Sprechende Agenten

Beschreibung einer Dialogschnittstelle für Mobile Agenten

Ulrich Pinsdorf **Mehrdad Jalali**

Fraunhofer Institut
für Graphische Datenverarbeitung
Rundeturmstraße 6
64283 Darmstadt

{ulrich.pinsdorf|jalali}@igd.fhg.de

In diesem Artikel wird eine Softwarearchitektur beschrieben, die Mobilen Agenten eine verbale Interaktion mit dem Menschen ermöglicht. Das Ergebnis des Dialoges zwischen Agent und Benutzer wird in einer formalen Nachricht zusammengefasst und dem Agenten zugestellt. Diese Nachricht ist in einer Agentenkommunikationssprache abgefasst, so dass der Agent keine Fähigkeiten benötigt, um natürliche Sprache zu verstehen. Das System ist prinzipiell in der Lage, über beliebige Ein- und Ausgabekanäle mit dem Benutzer zu interagieren (Multimodalität).

1. Vorwort

Das vorgestellte System wurde entworfen und implementiert am Fraunhofer Institut für Graphische Datenverarbeitung, Darmstadt, und setzt auf der SeMoA[1]-Plattform [8] auf. Bei SeMoA handelt es sich um ein ebenfalls an diesem Institut durchgeführtes Forschungsprojekt, mit dem Ziel der Konzeption und Implementierung einer Plattform für Mobile Agenten. Diese Plattform ist weitgehend sicher vor Angriffen zwischen den Mobilen Agenten, sowie zwischen Mobilen Agenten und Agentenservern [8]. Die SeMoA-Plattform wird, zusammen mit dem hier vorgestellten Dialogsystem, in einem vom BMBF geförderten Forschungsprojekt zum Einsatz kommen.

2. Motivation

Die rasante Entwicklung des World Wide Webs und des Electronic Commerce in den letzten Jahren zeigt den Trend, Informationen, Produkte und Dienstleistungen immer näher an den Kunden zu bringen. Die Entwicklung zeigt aber auch, dass Anbieter von Informationsinhalten bereit sind, in Technologien zu investieren, die es dem Kunden erlauben, immer einfacher und bequemer auf die angebotenen Informationen zuzugreifen. Dabei kommt es unter Wettbewerbern zu regelrechten Wettläufen, wenn es um einen möglichst einfachen Zugriff auf ihre Angebote geht [6].
Eine weitere Stufe der Vereinfachung, besonders in den Bereichen Electronic Commerce und Informationsbeschaffung, stellt die Verwendung Mobiler Agenten dar [6]. Mobile Agenten sind ein Paradigma, das die Softwareentwicklung über dieses Jahrzehnt hinaus prägen wird [3]. Aufgrund ihres autonomen Handelns, ihrer Mobilität und der Möglichkeit zur Personalisierung eignen sie sich in besonderer Weise für Anwendungsszenarien in den Bereichen des Electronic Commerce und der Informationsbeschaffung [5]. Möchte ein potentieller Kunde beispielsweise Angebote zu einem Produkt einholen, kann er einen personalisierten Mobilen Agenten starten, der die Aufgaben der Informationsbeschaffung, der Evaluation nach vorgegeben Kriterien und u.U. sogar den Einkauf für ihn erledigt.

Damit der Agent im Auftrag des Benutzers autonom agieren kann, benötigt er eine aufgabenspezifische Instruktion durch den Benutzer. Dabei muss besonderer Wert auf eine benutzerfreundliche Mensch-Maschine-Schnittstelle des Agenten gelegt werden, damit die Vorteile einer agentenbasierten Lösung in Bezug auf die Vereinfachung der Informationsbeschaffung nicht von einer schlechten Bedienbarkeit aufgewogen werden.
Eine Interaktion mit dem Agenten mittels natürlicher Sprache, oder sogar eine multimodale Interaktion, ist also notwendig, um die Nutzbarkeit und Akzeptanz Mobiler Agenten zu erhöhen. Ein weiterer Schritt in Richtung menschgerechter Benutzerschnittstelle stellt die Verwendung von multimodalen Avataren dar [11].
Das hier vorgestellte Dialogsystem erlaubt eine multimodale Interaktion zwischen Agenten und Benutzer. Weitere Ein- und Ausgabesysteme lassen sich leicht in das Dialogsystem integrieren.

