Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Projektbericht Masterprojekt 1

Malte Kantak

Vorarbeit für einen Erreichbarkeitsagenten

31. August 2012

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Inhaltsverzeichnis

1. Einleitung **3**
1.1. Vision . . . 3

2. Entwicklungsumfeld **4**

3. Framework **5**
3.1. Entwurf . . . 5
3.2. Aufbau . . . 6
3.2.1. Komponenten des Erreichbarkeitsagenten . . . 6
3.2.2. Komponentenschnittstellen . . . 8
3.2.3. Benutzerschnittstelle . . . 9
3.3. Implementierungsstand . . . 10

4. Zusammenfassung **12**
4.1. Ausblick . . . 12

Literatur **13**

A. Nachrichtenformate **15**

Abbildungsverzeichnis

1. Entwurf des Erreichbarkeitsagenten . . . 5
2. Beispielimplementation einer *Benutzerschnittstelle* . . . 11
3. Kontrolloptionen der *Benutzerschnittstelle* . . . 11

Listings

1. ReachabilityInformation . . . 15
2. ReachabilityInstruction . . . 15
3. ConfigOrder . . . 15
4. UbisenseTrackingMessage . . . 16
5. StressCompanionInformation . . . 16

1. Einleitung

Mit zunehmendem Grad an Heimarbeit und Technisierung verschwimmt die Grenze zwischen Arbeit und Privatleben immer mehr. Diese Vermischung von verschiedenen Kontexten und das stetige Wechseln zwischen diesen, kann zu einer stark verminderten Produktivität führen, da sich die Person bei jeder Kontextänderung erneut auf die neue Aufgabe einstellen muss (vgl. [UMI]). Sehr problematisch wird diese Kontextänderung, wenn eine Hauptaktivität durch eine zeitlich begrenzte Nebenaktivität vorübergehend unterbrochen wird. Hervorgerufen werden kann eine solche Unterbrechung zum Beispiel durch den Kommunikationsversuch einer kontextfremden Person. Aus dieser Problematik wurde in der Arbeit [Kan12a] die Idee eines Erreichbarkeitsagenten entwickelt. Dieser Softwareagent soll die Aktivitäten des Bewohners im Smart-Home Kontext mithilfe diverser Sensoren überwachen. Ziel des Agenten ist es die erhaltenen Daten zu analysieren und auf Basis von Erfahrungen und bestimmten Regeln daraus die aktuelle Erreichbarkeit des Bewohners abzuleiten. Aufgrund der beschriebenen Vermischung verschiedener Kontexte soll die Relation des aktuellen Kontextes und der kontaktsuchenden Personengruppe berücksichtigt werden. Diese abgeleiteten Informationen sollen dann anderen Systemen (wie z. B. einer Türklingel [Bor12] oder einem Stressagenten [Lin12]) zur Verfügung gestellt werden, welche diese Informationen dann verwenden können um sowohl die Produktivität, als auch das Wohlbefinden des Bewohners zu steigern.

1.1. Vision

Im Rahmen diese Masterprojektes sollen die Grundvoraussetzungen für einen Erreichbarkeitsagenten geschaffen werden. Dies umfasst sowohl die Anbindung von Sensoren als auch die Verarbeitung dieser Informationen und der Bereitstellung dieser für andere Systeme. Die Anwendung soll in der Umgebung des Living Place Hamburgs[1] entwickelt und evaluiert werden. Dazu soll sich die Software in die Infrastruktur dieser Umgebung integrieren, die vorhandenen Sensoren verwenden und die Informationen auf der zentralen ActiveMQ[2] publizieren. Zur Auswertung der Sensoren und der Ableitung der Erreichbarkeitszustände soll ein lernendes System zum Einsatz kommen, welches sich auf die persönlichen Präferenzen des Bewohners einstellt. Dazu ist es wünschenswert sowohl die Reaktion des Bewohners auf einen Kommunikationsversuch auszuwerten (indirektes Feedback), als auch dem Bewohner die Möglichkeit zu geben, dem System aktiv seine aktuelle Erreichbarkeit mitzuteilen (direktes Feedback) bzw. korrigierend in die Zustandsermittlung einzugreifen.

[1] ein Smart-Home der Hochschule für angewandte Wissenschaften Hamburg [LKG+10] – http://www.livingplace.org – Zuletzt besucht: 28.08.2012

[2] System zur Weiterleitung von Nachrichten in so genannten Nachrichtenschlangen – http://activemq.apache.org/ – Zuletzt besucht: 28.08.2012

