Saarbrücken Dissertations | Volume 39 |
in Language Science and Technology

Social Talk Capabilities for Dialogue Systems

Tina Klüwer

Deutsches Forschungszentrum
für künstliche Intelligenz
German Research Center for
Artificial Intelligence

universaar
Universitätsverlag des Saarlandes
Saarland University Press
Presses Universitaires de la Sarre

Tina Klüwer

Social Talk Capabilities for Dialogue Systems

Deutsches Forschungszentrum
für künstliche Intelligenz
German Research Center for
Artificial Intelligence

universaar
Universitätsverlag des Saarlandes
Saarland University Press
Presses Universitaires de la Sarre

Saarbrücken Dissertations
in Language Science and Technology
(Formerly: Saarbrücken Dissertations
in Computational Linguistics and Language Technology)

http://www.dfki.de/lt/diss.php

Volume 39

Saarland University
Department of Computational Linguistics and Phonetics

German Research Center for Artificial Intelligence
(Deutsches Forschungszentrum für künstliche Intelligenz, DFKI)
Language Technology Lab

D 291

Universitätsverlag des Saarlandes
Saarland University Press
Presses Universitaires de la Sarre

Postfach 151150; 66041 Saarbrücken
ISBN 978-3-86223-173-7 gedruckte Ausgabe
ISBN 978-3-86223-174-4 Online-Ausgabe
ISSN 2194-0398 gedruckte Ausgabe
ISSN 2198-5863 Online-Ausgabe
URN urn:nbn:de:bsz:291-universaar-1353

zugl. Dissertation zur Erlangung des Grades eines Doktors der
Philosophie der Philosophischen Fakultäten der Universität des Saarlandes
Tag der letzten Prüfungsleistung: 03.07.2014
Erstberichterstatter: Prof. Dr. Hans Uszkoreit
Zweitberichterstatter: Prof. Dr. Manfred Stede
Dekan: Univ.-Prof. Dr. Ralf Bogner

Projektbetreuung universaar: Susanne Alt, Matthias Müller

Satz: Tina Klüwer
Umschlaggestaltung: Julian Wichert

Gedruckt auf säurefreiem Papier von Monsenstein & Vannerdat

Bibliographische Information der Deutschen Nationalbibliothek:
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen
Nationalbibliografie; detailierte bibliografische Daten sind im Internet über
<http://dnb-d-nb.de> abrufbar.

Abstract

Small talk capabilities are an important but very challenging extension to dialogue systems. Small talk (or "social talk") refers to a kind of conversation, which does not focus on the exchange of information, but on the negotiation of social roles and situations. Small talk is often not as limited in topics and content as so-called "task talk", meaning conversations regarding a specific task of a dialogue system such as, e.g., providing bus schedule information.

Several studies have shown that human users tend to initialize social talk in conversations with dialogue systems, especially if the dialogue system is embedded in an application that includes aspects of human personality such as embodied avatars. Moreover, studies have shown that social conversations can effectively establish an "emotional" connection between the user and the machine and create a pleasant atmosphere which is appreciated by most users.

However, only few existing dialogue systems offer small talk support and nearly none systematic analysis of small talk usable for computational purposes has been proposed so far.

The goal of this thesis is to provide knowledge, processes and structures that can be used by dialogue systems to satisfactorily participate in social conversations.

For this purpose the thesis primarily presents, besides research in the fields of natural-language understanding and dialogue management, research on dialogue models and error handling. Regarding dialogue models, a new structured model of social talk based on a data analysis of small talk conversations is described. The functionally-motivated and content-abstract model can be used for small talk conversations on various topics. The model is based on a novel, theory-based set of social dialogue acts and is also available as computational model learned from conversation data.

Since it cannot be guaranteed that all contents for social conversations initialized by the users of a dialogue system have been modeled, this thesis also suggests new conversation strategies for the treatment of so-called "out-of-domain" (OoD) utterances. OoD utterances are utterances which do not fall within one of the knowledge domains of the system and thus lead to errors in the input interpretation. These errors cannot be handled using the typical error strategies such as a repair, because the knowledge necessary to understand these utterances is missing. The new strategies are based on information from human-human communication extracted from various sources.

The presented research is technologically encapsulated in a software toolkit. The toolkit provides software extensions to dialogue systems that enable social talk. For evaluation the tools are integrated into a conversational agent application: a barkeeper agent in a virtual world. Two evaluations, an overall usability evaluation of the agent and an evaluation of the two main tools, indicate a clear improvement in the users' perception of the agent when the tools are activated, especially in the areas of naturalness, natural-language understanding and conversation flow. The fun the users had while using the application seems to be strongly related to the system's social talk abilities.

Zusammenfassung

Small Talk Gespräche sind für Dialogsysteme einerseits eine lohnende Erweiterung, auf der anderen Seite aber auch eine große Herausforderung. Small Talk (oder auch „social talk“) bezeichnet eine Art von Gespräch, in der nicht der Austausch bestimmter Informationen im Vordergrund steht, sondern das Verhandeln von sozialen Rollen und Situationen. Small Talk ist dadurch in Themen und Inhalten häufig nicht so stark limitiert wie so genannter „Task Talk“, also Gespräche, die zu einem Aufgabenbereich eines Dialogsystems (wie bspw. der Auskunft zu einem Bussystem etc.) gehören.

Unterschiedliche Studien haben gezeigt, dass menschliche Benutzer dazu neigen, mit Dialogsystemen auch soziale Gespräche zu führen, die über eine bestimmte Aufgabe des Systems hinaus gehen. Dies gilt vor allem dann, wenn das Dialogsystem in eine Applikation eingebettet ist, die ein Verkörperung beinhaltet (z.B. menschliche Avatare). Darüber hinaus haben verschiedene Arbeiten gezeigt, dass soziale Gespräche effektiv eine „emotionale“ Verbindung zwischen Benutzer und Maschine herstellen und eine angenehmere Atmosphäre schaffen können und dass dies von vielen Benutzern geschätzt wird.

Trotzdem bieten nur wenige existierende Dialogsysteme Small Talk Unterstützung an und bisher wurde keine systematische Analyse von Small Talk, die auch für computationelle Zwecke eingesetzt werden kann, vorgeschlagen.

Die vorliegende Arbeit zielt darauf ab, Dialogsystemen Wissen, Prozesse und Strukturen anzubieten, die sie nutzen können, um in sozialen Gesprächen zufriedenstellend zu partizipieren.

Dazu gehört zum Einen ein neues strukturelles Modell von Social Talk basierend auf einer Datenanalyse von Small Talk Gesprächen. Das funktional-motivierte und inhaltlich abstrakte Small Talk Modell kann für Gespräche über diverse Themen genutzt werden. Das Modell basiert auf einem eigens entworfenen, theoretisch fundierten Set von sozialen

Dialogakten und ist ebenso als Computermodell, aus den Gesprächsdaten gelernt, verfügbar.

Da nicht garantiert werden kann, dass alle Inhalte für soziale Gespräche, die die Benutzer eines Dialogsystems initiieren, modelliert worden sind, schlägt die vorliegende Arbeit zum Anderen neue Gesprächsstrategien für das Behandeln so genannter „Out-of-Domain“ (OoD) Äußerungen vor. OoD-Äußerungen sind Äußerungen, die nicht in eine der Wissensdomänen des Dialogsystems fallen und deshalb zu Fehlern in der Eingabeinterpretation des Systems führen. Diese Fehler können nicht mittels der klassischen Fehlerstrategien (wie bspw. einer Reparatur) behandelt werden, da das notwendige Wissen um diese Äußerungen zu verstehen, fehlt. Die neuen Strategien basieren auf Informationen aus Mensch-zu-Mensch Kommunikation, die aus unterschiedlichen Quellen extrahiert wurden.

Die vorgestellte Forschung ist technologisch in einem Software-Toolkit gekapselt. Das Toolkit bietet Software-Erweiterungen für Dialog-Systeme, die soziale Gespräche ermöglichen. Zu Evaluationszwecken wurden die Tools in eine Agenten-Anwendung integriert, einem Barkeeper-Agenten in einer virtuellen Online-Welt. Zwei Evaluationen, eine allgemeine Usability-Evaluation des Agenten und eine Evaluation der beiden Haupttools des Toolkits, weisen auf eine deutliche Verbesserung in der Wahrnehmung des Agenten durch die Benutzer hin, wenn die Tools aktiviert sind, vor allem in den Bereichen “Natürlichkeit”, “Sprachverstehen” und “Gesprächsfluss”. Der Spaß an der Benutzung des Systems scheint stark mit der Fähigkeit zu sozialen Gesprächen zusammenzuhängen.

Contents

List of Figures

List of Tables

1 Introduction

Dialogue systems are part of many every-day life situations nowadays. They serve different purposes and can cross our way in a plurality of outside appearances such as a banking hotline, a journey planner system, Apple's Siri assistant[1], an artificial computer game character you can talk to, or even a talking robot in a consumer electronics retailer. All these miscellaneous applications make use of different sophisticated or simple dialogue systems to offer a conversational interface. The end-user, at the same time, often does not know about the insides of the dialogue system and talks to a system in the same way she would to another human. Human language in general but especially in discourse contains numerous very challenging phenomena on all linguistic levels such as syntax, semantics and pragmatics.

One common phenomena in dialogue is social talk or "small talk". Research in the field of dialogue systems, originating from computational linguistics, artificial intelligence or related fields has found many good solutions for many of the challenges occurring in dialogue. Nevertheless, there is still space for important improvements in "new" aspects of dialogue such as multi-modal input, handling of emotions and social behavior. This work focuses on social-talk capabilities for dialogue systems.

Small talk is often perceived as boring and superfluous chit-chat in which content exchange is irrelevant and negligible. Following this definition small talk represents the opposite of task-driven talk. The term "task-driven talk", or also "task-bound" or "task-oriented" talk, is used in this thesis to denote a kind of dialogue which serves the execution of a particular task.

Traditional dialogue systems are task-bound systems. That means they are developed for one special purpose, such as a banking service, a bus and underground information system, or navigating through a

[1] http://www.apple.com/ios/siri/

website. Task-oriented dialogue systems are a lot easier to develop than non-task-oriented systems, since unrestricted natural language input is impossible to process by machines.

Several studies have detected the "task" of *small talk* not to lie in knowledge negotiation, but in the management of social situation. In the early 1920s Bronsilaw Malinowski already introduced the term "phatic communion" to denote a kind of talk which "serves to establish bonds of personal union between people" (Malinowski, 1949, page 316). It seems small talk is far from superfluous, but an important medium to establish and negotiate social relationships. Although people know machines are not establishing social relationships, they tend to use social talk with machines, too.

In the last few years, dialogue systems have tried to account for the importance of social talk and have moved more and more in the direction of being more conversational, allowing for at least a bit of social talk. Two main groups can be differentiated. The first group, which I will call "entertaining applications", has the specified goal to allow for "open-domain talk", which means talking about an unrestricted number of topics and things. These systems were long time neglected by dialogue system research, because the knowledge-rich procedures, which had been the focus, cannot be applied to an unlimited number of domains. Instead, the problem was tackled by the chatbot tradition originating from Weizenbaum's ELIZA (Weizenbaum, 1966). The result is a huge amount of manually and community-based encoded databases of pattern-answer pairs in surface form. One example is the freely available ALICE chatbot[2] based on the Artificial Intelligence Markup Language (Wallace & Bush, 2001). The pattern-template pairs are powerful because of their quantity but produce many mistakes due to their limited knowledge, missing all benefits of linguistic and dialogue management research.

The second group are assistant conversational agents, which are part of many real-world applications today ((T. W. Bickmore & Cassell, 2000), (T. W. Bickmore & Picard, 2005), (T. Bickmore, 1999), (Kopp, Gesellensetter, Krämer, & Wachsmuth, 2005)). They populate virtual worlds and game environments, websites and software programs. In contrast to, e.g. automated banking hotlines, software assistants are often personalized. They may have a name and a voice, such as the Siri assistant, or an embodiment such as an avatar. Avatars are particularly common in website or software assistants such as the *Anna*[3] assistant on the Ikea website

[2] http://alicebot.blogspot.de/

[3] www.ikea.com

or the assistant paperclip *Clippy* in Microsoft Office 97-2003. In research, a new group of developed agents are so-called *relational agents*, which are, e.g., personalized health assistants (T. W. Bickmore & Picard, 2005). Relational agents aim at establishing a long-term relationship with the user.

Although conversational agents are not innately developed for small talk and may only contain a task-based dialogue system in the background, they are often subjected to social talk utterances. Embodied and personalized agents evoke particular behavior from human users, which also becomes noticeable in the language they use. Several studies such as the "Computers are social actors" paradigm (Nass, Steuer, & Tauber, 1994) and the "Threshold model of social influence" (Blascovich, 2002) have found that human users tend to communicate with such agents in a social way, especially if the agents possess human features such as embodiment or a human voice.

Some existing conversational agents already include modules, of varying sophistication, for handling some kind of open-domain small talk, often realized by integrating a chatbot. However, the integration of a chatbot can only be seen as a quick-and-dirty solution. Chatbots are separate components with nearly no intelligence. The integration of a traditional surface-pattern-matching chatbot in a dialogue system carries many drawbacks: The actual dialogue system loses the state and the control of the conversation, a shared memory is nearly impossible to integrate, the pattern matching may interact with the natural-language understanding component, and the chatbot will deliver useless answers which may affect the usability of the system. Some agent systems provide small-talk conversations based on other mechanisms than chatbots, but no systematic computational model of small talk has been developed so far. Instead, for example, systems reuse the same conversation sequence every time they engage in small talk (T. Bickmore, 1999). Social science theories offer descriptions of social talk, but only few concepts and ideas have been taken over to a systematic description of social talk conversations and in the development of dialogue systems. The reason could be that the theories are unable to provide usable communication patterns for social talk.

However, without any knowledge about social talk, conversational agents are not able to satisfactorily engage in a more realistic conversation and can not react to all potential user inputs, that might occur in a more conversational setting. Moreover, failed understanding is one of the main problems in dialogue systems confronted with social talk.

Understanding errors are problematic even for task-based systems, but the problem becomes particularly challenging in conversational dialogue systems which engage in social talk.

In general there are two main types of understanding problems: Non-understandings and misunderstandings. While the second one results from an erroneous interpretation process, that was not recognized and therefore incorrectly accepted by the machine, the first one occurs if no interpretation could be assigned to an incoming utterance at all. Failed interpretation may have several reasons. It occurs if the machine is not able to assign an interpretation result to an utterance it actually should understand, because something went wrong in the linguistic understanding process or in the ASR. Understanding errors also slip in if the utterance is not in the scope of the machine's knowledge. This is the case of *out-of-domain* utterances, i.e., utterances that are not related to the system's task or *knowledge domains*. Although conversational agent developers and theoretical work has found that out-of-domain utterances are one of the main reasons for failed understanding ((Bohus & Rudnicky, 2005), (Skantze, 2003)) and social talk is one of the main reasons for out-of-domain utterances, no systematic solution for these cases of understanding problems has been suggested so far.

This problem concerns dialogue systems without any social-talk knowledge which are unintentionally confronted with social talk, but also systems which already posses knowledge about social talk. Systems, that engage in social talk on their own initiative can be confronted with so many different utterances that anticipating them is not possible. Thus, social-talk systems are urgently forced to handle incoming out-of-domain utterances.

In summary one can say that social talk is a long neglected aspect in the development and research of dialogue systems. This thesis presents research in various important areas of dialogue systems, social talk and error handling which enables dialogue systems to engage in social talk beyond the existing unsatisfactory methods such as integration of an external chatbot component.

1.1 Thesis Goal

In this thesis I present research which focuses on several aspects of social talk in dialogue systems. The goal is to provide knowledge and technologies that can be used by dialogue system developers who want to

integrate social talk into their dialogue systems, without the disadvantages originating from chatbot integration. The presented insights and developed solutions are incorporated into a toolkit (described in section 1.3) providing components which can be used in dialogue systems either separately or in a bundle.

The thesis does not cover additional dialogue system research topics which may improve social talk such as emotions or affect but focuses on social conversation itself, namely what to say, how to say it and how to implement social talk.

Specifically, the thesis answers the following question:

What knowledge, structures and processes does a dialogue system need to take part in social talk as naturally as possible and how can these needs be technologically solved?

Figure 1.1 show the overview of components, subcomponents and research foci presented in this thesis. Related to the overall goal are several technology components which encapsulate knowledge and methods needed to implement social talk. The components are a social-talk component, a dialogue-act recognizer, a multi-threaded dialogue manager, a domain classifier, an uncertain answer module, and a topic recognizer. The components are bundled to the SOX toolkit.

The technologies incorporated into the SOX components are themselves the result of the conducted research or they are built upon the insights gained from research. The associated research areas are widespread. Research areas range from small talk over dialogue management to error handling (see also figure 1.2). Research in the field of social talk includes investigation of social dialogue acts, a model of social talk and dialogue-act recognition. In the area of dialogue management the thesis encapsulates research on thread-based dialogue management with graphs and multi-threading behavior. Research in the area of error handling contains exploration of domain classification, topic detection and new strategies applicable to understanding errors caused by out-of-domain utterances. Section 1.2 provides a more detailed description of the major research contributions.

Generally, social talk is built up of several characteristics. People who engage in social talk follow specific rules. Social talk is not, as it may appear from the outside, a completely unrestricted and uncontrollable behavior, although it is comparatively unrestricted in content. People engaging in social talk negotiate the social relationship between them

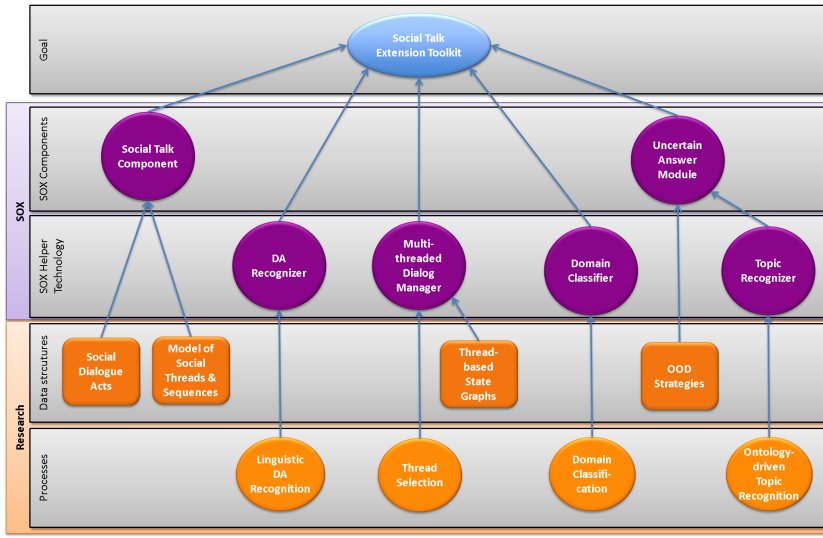

Figure 1.1: The SOX Components and Research Contributions

and are usually very sensitive to violation of the rules. Thus, the first research goal of this thesis is a deeper insight into the *abstract rules of social talk* proposed by empirical social science work and to propose a new set of social-talk communication patterns and a computational model of social talk[4].

The range of topics and content is definitely much more unrestricted in social talk than in task-based talk. This means, that social-talk utterances most probably will introduce topics and content that are not predictable beforehand and out of the known domains of a conversation. In the field of dialogue systems this leads to understanding errors. A major prerequisite for dialogue systems to enable social talk therefore is

[4]The author is aware of the fact that the way humans engage in social talk depends on many different factors, of which a very important one is certainly culture. The work described in this thesis investigates talk in western tradition. Most participants in the experiments that constitute the data basis described in the thesis were from western countries. The biggest group was from Germany. For work regarding conversational agents and culture, see, e.g., (Endrass, Rehm, & Andre, 2011).

an intelligent handling of these understanding errors caused by *out-of-domain utterances*. The second research goal of this thesis therefore lies in the field of error handling in dialogue systems, namely the handling of out-of-domain utterances. To improve error handling, this thesis presents a novel set of strategies mostly inspired by human-human communication originating from several studies.

The last research goal lies in the area of dialogue management and regards the support of multi-threading behavior for a graph-based dialogue manager. Different kinds of talks are often entangled; for example, a conversation could engage in small talk and task talk alternating over some period of time. People may, for example, talk about a task and, in a break, come up with talk about the weather. As the conversation progresses, turns containing utterances for both talks may alternate. This behavior is not limited to the combination of task talk and small talk and not in the number of different conversation threads either. The phenomenon may also occur with two different types of task talk and one small talk, or any other combination. However, the combination of one or more task talks and small talk is very common. A dialogue manager which aims at supporting social talk therefore needs to support multi-threading behavior.

Furthermore, the thesis aims to show how much the usability evaluation of conversational-agent applications depends on the integration of social abilities. The thesis wants to establish, how people using of conversational-agent applications perceive the integration of social talk on the whole, and especially the effects of single aspects of the suggested extensions.

1.2 Major Research Contributions

This section introduces the research contributions presented in this thesis. Generally speaking three main areas of research are addressed in this work:

Social Talk Social talk is the object of different research fields such as psychology and social science. Even work in conversation analysis and linguistics deals with *social conversation* or so-called *small talk*. Although the importance of social talk for dialogue systems is confirmed by many authors, the handling of social talk in dialogue systems is usually either neglected or solved ad-hoc. One of the

thesis' research goals is therefore to develop an abstract formalization of small-talk conversations and a computational model based on this formalization.

Error Handling Error handling in dialogue systems deals with the management of understanding errors caused by incoming utterances. One huge group of understanding errors are misunderstandings and non-understandings due to out-of-domain utterances. Particularly a system which evokes social talk either because of personalization or because it initiates social talk itself, will be frequently confronted with out-of-domain utterances. Hence, the second research focus of the thesis lies in the field of handling out-of-domain utterances in entertaining and conversational applications.

Dialogue Management Integration of social talk has several implications for dialogue management. It is impossible to know beforehand when users are going to start which kind of small talk. Moreover, social talk and other talk are often interwoven. Therefore, the thesis also presents a new approach to graph-based dialogue management which is based on conversation threads and enables multi-threading support.

Figure 1.2 shows the main research areas and their relation to the presented extensions.

The following sections give a more specific description of the research contributions described in the following chapters.

1.2.1 Dialogue Management

Thread-based State Graphs

The thesis suggests a structure of dialogue consisting of dialogue actions on the lowest level, dialogue sequences as groups of dialogue actions and *dialogue threads*, which encapsulate an arbitrary number of dialogue sequences. These threads are sub-phases of the core phase of conversation and group sequences that belong to a special conversational "goal". Threads are autonomous and provide a very modular description of dialogue structure. In this thesis threads constitute the most important unit for the graph-based dialogue model. Graph-based dialogue management is still frequently used as the backbone of dialogue systems in research and industry applications. Graphs provide many benefits such as easy development, including by non-experts. However they also carry

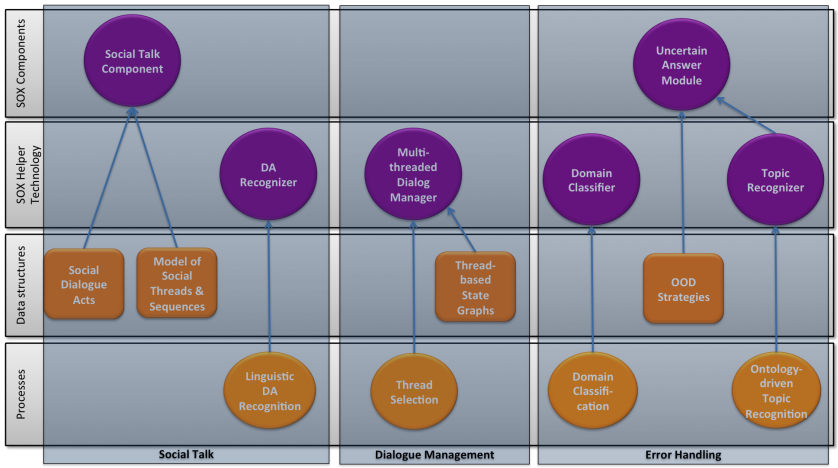

Figure 1.2: Main Research Areas and their Relation to the Social-Talk Extensions

several drawbacks such as a lack of flexibility. In particular, a transition of the nodes beyond the predefined order is not possible. If the developer wants a graph in which many different dialogue steps can be initialized at nearly every node in the graph, the graph is no longer readable by humans and cannot be produced by hand. These problems can be solved by a thread-based implementation, because the modular structure of the dialogue model adds the needed flexibility.

Thread Selection in Multi-threaded Environment

When people follow a path of conversation and then come up with a new topic or utterance, they introduce a new dialogue thread. This behavior can occur either after another thread was successfully closed or during an ongoing thread. In the latter case one can differentiate between embedded dialogue threads and interwoven dialogue threads. Whereas embedded dialogue threads are inserted into a mother thread as a completed unit, in an interwoven dialogue, two or more threads are alternated over a period of time.

Multi-threading therefore is a key competence for the management of dialogue threads: The dialogue manager must be able to keep track of interrupted task threads, handle unknown threads, and select appropriate threads to continue with.

Although multi-threaded dialogues are frequently initialized by humans in human-human interaction ((Shyrokov, Kun, & Heeman, 2007), (Yang, Heeman, & Kun, 2008)), support for multi-threaded conversations in dialogue systems is very rare. One example is (Lemon, Gruenstein, Battle, & Peters, 2002), who describes a way to integrate multi-threaded processing into an Information State Update model of dialogue management. However, Yang et al. (2008) criticize Lemon et al. (2002), because they neglect signals to indicate a conversation switch.

This thesis describes an approach which enables a thread-based dialogue manager to support embedding as well as interwoven behavior. In contrast to Lemon et al. (2002) it is not necessary to mirror the conversation structure in the components for multi-threading but the dialogue threads, which are subautomata of an overall conversation automaton, are the components of multi-threading themselves.

1.2.2 Social Talk

One of the two main research contributions regards social talk. The work done in the field of social talk presented in this work includes an empirical investigation and a theoretically grounded model of social talk, the development of an abstract set of dialogue acts for social talk, a set of dialogue threads for social talk, an annotation schema for the dialogue acts, and a computational small-talk model learned automatically from annotated data.

Social Dialogue Acts

Many dialogue act sets already exist in dialogue research. However, the social aspects of talk are not sufficiently represented in the existing sets. Therefore, this thesis presents an important research contribution: a new set of social dialogue acts, that can be used to annotate and implement social (also "small talk") conversations. The dialogue acts are ordered in a taxonomy which is based in the social-science theory of "face" by Erving Goffman (Goffman, 1967). Face means the perception of the self by the interactors participating in a conversation. Every person has an image of herself in social context. In direct communication participants' faces need to be negotiated. The two main classes of the taxonomy are

"request-face acts" and "support-face acts". Request-face acts express a request for the support of the talker's face, whereas support-face acts are for utterances that strengthen the listener's face. A dataset of small-talk conversations is annotated with the dialogue acts and inter- annotator agreement is calculated for evaluation.

Dialogue-Act Recognition

In the case of *dialogue-act recognition* the thesis provides a new solution that is based on a combination of semantic and syntactic relations, and machine learning. Although the importance of linguistic knowledge for dialogue-act recognition has already been noticed by several authors, e.g., (Jurafsky, Shriberg, Fox, & Curl, 1998), at the time when the dialogue-act recognition proposed in this work was being developed, dialogue-act recognition was nearly always based on words, n-grams of words, or words in combination with pronunciation or single linguistic cues such as the absence of a subject. Full syntactic relations had never been used. This has changed since then, partly because of the work described here, which was already in parts published in 2010, and it inspired other work to also integrate more linguistic knowledge in the recognition process.

Small Talk Communication Patterns and Model

Further research contributions in the area of social talk can be found in models of social talk. This includes an analysis of social-talk data resulting in a description of communication patterns for social-talk conversation. This structural representation of social talk is organized in dialogue threads. Moreover, an analogous computational model of social talk is learned from the data, which can be integrated into dialogue systems. This model is graph-based and encodes the knowledge about small-talk threads, sequences, and dialogue acts. The computational model is integrated into a software component that supports social talk and can be integrated into dialogue systems.

1.2.3 Error Handling

Dialogue Strategies for Out-of-Domain Errors

Another research contribution is in the field of error handling in dialogue systems, more precisely in the handling of understanding errors caused by out-of-domain (OoD) utterances. Out-of-domain utterances

(also "out-of-application utterances") as understood in this thesis are utterances targeting content that is outside of the knowledge domain of a system. All utterances which belong to these knowledge domains are in-domain utterances. All other utterances are out-of-domain. Out-of-domain errors necessarily result in system understanding errors, but handling these errors is very challenging. Common strategies used for error handling mostly try to repair the error. Others just ignore the input or confess the understanding problem ((San-Segundo, Pellom, Ward, & Pardo, 2000), (Komatani & Kawahara, 2000)). However, a repair is an inappropriate reaction to an error caused by an out-of-domain utterance and ignoring or confessing does not generate a conversational feeling. Initializing a repair strategy cannot lead to a solution, because the system will never be able to handle the out-of-domain input, no matter how it is expressed.

This thesis postulates a solution in which out-of-domain utterances are neither repaired nor ignored, nor answered by a chatbot, but actually answered by a set of new strategies. Around 25 different strategies for explicitly handling out-of-domain utterances in a conversational manner constitute the backbone of the solution. The strategies are taken from several different sources of human-human communication and are specified by linguistic information. The strategies are part of a computational tool, which can be used as an extension to dialogue systems. This represents a completely new approach to handling out-of-domain errors.

Domain Classification

Out-of-domain classification means the assignment of an incoming utterance to the class of in-domain utterances or to the group of utterances which are out of the domain of the system. OoD classification is a necessary preliminary step in the correct handling of understanding errors. Out-of-domain classification has already been done in some rare research ((Lane, Kawahara, Matsui, & Nakamura, 2007), (Fujita et al., 2011)). The approach used in this work is based on the comparison of the topic cloud of the incoming utterance and a bundle of different topic clouds learned from data annotated with topic and domain information. The OoD classifier is integrated into the computational tool which uses the dialogue strategies described above to handle out-of-domain utterances.

Topic Detection

In the field of *topic detection* the current state of the art is very vague. "Topic" itself is not consistently defined and although there is much work on topic detection for texts and paragraphs containing more than one utterance, such as the work done in the TDT Topic Detection and Tracking research program (Allan, 2002), topic detection for single utterances is quite uncommon and very challenging. This is due to the limited data that a single utterance produces. While a paragraph may easily contain around 100 words or more, a single utterance could consist of just three to ten words.

The approach used in this thesis is ontology driven. Through incorporating information about the syntactic structure of an incoming utterance, a topic cloud is generated by looking-up relevant concepts in WordNet and other domain-specific ontologies. These topic clouds are neither restricted in size nor the concepts they contain, and therefore represent a kind of vague representation of the topic of an utterance.

1.3 SOX: Social-Talk Extension to Dialogue Systems

The technologies described in this thesis are bundled into a toolkit for extending traditional dialogue systems to social talk called *SOX* (SOcial-talk eXtension). The goal of the SOX development is a toolkit which is easy to understand, usable for several kinds of dialogue systems, and straightforward to integrate into new applications, while providing enough interfaces on all necessary levels to enable communications with the mother system. SOX provides both pre-defined social-talk content and and a comfortable way to extend the system with more social talk content as desired by the developer of a particular system. Therefore, the toolkit provides a social-talk solution beyond the commonly used chatbot integration or other ad-hoc solutions.

