Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Ausarbeitung Projekt 2
Wintersemester 2011/12

Andreas Basener

Weiterentwicklung des Skriptsimulators

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Inhaltsverzeichnis

1 Einführung

Das Livingplaceprojekt der Hochschule für Angewandte Wissenschaften in Hamburg (von Luck u. a., 2010) dient der Erforschung, welche Möglichkeiten eine intelligente Wohnung einem Bewohner bieten kann, und wie die Interaktion zwischen Wohnung und Nutzer sinnvoll gestaltet werden kann.
Im Laufe der Zeit sind bereits diverse Projekte im Livingplace realisiert worden oder werden momentan erarbeitet. Z.B. eine Lichtsteuerung, ein Multitouchtisch, ein Bett, das die Schlafphasen des Bewohners erfassen kann, ein Sensornetzwerk und vieles mehr.

Abbildung 1: Rendermodell des Livingplace

Durch die zahlreichen Projekte wird eine große Menge an Daten erzeugt, die durch die Interaktion des Bewohners mit der Wohnung entstehen. Diese Daten können u.a. dazu verwendet werden, die Aktivitäten des Bewohners zu ermitteln und die Wohnung entsprechend darauf reagieren zu lassen.
Die Daten stehen aber nicht immer in der gewünschten Qualität zur Verfügung. Das ist zu allererst im experimentellen Charakter des Livingplaceprojektes begründet. Für eine ausreichende Datenbasis müsste die Wohnung mehrere Monate in einem realen Szenario bewohnt werden.
Da sich in der Wohnung aber ständig Personen aufhalten, um Projekte und Experimente durchzuführen, ist ein normales Wohnen nicht möglich, zumindest nicht über einen größeren

Zeitraum.
Weiterhin kann es durch ausgefallene oder fehlerhaft arbeitende Hard- und Software vorkommen, dass die erfassten Daten unvollständig oder fehlerhaft sind. Für viele Experimente werden zur Reproduzierbarkeit exakt die gleichen Daten benötigt, oder es werden Daten benötigt, die in der real existierenden Wohnung nicht verfügbar sind.
Hierfür ist der Einsatz eines Simulators sinnvoll. Mithilfe eines Simulators können die gewünschten Daten zum benötigten Zeitpunkt erzeugt werden. Im Projekt 1 (Basener, 2011b) ist bereits das Grundgerüst für einen Skriptsimulator entstanden. Im Projekt 2 wurde dieser Skriptsimulator weiterentwickelt. Dabei sind die Erkenntnisse, die ich in den Veranstaltungen Anwendungen 1 und 2 (Basener, 2011a)(Basener, 2011c) gewonnen habe, eingeflossen. Zudem habe ich die im Projekt 1 angesprochenen Verbesserungen zum größten Teil vorgenommen.

Die nachfolgenden Kapitel beschreiben die Weiterentwicklung des Skriptsimulators und in welchem Zusammenhang er zum Livingplaceprojekt steht. Weiterhin werde ich die Leistungsfähigkeit demonstrieren und den Praxiseinsatz des Skriptsimulators darstellen.

1.1 Ziel des Projektes

Ziel der Veranstaltungen Projekt 1 und 2 war, dass mit dem Skriptsimulator ein Werkzeug zur Verfügung steht, mit dem Daten erzeugt werden können, wie sie durch die Interaktion des Bewohners mit der Wohnung entstehen. Mit dem Skriptsimulator sollen sich zeitlich gesteuerte Skripte erstellen lassen, die aus einzelnen Einträgen bestehen. Jeder dieser Einträge beschreibt eine Aktion, die Daten erzeugt.
Ein einzelner Eintrag kann dabei sowohl ein einzelnes Datum erzeugen, wie z. B. das Öffnen einer Schranktür, oder aber eine ganze Reihe zusammenhängender Daten, wie z. B. die Positionsdaten beim Durchqueren eines Raumes.
Für die Bedienung des Skriptsimulators ist eine Benutzeroberfläche notwendig. Über diese Benutzeroberfläche soll der Benutzer ohne großen Aufwand Skripte erstellen, bearbeiten und abspielen können. Gleichzeitig soll die Benutzeroberfläche alle nötigen Informationen übersichtlich darstellen können.
Außerdem sollen die erzeugten Daten des Skriptsimulators parallel zu den real erzeugten Daten aus dem Livingplace verwendet werden können.

