Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Knowledge Sharing for Robots

Sommersemester 2012

Florian Johannßen

Masterseminar 2

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Florian Johannßen

Florian.Johannssen@haw-hamburg.de

Thema

Knowledge Sharing for Robots

Stichworte

Knowledge Sharing, Cloud Robotics, Cloud Computing, RobotShare, DAvinCi

Kurzzusammenfassung

Diese Ausarbeit zeigt wie man mithilfe von Cloud Computing die Lernmechanismen von Robotern steigern kann und wie heterogene Roboter Informationen über Pläne, Objekte und Umgebungen austauschen können. Des Weiteren wird darauf eingegangen, wie rechenlastige Roboteralgorithmen auf entfernte Server ausgelagert werden können, um billigere und smartere Roboter zu entwickeln. Sie analysiert und vergleicht drei Ansätze, die sich mit Knowlegde Sharing for Robots beschäftigen.

Florian Johannßen

Title of the paper

Knowledge Sharing for Robots

Keywords

Knowledge Sharing, RoboEarth, Cloud Robotics, Cloud Computing

Abstract

This paper introduces the subject area Knowledge Sharing for Robots. It demonstrates how robots can improve their learning mechanism with Cloud Computing and shows how heterogeneous robots can exchange information about plans, objects and environments between each other. Besides it deals with the approach to swap heavy robot algorithm to remote server infrastructure with the target to be able to develop cheaper and smarter robots. This work analyzes and compares three approaches, which are involved in this research area.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

Diese Arbeit beschäftigt sich damit, wie man das Internet dazu verwenden kann, Roboter smarter und intelligenter zu gestalten und um ihre Lernmechanismen zu verbessern. Das Internet ist mittlerweile zu einen der wichtigsten Kommunikationsmedien geworden und hilft uns wichtige Informationen von bereits gelösten Problemen abzurufen, damit wir selber schneller lernen und unsere Aufgaben effizienter bewältigen können. Des Weiteren ergibt sich dadurch die Möglichkeit eigene Lösungsansätze und Informationen anderen Menschen zugänglich zu machen. Wenn man sich im Bereich der Robotik aufhält, so fragt man sich, ob sich dieses Paradigma auch auf die Robotik übertragen lässt. Da heutzutage Roboterhersteller wie *Aldebaran Robtics* in der Lage sind WLAN fähige und programmierbare Roboter mit abstrakten Schnittstellen auszuliefern, bei denen bereits die spezifischen Aufgaben, wie z. B. Gesichtserkennung, Wegplanung und Spracherkennung hardwarenah und effizient gelöst sind, sind die Voraussetzungen dafür geschaffen. Dies bietet Software-Entwicklern die Möglichkeit das Verhalten eines Roboters auf einer hohen Abstraktionsebene zu implementieren und die Vorteile des Internets auszunutzen.

An dieser Stelle wird der Begriff *Cloud Robotics* eingeführt, welcher seit kurzer Zeit von immer mehr Firmen und Forschungseinrichtungen wie (Kuffner 2011) und (Quintas, Menezes und Dias 2011) geprägt wird. 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.

Diese Überlegung, komplexe und rechenintensive Operationen auf entfernte Server auszulagern, wurde bereits in den 1990er Jahren vom Japaner Masayuki Inaba in seiner Arbeit (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.

Beim Cloud-Robotic-Ansatz repräsentiert der Roboter den Client, der Dienste aus der Cloud-Infrastruktur beansprucht und die Server-Seite eine Cloud aus Servern, die gemeinsam die Funktionalität eines verteilten Systems erfüllt. Dieser Ansatz kann wie bei (M. Inaba 1993) dazu verwendet werden, rechenintensive Aufgaben auf leistungsstarke entfernte Server, auszulagern. Zusätzlich ergibt sich dadurch die Möglichkeit, dass sich Roboter mithilfe der Cloud untereinander verständigen, koordinieren und ihre Lernmechanismen durch Knowledge Sharing verbessern.

