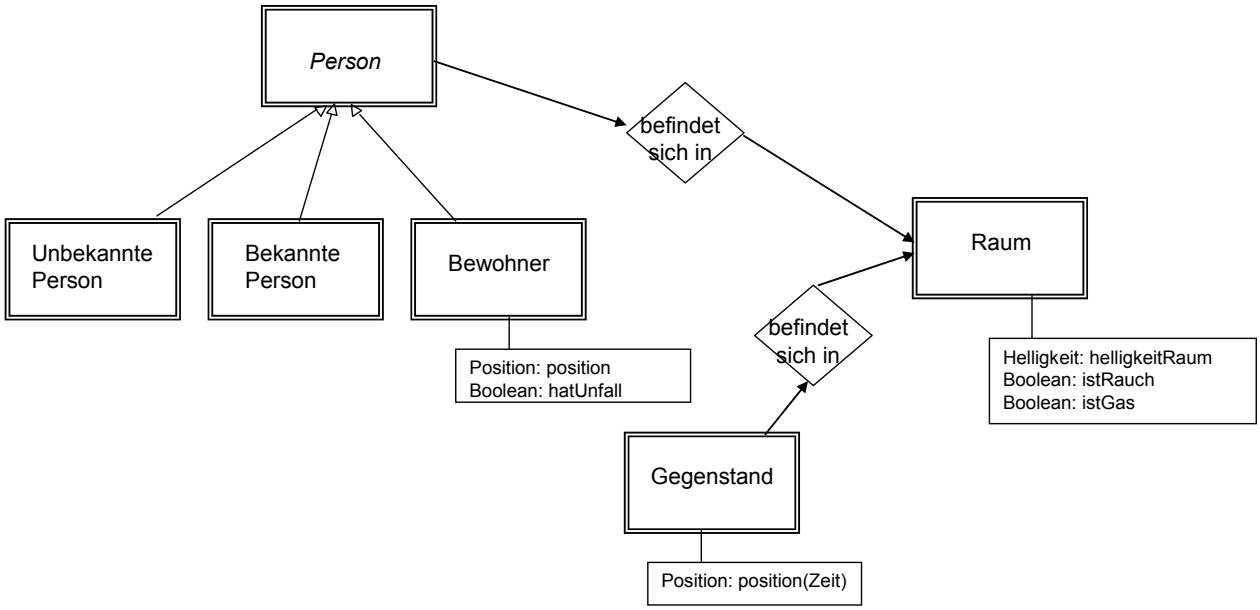


Abbildung: Konzeptuelles Kontextmodell „security“

A.2.2.3 Kontextmodell „comfort”

Nachfolgend wird ein Überblick über das Kontextmodell für den Dienstetyp „health“ gegeben. Dieses Kontextmodell bezieht sich auf den Bewohner, die häusliche Umgebung und die externe Umgebung. Die häusliche Umgebung muss inklusive der vorhandenen Infrastruktur beschrieben werden. Dazu gehören die möglichen Interaktionsmedien, Geräte und Aktoren. Auch können Gegenstände von Interesse sein, die mit dem Bewohner oder den Gegenständen interagieren. Auch der Zustand von Diensten als Teil der Infrastruktur ist von Interesse.

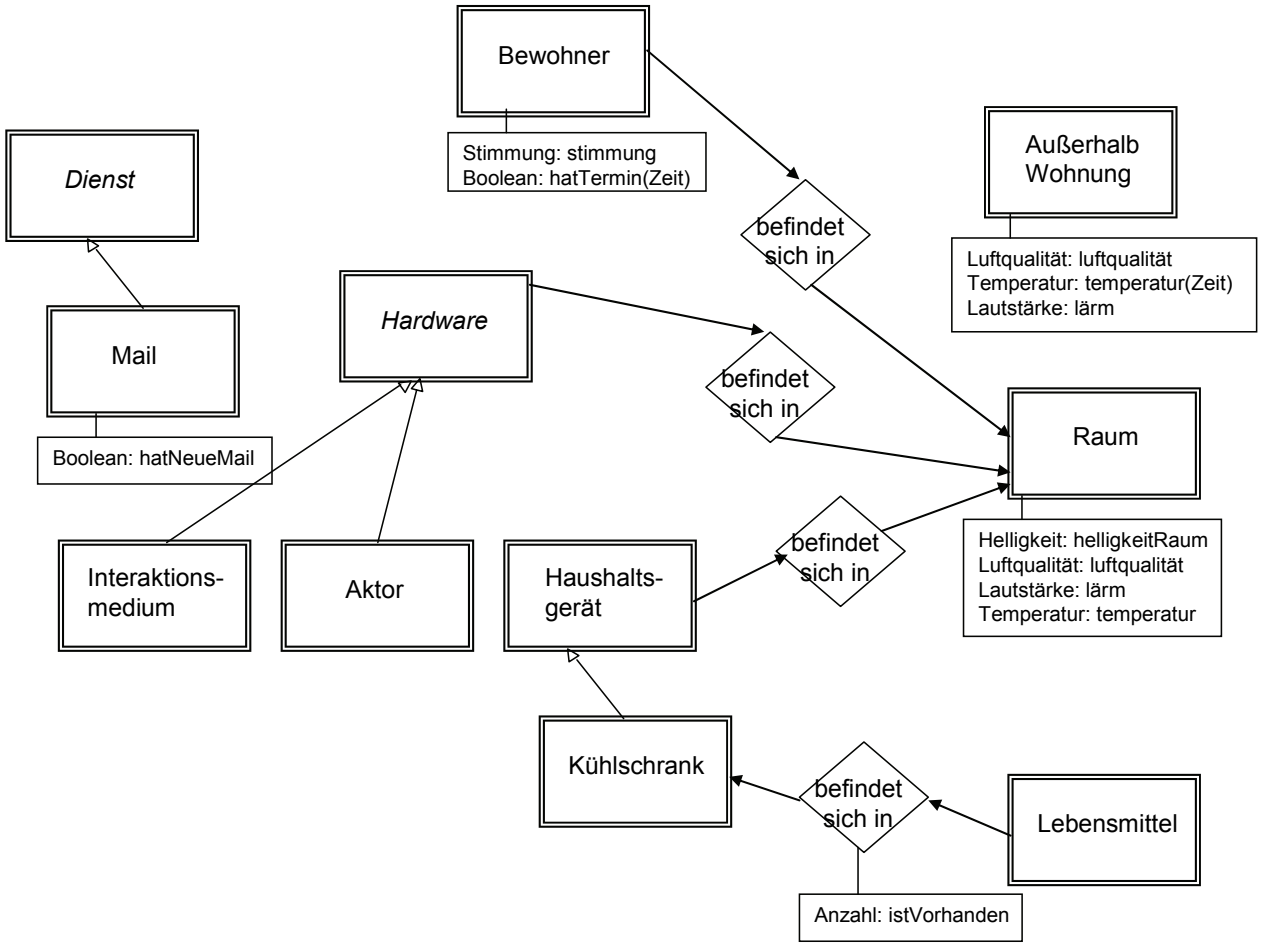


Abbildung: Konzeptuelles Kontextmodell „comfort“

A.2.2.4 Kontextmodell „social”

Nachfolgend wird ein Überblick über das Kontextmodell für den Dienstetyp „social“ gegeben. Dieses Kontextmodell bezieht sich auf den Bewohner, seine Erreichbarkeit und Aktivität, die Interaktionsmedien in seiner Umgebung, sowie die Personen, zu denen dieser in Beziehung steht.