[1] SeMoA steht für „*Se*cure *Mo*bile *A*gents“

Erschienen in: Spierling, Ulrike (Hrsg.): Digital Storytelling - Tagungsband. Stuttgart : Fraunhofer IRB Verlag, 2000, S. 151-161.

3. Anforderungen an ein Dialogsystem

Es werden nun eine Reihe von Anforderungen an ein Dialogsystem für Mobile Agenten vorgestellt, die sich während der Problemanalyse als sinnvoll ergeben haben. Das vorgestellte Dialogsystem ist so konzipiert, dass es alle genannten Anforderungen erfüllt.

Medienunabhängigkeit

Das Dialogsystem soll Mobilen Agenten die Interaktion mit dem Benutzer ermöglichen. Die zu verwendenden Interaktionskänale, also die Eingabe- und Ausgabemedien, sollen nicht vom Agenten, sondern vom Benutzer festgelegt werden. Dies ermöglicht eine weitgehende Entkopplung der eigentlichen Dialogführung vom Agentensystem. Dies ist sowohl für den Agenten, wie auch für den Benutzer von Vorteil. Einerseits ist der Agent dadurch unabhängig von den auf dem Migrationshost zur Verfügung stehenden Ein- und Ausgabemedien. Andererseits ist der Benutzer unabhängig von der Implementierung des Agenten; er kann dem System vorschreiben, über welche Medien er kommunizieren möchte. Dies ist zum Beispiel für solche Menschen von großem Vorteil, die auf Grund einer Behinderung nicht über alle Modi kommunizieren können.
Mit anderen Worten: der Agent bestimmt den Inhalt der Kommunikation und der Benutzer den Verlauf und die Modalität des Gesprächs.

Formale Repräsentation des Dialogergebnisses

Der Agent soll das Ergebnis des Dialoges, anstatt in Form von natürlicher Sprache[2], in einer formalen Repräsentation bekommen. Unter einem formalen Ergebnis wird hier die Repräsentation des Gesprächsergebnisses in einer Agentenkommunikationssprache (z.B. FIPA ACL[3], KQML[4], ...) verstanden. Dadurch wird es für den Agenten wesentlich einfacher, die erhaltenen Antworten zu interpretieren, zumal viele Agentensysteme ohnehin über die Fähigkeit verfügen, mittels Agentenkommunikationssprachen zu kommunizieren.
Das Ziel eines formalen Kommunikationsergebnisses wird erreicht durch a) eine Formalisierung der Benutzerantworten während des Dialoges und b) die Bereitstellung eines Übersetzungsdienstes auf dem SeMoA-Server (vgl. Abschnitt „Funktionsweise des KQML Translation-Service“). Formalisierung bedeutet, dass in der Dialogbeschreibung die möglichen natürlichsprachlichen Eingaben des Benutzers auf wohldefinierte Begriffe abgebildet werden. Durch diese, beim Design des Dialoges getroffene begriffliche Abbildung entfällt für den Agenten das sehr aufwendige Verstehen[5] der natürlichen Sprache.
Anhand der eindeutigen Begriffe kann eine Übersetzung des Dialogergebnisses in eine formale Sprache erfolgen. Im vorliegenden System wurde ein Übersetzungsservice für die Generierung von KQML-Nachrichten implementiert.