2 Aufbau der Arbeit

Diese Arbeit beginnt mit einer Einführung, in der die Idee beschrieben wird, die beiden Gebiete Cloud Computing und Robotik miteinander zu verknüpfen. Dabei wird insbesondere im *Kapitel 3* darauf eingegangen, wie Roboter untereinander über die Cloud Informationen austauschen können und wie Roboter ihre Lernmechanismen dadurch verbessern. Im *Kapitel 4* werden einige Probleme angesprochen, mit denen man sich auseinender setzen muss, wenn man sich in diesem Themengebiet bewegen möchte. Der Schwerpunkt der Arbeit liegt darin, verwandte Arbeiten zu beleuchten, die sich mit meinem Masterthema beschäftigen. Dazu werden im *Kapitel 5* drei Projekte vorgestellt und im darauffolgenden Kapitel Vergleiche und Evaluierungen durchgeführt. Abschließend erfolgt eine kurze Zusammenfassung.

3 Knowledge Sharing for Robots

Dieser Ansatz des Wissensmanagement für Roboter umfasst das Problem, wie Roboter untereinander Informationen austauschen, damit sie von den Erfahrungen anderer profitieren können. Da die drahtlose Kommunikation durch sichere und schnellere Kommunikation attraktiver geworden ist und viele Roboter über IEEE 802.11 standardisierte WLAN Schnittstellen verfügen, ist Knowledge Sharing für Roboter über das Internet zu einem interessanten Research-Bereich geworden.

Wie in (Quintas, Menezes und Dias 2011) beschrieben, können Roboter durch dieses Vorgehen ihre Lernmeachanismen erheblich steigern, da sie nicht nur auf sich alleine gestellt sind. Sofern der Roboter über WLAN mit einer Cloud verbunden ist, kann dieser seine erlernten Skills, wie die Erkennung eines Objektes, in der Cloud veröffentlichen und anderen Robotern zugänglich machen oder Informationen von der Cloud abrufen.

Abbildung 1 zeigt das erste Szenario, bei dem der Roboter neue Skills vom Menschen erlernt und diese in der Cloud veröffentlicht.

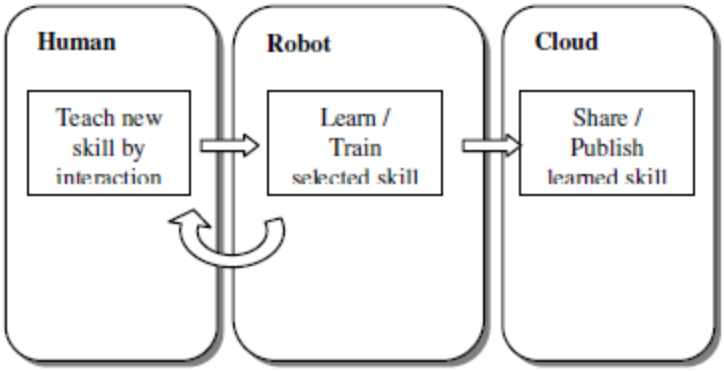

Abbildung 1: Roboter veröffentlicht Skills

Abbildung 2 demonstriert den Anwendungsfall, bei dem der Roboter vom Menschen nach einem bestimmten Dienst gefragt wird. Sobald der Roboter 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 Roboter von dessen Erfahrungen profitieren.

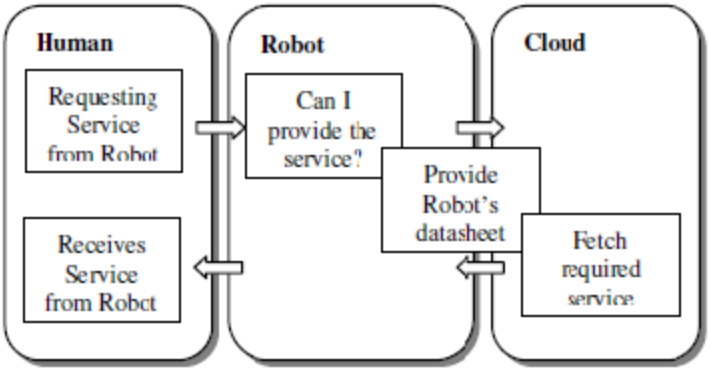

Abbildung 2: Roboter lädt Skills herunter

Durch die Vernetzung von Robotern über das Internet können Roboter gebaut werden, die nicht nur für eine bestimmte Aufgabe geeignet ist, wie es bisher in der Industrie üblich war. Falls ein Roboter für eine andere Tätigkeit zuständig sein soll, so kann dieser neues Wissen aus dem Internet erlangen. Hierbei ist natürlich darauf hinzuweisen, dass ein universell eingesetzter Roboter durch seine Hardware-Ressourcen begrenzt ist und daher nicht für jedes Aufgabengebiet eingesetzt werden kann.

