In: Eibl, Maximilian; Wolff, Christian; Womser-Hacker, Christa (Hg.): Designing Information Systems. Festschrift für Jürgen Krause. Konstanz: UVK Verlagsgesellschaft mbH, 2005. S. 101 – 106.

Speech Grammars und Dialogdesign

Ludwig Hitzenberger

Zusammenfassung
Mittels Speech Grammars und einer Dialogmodellierungsskriptsprache lassen sich Dialoge spezifizieren. Die Komplexität liegt dabei ausschließlich im Dialogdesign. Die Anforderungen an die Mächtigkeit einer Sprachgrammatik und die Dialogspezifikationssprache sind daher sehr gering, d.h. es reicht aus, einen Klassifikator für wordspotting einzuführen, der über den nächsten Interaktionsschritt entscheidet.

Abstract
Dialogues are specified through speech grammars and dialogue modelling languages. The complexity of the dialogues comes from the dialogue design. The power of the grammar and the specification language needs not to be very high. It is sufficient to have a very simple classificator, which decides on the next dialogue step.

1 Speech Grammars und Dialogdesign

Ein Mensch-Maschine-Dialog definiert sich über zwei Dinge: eine Grammatik, die die Sprache beschreibt, die im Dialog zur Anwendung kommen kann, und eine Dialogablaufspezifikation, die festlegt, welche Dialogschritte aufeinander folgen können und welche Aktionen der Maschine damit verbunden sein werden. Am Lehrstuhl für Informationswissenschaft der Universität Regensburg wurde ein Dialogmanager entwickelt, der mit einer Standardgrammatik und einer eigens entwickelten Dialogmodellierungssprache für die Spezifikation von Dialogen geeignet ist (Brey et. al 2003). Er wird u. a. verwendet, um Studenten in den Dialogentwurf einzuführen.

1.1 Speech Grammars

Speech Grammars sind Grammatiken gesprochener Sprache für den Sprachgebrauch in Dialogen. Sie müssen damit die Phänomene der gesprochenen Sprache abhandeln können und auch Probleme der Spracherkenner, wie z.B. Fehlerkennungen mit abdecken. Für diese Art von Grammatiken wird üblicherweise eine JSGF-Grammatik (Java Speech Grammar Format) verwendet. Es handelt sich dabei um ein von der Firma SUN Microsystems in Zusammenarbeit mit anderen Sprachtechnologiefirmen entwickeltes, herstellerunab-

hängiges Standardformat für Erkennergrammatiken, das eine kontextfreie Grammatik vom Typ Regelgrammatik definiert.

Speech Grammars haben eine sehr einfache Struktur und sind mit normalen Grammatiken für natürliche Sprachen nicht zu vergleichen. Sie haben weder das Ziel, die Grammatik einer natürlichen Sprache vollständig zu erklären, noch eine Diktiergrammatik zu sein, mit der bei automatischen Diktiersystemen die möglichen und richtigen Abfolgen der gesprochen Wörter erkannt werden sollen, sondern sie dienen letztlich nur dazu, ein Lexikon festzulegen, das die zu parsenden Schlüsselwörter des Dialogschritts enthält, evtl. zusätzlich mit ein paar Standardphrasen, wie sie typischerweise in diesem Sprachregister Dialoge verwendet werden. D.h. sie beschreiben nur einen ganz minimalen Ausschnitt der natürlichen Sprache.

1.2 Grammatikformalismus

Der Grammatikformalismus für Speech Grammars ist entsprechend einfach:
Es gibt nur zwei Patterns für Regeln:

(1) <ruleName> = ruleExpansion;
(2) public <ruleName> = ruleExpansion;

Die Regelexpansion definiert, wie die Regel in dem Interaktionsschritt, in dem sie angewendet wird, gesprochen werden kann. Es ist eine logische Kombination von tokens (= sprechbarer Text) und Referenzen auf andere Regeln (JSGF 1998, S. 8). Die als public erklärten Regeln stellen die aktiven Regeln des Erkenners dar.

In der ursprünglichen Grammar Format Spezifikation von Sun ist nicht definiert, wie sich die Semantik komplexer Ausdrücke aus der Semantik der Teilausdrücke zusammenfügen lässt. In der Originalversion können semantische Tags nur als applikationsspezifische Informationen an die Regeln hinzugefügt werden (JSGF 1998, S. 12), um die Verarbeitung der Erkennerergebnisse zu erleichtern. Für eine Dialoggrammatik ist es deshalb sinnvoll, dem Parser die Semantik in Form von Merkmal-Wert-Paaren zur Verfügung zu stellen. Dieses Grammatikformat wurde von der Firma TEMIC eingeführt (StarRec 2000, S. 11) und für den Dialogmanager des Lehrstuhls übernommen (Brey et. al 2003, S. 28).