Leichtgewichtigkeit der Agenten

Mobile Agenten werden gerne in Verbindung mit Mobile Computing verwendet, da sie keine permanente Netzverbindung benötigen. Nach der Instruktion durch ihren Auftraggeber können sie z.B. von einem PDA[6] über eine temporär aufgebaute Netzverbindung zu einem Server im Festnetz migrieren und dort ihre Aufgabe erledigen. Aufgrund der Fähigkeit zum autonomen Handeln kann die Netzverbindung nach der Migration des Agenten wieder abgebaut werden. Da die Bandbreiten mobiler Netzanbindungen sehr begrenzt sind, und dies wohl auch in Zukunft bleiben wird[7], ist Leichtgewichtigkeit ein wichtiges Akzeptanzkriterium für Mobile Agentensysteme. Leichte Agenten ermöglichen eine schnellere Übertragung und damit eine zügigere Erledigung ihres Auftrages.
Die vorgestellte Implementation nimmt auf diese Anforderung Rücksicht, indem soviel Verarbeitung wie möglich auf der Seite des Agentenservers erfolgt. Die Agenten werden von der Dialogführung und der Interpretation der

[2] Hier ist neben Audiosignalen auch die textuelle Form der Benutzeräußerung gemeint
[3] Von der „Foundation for Intelligent Physical Agents“ (FIPA) definierte „Agent Communication Language“ (ACL). Siehe [2]
[4] „Knowledge Query and Manipulation Language“, siehe [1]
[5] In der englischsprachigen Literatur spricht man im Kontext der Sprachverarbeitung von „Natural Speech Understanding“ (NLU) und „Natural Speech Recognition“ (NLR). Die NLR bildet aus einem Audiostrom Wörter und Sätze. Sie ist somit die Voraussetzung für die NLU, die versucht, den semantischen Inhalt des Gesagten zu erfassen. An dieser Stelle ist NLU gemeint.
[6] Personal Digital Assistent
[7] Es stehen zwar neue Techniken vor der Markteinführung, die höhere Bandbreiten für mobile Endgeräte garantieren, allerdings ist zu erwarten, dass im Zuge dieser Verbesserung auch Anspruch und Umfang mobiler Anwendungen steigen werden, und somit Bandbreite bei mobilen Netzzugängen eine knappe Ressource bleibt.

Erschienen in: Spierling, Ulrike (Hrsg.): Digital Storytelling - Tagungsband. Stuttgart : Fraunhofer IRB Verlag, 2000, S. 151-161.

natürlichen Sprache befreit, und müssen nur noch anhand der in einer formalen Sprache präsentierten Ergebnisse reagieren [10].

Einfache Verwendbarkeit des Systems

Für den Anbieter von Dienstleistungen oder Informationen soll es möglichst einfach sein, einen Agenten zu kreieren, der bspw. Benutzern Angebote unterbreitet, Umfragen durchführt oder Informationen einholt.
Im allgemeinen können für gleiche oder ähnliche Aufgaben immer die gleichen Agenten, mit geringen Modifikationen, wiederverwendet werden. Die inhaltliche Anpassung erfolgt anhand zweier Dateien, die der Agent mit sich führt. Bei der ersten Datei handelt es sich um eine Beschreibung des Dialoges, der mit dem Benutzer geführt werden soll. Sie ist in VoiceXML[8] [12] abgefasst, einer von XML abgeleiteten Dialogbeschreibungssprache, die für die Automatisierung von Call-Centern entwickelt wurde. Die Flexibilität von VoiceXML erlaubt die vollständige Beschreibung eines Dialogekontrollflusses mit Gesprächsverzweigungen und –Iterationen in Abhängigkeit von den Antworten des Benutzers.
Bei der zweiten, vom Agenten mitgeführten Datei, handelt es sich um eine Reihe von Übersetzungsschablonen. Mit deren Hilfe wird aus den formalisierten Antworten des Benutzers eine Nachricht erzeugt, die in einer Agentenkommunikationssprache abgefasst ist. Dieser Vorgang wird im Abschnitt „Funktionsweise des KQML Translation-Service“ näher beschrieben.