SOX contains two main components and several helper modules and technologies (see figure 1.1). The two main components are the *small-talk module* and the *uncertain-answer module*. While the small-talk component can be integrated into a dialogue system to enable social talk based on empirical groundwork, the uncertain-answer module enables a system to handle understanding errors caused by out-of-domain utterances.

The toolkit also includes some subcomponents, which are either helper modules or part of the main components that are usable on their own: A *domain classifier*, which categorizes incoming user utterances

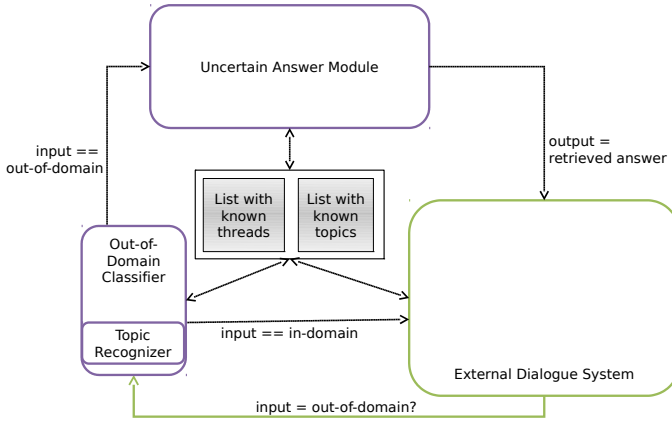

Figure 1.3: The Utilization of the Uncertain Answer SOX Component

to be in-domain or out-of-domain; a *topic detector*, which is integrated into the domain classifier but can also be used separately; a *dialogue-act recognizer*, which recognizes the dialogue act in the incoming utterances; and a *multi-threaded dialogue manager*, a graph-based dialogue manager that allows the execution of synchronously active dialogue threads.

Figures 1.3 to 1.4 show possible integration set-ups of the SOX components in a dialogue system. Figure 1.3 shows the integration of the *uncertain answer* error-handling tool. In the set-up shown, the tool is an external software component that takes input from a dialogue system and delivers an answer to the input back into the dialogue system. The other SOX component is the out-of-domain classifier, which incorporates the topic recognizer. External software such as the dialogue system are drawn in green; SOX components are presented in purple.

The full process starts with the detection of an understanding error in the external dialogue system. The dialogue system delivers the input that is responsible for the error to the domain classifier. The domain classifier is trained to detect if an incoming utterance is out-of-domain or in-domain. The basis for this decision is formed by the lists with safe topics and safe threads, which are shared between the dialogue system and the SOX tools. If the classification decides the input is in-domain, the initiative is given back to the dialogue system, which should react in a manner appropriate to the task and the goals of the overall system. The system could, e.g., ask for a rephrasing of the input. If the input is classified to be out-of-domain by the domain classifier, the input is delivered to the uncertain-answer module. This module then calculates a response to the input and delivers it to the dialogue system, which can than decide how to deal with the answer.

Figure 1.4 shows a possible utilization of the second main SOX tool, the small-talk module. The small-talk module contains knowledge about when to say what in a social talk. The external dialogue system knows when to say what in a task-based conversation. Again, SOX components are given in purple, whereas external components are green.

A typical process is that the dialogue system wants to know the dialogue act belonging to an incoming utterance. The SOX dialogue-act recognizer, which incorporates the topic recognizer, is asked. The SOX dialogue-act recognizer can detect social-talk dialogue acts. The result of the dialogue act recognizer is delivered to the SOX multi-threading component. The multi-threading component can be seen as a hub between the two different dialogue competence centers, the dialogue system for task, and the small-talk component for social talk. If the input to the dialogue system is classified by the dialogue act recognizer to primarily fulfill a task purpose, the input and the dialogue act found are given back to the dialogue system, which can deal with the interpretation in a task-appropriate manner. If the input is classified to primarily fulfill social-talk purposes, the input and the interpretation are given to the small-talk component, which handles the input in a small-talk-specific way. The behavior is not strictly turn-based. The dialogue system and the small-talk component may decide that several turns are necessary to finish the active part of the conversation in a satisfying way. If further input from the user interrupts the processing, the input is delivered to the dialogue-act recognizer which starts the process again. If no further user input appears then both conversation competence centers give the initiative back to the multi-threading component.

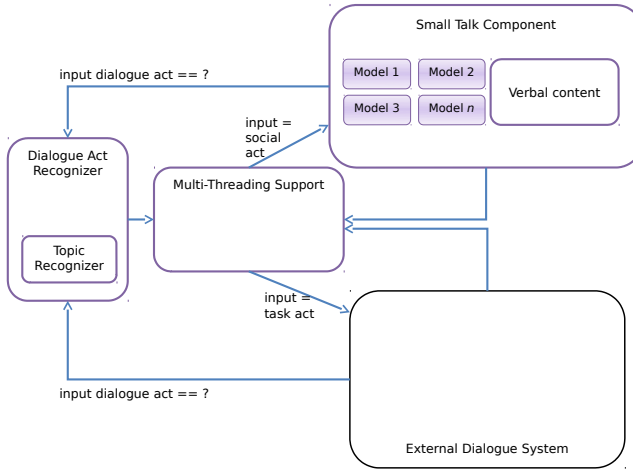

Figure 1.4: The Utilization of the Small-Talk SOX Component

The second possible way to use the small-talk module is to integrate the single models in the dialogue system's dialogue manager (see figure 1.5). This method of integration is used in the *KomParse* dialogue system described in chapter 3, which is the test bed for the solutions presented here. Integrating the models into the dialogue system is only possible if the system can understand the formalization of the models and interpret them on its own. The conversation structures in the dialogue manager should optimally be encoded in the same way as the small-talk models. In the case of *KomParse* both are encoded as graphs. The multi-threading component in the integrated scenario is part of the dialogue manager itself and can select specific models and task graphs according to the information found by the dialogue-act recognizer. The integrated scenario has the advantage that sharing other knowledge such as linguistic information about anaphora referents and so on between the dialogue system and the small-talk component is much easier.

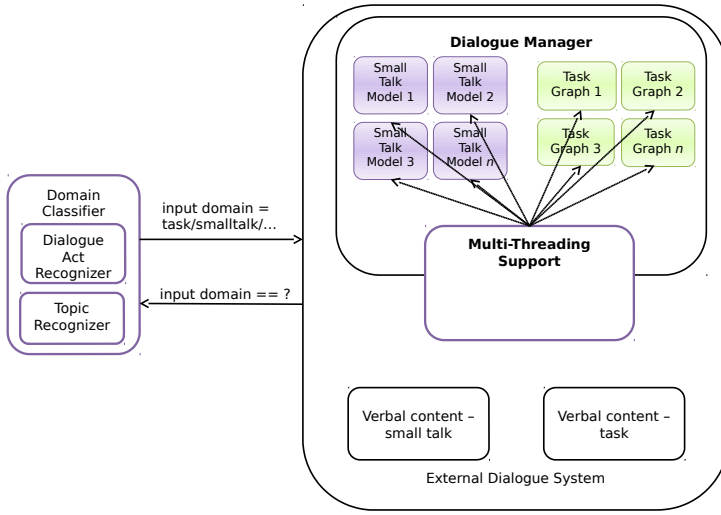

Figure 1.5: The Integration of the Small-Talk SOX Component

1.4 Research Project Context and Support

The main parts of the research presented in this thesis were investigated alognside the research project *KomParse*, carried out at the Language Technology Department of the German Research Center for Artificial Intelligence (DFKI). The KomParse project was funded by the ProFIT program of the Federal State of Berlin, cofounded by the European Union's EFRE program, running from April 2008 till September 2011. The aim of the KomParse project was the development of conversational embodied agents, so-called non-player characters (NPCs) in a virtual, 3D-world environment called *Twinity*. The NPCs should be equipped with limited, but adequate and robust natural-language capabilities, keeping in mind the real-time requirements of the interactive application. The toolkit SOX was successfully applied to the KomParse

dialogue system. Moreover, important information was gathered through the empirical groundwork carried out in KomParse. Chapter 3 introduces the technology, data and methods used in KomParse. Along with the DFKI, the *Center for General Linguistics (ZAS)* in Berlin was a participating partner of the project. KomParse was supported by the company *Metaversum*, the operator of the Twinity game, and the *Game Academy*, Berlin. More information about KomParse can be found on the project website at http://komparse.dfki.de. Smaller parts of the research presented here were supported by the research project *Sprinter*[5]. The *Sprinter* project, running from July 2012 till July 2014, deals with the integration of language technology into web-based language learning software. The project consortium consists of the Language Technology Department of the DFKI as well as the company *LinguaTV* in Berlin, Germany. The project is funded by the German Federal Ministry of Education and Research (BMBF) under grant number 01IS12006A.

1.5 Thesis Structure

Following is an outline of the thesis structure. In the next chapter (chapter 2) an introduction into dialogue systems including a definition of dialogue is given. The chapter explains the important terms used throughout the thesis and the baseline knowledge about dialogue systems, their architectures and challenges.

Chapter 3 describes the application *KomParse*, a dialogue system controlling a virtual agent in an online game world. The application is the test bed for the knowledge and computational tools developed in this thesis. This chapter explains the agent's architecture, the virtual world, and the data which was collected using the application and which is used for evaluation.

In chapter 4 the work related to the research goals achieved in this thesis is presented. This includes the state-of-the-art in relevant approaches to dialogue management, error handling, natural-language understanding and social talk.

The chapter 5 introduces and evaluates several new approaches to natural-language understanding challenges, namely dialogue-act recognition, topic detection, and out-of-domain classification. The presented approaches constitute important parts of the overall thesis research goals, which are dealt with in the next chapters.

[5]http://sprinter.dfki.de

Chapter 6 deals with the new approach to thread-based dialogue management presented in this thesis. The chapter includes a description of graph-based conversation-thread modeling as well as a dialogue manager, which uses the models and supports flexible multi-threading. The chapter closes with an evaluation of the dialogue manager's multi-threading behavior.

In chapter 7 the work on social talk is presented. The chapter starts with the description of the new dialogue-act taxonomy for social talk. Afterwards an analysis of social-talk data and common communication patterns for social talk organized in threads extracted from the data are presented. The chapter then describes a computational model of social talk that was learned from annotated data and covers the communication patterns found, and its integration into dialogue systems.

Chapter 8 presents work on handling of out-of-domain (OoD) utterances in dialogue systems. The chapter describes a novel set of 25 dialogue strategies which are used to react to OoD utterances. It also explains the sources which were exploited to generate the strategy set. Lastly, the chapter describes a computational tool that uses the new strategies to handle out-of-domain utterances.

An evaluation is presented in chapter 9. The chapter contains two parts. In the first part, a usability evaluation of the test bed agent architecture is described. In the second part another subjective user-oriented usability study is presented, which focuses on the new components described in this thesis. The second evaluation compares a baseline dialogue system with several set-ups of the same dialogue system including the new components. Both evaluations show that the social-talk extensions significantly improve the user's perception of the system.

In chapter 10, the thesis conclusion is presented.

1.6 Contributions to Literature

Partial results of this thesis were already published in conference and workshop proceedings as well as book chapters during writing. These publications to some extent have the same wording as the text in this thesis. More precisely, the described work has contributed to the following publications: (Klüwer, Adolphs, Xu, & Uszkoreit, 2012), (Adolphs et al., 2011), (Klüwer, Adolphs, Xu, & Uszkoreit, 2011), (Klüwer, 2011b), (Klüwer, Uszkoreit, & Xu, 2010), (Adolphs, Cheng, Klüwer, Uszkoreit, & Xu, 2010), (Klüwer, Adolphs, Xu, Uszkoreit, & Cheng, 2010). Some

of these publications were partly written with others, as indicated in the following list.

- Klüwer et al. (2011) describe the *KomParse* dialogue system and application, which is the test bed for the extensions presented in this thesis. The description of the KomParse system in chapter 3 is partly identical with this publication.

- Klüwer (2011b) gives an introduction to dialogue systems, and describes the difference between dialogue systems and chatbot technology. Parts of the book chapter are identical with the description of dialogue systems in chapter 2.

- Klüwer (2011b) describes the set of dialogue acts for social talk, the annotation of a corpus as well as the inter-annotator agreement rates, and some first social sequences extracted from annotated data. This is a part of the work presented in chapter 7.

- Klüwer, Uszkoreit, and Xu (2010) describe the mechanism used for dialogue-act recognition. The paper presents the information extraction methods used to get the syntactic and semantic relations, the recognition process itself, and the evaluation of the recognition accuracy on Wizard-of-Oz data. An expansion of this work is shown in chapter 5.

- Klüwer (2012) describes the extension for a finite-state-based dialogue manager to enable support for multi-threaded conversations. The paper provides information and an evaluation about the thread-selection algorithm described in chapter 6.

- Klüwer et al. (2012) present the usability evaluation of the agent application which is the test bed and embedding framework for the presented thesis. The usability evaluation is part of chapter 9.

2 Dialogue Systems

2.1 Introduction

If a machine is involved in a dialogue, this machine possesses dialogue capabilities provided by an underlying dialogue system. The style of a conversation with a dialogue system can be very different. A user and an electronic device can for example engage in a dialogue through a given menu in which the user can select what steps to carry out next by clicking buttons, such as a software installation process. This chapter and book focuses on dialogues mediated through natural language.

There are a lot of different aspects which differentiate natural-language dialogue systems. One important characteristic is the modality of the system. Traditionally, dialogue systems are understood as *spoken dialogue systems*, which means the systems can understand speech input and deliver spoken output back to the user. Spoken dialogue systems have their origin in telephone software, as it is used in call centers or for information hotlines ((J. F. Allen, Ferguson, Miller, Ringger, & Sikorski, 2000), (Peckham, 1993)). Wahlster (2000) shows the use of spoken dialogue technology embedded into a machine translation system. In recent years these systems are being increasingly replaced by *multi-modal systems*, integrating more sensory information such as gesture identification and recognition of facial and/or body expressions (Wahlster, 2006). In addition, a lot of pure *text-based systems* are used, especially for applications in text-based environments such as the world wide web. Multi-modal systems need additional modules which are able to handle the actual input (e.g., a camera for gestures and a component translating the images into a meaning) and combine different types of input to a merged representation. Furthermore, they need possibilities to plan and physically execute multi-modal output.

Additionally, a dialogue system can be described by the mode of initiative the system supports: Dialogues with the system are either

completely system-driven, which means that the system always controls the flow of the dialogue, leading the user through the conversation. These systems are very robust, because no unpredicted conversation states can occur. If the user is also able to take the initiative and determine the topic and direction of the conversation, the system is mixed-initiative. Finally, systems restricted to reactions are called "user-initiative". An example of a user-initiative system would be a simple question-answering machine.

An ongoing research focus is the group of dialogue systems incorporating models of emotions, affect and non-verbal interactions. This is important for dialogue systems embedded in embodied conversational agents: Their embodiment allows them to engage in non-verbal communication with users, using gaze, gestures, body movements, and more. One big research topic, for example, is the natural timing and expression of conversation feedback (Poppe, Truong, & Heylen, 2011).

Dialogue systems are also frequently classified according to their technological architecture and the integrated components. In particular, they are often clustered by means of their dialogue model and the dialogue-management component, the central component which controls the dialogue flow, and the execution of the system actions.

The huge number of different dialogue-system architectures makes comparison very complicated. For a better understanding, this chapter explains architectures and basic components of a dialogue system in section 2.4.

2.2 Dialogue Definition & Structure

Dialogue can be defined as a conversation between two or more participants or "agents", making use of at least one change of speaker. In pragmatics, dialogue is seen as one of two subgroups of discourse, whereof the other one is monologues, meaning most notably text. On the most abstract level a dialogue is a sequence of "dialogue turns" originating from different participants. A turn is an interval of expression by a single participant. A turn begins with the speaker getting the possibility to talk ("taking the turn") and ends when the speaker makes it possible for someone else to talk. A turn may contain one or more "dialogue utterances". Utterances are not necessarily sentences, since one sentence may contain several dialogue utterances. Also, turns and utterances usually correlate but do not necessarily have to. A turn can encapsulate several utterances.

Utterances in a dialogue are produced to achieve something. They are not just spoken entities but spoken actions. These actions can be described from several view points, most notably the informational or the intentional/functional aspect. The most common way to describe dialogue actions originates from the field of speech act theory ((Austin, 1975), (Searle, 1969)). Speech acts are descriptions of the intentions encapsulated in utterances and therefore focus on the functional level of dialogue. Speech acts encapsulate an important aspect of conversation since people react to the understood intention of a speaker, not necessarily only to the semantic content. If dialogue systems want to appropriately react to what a user has said, it is crucial to react not only to the informational content the user uttered but especially to what the user intended. The intention is not necessarily observable from the input's surface. Consider the following examples:

(2.1) Can you show me a red car please?
Show me a red car!

The intention behind the two utterances in the example may be the same: The speaker wants the hearer to show a red car. While this is straightforward in the second sentence, a system may understand the first one as a real question regarding the system's ability to be able to show a red car and answer with "yes" or "no". A dialogue-act recognition embedded in the system might detect that both utterances yield the same type of dialogue act denoting a request from the speaker. Therefore, it is commonly accepted to use speech acts to annotate dialogue data as well as to develop dialogue models for dialogue systems. Further research has modified and expanded the set and the characteristics of speech acts, which are now known under the names "dialogue acts", "conversation acts", "dialogue moves", and many more.

Another possibility for dialogue description originates from text analytics. Examples are the Discourse Representation Theory (DRT) ((Kamp, 1981), (Kamp & Reyle, 1993)), Rhetorical Structure Theory (RST) (Mann & Thompson, 1988), Linguistic Discourse Model (LDM) (Polanyi, 1996) and the Penn Discourse Treebank (PDTB) (Prasad et al., 2008). These theories and annotation schemata describe the semantic content of a discourse and the content-oriented relations between its elements, and do not focus on the intentional and functional aspect of discourse. DRT for example uses a representation language similar to first-order logic to describe sentences according to their semantic truth

value. These representations are then extended to a discourse representation by incorporating context. The Rhetorical Structure Theory focuses on the discourse relations between parts of a discourse. In text, these parts can be sentences or paragraphs. In general the theory differentiates between satellite and nucleus elements. The nucleus text span is considered to be more essential to the text. RST suggests but is not limited to a rich set of possible relations between nucleus and satellite text spans such as "condition" or "elaboration". The PDTB has the same focus: The treebank is a large corpus annotated with discourse relations. These approaches to describing discourse structure can be seen to focus on the informational and semantic content of a discourse and the - mainly informational - relations between single elements of discourse. In general, they are not easy to apply to dialogue instead of text. One question which arises is: What are appropriate baseline elements for description? Are discourse relations possible only within utterances by a participant, between utterances of the same turn, or even between utterances from different users? Another problem is that the existing sets of more informational oriented discourse relations are not always sufficient for dialogue interaction as shown in the following example taken from Stent (2000):

(2.2) **A1:** "Then they're going to have basically wait"
B1: "Why?"
A2: "Because the roads have to be fixed before electrical lines can be fixed."

The example shows a "motivation" relation between utterance A1 and utterance A2 on the informational level. However, it also contains a "functional" question-answer relation typical for dialogues between utterance B1 and A2. Without the utterance B1, utterance A2 and the motivation relation might never have occurred. Approaches exist to use text analytics theories to annotate dialogue data ((Stent, 2000), (Tonelli, Riccardi, Prasad, & Joshi, 2010)). However, these approaches suggest several changes to be appropriate for dialogues, e.g., to the set of possible relations. The question arises if these solutions not mix the informational level and the functional level of dialogue structure while they would best be kept apart.

Therefore, in this thesis the intentional view on dialogue is used for structure description and model development. Dialogue acts for describing single dialogue actions are the basis for a further taxonomy of dialogue structure-constituting elements. Actions can be further organized into "sequences". The most popular example of a dialogue sequence is the "adjacency pair" (Schegloff & Sacks, 1973). An adjacency pair is the combination of two actions adjacently placed of which the second is enforced by the first. Both actions need each other. Examples are the pairs "question - answer", "greeting - greeting" or "offer - accept/decline". Other sequences can be, for example, the necessary succession of several actions to fulfill a sub-task.

At the highest level of a dialogue structure are the "conversation phases". In general, conversations consist of just a few conversation phases such as an opening phase, closing phase, and one core phase, e.g., for one special task. The internal structure of a core phase is complicated to define and not completely investigated in existing research. One way to further structure the core phase focuses on the functional action-oriented aspect of the conversation. Most researchers agree on the opinion that the internal structure of core phases depends to some extent on the goal of the conversation. Common units constituting sub structures of core phases are therefore task-related "sub-phases". Commonly used names for these sub-phases are "sub-goals""sub-tasks" or "plans". These units contain one or more dialogue sequences themselves. Another possibility to define sub-entities in a core phase is according to the "topical organization" of the dialogue (G. Schank, 1981). However, the notion of "topic" in dialogue itself is very vague and the identification of a topical structure is determined by subjective perception.

In this thesis an action-oriented terminology with the following dialogue-constituting elements is used:

Dialogue Acts Dialogue acts are used to describe the smallest structure-constituting elements in dialogue, usually utterances, focusing on the functional and intentional aspect of an utterance (the action) not on the semantic content.

Dialogue Sequences Dialogue sequences are a succession of dialogue acts from one initiative turn to the next initiative turn (Brinker & Sager, 1989). Sequences can consist of two dialogue acts (adjacency pairs), but can also exceed this number. A sequence can be described by the initial dialogue act. An example is a compliment followed by a thank you, which is followed by a reassurance act.

Dialogue Threads Dialogue threads are containers for one or several dialogue sequences which are grouped according to a specific functional goal. A dialogue thread can be to make a compliment, for example, which may consist of just one sequence, or to negotiate an object, which consists of several different sequences.

Table 2.1 shows an overview of typical conversation-constituting elements from literature and the terminology used in this thesis. For comparison the elements of the HCRC annotation schema are given, too. The HCRC annotation schema (Carletta & Isard, 1996) was initially developed to annotate a set of map-task conversations. It is the baseline for many other annotation schemata (Savy, 2010) and inspired by the observations by (Sinclair & Coulthard, 1975) who analyzed the "organization of linguistic units above the rank of clause" in class room conversations.

2.3 Computational Dialogue Models

Dialogue models describe the structure of dialogues in a dialogue system and are used to calculate how the system should act next. Dialogue models encode the elements of dialogue described in section 2.2 such as dialogue actions, dialogue sequences and dialogue phases. However, they do not need to describe and model them explicitly. While structured dialogue models may explicitly describe all different levels of structure, other models such as probabilistic models could just implicitly cover dialogue structure without any description offering a black-box model of dialogue.

For formalizing a model of dialogue, a decision has to be made about what kind of description language should be used in the model.

General	HCRC	Mine
Phase	-	-
Sub-phase	Transactions	Threads
Sequences	Conversational Games	Sequences
Actions	Dialogue Acts	Moves

Table 2.1: Overview of Terminology for Dialogue-Constituting Elements

The following list describes three very common ways to formalize dialogue models in dialogue systems:

Graph Based The dialogue is modeled by a state graph. The graph encodes typical dialogue states and possible transitions between them via edges. Edges can for example be of conditional type, which means that they can get traversed if the condition they are representing becomes true only. A possible condition could be that the system received an input from the user, or that a special meaning was contained in the input. A simple finite state graph has to be encoded beforehand. Every possible progress of a conversation is pre-encoded and the resulting dialogue may lack flexibility. On the other hand, dialogue modeling through finite state graphs is very robust (see Cohen (1997) for more information). Graphs belong to the group of structured models, which means that the structure of possible dialogues encoded in the model is known and recognizable in the graph.

Frame Based The dialogue is controlled by a hidden electronic form, collecting information from the user ((Aust, Oerder, Seide, & Steinbiss, 1995), (Constantinides, Hansma, Tchou, Rudnicky, & Rudnicky, 1998), (Klüwer, Adolphs, et al., 2010)). An example application could be a travel-support hotline, delivering information about train schedules to a user. One frame would be a "Train-Travel"-Frame with field for "origin", "destination", "date" and other information the system needs from the user to fulfill its task. At the beginning the system does not possess any information about the user's wishes, so all information slots are empty. During the conversation the system tries to get the missing information from the user. Every time the user provides missing information, it gets stored in the internal form. What the system asks or does is therefore led by the empty or filled slots in the form. The actions are not hard-coded, but the system's behavior depends on the form and may differ from use to use. Therefore, this approach provides much more flexibility than the graph-based technology. It is often combined with finite-state graphs.

Plan Based In contrast to the above mentioned methods the plan-based approach is very flexible and supports a greater complexity of conversations ((Lesh, Marks, Rich, & Sidner, 2004), (Rich & Sidner, 1998)). The plan-based approach originates from AI research

on planning methods and involves the detection of the plans, beliefs and desires of the users. These are then incorporated into rich descriptions, which can be used for further reasoning (see also BDI agents ((J. Allen & Perrault, 1980), (Bratman, Israel, & Pollack, 1988)). Because of the multiple reasoning steps, the rich plan-based approaches are nearly impossible to use in real-world applications without integrating further ways to reduce the search space for the next action.

Probabilistic Models In the last few years the focus in dialogue management and dialogue models moved to probabilistic models of dialogue. A very common example is POMDPs, Partially Observable Markov Decision Processes (Young, Gasic, Thomson, & Williams, 2013). In this approach a dialogue state is encoded as a belief state containing a probability distribution across all states. The selection of the next best dialogue state is based on this probability accounting for all states. The best action to execute in a selected dialogue state is calculated using a policy. The policy model is trained by a reward function which assigns a reward to every system decision. For successful performance the system gets higher rewards. The structure of possible dialogues is not encoded explicitly in POMDP approaches, but inherently concealed in the models.

2.4 Dialogue System Architectures

A basic dialogue-system architecture is shown in figure 2.1. It contains a processing chain from a user, or environment input to a system answer. The following sections focus on the architecture of dialogue systems and their main components: natural-language understanding, dialogue management and output generation.

2.4.1 Dialogue Management

A *dialogue manager* is the component that decides what the system should do next. This can include various different actions depending on the application scenario and the business domain of the actual system. The dialogue manager is also responsible for triggering all necessary supplemental steps such as embedded reasoning. Moreover, it is the component which has access to the dialogue context, the current dialogue state and additional internal and external knowledge bases.

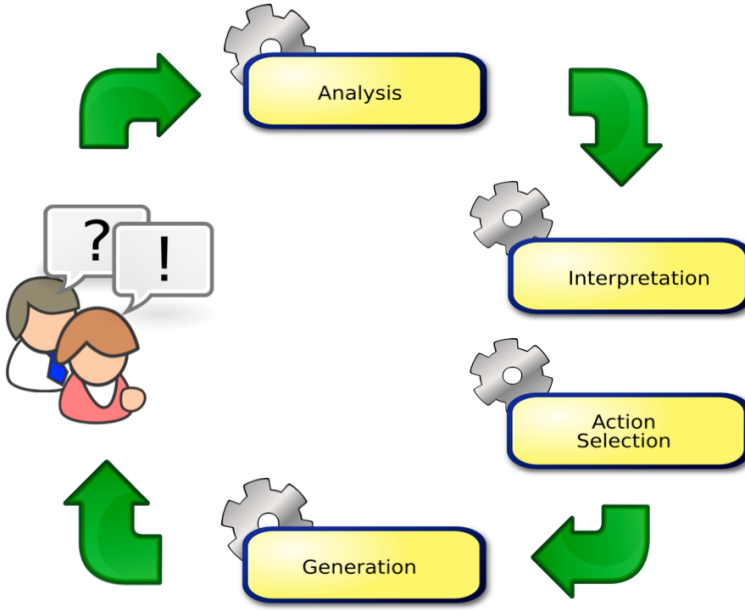

Figure 2.1: A Baseline Dialogue System

Thus, the dialogue manager needs to detect the next system action(s) according to the possibilities the system has in the current context of the conversation. The heart of the dialogue manager is the dialogue model. The dialogue model describes possible and necessary conversation steps. There are many different possibilities to encode a dialogue model (see section 2.3). One possibility is using structured dialogue models such as graphs in which the possible flow of a conversation is encoded beforehand. Another possibility is using plan-based models, in which the dialogue structure is encoded by modular plans and the dialogue manager is responsible for calculating the actual dialogue actions from the active plans.

In many systems, an explicit task model is used in addition. Task models can also be included in the dialogue model (implicit task model), but an explicit task model gives more flexibility to the system's author.

Task models contain knowledge about necessary steps or needed knowledge to fulfill a special task. These can then be translated by the dialogue manager to actual dialogue actions.

Apart from deciding what action fits best to the actual state and triggering all the necessary sub-tasks (such as reasoning or a simple database query), the dialogue management component often does the *interpretation* of incoming input. Since the dialogue manager is the main component with access to the conversation context, the input-analysis component may deliver a semantic representation covering the meaning of a single utterance and the dialogue manager tries to embed this representation into the context. A concrete example could be a mapping of a semantic description or a surface dialogue act to a context-aware dialogue act. The dialogue manager may also deliver a dialogue act as a description of the next action to the output generator, which then generates an output from this description.

2.4.2 Input Analysis

It is the *input-analysis* component's task to deliver a representation of the user's input which can be used by the other components of the system. This internal representation depends on the type of input the user enters into the system, e.g., typed text or an interim representation merging the results of different input components.

The "understanding" of a user's utterance is a highly complicated task due to the properties of natural language having unrestricted combinations, and problems such as ambiguity. Ambiguity can occur on various levels of language: Words can have several meanings and belong to more than one part of speech, and pieces of utterances as well as whole sentences can be ambiguous in structure and meaning. Furthermore, the input analysis of a dialogue system not only has to deal with problems occurring from word to sentence level, but also with all context phenomena appearing in natural language. Typical context phenomena that a discourse processing system such as a dialogue system has to deal with are anaphoric expressions (e.g., the resolution of what is denoted with personal pronouns or definite nominal phrases), implicatures and challenges originating from the interaction character of a discourse (e.g., beliefs and interests of the participants).

A common way to encode an internal representation of the user's input is a kind of semantic representation (J. Allen, Manshadi, Dzikovska, & Swift, 2007). A semantic representation of a user's input is generated

by adding linguistic information. The process of retrieving linguistic information typically contains several steps of linguistic analysis, proceeding from word level to the whole utterance, in which every element can be seen to deliver a more abstract representation of the input. In nearly all systems working with free user input this component includes work on input cleaning, because a huge amount of given user input is ill-formed, containing for example typing errors or ungrammatical combinations.

A traditional linguistic analysis pipeline could look like the following:

Processing Step	Tasks
Input Cleaning	A typical first step is the preprocessing of the input, which can involve various steps of input cleaning such as acronym resolution, smiley transformation, spell checking and deletion or substitution of unwanted characters. A speech-based system would most probably integrate a preprocessing of the speech input containing such as removing unwanted noise.
Segmentation	Segmentation of the user input in single "utterances."
Word-level Analysis	Processing of the words part of speech and morphological information, named entity recognition (NER).
Syntactic Analysis	Detection of the constituents in an utterance and the relations between these constituents using knowledge about the structural characteristics of a language.
Semantic Analysis	Transfer of the given information into a description of the meaning of the utterance.