2 Hauptteil

2.1 Weiterentwicklung des Skriptsimulators

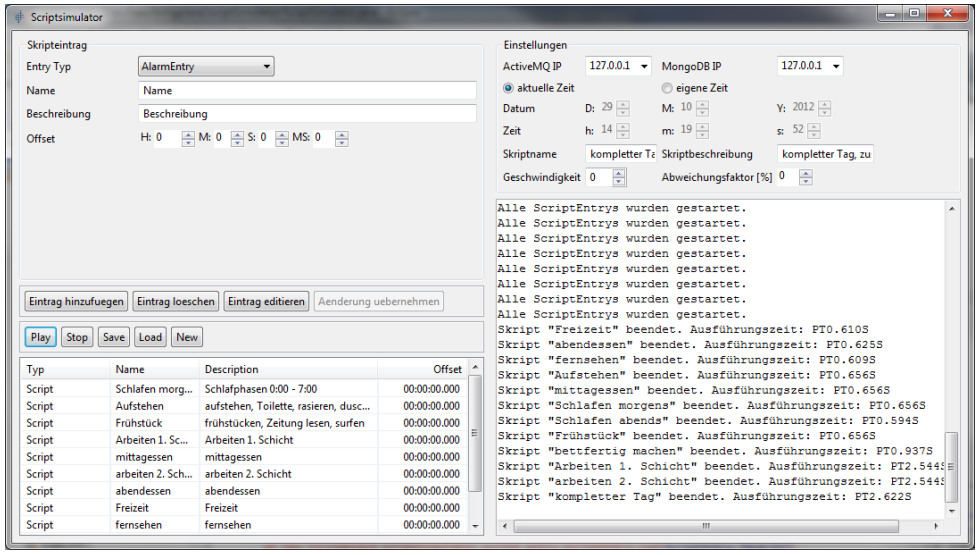

Abbildung 2: Skriptsimulator GUI

Im Projekt 1 habe ich die Grundlagen für den Skriptsimulator geschaffen. Dazu zählen die Architektur, fundamentale Funktionen des Simulators, sowie die Anbindung an die Infrastruktur des Livingplaces. Im Projekt 2 baue ich auf dieser Vorlage auf und habe den Skriptsimulator soweit fertiggestellt, wie ich ihn für meine Masterarbeit benötige.
In den folgenden Kapiteln sind die Änderungen und Erweiterungen am Skriptsimulator aufgeführt. Weiterhin wird erklärt, wie sich der Skriptsimulator in das Livingplace integriert und mit den anderen Projekten zusammen nutzen lässt.
Nachfolgend werden die wichtigsten Punkte beschrieben.

2.1.1 Veränderungen

Im Projekt 2 habe ich diverse Änderungen am Skriptsimulator vorgenommen. Einige dieser Änderungen sind Verbesserungen der vorhandenen Funktionen, andere wiederum sind neue Funktionen, die entweder für die spätere Masterarbeit benötigt werden oder aber die Benutzung des Skriptsimulators komfortabler werden lassen.

Die Veränderung, die als Erstes auffällt, ist die Umgestaltung der Benutzeroberfläche. Neue Funktionen benötigen ausreichend Platz auf der Oberfläche, um vernünftig bedient werden zu können. Gleichzeitig wird Platz benötigt, um Informationen über den Zustand des Skriptes und Hinweise an den Benutzer anzeigen zu können.
Daher habe ich mich entschieden, die einzelnen Elemente neu zu ordnen, wie in Abbildung 2 gut zu sehen ist. Die Oberfläche ist zweispaltig aufgebaut. In der linken Spalte sind alle Elemente enthalten, die für die Verwaltung und Manipulation der einzelnen Skripteinträge notwendig sind. In der rechten Spalte sind die Elemente untergebracht, die Informationen über das Skript beinhalten. U. a. kann hier das Datum eingestellt werden, zu dem das Skript in der Simulation starten soll. Zudem ist ein Textfeld hinzugekommen, in dem alle wichtigen Informationen angezeigt werden, z.B. wenn das Skript ausgeführt wird.
Dadurch ist die Benutzeroberfläche, im Gegensatz zur bisherigen Oberfläche (s. Basener (2011b)), sinnvoller und leichter zu verwenden.