4 Problemfelder

In diesem Abschnitt werden einige Teilprobleme angesprochen mit denen man sich im Rahmen dieses Gebietes auseinander setzen muss.

Latenzprobleme

Sobald man Berechnungen auf entfernte Server auslagert und Dienste über das Internet anspricht, verringert man durch die auftretenden Latenzzeiten die Performanz. Daher muss man sich darüber Gedanken machen und bei Echtzeitproblemen die Anforderungen entsprechend verringern.

Monkey and Banana Problem

Dies ist ein berühmtes Anwendungsproblem der Künstlichen Intelligenz. Es umfasst nach (Feldman 1977) die Herausforderung, wie ein Affe in einem Raum an ein an der Decke befestigtes Bananenbündel erreichen kann. Als Hilfsmittel stehen ihm ein Stuhl und ein Stock zur Verfügung. Hieber handelt es sich um ein typisches Planungsproblem, bei dem die optimale Sequenz von Handlungsschritten gesucht wird.

Falls nun ein Roboter seine optimale Handlungssequenz ermittelt hat und diesen Plan anderen Robotern zugänglich machen will, kann es sein, dass der Plan für einen anderen Roboter, der z.B. viel kleiner ist, nicht optimal ist. Hierzu werden gute Mechanismen benötigt, welche die Anforderung eines Plans mit den Fähigkeiten des Roboters vergleicht.

Heterogenität von Robotern

Da Roboter viele unterschiedliche Hardware-Plattformen und Fähigkeiten besitzen, können sie nicht direkt miteinander kommunizieren. Hierzu wird eine Middleware benötigt, welche von den Hardware-Eigenschaften der Roboter abstrahiert.

Standardisierte Datenformate

Wenn Roboter über eine zentrale Komponente Informationen austauschen wollen, muss dies über ein standardisiertes Datenformat erfolgen.

5 Analyse

In diesem Abschnitt werden drei aktuelle Forschungsprojekte vorgestellt, die sich mit den Themen *Cloud Robotics* und *Knowledge Sharing for Robots* beschäftigen.

5.1 RoboEarth

Die Forschungsgruppe RoboEarth entwickelte nach (Zweigle 2009) ein Word Wide Web für Roboter. Es stellt einen verteilten Datenspeicher dar, mit dem die Roboter untereinander Informationen austauschen können. Zudem können Roboter vom Verhalten anderer Roboter lernen und ihren Lernmechanismus verbessern. Durch RoboEarth ist es möglich, dass Roboter heterogener Plattformen untereinander Informationen über Objekte, Aufgaben und Umgebungen austauschen können. Die nächsten Abschnitte zeigen wie dies realisiert wurde.

5.1.1 Szenario

In diesem Anwendungsbeispiel gibt es zwei Roboter mit unterschiedlichen Fähigkeiten. Dem Roboter *A* wird die Aufgabe erteilt eine Flasche Wasser zu einem Patienten innerhalb eines Krankenhauses zu bringen. An dieser Stelle muss Roboter *A* selber anhand eigener Lernalgorithmen versuchen die Aufgabe zu bewältigen. Hierbei wäre unter anderem möglich, dass ein Mensch dem Roboter beibringt, wo sich die Flasche Wasser befindet und wo sich der Patient befindet. Falls dieser die Aufgabe erfolgreich gelöst hat, kann er seine erlernten Skills in einem standardisierten Format in die RoboEarth Datenbank hochladen und anderen Robotern zugänglich machen. Der Roboter *B* erhält dieselbe Aufgabe und kann nun vom RoboEarth Datenspeicher relevante Informationen herunterladen. Falls die Anforderungen der erhaltenden Informationen mit den Fähigkeiten des Roboters übereinstimmen, kann der Roboter *B* die Aufgabe bewerkstelligen ohne jemals zuvor in der Umgebung gewesen zu sein.

