Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Ausarbeitung Anwendungen 1

WiSe2011/12

Florian Johannßen

Cloud Robotics anhand des humanoiden Roboters NAO

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Florian Johannßen

Florian.Johannssen@haw-hamburg.de

Thema

Cloud Robotics anhand des humanoiden Roboters NAO

Stichworte

Cloud Robotics, Cloud Computing, NAO, NAOqi

Kurzzusammenfassung

Diese Ausarbeit zeigt sowohl einen Einblick des Themenbereichs *Cloud Robotics*, in welchem ich mich während meines Masterstudiums bewegen werde, als auch eine konkrete Vision, die ich verfolge. Dazu wird auf die Verknüpfung der Themen Robotik und Cloud Computing eingegangen. Diese Arbeit führt die Möglichkeiten, Vorteile und Risiken auf, die den Themenbereich *Cloud Robotics* auszeichnet und wie man heutzutage in der Lage ist, als Software-Entwickler High-Level-Applikationen für Roboter zu entwickeln.

Florian Johannßen

Title of the paper

Cloud Robotics on the basis of the humanoid robot NAO

Keywords

Cloud Robotic, Cloud Computing, NAO, NAOqi

Abstract

This paper introduces the subject area *Cloud Robotics*, which will be the main working part of my master study. It shows the options, advantages and risks of the subject and demonstrates how software developers are able to code high level applications for robots.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einführung

1.1 Motivation

Die Arbeit beschäftigt sich damit, wie man die Themen Robotik und Cloud Computing miteinander in Verbindung bringen kann. Immer mehr Forschungsgruppen und Firmen beteiligen sich nach (Kuffner 2011) der Entwicklung von Robotern, die in unserem alltäglichen Leben eingesetzt werden können. Hierbei ist besonders zu beobachten, dass Firmen wie Aldebaran-Robotics in der Lage sind, programmierbare Roboter auszuliefern, bei denen bereits die spezifischen Aufgaben, wie z. B. Gesichtserkennung, Wegplanung und Spracherkennung hardwarenah und effizient gelöst sind. Dies bietet Software-Entwicklern die Möglichkeit das Verhalten eines Roboters auf einer hohen Abstraktionsebene zu implementieren. Zusätzlich ist die Programmierung von Robotern durch die drahtlose Kommunikation ebenfalls attraktiver geworden, da sie wesentlich sicherer, schneller und universell einsetzbarer geworden ist. Viele Roboter verfügen daher über eine IEEE 802.11 standardisierte WLAN Schnittstelle, was eine einfache und bidirektionale Übertragung zwischen Robotern und Servern ermöglicht. Da Roboter nur geringe Hardwareressourcen besitzen, kann WIFI dazu verwendet werden, rechenintensive Aufgaben auf leistungsstarke Server auszulagern. Des Weiteren ergibt sich dadurch die Möglichkeit, dass sich Roboter untereinander verständigen, koordinieren und ihre Lernmechanismen durch geteiltes Wissensmanagement verbessern.

Bei diesen Gedanken ist der Begriff des Cloud Computing nicht weit entfernt. Dieses Konzept beschreibt den Ansatz IT-Infrastrukturen, Software oder Plattformen über das Internet anzubieten. Aus der Verknüpfung des Cloud Computing mit Robotik resultiert der Begriff *Cloud Robotics*. Im Folgenden wird gezeigt, welche Vorteile und Möglichkeiten sich daraus für die Robotik ergeben.

1.2 Aufbau der Arbeit

Diese Ausarbeit ist in sechs Kapitel untergliedert und beginnt mit der Erläuterung, wie man heutzutage als Anwendungsentwickler die Möglichkeit hat, am Beispiel der NAO-Plattform, Applikationen für Roboter auf einer hohen Abstraktionsebene zu implementieren. Im dritten Kapitel wird auf das Kernthema der Arbeit *Cloud Robtics* eingegangen, indem aktuelle Projekte vorgestellt werden. Außerdem erfolgt durch den humanoiden Roboter NAO in Form von Anwendungsszenarien eine Veranschaulichung des Themas. Zusätzlich wird im vierten und fünften Kapitel meine Version beschrieben, die ich während meines Masters verfolge.

Abschließend erfolgt eine kurze Zusammenfassung. Am Ende der Ausarbeit findet man im Anhang Informationen über aktuelle Konferenzen, die sich mit dem Thema aktuell beschäftigen.