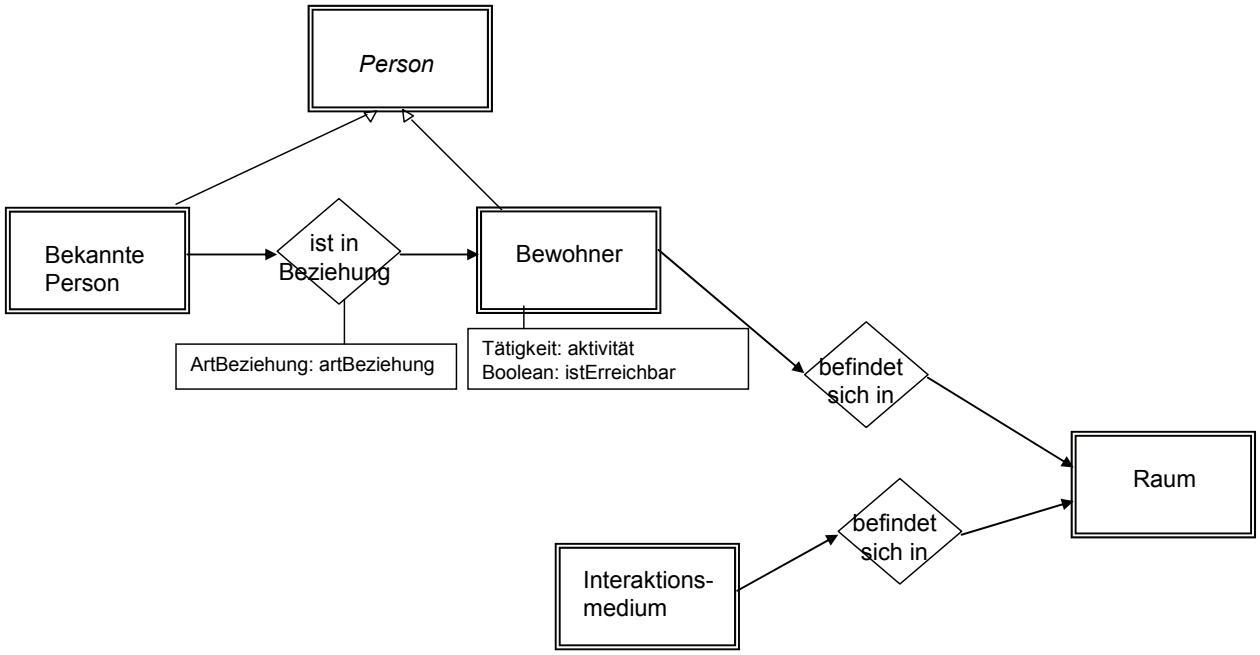


Abbildung: Konzeptuelles Kontextmodell „social“

A.2.2.5 Kontextmodell „economy”

Nachfolgend wird ein Überblick über das Kontextmodell für den Dienstetyp „economy“ gegeben. Dieses Kontextmodell beschreibt den Bewohner, die häusliche Umgebung, sowie die Umgebung außerhalb der Wohnung. In Bezug auf den Bewohner werden die aktuelle und geplante Anwesenheit als Information benötigt. Für die häusliche Umgebung werden das Raumklima, sowie Informationen über die Verbraucher benötigt. Für die externe Umgebung werden die Temperatur und die Helligkeit benötigt.

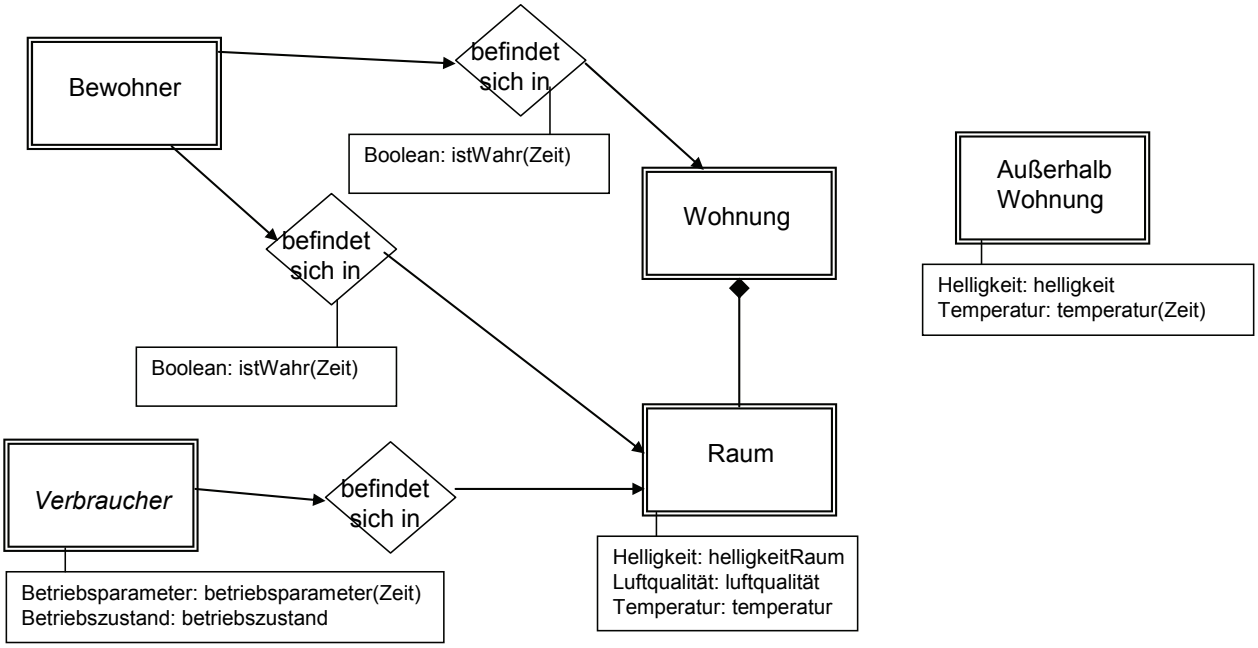


Abbildung: Konzeptuelles Kontextmodell „economy“

Literatur

[1] Statistisches Bundesamt Deutschland, Bevölkerung, [Online]: http://www.destatis.de/ jetspeed/portal/cms/Sites/destatis/Internet/EN/Navigation/Statistics/Bevoelkerung/Bevoelkerung.psml, Aug. 2007

[2] Ambient Assisted Living Joint Programme, [Online]: http://www.aal-europe.eu/, Aug. 2007