2. Entwicklungsumfeld

Der Erreichbarkeitsagent soll im Rahmen des Living Place Hamburg entwickelt und evaluiert werden. Die Infrastruktur des Living Place basiert aktuell auf einer ActiveMQ mit angeschlossener MongoDB[3]. Es gibt verschiedene Arten von Softwareagenten, welche in dieser Architektur miteinander interagieren. Einige Anwendungen stellen lediglich Daten zur Verfügung. Diese Softwareagenten werden im weiteren Verlauf als „Sensoren“ bezeichnet. Diese Daten können dann von zwei Gruppen von Softwareagenten weiterverwendet werden. Die eine Gruppe verwendet die Daten um auf deren Basis bestimmte Reaktionen zu erzeugen. Dies kann z. B. das Schließen der Rollos, das Anpassen der Lichteinstellungen oder die Anzeige von Information für den Bewohner sein. Diese Softwareagenten werden im Weiteren als „Aktoren“ bezeichnet. Die letzte Gruppe der Softwareagenten verwendet die Daten der ActiveMQ um diese zu aggregieren oder daraus weiterführende Schlussfolgerungen zu ziehen. Die daraus gewonnen Erkenntnisse werden dann erneute für andere Anwendungen auf der ActiveMQ bereit gestellt. Diese werden im Weiteren als „Interpreter“ bezeichnet.

Der zu entwickelnde Erreichbarkeitsagent gliedert sich nach dieser Einteilung als „Interpreter“ in die Systemlandschaft ein. Er soll verschiedene Informationen von Sensoren und anderen Interpretern verwenden können, um daraus den Erreichbarkeitszustand des Bewohners abzuleiten und diese Information dann wieder auf der ActiveMQ zur Verfügung zu stellen.

Die Kommunikation der verschiedenen Agenten über die ActiveMQ erfolgt in JSON[4], wodurch die Programmiersprache bei der Entwicklung weiterer Agenten frei wählbar bleibt. Durch diese einfache Schnittstelle können leicht weitere Agenten in die Systemlandschaft integriert werden. Da im Living Place kein allgemeines JSON Format zum Nachrichtenaustausch vorgegeben ist, verwendet jeder Sensor sein eigenes Nachrichtenformat. Dies soll bei der Entwicklung des Erreichbarkeitsagenten berücksichtigt werden.

Beispiele für schon im Living Place integrierte Sensoren sind z. B. das intelligente Bett [Har11] und das UbiSense Tracking System [EKV+11]. Darüber hinaus befinden sich aktuell weitere Projekte in der Entwicklung, wie z. B. ein StressCompanion [Lin12], ein Couchtisch [Kü12] und eine mobile Türklingel [Bor12] welche sich entweder als Sensoren oder als Datenabnehmer eignen werden.

[3] Eine dokumentenbasierte open-source Datenbank – http://www.mongodb.org/ – Zuletzt besucht: 26.08.2012

[4] **J**ava**S**cript **O**bject **N**otation – Ein Datenformat zur Kodierung von Objekten, welches sowohl von Menschen als auch Maschinen lesbar ist.

3. Framework

Um den Erreichbarkeitsagenten so anpassungsfähig wie möglich zu gestalten, soll im Rahmen diese Projektes ein flexibles Framework erstellt werden. Dieses Framework soll die Möglichkeit bieten sehr einfach und schnell neue Sensoren zu integrieren und verschiedene Regeln, abhängig von den Vorlieben des jeweiligen Bewohners zu definieren. Es soll sich in die Systemlandschaft des Living Place integrieren und dementsprechend einen Nachrichtenaustausch über die ActiveMQ Schnittstelle realisieren. Darüber hinaus sollen verschiedene lernende Verfahren als Reasoner[5] auf ihre Verwendbarkeit evaluiert werden, wodurch auch diese einfach austauschbar gestaltet werden sollen.

3.1. Entwurf

Um das einfache Hinzufügen von verschiedenen Sensoren und Reasonern zu verwirklichen, muss das Framework eine losen Kopplung zwischen den einzelnen Komponenten realisieren. Der Entwurf des Erreichbarkeitsagenten ist in Abbildung 1 dargestellt.