2 High-Level-Programmierung von Robotern

Wie in der Einführung erwähnt, ist man heutzutage als Anwendungsentwickler ohne hardware-technisches Know-how in der Lage, für Roboter Applikationen auf einer hohen Abstraktionsebene zu entwickeln. Bevor der Begriff des Cloud Robotik aus *Kapitel 3* näher beleuchtet wird, wird in diesem Abschnitt der humanoide Roboter NAO vorgestellt, welcher sowohl zur Veranschaulichung als auch für Testzwecke verwendet wird.

2.1 NAO

NAO ist nach (Aldebaran-Robotics 2012) ein humanoider Roboter. Das neuste Modell *NAO next Gen* brachte die französische Firma Aldebaran-Robotics im Dezember 2011 auf dem Markt. Er besitzt einen 1,6 GHz Intel Atom Prozessor, 2 HD fähige Kameras, 4 Mikrofone, 2 Lautsprecher und eine WLAN Schnittstelle. Außerdem verfügt NAO über 11 Sensoren, um seine Umgebung wahrzunehmen und über eine 55 Watt Batterie, die mit einer Laufzeit von 1.5 Stunden ausgestattet ist.

Abbildung 1: NAO

Die CPU befindet sich im Kopf des Roboters, der die Sensordaten empfängt und die entsprechende Aktorik ansteuert. Die neue Version des NAO´s verfügt über eine neue Spracherkennung, Bluetooth und einen verbesserten Algorithmus zur Wegplanung. Zusätzlich wurde die neue Version des NAO´s mit einer Java Virtual Maschine ausgestattet, um von Java Entwicklern programmiert zu werden.

2.2 NAOqi Framework

Software-technisch liegt das Augenmerk auf das NAOqi, welches eine High-Level Programmierung des humanoiden Roboters NAO ermöglicht. NAOqi ist nach (Aldebaran-Robotics 2012) ein modularisiertes Framework, dessen Hauptprozess der mainBroker darstellt. Jedes Modul ist für einen bestimmten Aufgabenbereich zuständig, wie z.B. Spracherkennung, Gesichtserkennung, Audioverarbeitung oder Text-To-Speech und wird nach Belieben in den mainBroker zur Ausführung geladen.

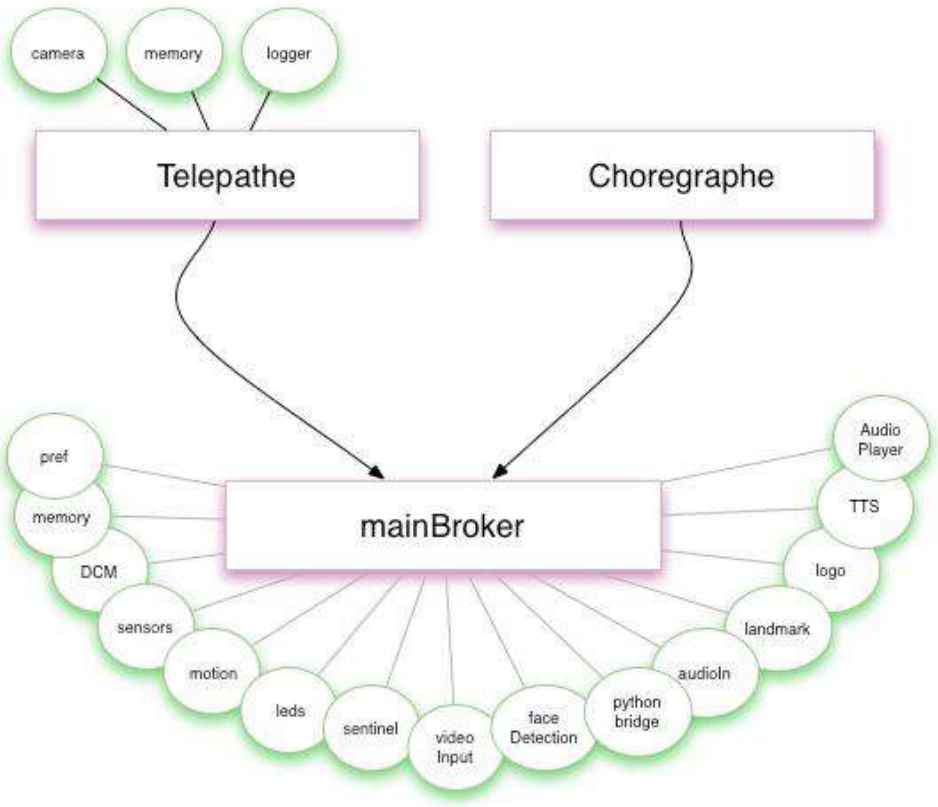

Abbildung 2: NAOqi Framework

Abbildung 2 zeigt die internen Module, die die Firma Aldebaran bereits hardwarenah in der Programmiersprache C++ implementiert haben.