Table 2.2: Linguistic Analysis Pipeline

Table 2.2 shows a schematic overview about a possible pipeline architecture, in which every module may consume the results of the preceding components. In modern systems there are various components encapsulating multiple steps, for example a grammar can directly produce semantic output. Moreover, there are several tasks not mentioned here and a dialogue system may not necessarily pass every step of a processing pipeline, but use one of the lower level descriptions, e.g., the results of a

syntactic analysis directly. The exact results of the single linguistic components and the overall linguistic input processing depend on the actual dialogue system. Some systems integrate very sophisticated syntactic analysis using a manually developed grammar or even rich semantic representations, whereas others get along with surface regular expression search or keyword spotting.

2.4.3 Output Generation

After the system has decided which action should be executed, the output generator component has to construct a physical message encapsulating this information. The message may be implemented as speech- or text-based output, or in graphical form. If the system has a graphical interface at its disposal, for information such as a long list of options, showing a table or a map may be the most suitable response. In multimodal systems, e.g., of an embodied agent, the output can also include gestures or other movements.

Common methods to generate output are:

Pre-stored Text The easiest way to generate output is the selection of an appropriate canned-text snippet. This approach is comparable to the simple surface templates used in chatbots.

Template Filling To gain some more flexibility, pre-encoded text snippets might contain variable slots in which different content can be inserted dynamically. These templates can produce one and the same output with slight variations and therefore provide more flexibility to the system. This method is equivalent to the more complex chatbot templates using variables.

Language Generation In sophisticated research and industrial dialogue systems, the output is planned at an abstracted level by the dialogue manager and processed via a language-generation pipeline similar to the language-understanding pipeline (see Kruijff et al. (2010), Reiter and Dale (1997)). The dialogue component could decide on how to act on the intentional level, for example, to react to a "REQUEST" dialogue act with a "PROVIDE_INFO" dialogue act and the appropriate information. The generation unit then has to calculate a possible semantic structure from this specification, which in turn can be transferred into multiple possible syntactic structures and, finally, surface structures. It is clear that this better protects the system from being repetitive, because one

meaning may result in several surface structures. Moreover through language generation it is possible to insert anaphoric references such as personal pronouns or deictic expressions such as "here" and "there" to further enhance the natural effect.

2.5 Error Handling

Error Handling is a very important topic in dialogue system research and development. In every interaction miscommunication can occur and especially in human-computer interaction mechanisms for detecting and dealing with miscommunication errors are necessary. Several studies have supported the hypothesis that failures in communication significantly decrease task success and user satisfaction ((Bohus & Rudnicky, 2008), (Bohus & Rudnicky, 2005)).

There are two main classes of miscommunication errors in dialogue systems: non-understandings and misunderstandings. While the second one contains all erroneous interpretation results for the incoming utterances, non-understandings subsumes all cases in which no interpretation could be assigned to an incoming utterance at all. Failed interpretation may have several reasons. It occurs if the machine is not able to assign a dialogue act to an utterance it actually should understand, because something went wrong in the linguistic understanding process or in the ASR. Non-understandings can also result from incoming utterances which are not in the scope of the machine's knowledge. This is the case for out-of-domain utterances, i.e., utterances that are not related to the system's task or knowledge domains.

In general there are two possibilities to deal with these communication errors: The machine can try to prevent the errors from happening or try to recover from the errors through conversation. Because the first one is nearly impossible, dialogue system developers usually settle on the second. Various dialogue systems possess recovery strategies to repair an error in communication with the user, e.g., the RavenClaw system uses several recovery policies in the overall plan-based system (Bohus, 2007) and Lee, Jung, Lee, and Lee (2007) describe some example-based error recovery strategies for integration in an example-based dialogue system.

Regarding misunderstandings, the crucial part is the detection of the misunderstanding, whereas the recovery from misunderstandings has received a lot of attention and already works very well. Most systems

use confidence scores to detect misunderstandings and employ several strategies such as the well-known *implicit* or *explicit confirmations*.

This is different for non-understandings: While the detection of non-understandings is basically straightforward, dealing with non-understandings is much more complex. The system automatically knows if an interpretation process was unsuccessful, but the strategies to repair non-understandings are not so well understood. Many systems use simple heuristics in combination with a handful of recovery strategies such as "Asking the user to repeat", "Asking the user to rephrase" or "Telling the user that the input was not understood" ((Bohus, 2007), (Jokinen & McTear, 2009)).

An empirical study (Bohus & Rudnicky, 2005) comes to the result that beside errors originating from the ASR, the next huge reason for non-understandings are "out-of-grammar" and "out-of-application" utterances[1]. But especially for these utterances, the described error recovery strategies are not recommendable. The system will never be able to understand an out-of-domain utterance, not even if it asks for repetition or rephrasing.

2.6 Conclusion

This chapter provides an introduction to dialogue systems including related fields which are important to understand dialogue and dialogue systems. The chapter includes a theoretical definition of dialogue, ways to describe dialogue structure, possibilities to computationally encode dialogue structure and dialogue systems' architectures, as well as methods for error handling in dialogue systems.

The chapter first gives a definition of dialogue and dialogue structure (section 2.2). Dialogue is seen as one of two subgroups of discourse, whereof the other one is monologues, meaning most notably text. We have seen that two main possibilities exist to describe the structure of dialogue, the intentional one and the informational one. The intentional view on dialogue is more common and originates from speech act theory ((Austin, 1975), (Searle, 1969)). It focuses on the intention contained in speaker utterances. The informational view on dialogue structures conversations according to the semantic content of the utterances

[1]The authors of (Bohus & Rudnicky, 2005) use the notion "out-of-application" to denote a group of utterances which are mainly in-domain but out of the scope of possible functions of the system. In the following work all utterances which are out of the scope of the machine's knowledge are called "out-of-domain utterances".

and originates from text analytics. Well-known theories are, for example, Discourse Representation Theory (DRT) ((Kamp, 1981), (Kamp & Reyle, 1993)) and Rhetorical Structure Theory (RST) (Mann & Thompson, 1988). However, in section 2.2 is shown that these theories are complicated to apply to dialogues. The chapter therefore favors the intentional way to describe dialogue structure and to develop dialogue models for computational use. The chapter suggests a taxonomy of dialogue structure-constituting elements made of dialogue acts, dialogue sequences and dialogue threads. Whereas dialogue acts are used to describe single conversational actions, dialogue sequences are successions of dialogue acts and dialogue threads are container, which can group various dialogue sequences belonging to one conversational goal.

The chapter also gives an overview of possibilities to encode dialogue models for computational use. Dialogue models can be structured or unstructured models. Section 2.3 explains common ways to encode dialogue models such as finite-state graphs, electronic frames, plan-based approaches and probabilistic models. Dialogue models are the backbone of dialogue systems, used in the dialogue manager component of a dialogue system, and encode the conversation flow known to the system.

The chapter proceeds with the description of possible and common architectures of dialogue systems in section 2.4. The main components of dialogue systems are a component for natural-language understanding, a dialogue manager and a reaction-generation component. The natural-language understanding component is responsible for analyzing incoming user utterances. Section 2.4.2 gives an introduction in natural-language understanding and constituting subtasks. Section 2.4.1 describes the tasks of the dialogue manager, the backbone of a dialogue system, and possible approaches to dialogue management. Section 2.4.3 presents common ways to realize verbal answers which constitute the dialogue system's reactions. The chapter shows that there are many different possibilities to develop a dialogue system.

Finally, the chapter examines the important area of error handling in dialogue systems (section 2.5). There are two main types of understanding errors in dialogue system: non-understandings and misunderstandings. While misunderstandings are all erroneous interpretation results for the incoming utterances, non-understandings subsume all cases in which no interpretation could be assigned to an incoming utterance at all. Several strategies have been proposed to handle errors in dialogue systems such a repairing the error. However, the section also shows that besides errors originating from speech recognition, also out-of-domain utterances

are a main causer of errors (Bohus & Rudnicky, 2005), which cannot be repaired because the needed knowledge is not part of the domains of the system.

3 The KomParse Application

3.1 Introduction

This chapter presents the KomParse application, the test bed and development environment for the described SOX toolkit.

KomParse is a dialogue system that is used as the backbone of two conversational agents in a virtual world. In virtual worlds or multi-user online games such as *World of Warcraft* and social platforms such as *Second Life*, non-player characters (NPCs) have become an essential element. NPCs moderate the game plot, make the artificial world more vivid and create an immersive environment. They are also necessary to populate new worlds which otherwise would be deserted and unamusing. Dialogue and natural-language abilities are especially important characteristics if NPCs are to be entertaining, but the capabilities in autonomous acting and communication are still very limited. Most NPCs do not allow for natural language input and provide only a simple drop-down menu for dialogues with the user's avatar.

KomParse provides conversational agents which can be applied as NPCs to virtual worlds. The prototype is running in a world named *Twinity*, a product of the Berlin start-up company *Metaversum*[1]. The system combines state-of-the-art technologies such as finite-state graphs with sophisticated language technology improvements such as statistical dialogue-act recognition and semantic analysis to offer robust and at the same time flexible NPCs for natural-language interaction.

Following is an outline of how this chapter is organized. Section 3.2 describes the types of NPCs and the virtual world used in the prototype. Section 3.3 explains the dialogue-system architecture, whereas 3.3.2 presents the input analysis and interpretation component, and 3.3.3 the dialogue-management module in detail. Finally, 3.5 gives a conclusion.

[1] http://www.metaversum.com/

3.2 NPCs in the Virtual World

KomParse uses the virtual world *Twinity*[2] as an application for the NPCs. *Twinity* provides a virtual 3D-version of selected cities in the real world (currently Berlin, Singapore, London, Miami and New York). Users can create customized avatars, meet other users and communicate with them using the integrated text chat function. They can also rent or buy their own flat, style it according to their tastes, visit bars and clubs, and explore the city in 3D.

For this world, two specific NPC characters were modeled: the furniture sales agent "Cheryl Chaise" and the barkeeper "Hank Slender". Both NPCs rely on conversational interaction with the users, because they carry out their tasks through natural-language dialogues. Interaction is text-based: In a text chat the user can type natural language input. The NPC also responds in text. Utterances are shown in speech bubbles.

The task of the furniture seller is to help users with the interior design of their virtual apartments. The furniture seller can be invited to users' apartments. Users can then buy pieces of furniture and room decoration from the NPC by describing their demands and wishes through natural-language dialogue. The agent proposes objects according to the user's wishes and directly puts the selected objects in the room. She is also able to discuss where the objects should be placed and move the objects around in the virtual rooms.

The barkeeper owns a bar in the *Twinity* world where he sells cocktails and entertains his guests with trivia-type information. The celebrities dialogue (called "gossip-mode") is a question-answering scenario in which the agent answers questions using the database containing knowledge about celebrities[3]. The barkeeper is designed to be a communication partner and entertainer. Virtual worlds such as *Twinity* differ from games such as *World of Warcraft* insofar as the users are not assigned a quest. One of the ideas lying behind these worlds is a social interaction platform. Therefore, NPCs that offer social interaction are badly needed.

The NPCs also differ in their technology: While the furniture sales agent is task bound and does not offer any further conversational abilities, the barkeeper is designed to be a more open-domain chat partner. The barkeeper therefore has to support conversations about cocktails and

[2] `http://www.twinity.com/`, accessed 1 May 2011

[3] For more information see (Adolphs et al., 2010) and (Xu, Adolphs, Uszkoreit, Cheng, & Li, 2009).

bar-related topics, but should also engage in social talk and gossip dialogues. Whereas the barkeeper's dialogue model is more complex than the furniture sales agent's model, the barkeeper's task is much less sophisticated. The furniture sales agent's task, on the other hand, involves rich knowledge models about domain objects and pragmatic strategies concerning goal negotiation, and includes various challenging sub-tasks such as the calculation of appropriate pieces of furniture (Bertomeu, 2012).

The difference between the agents is mirrored in the number of knowledge bases they use. The furniture sales agent possesses only one very large hand-crafted ontology, modeling furniture, styles, and related concepts. The ontology also includes concepts of colors and their relatedness to human feelings.

In contrast, the barkeeper has access to several knowledge bases, namely a cocktail ontology, the upper-class ontology "YAGO"(Suchanek, Kasneci, & Weikum, 2007), the lexical resource "WordNet" and a huge database containing information about celebrities and the relations between them.

3.3 The Agent Architecture

Each NPC consists of an "avatar", which is the physical appearance of the NPC in the virtual world and the "conversational agent" which provides the control logic for the agent's behavior. The agent is hosted by a multi-client, multi-threaded server written in Java, whereas the NPC's avatar is realized by a modified *Twinity* client. It sends all in-game events relevant to our system to the server and translates the commands sent by the server into *Twinity*-specific actions. Rather than using the particular programming language and development environment of the platform to realize the conversational agent, KomParse uses an interface tailored to the specific needs related to connecting the agent with the avatar. In addition to the *Twinity* platform, KomParse offers a web interface for both agents and implemented connections to another 3D environment and the system can be extended to other platforms.

The dialogue system handles all conversation actions. It understands the input action coming from the user, selects an appropriate responding action and generates a responding physical or verbal action. Due to the mixed-initiative approach used in the agent, the dialogue system is also responsible for controlling the system initiative on the basis of dialogue context.

The dialogue system's architecture is shown in figure 3.1. The main components are the input analysis, the output generator and the dialogue manager, consisting of the input interpretation component and the action selector. For more information on dialogue-system architectures, see chapter 2.

The internally used knowledge bases are the dialogue graph, a finite-state graph which determines the flow of the default conversation, an electronic form, the dialogue memory, and a spatial model. The external knowledge bases used are the celebrity database containing facts about nearly 600,000 people and the relationships between them (Adolphs et al., 2010) based on YAGO, and the domain ontologies for both agents, namely a cocktail and a furniture ontology.

The *celebrities database* originates from three different sources: existing Semantic Web resources, data derived from semi-structured textual web data, and data that was learned from unstructured texts. Learning was carried out using a bootstrapping relation-extraction method, based on the typical relations between the found people, such as family relations, marriages and professional relationships. The database is encoded in RDF.

The *domain knowledge* bases are handwritten OWL ontologies. Whereas the cocktail ontology is comparatively small, the furniture ontology consists of 975 classes, 54 properties, 327 instances and 1,712 facts. Examples for a class are upper classes such as "Sofa" and special types of sofas such as "Sofa_Isodora". Properties describe the relations of the objects such as "hasStyle" and "isMadeOf". The ontology provides a sophisticated model of Color and Style. Color, for example, is modeled according to the HSV color model with additional values for relative luminance. This allows the processing of user wishes such as "I want a lighter sofa".

3.3.1 Answer Generation

Answer generation is template-based. Templates consist of surface strings and variable slots for dynamic content. Variable types are atomic values and lists of values. In the virtual world environment, the provided voice over IP interface is connected to the MARY (Schröder, Charfuelan, Pammi, & Steiner, 2011) TTS server. The generated agent's utterances are sent to the TTS system, which delivers a spoken output for the agent's utterances.

3.3.2 Input Analysis & Interpretation

The input-analysis component maps the raw text input from a user to an internal semantic representation format which is used in the interpretation afterwards. The internal representation is a predicate argument structure similar to the structure of PropBank annotations (Palmer, Gildea, & Kingsbury, 2005), but with consistent argument labeling.

Analysis of the input is realized by a hybrid approach, using three different solutions: Pattern matching, dependency pattern matching and dependency tree lookup. First the system tries to find a matching pattern in the pattern database. The patterns in the database are very specific, and so they can map an utterance directly to a semantic representation. Furthermore, some patterns already specify the dialogue act belonging to the utterance. Patterns consist of strings combined with regular expressions.

If no pattern was found for the input, the system tries to match a dependency pattern. Similar to other linguistic-based systems (J. Allen, Manshadi, et al., 2007), dependency structure is derived from a syntactic analysis including various steps of linguistic processing such as part-of-speech tagging, tokenization and dependency analysis carried out by the Stanford Parser (Marneffe & Manning, 2008). The dependency structures that are found are then tested against a database of flattened

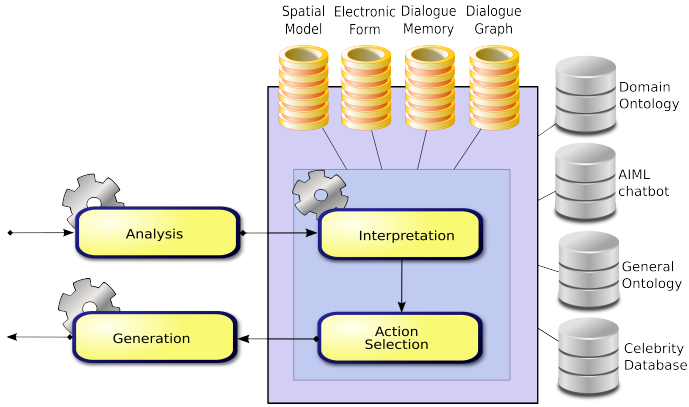

Figure 3.1: The System Architecture

dependency-structure patterns which map a string representation of the dependencies (combined with regular expressions) to a semantic representation. This allows for abstraction from the surface while maintaining the robustness of the pattern-matching algorithm. The last processing step is the extraction of a predicate argument structure from the dependency tree. This step is used for unseen input.

All approaches depend on some pre-processing steps. In the first step the input text is cleaned of smileys and common abbreviations. The system integrates the google spell checker[4]. In addition to tokenization and part-of-speech tagging, the named entity recognition tries to find entities known to the system, e.g., people's names and pieces of furniture.

The *interpretation* focuses on recognizing the dialogue acts and the topic object inherent in the user's utterances. It consumes a predicate argument structure and tries to deliver a dialogue act through a rule-based approach or a statistical model[5]. Analogously to the analysis, the rules in the interpretation component are responsible for the more specific and foreseen input, whereas the statistical model shall assign a dialogue act to unknown input. The model was trained by the Bayesian classifier AODEsr on a corpus with data from a Wizard-of-Oz experiment with the wizard controlling the furniture sales agent. Input features for the lookup are the predicate-argument representation as well as minimal dialogue context consisting of the preceding dialogue act and the last topic object.

3.3.3 Dialogue Flow

The core of the dialogue system is the dialogue graph which determines the next action according to the results of the input interpretation, database queries, or other environment changes. The dialogue graph is a finite-state graph combined with a frame-based approach. Frames constitute a declarative task model. The KomParse frames store the objects the users have discussed and bought so far together with the relations belonging to these objects. This information is taken from the knowledge bases at runtime. The graph is also data-driven in the sense that the conversation flow is determined by the results from knowledge base queries. Thus, the finite-state graph is made very flexible. Nevertheless, the positive characteristics of the finite-state approach such as robustness and easy authoring are maintained. For both agents the graph is the main

[4]http://code.google.com/p/google-api-spelling-java/

[5]The statistical dialogue-act classification is described in detail in chapter 5.

control engine. However, the barkeeper includes different dialogue domains and conversation types: The cocktail "task" conversations, some small-talk chat, and the dialogues about celebrities.

The dialogue memory stores all dialogue acts, topics, and utterances made by the system and the user, including the semantic representation. The memory is used for the detection of the topics and dialogue acts in the interpretation as well as the reference resolution of discourse entities and anaphora.

3.4 Data

This thesis contains several references to datasets used, which are generated using experiments with the KomParse system. Table 3.1 gives an overview of the datasets.

Dataset Name	Description
WoO1	Conversational data originating from a Wizard-of-Oz experiment with the wizard controlling the furniture sales agent in the virtual world
WoO2	Conversational data originating from a Wizard-of-Oz experiment with the wizard controlling the barkeeper agent in the virtual world
Eval	Conversational data originating from a field test with the barkeeper agent and game users in the virtual world

Table 3.1: The Datasets Used

The following sections describe the details of the single datasets.

3.4.1 WoO1

The data WoO1 is generated in a Wizard-of-Oz experiment (Bertomeu & Benz, 2009) in which 18 users furnish a virtual living room with the help of a furniture sales agent. Users buy pieces of furniture and room decoration from the agent by describing their demands and preferences in a text chat. During the dialogue with the agent, the preferred objects are then selected and directly put to the right location in the apartment. The scenario is task-driven and small talk is optionally initiated by the experiment participant only.

USR:	And do we have a little side table for the TV?
NPC:	I could offer you another small table or a sideboard.
USR:	Then I'll take a sideboard that is similar to my shelf.
NPC:	Let me check if we have something like that.

Table 3.2: Example Conversation From the WoO1 Dataset

In the experiments, 18 users spent one hour each furnishing the living room by talking to a human wizard controlling the virtual sales agent. The participants were German and the language used was English. The final corpus consists of 18 dialogues containing 3,171 turns with 4,313 utterances and 23,015 alpha-numerical strings (words). The example in table 3.2 shows a typical part of such a conversation. Table 3.3 presents one of the conversation parts in which small talk and task talk are mixed by the user.

The data is annotated with discourse and pragmatic information including dialogue acts, projects according to (Clark, 1996), sentence mood, the topic of the conversation and an automatically retrieved information state for every turn of the conversations.

NPC:	I think the colourful chair will go better with the style of the room.
USR:	Ok, show it to me.
USR:	Do you practice any sport?
NPC:	Here it is.
NPC:	Yes, I'm practising climbing.
USR:	Oh, show me another one...this is too hippy...
USR:	Oh thats a casuality...i practice climbing too...

Table 3.3: Small Talk Example Conversation from the WoO1 Dataset

3.4.2 WoO2

The dataset *WoO2* contains log files from text-based conversation between a barkeeper agent and participants of another Wizard-of-Oz experiment.

USR:	Wie lange arbeitest du schon hier in diesem Bar?
NPC:	“Bar” ist feminin, also “in dieser Bar”
NPC:	Schon 11 Jahre
USR:	Das ist schon sehr lange. Was hast du vorher gemacht?
NPC:	In Dortmund war ich Metzger.

Table 3.4: Example Conversation from the WoO2 Dataset

The participants of the experiment are German language learners from various countries, who order drinks and food in the virtual bar. The barkeeper is controlled by the wizard. The participants are explicitly briefed to conduct small talk and the wizard himself initiates small-talk conversation if the user does not. Conversations take place in German and the barkeeper also acts as language corrector.

This corpus contains 12 dialogues with 1,477 utterances. Table 3.4 shows a typical excerpt from a WoO2 conversation.

3.4.3 Eval

The conversation data in the *Eval* datasets originates from user experiment log files. In the experiments, the users talked to a barkeeper agent in the virtual world controlled by the *KomParse* dialogue system. The agent can talk about cocktails, gossip and some typical small-talk topics such as personal information. The users have to fulfill a list of tasks with the agent through text-based dialogues. The tasks included, but were not restricted to, ordering cocktails and asking for biographic information of a famous personality. Small talk with the barkeeper was also added to the list of tasks to ensure that the users took the opportunity to chat with the NPC. The conversations were in English.

Table 3.5 shows an example of a conversation between a user and the barkeeper agent.

12 users participated in the evaluation experiments. The resulting conversations consist of 1,408 turns, containing mixed small-talk and task-talk conversations.

AGNT:	There is an overwhelming choice of great cocktails: Do you like fresh or spicy?
USER:	a fresh one please
AGNT:	Ok, just a second.
AGNT:	Here it is. This cocktail is called Demeter. You want it?
USER:	my name is tom and you?
AGNT:	Pleased to meet you, tom.
AGNT:	So, what about the cocktail?

Table 3.5: Example Conversation From the EVAL Dataset

3.5 Conclusion

This chapter describes the KomParse application, which is the evaluation test bed and embedding framework for the research and the SOX software toolkit described in this thesis. The KomParse system operates software agents that can engage in natural-language dialogue. KomParse agents are used as non-player-characters (NPCs) in a virtual world. Such NPCs are important for virtual worlds, since they can create an immersive environment and thus enhance the fun to use the game.

The KomParse system hosts two types of agents, a furniture sales NPC and a barkeeper NPC. Both NPCs are based on dialogue interaction with the users. The furniture sales agent is a task oriented agent that helps to furnish virtual apartments by discussing wishes and pieces of furniture with the users. The barkeeper on the other hand can discuss cocktails, but its prior task is entertaining small talk about celebrities and other small talk topics. The barkeeper agent is the main test bed for the presented research and technologies.

The chapter presents the technology of the KomParse dialogue system as well as three conversational data sets. The data sets are used in the this thesis for various purposes such as analysis of conversation patterns, training of machine learning approaches and software testing.

The KomParse dialogue system combines robust technologies such as finite-state graphs with sophisticated methods such as dialogue-act recognition and semantic web resources. The system is mixed-initiative. Conversations with the agents is technically realized through typed natural-language input in the game's chat window. The received input is transferred from the game client to a KomParse dialogue system server. The server analyzes the incoming input using a natural-language processing pipeline based on linguistic analysis including part-of-speech analysis, named entity recognition, anaphora resolution and dependency parsing. The results of the linguistic analysis are used for the interpretation of the utterance keeping in mind the current dialogue context. Interpretation results in an appropriate dialogue act, a detected topic cloud and the domain of the utterance. A graph-based dialogue manager with an additional form-based task model consumes the results of the interpretation and decides what action to carry out next. Generation of utterances is realized by a template and slot-filling approach.

The KomParse agents and especially the barkeeper agent are typical representations of the group of embodied conversational agents. As already mentioned in chapter 1, this group of agents is often subjected to social talk utterances, because they posses aspects of human personality such as embodiment and a human voice ((Nass et al., 1994), (Blascovich, 2002)). Furthermore, the barkeeper agent itself actively engages in small talk. The system therefore provides a perfect test environment for the integration of social talk in dialogue systems.

4 Related Work

4.1 Introduction

This chapter details the latest research in areas relevant to the research topics in this thesis. The chapter contains a section for related work in the field of dialogue management (section 4.2). Topics relevant to natural-language understanding are presented in the sections "Dialogue-Act Recognition" (4.4.2) and "Domain Classification" (4.3.2). Section 4.3 describes relevant work in the research field of error handling in dialogue systems and especially for handling out-of-domain utterances. Section 4.4 gives an overview about existing work regarding social talk, from the dialogue-act schemata point of view as well as the development of conversational, personalized agents.

4.2 Dialogue Management

4.2.1 State-based Dialogue Management

There are a lot of different toolkits and dialogue-management implementations for dialogue systems available in research and industry. They rely on different dialogue-management approaches such as finite-state (RAD/CSLU toolkits (McTear, 1998)), agent based (TRIPS/PLOW (J. F. Allen, Miller, Ringger, & Sikorski, 1996) (J. Allen, Ferguson, & Stent, 2001) (J. Allen, Chambers, et al., 2007)), information-state update (TRINDI (Larsson & Traum, 2000)), form filling and agenda based (Ravenclaw/Olympus(Bohus & Rudnicky, 2003)) or mixtures of those. All of these approaches try to find a balance between complexity, ease of use, robustness, and flexibility, regarding the demands of the desired application.

In finite-state dialogue management the dialogue model and the task model are integrated into a finite-state automaton. All possible dialogue

moves are encoded at the authoring time of the dialogue and cannot be modified during conversation. A more complex dialogue can quickly lead to unreadable dialogue graphs - at least for humans. However, nowadays state-based dialogue systems include additional declarative task models, such as forms, and integrate knowledge resources to make the dialogues more flexible. These extensions allow for a more dynamic behavior calculated at runtime and not at development time. Although state-based systems are often seen to be inferior to sophisticated machines such as plan-based dialogue systems, they offer a handful of good properties which is why they are still often used, especially in industrial applications. One of the main strengths of these systems lies in the intuitive usage of the technology: Graphs are often easy to develop with comparatively less training time. Another often mentioned benefit is the performance when compared to, e.g., plan-based or agent systems, which often depend on heavy computing.

Dialogue systems using finite-state or extended finite-state approaches are, e.g., the CMU Communicator system, the previous version of the RavenClaw dialogue manager ((Bohus & Rudnicky, 2009), (Goddeau, Meng, Polifroni, Seneff, & Busayapongchaiy, 1996)) and the KomParse system described in chapter 3.

A lot of different industrial and research authoring tools for developing and modifying state transition graphs are available. Examples are the CSLU toolkit (McTear, 1998), ARIADNE (Denecke, 2002) or (Bui, Rajman, & Melichar, 2004), Scenejo (Spierling, Weiß, & Müller, 2006) or DialogOS (Bobbert & Wolska, 2007). The more advanced ones support Harel's state chart features (Harel, 1988) such as Cyranus (Iurgel, 2006), or SceneMaker ((Klesen, Kipp, Gebhard, & Rist, 2003), (Gebhard, Kipp, Klesen, & Rist, 2003), (Mehlman, 2009)). SceneMaker offers a very clear user interface for authoring graphs, incorporating Harel's state charts and a fast graph interpreter including a Java API. In contrast to, e.g., the CSLU toolkit, it is very easy to integrate custom-made components and knowledge into the SceneMaker graph.

4.2.2 Multi-Threaded Dialogue Management

Most current dialogue systems support the embedding of clarification sub-dialogues, which are most often adjacency pairs (Schegloff & Sacks, 1973). They will interrupt the active conversation if they don't understand the user's utterance, and start a clarification dialogue and continue with the mother thread afterwards.

In contrast to embedded dialogue threads, support for multi-threaded dialogues is still very rare, although Rosé, Di Eugenio, Levin, and Carol (1995) already stated the multi-threaded character of human conversation and dropped the idea of a tree-structured dialogue processor. In graph-based dialogue management, multi-threading has not been supported. The reason lies in the resulting complex structure of traditional graphs if they were to allow switching back and forth between nearly all parts of the graph. In a traditional graph the only way to encode switching back and forth is the addition of many edges between nearly every node. The manual generation of such a graph is impossible and the resulting graph is no longer readable. The afore mentioned Cyranus (Iurgel, 2006) offers the interesting concept of "reference nodes", which encapsulate behavior occurring in the graph several times. These nodes can be seen as a necessary step for integrating multi-threaded behavior. Nevertheless, they are limited to one instance and therefore not usable for real multi-threaded support.

To the best of my knowledge, the only explicit example of a multi-threaded dialogue system is described in (Lemon et al., 2002), who use an information-state update model of dialogue management. The multi-threaded support consists of several components: a pending list of questions the system has initialized but the user has not answered yet, a declarative activity tree which manages all activities of the system (tasks), and the system agenda, which stores what should be said by the system. The components are controlled by the Dialogue Move Tree (DTM), which operates as a message board for all dialogue contributions. The DTM decides which tree every incoming input shouldbe attached to, updates the pending list if a question was not answered, and writes answers on the system agenda.

The described technology is also integrated into the system described in Lemon and al. (2003), who suggest an architecture allowing interwoven threads, but focuses on "low-level phenomena", such as markers for turn management and providing feedback.

In robots and other architectures, especially plan-based systems, it is also quite common to have several active execution threads ((Nakano et al., 2011), (Firby, 1994)). A robot needs to be able to guide a user and talk to him at the same time. Nevertheless, this means managing various actions, e.g., a physical and a verbal action, not explicitly managing several conversational threads.