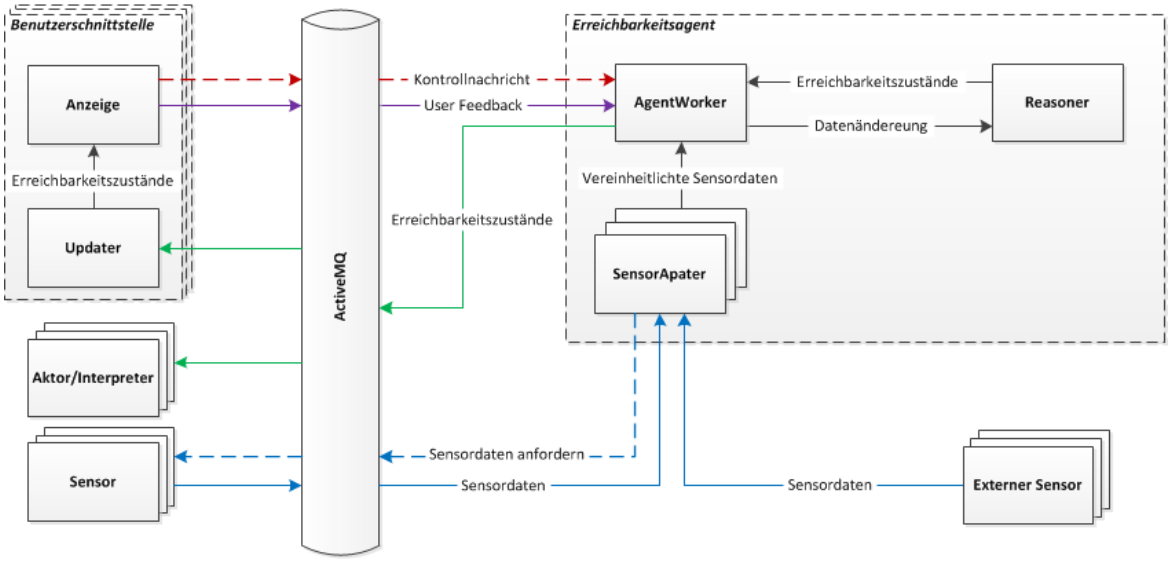

Abbildung 1: Entwurf des Erreichbarkeitsagenten

Aus dem Entwurf wird deutlich, dass der Großteil des Nachrichtenaustauschs über die ActiveMQ Schnittstelle erfolgt. Teil dieses Projektes sind die jeweils in schraffierter Linie eingefassten Systemteile. Zum Einen das Hauptsystem (*Erreichbarkeitsagent*) und zum Anderen die Schnittstelle des Systems zum Bewohner (*Benutzerschnittstelle*). Die anderen Systeme

[5] dt. logisch Denkender – Ein System, welches Daten evaluiert um aus ihnen neue Erkenntnisse zu gewinnen.

(*Aktor/Interpreter*, *Sensor* und *Externer Sensor*) stellen entweder schon in der Systemlandschaft des Living Places vorhandene oder zukünftig zu entwickelnde Sensoren, Aktoren oder Interpreter dar. Die Sensoren produzieren Daten, welche sie auf der ActiveMQ publizieren. Diese werden von den *SensorAdaptern* gelesen, ausgewertet und in einem einheitlichen Format an den *AgentWorker* weitergegeben. Pro Sensortyp, welcher im Erreichbarkeitsagenten berücksichtigt werden soll, muss ein solcher *SensorAdapter* existieren. Zusätzlich können auch weitere *SensorAdapter* implementiert sein, welche ihre Informationen nicht von der ActiveMQ beziehen (z. B. ein *SensorAdapter* für die aktuelle Uhrzeit). Bei einigen Sensoren kann es notwendig sein, die Sensordaten durch eine Anfragenachricht auszulesen. Auch dies soll über den *SensorAdapter* gelöst werden. Der *AgentWorker* verwaltet die *SensorAdapter* und den *Reasoner*. Er leitet die Sensordaten an den *Reasoner* weiter, welcher dann auf Basis bestimmter Regeln und Erfahrungen die aktuelle Erreichbarkeit des Bewohners in Relation zu verschiedenen, definierten Personengruppen ermittelt. Diese Zustände werden dann an den *SensorAdapter* zurück gegeben, welcher die Daten auf der ActiveMQ bekannt gibt. Diese Erreichbarkeitszustände können von verschiedenen *Aktoren* (z. B. Türklingel [Bor12]) und *Interpretern* (z. B. StressCompanion [Lin12]) gelesen und verwendet werden. Darüber hinaus werden diese Zustände vom *Updater* in potentiell mehreren *Benutzerschnittstellen* gelesen und dem Bewohner zur Anzeige gebracht. Von diesen *Benutzerschnittstellen* hat der Bewohner die Möglichkeit, sowohl ein aktives Feedback an den *Erreichbarkeitsagenten* zu geben, als auch diesen vorübergehend zu deaktivieren oder zu beenden.

3.2. Aufbau

Im Folgenden werden die im letzten Abschnitt beschriebenen Komponenten, welche Teil diese Projektes sind, definiert und beschrieben. Es werden sowohl die Schnittstellen zu anderen Systemen über die ActiveMQ, als auch die der einzelnen Komponenten der beiden zu entwickelnden Systeme (*Erreichbarkeitsagent*, *Benutzerschnittstelle*) erläutert. Im Anschluss daran, wird detailliert auf die Funktionen der Benutzerschnittstelle eingegangen.

3.2.1. Komponenten des Erreichbarkeitsagenten

Der *Erreichbarkeitsagent* setzt sich aus den drei Hauptkomponenten *SensorAdapter*, *AgentWorker* und *Reasoner* zusammen.