5.1.2 Architektur

Die RoboEarth Architektur besteht nach (Marco, et al. 2011) hauptsichtlich aus der Execution Engine, der verteilen Datenbank und dem Knowledge Processing Framework KnowRob. Ein Roboter sendet eine Anfrage, wie „Serve a drink“ im XML[1] oder Json[2] Format an die Execution Engine. Die Anfrage wird an die KnowRob Komponente delegiert, welche nach (Tenorth und Beetz 2010) für das Knowledge Processing zuständig ist. Durch KnowRob erfolgt eine semantische Suche in der RoboEarth Datenbank nach dem angeforderten Plan. Zusätzlich prüft sie, ob alle benötigen Informationen für die angeforderte Aufgabe vorhanden sind und lädt den Plan herunter. Solch ein Plan liegt in der RoboEarth Language vor und wird von der KnowRobo Komponente zur Execution Engine weitergeleitet. Dort wird der Plan in das Format der CRAM Plan Language (Rittweiler 2010) übersetzt. Dieses Format wird benötigt, damit heterogene Roboter ausführbaren Quellcode austauschen können. Jeder teilnehmende Roboter muss das Robotic Operation System (Quigley, et al. 2010) installiert haben. Die Bibliotheken von ROS enthalten entsprechende Schnittstellen zur CRAM Plan Language, damit die Pläne von RoboEarth auf dem Roboter ausgeführt werden können.

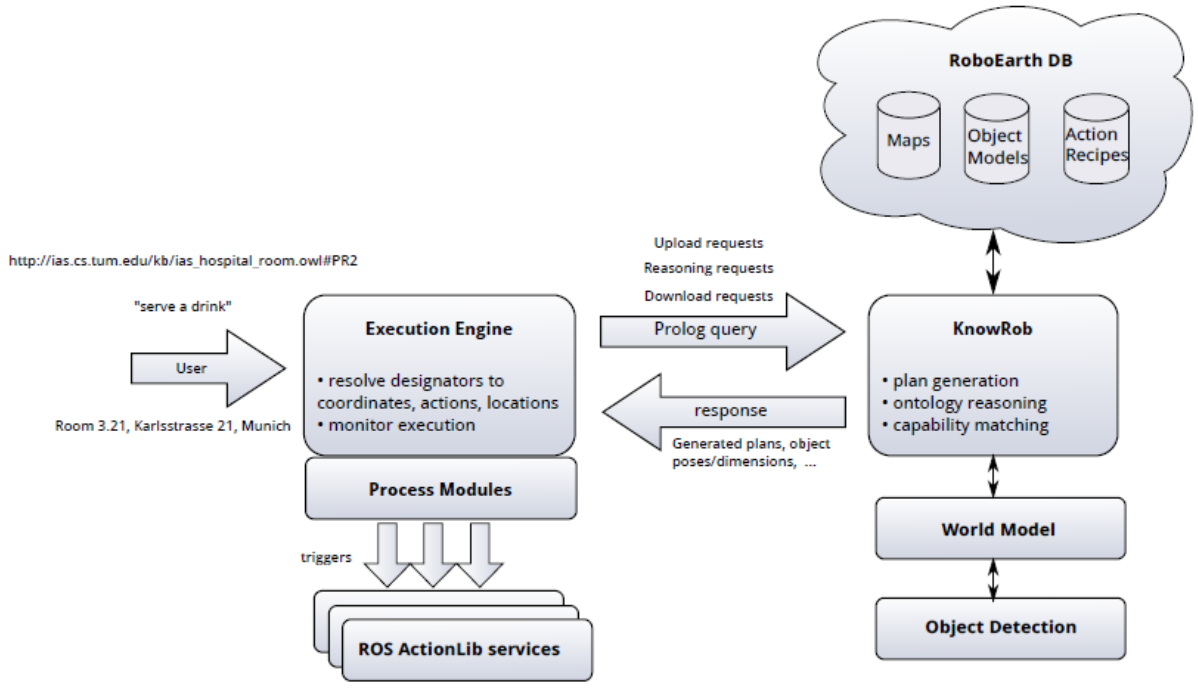

Abbildung 3: Architektur von RoboEarth

[1] Extensible Markup Language
[2] JavaScript Object Notation

Analog zum Heterogenitätsproblem aus *Kapitel 4*, verwendet RoboEarth die CRAM Plan Language und das Robotic Operation System. Dadurch ist eine Middleware geschaffen, die bewerkstelligt, dass Roboter unterschiedlicher Hardware die RoboEarth-Plattform nutzen können.