Eine weitere Änderung wurde an der Laden- und Speicherfunktion vorgenommen. Bisher wurde für das Laden und Speichern der Skripte eine einfache Objektserialisierung, die Java zur Verfügung stellt, verwendet. Das hatte den Vorteil, dass die Methoden für das Laden und Speichern der Skripte relativ einfach implementiert werden konnten. Der Nachteil war, dass lediglich über das Schlüsselwort **transient** ganze Klassenfelder von der Serialisierung ausgeschlossen werden können. Oft ist aber eine feinere Kontrolle darüber nötig, was serialisiert werden soll und was nicht. So ist es zwar z.B. wichtig, die Uhrzeit abzuspeichern, das serialisierte Objekt aus der Jodatime-Klasse Period[1] enthält aber auch viele Informationen, die eigentlich nicht wichtig sind und die gespeicherte Datei unnötig groß werden lässt.
Daher wird ein Speicher- und Lademechanismus benötigt, der flexibler zu verwenden ist. Das wird durch den Einsatz der GSON-Bibliothek[2], die von Google bereitgestellt wird, erreicht. Mit dieser Bibliothek ist es möglich, genau festzulegen, welche Informationen gespeichert werden sollen und wie diese Informationen im JSON-Format[3] dargestellt werden sollen. Dazu wird für jeden Skripteintrag ein Converter geschrieben, der dafür sorgt, dass die Objekte sinnvoll gespeichert und geladen werden können. Da die GSON-Bibliothek auch dazu verwendet wird, die Nachrichten für das ActiveMQ zu erzeugen, liegt eine weitere Verwendung hierfür nahe.
Die erzeugten JSON-Strings durch Verwendung der GSON-Bibliothek haben auch den Vorteil, dass die gespeicherten Skripte in einem leicht für Mensch und Maschine lesbaren Format vorliegen. Dadurch können sie ohne viel Aufwand außerhalb des Skriptsimulators gelesen und verändert werden, bzw. von anderen Programmen verwendet werden.

[1] http://joda-time.sourceforge.net/key_period.html
[2] http://code.google.com/p/google-gson/
[3] http://www.json.org/

2.1.2 Erweiterungen

Im Projekt 2 wurde der Skriptsimulator hauptsächlich um ein paar Skripteintragstypen erweitert. Bisher gab es lediglich ein paar Typen, die dazu dienten, die Funktionsweise des Skriptsimulators zu testen.
Nun wurden im Projekt 2 auch Eintragstypen für bestehende Projekte im Livingplace hinzugefügt. Dazu gehört das Sensornetz von Alexander Pautz (Pautz, 2012), das Indoorpositionssystem Ubisense[4], welches u. a. von Bastian Karstaedt aufgebaut wurde, und das intelligente Bett von Frank Hardenack (Hardenack, 2011).
Dadurch ist es jetzt möglich, die wichtigsten Projekte, die Daten erzeugen, im Livingplace zu simulieren. Weitere Projekte können dem Skriptsimulator jederzeit hinzugefügt werden.

Eine weitere Neuerung des Skriptsimulators ist die Möglichkeit, eine zufällige Ungenauigkeit für die Skripteinträge zu erzeugen. In der Benutzeroberfläche kann angegeben werden, wie stark ein Zufallsfaktor die Skripteinträge beeinflussen soll.
Der Skriptsimulator errechnet intern einen Zufallswert, der zwischen -1 und 1 liegt und eine Normalverteilung um den Wert 0 aufweist.
Dazu wird auf die Methode nextGaussian() der Javaklasse Random[5] zurückgegriffen. Diese Methode liefert normalverteilte Zufallszahlen mit dem Mittelwert 0 und einer Standardabweichung von 1. Um die gewünschten Zufallszahlen zu erhalten, die alle zwischen -1 und 1 liegen, werden Zahlen, die größer als 4 oder kleiner als -4 sind, einfach auf den Wert 0 gesetzt. Alle anderen Zahlen werden durch 4 geteilt und mit dem oben erwähnten Wert aus der Benutzeroberfläche multipliziert. Der Wert 4 ist willkürlich ausgewählt und berücksichtigt lediglich, dass sich ab diesen Werten die Gaußkurve bereits stark an die X-Achse annähert. Die weitere Berechnung wird dadurch erheblich vereinfacht.
Das ergibt zwar keine mathematisch exakte Normalverteilung, reicht aber für den Skriptsimulators aus, um eine nichtdeterministische Ungenauigkeit zu erzeugen.