SensorAdapter
Ein *SensorAdapter* liefert in erster Linie Daten, welche vom *Reasoner* bewertet werden sollen. Diese Daten können verschiedene Ausprägungen haben. So können sie zum Beispiel Eigenschaften der Wohnung, der Umwelt oder des Bewohners näher beschreiben. Dabei

müssen die Sensoren nicht zwingend über die ActiveMQ kommunizieren, es können auch andere Quellen verwendet werden, solange der *SensorAdapter* einem bestimmten Interface[6] genügt. Im Wesentlichen bedeutet dies, dass er die Methode *getSensorOutput()* unterstützt, welche ein Objekt vom Typ *SensorData* zurückliefert. In diesem Objekt werden die Daten der verschiedenen *SensorAdapter* in einem einheitlichen Format aufbereitet. Dabei stellt der *SensorAdapter* die Schnittstelle zu den tatsächlichen Informationen dar. Er sammelt die notwendigen Informationen und bereitet diese so auf, dass sie vom *Reasoner* verarbeitet werden können.

AgentWorker

Der *AgentWorker* hat die Aufgabe, die verschiedenen *SensorAdapter* zu verwalten und die von ihnen erzeugten Informationen an den *Reasoner* weiterzugeben. Dazu initialisiert und startet er beim Aufruf des Programms die unterstützten *SensorAdapter* und den *Reasoner*. Danach prüft er während der Laufzeit jeweils, ob die *SensorAdapter* über neue Daten verfügen und gibt diese gegebenenfalls an den *Reasoner* weiter. Darüber hinaus gibt er die ermittelten Erreichbarkeitsinformationen an die ActiveMQ in Form von JSON Nachrichten weiter. Er sorgt dafür, dass aktives Feedback vom Bewohner an den *Reasoner* weiter gegeben wird und dass die definierten Kontrollnachrichten (Start, Sleep, Stop) entsprechend der Living Place Agent Life Cycle Spezifikation[7] verarbeitet werden.

Reasoner

Der *Reasoner* ist der Teil des Erreichbarkeitsagenten, welcher die Sensordaten auswertet und auf deren Basis die Erreichbarkeitszustände des Bewohners zu ermitteln versucht. Dabei soll er gewonnene Erfahrungen, durch direktes und indirektes Feedback, berücksichtigen, um sich so auf die Vorlieben und persönlichen Eigenschaften des jeweiligen Bewohners einstellen zu können. Denkbar sind hier viele verschiedene Verfahren, von regelbasierten Systemen mit relativ starren, vorgegebenen Grundregeln, über stochastische Verfahren wie Bayes Netzen[8], bis hin zu Klassifikatoren wie Support Vector Machines[9] und künstlichen neuronalen Netzen[10]. Entscheidend ist hierbei nur, dass der *Reasoner* die Möglichkeit bietet sein Verhalten dem Bewohner anzupassen, eine Art Gedächtnis aufweist und einer definierten Schnittstelle entspricht, wodurch die tatsächliche Implementierung unabhängig vom Framework des Erreichbarkeitsagentens ist.

[6] dt. Schnittstelle

[7] Internet Konvention bezüglich bestimmter Kontrollfunktionen, welche die Agenten unterstützen müssen

[8] Gerichteter azyklischer Graph, mit Zufallsvariablen als Knoten und Abhängigkeiten als Kanten

[9] Mathematisches Verfahren zur Klassifizierung von Objekten auf Basis von Stützvektoren

[10] Mathematische Annäherung an die Funktionsweise des menschlichen Gehirns

3.2.2. Komponentenschnittstellen

Damit die im Entwurf geforderte Modularität der einzelnen Komponenten gewährleistet werden kann, müssen die Schnittstellen sowohl zwischen dem Komponenten, als auch zwischen den verschiedenen Systemen klar definiert sein.

SensorAdapater und AgentWorker
Um es so einfach wie möglich zu machen, neue Sensoren an den Erreichbarkeitsagenten anzubinden, müssen die *SensorAdapter* jeweils bestimmte Methoden bereit stellen. Essentiell ist hierbei die schon erwähnte Methode *getSensorOutput()*. Diese Methode soll, unabhängig von dem jeweiligen Sensor und dessen Antwortformat, den jeweils aktuellen Messwert dises Sensors in Form eines einheitlichen Datenobjekts (*SensorData*) zurück liefern. Wie dieses Datenobjekt aufgebaut sein wird, ist zum jetzigen Zeitpunkt noch nicht abschließend definiert. In der aktuellen Version müssen die Daten jeweils in Form einer reellen Zahl dargestellt werden. Zusätzlich zu diesem Wert enthält das Datenobjekt auch noch die Information darüber, von welchem Sensor dieser Wert stammt, damit der *Reasoner* diesen Wert entsprechend einordnen kann. Der tatsächliche Aufbau des Objekts lässt sich erst im Anschluss an die Bestimmung der relevanten Sensoren definieren, welche noch zu erfolgen hat (vgl. *Ausblick*, Abschnitt 4.1).