2.3 Programmierung mit dem NAOqi Framework

Neben den internen Modulen gibt es die Möglichkeit eigene Module zu implementieren. Die Programmierung kann durch die Programmiersprachen C/C++, Python, MATLAB, URBI und Java realisiert werden. Im Folgenden erfolgt ein Einblick in die Entwicklung mit dem NAOqi Framework.

2.3.1 Remote Module Architecture

Die sicherste Variante Module auf die NAO-Plattform zu deployen, ist es, nach (Aldebaran-Robotics 2012) die Module auf entfernte Server zu starten. Die Module werden, wie in *Abbildung 3* ersichtlich, in einem Broker-Prozess des Servers ausgeführt und nicht direkt im mainBroker des NAO´s.

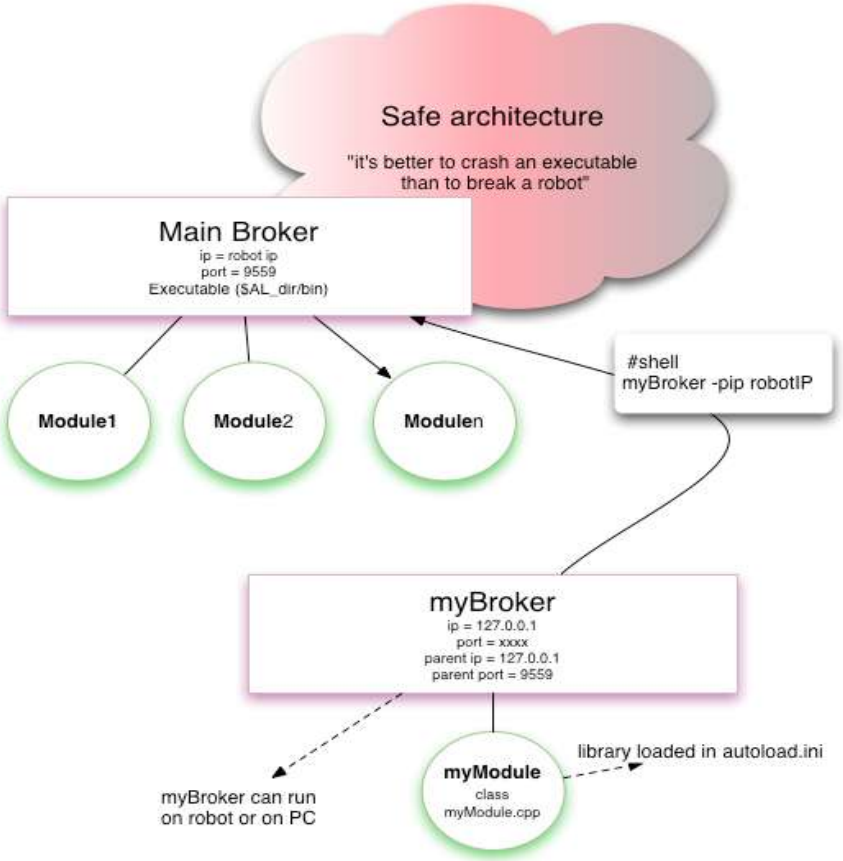

Abbildung 3: Rmote Module Architecture

Diese Architektur hat den Vorteil, dass bei einem Absturz des Moduls *MyModule* der mainBroker nicht negativ beeinflusst wird, was zu einer hohen Ausfallsicherheit beiträgt. Für die Programmierung mit dem NAOqi Bibliotheken wird daher diese Architektur gegenüber der Variante, Module im mainBroker des NAO´s direkt auszuführen, bevorzugt.

2.3.2 Choregraphe

Choregraphe ist ein plattformunabhängiges Werkzeug der Firma Aldebaran-Robtotics, mit dem man in der Lage ist, das Verhalten des NAO`s mithilfe von Flussdiagrammen auf einer hohen Abstraktionsebene zu implementieren. *Abbildung 4* stellt einen Ausschnitt der grafischen Entwicklungsumgebung dar, mit folgenden Bestandteilen:

1. Box-Bibliothek
2. Editor
3. 3D Ausgabe
4. Debug-Fenster

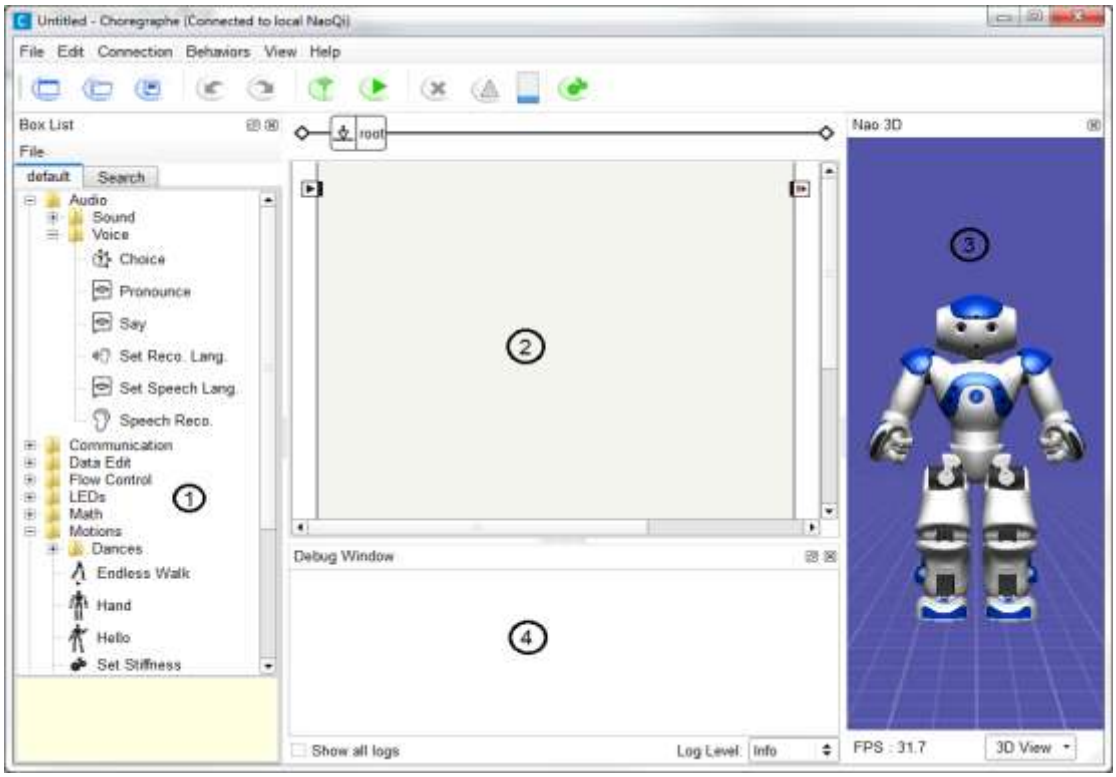

Abbildung 4: Choregraphe

Die Bibliothek besteht aus einer Menge von bereits fertig implementierten Boxen. Jede Box erfüllt eine gesonderte Aufgabe, wie z.B. *say* oder *sit down*.

Diese Boxen wurden in der Skriptsprache Python realisiert und können von Anwendungsprogrammierern verwendet werden. Um das Verhalten des NAO´s zu programmieren, kann man die bereits vorhandenen Boxen verwenden und diese miteinander verknüpfen. Des Weiteren hat man die Möglichkeit mittels Python eigene Boxen zu implementieren. Man kann das Programm entweder lokal auf seinem Rechner ausführen oder man verbindet sich in Choregraphe mittels WLAN mit dem NAO. Letzteres bietet die Möglichkeit die Befehle in Echtzeit auf dem NAO auszuführen.

2.3.3 Modulprogrammierung in einer IDE

Zudem ermöglicht das NAOqi Framework die Programmierung des NAO`s innerhalb einer IDE[1]. Somit bietet es z. B. Java Entwicklern die Gelegenheit mit typischen Java Entwicklungsumgebungen (Eclipse, Netbeans, IntelliJ IDEA) die Modulentwicklung eines NAO´s. Die folgenden Java Code Zeilen zeigen einen kurzen Einblick, wie man mit Java ein Hello World Programm für NAO implementiert.

```
import com.aldebaran.proxy.*;

static { System.loadLibrary("JNaoQi"); }

ALTextToSpeechProxy p = new ALTextToSpeechProxy("NAO.local", 9559);

p.say("hello world");
```

Über Remote Procedure Calls wird eine Verbindung zum Modul Text-To-Speech des NAO´s hergestellt. Der mainBroker-Prozess des NAO´s horcht auf Port 9559 und nimmt die Anfrage entgegen. Mittels der *say*-Methode des Text-To-Speech Proxy kann der String *hello* in eine Sprachanweisung umgewandelt werden. Für detaillierte Informationen ist hierbei auf die Homepage von Aldebaran-Robtics hinzuweisen.