[3] Bundesministerium für Bildung und Forschung, AAL, Marktpotentiale, [Online]: http://www.ambientassistedliving.de/marktpotenziale, Aug. 2007

[4] T. Strang, C. Linnhoff-Popien: “A Context Modeling Survey”, First International Workshop on Advanced Context Modelling, Reasoning And Management at UbiComp 2004, Nottingham, England, Sept. 2004

[5] M. Baldauf, S. Dustdar, F. Rosenberg: “A survey on context-aware systems”, International Journal on Ad Hoc and Ubiquitous Computing, Band 2, Nr. 4, S. 263-277, 2007

[6] M. Wojciechowski, J. Xiong: „Towards an Open Context Infrastructure”, Second Workshop on Context Awareness for Proactive Systems (CAPS 2006), Kassel, Deutschland, 2006

[7] M. Wojciechowski, J. Xiong: “A User Interface Context Model for Ambient Assisted Living”, 6th International Conference on Smart Homes and Health Telematics ICOST 2008, Ames, IA, USA, Lecture Notes in Computer Science 5120, S. 105-112, 2008

[8] M. Wojciechowski, J. Xiong: “On Context Modeling in Ambient Assisted Living”, Fifth International Workshop Modeling and Reasoning in Context (MRC 08), Delft, Niederlande, Juni 2008

[9] M. Wojciechowski: „End User Context Modeling in Ambient Assisted Living“, International Journal of Advanced Pervasive and Ubiquitous Computing“ IJAPUC, Band 1, Nr. 3, S. 61-80, 2009

[10] Working group ‘Smart Homes and Ambient Assisted Living’, [Online]: http://www.health-smarthomes.org/index.html, Sept. 2007

[11] T. Kleinberger, M. Becker, E. Ras, A. Holzinger, P. Müller: „Ambient Intelligence in Assisted Living: Enable Elderly People to Handle Future Interfaces”, C. Stephanidis (Ed.): Universal Access in HCI, Part II, HCII 2007, LNCS 4555, S. 103–112, Springer Verlag, 2007

[12] AmI-07: European Conference on Ambient Intelligence, [Online]: http://www.ami-07.org/workshops.html, Okt. 2007

[13] M. Weiser, R. Gold, J.S. Brown: “The origins of ubiquitous computing research at PARC in the late 1980s”, IBM Systems Journal 38(4), S. 693-696, 1999

[14] M. H. Coen: “Design Principles for Intelligent Environments”, Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI’98, Madison, WI, USA), S. 547-554, American Association for Artificial Intelligence, 1998

[15] R. Oppermann, R. Rashev, Kinshuk: “Adaptability and Adaptivity in Learning Systems”, Knowledge Transfer (Band 2), Ed.: A. Behrooz, pAce, London, S. 173-179, 1997

[16] Phillips, HomeLab, [Online]: http://www.research.philips.com/technologies/misc/homelab, Sept. 2007

[17] Carnegie Mellon University, CyLab – Ambient Intelligence Lab, [Online] : http://www.cmu.edu/vis/, Sept. 2007

[18] Kingston University London, Ambient Intelligence Research Group, [Online]: http://www.ambientintelligence.net/, Sept. 2007

[19] MIT Media Lab, Ambient Intelligence Group, [Online]: http://ambient.media.mit.edu/, Sept. 2007

[20] NTT Research. Ambient Intelligence Research Group, [Online]: http://www.brl.ntt.co.jp/cs/kikang/index.html, Sept. 2007

[21] University of Reading, Ambient & Pervasive Intelligence Research Group, [Online]: http://www.api.reading.ac.uk/, Sept. 2007

[22] M. Anastasopoulos, C. Bartelt, J. Koch, D. Niebuhr, A. Rausch: „Towards a Reference Middleware Architecture for Ambient Intelligence Systems”, Proceedings of the Workshop for Building Software for Pervasive Computing, 20th Conference on Object-Oriented Programming Systems, Languages and Applications (OOPSLA, San Diego, CA, USA), S. 145-157, Okt. 2005

[23] KNX Association, KNX Specifications, [Online]: http://www.knx.org/knx-standard/knx-specifications/, Juni 2010

[24] X10 – Wikipedia, [Online]: http://de.wikipedia.org/wiki/X10, Juni 2010

[25] LonMark International, Standards, [Online]: http://www.lonmark.org/technical_-resources/standards, Juni 2010

[26] PCS Powerline Control Systems, UPB Technology Description Version 1.4, [Online]: http://www.pulseworx.com/downloads/upb/UPBDescriptionv1.4.pdf, Juni 2010

[27] UPnP Forum, Standardized DCPs & Certification, [Online]: http://upnp.org/sdcps-and-certification/standards/international-standards/, Juni 2010

[28] ZigBee Alliance, [Online]: http://zigbee.org/, Juni 2010

[29] Z-Wave Alliance, [Online]: http://www.z-wavealliance.org/modules/AllianceStart/, Juni 2010

[30] Welcome to the OSGi Alliance, [Online]: http://www.osgi.org/, Okt. 2007

[31] Wikipedia, Home automation, [Online]: http://en.wikipedia.org/wiki/Home_automation, Sept. 2007

[32] Home Automation Guide, [Online]: http://www.diy-ha.com/, Sept. 2007

[33] Home automation newsgroup on usenet, [Online]: news://comp.home.automation/, Sept. 2007

[34] inHaus - Innovationszentrum der Fraunhofer-Gesellschaft, [Online]: http://www.inhaus-zentrum.de/site_en, Okt. 2007

[35] A. Schmidt: “Ubiquitous Computing – Computing in Context”, PhD Thesis, Lancaster University, [Online]: http://www.comp.lancs.ac.uk/~albrecht/phd/, 2002

[36] T. Rädisch: “Entkopplung von Kontextsensorik und -modell für einen informationslogistischen Kontextserver“, Diplomarbeit, Universität Dortmund, 2007

[37] Wikipedia, RFID, [Online]: http://de.wikipedia.org/wiki/Radio_Frequency_Identification, Juni 2010

[38] Wikipedia, Bewegungsmelder, [Online]: http://de.wikipedia.org/wiki/Bewegungsmelder, Juni 2010

[39] Siegenia, [Online]: http://www.siegenia-aubi.com/de/products/building_technology/radio_sensor/, 2008

[40] J. Meis, J. Draeger: „Modelling automated service orchestration for IT-based homeservices”, Proceedings of the IEEE/INFORMS International Conference on Service Operations and Logistics, and Informatics (Philadelphia, USA, Aug. 2007), IEEE, S. 155-160, 2007

[41] M. Kranz, A. Maldonado, R. Rusu, B. Hörnler, G. Rigoll, M. Beetz, A. Schmidt: “Sensing Technologies and the Player-Middleware for Context-Awareness in Kitchen Environ-

ments", 4th International Conference on Networked Sensing Systems, INSS 2007, Braunschweig, Deutschland, S. 179-180, 2007

[42] B. d. Ruyter, E. Pelgrim: "Ambient assisted-living research in carelab", interactions, Band 14, Nr. 4, ACM Press, New York, NY, S. 30-33, 2007