Für die Skripteinträge kann der Zufallswert unterschiedlich verwendet werden. Bei allen Skripteinträgen lässt sich die Startzeit damit manipulieren. Somit können schwankende Startzeiten für die simulierten Aktionen erzeugt werden.
Für einige Eintragstypen kann der Zufallswert verwendet werden, um die Sensordaten abweichen zu lassen. So können z. B. bei den Ubisenseeinträgen die Positionskoordinaten oder im Sensornetzwerk die Sensorwerte verfälscht werden.

Bei manchen Skripteinträgen ist eine Abweichung der Daten nicht möglich oder sinnvoll. Die Türklingel ist z. B. kaum geeignet für eine Abweichung. Hier könnte höchstens per Zufall entschieden werden, ob die Klingel tatsächlich klingelt, wenn sie gedrückt wird, oder ob sie

[4] http://www.ubisense.net/en/
[5] http://docs.oracle.com/javase/1.4.2/docs/api/java/util/Random.html#nextGaussian()

hin und wieder klingelt, obwohl sie nicht gedrückt wurde. Obwohl das technisch durchaus machbar ist, ist dieses Szenario für das Livingplace kaum sinnvoll.

2.2 Leistungsfähigkeit des Skriptsimulators

Um zu testen, wie leistungsfähig der Skriptsimulator ist, wurde ein kompletter Tag im Simulator nachgestellt. Dazu wurde der Tag von 00:00 Uhr bis 23:59 Uhr in 11 einzelne Abschnitte unterteilt, wie in Tabelle 1 zu sehen. Jeder Abschnitt ist ein eigenständiges Skript, das einen bestimmten Tagesabschnitt beschreibt. Die einzelnen Skripte werden dann zu einem übergeordneten Skript zusammengefasst.

Starzeit	Name
00:00	schlafen morgens
07:00	aufstehen
07:20	frühstücken
08:00	arbeiten 1. Schicht
12:00	Mittagspause
13:00	arbeiten 2. Schicht
17:00	abendessen
17:30	Hobby, Freizeit
20:00	fernsehen
22:00	bettfertig machen
22:10	schlafen abends

Tabelle 1: die einzelnen Abschnitte eines Tagesablaufes

Insgesamt umfasst das Skript 197 Einträge aus unterschiedlichen Eintragstypen, die 374 einzelne Events erzeugen.

Für die Erfassung der Leistungsfähigkeit des Skriptsimulators wurde der Skriptsimulator so eingestellt, dass alle Einträge ohne Pausen sofort abgespielt werden. Wenn ein Eintrag mehrere Events erzeugt, werden auch diese ohne Pause erzeugt. Dadurch entstehen innerhalb kürzester Zeit sehr viele Events, die verarbeitet werden müssen.

Der Test wurde auf einem mittelklassigen Laptop (Stand 2012) durchgeführt. Die CPU ist ein AMD A8-3500M, der Arbeitsspeicher ist 8GB groß und als Betriebssystem kam Windows 7 64 Bit zum Einsatz. Beim Abspielen des Skriptes wurde die Zeit ermittelt, die benötigt wird, alle Einträge zu starten und alle Events zu erzeugen. Es wurden insgesamt 10 Durchgänge ausgeführt.

Durchgang #	Zeit [s]
1	2,662
2	3,370
3	3,073
4	3,151
5	3,183
6	3,386
7	3,447
8	3,338
9	3,230
10	3,798
arithmetisches Mittel	3,204

Tabelle 2: Ausführungszeiten des Skriptes

Im Durchschnitt dauerte das 3,204 Sekunden. Die längste Zeit beträgt 3,447 Sekunden, die kürzeste 2,662 Sekunden. In Tabelle 2 sind alle Zeiten für die verschiedenen Durchgänge aufgeführt.