[1] *Integrated Development Environment*

3 Cloud Robotics

Dieser Ansatz sieht eine physikalische Trennung zwischen Hardware und Software vor. Die üblichen Hardwarekomponenten eines Roboters, wie z.B. Sensoren, Aktoren, Kameras und Lautsprecher befinden sich weiterhin auf dem Roboter. Was sich zum weitverbreiteten Ansatz, bei dem sich Software und Hardware auf einem Roboter befinden, unterscheidet, ist, dass das Gehirn des Roboters auf entfernte Server ausgelagert ist. Der Roboter repräsentiert den Client, der Dienste aus der Cloud-Infrastruktur beansprucht. Im Bezug zur Client Server Architektur repräsentiert die Server-Seite eine Cloud aus Servern, die gemeinsam die Funktionalität eines verteilten Systems erfüllt.

Die Überlegung, komplexe und rechenintensive Operationen auf entfernte Server auszulagern, wurde bereits in den 1990er Jahren vom Japaner Masayuki Inaba in seinem Paper (Inaba 1993) vorgestellt. Seine Idee war es, Roboter ohne Gehirn auszustatten. Da es in dieser Zeit keine Prozessoren gab wie heute, die leistungsstark und platzsparend zugleich sind, versuchte Inaba die benötigten Hardwareressourcen durch diesen Ansatz möglichst gering zu halten.

Dieser Architekturstil führt dazu, dass Roboter rechenintensive Aufgaben nicht mehr on-board berechnen müssen und somit weniger Hardwareressourcen benötigen, was wiederum die Akkulaufzeit erhöht. Außerdem ermöglicht dieses Konzept, dass Roboter über einen gemeinsamen Datenspeicher miteinander kommunizieren können. Die Roboter können dadurch voneinander lernen und ihre Lernmechanismen verbessern. Sogar ist es möglich, dass Applikationen direkt aus der Cloud auf den Roboter geladen und ausgeführt werden können. Der Update-Vorgang wird ebenfalls vereinfacht, da die Software in der Cloud aktualisiert wird und somit keinen Neustart des Roboters erfordert. Nachteilig zu sehen ist es, dass durch die Auslagerung von Operationen Latenzzeiten bei der Datenübertragung zwischen Roboter und Server auftreten können. Zusätzlich ist die Kommunikation zwischen ihnen von der Bandbreite abhängig.

3.1 Projekte

In diesem Abschnitt werden vier aktuelle Forschungsprojekte vorgestellt, die sich mit dem Thema *Cloud Robotics* beschäftigen.

RoboEarth

Die Arbeitsgruppe RoboEarth entwickelte nach (Zweigle 2009) ein Word Wide Web für Roboter. Es stellt einen gemeinsamen Datenspeicher dar, mit dem die Roboter

untereinander Informationen austauschen können. Zudem können Roboter vom Verhalten anderer Roboter lernen und ihren Lernmechanismus verbessern.

RobotAppstore

RobotAppstore ist nach (Inbar 2011) der erste Marktplatz für Roboterapplikationen. Dieser Appstore bietet Software-Entwicklern die Möglichkeit, Applikationen für Roboter zu veröffentlichen und herunterzuladen. Diese Einrichtung versucht das App-Paradigma, welches durch die Smartphones einen gigantischen Durchbruch erzielt hat, auf die Robotik zu übertragen. Hierbei ist darauf hinzuweisen, dass die Anwendungen hardwareunabhängig programmiert werden und somit auf verschiedenen Roboter-Plattformen funktionieren. Unterstützt werden die Applikationen z. B. von den Robotern NAO, PLEO, KAROTZ, BIOLOID, AIBO, LEGO NXT, KEEPON, DARwin und iRobot.

Bekannte Applikationen aus dem Appstore:

- Navigator
- Suchmaschine
- Email Client
- Clients für Soziales Netzwerke
- Koch-App
- Virtueller Lehrer
- Entfernte Steuerung via Smartphone
- Medizin Erinnerungs-App
- ...