5.1.3 RoboEarthLanguage

Sie repräsentiert nach (Perzylo, Tenorth und Roehm 2010) das Format, in der die Informationen in RoboEarth verwaltet werden. Sie ist als Web Ontology Language implementiert. Sie fügt den Daten semantische Informationen hinzu und definiert die Beziehungen zwischen Aktionen, Objekten und Umgebungen. Dazu wird die Domain spezifische Ontologie KnowRob verwendet. Sie definiert eine Wissensbasis, speziell für den Austausch zwischen Roboter. KnowRob ist in Prolog implementiert und dient dem RoboEarth System zur semantischen Abfrage der Datenbank, um Pläne, Umgebungen und Objekte zu finden. Wie aus *Abbildung 3* ersichtlich, prüft KnowRob in Anlehnung zum Monkey and Banana Problem, ob die Fähigkeiten eines Roboters und die Anforderungen eines Plans übereinstimmen.

5.2 RobotShare

RobotShare repräsentiert nach (Fan und Henderson 2007) einen weiteren web-basierten Ansatz für den Wissensaustausch zwischen Robotern. Hierzu wird eine Suchmaschine wie Google entwickelt, die es Robotern ermöglicht Informationen zu Objekten und Aktivitäten abzufragen und zu veröffentlichen.

5.2.1 Szenario

Ein Roboter, der im Haushalt arbeitet, hat die Aufgabe den Abwasch zu erledigen. Nachdem er einiges an Geschirr erfolgreich gesäubert hat, ist nun ein Kochtopf an der Reihe. Jedoch hat der Roboter bisher noch nie einen Kochtopf gesehen und er weiß nicht wie er ihn zu behandeln hat. Der Roboter fragt einen Menschen, wie dieses Objekt heißt und kann den RobotShare Server nach den Kochtopf durchsuchen. Dementsprechend erhält der Roboter eine Antwort mit den URL`s, die auf das geforderte Wissen verweisen. Die Antwort kann Informationen darüber enthalten, was ein Kochtopf ist und wie ein solcher zu waschen ist. Der Roboter verbindet sich mit den Webseiten, lädt die gewünschten Informationen herunter und erhält genügend Wissen, um den Kochtopf zu säubern.

5.2.2 Architektur

Jeder Roboter erzeugt, wie in *Abbildung 4* ersichtlich, ein web-basiertes Knowledge-Repository, welches das eigene Wissen repräsentiert. Das Wissen, welches durch RoboShare verwaltet und anderen Robotern als Suchmaschine bereitstellt wird, kann aus Sensordaten, Beschreibungen oder Metadaten bestehen. Möchte ein Roboter sein Wissen der RobotShare Engine zur Verfügung stellen, so muss er diese Daten aus seiner internen Repräsentation in ein universelles Format transformieren.