[43] S. Meyer, A. Rakotonirainy: "A Survey of Research on Context-Aware Homes", ACSW Frontiers '03: Proceedings of the Australasian information security workshop conference on ACSW frontiers 2003 (Darlinghurst, Australien), Australian Computer Society, S. 159-168, 2003

[44] J. Nehmer, M. Becker, A. Karshmer, R. Lamm: „Living assistance systems: an ambient intelligence approach.", Proceeding of the 28th International Conference on Software Engineering (Shanghai, China, Mai 2006), ACM Press, New York, NY, S. 43-50, 2006

[45] IST-2001-38142-CONTEXT: "CONTEXT AWARE SERVICE ENGINEERING IN SUPPORT OF FUTURE BUSINESS NETWORKS", [Online]: http://context.upc.es/Papers/ContextCOCONET.pdf, 2001

[46] G. Banavar, J. Black, R. Cáceres, M. Ebling, E. Stern, J. Kannry: „Deriving Long-Term Value from Context-Aware Computing", Information Systems Management, special issue on Ubiquitous Computing, Band 22, Nr. 4, Taylor & Francis, S. 32-42, 2005

[47] B. Truong, Y. Lee, S. Lee: "Modeling Uncertainty in Context-Aware Computing", Proceedings of the Fourth Annual ACIS International Conference on Computer and Information Science (ICIS'05, Jeju Island, Süd Korea, Juli 2005), S. 676-681, 2005

[48] K. Henricksen, J. Indulska: "Modelling and using imperfect context information", Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops (Orlando, FL, USA, März 2004), S. 33-37, 2004

[49] S. Greenberg: "Context as a Dynamic Construct", Human-Computer-Interaction, Band 16, Nr. 2, L. Erlbaum Associates, S. 257–268, 2001

[50] S. W. Loke: "Facing Uncertainty and Consequence in Context-Aware Systems: towards an Argumentation Approach", Workshop on Advanced Context Modelling, Reasoning and Management (Tokio, Japan), [Online]: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.1.6283&rep=rep1&type=pdf, 2004

[51] E. Rukzio, J. Hamard, C. Noda, A. De Luca: "Visualization of Uncertainty in Context Aware Mobile Applications", Proceedings of the 8th conference on Human.computer interaction with mobile devices and services (Helsinki, Finnland, Sept. 2006), S. 247-250, 2006

[52] A. Mitseva, M. Imine, N. R. Prasad: „Context-Aware Privacy Protection with Profile Management“, Proceedings of the 4th international workshop on Wireless mobile applications and services on WLAN hotspots (Los Angeles, CA, USA, Sept. 2006), S. 53 – 62, 2006

[53] S. Campadello, O. Coutand, C. del Rosso, S. Holtmanns, T. Kanter, C. Räck, B. Mrohs, S. Steglich: „Trust and Privacy in Context-Aware Support for Communication in Mobile Groups”, Workshop on Context Awareness for Proactive Systems CAPS 2005, Helsinki, Finland, [Online]: http://www.cs.helsinki.fi/u/ptnurmi/papers/322.pdf, 2005

[54] P. Osbakk: “A Privacy Enhancing Infrastructure for Context-Awareness”, PhD Thesis, University of Kent, Canterbury, Kent, UK, [Online]: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.66.9348&rep=rep1&type=pdf, 2007

[55] A. Schmidt, K. Van Laerhoven: „How to Build Smart Appliances?”, IEEE Personal Communications, August 2001, [Online]: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.21.3362 , 2001

[56] C. Chong, S. Kumar: “Sensor Networks: Evolution, Opportunities, and Challenges”, Proceedings of the IEEE, Vol. 91, No. 8, S. 1247-1256, [Online]: http://www-net.cs.umass.edu/cs791_sensornets/papers/chong.pdf, 2003

[57] A. Dey: “Providing Architectural Support for Building Context-Aware Applications”, PhD Thesis, Georgia Institute of Technology, [Online]: http://citeseerx.ist.psu.edu/ viewdoc/download?doi=10.1.1.71.1808&rep=rep1&type=pdf, 2000

[58] C. Becker, D. Nicklas: “Where do spatial context-models end and where do ontologies start? A proposal of a combined approach”, Proceedings of the First International Workshop on Advanced Context Modelling, Reasoning and Management in conjunction with UbiComp 2004 (Nottingham, England, Sept. 2004), S. 48-53, 2004

[59] Wikipedia, Standortbezogene Dienste, [Online]: http://de.wikipedia.org/wiki /Standortbezogene_Dienste, Juni 2010

[60] W. Schilit: “System Architecture for Context-Aware Mobile Computing”, PhD thesis, Columbia University, [Online]: http://schilit.googlepages.com/schilit-thesis.pdf, 1995

[61] M. Korkea-aho: “Context-Aware Applications Survey”, [Online]: http://users.tkk.fi/~mkorkeaa/doc/context-aware.html, 2000

[62] R. Want, A. Hopper, V. Falcao, J. Gibbons: “The Active Badge Location System”, ACM Transactions on Information Systems, Band 10, Nr. 1, ACM, S. 91-102, 1992

[63] R. Want, et al.: “An Overview of the PARCTAB Ubiquitous Computing Experiment”, IEEE Personal Communications, Band 2, Nr. 6, S. 28-43, 1995

[64] Wikipedia, PDA, [Online]: http://de.wikipedia.org/wiki/Personal_Digital_Assistant, Juni 2010

[65] D. Salber, A. Dey, G. Abowd: “The Context Toolkit: Aiding the Development of Context-Enabled Applications”, CHI '99: Proceedings of the SIGCHI conference on Human factors in computing systems (Pittsburgh, PE, USA, Mai 1999), ACM, S. 434-441, 1999

[66] A. Dey, D. Salber, G. Abowd, M. Futakawa: “The Conference Assistant: Combining context-awareness with wearable computing”, Proceedings of the 3rd IEEE International Symposium on Wearable Computers (San Francisco, CA, USA), S. 21-28, 1999

[67] S. Long, et al.: ”Rapid Prototyping of Mobile Context-aware Applications: The Cyberguide Case Study”, Proceedings of the 2nd annual international conference on Mobile computing and networking (MobiCom’96, Rye, NY, USA, Nov. 1996), S. 97-107, 1996

[68] N. Davies, K. Mitchell, K. Cheverest, G. Blair: “Developing a Context Sensitive Tourist Guide”, First Workshop on Human Computer Interaction with Mobile Devices, GIST Technical Report G98-1, [Online]: http://citeseerx.ist.psu.edu/viewdoc/download ?doi=10.1.1.48.1651&rep=rep1&type=pdf, 1998

[69] J. Yang, W. Yang, M. Denecke, A. Waibel: „Smart sight: a tourist assistant system”, Proceedings of the 3rd IEEE International Symposium on Wearable Computers (San Francisco, CA, USA), S. 73-78, 1999

[70] Wikipedia, GPS, [Online]: http://de.wikipedia.org/wiki/Global_Positioning_System, Juni 2010