4.3 Error Handling

4.3.1 Error-Handling Strategies

Strategies for error handling in dialogue systems are a well known necessity and are already commonly explored. Basically, there are two major groups of errors in dialogue systems: misunderstandings and non-understandings. Misunderstandings occur if the input analysis and interpretation assign the wrong meaning to an input. Non-understandings occur if the input analysis and interpretation is not able to assign a representation to the incoming utterance at all. The main research focus in error-handling strategies has long been recovering from misunderstandings. The crucial part in handling misunderstandings is the detection of the misunderstanding, whereas the recovery from misunderstandings has already received a lot of attention and several well-accepted solutions have been suggested. Several strategies for repairing misunderstandings have been provided in literature such as explicit or implicit confirmations (Krahmer, Swerts, Theune, & Weegels, 2001) and specific clarification strategies (Schlangen, 2004). For detection, most systems use confidence scores.

On the other hand, dealing with non-understandings is not very well understood. Many systems use a handful of common recovery strategies such as "Ask the user to repeat", "Ask the user to rephrase" or they ignore the nonunderstood input. Others indicate the non-understanding to the user and just move on with the dialogue ((Bohus, 2007), (Jokinen & McTear, 2009), (San-Segundo et al., 2000), (Komatani & Kawahara, 2000)). Most of these strategies are taken from empirical investigations of human-human interaction ((Zollo, 1999), (Skantze, 2003) (Koulouri & Lauria, 2009)). Several works have considered the question: Which of the given error-recovery strategies are most suitable and best liked by users? Henderson, Matheson, and Oberlander (2012), for example, show that a mixture of two strategies to handle non-understandings is perceived as much better by the users than just one recovery strategy. They also suggest three new strategies for use in a conversational museum-guide application: "Ask if the user would like to hear more information about an item", "Ask if the user is more interested in hearing about aspect A or aspect B of an item", and "Fake having forgotten to say something of interest about the item". Their evaluation shows that these strategies are judged significantly better than the baseline "Ask for repetition" strategy. This observation is already reported in Bohus and Rudnicky

(2008), who also show that from a set of different strategies "Move on with the dialogue" gets best results.

However, most system still use the standard set of strategies, although results in empirical studies such as (Bohus & Rudnicky, 2005) show that, besides errors originating from the ASR, the next major reasons for non-understandings are "out-of-grammar" and "out-of-application" utterances[1]. Particulary for these utterances, the described error-recovery strategies are not recommendable. The system will never be able to understand an out-of-domain utterance, not even if it asks for repetition or rephrasing. In fact, the dialogue may go from bad to worse if the system insists on understanding a task-related utterance where this is none.

The reason that many systems still rely on the standard strategy set may lie in the fact that most existing dialogue systems are still task bound, meaning they are built to assist users in a special task, such as a journey-travel system. If a journey-travel system is confronted with non-understandings, it may decide to use one of the traditional strategies for error recovery such as "Ask for repetition" or "Move on with the dialogue". As long as the system does not aim at being conversational and entertaining, it is not forced to handle out-of-domain understanding errors in another way than in-domain understanding errors.

If the system should be conversational the problem is often avoided by integrating chatbots or ad-hoc solutions. Not many suggestions for strategies that specifically handle out-of-domain errors have been made so far. A rare example of a system explicitly dealing with out-of-domain errors is described in (Patel, Leuski, & Traum, 2006). In their work involving virtual characters, the authors describe a system that uses eight different classes of possible out-of-domain input such as "question makes no sense" or "question about specific human characteristics". They use different classifiers to assign one of 55 in-domain classes or one of the eight out-of-domain classes to an input. Their evaluation shows a significant improvement integrating the explicit handling of out-of-domain utterances. However, their work differs from the work described in this thesis, because they do not differentiate between understood and nonunderstood input. The suggested classes mainly need some kind of understanding of the user's input. Only the classes "questions without sense"

[1]The authors of (Bohus & Rudnicky, 2005) use the notion "out-of-application" to denote a group of utterances that are mainly in-domain but out of the scope of possible functions of the system. In this thesis all utterances which are out of the scope of the machine's knowledge are called "out-of-domain utterances".

and "out-of-domain" are classes of nonunderstood utterances. If we already know that a question is about specific human characteristics we already have understood quite a lot of the input. The classes are also not all usable outside the given application, since they are specific to the virtual character. Also, the realization of the classes consist of only a few possible pre-canned answers for every class. The evaluation consists of the rating of several question-answer pairs by three evaluators regarding an unknown evaluation measure which can best be described with some kind of informational coherence between the input and the answer. Other aspects such as the naturalness or the degree of how much the evaluators subjectively liked the answer are not considered.

4.3.2 Out-of-Domain Classification

In contrast to the huge amount of existing solutions for other classification tasks in natural-language interpretation e.g. dialogue-act recognition, domain classification is very rare. Although, e.g., Henderson et al. (2012) emphasize the necessity of using other error strategies for entertaining and conversational systems, they still do not include an out-of-domain classification, but handle all non-understandings the same way. One of the rare domain verification approaches is described in (Lane et al., 2007). Lane et al. (2007) point out the importance of the in-domain and out-of-domain differentiation in relation to correct recognition of in-domain utterance, which should be repaired. They use a two-step approach to in-domain verification to specifically handle the in-domain non-understandings. Their approach is based on topic detection. Topics are a set of application-specific pre-encoded categories. The first step of the approach is to calculate the probability of an incoming utterance belonging to a single topic category against every known topic. In the second step, all probability values are used as input to an in-domain classification which decides if the utterance is in-domain or out-of-domain. The authors use an adoption of a linear discriminant model for the verification. Linear discriminant weights are applied to the confidence scores coming from the topic classification. Afterwards the weighted sum is compared to a defined threshold. If the sum is greater than the threshold, the utterance is classified to be in-domain. The evaluation result is given by means of the equal error rate (EER), the percentage value in which false negatives and false positives are equal. EER is commonly used in biometric systems. Their best result is a drop of the EER from 26.4% to 17.3%.

Another approach to out-of-domain recognition is described in (Fujita et al., 2011). Fujita et al. (2011) do not incorporate topic information but use a bag-of-words as input to the classifier. Classification is done using a support vector machine. The full set of input features is the bag-of-words vector consisting of the frequency of each word in a word list, which is a list containing in-task utterances training data, the number of words, the frequency of unknown words, and a similarity score that indicates the similarity with examples in an in-task database. The authors achieve a significant improvement in classification using the described features against an example-based baseline. Fujita et al. (2011) report the results of the classification in terms of EER, too. In the best reported result the EER drops from 21.3% to 13.0%.

Patel et al. (2006) use a classifier trained on in-domain and out-domain utterances to classify incoming utterances in one of 55 in-domain classes or one of eight out-of-domain classes. Some of the classes are very specific and an utterance classified in one of these classes can no longer be called an out-of-domain utterance, since a major part of the utterance was actually understood. They use their own classifier based on statistical language modeling techniques used in cross-lingual information retrieval (Leuski, Patel, & Traum, 2006).

4.4 Social Talk

4.4.1 Models of Social Talk in Conversational Agents

Several papers have stated the relevance of social talk for conversational agents (T. W. Bickmore & Cassell, 2000), especially in embodied agents. Agents which are able to deviate from task talk to social talk are found to be more trustworthy (T. Bickmore, 1999) and entertaining (Kopp et al., 2005). One example is REA, a relational agent for real-estate sales (Cassell et al., 1999) developed at the M.I.T Gesture and Narrative Language Group. REA incorporates several measures for social "closeness" to the user, which she uses to improve her rhetorical ability in the delicate domain of money and real estate. Nevertheless, the small talk is system-initiated and user utterances are partly ignored (T. W. Bickmore & Cassell, 2001).

In an earlier implementation of REA, the system used a conversational sequence for small talk (T. Bickmore, 2003), which was first formalized by Klaus Schneider (Schneider, 1988) in his analysis of small

talk as genre. The sequence consists of four turns and can be used for all small-talk topics.

1. A query from the dominant interactor
2. An answer to the query
3. A response to the answer, consisting of one of the following possibilities: Echo-question, check-back, acknowledgement, confirming an unexpected response, positive evaluation.
4. An unrestricted number or zero steps of idling behavior

Although the small-talk sequence found by Schneider seems to be typical for small-talk conversations, it is not the only possible one, and repetitions of the conversation pattern quickly become unnatural.

Another agent which uses small talk is "Max" developed at the university of Bielefeld (Kopp et al., 2005). Max is an embodied agent acting in real-world scenarios such as a museum guide. Max possesses small talk capabilities in order to be an enjoyable and cooperative interaction partner. Small talk is based on rules which assign keywords and key phrases to appropriate reactions. Although the system uses a set of dialogue acts it is not clear if this also applies to the small-talk rules.

An agent which makes use of small-talk topic knowledge is "helper agent" described by Isbister, Nakanishi, Ishida, and Nass (2000). The system supports human conversation partners interacting in a virtual room by suggesting safe small-talk topics if conversation pauses. The system has knowledge of small-talk topics but no model for small talk itself. To propose a new topic the agent always follows the same sequential pattern.

Another solution is the integration of a chatbot, such as the AIML-based chatbot ALICE[2]. Chatbots are built especially for open-domain small talk. However, chatbots are simple stimulus-response machines commonly based on surface-pattern matching without any explicit knowledge about dialogue acts, strategies or sequences (Klüwer, 2009). They also lack a satisfying memory implementation and rely on stateless pattern-answer pairs. Thus, conversations with chatbots are often tedious and unnatural. Because of their stateless approach it is also complicated to integrate them into dialogue systems.

[2]ALICE is an open source chatbot with a database of approximately 41,000 pattern-template pairs (http://alicebot.blogspot.com/)

4.4.2 Dialogue Act Recognition

Dialogue acts (DAs) represent the functional level of a speaker's utterance, such as a greeting, a request or a statement. Dialogue acts are verbal or non-verbal actions that incorporate participant's intentions originating from the theory of speech acts by Searle and Austin (Searle, 1969). They provide an abstraction from the original input by detecting the intended action of an utterance, which is not necessarily inferable from the surface input. See the two requests in the following example.

Can you show me a red car please?
Please show me a red car!

To detect the action included in an utterance, different approaches have been suggested in recent years which can be clustered into two main classes. The first class uses AI planning methods to detect the intention of the utterance based on belief states of the communicating agents and world knowledge. These systems are often part of an entire dialogue system, e.g., in a conversational agent that provides the necessary information about current beliefs and goals of the conversation participants at runtime. One example is the TRIPS system (J. F. Allen et al., 1996). Because of the huge amount of reasoning, systems in this class generally gather as much linguistic information as possible. The second class uses cues derived from the actual utterance to detect the right dialogue act, mostly using machine learning methods. This class gained a lot of attention due to lower computational costs. The probabilistic classifications are carried out via training on labeled examples of dialogue acts described by different feature sets. Frequently used cues for dialogue acts are lexical features such as the words of the utterance or n-grams of words ((Verbree, Rienks, & Heylen, 2006), (Zimmermann, Liu, Shriberg, & Stolcke, 2005), (Webb & Liu, 2008)). Although the performance of the classification task is difficult to compare, because of the variety of different corpora, dialogue-act sets, and algorithms used, these approaches do provide emphatically good results; for example, Verbree et al. (2006) achieve accuracy values of 89% on the ICSI Meeting Corpus containing 80,000 utterances with a dialogue-act set of five distinct dialogue-act classes and "ngrams of words" and "ngrams of POS information", amongst other features.

Another group of systems utilizes acoustic features derived from automatic speech recognition for automatic dialogue-act tagging (Surendran & Levow, 2006), context features like the preceding dialogue act, or n-grams of previous dialogue acts (Keizer & Akker, 2006).

However, grammatical and semantic information is not incorporated into feature sets that often, with the exception of single features such as the type of verbs or arguments, or the presence or absence of special operators e.g. wh-phrases (Andernach, 1996). Keizer, Akker, and Nijholt (2002) use, among others, linguistic features like sentence type for classification with Bayesian networks. Although Jurafsky et al. (1998) already noticed a strong correlation between selected dialogue acts and special grammatical structures, approaches using grammatical structure were not very successful.

While grammatical and semantic features are not frequently incorporated into dialogue-act recognition, they are commonly used in related fields like automatic classification of rhetorical relations. For example Sporleder and Lascarides (2008) and Lapata and Lascarides (2004) extract verbs as well as their temporal features derived from parsing to infer sentence internal temporal and rhetorical relations. Their best model for analyzing temporal relations between two clauses achieves 70.7% accuracy. (Subba & Di Eugenio, 2009) also show a significant improvement of a discourse relation classifier incorporating compositional semantics compared to a model without semantic features. Their VerbNet based frame semantics yield a better result of 4.5%.

4.4.3 Social Dialogue Acts

There are many sophisticated and systematic dialogue-act annotation schemes. Two of the most popular ones are DAMSL (J. Allen & Core, 1997) and DIT++ (Bunt, 2011). DAMSL, Dialogue Act Markup using Several Layers, was first published in the 1990s with a focus on multidimensionality. DAMSL is used for many corpora annotations, often in slightly modified versions. In general, DAMSL does not offer a special annotation layer for social acts, although some social information is coded in existing classes; for example, in the Switchboard DAMSL version (SWDB-DAMSL) (Jurafsky, Schriberg, & Biasca, 1997) there are several feedback dialogue acts which have social meaning such as the sympathy feedback, and the downplayer as a reply to compliments.

In the ICSI-MRDA annotation scheme (Shriberg, Dhillon, Bhagat, Ang, & Carvey, 2004) a new category is introduced for "politeness mechanisms" including downplayers, sympathy, apology, thanks and welcome.

DIT++ offers a special dimension for "social obligations management", in which general communicative functions can get a social interpretation. Moreover, there are some communicative functions especially for the social obligations management dimension, which are similar to the mentioned DAMSL acts: Initial Greeting, Return Greeting, Initial Self-Introduction, Return Self-Introduction, Apology, Apology-Downplay, Thanking, Thanking-Downplay, Initial-Goodbye, Return-Goodbye. However, DIT++ is also not equipped to model a definite small-talk sequence, such as a compliment-downplay-feedback sequence. As far as we know, a compliment, for example, can not be explicitly marked as such.

The mentioned schemes are further developed in the ISO project "Semantic annotation framework" (Bunt et al., 2010), whose central focus lies on the multidimensionality and multi-functionality of dialogue utterances. This matches the observation from Coupland, Coupland, and Robinson (1992) regarding social talk. They discovered that every utterance carries a special degree of "phatecity". Nevertheless, no further social dialogue acts are introduced by the ISO project.

4.5 Conclusion

This chapter describes latest research in areas relevant to this thesis. The main three thematic blocks are *error handling*, *social talk* and *dialogue management*.

Regarding dialogue management, the chapter gives an overview of graph-based dialogue management by contrast with other dialogue management possibilities such as plan-based systems (section 4.2). The section shows that graph-based models for dialogue management are still very common and offer some benefits such as ease of use, but also suffer from several serious drawbacks ((McTear, 1998),(Bohus & Rudnicky, 2009), (Goddeau et al., 1996)). One is the flexible management of several conversation threads. The section therefore also deals with the existing work in the field of flexible dialogue models enabling multi-threaded behavior. We have seen that only very few approaches exist for multi-threading (Lemon et al., 2002) and more work has been conducted in the area of parallel execution of different physical actions such as speaking and moving in robots ((Nakano et al., 2011), (Firby, 1994)).

In section 4.3 we have seen several strategies for error handling in dialogue systems. It is shown that nearly no existing work deals with the

explicit handling of errors which were caused by out-of-domain utterances, but most systems use a common set of dialogue strategies which either try to repair an understanding error, ignore the error, or signal the error, but move on with the dialogue ((Bohus, 2007), (Jokinen & McTear, 2009), (San-Segundo et al., 2000), (Komatani & Kawahara, 2000)). However, repairing an error originating from an out-of-domain utterance cannot succeed, because the utterance cannot be understood and too much ignoring and signaling of errors can significantly lower the user's satisfaction with the dialogue. The section also includes an overview of existing domain classification approaches and in-domain verification.

Lastly, the chapter describes the latest ideas in the field of social talk models and dialogue acts (section 4.4). This section gives an overview of social dialogue acts in existing dialogue-act sets such as DAMSL (J. Allen & Core, 1997), SWDB-DAMSL (Jurafsky et al., 1997), DIT++ (Bunt, 2011), and the ICSI-MRDA annotation scheme (Shriberg et al., 2004). We have seen that all considered sets fail in providing sufficient categories for social dialogue acts. Additional work is needed on explicit social-talk dialogue acts. The section also describes common approaches to dialogue-act recognition and a description of related work in social talk for conversational agents. It shows that communication patterns for social talk are not well known. Also, currently, no extensive model of social talk exists that could be used in dialogue systems.

The chapter shows that despite some helpful and promising ideas, existing work in dialogue management, error handling and social talk cannot satisfactorily produce social talk in dialogue systems. The discussion of methods, advantages and short-comings in existing approaches reveals a great need for further work to generate computational usable social talk models, as well as support for flexible dialogue management and new strategies for handling out-of-domain errors.

5 Natural-Language Understanding

5.1 Introduction

This chapter describes the parts of the thesis work that belong to the research field of natural-language understanding (see chapter 2, specifically:

- Domain classification
- Dialogue-act recognition
- Topic detection

Each of these three parts predominantly belong to one of the three main research areas of the thesis (domain classification, for example, is especially needed for error handling). However, other parts of the system also depend on the technology and results of the corresponding components. The dialogue manager in particular needs values for topic, domain, and dialogue act to handle the incoming utterance. This chapter describes the natural-language understanding approaches presupposed in the later chapters of the thesis.

5.2 Dialogue Act Recognition

Dialogue-act recognition is an essential task for dialogue systems. Automatic dialogue-act classification has received much attention in the past few years either as an independent task or as an embedded component in dialogue systems. Various methods have been tested on different corpora using several dialogue-act classes and information coming from the user input.

In contrast to existing systems using mainly lexical features, i.e., words, single markers such as punctuation (Verbree et al., 2006), or combinations of various features (Stolcke et al., 2000) for the dialogue-act

classification, the results of the interpretation component presented in this chapter are based on syntactic and semantic relations. The system first gathers linguistic information for an utterance coming from different levels of deep linguistic processing similar to (J. Allen, Manshadi, et al., 2007). The information retrieved is used as input for an information extraction component that delivers the relations embedded in the actual utterance (Xu, Uszkoreit, & Li, 2007). These relations combined with additional features (a small dialogue context and mood of the sentence) are then utilized as features for the machine-learning based recognition.

Dialogue-act recognition is carried out via the Bayesian network classifier AOEDsr from the WEKA toolkit. AODEsr augments AODE, an algorithm that averages a small number of alternative naive-Bayes-like models that have weaker independence assumptions than naive Bayes, with subsumption resolution (Zheng & Webb, 2006).

The classifier is trained on a corpus originating from a Wizard-of-Oz experiment that was semi-automatically annotated. It contains automatically annotated syntactic relations, namely predicate argument structures that were checked and corrected manually afterwards. Furthermore, these relations are enriched by manual annotation with semantic frame information from VerbNet to gain an additional level of semantic richness. These two representations of relations, the syntax-based relations and the VerbNet semantic relations, were used in separate training steps to detect how much the classifier can benefit from either notation.

A systematic analysis of the data has been conducted. It turns out that a comparatively small set of syntactic relations cover most utterances, which can, moreover, be expressed by an even smaller set of semantic relations. Because of this observation as well as the overall performance of the classifier, the interpretation is extended with an additional rule-based approach to ensure the robustness of the system.

5.2.1 Classification Features

Pragmatic Features

The pragmatic information selected as input features in the recognition system are:

- The sentence mood. Sentence mood was annotated with one of the following values: declarative or imperative, interrogative.
- The topic of the utterance

- The topic of the directly preceding utterance.
- The last preceding dialogue act.

Predicate Argument Structure

The second level of information automatically enriches the input vector with predicate argument structures. Each utterance is parsed with a predicate argument parser and annotated with syntactic relations organized according to PropBank (Palmer et al., 2005), and containing the following features: Predicate, subject, objects, negation, modifiers, and copula complements.

A single relation mainly consists of a predicate and the associated arguments. Verb modifiers like attached PPs are classified as "argM" together with negation ("argM_neg") and modal verbs ("argM_modal"). Arguments are labeled with numbers according to the information found for the actual structure. PropBank is organized in two layers. The first one is an underspecified representation of a sentence with numbered arguments. The second one contains fine-grained information about the semantic frames for the predicate comparable to FrameNet (Baker, Fillmore, & Lowe, 1998). While the information in the second layer is stable for each verb, the values of the numbered arguments can change from verb to verb. While for one verb the "arg0" may refer to the subject of the verb, another verb may encapsulate a direct object behind the same notation "arg0". This is very complicated to handle in a computational set-up, which needs continuous labeling for the successive components. Therefore the arguments were generally named as in PropBank but consistently numbered by syntactic structure. This means, for example, that the subject is always labeled as "arg1".

Consider the example: "Can you put posters or pictures on the wall?" The syntactic relation will yield the following representation:

```
<predicate: put>
<ArgM_modal: can>
<Arg1: you>
<Arg2: posters or pictures>
<ArgM: on the wall>
```

Predicate Argument Structure Parser The syntactic predicate argument structure that constitutes the syntactic relations and serves as

a basis for the VerbNet annotation is automatically retrieved by a rule-based predicate argument parser. The rules utilized by the parser describe subtrees of dependency structures in XML by means of relevant grammatical functions. For detecting verbs with two arguments in the input, for instance, a rule can be written describing the dependency structure for a verb with a subject and an object. This rule would then detect every occurrence of the structure "Verb-Subj-Obj" in a dependency tree. This sample rule would express the following constraints: The matrix unit should be of the part of speech "verb", and the structure belonging to this verb must contain a "nsubj" dependency and an "obj" dependency.

The rules deliver raw predicate argument structures, in which the detected arguments and the verb serve as hooks for further information look-up in the input. If a verb fulfills all requirements described by the rule, the second step is to recursively acquire all modificational arguments existing in the structure. The same is done for modal arguments as well as modifiers of the arguments, such as determiners, adjectives, or embedded prepositions. After generating the main predicate argument structure from the grammatical functions, the last step inserts the content values present in the actual input into the structure to get the syntactic relations for the utterance.

The embedded dependency parser is the Stanford Dependency Parser (Marneffe & Manning, 2008), but other dependency parsers could be employed instead. The predicate argument parser is stand-alone software and can be used either as a system component or for batch processing a text corpus.

5.2.2 Dialogue Act Recognition Evaluation

The presented dialogue-act classification approach presented here is evaluated using the conversation data "WoO1" from a Wizard-of-Oz experiment. In the experiment 18 users furnish a virtual living room with the help of a furniture sales agent. Users buy pieces of furniture and room decoration from the agent by describing their demands and preferences in a text chat. The dataset is described extensivly in chapter 3.

The conversations' log files constitute the corpus for annotation. Conversations are manually segmented into utterances beforehand. The annotation of the corpus includes manual annotation with the pragmatic information, the automatical annotation with the syntactic predicate argument structure, and the manual addition of semantic predicate class

and semantic role information. The idea is to compare the recognition results achieved with the syntactic information against the recognition results of the semantic information. VerbNet (Schuler, 2005) is utilized as a source for semantic information. The VerbNet role set consists of 21 general roles used in all VerbNet classes. Examples of roles in this general role set are "agent", "patient" and "theme". Words in the utterance are used as the baseline comparison.

Thus, the features used for dialogue-act recognition are:

- Context features: The last preceding dialogue act, equality between the last preceding topic and the actual topic, and sentence mood.
- Syntactic relation features: Syntactic predicate class, arguments, and negation.
- VerbNet semantic relation features: VerbNet predicate class, VerbNet frame arguments,and negation.
- Utterance features: The original utterances without any modifications.

For the manual addition of the semantic frame information, a web-based annotation tool has been developed. The annotation tool shows the utterance that should be annotated in the context of the dialogue,

Dialogue Act	Meaning	Frequency
REQUEST	The utterance contains a wish or demand	449
REQUEST_INFO	The utterance contains a wish or demand regarding information	154
PROPOSE	The utterance serves as a suggestion or for showing an object	216
ACCEPT	The utterance contains an affirmation	167
REJECT	The utterance contains a rejection	88
PROVIDE_INFO	The utterance provides information	156
ACKNOWLEDGE	The utterance is positive feedback	9

Table 5.1: The Dialogue-Act Set Used

including the information from the preceding annotation steps. All VerbNet classes containing the current predicate are listed as possibilities for the predicate classification together with their syntactic frames. The annotators can select the appropriate predicate class and frame according to the arguments found in the utterance. If an argument is missing in the input that is required in the selected frame a null argument is added to the structure. If the right predicate class exists, but the predicate is not yet a member of the class, it is added to the VerbNet files.

In the event that the right predicate class is found but the fitting frame is missing, the frame is added to the VerbNet files. Thus, during annotation 35 new members have been added to the existing VerbNet classes, four frames and four new subclasses. Through these modifications, a version of VerbNet has been developed that can be regarded as a domain-specific VerbNet for the sales domain.

During the predicate classification, the annotators also assign the appropriate semantic roles to the arguments belonging to the selected predicate. The semantic roles are taken from the selected VerbNet frame. From the annotated semantic structure, semantic relations are inferred, such as the one in the following example:

```
<predicate: put-3.1>
<agent: you>
<theme: posters or pictures>
<destination: on the wall>
```

From the annotated data, two datasets are derived: A dataset containing the user's utterances (CST) and a dataset containing the wizard's utterances (NPC), whereas the NPC corpus is cleaned from the "protocol sentences". Protocol sentences are canned sentences the wizard used in every conversation, for example to initialize the dialogue.

For the experiments, the two single datasets "NPC" and "CST", as well as a combined dataset called "ALL", are used. Unfortunately, from the original total of 4,313 utterances, many could not be used for the final experiments. Firstly, fragments are removed and only the utterances found by the parser to contain a valid predicate argument structure are used. After protocol sentences are taken out too, a dataset of 1,702 valid utterances remains.

Moreover, 292 utterances are annotated to contain no valid dialogue act and are therefore not suitable for the recognition task. Of the remaining utterances, 171 predicate argument structures were annotated as incorrect because of completely ungrammatical input. In this way, we

arrive at a dataset of 804 instances for the users and 435 for the wizard, summing up to 1,239 instances in total.

Different sets of features for training and evaluation are generated by the annotated data:

DATASET_Syn: All utterances of the specified dataset described via syntactic relation and context features.

DATASET_VNSem: All utterances of the specified dataset described via VerbNet semantic relations and context features.

DATASET_Syn_Only: All utterances of the specified dataset only described via the syntactic relations.

DATASET_VNSem_Only: All utterances of the specified dataset only described via the VerbNet semantic relations.

DATASET_Context_Only: All utterances of the specified dataset described via the context features and negation without any information regarding relations.

DATASET_Utterances_Context: The utterances of the specified dataset as strings combined with the whole set of context features without further relation extraction results.

DATASET_Utterances: Only the utterances of the specified dataset as strings. This and the last "Utterances" set serve as baselines.

The subsumed set of dialogue acts shown in table 5.1 constitutes the classification classes[1]

Evaluation is performed using cross-folded evaluation. All results in the experiments are given in terms of accuracy.

Results for the dataset "All" comparing the syntactic relations with VerbNet relations, as well as the pure utterances and context, are shown in table 5.3.

[1]Please note that the evaluation here does not include the social dialogue acts. At the time the evaluation was carried out, the social dialogue acts set was still under construction. However, the classification described in this chapter is nevertheless successfully used in combination with a rule-based approach (see chapter 3) for dialogue-act recognition in the overall system, including all social dialogue acts. However, future work shall certainly include a new evaluation incorporating all dialogue acts.

Utterance	Correct	Classified As
What do you think about this one?	request_info	propose
Let see what you have and where we can put it	request_info	request

Table 5.2: Wrongly Classified Instances

The best result is achieved with the syntactic information, although the VerbNet information provides an abstraction over the predicate classification. Both the set containing the VerbNet relations as well as the syntactic relations are much better than the set containing only the context and the original utterances. The dataset containing only the utterances could not reach 50%.

Although the experiments show much better results using the relations instead of the original utterance, the overall accuracy is not very satisfying. Several reasons for this phenomenon come into consideration. While it can, to a certain extent, be the fault of the classifying algorithm (see table 5.7 for some tests with a ROCCHIO-based classifier), the main reason might just as well lie in the imprecise boundaries of the dialogue-act classes: Several categories are hard to distinguish even for a human annotator, as you can see from the wrongly classified examples in table 5.2. Another possibility is the comparatively small number of total training instances.

Dataset	Accuracy
All_Syn	67.4%
All_VNSem	66.8%
All_Utterances_Context	61.9%
All_Utterances	48.1%

Table 5.3: Dialogue Act Classification Results for the “ALL” Datasets

For the NPC dataset the results are slightly better and much better still for the CST set, which is due to a smaller number (6) of dialogue acts: The dialogue act "PROPOSE", which is the act for showing an object or proposing a possibility, was not used by any user, but only by the wizard.

Dataset	Accuracy
CST_Syn	73.1%
NPC_Syn	68.5%

Table 5.4: Dialogue Act Classification Results for Datasets "CST" and "NPC"

To find out if one sort of feature is especially important for the classification, the training sets were reorganized to contain only the context features without the relations (All_Context_Only) on the one hand, and only the relational information without the context features on the other (All_Syn_Only and All_VNSem_Only). Results are shown in table 5.5.

Dataset	Accuracy
All_Context_Only	56.6%
All_VNSem_Only	53.5%
All_Syn_Only	50.8%

Table 5.5: Dialogue Act Classification Results for Context and Relation Sets

Table 5.5 shows that the results are considerably worse if only parts of the features are used. The set with the context feature performs 3.1% better than the best set with the relations only. Furthermore, the VerbNet semantic relation set leads to nearly 3% better accuracy, which may mean that the abstraction of semantic predicates provides a better mapping to dialogue acts after all, if it is used without further features that might be ranked as more important by the classifier.

Besides the experiments with the Bayesian networks, additional experiments are performed using a modified ROCCHIO algorithm similar to the one in (Neumann & Schmeier, 2002). Three different datasets were tested (see table 5.6).

Predicate	Items	Example
see-30.1	59	I would like to see a table in front of the sofa
put-9.1	74	Can you put it in the corner?
reflexive_ appearance-48.1.2	80	Show me the red one
own-100	137	Do you have wooden chairs?
want-32.1	153	I would like some plants over here

Table 5.6: The Main Semantic Relations Found in the Data Sorted by Predicate

Table 5.7 shows that the baseline dataset containing the utterances only already provides much better results with the ROCCHIO algorithm, delivering 70.1% which is much more than compared to the 48.1% of the Bayesian classifier. When tested together with the context features, the accuracy of the utterance dataset rises to 73.2% and, after including the relational information, even to 74.4%. Thus, the results of this ROCCHIO experiment also prove that the employment of the relation information leads to improved accuracy of the classification.

5.3 Out-of-Domain Classification

This section describes a probabilistic domain classification approach. Domain classification is a necessary step in a dialogue system's error-handling mechanism. Understanding errors can originate from errors in

Dataset	Accuracy
All_Utterances	70.1%
All_Utterances_Context	73.2%
All_Syn	74.4%

Table 5.7: Dialogue Act Classification Results Using the ROCCHIO Algorithm

the natural-language understanding (NLU) process, e.g., when a valid dialogue act for the utterance is known to the system, but could not get assigned because the linguistic analysis or the ARS of the utterance is erroneous. Errors can also occur if the system is not prepared to handle the incoming utterance, because it is out of the scope of the knowledge or target task of the dialogue system. These are so-called *out-of-domain utterances* (OoD utterances). Even if the linguistic analysis of an OoD utterance results in a perfect linguistic structure, the interpretation component cannot assign a valid dialogue act because the necessary knowledge for an interpretation is missing.