Der Hauptverantwortliche Elad Inbar stellt an die Applikationen folgende Anforderungen:

- *Performanz:* Die meisten Roboter weisen nur geringe Hardwareressourcen auf
- *Benutzbarkeit:* Alle Anwendungen sollen eine einfache und intuitive Benutzerschnittstelle mittels Spracherkennung und Gestenerkennung anbieten
- *Robustheit:* Die App-Programmierer müssen die unterschiedlichen Verfügbarkeiten bestimmter Ressourcen wie z. B. Internetzugang berücksichtigen
- *Portabilität:* Die Anwendung sollte auf allen berücksichtigten Roboterplattformen erfolgreich getestet werden

Laas

Das Laboratory for Analysis and Architecture of Systems aus Frankreich entwickelte nach (Kuffner 2011) ein User Manual Repository für Roboter. Dieser gemeinsame Datenspeicher ordnet Objekten Bedienungsanleitungen zu. Ein Roboter hat dadurch die Gelegenheit, die zu einem Objekt gehörende Bedienungsanleitung aus der Cloud herunterzuladen und entsprechend anzuwenden. Des Weiteren können diejenigen Roboter, die bereits den Umgang mit einem Objekt erlernt haben, diese Informationen im Repository veröffentlichen und anderen Robotern zugänglich machen.

ASORO

A-Star Social Robotics Laboratory ist nach (Li 2008) eine Arbeitsgruppe aus Singapur, welche eine Cloud Computing Infrastruktur entwickelte. Dieser Service dient dazu, eine 3D Karte aus der Umgebung des Roboters zu generieren. Da die Berechnungen in der Cloud geschehen, erfolgt eine schnellere Generierung der 3D Karte als on-board auf dem jeweiligen Roboter.

3.2 NAO in the Cloud

Dieser Abschnitt zeigt nach (Quintas 2011), wie ein Roboter am Beispiel vom NAO, eine Cloud-Infrastruktur nutzen kann und welche Vorteile sich daraus ergeben. Sofern NAO über WLAN mit einer Cloud verbunden ist, kann dieser seine erlernten Skills in der Cloud veröffentlichen und anderen Robotern zugänglich machen oder Informationen von der Cloud abrufen. *Abbildung 5* zeigt das erste Szenario, bei dem der NAO neue Skills vom Menschen erlernt und diese in der Cloud veröffentlicht.

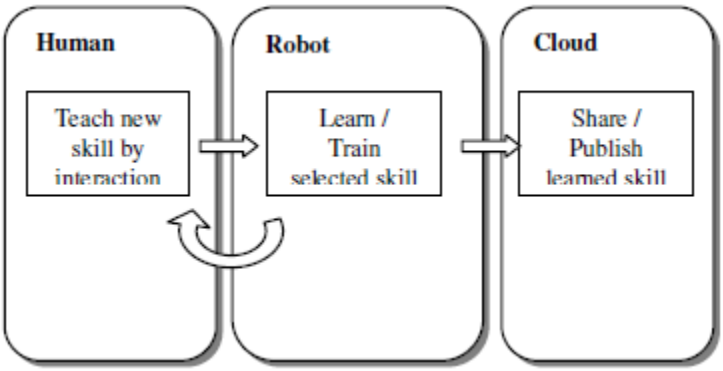

Abbildung 5: Roboter veröffentlicht Skills

Abbildung 6 demonstriert den Anwendungsfall, bei dem der NAO vom Menschen nach einem bestimmten Dienst gefragt wird. Sobald der NAO die Anfrage nicht bearbeiten kann, wendet er sich an die Cloud, um zu prüfen, ob bereits ein anderer Roboter diese Aufgabe bewältigt hat. Wenn dies der Fall ist, kann der NAO von dessen Erfahrungen profitieren.

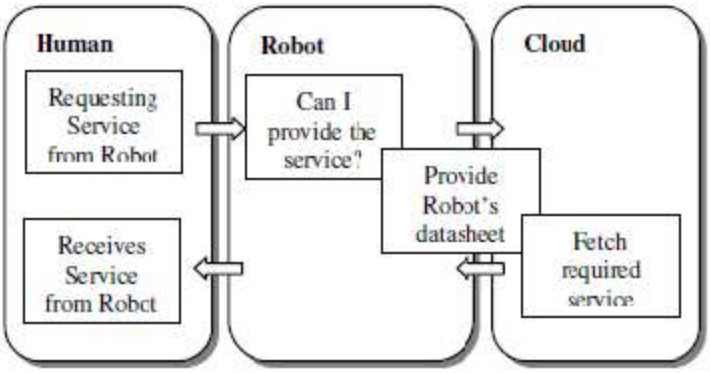

Abbildung 6: Roboter lädt Skills herunter

4 Vision

Nachdem die Grundlagen der NAO-Plattform und des Cloud Robotics erläutert wurden, wird nun der Aufgabenbereich vorgestellt, indem ich mich während meines Masters bewegen werde. Grundsätzlich möchte ich mich damit beschäftigen, wie man Anwendungen für Roboter auf einer hohen Abstraktionsebene implementiert. Dazu werde ich mich mit Roboter-Plattformen beschäftigen, wie z.B. das im *Kapitel 2* beschriebene NAOqi Framework, die es erlauben, High-Level-Applikationen für Roboter zu entwickeln.