Die ermittelten Zeiten zeigen, dass der Skriptsimulator in der Lage, ist ausreichend viele Events innerhalb kurzer Zeit zu erzeugen. Der Testaufbau stellt ein Extrem dar, das so kaum in der Praxis auftauchen wird. Ein typisches Skript wird lediglich alle paar Sekunden neue Events erzeugen.

2.3 Der Scriptsimulator im Zusammenhang des Livingplaces

Der Skriptsimulator ist ein Teilprojekt innerhalb des Livingplaces der Hochschule für Angewandte Wissenschaften in Hamburg. Er existiert und funktioniert parallel zu den anderen Teilprojekten und greift, wie diese, auf gemeinsame Ressourcen zu.
Das zentrale Element der Livingplaceinfrastruktur ist das ActiveMQ-Nachrichtensystem[6]. Über das ActiveMQ können die verschiedenen Projekte untereinander Nachrichten versenden. Dazu stehen ein Warteschlangen- und ein Publish-Subscribe-Mechanismus zur Verfügung. Zusätzlich wurde mit dem ActiveMQWrapper von Kjell Otto und Sören Voskuhl ein Push-Pull-Mechanismus implementiert (Otto und Voskuhl, 2010) (Otto und Voskuhl, 2011), mit dem Daten angefordert werden können.

[6] http://activemq.apache.org/

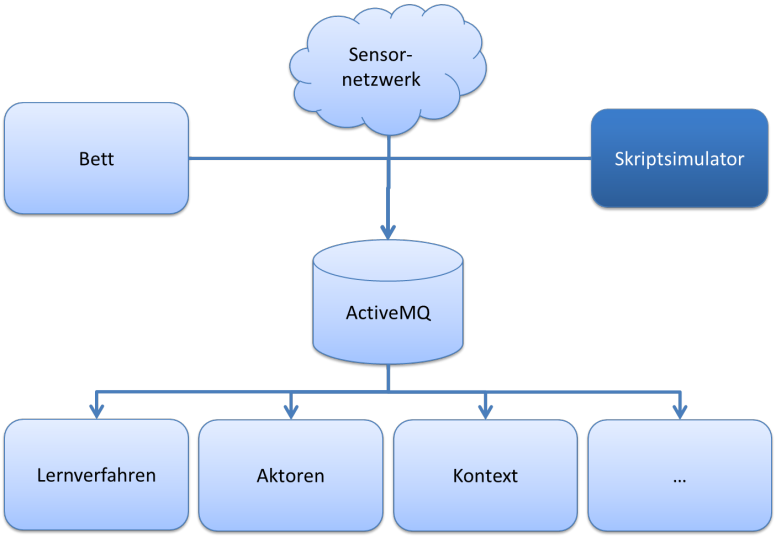

Abbildung 3: Vereinfachter Nachrichtenfluss im Livingplace

Dadurch ist es möglich, dass Projekte, die Daten erzeugen und zur Verfügung stellen, sich beim ActiveMQ anmelden und so ihre Daten veröffentlichen können. Andere Projekte, die sich für diese Daten interessieren, können nun die Daten einfach über das ActiveMQ abholen. Dazu müssen sich die Projekte nur darauf einigen, wie der jeweilige Nachrichtenstrom heißt.

Der Vorteil des ActiveMQs ist, dass sowohl real erzeugte, als auch durch den Skriptsimulator erzeugte Daten simultan verwendet werden können.

Ein weiterer Vorteil der Livingplaceinfrastruktur ist der Einsatz der MongoDB[7]. Mit diesem Datenbanksystem ist es problemlos möglich, Datenströme im ActiveMQ aufzuzeichnen und für eine spätere Verwendung oder Analyse zu speichern.

Durch die Kombination aus ActiveMQ und MongoDB ist es für den Skriptsimulator möglich, die erzeugten Daten anderen Projekten zur Verfügung zu stellen und die erzeugten Nachrichten persistent zu speichern.
Für jeden Eintragstyp des Skriptsimulators wird ein eigenes Topic für die Nachrichten erzeugt. Dadurch können die verschiedenen Nachrichtenarten leicht unterschieden werden und weitere Anwendungen brauchen sich nur für die Arten von Nachrichten zu registrieren, die sie auch interessieren.