1.3 Mächtigkeit

Für eine Dialoggrammatik kommt es lediglich darauf an, über den nächsten Dialogschritt bzw. die nächste Aktion zu entscheiden. Dafür ist vom Parser keine komplexe Analyse der Äußerung notwendig, sondern lediglich das Erkennen der Semantik der Benutzerintention. Diese Intention lässt sich in der Regel aus Schlüsselwörtern der Benutzeräußerung ableiten, da aus der Kenntnis der Dialogsituation die Möglichkeiten der Dialogfortführung sehr eingeschränkt sind.

Das folgende Beispiel zeigt den Zusammenhang:

(1) public <yes,no> = <yes> | <no>;
(2) <yes> = ja [in Ordnung] | ok | richtig {answer = yes} ;
(3) <no> = nein | falsch | nö {answer = no} ;

Da die natürliche Sprache hoch redundant ist, sind in den meisten Benutzeräußerungen offensichtlich viele Wörter vorhanden, die nichts Entscheidendes zur Semantik der Äußerung in Sinne der oben erwähnten Unterscheidbarkeit für einen Klassifikator beitragen. Der Parser kann sich also in der Regel auf wordspotting beschränken. Trotzdem ist es für den Erkenner wichtig zu wissen, wo sich diese Füllwörter befinden. Die JSGF-Grammatik definiert nicht, wie solche Wörter oder Phrasen zu codieren sind. Für diese Wörter können für den Erkenner Füllwörter (Garbage) definiert werden, die für jedes beliebige Wort ohne eine spezielle Bedeutung stehen. Die folgende Regel akzeptiert mit Hilfe des Wiederholungsoperators (*) eine beliebige Zahl von zu ignorierenden Wörtern, gefolgt von Schlüsselwort „Hilfe“ und wiederum gefolgt von beliebigen weiteren Wörtern.

(1) <help> = %unknowns* Hilfe %unknowns*

Was bleibt ist eine Grammatik, die das eigentliche Ziel einer Grammatik, nämlich die möglichst vollständige Beschreibung einer Sprache, nicht mehr erfüllen muss. Und das ist im Sinn einer Dialoggrammatik auch nicht notwendig. Der Sinn einer Dialoggrammatik reduziert sich auf das Klassifizieren von Schlüsselwörtern und der Zuordnung einer rudimentären Semantik. Diese Grammatiken sind linguistisch gesehen damit nicht mehr erklärungsadäquat, sondern haben nur noch die Anwendungsadäquatheit als Ziel. Dieser Ansatz ist nicht grundsätzlich neu, sondern er findet sich z.B. bereits bei Krause 1979, dort allerdings nicht für gesprochene Sprache, sondern für über eine Tastatur eingegebene schriftliche Dialoge.

2 Dialogdesign

Wie bereits für die Grammatik festgestellt, gilt auch für die Dialogstruktur, dass die einzelnen Schritte in der Regel sehr gut vorhersagbar sind, und dass Dialoge eine sehr einfache Grundstruktur aufweisen. Aus der jeweiligen Dialogposition lässt sich viel für die Interpretation der einzelnen Schritte ableiten. Folgendes Beispiel zeigt, dass die Inhalte der einzelnen Dialogschritte semantisch sehr arm sein können und sich doch die Dialogintention immer noch gut erkennen lässt. So ist z.B. auch der Dialog in der zweiten Spalte im richtigen Kontext durchaus analog zur ersten Spalte interpretierbar:

Verzeihung	Ej!
Ja, kann ich Ihnen helfen?	Uh?
Äh, könnten Sie das mal anschauen?	Hier!
Sicher, einen Moment bitte.	Mhm
Sehen Sie das Problem?	Ok?
Nein, ich fürchte ich versteh nicht, was Sie meinen.	Eh?
Aber schauen Sie hier!	Hier!
Ah, ja, jetzt weiß ich was Sie meinen.	Ah!

Tab. 1: Dialogbeispiel nach Campbell, 2003, S.5; übers. d.V.

Der Schluss daraus ist, dass wesentliche Teile der Dialogsemantik aus dem Ablauf und der Situation ableitbar sind. Von daher kommt dem zweiten Teil der Spezifikation eines Dialogs, nämlich der Festlegung des Dialogablaufs und der dazu durchzuführenden Aktionen, genauso viel Bedeutung zu wie der Grammatikdefinition. Da ein Dialog ein potentielles Sprachdokument darstellt, bietet sich in Analogie zu textuellen Dokumenten an, die abstrakte Beschreibung in einer SGML- bzw. XML-basierten Notation auszuführen.