Phase 1: *Implementierung einer prototypischen Applikation für den NAO*

Als erstes wird für den humanoiden Roboter NAO eine prototypische Anwendung implementiert. Hierzu kann z. B. ein Modul entwickelt werden, die es dem NAO ermöglicht, als Sprachübersetzer zu fungieren. Software-technisch wird dazu das NAOqi SDK verwendet. Bevor die Anwendung für den NAO entwickelt wird, erfolgt eine Prüfung der geeignetsten Programmiersprache. Mit dem NAOqi SDK kann man Module in den Sprachen C++, Python oder Java entwickeln. Dadurch, dass der NAO eine Java Virtual Maschine installiert hat, ist es sogar möglich andere JVM-basierte Sprachen wie Scala zu verwenden. Es wird ermittelt, in wie weit sich die Vorteile einer funktionalen Sprache wie Scala auf die Entwicklung von Roboterapplikationen auswirken. Falls sich Scala als geeignete Sprache herausstellt, muss zusätzlich eine Wrapper-API entwickelt werden, die auf die Java Bibliothek des JNaoQi SDK´s aufsetzt.

Phase 2: *Entwicklung einer Schnittstelle zwischen dem NAOqi SDK und RoboEarth*

Damit Roboter untereinander Informationen austauschen können, ist ein gemeinsamer Datenspeicher wie RoboEarth notwendig. Bevor jedoch ein NAO Informationen veröffentlichen oder herunterladen kann, muss eine Schnittstelle zwischen dem NAOqi SDK und einer Cloud-Infrastruktur wie RoboEarth realisiert werden. Für eine konkrete Implementierung der Schnittstelle können Webservice-Technologie wie REST[2] oder SOAP[3] benutzt werden. Diese Schnittstelle wird für den Cloud-Zugriff in den *Phasen 3* und *4* verwendet.

Phase 3: *Veröffentlichung seiner Skills in einer Cloud*

In diesem Schritt soll der NAO, wie im *Kapitel 3.2* beschrieben, seine erlernten Erfahrungen in die Cloud hochladen, um diese Informationen anderen Robotern zugänglich zu machen. Als Szenario kann die Anwendung aus *Phase 1* verwendet werden. Sobald der Anwender dem NAO ein neues Wortpaar aus Quell und Zielsprache beibringt, kann der NAO seine neu erworbenen Skills über die Schnittstelle aus *Phase 2* in der Cloud veröffentlichen.

[2] Representational State Transfer
[3] Simple Object Access Protocol

Phase 4: *Herunterladen veröffentlichter Skills von der Cloud*

Diese Phase umfasst die Umsetzung des zweiten Szenarios aus *Kapitel 3.2*. Bei diesem Anwendungsfall kann der NAO einen bestimmten Dienst des Anwenders nicht erfüllen, welcher bereits von einem Roboter umgesetzt worden ist. Indem der NAO die Informationen für diesen Dienst aus der Cloud herunterlädt, kann er aus den Erfahrungen des anderen Roboters profitieren und den Dienst erfüllen. Hierzu kann ebenfalls das Anwendungsszenario eines Übersetzers für Testzwecke hinzugezogen werden.

Phase 5: *Deployment einer eigenen Anwendung in den Appstore*

Wie bereits im *Kapitel 3.1* erwähnt, wurde bereits ein erster Appstore für Roboter entwickelt. In dieser Phase wird versucht den Anwendungsfall aus *Phase 1* plattformübergreifend aufzubereiten und im Appstore zu deployen, damit andere Entwickler diese Applikation auf unterschiedlichen Roboterplattformen testen können.

5 Probleme und Risiken

Dadurch, dass man sich bei der High-Level-Programmierung von Roboterapplikationen auf einem hohen Abstraktionsniveau befindet, bedeutet dies auch, dass man keinen Einfluss auf die darunter liegende Hardware hat. Wenn ein Anwendungsfall eintritt, bei dem beim NAO aufgrund seiner Statur oder seiner Hardware-Ressourcen Probleme auftreten, muss man die Anforderungen am NAO verringern und an seine physikalischen Gegebenheiten anpassen.