[71] M. Lamming, M. Flynn: “Forget-me-not: Intimate Computing in Support of Human Memory”, Proceedings of FRIEND 21: International Symposium on Next Generation Human Interfaces (Tokio, Japan), S. 125-128, 1994

[72] B. J. Rhodes: ”The wearable remembrance agent: a system for augmented memory”, Proceedings of the 1st IEEE International Symposium on Wearable Computers (Cambridge, MA, USA, Okt. 1997), S.123-128, 1997

[73] J. Healey, R.W. Picard: “Startlecam: A Cybernetic Wearable Camera”, Proceedings of the 2nd International Symposium on Wearable Computers (Pittsburgh, PE, USA, Okt. 1998), S. 42-49, 1998

[74] A. Ranganathan, R. H. Campbell, A. Ravi, A. Mahajan: “ConChat: A Context-Aware Chat Program”, IEEE Pervasive Computing, Band 1, Nr. 3, IEEE Computer Society, S. 51-57, 2002

[75] Y. Takeuchi, M. Sugimoto: “CityVoyager: An Outdoor Recommendation System Based on User Location History”, In Proceedings of the 3rd International Conference on Ubiquitous

Intelligence and Computing (UIC 2006, Wuhan and Three Gorges, China, Sept. 2006), S. 625-636, 2006.

[76] D. Siewiorek, A. Smailagic, J. Furukawa, A. Krause, N. Moraveji, K. Reiger, J. Shaffer, F. L. Wong: “SenSay: A Context-Aware Mobile Phone”, Proceedings of the 7th IEEE international Symposium on Wearable Computers (ISWC’03, White Plains, NY, USA, Okt. 2003), S. 248, [Online]: http://www.cs.cmu.edu/~aura/docdir/sensay_iswc.pdf, 2003

[77] H. Uszkoreit, F. Xu, W. Liu, J. Steffen, I. Aslan, J. Liu, C. Müller, B. Holtkamp, M. Wojciechowski: „A Successful Field Test of a Mobile and Multilingual Information Service System COMPASS2008”, Proceedings of the 12th International Conference on Human-Computer Interaction (HCI’07, Beijing, China, Juli 2007), LNCS 4553, Springer, S. 1047-1056, 2007

[78] L. Fritsch, J. Muntermann: “Aktuelle Hinderungsgründe für den kommerziellen Erfolg von Location Based Service-Angeboten“, Proceedings zur 5. Konferenz Mobile Commerce Technologien und Anwendungen (MCTA 2005, Augsburg, Deutschland, Febr. 2005), LNI, Gesellschaft für Informatik GI, [Online]: http://subs.emis.de/LNI/Proceedings/ Proceedings59/GI-Proceedings.59-11.pdf, S. 143-156, 2005

[79] M. Samulowitz, F. Michahelles, C. Linnhoff-Popien: „Capeus: An architecture for context-aware selection and execution of services“, New developments in distributed applications and interoperable systems, Kluwer Academic Publishers, S. 23-39, 2001

[80] B. Schilit, N. Adams, R. Want: “Context-aware computing applications”, IEEE Workshop on Mobile Computing Systems and Applications (Santa Cruz, CA, USA, Dez. 1994), S. 85-90, 1994

[81] A. Held, S. Buchholz, A. Schill: “Modeling of context information for pervasive computing applications”, Proceedings of the Eighth World Multi-Conference on Systemics, Cybernetics and Informatics/International Conference on Information Systems, Analysis and Synthesis (SCI 2002/ISAS 2002, Orlando, FL, USA, Juli 2002), [Online]: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.20.3976&rep=rep1&type=pdf, 2002

[82] J. Indulska, R. Robinsona, A. Rakotonirainy, K. Henricksen: “Experiences in using cc/pp in context-aware systems”, Proceedings of the 4th international conference on mobile data management (MDM2003, Melbourne, Australien, Jan. 2003), LNCS 2574, Springer, S. 247-261, 2003

[83] N. Ryan: “ConteXtML: Exchanging Contextual Information between a Mobile Client and the FieldNote Server”, [Online]: http://www.cs.kent.ac.uk/projects/mobicomp/fnc/ConteXtML.html, 1999

[84] J. Bauer: “Identification and Modeling of Contexts for Different Information Scenarios in Air Traffic”, Technische Universität Berlin, Diplomarbeit, [Online]:

http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.98.7895&rep=rep1&type=pdf, 2003

[85] K. Henricksen, J. Indulska, A. Rakotonirainy: “Generating Context Management Infrastructure from High-Level Context Models”, Proceedings of the 4th International Conference on Mobile Data Management (MDM2003, Melbourne, Australien, Jan. 2003), S. 1-6, 2003

[86] A. Schmidt, M. Beigl, H.-W. Gellersen: „There is more to context than location“, Computers and Graphics, Band 23, Nr. 6, Elsevier Science, S. 893-901, 1999

[87] K. Cheverst, K. Mitchell, N. Davies: “Design of an object model for a context sensitive tourist GUIDE”, Computers and Graphics, Band 23, Nr. 6, Elsevier Science, S. 883-891, 1999

[88] S. Haseloff: “Context Awareness in Information Logistics”, Technische Universität Berlin, Dissertation, [Online]: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.88.8498&rep=rep1&type=pdf , 2005

[89] M. Reinfrank: „Formeln und Modelle – Wissensrepräsentation mit Logik“, in P. Struß: Wissensrepräsentation, Oldenbourg Verlag, S. 23, 1991

[90] J. McCarthy, Buvac: „Notes on formalizing context“, Proceedings of the Thirteenth International Joint Conference on Artifical Intelligence (Chambery, France), Morgan Kaufmann, S. 555-560, 1993

[91] C. Ghidini, F. Giunchiglia: “Local models semantics, or contextual reasoning = locality + compatibility”, Artificial Intelligence, Band 127, Nr. 2, Elsevier Science, S. 221-259, 2001

[92] V. Akman, M. Surav: “The use of situation theory in context modeling”, Computational Intelligence, Band 13, Nr. 3, Blackwell Publishers, S. 427-438, 1997

[93] T. Strang: “Service Interoperability in Ubiquitous Computing Enviroments”, Ludwig-Maximilian-Universität München, Dissertation, 2003

[94] T. Strang, C. Linnhoff-Popien, K. Frank: „Applications of a Context Ontology Language“, Proceedings of International Conference on Software, Telecommunications and Computer Networks (SoftCom2003, (Split & Dubrovnik, Kroatien) & (Ancona & Venice, Italien), Okt. 2003, S. 14-18, [Online]: http://elib.dlr.de/7326/1/SoftCom2003CameraReady Version.pdf, 2003

[95] X. Wang, D. Zhang, T. Gu, H. Pung: „Ontology Based Context Modeling and Reasoning using OWL“, Workshop Proceedings of the second IEEE Conference on Pervasive Computing and Communications (PerCom2004, Orlando, FL, USA, März 2004), S. 18-24, 2004