4. Aufbau des Systems

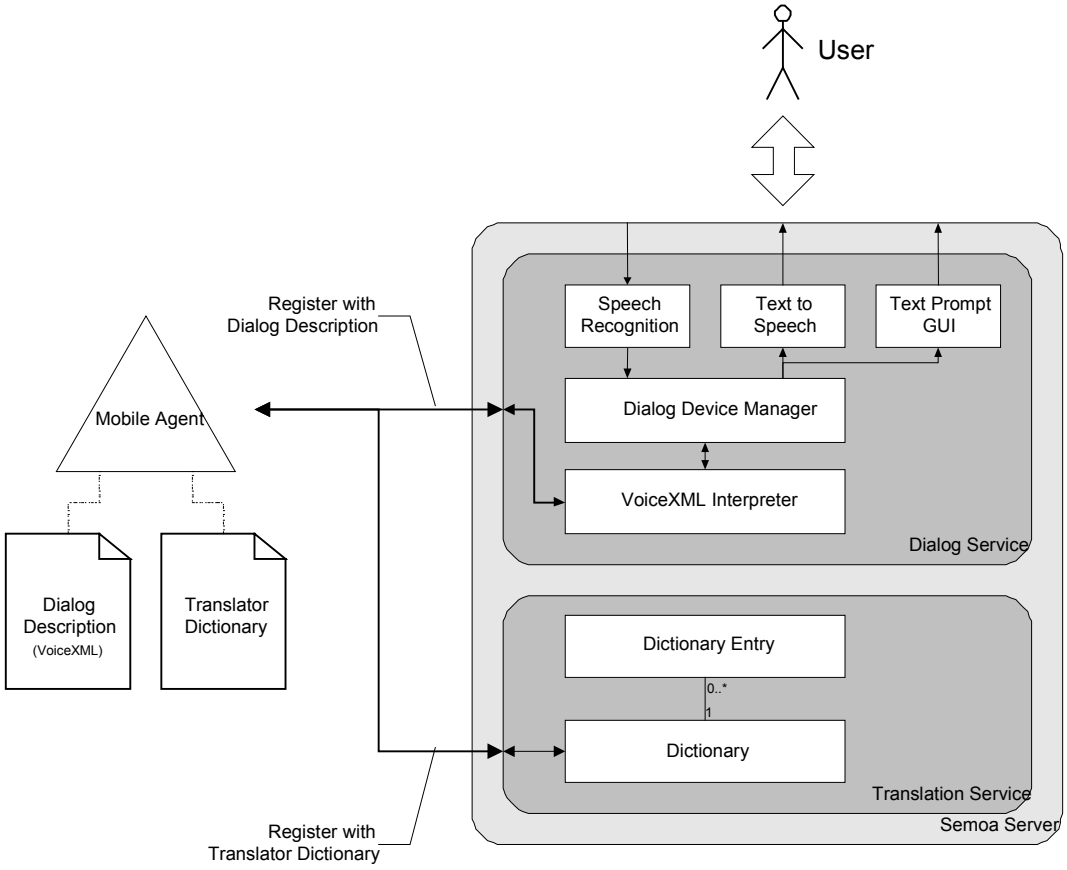

Abbildung 1 : Systemarchitektur des SeMoA-Dialogsystems

Der Systemaufbau des SeMoA-Dialogsystems ist in Abbildung 1 dargestellt. Er ist vollständig in Java™ realisiert. Um die oben definierten Ziele zu erreichen, wurde das System als SeMoA-Service implementiert. Mittels der Services kann ein SeMoA-Server seine Funktionalität erweitern und den Agenten spezielle Dienstleistungen anbieten. Die Teilsysteme werden nun näher erläutert.

[8] VoiceXML ist eine Dialog-Beschreibungssprache, definiert von einem Konsortium, angeführt von IBM, Motorola, Lucent Technologies und AT&T.