For an appropriate treatment of errors it is necessary to know the source of the error. If the error results from an NLU mistake, the system may decide to try a repair strategy. It can for example ask the user to rephrase the last utterance or try to re-establish common ground through asking other questions. This is the traditional way a task-based dialogue system reacts to understanding problems.

However, it makes no sense for the system to ask for a reformulation of the last utterance if the utterance is an OoD utterance and just cannot be understood by the system. The dialogue system should handle these errors in another way, rather than through a traditional repair.

Therefore, this section describes a new approach to out-of-domain classification.

Similar to (Lane et al., 2007) the basis for the recognition process is utterance topics. However there are some important differences. While in (Lane et al., 2007) the topic recognition process already is a classification process that assigns a probability to an incoming utterance for each predefined topic class, in the approach described in this chapter no predefined topic classes are needed. Topics are detected in an unsupervised data-driven way. Utterance topics are represented as fuzzy topic clouds consisting of ontology concepts. For this step, no training data is necessary, topics are not restricted by content or number, and they are retrieved for every utterance not just in-domain utterances.

In the second step the found topic cloud for an incoming utterance is then classified as in- or out-of-domain. This is similar again to the approach suggested in (Lane et al., 2007). However, the approach presented in this chapter only needs a set of in-domain utterances to work and no topic annotated data. The out-of-domain decision is based on a classification using the actual topic cloud of an incoming utterance and a topic cloud of the overall system. Classification itself is done by a verification-score algorithm that estimates the chances that the incoming topic cloud

are in-domain by comparing the two clouds. If the verification score is greater than a predefined threshold, the utterance is classified in-domain. The accuracy achieved for this approach is very good. For comparison, a Lucene[2] index is built from automatically retrieved topic clouds for the in-domain utterances. However, the Lucene approach cannot beat the cloud comparison.

5.3.1 Topic Detection

The first step of the domain classification is to detect the topics in the incoming utterance. An utterance is seen to contain an arbitrary number of topics, which are not organized and not restricted. Topic, as it is defined in this work, differs from the linguistic topic of a sentence. Whereas the linguistic topic can be defined as the constituent, a sentence says something about (Reinhart, 1982) and most often corresponds with the syntactic subject, for the described application, e.g., the object position of the syntactic structure, appears to be much more important. The topic in the described system does not correspond to linguistic constituents in a one-to-one relationship. Instead the system incorporates ideas originating from topic detection algorithms for text, which use a list of the most regularly occurring words in a paragraph as a vague topic description.

Topic detection results in a fuzzy cloud of ontology-class identifiers. For every utterance the algorithm tries to get as many valid class identifiers as possible. Class identifiers are superclass concept names originating from the used ontologies. In the prototype application the available ontologies are the cocktail domain ontology, a furniture domain ontology and WordNet. The knowledge bases used are described in chapter 3.

Topic detection proceeds in two cycles. Firstly, the algorithm does an ontology look-up of every word in the incoming utterance. Secondly, the system tries to retrieve topic identifiers matching the verb arguments of the utterance. The system prefers lemma information and only uses word surfaces if no lemma information can be retrieved. All found topic identifiers are added to one list of topics representing the topic cloud belonging to the incoming utterance. By using this procedure, class identifiers can occur multiple times in one topic cloud. The domain recognition process uses the multiple occurrences as an inherent quantification, naturally encoding more important key identifiers for the given input.

[2] http://lucene.apache.org/core/

To omit the integration of improper superclasses delivered by the knowledge bases, the algorithm only uses the highest super class that relates to the search term.

5.3.2 Evaluation

The out-of-domain classification was evaluated using the dataset *Eval*. The data is retrieved from experiment chat logs containing conversations between human users and the *KomParse* system, which was controlling a conversational barkeeper agent. 12 users participated in the experiments. The application and the dataset are described in detail in chapter 3. The data is segmented into utterances and manually annotated with a domain flag. Annotation was done by two independent annotators. If the annotators do not agree on how to annotate an entity, a third annotator's judgement was added.

The classification was evaluated using 10-fold cross-validation. The classification was compared to a classification using a Lucene index. In general the classification works very well. Due to the imbalanced data, including far fewer entities for task utterances than non-task utterances, the number of false positives, meaning task utterances recognized as non-task utterances was very small in both evaluations. The cloud classification gets very good results, without any false positives at all. The number of false negatives (non-task utterances classified to be task utterances) is a little higher with 7.4%. The evaluation statistic for the cloud evaluation is shown in table 5.8. The evaluation statistic for the Lucene evaluation is shown in table 5.9.

5.4 Conclusion

This chapter presents methods and approaches to three natural-language understanding challenges. Because they are shared in several of the software extensions described in this thesis, they are bundled into one chapter. The chapter describes an out-of-domain classification approach incorporating unsupervised topic detection and a novel dialogue-act recognition process incorporating linguistic input features.

The dialogue-act recognition (section 5.2) is a novel approach that uses syntactic and semantic relations as input features, instead of the traditional features such as n-grams of words ((Verbree et al., 2006), (Zimmermann et al., 2005), (Webb & Liu, 2008)). Different feature sets

Measure	Value
Total Number of Task Utterances:	149.0
Misclassified as Non-Task:	0.0 (0.0%)
Correctly Classified:	149.0 (100.0%)
Total Number of Non-Task Utterances:	1890.0
Misclassified as Task:	140.0 (7.41%)
Correctly Classified:	1750.0 (92.60%)
Recall:	0.52
Precision:	1.0
Accuracy:	93.13%
Error Rate:	6.87%
F-Measure:	0.68

Table 5.8: Results for the Out-of-Domain Classification Using Topic Clouds

are constructed via an automatic annotation of syntactic predicate argument structures and a manual annotation of VerbNet frame information. On the basis of this information, both the syntactic relations as well as the semantic VerbNet-based relations included in the utterances can be extracted and added to the feature sets for the recognition task. Besides the relation information, the features employed include information from the dialogue context (e.g., the last preceding dialogue act) and other features like sentence mood.

The feature sets have been evaluated with a Bayesian network classifier as well as a ROCCHIO algorithm. Both classifiers demonstrate the benefits gained from the relations by exploiting the information that was additionally provided. While the difference between the best baseline feature set and the best relation feature set in the Bayesian network classifier yields a 5.5% boost in accuracy (61.9% to 67.4%), the ROCCHIO set-up exceeds the boosted accuracy by another 1.5% , starting from a higher baseline of 73.2%. Based on the observed complexity of the classification task it is expected that the benefit of the relational information may turn out to be even more significant with more learning data. The out-of-domain classifier detects if incoming utterances are out

Measure	Value
Total Number of Task Utterances:	149.0
Misclassified as Non-Task:	10.0 (6.71%)
Correctly Classified:	139.0 (93.29%)
Total Number of Non-Task Utterances:	1890.0
Misclassified as Task:	340.0 (17.99%)
Correctly Classified:	1550.0 (82.01%)
Recall:	0.29
Precision:	0.93
Accuracy:	82.83%
Error Rate:	17.17%
F-Measure:	0.44

Table 5.9: Results for the Out-of-Domain Classification Using Lucene

of the knowledge domains of a dialogue system. Although the possibility to decide if an incoming utterance is out-of-domain is badly needed by error handling mechanisms, only few suggestions for out-of-domain classifiers have been proposed yet. The only exception known to the author is the topic-based approach described in Lane et al. (2007). The new approach presented in this chapter is also based on topic detection, but the classification decision is based on comparison with topic clouds of in-domain topics automatically processed from a set of in-domain utterances. Only a set of in-domain utterances and no topic annotated data is needed.

Topic detection itself is done using an unsupervised, data-driven approach. All words of an utterance are checked against the used knowledge bases to find matching concepts. Predicate argument information is used to weight the importance of found knowledge base concepts for the actual utterance. The result of topic detection for an utterance is a topic cloud that consists of an arbitrary number of knowledge base RDF URIs neither limited in size nor in the kind of concepts.

6 Multi-Threaded Dialogue Management

6.1 Introduction

The wide variety of dialogue systems available in research and industry suggests many different conversation-constitutive elements to describe conversation structure in dialogue management. As described in section 2.2, the baseline conversation-constitutive elements are the following: The smallest units are conversation steps or utterances (e.g., single actions); multiple (at least two) conversation steps produce a conversation sequence; and the highest-level units of conversation structure are "phases". A conversation contains at least one opening phase, one closing phase and one core phase. While opening and closing phases are frequently investigated and the structure of these phases is well known, the structure of the core phase is complicated to define.

This is a problem from the theoretical point of view as well as from a dialogue system developer's point of view, because further structuring of a core phase in sub structures besides sequences is badly needed. Several authors have proposed *topic-specific* groups of sequences in discourse analysis ((R. C. Schank, 1977), (G. Schank, 1981)) as well as in dialogue system development. However, topic-specific classification carries many disadvantages. Estimation of when a topic begins and when it ends is complicated and subjective. Also, several different topics may be negotiated at the same time and the definition of topic itself is vague. In automatic processing such as dialogue systems, the detection of topic change is very challenging. Moreover, from the system developer's point of view a topic-centered approach to conversation organization also necessarily means the non-reusability of described structures.

Another frequently used method in dialogue models of dialogue systems is to insert additional functional units inspired by goals and plans of the system and the user, such as "sub-goals" or "sub-tasks" ((Wong, Cavedon, Thangarajah, & Padgham, 2012), (J. Allen & Perrault, 1980)).

Goals and plans can be functionally described without any content such as topics. They can be theoretically reused for several different occasions and also for different dialogue systems in the same domain and task. In reality these sub-phase units are most often task-oriented, e.g., the Map-Task annotations (Carletta & Isard, 1996) use start point and end point in a map to structure and describe sub-phases.

This thesis uses the concept of *conversation threads* to denote the sub-phase structure in conversations. The thesis favors the term *thread* because it does not carry any task- or content-specific meaning. Similar to sub-tasks, the term does denote a subsumption of an arbitrary number of dialogue sequences which belong to a special sub-goal of the conversation. However, this sub-goal should be in aid of functional communication, and not contain task- or content-specific knowledge, but knowledge about the type of conversation only. A task conversation may consist of several threads such as "negotiate object", which can also be used for several different types of conversation.

From a dialogue system developer's viewpoint, dialogue threads offer a very modular way of implementing dialogue models. The addition of new threads is easy and threads can be developed autonomously by different authors, or learned automatically from data.

Moreover, a dialogue model based on dialogue threads offers a perfect infrastructure for handling *multi-threading support*. As (Brinker & Sager, 1989) point out, the realization of conversation goals often runs discontinuously in time. It is common behavior that people come back to "old" goals or negotiate several goals in parallel. The analysis of Wizard-of-Oz data described in section 3.4 (chapter 3) supports this observation. People tend to use several conversation threads in parallel, such as discussing which cocktail to drink and at the same time doing some small talk about the bar. Especially in text-based instant-messaging talk this seems to be a very frequent behavior. The phenomenon also occurs if humans not only engage in one kind of talk, but mix several types of conversation such as goal-driven and small talk conversations.

To achieve the goal of this thesis - the smooth integration of social talk into dialogue systems - conversation-thread modeling and multi-threading support is crucial. Social-talk sequences are, e.g., modeled and integrated as additional conversation threads in the dialogue manager (The actual models of social talk and the sequences for handling out-of-domain talk are described in depth in chapter 7 and chapter 8). This

chapter introduces a new dialogue management approach using a structured, graph-based dialogue model that encodes conversation threads and offers support for multi-threading.

Following is an outline of how this chapter is organized. Section 6.3 describes the implementation of the multi-threaded architecture. Section 6.3.1 describes the thread-selection algorithm. Section 6.3.2 explains the verbalization of thread change and section 6.4 reveals the results of the evaluation. Section 6.5 is a short conclusion.

6.2 Graph-Based Conversation Thread Models

In the approach presented in this thesis, a graph-based dialogue model is used for dialogue management. Graph-based dialogue management is still one of the most common ways to encode possible conversation structures. Especially in industry, graph-based dialogue management is very common. There are several reasons for that. One is certainly that graphs offer a simple overview of the developed dialogue flow. Since they belong to the structured models, the possible dialogue structures that are encoded in a graph-model can be easily understood. On the other hand, graphs that want to enable more flexibility in the way and order of dialogue quickly become complicated to read. Another reason in favor is that graph-based dialogues can usually be intuitively generated and non-experts can also very quickly learn to use authoring tools for graph-based dialogue management.

In the approach described in this thesis, conversation threads are implemented as supernodes of the overall finite-state automaton in the Harel state-chart notation. A supernode contains a sub-automaton of the overall automaton consisting of an arbitrary number of nodes and supernodes (Harel, 1987).

Definition 1 *A conversation thread S is a 5-tuple* $T = (States, Transitions, Variables, Types, Commands)$ *with*

$States(s) :=$ *a set of conversational states*
$Transitions(s) :=$ *a set of transitions between these states*
$Variables(s) :=$ *a set of variable definitions*
$Types(s) :=$ *a set of type definitions*
$Commands(s) :=$ *a set of command definitions*

The conversational states and the transitions are the main elements of the thread, defining which dialogue situations belong to this thread and to which situations they can lead. A thread also includes a set of commands accessing conversational behavior and data structures to contain thread-specific information (variables and types).

A dialogue thread is specified by the means of the conversational domain and the dialogue topic:

$$typ(domain), \in Types(s) \qquad var(topic) \in Variables(s) \tag{6.1}$$

The domain type depends on the domains available for the actual dialogue system. The dialogue topic contains the content of what the actual conversation is about and is set at runtime of the system[1].

Dialogue threads are generally functionally motivated, implemented at development time, and underspecified in content. This allows for initiating dialogue threads multiple times without implementing them multiple times in the graph. Threads are then further specified at runtime according to the current content, mainly the topic. The topic is used to distinguish several dialogue threads of the same kind between each other, for example in a furniture-selling dialogue there may be two active threads with the goal "selectObject", one with the topic "carpet" and one with the topic "sofa", because the user wants to buy a carpet which matches a sofa and therefore discusses both objects together.

6.3 Multi-Threading Support

Dialogue threads are managed by the dialogue manager. The dialogue manager controls the threads' life cycles. Threads can be in three different conditions: active, paused, or inactive. The active thread is the current dialogue thread, which determines the system's behavior and offers the local search space for the understanding of the user's utterance. As long as no environment modification occurs, the system will follow the conversation flow encoded in the active dialogue thread.

[1]Please note that the notation of dialogue topic is very vague even in research and a discussion of the subject may fill whole books. For the understanding of the presented application it is important to know that the topic as it is used here is not equivalent to the linguistic topic. It may for example be a physical object that is being negotiated over. That is why the topic in the presented application can theoretically be nearly everything that can be expressed in one single concept. To have at least a little control of the naming of topic concepts, ontology concept identifiers from the integrated domain ontologies and WordNet are used.

The active thread can be terminated in two ways: Either through interruption in the middle of the thread conversation or after the part of the conservation which is encapsulated in the thread is completely over. Both cases result in the thread becoming inactive. Another way to deactivate a thread is to pause it. A paused thread is not completely inactive, but is marked as still to finish by the dialogue manager.

The dialogue manager therefore handles lists of inactive and paused threads as well as one active thread. Such a multi-threaded dialogue manager has to offer an architecture that can handle multiple active and inactive threads, and which is able to switch between them, and a logic for selecting appropriate dialogue threads. Because empirical research has found out that changes of conversation threads by the system can easily become confusing (Heeman, Yang, Kun, & Shyrokov, 2005) for the user, especially if the system does not provide a discourse marker to notify the change to the user (Yang et al., 2008), the described approach also provides verbal markers to signal switches to the user. Figure 6.1 shows the multi-threaded architecture.

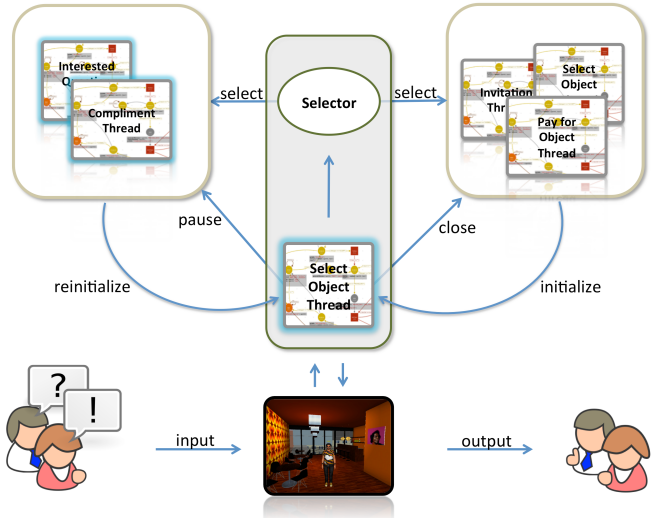

Figure 6.1: Multi-Threaded Dialogue System

6.3.1 Selection of Dialogue Threads

There are two different scenarios in which the dialogue manager needs to select an appropriate dialogue thread:

- User Initiative: Selection of a dialogue thread according to a user input.
- System Initiative: Selection of a dialogue thread without a user input.

While in the first scenario the dialogue manager needs to detect the correct dialogue thread to which the incoming user utterance belongs, in the second scenario the dialogue manager should select an appropriate dialogue thread to continue with, e.g., if a dialogue thread is finished and the user pauses speaking. In both cases the dialogue manager needs to select one thread among all the active and inactive threads. Because the dialogue logic is inherent in the dialogue graph, the selection of a dialogue thread is part of the graph as well as the natural-language understanding (NLU) component. Algorithm 1 describes the main parts of the algorithm.

User Initiative

Selection of an appropriate thread for the incoming user utterance depends on the results of the natural-language understanding process. The first search space for a matching thread is the currently active thread. All general utterances such as “yes” and “no” are interpreted in the context of the currently active thread. It is not until the active thread fails to offer an interpretation of the incoming utterance that the system searches other threads to match. The natural-language understanding components provide a dialogue act, and if possible domain and topic information belonging to the input, too. The thread selection is primarily based on the dialogue act and domain information, for example an incoming compliment dialogue act belongs to a compliment dialogue thread. If several possible threads are found for the given dialogue act and domain, and topic information is found for the input, the topic is used to disambiguate the possible threads and select the best one. This can be the case if the user’s utterance is ambiguous; for example, the user can express her wish for a special piece of furniture in a furniture sales scenario. In the set of active threads there is a paused thread for discussing a special piece of furniture, e.g., a sofa. In the context of this

thread the utterance can, e.g., overwrite the current wish for a sofa. On the other hand the utterance can express the wish for a completely new piece of furniture such as a table. In this case the correct behavior for the dialogue manager would be to initialize a new thread for selecting a table and come back to the sofa thread again later.

System Initiative

In the second scenario, the dialogue manager needs to select a dialogue thread to continue the dialogue after another dialogue thread was finished. This is the system initiative scenario. The system has to decide which of the paused dialogue threads should be reactivated or which inactive threads will be newly activated. The presented approach uses two mechanisms for this selection: Time information (for how long an active thread has been paused) and importance information (how important the thread is for the conversation). The importance values of threads in the described system are manually defined by the author at the development time of the dialogue graph. However they can be seen as hooks for values originating from an additional user model or task model. The thread-selection algorithm will use both types of information to select a thread if importance values are available. If no importance values can be determined, time information is used by itself. As an example, one can imagine an agent and a user in a barkeeper scenario. They just finished a dialogue thread about discussing a cocktail to drink. There is a lull in conversation. The system needs to select the next thread to talk about. It has the possibility of coming back to a thread that had already been started about the weather, but was interrupted by the cocktail-discussion thread and is paused, or it can select a completely new conversation thread, e.g., some small talk about the bar or a payment dialogue thread. The system gathers time information for the active threads and importance values for active and inactive threads. Although we have one already active thread in this case, the weather thread, the system would select the payment thread, because it probably has the highest importance value.

6.3.2 Verbalization of Thread Change

Empirical research shows that, when a system changes the conversation, it can easily become confusing (Heeman et al., 2005) for the user, especially if the system does not provide a discourse marker to notify the change to the user (Yang et al., 2008). Therefore, the presented system

Algorithm 1 Thread-Selection Algorithm

```
procedure SELECTTHREAD(lastTurn)
    if lastTurn.TurnType == UserTurn then
        dialogueAct ← recognizedDA
        domain ← recognizedDomain
        thread ← GETTHREADFORDAANDDOMAIN(dialogueAct,
domain)
    else
        aThreads ← GETACTIVETHREADS()
        if aThreads ≠ null then
            sThreads ← SORTBYIMPORTANCEANDTIME(aThreads)
            thread ← POP(sThreads)
        else
            pThreads ← GETPOSSIBLETHREADS()
            sThreads ← SORTTHREADSBYIMPORTANCE(pThreads)
            thread ← POP(sThreads)
            if thread == lastThread then
                thread ← POP(sThreads)
            end if
        end if
    end if
     return thread
end procedure
```

generates so-called "bridging utterances" as markers for the user. Bridging utterances consist of two parts. The first part is a general reference to the newly activated or reactivated thread. The second part is to repeat the last utterance that was made by the system, if existing. It was found in some preliminary tests that verbalization of the topic in the dialogue thread appears more natural and smooth than verbalization of the dialogue goal or the domain, which are too abstract and not unique for the users. Hence, a typical first part of the bridging utterance could be "So, what about the TOPIC?", where "TOPIC" is the placeholder for the actual topic of the thread, as opposed to "Let's come back to the small talk", which would be a bridging utterance using the domain

or "What about the discussing an object to sell?" using the dialogue thread's goal.

The second part of the thread-change verbalization is particularly important for reactivated dialogue threads and makes sure that user and system are at the same state of the conversation. The dialogue memory, which is part of the application, is asked for the last system turn belonging to the dialogue thread and a system output is newly generated using the system turn. Due to the application system's generation approach, in most cases the system turn does not consist of a ready utterance but specifies a simple semantic representation of an utterance that can lead to several variants in the surface form. Thus, the system may repeat the last utterance made by the system but avoids seeming too repetitive.

6.4 Evaluation

The multi-threaded dialogue manager was evaluated on the basis of user experiments. In the experiments, the users talked to a conversational agent, which is a front-end to the described dialogue system. The agent is a barkeeper agent in a virtual world, who can talk about cocktails, gossip and some typical small-talk topics, such as personal information. Each conversation was logged to a file. The resulting dataset *Eval* is described in 3.

The data was manually annotated with dialogue-thread information. Each utterance which was found to contain a dialogue thread function was annotated. Thread functions include the opening of new threads, the reinitialization of paused threads, and the selection of threads according to user utterances. Table 6.1 shows the thread work functions.

The evaluation shows that the thread selection by the system works very well: The dialogue manager did not select a single wrong thread for reinitialization and no wrong thread according to the result of the input interpretation component. There were few errors in initializing new dialogue threads, mainly originating from behavior that was not implemented at evaluation time. The selection algorithm did not incorporate the number and time of user rejections to suggested threads. Therefore, on some occasions the system suggested a dialogue thread that the user had already rejected three times.

Figure 6.2 shows the division of the system's reaction to dialogue threads initialized by the user. There are three different possibilities: The system selected the correct dialogue thread, the one the user wanted to initialize; the system's input analysis did not understand the user's

Acronym	**Explanation**
Dialogue threads initialized by the system	
S-I	System-initialized thread
S-I-C	Threads correctly initialized by the system
S-I-W	Threads wrongly initialized by the system
Dialogue threads reinitialized by the system	
S-R-T	System-reinitialized thread
S-R-T-C	Correctly reinitialized thread
S-R-T-W	Wrongly reinitialized thread
Dialogues threads initialized by the user	
U-I	User-initialized thread
S-S-C	Threads correctly selected by the system
S-S-W	Threads incorrectly selected by the system
U-I-NU	Dialogue-thread initialization not understood by the input analysis
U-I-UT	Not-known dialogue thread initialized by user

Table 6.1: Categories of Annotated Dialogue Thread Functions

utterance; and the dialogue thread the user wanted to initialize was unknown to the system. One possible outcome did not occur, which is that the input analysis delivers a semantic representation, but the dialogue manager selects an incorrect thread. The diagram shows that selecting the dialogue thread that matches the user utterance is no problem for the system, but depends heavily on the correct work of the input analysis. Since the dialogue management is developed in a modular way, the contents for input analysis and interpretation are the responsibility of the thread authors. In total, 23 of 102 user attempts to initialize new dialogue threads were not understood by the input analysis (25.48%). Additionally, 106 user utterances in an ongoing thread without any thread function were not understood by the input analysis. Nevertheless, all of the non understood thread initializations and utterances were answered with the correct selection of the uncertainty dialogue thread by the dialogue management. The uncertainty thread offers vague answers or clarification dialogues.

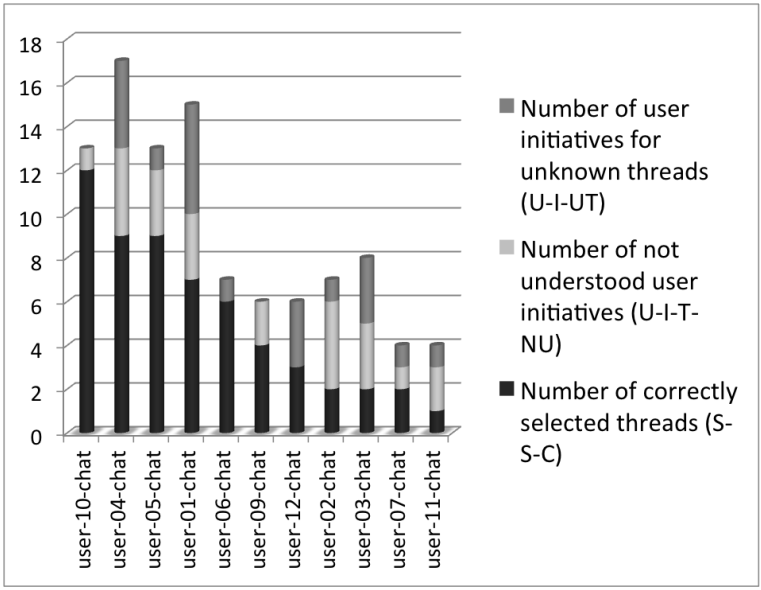

Figure 6.2: User Initialized Dialogue Threads and System Reaction

Figure 6.3 presents the results of system initiative in selecting inactive dialogue threads to activate. The diagram shows that there were some minor errors in the selection process. However, 13 of the 16 errors are due to the already mentioned missing behavior in the selection algorithm, which did not consider the number of rejections already uttered by the user. This behavior has been added to the current version of the algorithm.

In table 6.2, performance in selecting the correct paused thread to reactivate is shown. The system reinitialized 63 paused threads. All of them were correct.

Although these evaluation results are very promising, it is necessary to note that in the evaluation experiments the user did not make use of the interwoven multi-threaded dialogue possibilities. In contrast to the Wizard-Of-Oz experiments carried out in the same application and scenario (see chapter3), the users in the experiments used embedded

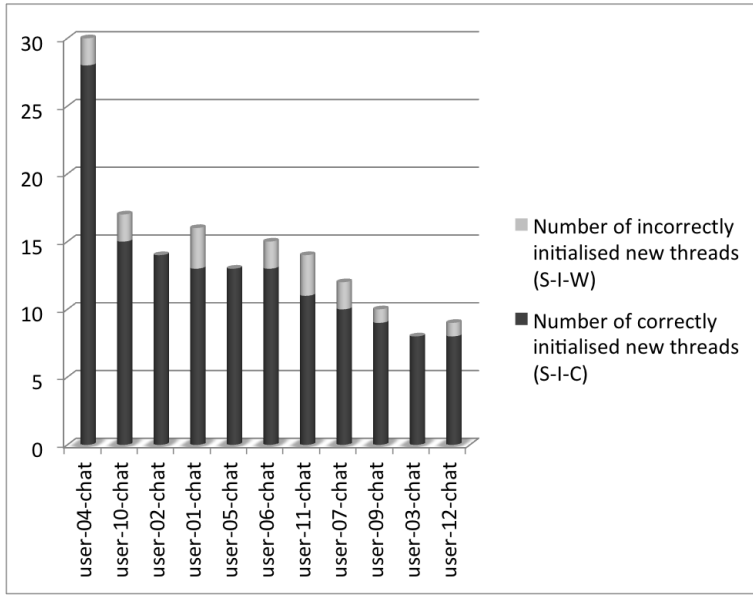

Figure 6.3: System-Initialized Dialogue Threads

dialogue threads only. This may result from the comparatively low performance of the input-understanding component, which causes the users to simplify the structure of the conversation. However, we expect the system to perform very well if confronted with interwoven threads too, because there is no technological difference in the handling of embedded and interleaved threads. The manager treats all possible threads in the same way.

6.5 Conclusion

In this chapter, I have described a novel approach to dialogue management incorporating the concept of *dialogue threads*. Dialogue threads are conversation-structuring constituents on the level between dialogue sequences and dialogue phases. Dialogue threads are containers for

Conversation	Correctly reinitialized (number/percentage)
user-01-chat	16 (100%)
user-02-chat	14 (100%)
user-03-chat	8 (100%)
user-04-chat	29 (100%)
user-05-chat	13 (100%)
user-06-chat	15 (100%)
user-07-chat	12 (100%)
user-09-chat	10 (100%)
user-10-chat	17 (100%)
user-11-chat	14 (100%)
user-12-chat	9 (100%)

Table 6.2: Total Number and Percentage of the Correctly Reinitialized Dialogue Threads

one or several dialogue sequences which are grouped according to a conversation-functional goal (see chapter 2).

Many studies (e.g., Brinker and Sager (1989)) and also the data analysis reported in this thesis show that dialogues between humans are not linear but very flexible in the order of topics and parts of the conversation. Speakers may, e.g., pause the current part of the conversation, initiate another one and come back later to the paused one. Although this is a common behavior in every conversation, it is particularly frequent if social talk is included and it is also observable in conversations between humans and machines. However, the ability to handle such flexible dialogue behavior in dialogue systems is very rare.

The dialogue manager presented in this chapter has a thread-based dialogue model and is able to switch between threads, pause threads, activate new threads, and reactivate paused ones. The dialogue manager uses extended state graphs to encode the dialogue model, in which threads are specified as supernodes in the sense of Harel's state charts. Supernodes therefore encapsulate autonomous conversation behavior specific for single conversation threads and can be combined in a modular way. This offers a very flexible graph-based architecture, which is nearly as modular and powerful as a plan-based system.

An evaluation of the thread selection (section 6.4) shows that the selection algorithm works very well and that the modular thread-based approach can successfully be used to generate a smooth conversation flow without the necessity to include other heavy computing approaches to dialogue management. There are two different cases of thread selection: selection reacting to user initiative and selection for system initiative. While in the first scenario the systems hat to understand the thread desired by the user, in the second scenario the system has to select a thread without user initiative. Naturally, the thread selection according to the user initiative depends on the results of the natural-language understanding component. The algorithm worked very well, but 23 of 102 user initiated threads were not correctly identified because of errors in the NLU component. During thread selection for system initiative only minor errors occured and in the reactivation of threads the algorithm did not select any wrong thread.