[96] H. Chen, T. Finin, A. Joshi: „Using OWL in a Pervasive Computing Broker“, Proceedings of Workshop on Ontologies in Open Agent Systems (AAMAS 2003, Melbourne, Australien, Juli 2003), [Online]: http://www.cs.umbc.edu/~finin//papers/aamas03a.pdf, 2003

[97] OMG: OMG's MetaObject Facility, [Online]: http://www.omg.org/mof/, Apr. 2008

[98] OMG: Unified Modeling Language (UML), Version 2.1.2, [Online]: http://www.omg.org/technology/documents/formal/uml.htm, Apr. 2008

[99] J. Seiie, W. Woontack: "Unified Context Describing User-Centric Situation: Who, Where, When, What, How and Why", IPSJ SIG Technical Reports, Band 2005, Nr. 60, S. 175-180, 2005

[100] X. Wang, D. Zhang, J. Dong, C. Chin, S. Hettiarachchi: "Semantic Space: A Semantic Web Infrastructure for Smart Spaces", IEEE Pervasive Computing, Band 3, Nr. 3, S. 32-39, 2004

[101] K. Henricksen, J. Indulska, A. Rakotonirainy: "Modeling Context Information in Pervasive Computing Systems", Proceedings of the International Conference on Pervasive Computing (Pervasive 2002, Zürich, Schweiz, Aug. 2002), LNCS 2414, S. 167-180, 2002

[102] P. Nurmi, M. Martin, J. Flanagan: „Enabling Proactiveness Through Context Prediction“, Workshop on Context Awareness for Proactive Systems (CAPS2005, Helsinki, Finnland, Juni 2005), [Online]: http://www.cs.helsinki.fi/u/ptnurmi/papers/caps2005.pdf, 2005

[103] R. Mayrhofer: "Context Prediction Based on ContextHistories: Expected Benefits, Issues and Current State-of-the-Art", First International Workshop on Exploiting Context Histories in Smart Environments (ECHISE 2005, München, Deutschland), [Online]: http://www.ipsi.fraunhofer.de/ambiente/echise2005/papers/echise2005-s17-ContextPredictionBasedOnContextHistories-Mayrhofer.pdf, 2005

[104] B. Truong, Y. Lee, S. Lee: "Modeling Uncertainty in Context-Aware Computing", Proceedings of the Fourth Annual ACIS International Conference on Computer and Information Science (ICIS'05, Jeju Island, Süd Korea, Juli 2005), S. 676-681, 2005

[105] S. Antifakos, A. Schwaninger, B. Schiele: "Evaluating the Effects of Displaying Uncertainty in Context-Aware Applications", Proceedings of the Sixth International Conference on Ubiquitous Computing (UbiComp 04, Nottingham, England, Sept. 2004), LNCS 3205, Springer, S. 54-69, 2004

[106] Wikipedia, WLAN, [Online]: http://de.wikipedia.org/wiki/Wireless_Local_Area_Network, Juni 2010

[107] Wikipedia, Bluetooth, [Online]: http://de.wikipedia.org/wiki/Bluetooth, Juni 2010

[108] T. Basten, L. Benini, A. Chandrakasan, M. Lindwer, J. Liu, R. Min, F. Zhao: „Scaling into Ambient Intelligence“, Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE'03, München, Deutschland, März 2003), S. 76-81, 2003

[109] D. Wagelaar: "Towards a Context-Driven Development Framework for Ambient Intelligence", Proceedings of the 24th International Conference on Distributed Computing Systems Workshops (ICDCS 2004, Tokio, Japan, März 2004), IEEE Computer Society, S. 304-309, 2004

[110] P. Levis, S. Madden, J. Polastre, R. Szewczyk, K. Whitehouse, A. Woo, D. Gay, J. Hill, M. Welsh, E. Brewer and D. Culler: "TinyOS: An Operating System for Sensor Networks", In: Ambient Intelligence, Springer Berlin Heidelberg, S. 115-148, 2005

[111] V. Issarny, D. Sacchetti, F. Tartanoglu, F. Sailhan, R. Chibout, N. Levy, A. Talamona: "Developing Ambient Intelligence Systems: A Solution based on Web Services", In: Journal of Automated Software Engineering, Springer, Band 12, Nr. 1, S. 101-137, 2005

[112] N. Gui, V. De Florio, H. Sun, C. Blondia: "A Service-oriented Infrastructure for Ambient Assisted Living Communities", Proceedings of the First IEEE WoWMoM Workshop on Adaptive and DependAble Mission- and bUsiness-critical mobile Systems (ADAMUS 2007, Helsinki, Finnland, Juni 2007), IEEE Computer Society, S. 1-6, 2007

[113] M. Wegdam: "AWARENESS: A project on Context AWARE mobile NEtworks and ServiceS", Proceedings of the 14th IST Mobile and Wireless Communications Summit 2005 (Dresden, Deutschland, Juni 2005), [Online]: http://www.eurasip.org/Proceedings/Ext/IST05/papers/293.pdf, 2005

[114] D. Preuveneers, J. Van den Bergh, D. Wagelaar, A. Georges, P. Rigole, T. Clerckx, Y. Berbers, K. Coninx, V. Jonckers, K. De Bosschere: "Towards an extensible context ontology for Ambient Intelligence", Ambient Intelligence: Second European Symposium (EUSAI 2004, Eindhoven, Niederlande), LNCS 3295, Springer, S. 148-159, 2004

[115] C. Kunze, C. Holtmann, A. Schmidt, W. Stork: „Kontextsensitive Technologien und Intelligente Sensorik für Ambient-Assisted-Living-Anwendungen“, 1. Deutscher Kongress Ambient Assisted Living (AAL 08), VDE Verlag, [Online]: http://publications.andreas.schmidt.name/Kunze_Holtmann_Schmidt_Stork_AAL08.pdf, 2008

[116] P. Nurmi, P. Floréen, M. Przybilski, Greger Lindén: „A Framework for Distributed Activity Recognition in Ubiquitous Systems", Proc. International Conference on Artificial Intelligence (IC-AI 2005, Las Vegas, NE, USA, Juni 2005), CSREA Press, Band 2, S. 650-655, 2005

[117] Sensor + Test 2008, [Online]: http://www.sensor-test.com, 2008

[118] K. Chang, S. Liu, H. Chu, J. Hsu, C. Chen, T. Lin, C. Chen, P. Huang: „The Diet-Aware Dining Table: Observice Dietary Behaviors over a Tabletop Surface“, Proceedings of the

International Conference on Pervasive Computing (Pervasive 2006, Dublin, Irland, Mai 2006), LNCS 3968, Springer, S. 366-382, 2006

[119] Y. Isoda, S. Kurakake, K. Imai: „Context-Aware Computing System for Heterogeneous Applications“, Proceedings of the First International Workshop on Personalized Context Modeling and Management for UbiComp Applications (ubiPCMM, Tokio, Japan, Sept. 2005), S. 17-25, 2005