Der Mobile Agent

Möchte der Agent einen Dialog mit dem Benutzer führen, übergibt er dem Dialog-Service eine Beschreibung des zu führenden Dialoges. Der Service kümmert sich dann darum, dass der Dialogauftrag in eine Warteschlange eingereiht und zur Ausführung gebracht wird. Typischerweise führt der Agent die Dialogbeschreibung in Form einer Datei mit sich; es spricht aber prinzipiell nichts dagegen, dass sie erst zur Laufzeit vom Agenten kreiert und an den Service übergeben wird.
Die Dialogbeschreibung ist in VoiceXML abgefasst. Eine einfache Dialogbeschreibung könnte z.B. wie in Abbildung 2 dargestellt aussehen.
Es wird zunächst ein Formular „pizza_order“ definiert, das seinerseits ein Dialogfeld „topping“ enthält. Dieses besteht aus Ausgaben (prompts) und möglichen Eingaben (grammar). Letztere werden in einer der Backus-Naur-Form ähnlichen Syntax angegeben. Um den Dialog für den Benutzer abwechlungsreicher gestalten zu können, besteht die Möglichkeit, mehr als einen Prompt zu definieren. Weitere Prompts werden mit einem *count*-Attribut unterschieden und ausgewählt [12]. Damit lassen sich Ansprachen realisieren, die zunächst allgemein formuliert sind und – wenn der Benutzer nicht antwortet – immer genauer beschreiben, wie er antworten muss (inkrementelle Prompts [13]).

```
<vxml>
      <form id="pizza order">
            <field name="topping">
                  <prompt>
                        Hello, I'm your pizza order agent.
                        I can order various pizzas for you.
                        What kind of pizza would you like?
                  </prompt>

                  <prompt count=2>
                        Please tell me the topping of your pizza.
                        You can have tunafish, mushrooms or
                        tomatoes.
                  </prompt>

                  <grammar>
                        ( [I'd like to have a] |
                          [Give me a] )
                        pizza with
                        ( tunafish  {tonno}  |
                          mushrooms {funghi} |
                          tomatoe   {napoli} )
                  </grammar>
            </field>
            ...
      </form>
      ...
</vxml>
```

Abbildung 2 : Beispiel einer in VoiceXML abgefassten Dialogbeschreibung

Der SeMoA Server

Der SeMoA Server stellt die Laufzeitumgebung und die Sicherheitsinfrastruktur für die Agenten zu Verfügung. Die Basisfunktionalität des Servers lässt sich durch Services erweitern. Vom Host angebotene Services können von einem Agenten genutzt werden[9]. Das hier vorgestellte Dialogsystem verwendet die beiden Services *Dialog-Service* und *KQML-Translation-Service*.

Funktionsweise des Dialog-Services

Der Dialogservice nimmt vom Mobilen Agenten die auszuführende Dialogbeschreibung entgegen und sortiert sie in eine FIFO-Warteschlange ein. Sobald der Dialog zur Ausführung kommt, wird die Beschreibung einem VoiceXML-Interpreter übergeben, der die Datei parst und die notwendigen Ein- und Ausgaben veranlasst. Der VoiceXML-Interpreter steuert den Kontrollfluss im Dialog und hält auch VoiceXML-Variablen und den Dialogzustand vor. Für die Ein- und Ausgabe bedient sich der VoiceXML-Interpreter eines Dialoggeräte-Managers (Dialog-Device-Manager) und ist damit geräteunabhängig.

[9] Das Sicherheitskonzept von SeMoA sieht vor, dass der Agent dazu eine Berechtigung benötigt [8, 9].

Soll eine Ausgabe an den Benutzer erfolgen, übergibt der Interpreter den auszugebenden Text an den Dialoggeräte-Manager. Dieser sendet dann einen p*rompt*-Request an alle ihm bekannten Ausgabegeräte (DialogOutputDevice). Diese einzelnen Ausgabegeräte sind dann für die Ausgabe auf ein angeschlossenes Gerät verantwortlich. Im Rahmen der vorgestellten Entwicklung wurden Geräte entwickelt, die eine Sprachausgabe (via Text-To-Speech) sowie eine Textausgabe in einem Fenster erlauben. Das Entwickeln weiterer Ausgabegeräte stellt kein Problem dar; so könnte z.B. ohne großen Implementierungsaufwand eine Braille-Zeile oder ein Avatarsystem als Ausgabekanal integriert werden.