AgentWorker und Reasoner
Der *AgentWorker* und der *Reasoner* interagieren über eine ähnliche Schnittstelle, wie der *AgentWorker* und die *SensorAgents*. Die erhaltenen *SensorData* Objekte werden alle vom *AgentWorker* an den *Reasoner* weitergereicht. Dieser liefert dann, im Anschluss an seinen Bewertungsprozess, eine Zuordnung von Erreichbarkeitszuständen zu den Personengruppen an den *AgentWorker* zurück, welcher diese dann auf der ActiveMQ veröffentlicht.

Updater und Anzeige
Der *Updater* bezieht die Erreichbarkeitsinformationen aus den Nachrichten des Erreichbarkeitsagenten über die ActiveMQ. Sobald Zustandsänderungen vorliegen, informiert er die Anzeige, welche daraufhin ihre Ausgabe aktualisiert.

Sensor und Erreichbarkeitsagent
Die Sensoren können auf verschiedene Arten und Weisen mit dem Erreichbarkeitsagenten kommunizieren. Für die Interaktion ist es nur wichtig, dass der entsprechende *SensorAdapter* das jeweilige Format des Sensors interpretieren kann. So besitzen zum Beispiel Sensoren, welche über die ActiveMQ mit dem Erreichbarkeitsagenten verbunden sind, häufig ein eigenes JSON Format, da sie ihre Daten nicht ausschließlich für den Erreichbarkeitsagenten publizieren. Die Beiden, in der aktuellen Version über die ActiveMQ an den Erreichbarkeitsagenten angeschlossenen, Sensoren *Ubisense Tracking* [EKV+11] und *StressCompanion* [Lin12] kommunizieren beide über verschiedene Formate (Vergleich Listing 4 (*UbiSenseTrackingMessage*) und 5 (*StressCompanionInformation*)).

Erreichbarkeitsagent und Aktoren/Interpreter
Der *Erreichbarkeitsagent* teilt den *Aktoren* bzw. *Interpretern* die aktuellen Erreichbarkeitszustände in Form von JSON Nachrichten über die ActiveMQ mit. Dazu werden Nachrichten produziert, welche dem Format aus Listing 1 (*ReachabilityInformation*) entsprechen. Eine solche Nachricht wird jeweils pro Personengruppe gesendet. Die einzelnen Personengruppen, welche zum selben Bewertungsvorgang des Reasoners gehören, sind über die gleiche *GroupId* verknüpft. Neben dieser Kennung ist jeweils die betroffene Gruppe (*Group*) und der ermittelte Status (*Status*) in dieser Nachricht enthalten. Diese Informationen werden um eine textuelle Bemerkung (*Remark*), die Version des Erreichbarkeitsagenten (*Version*) und eine eindeutige Nachrichtenkennung (*Id*) ergänzt. Diese Nachrichten können jetzt von anderen Systemen von der ActiveMQ gelesen und ausgewertet werden.

Benutzerschnittstelle und Erreichbarkeitsagent
Eine *Benutzerschnittstelle* und der *Erreichbarkeitsagent* können durch verschiedene JSON Nachrichten über die ActiveMQ miteinander kommunizieren. Zum Einen werten die *Benutzerschnittstellen* die vom *Erreichbarkeitsagenten* erzeugten Informationen aus, um sie dem Bewohner zur Anzeige zu bringen. Die *Benutzerschnittstelle* tritt in diesem Fall als *Aktor* auf, die Schnittstelle entspricht der im voran gegangenen Abschnitt beschriebenen. Zum Anderen können auch Informationen von der *Benutzerschnittstelle* an den *Erreichbarkeitsagenten* weitergeben werden.

Aktive Rückmeldungen des Bewohners werden im Format einer *ReachabilityInstruction* (vgl. Listing 2) an den *Erreichbarkeitsagenten* weiter gegeben. In dieser Nachricht wird sowohl die betroffene Gruppe (*Group*) als auch der vom Bewohner gewählte Erreichbarkeitszustand (*Status*) angegeben. Zusätzlich können eine Bemerkung (*Remark*) und die Nachrichtenkennung der *ReachabilityInformation*, auf welche sich dieses Feedback bezieht, angegeben werden.

Kontrollnachrichten, wie das Starten, Suspendieren und Beenden des *Erreichbarkeitsagenten* (vgl. Abschnitt 3.2.3), werden in Form einer *ConfigOrder* (vgl. Listing 3) über den globalen Kontrollstrom an den *Erreichbarkeitsagenten* geschickt. Diese Nachricht enthält die jeweils auszuführende Anweisung (*Order*) und als *Id* die betreffende Anwendung (In diesem Fall immer „*ReachabilityAgent*“).

3.2.3. Benutzerschnittstelle

Die Benutzerschnittstelle ist ein sehr wichtiger Teil dieses Projektes. Sie soll dem Bewohner die Möglichkeit geben, die im System vorhandene Einschätzung seiner Erreichbarkeit zu jedem Zeitpunkt einsehen zu können. Dies soll ihm das Gefühl geben, zu jedem Zeitpunkt die Kontrolle über das System zu haben. Ohne dieses Kontrollgefühl würde der Bewohner dem System misstrauen, was der Akzeptanz des Erreichbarkeitsagenten entgegenwirken würde.