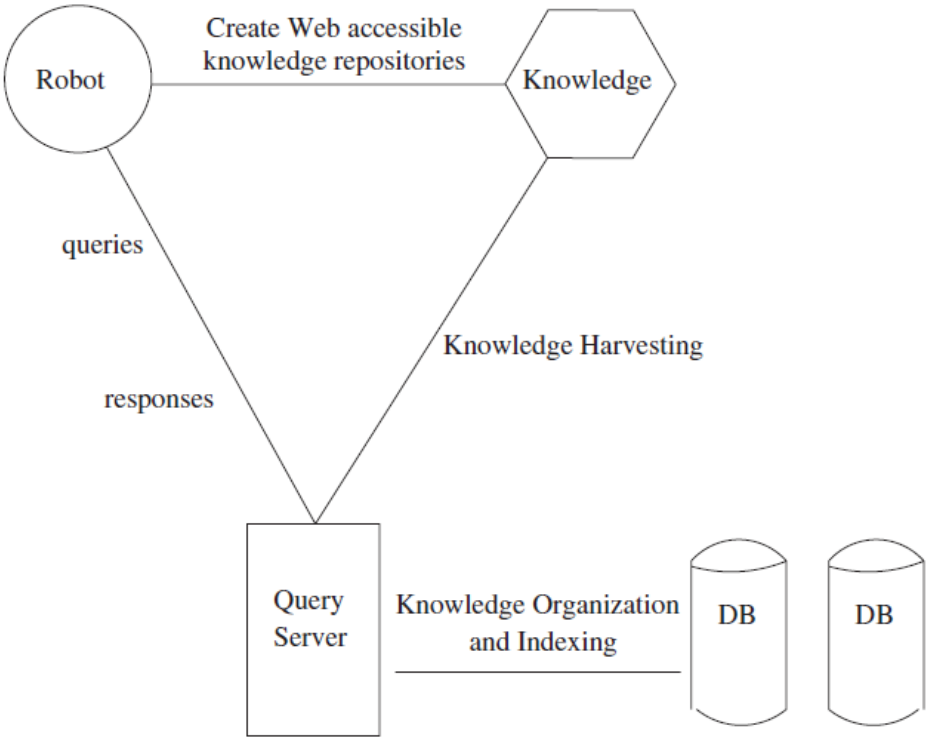

Abbildung 4: Architektur von RobotShare

Als Datenbeschreibungssprache wurde, wie *Abbildung 5* zeigt, XML verwendet. Der RobotShare Query Server fügt von allen teilnehmenden Robotern Informationen hinzu, die von den Knowledge Repositories zur Verfügung gestellt werden. In einer weiteren Transformation werden die Daten, ähnlich wie bei der Suchmaschine Google, indiziert und somit effizient in der Datenbank gespeichert. Dabei werden sie als Vektoren gespeichert. Versucht man eine Suchmaschine für Roboter zu entwickeln, besteht ein großes Problem in der Vielfältigkeit der Anfragetypen. Anders als bei Google, wo die Anfragen textuell erfolgen, besitzen die Sensordaten unterschiedliche Datenformate, wie Bilder oder Audiodateien. So werden von RoboShare intern je nach Datenformat unterschiedliche Strategien zur Indizierung verwendet.

Die nachfolgende Abbildung zeigt den Prozess dar, wenn ein Roboter dem RobotShare System eine Anfrage im XML Format stellt. Der Query Processor fungiert als Parser und übersetzt die Anfrage in ein Array von Vektoren. Die Indexing Structure Komponente versucht diese Vektoren mit den gespeicherten Vektoren zu matchen.

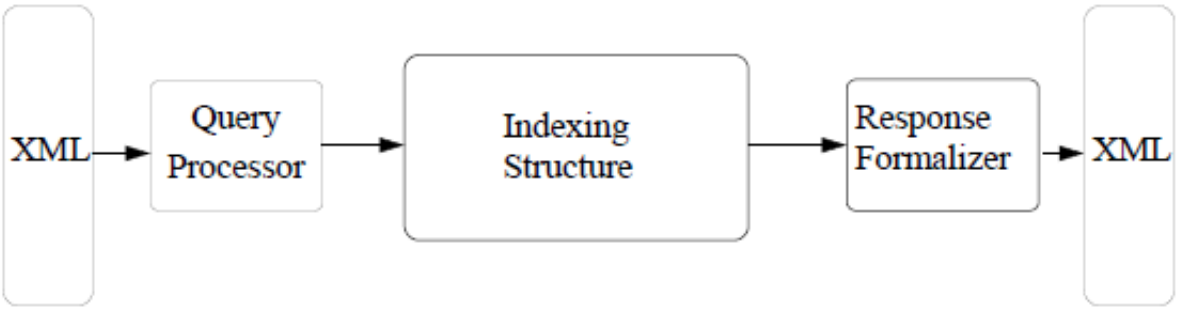

Abbildung 5: Anfrageprozess von RobotShare

Die Ergebnisse enthalten unter anderem die URL´s, die zu den entsprechenden Wissensquellen verweisen. Die Ergebnisse werden vom Response Formalizer in eine XML Datei zusammengetragen und zum anfragenden Roboter zurückgeschickt.