[120] C. Shin, D. Han, W. Woo: „Conflict Management for Media Services by exploiting Service Profile and User Preferences“, Proceedings of the First International Workshop on Personalized Context Modeling and Management for UbiComp Applications (ubiPCMM, Tokio, Japan, Sept. 2005), S. 48-57, 2005

[121] J. Nehmern, A. Karshmer, M. Becker, R. Lamm: „Living Assistance Systems – An Ambient Intelligence Approach“, Proceedings of the 28th International Conference on Software Engineering (ICSE, Shanghai, China, Mai 2006), S. 43-50, 2006

[122] Bundesministerium für Bildung und Forschung: AAL - Geförderte EU-Projekte, [Online]: http://www.aal-deutschland.de/geforderte-eu-projekte, April 2008

[123] S. Ahn, D. Kang, H. Ko: „Modeling of Context-based Interaction Pattern“, Proceedings of the First International Workshop on Personalized Context Modeling and Management for UbiComp Applications (ubiPCMM, Tokio, Japan, Sept. 2005), S.1-4, 2005

[124] M. Wojciechowski, B. Holtkamp: „Experiences with Situation Aware Service Provision”, First International Workshop on Personalization in Grid and Service Computing (PGSC07, Urumchi, China, Aug. 2007), Proceedings of the Sixth International Conference on Grid and Cooperative Computing, S. 863-870, 2007

[125] T. Zhang, B. Brügge: „Empowering the User to Build Smart Home Applications“, International Conference on Smart Home and Health Telematics (ICOST'04, Singapur, Sept. 2004), In: Toward a Human-Friendly Assistive Environment, IOS Press, S. 170-176, 2004

[126] S. Ballegaard, J. Bunde-Pedersen, J. Bardram: „Where to, Roberta?: Reflecting on the Role of Technology in Assisted Living”, Proceedings of the 4th Nordic conference on Human-Computer Interaction (nordichi 2006, Oslo, Norwegen, Okt. 2006), ACM, S. 373-376, 2006

[127] G. Vossen: “Datenmodelle, Datenbanksprachen und Datenbank-Management-Systeme”, 2. Auflage, Addison-Wesley, ISBN 3-89319-566-1, 1994

[128] M. Samulowitz: “Kontextadaptive Dienstnutzung in Ubiquitous Computing Umgebungen”, Ludwig-Maximilians-Universität München, Dissertation, [Online]: http://edoc.ub.uni-muenchen.de/591/1/Samulowitz_Michael.pdf, 2002

[129] University of Kent: “MobiComp:About – MobiComp”, [Online]: http://www.mobicomp.org/mediawiki/index.php/MobiComp:About, 2006

[130] Universität Stuttgart, Projekt SFB 627 Nexus, [Online]: http://www.ipvs.uni-stuttgart.de/abteilungen/vs/forschung/projekte/SFB627, Juni 2010

[131] K. Rothermel: “Vorstellung des Sonderforschungsbereichs 627, Umgebungsmodelle für mobile kontextbezogene Systeme”, Universität Stuttgart, interner Foliensatz, 2003

[132] K. Henricksen, S. Livingstone, J. Indulska. ”Towards a hybrid approach to context modelling, reasoning and interoperation”, Proceedings of the First International Workshop on Advanced Context Modelling, Reasoning and Management (Nottingham, England, Sept. 2004), [Online]: http://www.itee.uq.edu.au/~pace/cw2004/Paper22.pdf, 2004

[133] The CoDAMoS Project: Context-Driven Adaptation of Mobile Services, [Online]: http://www.cs.kuleuven.ac.be/distrinet/projects/CoDAMoS/, 2003

[134] W3C: “OWL Web Ontology Language Guide”, [Online]: http://www.semaweb.org/dokumente/w3/TR/2004/REC-owl-guide-20040210-DE.html, 2004

[135] OASIS: UDDI Spezifikation, [Online]: http://www.oasis-open.org/committees/uddi-spec/doc/tcspecs.htm, Apr. 2008

[136] W3C: Extensible Markup Language (XML) 1.0 (Fith Edition), [Online]: http://www.w3.org/TR/REC-xml/, Nov. 2008

[137] W3C: XML Schema, [Online]: http://www.w3.org/XML/Schema, Okt. 2004

[138] W3C: XSL Transformations (XSLT) Version 1.0, [Online]: http://www.w3.org/TR/xslt, Nov. 1999

[139] M. Born, E. Holz, O. Kath: “Softwareentwicklung mit UML 2”, Addison-Wesley, ISBN 3-8273-2086-0, 2004

[140] OMG: „OMG Unified Modeling Language (OMG UML), Infrastructure“, [Online]: http://www.omg.org/spec/UML/2.3/Infrastructure/PDF , 2010

[141] OMG: „OMG Unified Modeling Language (OMG UML), Superstructure”, [Online]: http://www.omg.org/spec/UML/2.3/Superstructure/PDF, 2010

[142] OMG: “MOF 2.0/XMI Mapping, Version 2.1.1”, [Online]: http://www.omg.org/cgi-bin/doc?formal/07-12-01.pdf, 2007

[143] Eclipse: „Eclipse Modeling Framework Project (EMF)“, [Online]: http://www.eclipse.org/modeling/emf/, 2010

[144] Eclipse: “Eclipse.org home”, [Online]: http://www.eclipse.org/, 2010

[145] W3C: „RDF Vocabulary Description Language 1.0: RDF Schema“, [Online]: http://www.w3.org/TR/rdf-schema/“, 2003

[146] DAML: “DAML+OIL”, [Online]: http://www.daml.org/2001/03/daml+oil-index.html”, 2001

[147] M. Kifer, G. Lausen: “F-logic: a higher-order language for reasoning about objects, inheritance, and scheme”, ACM SIGMOND, Band 18, Nummer 2, S. 134-146, 1989

[148] B. Holtkamp, N. Weißenberg, M. Wojciechowski, R. Gartmann: “Matching dynamic demands of mobile users with dynamic service offers”, P. Rittgen: Handbook of Ontologies for Business Interaction. Hershey: Information Science Reference, S. 278-293, 2007

[149] S. Cranefeld, M. Purvis: “UML as an Ontology Modelling Language”, Proceedings of the Workshop on Intelligent Information Integration (IJCAI-99, Stockholm, Schweden, Juli 1999), S. 46-53, 1999

[150] OMG: “Ontology Definition Metamodel, Version 1.0”, [Online]: http://www.omg.org/spec/ODM/1.0/PDF/”, 2009

[151] A. Brucker, J. Doser: “Metamodel-based UML Notations for Domain-specific Languages“, 4th International Workshop on Language Engineering (ATEM 2007, Nashville, TN, USA, Sept. 2007), LNCS 5002, S. 28-33, 2008

[152] T. Lodderstedt, D. Basin, J. Doser: “SecureUML: A UML-Based Modeling Language for Model-Driven Security”, Proceedings of the 5th International Conference on the Unified Modeling Language (UML 2002, Dresden, Deutschland, Sept. 2002), LNCS 2460, S. 426-441, 2002