Das Entgegennehmen von Eingaben funktioniert analog zur Ausgabe. Die, vom Dialoggeräte-Manager verwalteten Eingabegeräte (DialogInputDevices), melden diesem eine gültige Eingabe. Hierbei können mehrere Geräte um die Eingabe konkurrieren. Diese Architektur bereitet eine echte Multimodalität, bei der sich mehrere Eingabemodi ergänzen, bereits vor. In dem vorliegenden System wird vom Dialoggeräte-Manager die erste gültige Eingabe als Antwort gewertet. Das bedeutet, dass sich der Benutzer bei jeder Interaktion, je nach Art der Frage, für den angenehmsten Modus entscheiden kann.

Ist die Dialogbeschreibung auf diese Weise vom VoiceXML-Interpreter abgearbeitet worden, werden die Ergebnisse an den Agenten zurückgegeben. Diese Ergebnisse sind bereits formalisiert und bestehen aus einer Liste von Schlüssel-Wert-Paaren. Der Schlüssel ist stets der Name eines Dialogfeldes, der Wert ein in der Antwortgrammatik definiertes Ergebnisstring[10]. Beispielsweise wird in der, in Abbildung 2 dargestellten Dialogbeschreibung die Antwort „I'd like to have a pizza with mushrooms" abgebildet auf `topping=funghi`.

Funktionsweise des KQML Translation-Service

Der Translation-Service übersetzt die Resultate[11] des Dialog-Services in eine formale Sprache. Im vorgestellten System wird eine Übersetzung in KQML vorgenommen. Um den Translation-Service nutzen zu können, übergibt der Agent dem Service ein sogenanntes *Dictionary*. Dieses spezifiziert, wie die einzelnen Parameter der Nachricht jeweils zu füllen sind. Abbildung 3 zeigt ein Beispiel eines solchen Dictionaries.

```
# Global settings
sender   = mobilePizzaAgent
language = LISP
ontology = pizzaOrder

[order]
performative = TELL
content = (order (pizza ($(topping) $(extra_cheese))))

[price request]
performative = ASK
content = (price (amount $(count)) (pizza ($(topping) $(extra_cheese))))
```

Abbildung 3 : Beispiel eines Dictionaries für KQML

Im dargestellten Fall besteht das Dictionary aus einem Abschnitt mit globalen, d.h. für alle Nachrichten gültigen Angaben, und zwei nachrichtenspezifischen Abschnitten. Möchte der Agent den Translation-Service verwenden, übergibt er diesem den Rubrikennamen und eine Menge von Schlüssel-Wert-Paaren. Der Service sucht im ersten Schritt den übergebenen Rubrikennamen im Dictionary. Aus den dort angegebenen, sowie den globalen Informationen wird die KQML Nachricht zusammengesetzt. Die in der Schablone verwendeten Variablen werden mittels der Schlüssel-Wert-Paare substituiert. Der vollständige Übersetzungsvorgang bis zur fertigen KQML-Nachricht ist in Abbildung 4 dargestellt.

[10] Dieser wird in VoiceXML mit geschweiften Klammern in die Grammatik eintragen.
[11] eine Menge von Schlüssel-Wert-Paaren