5.3 DAvinCi

Das DAvinCi Projekt entwickelt nach (Arumugam, et al. 2010) ein Cloud Computing Framework für Service Roboter. Sie verwenden die Map-Reduce Technik zur Parallelisierung von Robotikalgorithmen. Hierbei wird das Framework als Software as a Service verwendet, um rechenintensive Roboteroperationen, wie Bildverarbeitung oder das Erstellen von Karten, auf entfernte Server auszulagern. Ein weiterer Schwerpunkt dieses Projektes ist, dass heterogene Roboter untereinander Sensordaten austauschen können. Durch diesen Prozess werden die lokalen Karten der Roboter durch Sensordaten fremder Roboter angereichert und führt zu einer schnelleren Orientierung in fremden Umgebungen.

5.3.1 Architektur

Die DavinCi Architektur besteht, wie *Abbildung 4* zeigt, hauptsächlich aus dem Hadoop Distributed File System (HDFS) für die Datenspeicherung und dem Hadoop Map/Reduce Framework für die Parallelisierung von Roboteralgorithmen. Das Robotic Operation System (ROS) wird zur Kommunikation zwischen den Robotern und dem Cloud Service verwendet. Durch ROS wird bewerkstelligt, dass die Services von Robotern verschiedenartiger Plattformen beansprucht werden können.

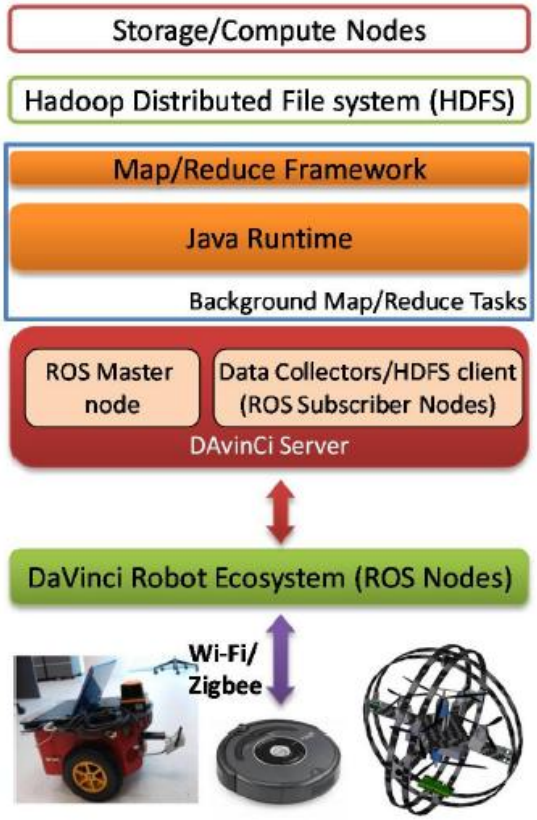

Abbildung 6: Arichtektur von DAvinCi

Roboter verbinden sich mit einem zentralen Controller und können ihre Sensordaten hochladen. Dadurch können live-Karten errichtet werden. Dies bringt den großen Vorteil, dass wenn sich mehrere Roboter in einer größeren Umgebung befinden, erhalten sie somit viel schneller eine globale Sicht auf ihre Umwelt.

5.3.2 Algorithmen

Die folgenden Algorithmen wurden mithilfe des Map/Reduce Framework parallelisiert und effizient implementiert. Ein ROS-kompatibler Roboter kann durch Ansprechen des DAvinCi Server diese Dienste ansprechen.

Simultaneous Localization And Mapping (SLAM): Diese Art von Algorithmen werden dazu verwendet, damit Roboter eine Karte ihrer Umgebung erstellen und gleichzeitig ihre Orientierung und Position errechnen können.

Path Planning: Diese Probleme beschäftigen sich damit, wie ein Roboter anhand eines Plannungsalgorithmus von einer Quelle ein bestimmtes Ziel erreicht.

Sensor Fusion: Ist eine Methode, um unterschiedliche Sensoren zu aggregieren und entsprechend zu verarbeiten. Im DAvinCi Cloud Service wird diese Technik verwendet, um aus den eingehenden Sensordaten mehrerer Roboter globale live-Karten zu erstellen.