[7] http://www.mongodb.org/

3 Fazit

3.1 Zusammenfassung

Der Skriptsimulator wurde entwickelt, um eine Reihe von Problemen zu lösen, die bei der Projektentwicklung im Livingplace entstehen. Dabei sind die nun vorliegende Architektur und Funktionen des Skriptsimulators entstanden.

Der Hauptzweck des Skriptsimulators ist, Daten zu erzeugen, die im Livingplace von diversen Projekten, wie das Sensornetzwerk oder das intelligente Bett, erzeugt werden. Diese Fähigkeit wird dadurch gelöst, das ein Handlungsablauf in einzelne Schritte aufgebrochen wird und diese einzelnen Schritte als separate Einträge zu einem Skript zusammengefügt werden. So wird z.B. die Aktion *Kaffekochen* (vereinfacht) in die einzelnen Schritte *Kaffeepulver auffüllen*, *Wasser auffüllen*, *Kaffeemaschine einschalten* und fertigen *Kaffee abfüllen* aufgeteilt. Die einzelnen Schritte erzeugen dann wiederum Daten, die von anderen Programmen beobachtet werden können.

Neben dem einfachen Erstellen und Abspielen von Skripten bietet der Skriptsimulator eine Reihe an weiteren nützlichen Funktionen.
So ist es mit dem Skriptsimulator möglich, beliebige Zeiträume und Tageszeiten zu den Zeitpunkten zu simulieren, wann sie gebraucht werden. Damit können z.B. Aktionen, die in der Nacht stattfinden, während des Tages simuliert werden. Weiterhin können Zeiträume mit einer erhöhten Geschwindigkeit abgespielt werden. Dadurch können lange Zeiträume, wie Wochen und Monate, innerhalb kürzester Zeit simuliert werden und die Ergebnisse schneller ausgewertet werden.
Ein weiteres Merkmal des Skriptsimulators ist die Fähigkeit, Daten reproduzierbar zu erzeugen. Dadurch können Experimente genauer und zuverlässiger durchgeführt und wiederholt werden.
Auf Wunsch kann der Skriptsimulator die Daten mit einem zufälligen Fehler versehen. Das kann nützlich sein, wenn man Verfahren darauf testen möchte, wie sie mit ungenauen Datenquellen umgehen können.

Durch die Verwendung von ActiveMQ für das Senden und Empfangen von Nachrichten kann der Skriptsimulator parallel zu den weiteren Projekten im Livingplace eingesetzt werden. Dadurch ist es möglich, die simulierten Daten mit real erzeugten Daten zu kombinieren.

Der Skriptsimulator lässt sich beliebig erweitern. Neue Projekte, die Daten erzeugen, können ohne großen Aufwand für den Skriptsimulator implementiert werden. Dadurch ist gewährleistet, dass der Skriptsimulator auch für zukünftige Szenarien einsetzbar bleibt.
Bisher ist der Skriptsimulator darauf ausgelegt, das Livingplace zu simulieren und Daten zu erzeugen. Eine denkbare Erweiterung des Skriptsimulators ist, nicht nur Daten zu erzeugen,

sondern auch direkt Aktoren im Livingplace anzusteuern. Die Skripte könnten dann so geschrieben werden, dass sie wie Theateranweisungen Schritt für Schritt einzelne Aktionen auslösen. Die Basis für diese Funktion ist im Skriptsimulator bereits vorhanden. Es müssen nur geeignete Skripteinträge implementiert werden, die die Steuerbefehle der einzelnen Aktoren erzeugen.

3.2 Ausblick auf Masterarbeit

Mit dem Skriptsimulator steht mir nun ein leistungsfähiges Werkzeug zur Verfügung, mit dessen Hilfe ich die nötigen Daten erzeugen kann, die ich in meiner Masterarbeit benötige.
In meiner Masterarbeit möchte ich für das Livingplace ein Lernverfahren entwickeln, das aus erzeugten Daten die Aktivitäten des Bewohners ermitteln kann. Ein ähnliches Verfahren wurde u.a. im CASAS-Projekt der Washington State University entwickelt (Rashidi u. a., 2011). Damit so ein Verfahren überhaupt funktionieren kann, benötigt es viele Daten.