[153] M. Huebscher, H. McCann: “An adaptive middleware framework for context-aware applications“, Personal Ubiquitous Computing, Band 10, Nr. 1, Springer, S. 12-20, 2006

[154] K. Unterstein: „Realisierung einer Context Sensor Registry für eine verteilte Kontextinfrastruktur“, Fraunhofer ISST, Diplomarbeit, 2009

[155] T. Strang, C. Linnhoff-Popien, K. Frank: “CoOL: A Context Ontology Language to enable Contextual Interoperability”, 5th International Conference on Distributed Applications and Interoperable Systems (DAIS 2003, Athen, Griechenland, Juni 2005), LNCS 2893, S. 236-247, 2003

[156] I. Horrocks: “OWL: a Description Logic Based Ontology Language”, P. v. Beek: Principles and Practice of Constraint Programming – CP 2005, LNCS 3709, S. 5-8, 2005

[157] Wikipedia, Web Ontology Language, [Online]: http://en.wikipedia.org/wiki/Web_Ontology_Language, 2010

[158] OMG: “Object Constraint Language OMG Available Specification Version 2.0”, [Online]: http://www.omg.org/cgi-bin/doc?formal/2006-05-01, 2006

[159] J. Brezillon, P. Brezillon: „Context modeling: context as a dressing of a focus“, Proceedings of the 6th international and interdisciplimanry conference on Modeling and using context (CONTEXT’07, Roskilde University, Dänemark, Aug. 2007) , LNCS 4635, S. 136-149, 2007

[160] S. Yau, J. Liu: “Hierarchical Situation Modeling and Reasoning for Pervasive Computing”, Proceedings of the Fourth IEEE Workshop on Software Technologies For Future Embedded and Ubiquitous Systems, and the Second international Workshop on Collaborative Computing, integration, and Assurance (Seus-Wccia'06, Gyeongju, Korea, April 2006), IEEE Society, S. 5-10, 2006

[161] P. Costa, L. Pires, M. van Sinderen, J. Filho: “Towards a service platform for mobile context-aware applications”, First International Workshop on Ubiquitous Computing (IWUC 2004, Proto, Portugal, April 2004), INSTICC Press, S. 48-62, 2004

[162] Wikipedia, Lightweight Directory Access Protocol, [Online]: http://de.wikipedia.org/wiki/Lightweight_Directory_Access_Protocol, August 2010

[163] OASIS: “Directory Services Markup Language v2.0”, [Online]: http://www.oasis-open.org/committees/dsml/docs/DSMLv2.doc, 2001

[164] C. Becker, F. Dürr: “On location models for ubiquitous computing”, Personal Ubiquitous Computing, Band 9, Nr. 1, Springer, S. 20-31, 2005

[165] S. W. Loke: “Facing uncertainty and consequence in context-aware systems: towards an argumentation approach”, Proceedings of the First International Workshop on Advanced Context Modelling, Reasoning and Management (Nottingham, England, Sept. 2004), [Online]: http://citeseer.ist.psu.edu/loke04facing.html, 2004

[166] W3C: “Web Services Description Language (WSDL) 1.1”, [Online]: http://www.w3.org/TR/wsdl, 2001

[167] Wikipedia, Semantic Web Services, [Online]: http://de.wikipedia.org/wiki/Semantic_Web_Services, 2010

[168] WSSI WSMO working group: “web service modeling ontology”, [Online]: http://www.wsmo.org, 2006

[169] W3C: “OWL-S: Semantic Markup for Web Services”, [Online]: http://www.w3.org/Submission/OWL-S/, 2004

[170] University of Georgia: “METEOR-S: Semantic Web Services and Processes”, [Online]: http://lsdis.cs.uga.edu/projects/meteor-s/, 2010

[171] Wikipedia, UNSPSC, [Online]: http://de.wikipedia.de/wiki/UNSPSC, 2009

[172] I. Horrocks, P. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, M. Dean: “Swrl: A semantic web rule language combining owl and ruleml”, [Online]: Available at http://www.daml.org/2003/11/swrl/, 2003

[173] KIF: “Knowledge Interchange Format: Draft proposed American National Standard (dpans)”, [Online]: http://logic.stanford.edu/kif/dpans.html, 1998

[174] M. Ghallab: “PDDL-The Planning Domain Definition Language V. 2”, Technical Report CVC TR-98-003/DCS TR-1165, Yale Center for Computational Vision and Control, 1998

[175] M. Keidl, A. Kemper: “Towards Context-Aware Adaptable Web Services”, Proceedings of the 12th International World Wide Web Conference (WWW’2003, Budapest, Ungarn, Mai 2003), S. 55-65, 2003

[176] I. Chen, S. Yang, J. Zhang: „Ubiquitous Provision of Context Aware Web Services“, International Conference on Services Computing (SCC’06, Chicago, USA, Sept. 2006), S. 60-68, 2006

[177] A. K. Dey, Raffay Hamid, Chris Beckmann, Ian Li, Daniel Hsu: “a CAPpella: Programming by Demonstration of Context-Aware Applications”, Proceedings of ACM Conference on Human Factors in Computing Systems, (CHI 2004, Wien, Österreich), S. 33-40, 2004

[178] Eclipse Modeling Framework Project EMF, [Online]: http://www.eclipse.org/modeling/emf/, 2010

[179] Metz mecaHome+, [Online]: http://www.metz.de/de/presse/pressemeldungen-tv-erlebnis/2009/09-07-mecahome.html, Jan. 2010

[180] Z. Wang, S. Elbaum, D. Rosenblum: “Automated Generation of Context-Aware Tests” Proceedings of the 29th International Conference on Software Engineering (ICSE ’07, Mineapolis, USA, Mai 2007), S. 406-415, 2007

[181] T. Beer, J. Rasinger, W. Höpken, M. Fuchs, H. Werthner: ”Exploiting E-C-A Rules for Defining and Processing Context-Aware Push Messages”, International RuleML Symposium on Rule Interchange and Applications (Orlando, FL, USA, Okt. 2007), LNCS 4824, Springer, S. 199-206, 2007

[182] F. Dantas, T. Batista, N. Cacho, A. Garcia: ”Towards Aspect-Oriented Programming for Context-Aware Systems: A Comparative Study”, First International Workshop on Software Engineering for Pervasive Computing Applications, Systems, and Environments (SEPCASE ’07, Minneapolis, USA, Mai 2007), Proceedings of the 29th International Conference on Software Engineering Workshops, S. 190-193, 2007

[183] R. Reichle, M. Wagner, M. Khan, K. Geihs, M. Valla, C. Fra, N. Paspallis, G. Papadopoulos: “A Context Query Language for Pervasive Computing Environments”, Proceedings of the 5th IEEE Workshop on Context Modeling and Reasoning (CoMoRea, Hong Kong, März 2005), S. 434-440, 2008

[184] T. Gu, H. Pung, D. Zhang; “A Middleware for Building Context-Aware Mobile Services”, Proceedings of IEEE Vehicular Technology Spring Conference (VTC2004-Spring, Milan, Italien, Mai 2004), Band 5, S. 2656 - 2660, 2004