Ein weiteres Risiko besteht darin, dass man während der gesamten Arbeit mit dem NAO von der Weiterentwicklung und Wartung des NAOqi Frameworks durch Aldebaran-Robotics abhängig ist. Zusätzlich hat man Abhängigkeiten von der Cloud-Infrastruktur RoboEarth.

Verwendet man eine Cloud als gemeinsamen Datenspeicher für Roboter, so muss man sich auch mit dem Gedanken der Sicherheit beschäftigen. Es könnte sich zu einem großen Problem entwickeln, falls der Roboter für einen Dienst die dazugehörigen Informationen anfordert und gefälschte Anweisungen aus der Cloud zurück bekommt. Man muss sich sicher sein, dass diese Informationen aus der Cloud vor Angreifern geschützt sind.

6 Zusammenfassung

Diese Arbeit gab sowohl einen Einblick in die Roboterplattform NAO als auch eine Einführung in das Thema Cloud Robotics. Sie zeigte, wie man als Software-Entwickler Applikationen auf einer hohen Abstraktionsebene für Roboter entwickelt und demonstrierte die Möglichkeiten, Vorteile und Risiken, die die Robotik mithilfe des Cloud Computing erfährt. Des Weiteren wurde ein erster Appstore für Roboterapplikationen vorgestellt, der es ermöglicht, plattformunabhängige Roboteranwendungen in einen Marktplatz mit anderen Entwicklern und Anwendern zu teilen. In *Kapitel 4* wurde meine Vision vorgestellt, die ich während meines Master verfolge. Sie ist in 5 Phasen untergliedert und beschreibt die Idee, den humanoiden Roboter NAO mit einer Cloud-Infrastruktur zu verknüpfen, die es ihm erlaubt, zum Einen vom Wissen anderer zu profitieren und zum Anderen selbst erworbenes Wissen mit anderen Robotern zu teilen. Zusätzlich besteht die Aufgabe eine erste Roboterapplikation des NAO´s plattformunabhängig aufzubereiten und im Appstore zu deployen.

Literaturverzeichnis

Aldebaran-Robotics 2012 http://users.aldebaran-robotics.com/docs/site_en/index_doc.html (Zugriff am 27. 02 2012).

Aldebaran-Robotics. *NAO H25 Humanoid Robot Platform.* 2012.

Aldebaran-Robotics. *NAO Open Source.* 2011.

Cellbots. 2010. http://www.cellbots.com/ (Zugriff am 25. 02 2012).

Inaba, Kagami, Ishikawa, Kanehiro, Takeda, Inoue. *Vision-based adaptive and interactive behaviors in mechanical animals using the remote-brained approach.* Tokyo, 1994.

Inaba, Kagami, Sakaki, Kanehiro, Inoue. *Vision-based multisensor integration in remote-brained robots.* Tokyo, 1994.

Inaba, Masayuki. *Remote Brained Robots.* Tokio, 1993.

Inbar, Elad. *robotappstore.* 2011. http://www.robotappstore.com/ (Zugriff am 25. 02 2012).

Kuffner, James. *Cloud-Enabled Humanoid Robots.* 2011.

Kuffner, James. *Robots with their Heads in the cloud.* 2011.

Li, Dr. Haizhou. *ASORO.* 2008. http://www.asoro.a-star.edu.sg/ (Zugriff am 25. 02 2012).

Parent, Bastien. *NAO Open Source.* 2011.

Quintas, Menezes, Dias. *Cloud Robotics: Towards context aware Robotic Network.* 2011.

Zweigle, Molengraft, Andrea, Häussermann. *RoboEarth – connecting Robots worldwide.* Eindhoven, 2009.

Anhang

Konferenzen

ICRA: The 2011 IEEE International Conference on Robotics and Automation

(Shanghai, 09-13 May 2011)

- Ankündigung von (Aldebaran-Robotics 2011) große Teile ihres NAOqi Framework als Open Source Projekt zu veröffentlichen
- http://2011.ieee-icra.org/

RAS: 10th IEEE-RAS International Conference on Humanoid Robots

(Nashville, 06-08 December 2010)

- (Kuffner 2011) erwähnte zum ersten Mal den Begriff *Cloud Robotics*
- http://humanoids2010.org/

ICARA: 5th International Conference on Automation, Robotics and Application

(New Zealand, 06-08 December 2011)

- http://icara.massey.ac.nz/

IASTED: The Second IASTED International Conference on Robotics

(Pittsburgh, 07-09 November 2011)

- (Quintas 2011) veröffentlichte ein Paper über *Cloud Robotics*
- http://www.iasted.org/conferences/pastinfo-752.html