Im Kapitel 1 habe ich bereits aufgeführt, warum die notwendigen Daten im Livingplace sehr schwer zu erzeugen sind. Daher werde ich die notwendige Datenbasis mit dem Skriptsimulator erzeugen. Dadurch verspreche ich mir eine bessere Kontrolle über die Daten, sodass ich mich besser auf die eigentliche Entwicklung eines Lernverfahrens konzentrieren kann.

Literatur

[Basener 2011a] BASENER, Andreas: Ausarbeitung AW1 Drahtlose Sensornetzwerke im Kontext Ambient Assisted Living / HAW Hamburg. URL http://users.informatik.haw-hamburg.de/~ubicomp/projekte/master10-11-aw1/basener/bericht.pdf, Februar 2011. – Forschungsbericht

[Basener 2011b] BASENER, Andreas: Ausarbeitung AW2 Entwicklung eines Skriptsimulators für das Living Place / HAW Hamburg. URL http://users.informatik.haw-hamburg.de/~ubicomp/projekte/master2011-proj1/basener.pdf, August 2011. – Forschungsbericht

[Basener 2011c] BASENER, Andreas: Ausarbeitung AW2 Erlernen und Erkennen von Verhaltensmustern in Sensordaten / HAW Hamburg. URL http://users.informatik.haw-hamburg.de/~ubicomp/projekte/master2011-aw2/basener/bericht.pdf, August 2011. – Forschungsbericht

[Hardenack 2011] HARDENACK, Frank: *Das intelligente Bett - Sensorbasierte Detektion von Schlafphasen*. Hamburg, Hochschule für Angewandte Wissenschaften Hamburg, Diplomarbeit, 2011. – URL http://users.informatik.haw-hamburg.de/~ubicomp/arbeiten/master/hardenack.pdf

[von Luck u. a. 2010] LUCK, Prof. Dr. K. von ; KLEMKE, Prof. Dr. G. ; GREGOR, Sebastian ; RAHIMI, Mohammad A. ; VOGT, Matthias: Living Place Hamburg – A place for concepts of IT based modern living / Hamburg University of Applied Sciences. URL http://livingplace.informatik.haw-hamburg.de/content/LivingPlaceHamburg_en.pdf, Mai 2010. – Forschungsbericht

[Otto und Voskuhl 2010] OTTO, Kjell ; VOSKUHL, Sören: Entwicklung einer Architektur für den Living Place Hamburg / HAW Hamburg. URL http://users.informatik.haw-hamburg.de/~ubicomp/projekte/master2010-proj1/otto_voskuhl.pdf, 2010. – Forschungsbericht

[Otto und Voskuhl 2011] OTTO, Kjell ; VOSKUHL, Sören: Weiterentwicklung der Architektur des Living Place Hamburg / HAW Hamburg. URL http://users.informatik.haw-hamburg.de/~ubicomp/projekte/master10-11-proj2/otto-voskuhl.pdf, 2011. – Forschungsbericht

[Pautz 2012] PAUTZ, Alexander: *Kabellose, stromsparende Sensornetzwerke im Bereich Ambient Intelligence*. Hamburg, Hochschule für Angewandte Wissenschaften Hamburg, Diplomarbeit, 2012. – URL http://users.informatik.haw-hamburg.de/~ubicomp/arbeiten/master/pautz.pdf

[Rashidi u. a. 2011] RASHIDI, Parisa ; COOK, Diane J. ; HOLDER, Lawrence B. ; SCHMITTER-EDGECOMBE, Maureen: Discovering Activities to Recognize and Track in a Smart Environment. In: *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING* (2011). – URL http://eecs.wsu.edu/~cook/pubs/tkde10.pdf

[Voskuhl 2011] VOSKUHL, Sören: Seminarausarbeitung Entwicklung einer Architektur für Context Aware Systeme / HAW Hamburg. URL http://users.informatik.haw-hamburg.de/~ubicomp/projekte/master10-11-seminar/voskuhl/bericht.pdf, Februar 2011. – Forschungsbericht