As we have seen empirical studies have found out that conversation changes by a machine quickly become confusing for the user ((Heeman et al., 2005), (Yang et al., 2008)). Therefore, the presented multi-threaded dialogue manager also supports the verbalization of thread change (section 6.3.2). Thread changes are marked by bridging utterances which verbalize the topic of the new thread.

7 Small Talk

7.1 Introduction

Social talk, or "small talk", is often perceived as unsophisticated chit-chat in which content exchange is irrelevant and negligible. Following this definition, small talk represents the opposite of task-driven talk. On the other hand, several studies have detected the "task" of small talk not to lie in knowledge negotiation, but in the management of social situation. In the early 1920s Bronsilaw Malinowski already introduced the term "phatic communion" to denote a kind of talk that "serves to establish bonds of personal union between people" (Malinowski, 1949, page 316). This establishment of social contact is the primary goal of phatic talk and dominates or even excludes the exchange of conversation content.

Social talk is one of the main strategies between human conversation partners to establish a friendly and comfortable social relationship. Especially in situations in which people meet for the first time, small talk is a common instrument to warm up the relationship or to overcome pauses.

Several authors have described why it is important for conversational agents to engage in social talk. Social talk can be used to ease the situation and to make a user feel more comfortable in a conversation with an agent (T. W. Bickmore & Cassell, 2000). Also, when applied to real-world environments, agents are nearly always confronted with small-talk utterances and have to react to them in an appropriate way (Kopp et al., 2005).

Although some agent systems provide small-talk conversations, no systematic (computational) model of small talk has been developed so far. The macro-structure of small-talk conversations in particular has not been intensely studied. Referring back to the different levels of dialogue structure explained in 2, namely differentiating between dialogue steps, dialogue sequences, and dialogue phases, this means that a more specific

description of the relationship between sequences and phases is needed. While in other types of conversation an additional level of "sub-phases" or "sub-goals" is often used to bundle several sequences belonging to one "goal", similar methodical suggestions for social-talk conversations are still missing (Spranz-Fogasy & Spiegel, 2001).

Social-science theories offer analyses of social talk, but only few concepts and ideas have been taken over to dialogue systems. One example is the small-talk sequence found by Schneider (Schneider, 1988). The Schneider sequence was integrated into dialogue systems ((T. Bickmore, 1999), (Endrass et al., 2011)), but the use of just one sequence for all small-talk phases in a conversation leads to an unnatural dialogue. For a natural conversation more flexibility in social talk is needed. A deeper analysis of the intentions in small-talk utterances, as well as the rules which combine utterances, questions, and feedback to small-talk sequences, can provide the knowledge needed to develop a more appropriate model.

This chapter describes a new structured and knowledge-driven model of social talk learned from data annotated with a new set of dialogue acts for social talk[1]. The taxonomy of dialogue acts is functionally motivated and based on the social-science theory of "face", described by Erving Goffman (Goffman, 1967). The taxonomy is completely new and aims at covering the gaps that exist for social acts in the existing dialogue-act sets. The set of dialogue acts is validated by the annotation and analysis of a corpus with small-talk conversations.

This data is analyzed and a novel set of function-oriented communication patterns for social talk is retrieved that further describes and structures the core phase of small-talk conversations according to conversation threads. These conversation threads constitute "sub-phases" of social talk. The chapter does not suggest a linear order of sub-phases that is valid for every small-talk conversation, because there is no order of conversation threads in general and especially not in social talk (see chapter 6).

[1] As stated before, conversation itself and particularly social talk greatly depends on culture. However, this work does not deal with culture in an explicit way. The participants of the experiments are mainly from Germany, Poland, and Spain. A model of culture-related differences and the integration in conversational agents is described in (Endrass et al., 2011).

Moreover, a computational model of social talk for integration into dialogue systems is automatically learned from the data. The model encodes the small talk communication patterns resulting from the analysis and is integrated into the agent architecture described in chapter 3.

This chapter is organized as follows: The next section (section 7.2) describes the dialogue-act set for social talk starting with section 7.2.1 and section 7.2.2, which introduce the social science work on "face" by Erving Goffman. Section 7.2.3 describes the taxonomy of small talk dialogue acts and section 7.2.4 describes the small talk sequences found in the corpus data. Section 7.3 explains the analysis of the data and the resulting communication patterns for social talk. The computational model and the process of learning the model are described in section 7.4. Finally, section 7.5 summarizes the chapter and suggests necessary further work.

7.2 Dialogue Acts for Social Talk

7.2.1 Erving Goffman: Face

The dialogue acts presented in this paper are inspired by the work of Erving Goffman, an American social scientist. The main concept in Goffman's work about social interaction is "face". In his work, face is an "image of self delineated in terms of approved social attributes" (Goffman, 1967, page 5). Face means the perception of the self of both interactors. Every person has an image of herself in a social context. In direct communication the faces of both participants need to be protected and supported. This procedure is called "face work". Goffman regards the face of an individual as a sacred entity in the modern secular world (Goffman, 1967). Therefore, the "face work", in which the interactors manage their faces, is comparable to religious rites. The two main ritual face work patterns are the "presentational" and the "avoidance" rite. Avoidance rites will occur if one of the participant's faces is threatened with damage, e.g., through a malicious insult. Presentational rites on the other hand are used to support the face of one of the interactors. The rituals are organized in a dialogical way. This means the smallest possible rite consists of two steps: the "sacrifice" of the giver and an acknowledgement from the recipient. Accordingly, a presentational ritual consists, at the very least, of the giving of a positive act and an appreciative answer.

Goffman identifies two main interpersonal interaction policies for face work: the "supportive interchanges" and the "remedial interchanges" (Goffman, 1971). Interchanges are sequences of possible conversational turns consisting of gestures, glances, touch and verbal utterances. This work focuses on supportive interchanges in which the participants want to positively maintain their faces.

Several agents with small-talk ability are inspired by the social science work on face. For example, the afore mentioned REA system determines the measure of face threat in her own planned utterances (T. W. Bickmore & Cassell, 2001). If the system identifies the planned utterance to be too threatening for the user's face (e.g., a question about money), the agent engages in small-talk sequences until the measures of interpersonal closeness and user comfort are high enough to continue with the dangerous utterance.

7.2.2 Dialogue Acts & Dialogue Sequences by Goffman

Erving Goffman's analysis of interchanges can be regarded as a model of social-dialogue acts and their combination in sequences. Goffman himself uses the terminus "act" to refer to a single verbal or nonverbal action (Goffman, 1971).

By the late 1970s, Werner Holly had already interpreted Goffman's interactions in a linguistically motivated formalization (Holly, 1979). Following Goffman's distinctions of sequences, Holly describes two different categorizations of supportive utterances. One category is built by the means of shared interpersonal topics and another one according to function. The first group (interpersonal topics) contains utterances of sympathy and interests, the second, polite offers. Holly follows Goffman by distinguishing "ratifications" and "access" acts (see table 7.1). Another distinction is the utterance's target face. Possible values are the speaker's own face as well as the listener's face.

From the dialogue system point of view, Goffman's classification seems problematic in various aspects. Group one, for example, appears to be comparatively large and unstructured. Moreover, the distinction between topic and function is very vague. It seems unclear why congratulations are grouped by function and compliments by topic. A compliment can easily be seen as a ratification step; for example, a compliment about a new haircut. This means that parts of the groups categorized by schema two are also a subgroup of a group built by schema one, but are not specified in group one.

Interpersonal Topics	Function
Group 1: Utterances of sympathy and interest interested questions, compliments, polite answers, downgrading	**Group 3: Ratification rituals** congratulations, condolences, acknowledgement of changed personal situation
Group 2: Polite offers invitations, welcoming, introducing somebody	**Group 4: Access rituals** greeting, good-bye, initializing and closing sub-dialogues

Table 7.1: Categories of Acts Used in Positive Sequences by Goffman and Grouped by Holly

It is also complicated to generate valid sequences from the mentioned groups, because initiative and reactive acts are not distinguished. Goffman describes only a very abstract sequence for supportive interchanges: A supportive act answered by a supportive act. In general it would be preferable to have one categorization for supportive acts and not two that partly overlap. The taxonomy presented in this chapter provides a clear distinction between the function and the topic layer.

7.2.3 A Taxonomy of Cooperative Social-Dialogue Acts

In this section a taxonomy of dialogue acts is presented that can be used for cooperative social talk. All dialogue acts are integrated into one functionally motivated taxonomy. The dialogue acts are inspired by Goffman's work, but categorized according to two main types of face work: Requesting support of the speaker's face and providing support for the addressee's face.

The taxonomy includes dialogue acts that have primary social functions are social as well as dialogue acts that can be used either in small talk or in other conversation domains. This matches the observation from Coupland & Coupland regarding social talk (Coupland et al., 1992). They discovered that every utterance carries a special degree of phatecity. Dialogue-act classes that are primarily social acts are: Compliments, self-compliments, self-criticism, invitations, self-invitations and some forms

Utterance	Social Talk	Task
We only have a limited choice of sofas.	-	Inform
I have been living in Berlin for over 20 years now.	Inform	-

Table 7.2: Multidimensionality

of interested feedback. Dialogue acts that are not domain-bound are: Inform, provide opinion, request information, request opinion. The latter are global functions that can be used in task talk as well. In DiAML and DIT++, this class is called a "general-purpose function" which can be assigned to all dimensions. The presented dialogue acts are intended to be an addition to existing dialogue-act sets. To maintain multidimensionality an additional dimension of social talk could be the solution. The following examples[2] show the dialogue act "Inform" in task talk and the additional dimension social talk.

Figure 7.1 shows the dialogue acts organized according to the two classes.

Request Face Support

The category "Request Face Support" contains all dialogue acts which express a request for support of the speaker's face. Utterances expressing a request for face support imply the demand to reinforce, strengthen or accept the presented face of the speaker. If speaker A informs hearer B about his opinion towards something, A expects B to display interest in his information through a face-supporting act; for example, a verbal or mimic utterance.

But this is not the only level on which face work occurs. In human-human conversations, face work originates from several layers, from which only a few are verbal. Additionally, there are politeness constraints which define conversation rules. An omitted response to a question, for example, means a face threat to the speaker's face.

[2]All examples are taken from the corpus of Wizard-of-Oz experiment conversations WoO1 and WoO2 described in 3

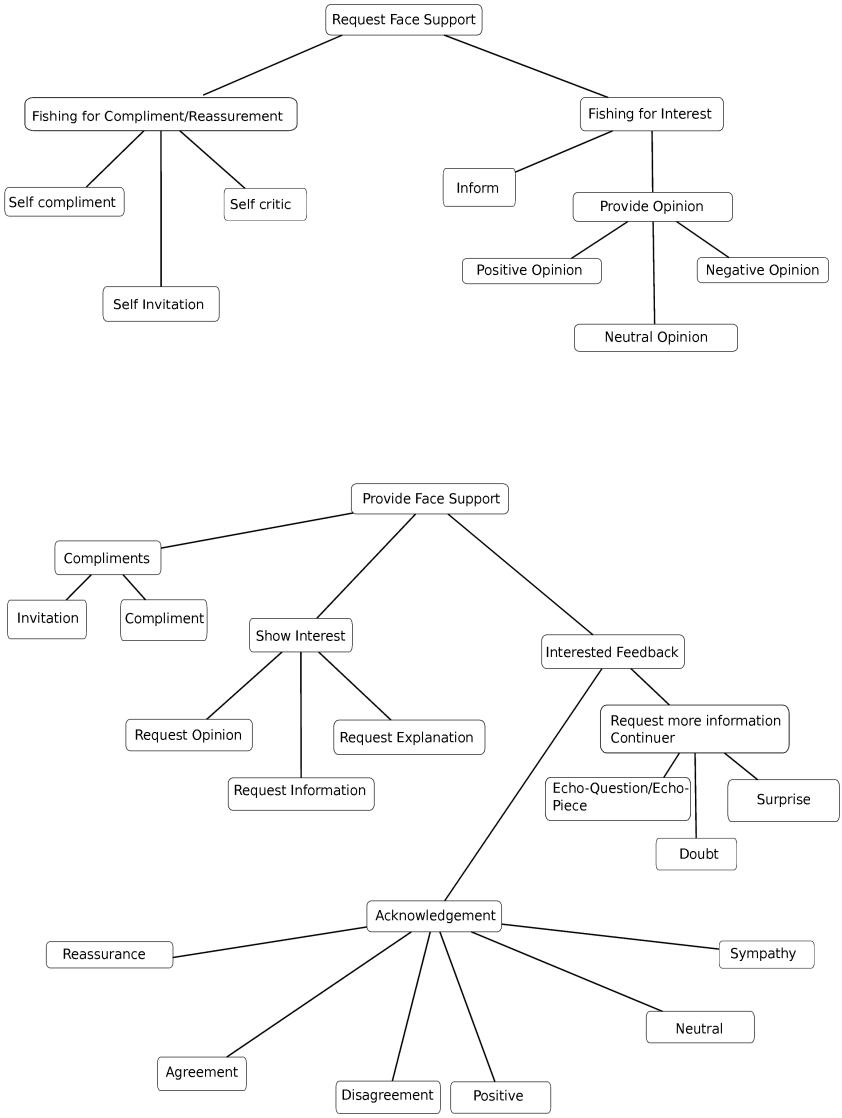

Figure 7.1: The Small Talk Taxonomy

Dialogue Act	Short Definition	Example
Self Compliment	With a self-compliment the speaker praises herself	I am really good at guessing names.
Self Invitation	A self-invitation is an utterance which allows the speaker to do something that otherwise would need an invitation from the interlocutor.	*May I call you Alex?*
Self Critic	With a self-criticism the speaker criticizes herself or her own behavior.	*I am too old for that.*
Inform	An inform provides information to the hearer.	*I am from Poland.*
Positive Opinion	An utterance that expresses a positive opinion contains the speaker's positive opinion towards something.	*It is one of the most beautiful cities I know.*
Negative Opinion	A negative opinion contains a speaker's negative opinion towards something	*The organization is not very good.*
Neutral Opinion	A neutral opinion act contains the speaker's neutral opinion towards something.	*That is long.*

Table 7.3: Request Face Support Dialogue Acts

In a conversation consisting of a sequence of statements, the responding act itself as well as closeness to the semantic content and topic of the preceding utterance determines the grade of face support. These observed mechanisms follow a cooperative rule described in the relevance maxim formulated by Herbert Grice (Grice, 1975). A's face therefore not only depends on B's positive reaction towards his information, but on B showing any reaction at all. However, it may not be possible to capture these conversational rules through dialogue acts and they are not part of the work presented here, which deals with explicit social-talk utterances.

The group is subdivided into the two subclasses **"Fishing for compliment"** (group A) and **"Fishing for interest"** (group B). Figure 7.1 shows the taxonomy for request face dialogue acts. Dialogue acts are ordered hierarchically with growing specificity towards the leaves of the tree. The dialogue acts in group A should be assigned to utterances that expect a reaction of either compliments, reassurances, or invitations.

Utterances implying group B dialogue acts are reaching out a kind of interested feedback. Whereas the dialogue acts in group A in general carry a strong face request, the acts in group B vary in the degree of intended face work, and content may play an important role. Table 7.3 shows the dialogue acts with examples from the data.

Provide Face Support

The second category provides dialogue acts which strengthen the hearer's face. It contains three subclasses: **"Compliments & Invitations"** (group C), **"Show interest"** (group D) and **"Interested feedback"** (group E). Strong face support can be expressed through compliments and invitations (group C). Invitations may refer to physical or verbal actions, or other actions that offer a more intimate relationship. Other face-support acts concern the expression of interest in the other person (group D and E). Interest can be "factual",which is mainly expressed through request for information, explanations, and opinions (group D). Group E on the other hand subsumes various forms of interested feedback without introducing any new factual content. This class can not be used as an initial step. The group is divided into "continuers", dialogue acts that aim at expressing interest while at the same time encouraging the other interlocutor to keep talking, and "acknowledgements" which do not need an answer.

Figure 7.1 shows the taxonomy of dialogue acts in this category. Table 7.4 gives examples and a short definition for every dialogue act.

A Word on Topic

In addition to the functional layer of an utterance, topic plays an important role in the interpretation of small-talk utterances. The dialogue act "Request Information", for example, normally is a member of the category "Provide Face Support", but it can occur in category "Request Face Support" as well. If the topic is related to the speaker, the dialogue act is assigned to the "Fishing for interest" group. Similarly, the dialogue act "Inform" is classified as "Show interest" if the topic is related to the addressee. Moreover, topic often determines the expected reaction, especially for the dialogue act "Inform". Information about a serious injury should result in a different reaction from the hearer as information about a new flat.

Dialogue Act	Short Definition	Example
Compliment	All compliments and kind words.	*You are very intelligent*
Invitation	Invitations to verbal or physical actions.	*I want to buy you a coffee.*
Request Opinion	With a request opinion, the speaker asks for the hearer's opinion.	*Did you like the movie?*
Request Information	With a request information the speaker asks for information that is not an opinion.	*Do you have many customers?*
Request Explanation	Request explanation is the dialogue act for all requests for further explanation.	*Why?*
Echo-Piece	An echo-piece is used to express interest in the part of the utterance that is repeated	*You are on holiday*
Doubt	An utterance that expresses doubt.	*You think so?*
Surprise	The dialogue act for expressing surprise.	Ah, really?
Reassurance	An utterance that is meant to ease somebody contains a reassurance act	Don't bother yourself.
Agreement	An expression of agreement.	*You are right.*
Disagreement	An Expression of disagreement.	*That's not true.*
Positive	Positive feedback that includes admiration or joy.	*Wow!*
Neutral	Neutral acknowledgement.	*OK.*
Sympathy	A sympathy feedback is the response to negative or sad information.	*I can understand that.*

Table 7.4: Provide Face Support Dialogue Acts

7.2.4 Data Verification

The aim of the verification is to show if the found dialogue-act set is applicable to conversations with virtual conversational agents and what

modifications to the scheme may be necessary. To verify the appropriateness of the dialogue act taxonomy, the dialogue acts are applied to a corpus containing data from the two Wizard-of-Oz (WoZ) experiments *WoO1* and *WoO2*. The datasets contain task-driven and small-talk conversations.

In the first experiment resulting in the dataset *WoO1*, the wizard controls a furniture sales agent who supports the user in buying pieces of furniture for an apartment. The scenario is task-driven and small talk is only optionally initiated by the experiment participant. This data consists of 18 dialogues containing 3,171 turns with 4,313 utterances. In the second experiment, which resulted in dataset *WoO2*, the participants are German-language learners from various countries having a conversation with a barkeeper, who is controlled by the wizard. The participants are explicitly briefed to conduct small talk and the wizard himself initiates a small-talk conversation if the user does not. This corpus contains 12 dialogues with 1,477 utterances. A more detailed description of the datasets is given in chapter 3.

The final corpus used for data verification of the dialogue acts consists of the dialogues in *WoO1* and *WoO2*, from which small-talk utterances were found. This corpus contains 4,161 utterances from which 990 are categorized to predominantly fulfill small-talk functions. These utterances were annotated with dialogue-act information by two annotators of which only one has previous knowledge of dialogue systems and dialogue research.

The inter-annotator agreement value between the annotators shows that the annotation scheme is an adequate first description of the data: From the 990 utterances belonging to small-talk sequences, they annotated 772 with an identical dialogue act. This results in a kappa value of 0.741. The majority of confusions occurs within the fine-grained set of feedback acts, e.g., between neutral and surprised feedback. Another source of confusion is discriminating between request opinion and request information.

7.3 Analysis of Social Talk Conversations

Using the social-talk data described in 7.2.4, additionally annotated with sequence information, an analysis of small-talk dialogues was done. The analysis results in a specific description of the structure of social talk and communication patterns for small-talk conversations organized in conversation threads, sequences and actions. *Actions* are the smallest

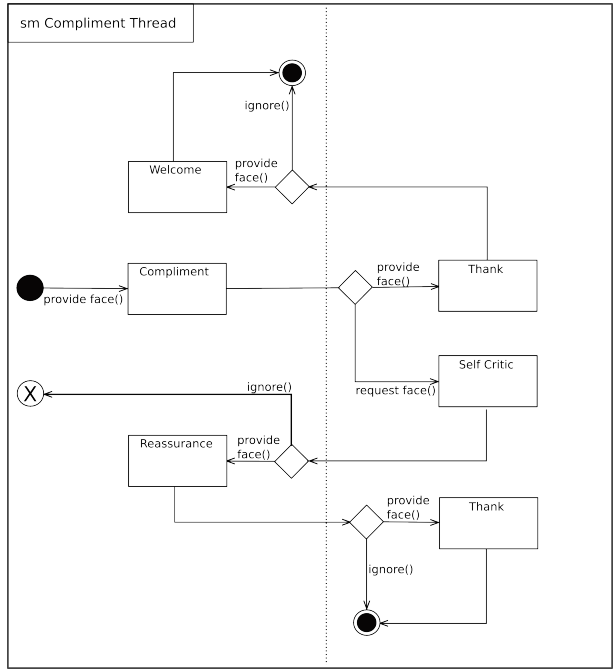

Figure 7.2: The Compliment Thread

communication units of dialogue, most often utterances, which contain an intention and a (spoken) action by a conversation participant. Following (Brinker & Sager, 1989), a dialogue *sequence* is defined as a succession of such actions from one initiative turn to the next initiative turn. *Threads*, on the other hand, are higher-level units, encapsulating one or more sequences which belong to one special functional conversation goal (Please see chapter 2 for detailed information).

The data contains 314 valid sequences of social talk consisting of 990 small talk dialogue actions. Although the most simple sequence found in the data is the adjacency pair (Schegloff, 2007), several different sequences were found. The average length of a sequence in the data consists of three dialogue acts, with the shortest unit being only one turn and the longest sequence consisting of five turns.

Initial Dialogue Act	Frequency	Initial Dialogue Act	Frequency
request information	183	request opinion	8
inform	54	invitation	7
opinion	25	request explanation	7
compliment	15	self criticism	2
others	13		

Table 7.5: Dialogue Sequences Sorted According to the Initial Dialogue Act

As you can see from table 7.5, most small-talk sequences are initiated by a "Request Information" dialogue act (183 sequences). Request information is the dialogue-act class for various kinds of interested queries. The second most frequently used class is "Inform", the class for providing information (54), followed by "Opinion", the class for expressing an opinion (25) and "Compliments" (15). "Others" are other domain dialogue acts (e.g., task dialogue acts) that are answered by small-talk utterances.

Sequences are organized into dialogue threads. Figure 7.2 shows the compliment dialogue thread containing the communication pattern for uttering and reacting to compliments. States encode dialogue actions, which are described by dialogue acts from the face taxonomy explained in section 7.2.3. The actions depend on the decision of the participants to react to a request or to provide face. The dotted line indicates the scopes of the two different speakers. The thread contains two valid final states and one termination state. The termination occurs if speaker A decides to ignore the request face action *self-critic*, because this is a strong request for face support and ignoring this request strictly means a violation of the rules of a cooperating conversation.

A selection of sequence distribution in the data is shown in table 7.6. The percentage describes the frequency of this particular sequence compared to the absolute number of sequences in a thread.

Compliment - Thanks (26%)
Compliment - Thanks + Compliment - Thanks(6%)
Compliment] - Surprised Feedback + Thanks (13%)
Request Information - Inform (46%)
Request Information - Inform - Feedback (15%)
Request Information - Accept/Reject + Inform (13%)
Request Opinion - Provide Opinion (37%)
Request Opinion - Provide Opinion - Feedback (12%)
Request Opinion - Feedback + Provide Opinion - Feedback (12%)
Self-Critic - Feedback Reassurance (100%)
Self-Invitation - Invitation (50%)
Self-Invitation - Accept (25%)
Inform - Feedback (18%)
Inform - Request Information - Inform (18%)
Provide Opinion - Feedback (24%)
Provide Opinion - Request Opinion - Accept/Reject/Uncertain (12%)

Table 7.6: Some Selected Small Talk Sequences

7.4 Computational Model

From the annotated data, graph-based models of small talk were automatically generated for use in computational set-ups. The program used for the generation translates the sequences found in the corpus conversations into a graph format, which can be processed by a dialogue system's dialogue manager directly. The program tries to merge as many identical conversation steps in different paths as possible to omit redundancy in the resulting graph. The results of this processing step are graph models for all different small-talk threads containing their sequences, such as a compliment thread, a self-critic thread, an inform or an provide-opinion thread.

The graph-generation algorithm first stores all dialogue sequences starting with the same dialogue act. In the next step the first version of the graph is constructed from the first dialogue sequence found. Afterwards, in iterating steps, the graph gets augmented with more and more possible transitions and states according to the other sequences in the

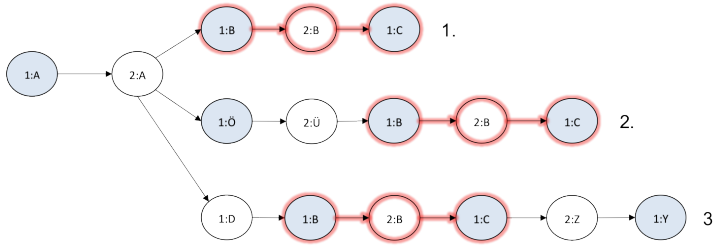

Figure 7.3: Merging Using a Suffix Tree

data. If the algorithm founds a conversation step to be equal, it tries to merge the corresponding parts of the graph. After the initial merging process, the algorithm uses a suffix tree to further merge identical suffix paths. Figure 7.3 shows a graph ending in three different paths. Identical graph parts are marked in red. The merging algorithm would use the suffix tree to merge path one and path two. Path three cannot be merged with path one or path two, because the resulting graph would generate a path that is not validated by the data; path one, for example, would be extended with an invalid ending consisting of the last two steps of path three.

Figure 7.4 shows a simplified example of graph generation from the data. In the picture, the states contain system utterances in the meaning of dialogue acts: “th” stands for “thanks” and “com” for compliment. Edges contain conditions. Conditions can be one of the following: Incoming dialogue acts or probabilistic values. If no condition is given, the edge is an epsilon edge. Additionally, a starting and an end state are given. In the first step, a graph is generated from the first example conversation containing only two turns of the conversation participants: An initial compliment and a thanks in return. The dialogue examples contain one turn per line, starting with the dialogue act, followed by the numerical id of the speaker and the actual utterances in the turn. The graph looks like this accordingly: From the starting state, one edge with the condition of an incoming compliment is generated, which leads to a “thanks” state. This edge is an epsilon edge, because at this point in the generation process no other transitions are possible. Another epsilon edge leads to the end state of the graph.

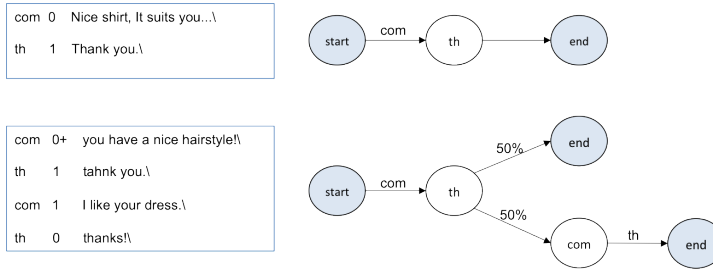

Figure 7.4: Graph Generation Example

In the next step, the next dialogue is used to extend the graph. The second dialogue contains four turns: An initial compliment, a thanks in response directly followed by a compliment by the same speaker, and a thanks by the first speaker. The algorithm will adapt the new sequence possibilities to the existing graph. Instead of the epsilon edge originating from the existing thanks state, two probabilistic edges are generated, offering transitions to two different valid states, either the end state or a "th" state in which the second compliment is uttered. Both probabilistic edges contain the value 50%. In further iteration steps, the algorithm will adjust the values of the probabilistic edges according to the frequency of occurring states. Lastly, a new edge, with the condition being an incoming thanks is added to the compliment and results in an end state.

The result is one graph containing all valid ways to lead a compliment initial dialogue originating from the data. Through this processing, 13 different small-talk graphs were generated, with different numbers of states and edges, starting with four nodes in the smallest graph and up to 109 nodes in the biggest graph.

The newly generated graphs were manually post-edited. In the data, many examples were found in which statements or questions are not answered. This phenomenon may be caused by the nature of the human-computer interaction, in which a violation of the necessity to answer does not have any negative consequences. This behavior is acceptable for human users. However, the barkeeper agent is designed to be a cooperative conversation participant. We therefore modified the graphs accordingly: While the possibility for the user to terminate a small-talk

dialogue is preserved in most states, the agent will never terminate an ongoing conversation.

7.4.1 Language Content

The described small-talk models are abstract sequences of dialogue acts without any specified language content. In addition to the dialogue acts, only abstract conditions on the identity of topics are defined. The models also need language content to be usable. Small-talk models and language content are knowledge driven and specific. This means that for every topic the system should be able to take part in a discussion in using one of the small-talk models, the language content specifying what exactly can be understood and generated for this topic needs to be defined.

During learning of the social talk models, possible language content is learned simultaneously from the same data. As described in chapter 2, language content is needed for the natural-language understanding as well as for the answer generation. The input analysis delivers a result which is the baseline for the input interpretation. In the dialogue system described in chapter 3 that is the test bed for the social-talk technologies presented in this thesis, input interpretation results in a representation of the meaning of the utterance, primarily the dialogue-act information but potentially topic information and domain as well.

Several possible ways to recognizing a dialogue act from language input are used in the input interpretation: The statistical dialogue act classification described in chapter 5 and a rule-based dialogue-act recognition. Rule-based recognition and statistical recognition incorporate linguistic information automatically extracted from the incoming utterance by the input analysis and the minimal dialogue context represented by the information about the preceding utterance. The input analysis delivers linguistic information from various levels, such as part-of-speech tagging, tokenization, named entity recognition and parsing. The most specific representation which can result from the analysis is a predicate argument structure. Both recognitions use as much linguistic information as they can get from the analysis. The rule-based approach, moreover, optionally contains assignments from surface forms of utterances directly to a dialogue act and topic. These rules are especially helpful for highly idiosyncratic phrases and words, such as in greetings.

Since the small-talk models are based on dialogue-act sequences that account for topic identification, dialogue act and topic information is

available from the training data. During the learning of small talk models, each utterance as well as the annotated dialogue act and topic information are also used for learning of language content. The current utterance gets analyzed by the component which constitutes the input-analysis component of the final dialogue system. If the analysis can retrieve useful linguistic information, a new rule is generated that assigns the automatically retrieved linguistic representation, in combination with the information about the preceding utterance, to the annotated dialogue act. If no linguistic information can be retrieved a rule is created which assigns the surface form of the utterance straight to the dialogue act and optional topic. Surface forms can contain regular expressions. Since the data is used to learn small talk models, the domain of the utterances is always considered to be small talk, except for a predefined list of exceptions such as "yes" or "no", which belong to the "general" domain.