2.1 SDML

Am Lehrstuhl für Informationswissenschaft wurde eine einfache Dialogmodellierungssprache (DML) entwickelt, die etwa im Gegensatz zu VoiceXML ausschließlich zur Dialogbeschreibung dienen soll. Diese Dialogmodellierungssprache enthält nur die Auszeichnungen (Tags), die zur Ablaufspezifizierung notwendig sind. Sie werden in einer DTD-Vereinbarung definiert. Ein DTD-Parser kann die Wohlgeformtheit von XML- bzw. DML-Dokumenten feststellen. In der DTD werden die Elemente mit ihrer Hierarchie und ihrem potentiellen Inhalt, sowie ihre dazugehörigen Attribute angegeben.

Im Dialogdesign ist zunächst festzulegen, welche Dialogschritte unter welchen Bedingungen aufeinander folgen können und welche Aktionen sie auslösen können. Die wichtigsten dafür notwendigen Elemente der DML sind die hier folgenden Tags:

(1) Step
(2) Speech output
(3) Speech input
(4) Action
(5) Transition
(6) Prompt

Ein Dialog besteht aus einzelnen Dialogsteps (1). Für jeden Step gibt es optional eine Sprachein- bzw. -ausgabe mit Prompts (2,3,6). Im Tag Speech Input (3) wird die für diesen Step gültige Grammatik angegeben. Durch diese Möglichkeit, für jeden Dialogschritt eine eigene Grammatik anzugeben, kann der jeweils zu erkennende Wortschatz klein gehalten werden und dadurch die Erkennungssicherheit deutlich erhöht werden.

Aktionen (4) werden ausgeführt, wenn die zugehörigen Bedingungen erfüllt sind. Durch die Interaktion mit dem Applikationsinterface dienen die Aktionen zur Steuerung und zur Verwaltung dialogrelevanter Daten. Bei allen bisherigen Anwendungen war es ausreichend vier Aktionen zu benutzen:

(1) Store
(2) Submit
(3) Request
(4) Reset

Alle weiteren Aktionen werden über das Applikationsinterface ausgeführt. Diese sind z.B. Variablen setzen oder abfragen, Datenbankzugriffe etc.

Die Transition (5) beschreibt den Übergang zum nächsten Step. Mit diesen einfachen Mitteln lässt sich im wesentlichen ein Dialog steuern und beschreiben. Der Vorteil bei diesem Vorgehen ist, dass funktionale Teile aus dem Dialog weitgehend ausgelagert sind und sich die DML fast ausschließlich auf die Dialogbeschreibung beschränkt.

2.2 Systeminitiative vs. Benutzerinitiative

Eine weitere Möglichkeit im Dialogdesign ist der Wechsel zwischen Systeminitiative und Benutzerinitiative. Bei Fehlern, Missverständnissen oder Problemen im Dialog kann das System von sich aus die Dialoginitiative ergreifen

und dadurch zu einer Reduzierung der Komplexität beitragen. Der Dialog kann soweit vereinfacht werden, dass der Benutzer nur noch mit einer Ja-Nein-Entscheidung antworten kann. In dieser Situation sind dann weder Fehlerkennungen noch Fehlentscheidungen möglich. Das macht zwar das Dialogdesign wesentlich aufwändiger, versteckt aber die Komplexität vor dem Benutzer, der sonst selbst eine geeignete Fehlerbehebungsstrategie entwickeln müsste.

3 Fazit

Wie oben gezeigt, steckt die Komplexität eines Dialogs fast ausschließlich in der Komplexität des Designs. Dies erlaubt eine Reduktion der Komplexität und Mächtigkeit der Grammatik und gleichzeitig eine Reduktion der Komplexität des Dialogs gegenüber dem Benutzer. Auch die zum Dialogdesign notwendige Skriptsprache ist von ihrer Mächtigkeit her sehr einfach strukturiert. Der Dialog reduziert die Aufgabe der Grammatik auf einen Entscheider, welche der Möglichkeiten im Step ausgewählt werden müssen und ist damit nur noch ein einfacher Klassifikator.

4 Literaturverzeichnis

Brey, Thomas; Kaiser, Jan; Stauber, Jürgen (2003). VSE Speech Kit. Handbuch. Speech Experts, Regensburg (masch.).

Campbell, Nick (2003). “Talking with Robots – the Message behind the Words.” In: TSD2003, 6th. Conf. on Text, Speech and Dialogue. České Budéjovice, Czech Rep, Sept. 8-11, 2003.

JSGF (1998). Java Speech Grammar Format. Version 1.0, Oktober 1998. Mountain View/CA: Sun Microsystems.
http://java.sun.com/products/java-media/speech/forDevelopers/JSGF/index.html
[Zugriff September 2004].

Krause, Jürgen (1979). “Powerful Grammar Structures. An approach to Question-answering-Systems and Automatic Indexing.” In: Malachin, Z. (ed.) (1979). Proceedings of the Shefayim International Conference on Literary and Linguistic Computing. Jerusalem, 23-41.

StarRec GDS (Grammar Development System) (2000). User´s Guide. Temic Speech Processing, Ulm.