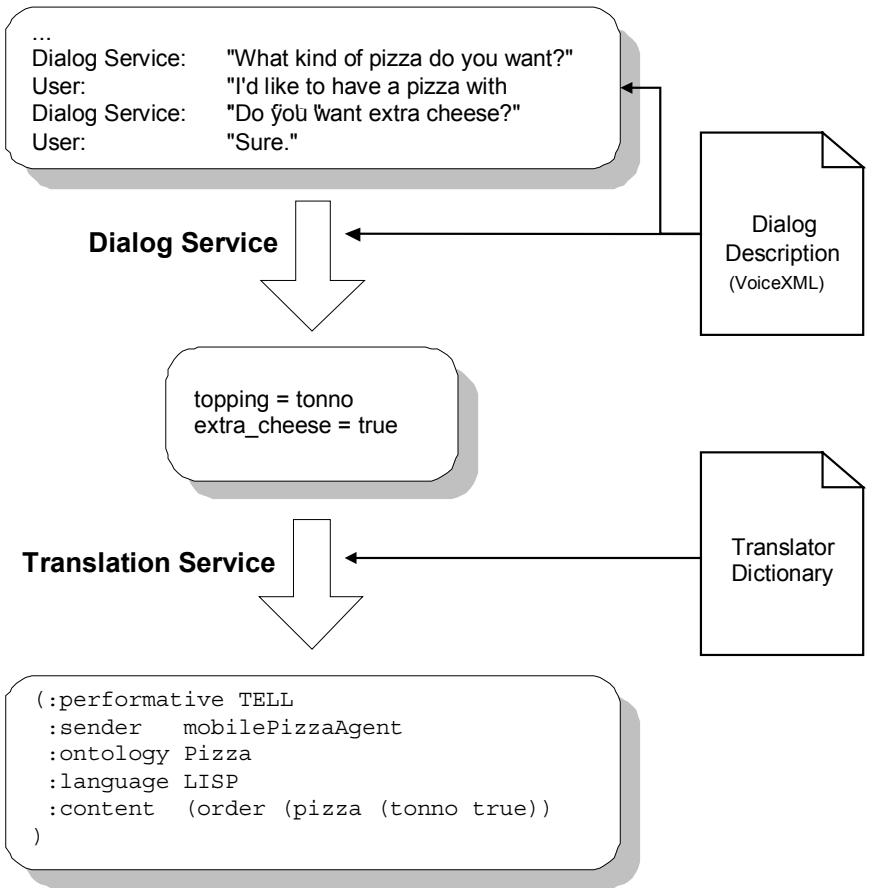

Abbildung 4 : Übersetzung der natürlichen Sprache nach KQML

Trotz fest definierter Übersetzungs- und Dialogbeschreibung ermöglicht der so konzipierte Translations-Service größtmögliche Flexibilität zur Laufzeit. Der Agent bzw. dessen Entwickler kann an folgenden Punkten Einfluss auf die Auswertung des Dialoges nehmen:

- Der Dialog-Kontrollfluss kann von Antworten des Benutzers abhängig gemacht werden.
- Je nach Ablauf des Dialoges können andere Dialogvariablen als Ergebnis zurückgegeben werden.
- Der Agent kann das Übersetzungs-Topic anhand des Dialog-Resultats zur Laufzeit bestimmen.
- Der Agent kann das Dialogresultat, d.h. die Schlüssel-Wert-Paare vor der Übersetzung verändern.
- Die Variablenexpansion erlaubt zur Laufzeit parametrisierbare Nachrichten.

Der Benutzer

Ein agentenbasierter Arbeitsplatz hält eine Reihe von Agenten bereit, die jeweils auf eine bestimmte Aufgabe spezialisiert sind, bspw.. Shopping, Terminvereinbarung, Umfragen, etc. Der Benutzer startet bei Bedarf einen Agenten in seiner Arbeitsumgebung und dieser erfragt dann in einem Dialog die Randbedingungen seines Auftrags. Da Agenten Aufgaben autonom erledigen, steht dem Benutzer damit ein Delegationssystem zur Verfügung. Möchte ein Benutzer z.B. eine Umfrage zu einem bestimmten Thema durchführen, delegiert diese Aufgabe an einen (oder mehrere) Agenten, die den Rechner der zu befragenden Personen aufsuchen und seine Meinung einholen. Die Verwendung von Mobilen Agenten setzt ein Vertrauen bei Auftraggeber und Empfänger des Agenten voraus [7].