Um dieses Gefühl weiter zu verbessern und eine Möglichkeit zu schaffen, sich auf die Vorlieben des Bewohners einstellen zu können, soll die Benutzerschnittstelle auch in der Lage sein, eine aktive Rückmeldung des Bewohners an den Erreichbarkeitsagenten weiterzuleiten. Dies gibt dem Bewohner das Gefühl aktiv auf seine Erreichbarkeit einwirken zu können und dem System die Möglichkeit seine Einschätzung zukünftig zu verbessern.

Neben diesen Fähigkeiten soll die Benutzerschnittstelle auch einfache Kontrollfunktionen, wie das vorübergehende Suspendieren des Erreichbarkeitsagenten, auf Wunsch des Bewohners ausführen können.

3.3. Implementierungsstand

Im Verlauf des ersten Masterprojektes konnte schon ein Großteil der beschriebenen Architektur realisiert werden. Dazu zählen verschiedene *SensorAdapter*, ein exemplarischer *Reasoner*, der *AgentWorker* und das Modell einer *Benutzerschnittstelle*.

Der *AgentWorker* realisiert vollständig die Verwaltung der *SensorAdapter* und des *Reasoners*. Er beherrscht die Weiterleitung der ermittelten Sensordaten an den *Reasoner* und die Kommunikation über die ActiveMQ. Für diese Kommunikation wurde eine Utlity-Klasse entwickelt, welche sich der, im Living Place weitestgehend verwendeten, LivingPlaceMessaging Bibliothek bedient. Alle Konfigurationen des Erreichbarkeitsagenten sind derzeit über eine zentrale Konfigurationsdatei einstellbar. Die einzelnen Aktivitäten der Komponenten können bei Bedarf auf der Konsole oder in einer separaten Logdatei ausgegeben werden. Die Anbindung der *SensorAdapter* und des *Reasoners* erfolgt über eine definierte Schnittstelle, welche die jeweiligen Implementationen erfüllen müssen. Sie können zur Zeit über die Konfigurationsdatei eingebunden und verwaltet werden.

Als Beispiele für die Sensoranbindung wurden *SensorAdapter* für drei verschiedene Sensoren implementiert. So konnte erfolgreich der Positionssensor des UbiSense Trackingsystems [EKV+11], der StressCompanion [Lin12] und, als Beispiel für einen nicht über die ActiveMQ kommunizierenden Sensor, ein Zeitsensor an das Framework angebunden werden. Da sich die Anbindung der Sensoren als sehr einfach herausstellte, wurde auf die Implementierung weiterer Sensoren vorerst verzichtet. Dies soll allerdings im nächsten Projektschritt sukzessive fortgesetzt werden (vgl. Ausblick, Abschnitt 4.1).

Ein exemplarischer Reasoner wurde in Form eines regelbasierten Systems implementiert. Verwendet wurde dabei die Jess Rule Engine[11] welche durch eine Regeldatei konfiguriert werden kann. Die tatsächliche Regelausprägung wurde bislang nicht realisiert, da hierzu

[11] Java-basierte Regelmaschine – http://www.jessrules.com/ – Zuletzt besucht: 26.08.2012

erst die relevanten Faktoren bestimmt werden müssen, aus welchen sich die Erreichbarkeitszustände ableiten lassen (hierauf wird näher in Abschnitt 4.1 und [Kan12b] eingegangen). Somit wird mit dieser prototypischen Implementation lediglich die Funktionsweise des Frameworks und des Nachrichtenaustauschs der einzelnen Komponenten gezeigt.

Als Modell einer Schnittstelle zum Benutzer wurde die in Abbildung 2 dargestellt Anwendung erzeugt. Hier werden dem Bewohner die aktuell auf der ActiveMQ verfügbaren Erreichbarkeitszustände angezeigt. Es gibt eine Spalte pro Personengruppe, welche jeweils über den Erreichbarkeitszuständen angezeigt wird. Darüber hinaus wird der für diese Gruppe aktuell geltende Erreichbarkeitszustand farblich hervor gehoben. Der Bewohner hat nun zum Beispiel die Möglichkeit bei einer bestimmten Personengruppe einen anderen Zustand anzuklicken, woraufhin diese Information an den Erreichbarkeitsagenten als aktives Feedback weitergeben wird.