Rules are encoded in XML. The following example shows a rule assigning a predicate argument structure to a pragmatic meaning. The enclosing tag is the rule element. Inside the rule element the "semantics" group contains the left-hand side and the "pragmatics" group the right-hand side of the rule. The left-hand side specifies the found predicate argument structure including predicate, first argument, and second argument, as well as sentence mood (question type). Sentence mood is detected using heuristics. The right-hand side contains the resulting pragmatic information, namely dialogue act, domain, and topic.

```
<rule>
  <semantics>
    <pred type="word" path="like"/>
    <arg1 type="word" path="SPEAKER"/>
    <arg2 type="word" path="bar"/>
    <questiontype val="statement"/>
  </semantics>
  <pragmatics>
    <dialogact type="Compliment"/>
    <domain type="smalltalk"/>
    <topic type="wordnet:synset-barrom-noun-1"/>
  </pragmatics>
</rule>
```

Learning of language content results in 99 unique rules, varying in the amount of successfully retrieved linguistic information and covering approximately 36 topics. These rules are used as a basis for the further manual development of new rules. At the time of writing, the current

state of the SOX social-talk component contains around 123 rules for 49 social-talk topics in the input analysis.

Answer generation in the described system is template based. Templates are similar to the analysis rules: They assign possible answers to meaning objects containing dialogue acts, topics, and domain. They differ in syntax: Templates assign meaning objects directly to surface forms and allow for several different surface forms to avoid repetition. In addition to the meaning, templates consider information about the last preceding utterance either as surface form or as predicate argument structure similar to the input analysis.

The same mechanism used for learning input analysis rules is used to learn some initial answer templates. The following example shows a typical generation template.

```
<rule>
  <pattern>
    <da>REQUEST_INFO_YOU</da>
    <topic>wordnet:synset-residency-noun-1</topic>
    <uturn>
        <utterance>I live in .*</utterance>
        <pred>live</pred>
        <arg1>SPEAKER</arg1>
        <arg2>"+"</arg2>
    </uturn>
  </pattern>
  <template>
        <utterance>How do you like it?</utterance>
        <utterance>Do you like ${arg2}?</utterance>
  </template>
</rule>
```

The enclosing element is the rule element. The rule element has two children: The pattern and the template element. The pattern element constitutes the left-hand side of the rule. In the left-hand side, constraints over dialogue act, topic, and the last preceding user utterance ("uturn") are defined. This template can, e.g., only be used if the dialogue act is "REQUEST_INFO_YOU". The uturn element contains either surface forms of the last preceding user utterance in the "utterance" element or a predicate argument structure, or both. Surface forms can contain regular expressions. The template element contains the right-hand side of the rule. In the "utterance" element, surface forms of possible answers are defined. Surface forms can incorporate slots in which

content from the preceding user utterance can be integrated. In the example, the second surface form contains slots marked with "$arg2". This slot is filled with the value of the second argument of the last utterance, if the surface form is selected for answer generation.

48 generation templates are generated for the SOX component during learning of small-talk models, covering 25 topics. This initial set has been extended to 53 templates for 28 small-talk topics at the time of writing.

7.4.2 Integration Into Dialogue Systems

The learned graph-based models are integrated into the *social-talk component* of the SOX toolkit. The component can be integrated into a dialogue system either separately or in combination with the other modules from the toolkit. Generally, the component provides the learned models described in section 7.4, grouped in various small-talk conversation threads. As mentioned before, the agent's dialogue manager is thread-based and very modular: Small subparts of the dialogue are encapsulated in single super nodes. This behavior is possible due to the support of Harel's state charts. The supernodes are designed to contain parts of the conversation which roughly correspond with dialogue goals. The supernodes get activated according to the user's utterance or through the agent's own initiative in a data-driven way (see chapter 6).

The component can either work on its own, without communicating with the overall dialogue system, or in a shared mode. The shared mode necessarily requires a deeper understanding of the dialogue-system architecture (see chapter 2), the development of shared knowledge between the overall system and the small-talk component (such as a memory having access to all components selecting threads, and a list of shared safe threads and topics). Although the shared mode could be the more complicated version, it is also the mode of choice, because it guarantees consistent system behavior.

For a completely shared operating mode, selective integration of the component's single small-talk threads is enabled. To use this method of integration, the conversation structures in the overall dialogue manager need to be identical with the thread formalization. In the *KomParse* dialogue system (chapter 3), for example, all conversation structures are implemented as graphs organized in threads. The main graph includes several conversation threads. The main graph is developed with the SceneMaker tool's GUI editor ((Gebhard et al., 2003) (Mehlman, 2009)). The GUI was extended to allow graphical generation and integration of

conversation threads. The ready-made small-talk models can be loaded as additional conversation threads from a context-menu. After loading, the conversation threads can be added to the existing graph by drag-n-drop and initialized according to the constraints defined by the authors of the embedding dialogue framework.

7.5 Conclusion

This chapters deals with modeling social talk for integration in dialogue systems. Although the importance of integrating social talk in dialogue systems has already been reported by many people ((T. W. Bickmore & Cassell, 2000), (Spranz-Fogasy & Spiegel, 2001)), few solutions have been suggested so far. Many systems integrate parallel chatbots or other workarounds to enable social talk. However, these are insufficient ad-hoc solutions, carrying many drawbacks. For a better integration of social talk, more knowledge about social utterances and social-talk structure is needed. This chapter therefore provides an analysis of social talk based on a new dialogue-act set for social talk inspired by the social-science work of "face" by Erving Goffman. The analysis provides insights into the structure of social talk which are also supported by computational models of social talk that can be integrated into dialogue systems.

In the chapter I present a new dialogue-act taxonomy for social dialogue acts according to the social-science theory of face (section 7.2.3). The taxonomy is split into two main groups: One group for dialogue acts which primarily fulfill requests for face support for the speaker's face and one group for dialogue acts which provide face support for the hearer's face. The dialogue-act set is validated on a corpus containing small-talk conversations from Wizard-of-Oz experiments resulting in a kappa value of 0.741 for inter annotator agreement.

The dialogue act annotated data is analyzed regarding the possible structure of small-talk conversations and social talk conversation threads (section 7.3). The resulting model contains valid communication patterns for small talk conversations. This chapter, therefore, not only presents a usable model of social talk but also provides new insights on how the core phase of small-talk conversations are organized by the means of social-talk threads. However, the chapter does not suggest a linear order of social threads to constitute small talk, since in the baseline data no linear sequence of dialogue threads is found. This correlates with results from several studies not only for small talk specifically but for conversations

generally, which all state that the succession of conversation sub-phases is not linear and not ordered (Brinker & Sager, 1989).

To further test and use the results of the data analysis, a structured computational model of small talk is automatically learned from the annotated data (section 7.4). The model is graph-based and maintains the thread-oriented structure of the conversations. The chapter describes the generation algorithm and the graph-based model as well as its integration into a dialogue system. The learning process resulted in a model containing 13 different small-talk threads, which can be integrated into dialogue systems in various ways.

8 Uncertain Answer Module

8.1 Introduction

In the preceding chapters, tools that can handle small talk and multi-threaded conversations were described. These are necessary components to enable social talk in dialogue systems. Unfortunately, a system that signals social-talk capabilities to the users will be confronted with a lot of unknown topics and social-talk content, so-called "out-of-domain" input. Particularly, if the system is a comparably new machine, data for social-talk conversations might not be sufficient.

Out-of-domain utterances (also "out-of-application" utterances) as understood in this thesis are utterances targeting content that is out of the knowledge domain of a system. Considering an example dialogue system that can talk about bus schedule information and the weather, the knowledge domains of the system would be the bus schedule and the weather. All utterances which belong to these two knowledge domains are in-domain utterances. All other utterances are out-of-domain. In-domain utterances can be processed by the system: The interpretation process of the system (see chapter 2) can assign a valid interpretation representation to the utterance such as a dialogue act and a topic. If an utterance cannot be processed by the natural-language understanding (NLU) component of the system even thought it is in-domain, the utterance may be "out-of-grammar". These are utterances which should be comprehended by the system, but the NLU component cannot analyze them because the component is insufficient. To successfully interpret an utterance the system needs knowledge about the domain the utterance belongs to. If knowledge is missing, as in the case of out-of-domain utterances, the incoming utterance cannot be interpreted. Although the input analysis may successfully assign a semantic representation, the system will not be able to assign a correct meaning in the context of the

dialogue to the found syntactic representation. Instead, the interpretation will result in an understanding error, either a misunderstanding or a non-understanding. While misunderstandings occur if the natural-language understanding component assigns the wrong interpretation to an incoming utterance, in the case of non-understandings, the system cannot assign an interpretation at all. The example dialogue system mentioned before may, for example, produce a misunderstanding if confronted with a question about a train schedule information instead of the bus schedule, and a non-understanding if confronted with a compliment about its intelligence.

Following this definition it is not possible for a dialogue system to "understand" an out-of-domain utterance, because it could never successfully assign a valid and appropriate interpretation result to an utterance out of its knowledge domains. Although a syntactic representation might be processed, the system needs domain knowledge to provide an interpretation. In case of dialogue acts as a result of interpretation process, the system needs to, for example, be trained with a dataset including all known dialogue acts. If the training includes dialogue acts that are specific "out-of-domain" dialogue acts, the system actually already has knowledge about these dialogue acts. This means that utterances, which should be assigned such a dialogue act by the interpretation process, can strictly speaking not be out-of-domain anymore, since the system obviously already knows enough about the utterance to correctly interpret it. The system can only guess if an incoming utterance may be out-of-domain or not, but it will never be able to assign a valid interpretation.

In an empirical study (Bohus & Rudnicky, 2005) discover nearly 20% of non-understanding errors are caused by out-of-domain utterances. Thus, this study shows that this group of out-of-domain errors is the biggest error group in non-understandings, outnumbered only by errors originating from ASR level (62%). The system used in the study is a task-based system for conference-room reservations. It seems very likely that, in a conversational system, errors originating from out-of-domain utterances are even more frequent, since the system encourages the user to use small talk, which is much more unpredictable, but the system will never be able to know about every knowledge domain. Thus, in addition to the sophisticated knowledge-driven approach to small talk described in chapter 7, a system supporting social-talk conversations needs a component to handle incoming out-of-domain utterances.

However, an appropriate handling of understanding errors accounting for the out-of-domain origin of the errors has not attracted much

attention in research. Some of the latest research, especially with empirical focus, has suggested new strategies to handle non-understandings without targeting a repair ((Henderson et al., 2012), (Bohus & Rudnicky, 2005), (Skantze, 2003)), but most dialogue systems still use simple recovery strategies such as "Asking the user to repeat" or "Asking the user to rephrase" in the hope that a paraphrase of the original utterance can be understood by the machine. Other systems simply tell the user that the input was not understood ((Bohus, 2007), (Jokinen & McTear, 2009), (San-Segundo et al., 2000), (Komatani & Kawahara, 2000)). While for task-based systems ignoring the input or asking for repetition may be an acceptable behavior, using these strategies in more conversational systems may lead to irritated and frustrated users. (Henderson et al., 2012) is aware of the fact that conversational systems need other recovery strategies than task-based systems, but they still miss the importance of the differentiation between out-of-domain non-understandings and in-domain non-understandings, which may in fact be repairable by rephrasing strategies.

But, especially for out-of-domain errors, the common error-recovery strategies are not recommendable. The system will never be able to understand an out-of-domain utterance, not even if it asks for rephrasing. In fact, the dialogue may go from bad to worse if the system insist on understanding a task-related utterance where there is none.

A rare example of a system explicitly dealing with out-of-domain errors is described in (Patel et al., 2006). In this work, in the context of virtual characters, the authors describe a system which uses eight different classes of possible out-of-domain inputs such as "Question makes no sense" or "Question is about specific human characteristics". They use different classifiers to assign one of 55 in-domain classes or one of the eight out-of-domain classes to an input. Their evaluation proves that the explicit handling of out-of-domain errors significantly improves the system. However, their approach has some drawbacks. Firstly, it is not easy to use outside the application domain. It is also not developed to explicitly handle utterances which were not understood. For most of the classes, understanding of parts of the content is necessary. The classes are not designed to blindly react to all kinds of incoming out-of-domain utterances in an content-abstract manner.

This chapter introduces a new set of 25 dialogue strategies which can be used to react to out-of-domain utterances in a conversational way without any classification of the input in a content-dependent class. The

basis for the strategies is the investigation of strategies that humans apply to similar problems. Unfortunately, it is very complicated to get data for hidden non-understandings from human-human interaction. Usually, human speakers signal understanding problems by initiating a repair dialogue. If they hide a non-understanding it is impossible to tell from the outside that a non-understanding occurred. The only thinkable case would be a hidden understanding error which is revealed in the following conversation due to follow-up misunderstandings, but there is no explicit data for these cases and it would be very complicated to generate this data artificially. Therefore, various similar sources are exploited for usable strategies and described in this chapter.

The strategies are incorporated into a computational tool called the *uncertain-answer module*, which selects and realizes reactions to out-of-domain utterances. The system is able to react to out-of-domain utterances in a conversational way. The detection of the understanding errors lies in the responsibility of the embedding system. The *uncertain-answer module* is an independent SOX-toolkit component that can be used separately and in combination with the other tools. It consists of a strategy-selection module, a strategy realizer and a memory that stores all selected strategies and topics as well as how they were realized and delivered to the invoking system. The interfaces to the system are: The input interface, which delivers the particular utterance as well as results of the linguistic analysis from the overall system to the component; the output interface, which delivers a ready to use answer string from the component back to the system; and the memory interface, which enables communication between the overall system memory and the component memory for e.g. the alignment of already used topics.

This chapter proceeds with a detailed description of the new strategies and the different sources used for retrieving the strategies in section 8.2 and 8.3. Section 8.4 describes the computational tool and how it integrates (section 8.4.1) and realizes (section 8.4.2) the strategies. Section 8.5 gives a conclusion.

8.2 Sources for Strategies

To gather as many strategies as possible, several resources of potential strategies were exploited. The following paragraphs will explain the sources and the types of strategies found.

The strategies are retrieved from the following different sources:

Chatbot Data Some of the strategies are inspired by data belonging to the freely available AIML (Wallace & Bush, 2001) chatbot ALIZE[1] and my own work regarding chatbots using syntactic and semantic information ((Klüwer, 2009), (Klüwer, 2011a)).

User Questionnaires Some strategies originate from user questionnaires, in which the users should give an answer to partially given input.

Wizard of Oz Experiments In a Wizard-of-Oz experiment with a barkeeper agent (see chapter 3 for a description of the application and the experiments), the users initialized out-of-domain utterances, which are answered by the wizards. Although the wizards always understood the input, some abstracted strategies can be taken from this data.

Theoretical Work Theoretical psychology work on the communication behavior of hearing-impaired people has described some examples of strategies for hiding understanding errors.

8.2.1 Chatbot Data

Chatbots are based on a simple stimulus-response pattern-matching algorithm, without normally incorporating linguistic knowledge, knowledge about the world, and dialogue machines to manage the conversation structure. They rely on huge amounts of data describing pattern-template pairs.

Although the authors of chatbot data may not be aware of the fact that they use special strategies to handle input that is not understood, they unconsciously apply strategies to handle unknown input. In many cases the only input information a chatbot has to help it decide what to say next is only the incoming utterance. Since natural language is unlimited in generating possible input, the authors cannot foresee an answer to every input. They have to use cues in the utterance to generate an answer for every input or just deliver a default answer which has no relation to the input. Cues are often parts of the input such as single keywords, just the start of the utterance, or regular expressions for surface strings.

What to say, when only a marginal part of the utterance was understood, was partly inspired by the authors of the free ALICE chatbot

[1] http://alicebot.blogspot.de/

data. Several example pattern/template pairs in the ALICE data show that the data authors, probably unknowingly, try to get hold of significant syntactic and semantic information to which they can refer to in the answer, such as parts of the verb argument, mainly the subject and objects, utterances starting with a pronoun, and other indicators of sentence mood. Unfortunately, they are bound to the insufficient methods described (such as regular expressions) in order to find this information. This observation corresponds to the results from my earlier research in which an A.I.M.L. chatbot is enhanced with semantic information encoded as Robust Minimal Recursion Semantics (Copestake, 2007). The inclusion of linguistic information could significantly reduce the number of necessary patterns for a chatbot ((Klüwer, 2009), (Klüwer, 2011a)). This general strategy of using linguistically relevant elements to refer to is integrated into the set of strategies for verbal content.

8.2.2 Psychology Work regarding Hearing-Impaired People

Another source of inspiration for the set of strategies is literature from the field of psychology research regarding the psychological condition of deaf people. Several authors have stated the problems many hearing-impaired people have in accepting their handicap. Because of social disrespect and lack of understanding, they try to pretend they are not hearing impaired; for example, to cover their handicap they avoid wearing hearing-aid devices. Moreover, the authors report on people who are very successful at covering their non-understandings in conversations. Krug and Claußen (1949) state that some hearing-impaired people in his survey were very good in leading the conversation. Their speaking time in contrast to other conversation partners is very high and they rarely hand over the turn to another person. If another person takes the turn anyway, they pretend to listen for a short while and then interrupt with a phrase such as "But listen to this..." and continue to dominate the conversation. Thereby they do not have to confess their understanding problems.

Another trick reported in Krug and Claußen (1949) as well as in Bircher-Müller (1997) is the so-called "yes-man". Hearing-impaired people, who want to hide their understanding errors, tend to answer using several forms of agreement, as well as nods and interested looks. They cannot understand what was said, but they pretend to understand through agreement and acceptance.

Bircher-Müller (1997) and Schreiber (2000) also mention the skills of some hearing-impaired people to generate sentences that are always appropriate, but they do not give concrete examples.

8.2.3 Wizard-of-Oz Experiments

The Wizard-of-Oz experiments *WoO1* and *WoO2* in the barkeeper and furniture-selling scenario described in detail in chapter 3, deliver some insights into the handling of out-of-domain utterances by human wizards. Although the person observed in the experiment was the user not the wizard, the wizards were not briefed for special handling of out-of-domain utterances. Therefore, their reaction to such utterances can be seen to carry strategies in how to deal with them.

The experiments were carried out by five different wizards, of which only three were confronted with out-of-domain utterances in the conversation. The experiments and especially the experiments with the furniture sales agent were designed as task-bound or mixed task and small-talk conversations, and the total number of 36 out-of-domain utterances in this scenario is not very high. Additionally, the main difference for this data is that the wizards of course understood every single word of the incoming utterance, so that they could answer the utterances without any difficulty. However, there are still some interesting findings in the comparatively small sample data. Firstly, it is interesting to note that all three wizards handled a large number of the utterances by just ignoring the input (33%). Except for one conversation, all users seemed to accept this behavior: They neither insisted nor commented on it. Another interesting fact is that, in nearly all cases where the wizards answered the out-of-domain utterance, they immediately continued the conversation through switching to a known domain afterwards. In only three cases did the wizards respond to an unknown domain. Again, this behavior did not seem to lead to a negative attitude against the wizard. The conversation just continued with the users showing no frustration, anger or some comparable feeling. In general, one can say that the results from the experiments confirm two main assumptions. The first one is that ignoring utterances is acceptable, at least to some degree. The second one is that switching to another domain after answering seems to be common behavior for humans in a wizard role at least. Moreover, the wizards adopted ways to switch back to another known domain. They either generated a whole sentence to verbalize the switch, inserted a switching marker at the beginning of the new utterance (such as “so”, “now”) or

just continued with a new utterance in another domain without signaling the change.

8.2.4 User Questionnaires

User questionnaires are an important source of finding strategies humans use when confronted with partial or non-understandings. To fill out the questionnaires, the participants should imagine being invisible assistants to a barkeeper in a virtual game world. In the scenario the participants have to envision, they are in the virtual bar eavesdropping on the conversations between the barkeeper and his customers. The conversation is carried out through text-based chat. Every now and then the communication pipeline between the two will stop working and only pieces of the utterance the customer typed are delivered to the game. The barkeeper and the participant herself can see the broken input only. The barkeeper is unable to deal with this understanding errors. He no longer reacts. But the participant, being the invisible listener, can jump in and say something for the barkeeper instead. The customer does not realize that it is the participant speaking instead of the barkeeper. The participant can so rescue the conversation. If the participant manages to come back to one of a set of predefined "safe" topics, the test-chat function will start working properly again. The general goal of the participants is to be entertaining and nice, so that the customer will come back to the bar again.

With this scenario in mind the participants filled out the questionnaires. The questionnaire consists of a list of tables each containing short dialogues between the barkeeper and the customer. The last turn in the dialogues is always a broken input. The participants are requested to come up with an answer to the broken inputs and to make a bridge utterance to a known topic. Table 8.1 shows an example of the user questionnaire.

The example in table 8.1 contains a small conversation between the barkeeper and the customer in which the customer orders a drink. The conversation ends with the broken input from the customer "Ok, how do you...". In the next row the participant should write down how she would respond.

The questionnaire aims at substituting a complex Wizard-of-Oz experiment in the virtual world. It may be possible to set-up an experiment similar to the experiment described in (Skantze, 2003), in which the input

Dialogue 1	
Dialogue	Costumer: I would like a Vodka Martini and some crisps Barkeeper: Ok, One moment please. Costumer: Ok, how do you...
Your answer:	

Table 8.1: Example from the User Questionnaire

coming from the user needs to be substituted by the results of a natural-language understanding component. However, that's very challenging technically. The results of the dialogue system's NLU component are semantic and syntactic representations, even for incomplete utterances. To enable understanding of these representations to a human experiment participant, it would be necessary to translate it back to a format readable by humans and it is unclear how one could do that automatically in a satisfying way. In addition, the existing system needs to be essentially changed to allow a human operator to jump into the processing pipeline. Moreover, it is not guaranteed that the particular case we are interested in, meaning a partial understanding of the NLU component which leads to an understanding error, will occur often enough in many different appearances during the experiments.

On the other hand, the parameter of disposable time the users have to think about a response does not seem to be critical for the scenario. Interesting and funny answers are even better for developing an entertaining barkeeper. This means the results from the questionnaires are not natural human-human dialogue strategies, but may deliver strategies humans would like to hear.

Overall, the questionnaire contained eight dialogues ending with a partial input from the customer. The partial input differs in the amount of linguistic information available, from single words to a subject-verb or verb-object structure. Some of the partially delivered utterances are recognizable as wh-questions, yes/no-questions, unknown questions or declarative sentences. Two are not classifiable according to sentence mood. The whole questionnaire can be found in the appendix.

Five participants filled out the questionnaire, resulting in 37 answers to incomplete input and 37 bridges to safe topics. This is short of the full 40 answers (five participants multiplied by eight dialogues) because one participant refused to answer three of the dialogues. From the 37 given answers, most contain strategies that are already known from other sources. An strategy that is often used is the ignore strategy (15 occurrences). However, a new type of strategy used by the participants is the ignore strategy in combination with a strategy that can be described as "keep talking about last safe topic". The clear marking of the understanding errors is used only once. Another strategy commonly applied by the users is to combine the bridge and the answer in one utterance, as in the following example:

(8.1) **Customer:** "I would like a Vodka Martini please."
Barkeeper: "Sure, one second."
Customer: "....cats..."
Participant: "Cocktails are often named after animals. Do you want anything to drink?"

8.3 Uncertain-Answer Strategies

This section describes the full set of conversational strategies extracted from the sources described in the preceding section 8.2 and formalized into a unified topology.

Six main strategies are extracted from the strategy investigation:

Ignore The strategy *Ignore* is to ignore an incoming utterance that is not understood. This strategy originates from the Wizard-of-Oz experiments described in section 8.2.3.

Answer The strategy *Answer* is to give all kinds of answers which try to more or less touch on the content of the input and generate a vague response that hopefully might fit into the conversation. This group also contains answers that confess that the input belongs to a domain or topic which cannot be understood by the system. This strategy is taken from the Wizard-of-Oz experiments (section 8.2.3) and the user questionnaires (section 8.2.4).

Counter Question The strategy *Counter Question* is to ask a counter question to a known domain and topic instead of answering the unknown input. The strategy originated from the user questionnaires (section 8.2.4).

Echo Question The *Echo Question* strategy generates echo questions to the incoming unknown utterance and is observed in the user questionnaires (section 8.2.4).

Comprehension Question The strategy *Comprehension Question* realizes comprehension questions based on the incoming utterance in combination with another known topic and domain. These type of questions also originate from section 8.2.4.

Nod The strategy *Nod* is realized through all kinds of acknowledgements and signals of acceptance and refers to the "yes-man" strategy from section 8.2.2.

The main strategies are split into further subgroups by several features. Although the situation in which the strategies are applied is always caused by an input unknown to the machine and without successful interpretation, the input analysis of the embedding dialogue system can nevertheless extract linguistic information. The best subgroup of a strategy to use is decided according to the linguistic cues successfully extracted from the incoming utterance. The more linguistic information that is understood, the more specific the answer can be.

The features which further narrow down the strategies are:

- Topic
- Fragment
- Similar Topic
- Sentence Mood
- Wh-Pronoun
- Subject
- Verb
- Objects

The *topic* feature can hold the information of potentially understood topics from the incoming utterance. The *fragment* feature indicates whether the input analysis could assign a complete syntactic analysis to the input, whereas the *similar topic* feature is a feature that has as value a safe topic known to the system which is similar to the topic of

the incoming utterance. The safe topic selection and the related thread selection is described in detail in section 6.3.1. Possible values for the *sentence mood* feature are: Statements, yes/no questions, wh-questions and unknown. For the other linguistic features, possible values include lemma, surface, or an abstract boolean value indicating if the token exists or not.

The final set consists of 25 strategies. Figure 8.1 presents the full taxonomy of strategies. Strategies are printed in italics, group names are printed in bold. The abbreviation in some strategy names indicate necessary predicate argument features; for example, the strategy "Decl_VO" can be applied if the input analysis could successfully determine the verb and at least one object in the incoming utterance. Strategies are not exclusive: If the incoming utterance is analyzed to be a yes/no-question with a determined subject, possible strategies to apply are "YN_S" but also "General_S", as well as "General_Confess" and "General_Evade".

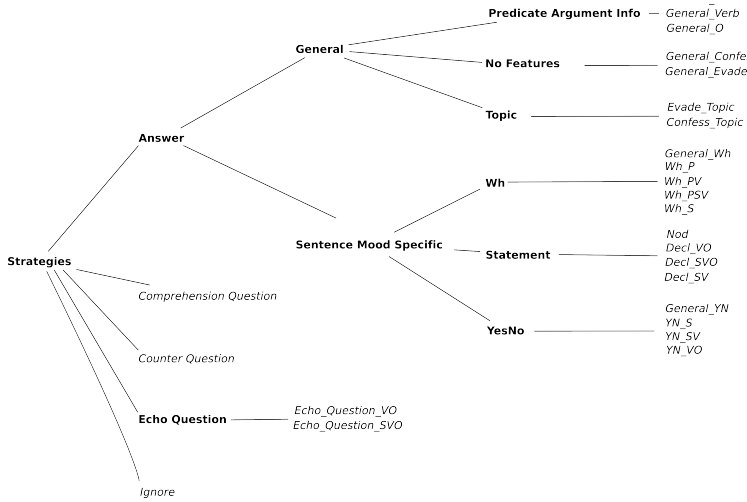

Figure 8.1: The Full Strategy Taxonomy

Table 8.2 and 8.3 present the final set of strategies that are not sentence mood specific including a description and examples. The strategy "Answer" is split into two main subgroups: "Evade" and "Confess". While the *Evade* group contains all attempts to answer the input without signaling a problem, the group *Confess* contains strategies to confess the understanding error. This is done with regard to a future evaluation,

Strategy	Description & Example
General Evade	A vague answer without any understanding. Examples: "I don't care", "That's boring"
General Confess	Confess the error Examples: "I am sorry, you are talking about topics I don't know."
Ignore	Ignore the unknown input
Evade Topic	A vague answer using the probable topic or a recognized entity Examples: "I don't care about *cats*."
Confess Topic	Confess the understanding error using the probable topic or a recognized entity Examples: "Sorry, the topic *cats* is out of my knowledge."
General S	Refer to the subject of the utterance Examples: "Subj=You: Let's stop talking about me."
General O	Refer to the object(s) of the utterance Examples: "Obj=boats: Most people like boats."
General V	Refer to the verb of the utterance Examples: "Verb=run: Running is sometimes a good idea."

Table 8.2: General Uncertain Answer Strategies I

since empirical work (Skantze, 2003) states that users are more likely to judge that a conversation with a machine has failed if the machine frequently confesses understanding errors.

Additionally, subgroups of the *EVADE* strategy can be distinguished according to sentence mood and further predicate argument information. Tables 8.4 to 8.6 show strategies that can be applied to input for which one of the three sentence mood values "statement", "wh question", and "yes/no question" is found. These strategies can be seen to form a hierarchy, starting with the group of the most underspecified strategy (such as "WH" or "YN"), and becoming more and more specific the more features are filled; for example, an incoming utterance which can be answered by a "YN_SV", can also be answered with the less specific strategy "YN_S".

Strategy	Description & Example
Comprehension Question	A fake repair Examples: "...exhausting..." - "Do you mean my job?"
Counter Question	Counter question Examples: "...cats..." - "Do you know that Lady Gaga loves cats?"
Echo Question VO	An echo question generated from the verb and the object(s) of the incoming utterance, a "you" inserted as a subject. Examples: "You like rafting?"
Echo Question SVO	An echo question generated from the verb, the turned subject and the object(s) of the incoming utterance. Examples: "Do I like rafting?"

Table 8.3: General Uncertain Answer Strategies II

8.4 The Computational Tool

The strategies presented are incorporated in a software tool that answers detected out-of-domain utterances and can be used as an extension to dialogue systems. The tool includes an out-of-domain classifier and an answer generator. While the out-of-domain classifier detects if an incoming utterance is out-of-domain or in-domain, the answer generator uses the described strategies to generate an appropriate system reaction to the incoming input. The out-of-domain classifier is a part of the natural-language understanding competence of a dialogue system and described in section 5.3. The following sections describe the components of the answer generator, and how it functions.

8.4.1 Uncertain Answer Thread

The answer generator uses the strategies described in 8.3 to answer utterances which were classified as out-of-domain. Similar to the embedding dialogue system, which provides dialogue threads encoding possible conversation sequences for the application domains, and the social-talk component, which provides social-talk conversation threads, the uncertain-answer module provides a dialogue thread for handling out-of-domain

Strategy	Description & Example
General_YN	Vague answer to unknown yes/no-question Examples: “Yes!”, “No idea.”
YN_S	Use the subject to generate an answer Examples: “No, I don’t.”
YN_SV	Use the subject and the verb Examples: “Did you know...” - “No, really?”
YN_VO	Use the verb and the object(s) Examples: “...like/likes cats?” - “I like dogs.”

Table 8.4: Uncertain Answer Strategies to Yes/No-Questions Based on Partial Linguistic Understanding

utterances. This thread contains a communication pattern to handle out-of-domain utterances using a simple baseline sequence that originates from the strategy observations described in the preceding sections.

As opposed to the dialogue sequences in the small talk or task-talk threads, which differ in length between two and six dialogue acts, the thread for uncertain answers is more restricted. This is due to the fact that allowing the user to talk about an unknown topic for as long as she wants is too dangerous for the system. The uncertain-answer thread contains one abstract answer sequence consisting of two conversation steps in which the strategies found can be carried out. While most strategies apply to step one of the sequence, some strategies combine step one and two. The sequence is shown in table 8.7.

The baseline shown in table 8.7 incorporates the observations from the Wizard-of-Oz experiments described in section 8.2.3, in which the wizards nearly always answered briefly and immediately came back to a known topic and domain in the next utterance.