5. Zusammenfassung

Die vorgestellte Architektur erlaubt einem Mobilen Agenten, mit dem Benutzer eines SeMoA-Servers in einen interaktiven Dialog zu treten. Der Dialog wird vom Auftraggeber des Agenten in einer Dialogbeschreibung spezifiziert. Diese Flexibilität von VoiceXML erlaubt die Kontrolle des Dialogflusses in Abhängigkeit von den Antworten des Benutzers. Aus Gründen der Sicherheit, der Synchronisation und der einfacheren Schnittstelle zum Dialogservice wird der Dialog nicht vom Agenten selbst, sondern von einem Dialog-Service für den Agenten geführt. Dabei verwendet der Dialogservice die ihm bekannten, lokal zu Verfügung stehenden Medien, um mit dem Benutzer zu kommunizieren. Die Antworten des Benutzers werden, in einer durch die Dialogbeschreibung festgelegten Weise, als formales Resultat an den Agenten zurückgegeben.

Erschienen in: Spierling, Ulrike (Hrsg.): Digital Storytelling - Tagungsband. Stuttgart : Fraunhofer IRB Verlag, 2000, S. 151-161.

Der Translation-Service übersetzt dieses formale Resultat dann mit Hilfe von, vom Agenten mitgeführten, Übersetzungsregeln in eine formale Sprache – hier KQML. Dabei wählt der Agent eine passende Übersetzungsschablone in Abhängigkeit vom Ergebnis des zuvor geführten Dialoges. Die Verwendung von Variablen in den Schablonen erlaubt es, Nachrichten erst zur Laufzeit zu parametrisieren.

6. Quellen

[1] Finin, T.; Labrou, Y.; Mayfield, J. :
KQML as an Agent Communication Language. In: Bradshaw, J. (Hrsg.) :
Software Agents, MIT Press, 1997.

[2] Foundation for Intelligent Physical Agents, http://www.fipa.org/

[3] Gilbert, Don :
The Role of Intelligent Agents in the Information Infrastructure,
IBM Intelligent Agent Center of Competency, 1995

[4] Lenzmann, Britta :
Benutzeradaptive und multimodale Interface-Agenten. (Dissertationen zur künstlichen Intelligenz; Bd. 184)
Infix, Sankt Augustin 1998.

[5] Maes, Pattie; Guttmann, Robert; Moukas, Alexandros:
Agents that Buy and Sell: Transforming Commerce as we Know It, In: Communication of the ACM,
3/1999.

[6] Mattern, Friedemann :
Mobile Agenten. In: it+ti - Informationstechnik und Technische Informatik, 4/98, S. 12ff, Fachbereich
Informatik, Technische Universität Darmstadt,
http://www.informatik.tu-darmstadt.de/VS/Publikationen/papers/mobags.html

[7] Norman, Donald:
How Might People Interact with Agents. In: Communications of the ACM, 7/1994, S.68ff.

[8] Roth, Volker; Jalali, Mehrdad :
Semoa – Secure Mobile Agents, Unveröffentlichtes Manuskript, Fraunhofer Institut für Graphische
Datenverarbeitung, Darmstadt.

[9] Roth, Volker; Jalali, Mehrdad:
Access Control and Key Management for Mobile Agents. In: Computer & Graphics, Vol. 22, 4/1998,
S. 457-461.

[10] Russel, Stuart; Norvig, Peter :
Artificial Intelligence. A Modern Approach. Pentice Hall International, 1995.

[11] Spierling, Ulrike; Behr, Johannes :
Conversational Integration of Multimedia and Multimodal Interaction. In: CG topics 4/1999, S.8ff.

[12] VoiceXML Consortium, http://www.voicexml.org/

[13] Zink, Michael :
Zusammenspiel. Mensch-Maschine-Schnittstellen in Sprachsystemen. In: c´t 3/1999, S.133ff.

Erschienen in: Spierling, Ulrike (Hrsg.): Digital Storytelling - Tagungsband. Stuttgart : Fraunhofer IRB Verlag, 2000, S. 151-161.

Citation:

Pinsdorf, Ulrich; Jalali-Sohi, Mehrdad:
Sprechende Agenten - Beschreibung einer Dialogschnittstelle für Mobile Agenten.
In: Spierling, Ulrike (Hrsg.): Digital Storytelling - Tagungsband.
Stuttgart : Fraunhofer IRB Verlag, 2000, S. 151-161
(Computer Graphik Edition 2).