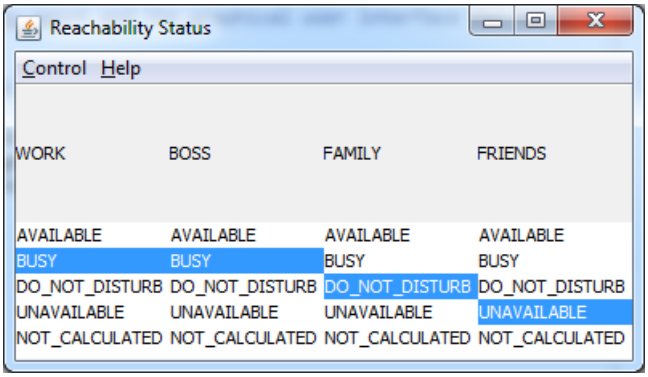

Abbildung 2: Beispielimplementation einer *Benutzerschnittstelle*

Neben dieser Anzeige- und Feedback-Funktionalität bietet diese *Benutzerschnittstelle* die Möglichkeit dem Erreichbarkeitsagenten Kontrollnachrichten entsprechend des Living Place Agent Life Cycles zu Übermittelt. Diese Nachrichten können über den Menüpunkt *Control* ausgelöst werden (vgl. Abbildung 3). Auf diese Weise kann der Erreichbarkeitsagent gestartet (Start), vorübergehend deaktiviert (Sleep) oder beendet (Stop) werden.

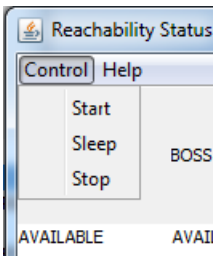

Abbildung 3: Kontrolloptionen der *Benutzerschnittstelle*

4. Zusammenfassung

Im Rahmen des ersten Masterprojektes zur Realisierung eines Erreichbarkeitsagenten konnte ein Framework erzeugt werden, mit welchem es möglich ist weiterführende Untersuchungen bei der Ermittlung relevanter Informationen zur Bestimmung der Erreichbarkeit eines Smart-Home Bewohners durchzuführen. Es stellt die Basis dar, um einen lauffähigen Prototypen in Umfeld des Living Place Hamburgs zu integrieren und dort mithilfe von Testpersonen zu evaluieren. Es wurde eine Architektur geschaffen, welche es dem Entwickler erlaubt einfach weitere Sensoren zu ergänzen und den Reasoner auszutauschen, um so verschiedene Reasoningverfahren gegeneinander zu testen. Dem Bewohner wurde die Möglichkeit gegeben sich durch die *Benutzerschnittstelle* über seine aktuell eingeschätzte Erreichbarkeit zu informieren und dem Erreichbarkeitsagenten ein aktives Feedback zu geben.

4.1. Ausblick

Mit den Fortschritten dieses Masterprojektes wurden die Grundlagen für das weitere Vorgehen bei der Entwicklung eines Erreichbarkeitsagenten im Smart-Home Umfeld gelegt. Im weiteren Verlauf sollen viele verschiedene Sensoren nach dem so genannten *Wizard of Oz* Verfahren[12] simuliert werden, da zum aktuellen Zeitpunkt noch nicht erkennbar ist, welche Faktoren es tatsächlich möglich machen, die Erreichbarkeit des Bewohners zu bestimmen. Dabei werden bei dem *Wizart of Oz* Verfahren viele Sensoren manuell auf Basis eines Zeitstrahls simuliert. Dies hat den Vorteil, dass nicht fehlende Sensoren teuer angeschafft und zeitaufwendig integriert werden müssen. Durch die Architektur des Erreichbarkeitsagenten ist es sehr einfach simple *SensorAdapter* zu implementieren, welche dann die simulierte Daten von der ActiveMQ auslesen und verarbeiten. Auf dieser Grundlage sollen dann durch verschiedene Verfahren, welche aus anderen Projekten übernommen werden (ausführlich beschrieben in [Kan12b]), die relevanten Faktoren für die Bestimmung der Erreichbarkeit ermittelt werden. Sobald diese Faktoren determiniert sind, sollen verschiedene lernende Verfahren verglichen werden, um das Verfahren zu ermitteln, welches die besten Eigenschaften in Bezug auf Adaption und Korrektheit gegenüber dem Bewohner aufweist.

In der aktuellen Version des Agenten werden die ermittelten Werte nicht gespeichert, was es im Falle eines aktiven Feedbacks schwer macht aus der Fehleinschätzung zu lernen. Dies wird im weiteren Verlauf geändert, damit aus dem Feedback Erfahrungen für die Zukunft bezüglich der Vorlieben des Bewohners gewonnen werden können. Nur wenn sich das System bestmöglich und schnell an den Bewohner anpasst, wird es Akzeptanz und Anwendung finden. Das System soll im Anschluss an die beschriebenen Ziele nach Möglichkeit durch Testpersonen im Rahmen des Living Place Hamburg evaluiert werden.