6 Vergleich

Alle drei Ansätze beschäftigen sich damit, wie Roboter vom Prinzip des Cloud Computing profitieren können. Sie versuchen durch die Aggregierung von Informationen verschiedener Roboter deren Lernmechanismen zu verbessern. Dazu stellen sie Repositories zur Verfügung, welche die Roboter dazu verwenden, um über das Internet Wissen hochladen und abfragen zu können. Alle drei Projekte unterstützen die Kommunikation heterogener Roboter. RoboEarth und DAvinCi verwenden dazu das Robotic Operation System und RoboShare die Datenbeschreibungssprache XML.

RoboEarth ist der Ansatz, der das Prinzip des *Knowledge Sharing for Robots* am effektivsten umgesetzt hat. Im Gegensatz zu RobotShare, biete RoboEarth die Möglichkeit, dass heruntergeladene Pläne direkt auf den Robotern ausgeführt werden können. RobotShare hat als Schwerpunkt den Austausch von Objekten. Zwar können mit RobotShare auch Pläne ausgetauscht werden, jedoch wird ein Mechanismus zum Ausführen von Plänen nicht angeboten. Deshalb wurde auch eine ausführbare Repräsentation von Plänen, wie RoboEarth dies durch die CRAM Plan Language realisiert hat, nicht eingeführt. Desweiteren erlaubt RoboEarth eine intelligente und ontologie basierte Abfrage von Informationen. Da RobotShare eine Suchmaschine führ Roboter entwickelte, lag dort der Schwerpunkt auf der effizienten Indizierung der Daten und nicht auf den Aufbau einer roboterspezifischen Ontologie, um die Beziehungen zwischen Objekten und Plänen zu definieren.

DAvinCi dagegen verfolgt eher den Ansatz rechenlastige Roboteralgorithmen auf entfernte Server auszulagern. Die einzigen Informationen die hier ausgetauscht werden sind Sensordaten, die benutzt werden, um schneller globale Karten zu generieren.

7 Zusammenfassung

Diese Arbeit gab eine Einführung ist das Thema Knowledge Sharing for Robots und zeigte die Vorteile auf, die sich durch die Kombination von Cloud Computing und Robotik ergeben. Der Schwerpunkt dieser Arbeit lag darin, mehrere Ansätze vorzustellen, die sich mit diesem Themengebiet auseinander gesetzt haben. Es wurde sowohl ein Ansatz vorgestellt, welcher es erlaubt Pläne zwischen heterogenen Roboter auszutauschen und auszuführen, als auch ein Ansatz, der eine Suchmaschine für Roboter realisiert hat. Zusätzlich zeigte der Ansatz DaVinCi einen Vorschlag, um Roboteralgorithmen auszulagern, damit billigere und smartere Roboter entwickelt werden können. Nach detaillierter Analyse und Vergleiche der Ansätze hat sich das RoboEarth Projekt als interessantesten und weit fortgeschrittensten Vertreter ergeben, um Wissen zwischen Robotern auszutauschen. Daher wird dieser Ansatz auch den größten Einfluss auf meine angehende Masterarbeit haben.

Literaturverzeichnis

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

Arumugam, Rajesh, et al. *DAvinCi: A Cloud Computing Framework for Service Robots.* 2010.

Fan, Xiuyi, and Thomas C. Henderson. *RobotShare: A Google for Robots.* 2007.

Feldman, Jerome A. *Decision Theory and Artificial Intelligence: the Hungry Monkey.* 1977.

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

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

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

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

Marco, Daniel Di, Moritz Tenorth, Kai Häussermann, Oliver Zweigle, and Paul Levi. *RoboEarth Action Recipe Execution.* 2011.

Perzylo, Alexander, Moritz Tenorth, and Tobias Roehm. *The RoboEarth Language - Description of Requirements.* 2010.

Quigley, Morgan, Brian Gerkey, Ken Conley, Josh Fausty, and Tully Footey. *ROS: an open-source Robot Operating System.* 2010.

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

Rittweiler, Tobias Christian. *CRAM - Design & Implementation of a Reactive Plan Language.* 2010.

Tenorth, Moritz, and Michael Beetz. *KnowRob - Knowledge Processing for Autonomous Personal Robots.* München, 2010.

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