[12] Technik zur Simulation von Sensorinputs (vgl. [FHA+05])

Literatur

[Bor12] BORNEMANN, Sven B.: *Mobile Türklingel für Smart Homes*. Hamburg, Germany, HAW Hamburg, AW1 Ausarbeitung, 2012. – http://users.informatik.haw-hamburg.de/~ubicomp/projekte/master11-12-aw1/bornemann/bericht.pdf – Zugriff: August, 2012

[EKV+11] ELLENBERG, Jens ; KARSTAEDT, Bastian ; VOSKUHL, Sören ; LUCK, Kai von ; WENDHOLT, Birgit: An environment for context-aware applications in smart homes. In: *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. Guimarães, Portugal, Sept. 21–23, 2011. – http://ipin2011.dsi.uminho.pt/PDFs/Poster/50_Poster.pdf

[FHA+05] FOGARTY, James ; HUDSON, Scott E. ; ATKESON, Christopher G. ; AVRAHAMI, Daniel ; FORLIZZI, Jodi ; KIESLER, Sara ; LEE, Johnny C. ; YANG, Jie: Predicting human interruptibility with sensors. In: *ACM Transactions on Computer-Human Interaction (TOCHI)* 12 (2005), March, Nr. 1, S. 119–146. – ISSN 1073–0516. – http://www.interruptions.net/literature/Fogarty-TOCHI05.pdf – Zugriff: August, 2012

[Har11] HARDENACK, Frank: – *Das intelligente Bett – Sensorbasierte Detektion von Schlafphasen*. Hamburg, Germany, HAW Hamburg, Masterarbeit, 2011. – http://users.informatik.haw-hamburg.de/~ubicomp/arbeiten/master/hardenack.pdf

[Kü12] KÜHN, Philipp: *Ein interaktiver Couchtisch*. Hamburg, Germany, HAW Hamburg, Ausarbeitung Anwendungen 1, 2012. – http://users.informatik.haw-hamburg.de/~ubicomp/projekte/master11-12-aw1/kuehn/bericht.pdf – Zugriff: August, 2012

[Kan12a] KANTAK, Malte: *Erreichbarkeit in Smart-Homes*. Hamburg, Germany, HAW Hamburg, Ausarbeitung Anwendungen 1, 2012. – http://users.informatik.haw-hamburg.de/~ubicomp/projekte/master11-12-aw1/kantak/bericht.pdf – Zugriff: August, 2012

[Kan12b] KANTAK, Malte: *Erreichbarkeit in Smart-Homes – Related Work*. Hamburg, Germany, HAW Hamburg, Ausarbeitung Anwendungen 2, 2012

[Lin12] LINDEMANN, Benjamin: *Stress am IT-Arbeitsplatz – Projektbericht*. Hamburg, Germany, HAW Hamburg, Masterprojekt 1, 2012

[LKG+10] LUCK, P. D. K. ; KLEMKE, P. D. G. ; GREGOR, S. ; RAHIMI, Mohammad A. ; VOGT, Matthias: Living Place Hamburg - A place for concepts of IT based modern living / HAW Hamburg. Hamburg, Germany, Mai 2010. – Forschungsbericht. – http://livingplace.informatik.haw-hamburg.de/content/LivingPlaceHamburg_en.pdf – Zugriff: August, 2012

[UMI] *Multitasking and Task Switching*. http://www.umich.edu/~bcalab/multitasking.html. – Zugriff: August, 2012

A. Nachrichtenformate

Im Folgenden werden Beispiele für JSON Nachrichten aufgelistet, welche bei der Kommunikation des Erreichbarkeitsagenten verwendet werden. Es werden jeweils nur die im Zusammenhang mit dem Erreichbarkeitsagenten relevanten Attribute dargestellt. Die Nachrichten sind im JSON[13] codiert.

Listing 1: ReachabilityInformation

```
{
  "GroupId" : "ReachabilityAgentWorker_1080427099",
  "Group"   : "FAMILY",
  "Status"  : "AVAILABLE",
  "Remark"  : "",
  "Version" : "2012.2",
  "Id"      : "ReachabilityAgentWorker_25480694",
  [. . .]
}
```

Listing 2: ReachabilityInstruction

```
{
  "Group"   : "FAMILY",
  "Status"  : "DO_NOT_DISTURB",
  "Remark"  : "",
  "Version" : "2012.1",
  "Id"      : "ReachabilityUI_169594624",
  "RefId"   : "ReachabilityAgentWorker_1080427099",
  [. . .]
}
```

Listing 3: ConfigOrder

```
{
  "order"   : "sleep",
  "Version" : "2012.1",
  "Id"      : "ReachabilityAgent",
  [. . .]
}
```

[13] **J**ava**S**cript **O**bject **N**otation

Listing 4: UbisenseTrackingMessage

```
{
  "UbisenseTagId" : "123-234-456-213",
  "Unit"          : "meter",
  "NewPosition"   : {
    "X" : 5.677,
    "Y" : 4.196,
    "Z" : 0.0
  },
  "Version" : "0.6",
  [. . .]
}
```

Listing 5: StressCompanionInformation

```
{
  "state"   : "HIGH",
  "Version" : "0.2",
  "Id"      : "StressCompanion_31239932",
  [. . .]
}
```