8.4.2 Step One - Reaction

In the first step of the answer sequence, a reaction to the user utterance according to a selected strategy is processed. While the second system action (the switching utterance) is a controllable behavior, the first one

Strategy	Description & Example
General_WH	Vague answer to unknown wh-question Examples: "Don't pester me with questions!", "Better ask me things about the menu!"
WH_P	Use the wh-pronoun to generate an answer Examples: "Where..." - "Somewhere."
WH_PV	Use the wh-pronoun and the verb Examples: "What is..." - "I don't know what it is"
WH_PSV	Use the wh-pronoun, the subject and the verb Examples: "What is..." - "I don't know what it is"
WH_S	Use the wh-pronoun and the subject Examples: "How do you like...." - "I like *drinks*."

Table 8.5: Uncertain Answer Strategies to Wh-Questions Based on Partial Linguistic Understanding

is very challenging. A strategy from the set of presented strategies is selected and integrated into the baseline sequence. Table 8.8 shows possible sequences after filling in the main strategy classes sorted by sentence mood.

The strategy selected for answering the actual out-of-domain utterance depends on the linguistic information found. The selection process uses a decision graph (section 8.4.2). The more information that is found through analysis of the input, the more different strategies can be used to react to the out-of-domain utterance.

After selecting a strategy, a verbal reaction is created by the answer generator based on knowledge of the selected strategy (section 8.4.2). The answer generator contains strategy-specific knowledge for all given strategies. This knowledge covers possibilities to generate natural-language utterances for this strategy. Generation is based on templates and slot-filling. Templates can contain various slots that need to be filled,

Strategy	Description & Example
Nod	Blindly accept or acknowledge Examples: "Yes, you're right!", 'Interesting"
DECL_VO	Use the verb and the object(s) to generate an answer Examples: "Who likes Brussels sprouts? Nobody."
DECL_SVO	Use the subject, verb and object(s) Examples: "I don't like Brussels sprouts either."
DECL_SV	Use the subject and the verb Examples: "What is..." - "I don't know what it is"

Table 8.6: Uncertain Answer Strategies to Statements Based on Partial Linguistic Understanding

e.g., with information from the original user utterance, with information retrieved from the knowledge bases, or with inflected forms.

Strategy Selection

The decision on which strategy to apply to an incoming out-of-domain utterance is made by a decision graph. Firstly, the incoming utterance is tested against all features described in the section 8.3: Topic, similar topic, fragment, sentence mood, and predicate argument structure. The set of positively tested features is then used as input vector for the decision graph.

The decision graph is implemented in a compact version. In addition to the standard node types, namely "leaf nodes", for nodes holding results, and "condition nodes", which contain a condition to further process the graph, a new node type, "blank nodes", is integrated. Blank nodes can hold results and are used to provide nodes, which are neither leafs nor contain conditions, but can be traversed and contribute results for the sub-graph below them. Moreover, conditional nodes can

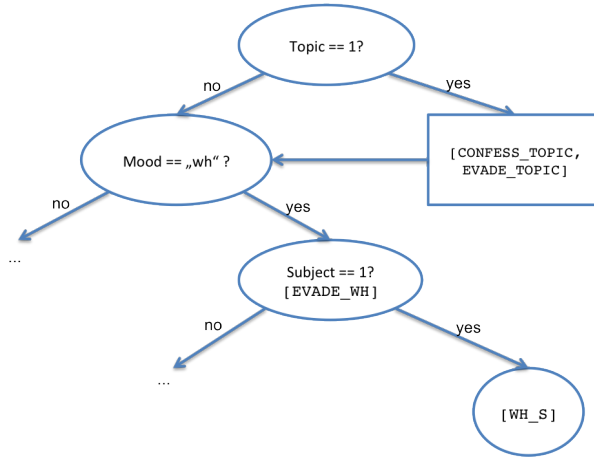

Figure 8.2: Decision Tree Example

also transport results through the graph. This enables the graph interpreter to accumulate possible results while traversing the graph. Due to the hierarchical structure of parts of the strategies, this behavior frees the author to insert all possible strategies of a strategy hierarchy onto leaves, and it avoids repetition. The final result of a tree decision is a set of possible strategies that can be applied to an incoming utterance.

Figure 8.2 shows part of the compact decision tree example. The first node is a simple condition node containing the condition “topic ==

User	Step 1 System	Step 2 System
An out-of-domain utterance	A reaction to the utterance	A switching utterance to move on to the next safe topic

Table 8.7: Baseline Sequence for Uncertain Answer Talk

User	Step 1 System	Step 2 System
Statement	Nod	A switching utterance
	Comprehension Question	
	Answer	
	Ignore	
	Echo Question	
	Counter Question	
Question	Accept/Answer	A switching utterance
	Reject/Answer	
	Inform/Answer	
	Ignore	
	Echo Question	
	Counter Question	
Unknown	Inform/Answer	A switching utterance
	Ignore	
	Echo Question	
	Counter Question	

Table 8.8: Final Sequence for Turn One and Two of the Uncertain-Answer Module

1?", which tests if a value for the topic feature is existent in the input vector. If the vector contains a value for topic, the next node to enter is a blank node (square box). The blank node carries two results: The strategies "Confess_Topic" and "Evade_Topic". A blank node has just one outgoing edge which does not carry any values. In the example, the blank node's outgoing edge leads to the same conditional node that is entered by the second outgoing edge of the first node. This means it does not matter if an input vector contains a topic value, the conditional node containing the condition "mood == wh?" is always traversed. The only difference is that the path coming from the blank node already carries two strategies that can be used to answer the input.

The next step filters the input according to the value of the sentence mood. If the input vector contains the value "wh" for the sentence mood

feature, the next node to enter is the conditional node "subject == 1?", which looks up the existence of a value for the subject feature in the input. However, independently of the subject decision, all leaf nodes in the sub-graph below this conditional node could end with the strategy "Evade_WH". Thus, the result is already provided by the conditional node which passes it down to the whole sub-graph. The last node is a leaf node containing one result, the strategy "WH_S".

If all tested features in this example graph are positive the incoming vector would look like the following:
[topic = 1, mood = wh, subject = 1]

From a retrieved set of strategies one is selected at random, respecting the following conditions:

- If the last uttered response originates from the same strategy as the currently selected strategy and there are no other ways to realize the selected strategy, select another one. This behavior avoids repetition.
- If the last selected strategy was "Ignore" and the currently selected strategy is "Ignore", select another one. This avoids too frequent selection of the strategy "Ignore".

Strategy Realization

After the system has selected a strategy to handle the incoming utterance, the strategy has to be realized regarding the actual content of the specific utterance. The backbone of the strategy realization is a template generation approach. However, several different pre-processing steps are included and results of these steps can be filled as slot filler into the template generation. The strategy realization component makes heavy use of WordNet and wikipedia[2] as sources of linguistic information and world knowledge, especially is-a hierarchies.

Each strategy is realized individually. Since there are many very different strategy realization possibilities encoded in the component, it is impossible to describe all of them. The following paragraphs therefore give some prototypical examples of strategy realizations.

Comprehension Question The strategy *Comprehension Question* is one of the most specific and complicated strategies to realize. A comprehension question is for example the following:

[2]www.wikipedia.org

(8.2) Input: I'd like to have rabbits!
Answer: Do you mean the pet?

The idea behind the strategy is to build a bridge to a known topic in a known domain which is similar enough to a topic from the input sentence that the question feels natural. A comprehension question can only be applied if at least one topic was found for the input and from the set of known topics in the system one was found to be similar enough to this topic. Only if these conditions are satisfied can the strategy be applied. This means that the strategy is not selected very frequently.

In the example above the topic recognized in the utterance is "rabbit". The example presumes that the underlying system has the topic "pets" in the set of its known safe topics. The goal of the system is to switch to one of these safe topics. For each of the system's safe topics, similarity with the found topic "rabbit" is calculated. The topic with the highest value above a defined threshold is selected as the most similar topic. In the event that no similarity between a safe topic and the known topic exceeds the threshold, the strategy can not be applied.

If a topic was found to be similar enough, a surface form for a comprehension question is generated from templates. Surface forms can, e.g., be "Do you mean [_]" as in the example above or "Are you talking about [_]", and other possibilities. The surface word expressing the selected topic is integrated into the template generation as a slot filler. According to the selected surface form of the comprehension question this step may include morphological manipulation. Manipulation includes, for example, the processing of the plural form or the insertion of an appropriate determiner.

Evade Topic *Evade_Topic* is a strategy that can be applied if at least one topic could be found for the incoming utterance. The system does not explicitly answer the unknown input but generates an answer that refers to the topic found. Examples for *Evade_Topic* include:

(8.3) Input: How many bones are in the human wrist? Answer: I don't care about humans. / Can we talk about bones later maybe?

Again, the actual surface is generated through templates integrating the topic. Topics are usually manipulated to occur in the plural form in the actual response with the exception of named entities (e.g., the dome of *Worms*).

General_SV *General_SV* is one of the strategies that do not use any results from the interpretation step, but instead use predicate argument information from the incoming utterance. In this case, the subject and the verb information. This strategy can be realized in many different ways. One example is to have more specific templates for subject/verb combinations that the system should definitely handle a special way, such as mentioning violence in the example below:

(8.4) Input: I killed a cat! Answer: I can't believe you're being so aggressive and I don't wanna talk about that any more.

These "special" templates are of course similar to a simple pattern-matching chatbot, but the difference is that we know exactly what the predicate argument structure is and do not need to handle various surface forms of similar inputs. Moreover, the system allows groups of related synonyms as values for verbs, so that "I hit a cat" may be answered with the same response.

Another completely different way to realize this strategy is, for example, to look up an antonym of the verb in WordNet and generate a response using the original verb and the antonym.

(8.5) Input: I won the last game!! Answer: Better you win than you lose!

In combination with the subject, the verb may need inflection by morphological manipulation. This is a very dynamic way to realize the strategy, and it can be applied without any special knowledge about the extracted values for subject and verb.

8.4.3 Step Two - Bridging

It is very important for the system to keep the turn after reacting to an out-of-domain utterance and to change the topic and thread of the conversation to "safe ground". Keeping the turn forbids giving the turn back to the user after a response to the incoming utterance. A new input from the user without switching to a known safe topic and conversation thread will most probably result in a new understanding error. The goal is to lead the user to another topic directly after answering the out-of-domain utterances or even include the bridge in the answer. Therefore, the second part of the baseline answer sequence in the uncertain-answer thread (table 8.7) contains the switch to a safe dialogue thread and topic.

A second necessary group of strategies encloses bridging strategies from the unknown domain to a known topic and thread, so that the agent will be able to engage in the conversation again. The strategies for constructing a bridge are verified by the Wizard-of-Oz experiments as well as the user questionnaires (section 8.2.4 and 8.2.3) and can be subsumed as follows:

No Bridge No bridge is used to change from one topic and conversation thread to another.

Switching Marker Verbal markers such as "So" and "Now" are used to signal the topic and thread switch to the user

Explicit Switching Utterance An explicit utterance is used to signal the topic and thread switch to the user, such as "Let's come back to..."

Implicit Switching Utterance An utterance containing an implicit switch is used, e.g., for utterances that are a response and a bridge at the same time.

In some strategies used to generate an answer in step one, the two steps of the dialogue thread are not very clearly separated: Answer and bridge can occur in one utterance or as separate units in several utterances. Some strategies include at least parts of the bridging utterance in the first system turn. These are the questions "Counter Question", "Comprehension Question", and "Echo Question". The idea is that the question should not be answered by the user but directly by the system itself as the following example of a comprehension question taken from the user questionnaires shows:

(8.6) **Customer:** ... [Non-understood utterance with the topic "work"]
Participant: Do you mean my job? It's sometimes exhausting but I like it very much!

To enable a switch that feels as smooth as possible, the *uncertain-answer tool* calculates the best possible, known dialogue thread and topic to switch to for every incoming utterance. Basically, the selection of a new thread is part of the overall multi-threading dialogue manager described in chapter 6. In general, if no thread initialization by the user was detected, the selection algorithm favors already active threads over fresh threads. If no suitable active threads exists, importance values determine the selection of a new thread.

Topic	Threads
cocktail:Cocktail	Task:NegotiateObject
wordnet:shirt	SmallTalk:Compliment
wordnet:occupation	SmallTalk:InterestedQuestion SmallTalk:ProvideInfo

Table 8.9: Assignment of Possible Safe Topics to Threads

However, the *uncertain-answer module* has a method to select an appropriate thread as the target of a switch. The overall dialogue manager is used if this method cannot provide a thread/topic pair to switch to only.

The main information needed to decide which safe thread to initiate is the topic. The tool needs to know the set of dialogue threads which are safe for the system. Additionally, the system manages a list of possible topics and the relationship of topics and threads. There are several topics which can be used by different threads and some which can only be discussed in one special thread. In the barkeeper application the dialogue system, for example, is able to lead a discussion thread with the topic "cocktail", but not to engage in a compliment thread with the same topic. Topics and threads are safe for the system if the system possesses enough linguistic content to handle them in a meaningful way. Table 8.9 shows some examples of topics and threads.

The selection of an appropriate thread-topic combination to switch to starts with the comparison of the topics found for the incoming utterance with the topics being safe for the system. Topic comparison is done by calculating semantic similarity in WordNet. The calculation of the relatedness is implemented through semantic similarity in WordNet with the "Resnik" algorithm (Resnik, 1995). The software package used is the Semantic::Similarity package by Ted Pedersen (Pedersen, Patwardhan, & Michelizzi, 2004). A threshold is used to define a minimal value of similarity that was estimated through some informal tests. All of the system's safe topics are compared with all topics found for the incoming utterance. If one comparison achieves a higher value of similarity than the defined threshold, this topic is added to the list of possible topics to switch to. Because the topic selected in this way has to be semantically

related to the last topic, it is guaranteed that the topic change feels natural to the user, in a similar way to how a human would make free associations.

After the comparison, the topic with the highest similarity value is selected and the system looks up a possible dialogue thread for the selected topic to initialize. If several threads are possible for one topic, the system considers the memory of the uncertain-answer component. If the topic and the selected thread in this combination have been active before, another combination is tried. If no "fresh" combination can be found, the system drops the selected topic and tries another topics from the list, with lower similarity values. If no lower similarity values are found in the comparison that still extend over the threshold, the system initializes one of the default thread-topic combinations for the application.

For strategies including bridging behavior already in the answer (step one) the successful selection of an appropriate new safe topic and thread is a necessary precondition. These strategies are applied if a successful bridge from an incoming topic to a safe topic can be generated only.

8.5 Conclusion

This chapter describes a novel set of strategies which can be used for handling out-of-domain utterances in dialogue systems. We have seen that a system which engages in social talk will be confronted with a huge amount of understanding errors caused by out-of-domain utterances. Since the target systems are conversational and cooperative systems, e.g., conversational agents, the traditional strategies to handle non-understandings such as "confessing the understanding" or "trying to repair" are unsatisfactory. Empirical work has shown that if machines frequently confess to understanding errors, it leads to a higher perception of failure for the whole dialogue by the human user (Skantze, 2003). Repairing an understanding error caused by an out-of-domain utterance is condemned to fail, since the utterance and the needed knowledge to understand the utterance are just outside of the scope of the machine. Although this is quite an obvious problem, research has long neglected the out-of-domain origins of understanding problems. The only example known to the author is (Patel et al., 2006). Their approach can be seen to enable more understanding through adding some out-of-domain classes mostly specific to the application domain. In the last few years, some work, especially with empirical focus, has suggested new strategies

to handle non-understandings ((Henderson et al., 2012), (Bohus & Rudnicky, 2005), (Skantze, 2003)), However, they still miss the importance of detecting the origin of the non-understanding and suggest strategies which are not specific for out-of-domain utterances.

In this chapter I therefore introduce a way to handle out-of-domain utterances based on a completely new set of 25 conversational strategies. The strategies are retrieved by an investigation of human communication from several sources (section 8.2) in which human behavior focuses on hiding non-understandings instead of trying a repair. The strategies are presented in a unified taxonomy organized by means of linguistic information (section 8.3). The main strategies are *Ignore*, *Answer*, *Counter Question*, *Echo Question*, *Comprehension Question*, and *Nod*. These are further specified according to linguistic information such as recognized predicate argument structure and sentence mood.

The strategies are incorporated into a computational tool which constitutes the *uncertain-answer module* of the presented SOX toolkit. The tool offers a conversation thread for answering unknown out-of-domain utterances consisting of an abstract dialogue sequence made of two steps: An answer, and a bridge to a new topic and thread known to the system. The strategies are used to fill the abstract sequence at runtime with content. In general, the strategies are applied to step one of the sequence, the answer. For initializing step two - the bridge - a small set of strategies is used. A description of bridging behavior and thread selection is given in section 8.4.3. However, some of the strategies do not clearly separate between the two steps. For those, the bridge is not calculated autonomously.

In the tool a compact decision tree is used for strategy selection (section 8.4.2) based on linguistic information. A template and slot-filling approach is used for strategy realization (see section 8.4.2).

9 Evaluation

This chapter presents two evaluations. The first evaluation focuses on the usability of the barkeeper game application in which the SOX toolkit components are embedded. The second evaluation is a test of the particular effects of the main SOX components proposed in this work, social talk and uncertain-answer module, on measures such as naturalness and fun while using the system.

For the first evaluation, a usability test based on a field test in a virtual game application and questionnaires was used. The results of the first evaluation are very satisfying and promising: The results show that the barkeeper application is useful and well-accepted by the users. Users regard the barkeeper as a virtual person, even as an open-minded personality. All participants enjoyed using the application and affirm that they would like to use the application again. Even if the system does not always understand the users well and said unexpected things, it could still provide appropriate responses to help users solve their problems and to entertain them.

The second evaluation provides interesting and useful feedback regarding the effect of the proposed toolkit components on the overall satisfaction of the users. The participants could choose one of four different videos, all of them showing a conversation between a human user and the barkeeper. Each conversation was conducted with a different setting of the dialogue system, such as dialogue system with small talk and uncertain answers activated or with small talk activated but without the uncertain-answer module. After watching a video, the participants judged the conversation they had seen using a questionnaire. In general, the results confirm the assumption that both components produce nicer and more natural conversations.

Following is how the remainder of the chapter is organized. In the two main sections, firstly the usability evaluation of the whole application, and afterwards the evaluation of the toolkit components, is presented.

The sections include the descriptions of the evaluation designs, the number of participants, the questionnaires used, and the evaluation results, as well as a discussion of the evaluation results. In the conclusions, the results are briefly summarized.

9.1 Usability Evaluation of the *KomParse* System

This section describes the evaluation of the *KomParse* application (see chapter 3). *KomParse* offers technology for two conversational agents that are non-player characters (NPCs) in a virtual online game. It is important to know whether the conversational agents are accepted by the end users of such a game. Since the focus lies in user satisfaction, the evaluation belongs to the group of subjective and user-oriented evaluations (Dybkjær, Hemsen, & Minker, 2007).

The subjective measurement of the acceptance of an application or technology belongs to the group of usability evaluation. Usability evaluation focuses on users and the users' needs. Usability evaluation wants to know if a system can be used for the specific purpose from the user point of view, and if it allows the users to achieve their goals in the manner they expected. The most important criterion for measuring usability is the user satisfaction. Mainly interviews or questionnaires are used to obtain information about user satisfaction. The aspects that are important for a usability evaluation depend on the type of application tested. Test parameters usually cover such aspects as the efficiency in reaching a goal, and the effectiveness of single system characteristics. This is the case in the ISO standard 9241/10, for example. This standard is intended to calculate the usability summing-up values for effectiveness (percentage chance of achieving a goal), efficiency, and user satisfaction. Efficiency parameters include time required to reach a goal, the error rate and the amount of effort needed to achieve a goal.

Another commonly used usability test is SUMI (Software Usability Measurement Inventory)[1], the industry de-facto usability evaluation standard for analyzing users' opinions towards software products. SUMI covers most of the principles described in the ISO standard but focuses on the dimensions efficiency, affect, helpfulness, control and learnability.

A popular framework for evaluating dialogue system's usability is *PARADISE* (Walker, Litman, Kamm, & Abella, 1997). PARADISE measures system usability on the basis of task success and task effort.

[1] http://sumi.ucc.de

Dimension	Sample Item
Usability	
Dialogue Control	I felt like the conversation with the barkeeper was under my control.
	I always knew what to say next.
Reliability	The barkeeper sometimes did something unexpected.
Aesthetics	The barkeeper has a virtually appealing presentation.
Cognitive Demand	There were times while talking with the barkeeper when I felt quite tense.
	I had to look for assistance when I talked to the barkeeper.
Acceptability	
Satisfaction	The barkeeper is a nice person.
	I would like to visit the bar again.
	I liked the barkeeper's behavior.
Naturalness	The barkeeper behaved naturally (like a real barkeeper).
Entertainment	Conversation with the barkeeper was fun!
	Talking to the barkeeper was boring.
Usefulness	The barkeeper makes virtual worlds like Twinity more interesting.
Improvements/ Remarks	What has to be changed?/What did you like?

Table 9.1: The Post-Test Questionnaire

The main idea is that minimizing effort combined with the parallel maximizing of task success improves user satisfaction. Task effort is measured using efficiency parameters such as time needed to fulfill a task.

Another framework developed for evaluation of language-based interaction is *Quality of Experience* (QoE) ((Möller, 2002), (Möller, 2005)). The aim of the QoE evaluation is measurement of the grade of system quality perceived by the users. Quality is defined as the compromise between the users' expectation and the experience while using the system, and is measured using parameters such as sound quality, system comprehensibility, and the perception of the natural-language understanding capabilities of a system, which are important for speech-based interaction.

For entertaining applications such as games, the usual evaluation measures that target task-based systems are not necessarily useful. As Gustafson, Bell, Boye, Lindström, and Wirén (2004) point out, computer

Dimension	Sample Item
Understanding	I had the feeling the barkeeper understood me well.
Response Generation	What the barkeeper said made sense to me.
Dialogue Moves	The barkeepers reactions are appropriate. NPC has done unexpected things.
Large-Scale Knowledge	The barkeeper seemed informed about the world.

Table 9.2: The Post-Test Questionnaire for Components

games are usually evaluated by professional game reviewers, since user satisfaction may increase rather than decrease with task completion time, for example. Unfortunately, professional reviewing was not available for the described scenario. Therefore, the post-test questionnaire contains additional questions about naturalness, the agent's personality, and fun while using the system to indicate the positive or negative perception of the entertaining aspects of the barkeeper.

9.1.1 Evaluation Design

The method used to evaluate the usability of the KomParse system is built on top of a successful field test used to evaluate the *compass2008* system (Uszkoreit et al., 2007). The evaluation combines parts of the SUMI questionnaire and the ISO NORM 9241/10 and was designed by experts for software quality and usability at the Deutsche Telekom. A *field test* normally takes more than ten users to test a software application in a real environment. A virtual field test is a reasonable test for the *KomParse* system, which works in a virtual environment. In the field test, the usability problems are collected by subjective reports of test users (e.g., online questionnaires). Log data can be used to assess usage duration and quality.

The virtual field tests take place in front of personal computers. The subjects have to fulfill a list of tasks with the *KomParse* system by conducting dialogues with the NPC barkeeper. The tasks included, but were not restricted to, ordering cocktails and asking for the biographic

information of a famous personality. We also added "small talk with the barkeeper" to the list of the tasks, to ensure that the users took the opportunity to chat with the NPC. For each task, the subjects had to fill out a post-task questionnaire. The Likert scales used contain values for indicating agreement, namely "totally agree", "agree", "neutral", "disagree" and "totally disagree".

Table 9.3 shows the post-task questionnaire items and results for the tasks.

Additionally, the evaluation questionnaire covers questions regarding the performance of the overall system as well as the system's single communication components. Table 9.2 shows the features that are included in the components questionnaire.

The system evaluation post-test questionnaire includes the items in table 9.1.

The dimensions in the questionnaires are selected carefully from SUMI and ISO NORM 9241/10.

9.1.2 Evaluation Results

Twelve people participated in the test, mainly students with different knowledge and experience of virtual games. The evaluation results are very useful and provide valuable insights into the acceptability of the system. Table 9.3 shows the results of the post-task questionnaires. The average values from the Likert scale are calculated by translating the agreement values to numerical values: "totally agree" (+2), "agree" (+1), "neutral" (0), "disagree" (-1), and "totally disagree" (-2).

The users report positive results for all given tasks in the post-task questionnaire. Even for the small-talk task, which was to guarantee that the users will make use of the small-talk opportunity, feedback is positive. Most users could complete all tasks. Only two users reported that they could not fulfill a task. That means that *task completion* is very high. However, a comparatively large number of users reported problems that occurred during a task (10, versus 24 without problems). Some of the problems are mentioned in the general positive/negative remarks in table 9.5.

On average, the users stated that they did not feel that it takes too much time to complete the tasks. Especially task one "Ordering a drink" gets a good result with -0.917 on average. Therefore, the KomParse system's *dialogue efficiency* is satisfactory. The *dialogue control* in the task seems to be a little better in the two other tasks, "information about

Question	Average Results Task 1	Average Results Task 2	Average Results Task 3
Could you complete your task?	yes:8, no:1, problems:3	yes:10, no:1, problems:1	yes:6, no:0, problems:6
It took me much time to complete the task	-0.917	-0.750	-0.750
I always knew what to say next.	+0.333	+0.667	+0.667
The agent has always done what I was expecting.	-0.417	0	0

Table 9.3: The Results of the Post-Task Questionnaire

a celebrity" and "small talk". Both get an average value of +0.667 for the statement "I always knew what to say next". The agent's *reliability* (NPC did what I expected) is rated rather neutral on average. The drink task gets an average value of −0.417. That indicates that the agent's ability to act appropriately to the user's expectations must still improve.

The evaluation includes another post-test questionnaire covering questions regarding the usability of the overall system, as well as the system's single communication components. Table 9.4 shows the overview of the results from this post-test questionnaire.

The users gave positive feedback here, too. Results show users are very satisfied with the system: Eight out of twelve users would visit the bar again, the remaining four have a neutral opinion. No subject disliked the idea of visiting the bar again. The average value is +1.083. *Interest in using the system* is encouraging.

Likewise, most people liked the barkeeper's behavior (+0.667) and agreed with the opinion that the barkeeper is a nice person (+0.833). This indicates that *satisfaction and attitude towards the NPC* among the users is very high. Moreover, some users agreed that the barkeeper behaves *naturally*, like a real barkeeper. The average value is +0.250.

Question	Average Results
The barkeeper makes virtual worlds like Twinity more interesting.	+1.250
Conversation with the barkeeper was fun!	+0.833
There have been times during talking with the barkeeper when I have felt quite tense.	-0.500
I liked the barkeeper's behavior.	+0.667
The barkeeper has done something unexpected at some time.	+1.167
I felt the conversation with the barkeeper being under my control.	-0.333
The barkeeper has a visually appealing presentation.	+0.917
I always knew what to say next.	+0.667
I had the feeling the barkeeper understood me well.	-0.167
The barkeeper's reactions are appropriate.	+0.500
What the barkeeper said made sense to me.	+0.250
I had to look for assistance when I talked to the barkeeper.	-0.750
The barkeeper seemed informed about the world.	+0.833
The barkeeper behaved naturally (like a real barkeeper).	+0.250
Talking to the barkeeper was boring.	-0.750
The barkeeper is a nice person.	+0.833
I would visit the bar again.	+1.083

Table 9.4: The Results of the Post-Test Questionnaire

The barkeeper also did a good job of *entertaining* the users: Nine out of twelve users agreed with the opinion that the conversation with the barkeeper was fun - the average scale value is +0.833 - and denied the statement that talking to the barkeeper was boring (−0.750).

Understandably, ten out of twelve subjects thought the barkeeper would make virtual worlds more interesting, which acknowledges the *usefulness* of the system.

Most users liked the *appearance* of the NPC. The average scale value is +0.917. The *aesthetics and naturalness* of the NPC is hence confirmed

Positive User Feedback	**Negative User Feedback**
the smilies	sometimes the answers to questions don't appear
you an answer the questions with numbers	sometime the reaction will need to long
some funny answers	it seems that the NPC ignores the input
the way he offers an answer is very natural	more phrases for interaction will be better, e.g., "I want ...", "Please make me a ..."
an open-minded person, Hank Slender	Hank does not know the ingredients of a cocktail
works well in general	I had the feeling, if I am not fast, he will change the topic.
Hank tried to change the topic himself. It makes the conversation natural	a little less quiz would be better

Table 9.5: Answers Given to the Open-Ended Question in the Post-Test Questionnaire

by the users. However, the *dialogue control* seems to be problematic. The average value for the statement "I felt the conversation with the barkeeper being under my control" is slightly negative with -0.333. On the other hand deciding, what to say next is uncomplicated ($+0.667$). Regarding the general remarks (see table 9.5), it seems that the system initiative sometimes is too fast for the user. We may have to adjust the time and selection values for the system initiative. This is also reflected in the comparatively high value for the *reliability* of the barkeeper: The average value is $+1.167$. However, this does not become noticeable in the *cognitive demand*. Very few users (two out of 12) felt stressed while talking with the barkeeper (-0.500) and there was no huge need for assistance (-0.750).

The subjective component evaluation provides very useful feedback. Only three out of 12 users agreed with the statement that the NPC understood them well. The scale is more negative with a value of -0.167. This means that the *natural-language understanding* component is still the bottleneck in the system. Although dialogue-act recognition works well in comparison to related work (Klüwer, Uszkoreit, & Xu, 2010), the system still needs more paraphrases and rules for generating valid

syntactic relations and more training data for the classification of rare utterances to dialogue acts.

On the generation side, about 58% of users accepted the NPS's *response* (the scale value is +0.250) and 83% of users found the NPC's reaction to be appropriate.

The NPC's perceived level of knowledge gets a positive value of +0.833. We take that as an indicator that the *knowledge* bases create a sufficient basic knowledge for the agent.

The post-test questionnaire includes one open-ended question asking for general positive and negative feedback about the system. Table 9.5 shows some answers the users gave. There is both positive and negative feedback that will be very important for further development and research. The positive comments show that the avatar's interface design is widely accepted by the users and that they enjoy talking to the barkeeper. The negative remarks tell us that an even more fine-grained ontology of cocktails would be helpful for our scenario and that we have to adjust the system initiative to make the barkeeper's behavior more reliable.

In general, the evaluation shows that our NPC, the barkeeper Hank Slender, is useful and well-accepted by the users.

Regarding the evaluation of social abilities, the evaluation results also include mainly positive feedback from the users. Even if the task "small talk" was given to the subjects to guarantee that they use the opportunity to make small talk and not just to use it to evaluate tasks, the users reported positive results here. In the post-task questionnaire, for example, users on average stated that they always knew what to say next (+0.667), which suggests a good orientation and not too much effort to engage in small talk with the agent. The controllability (NPC did what I expected) of the agent was rated neutral in average. That indicates that the agent's ability to act how the users expected was neither particularly good nor particularly bad.

It is expected that the described measures and the positive feedback include the user's perception of the integrated social talk. However, it is complicated to tell to what extent competence in social talk affected the evaluation. To get a better impression of the effect of integrated social talk on the overall perception of the system, the next section describes another evaluation focusing on the SOX components.

9.2 SOX Component Evaluation

The overall usability evaluation of the KomParse system clearly shows that the users like the barkeeper's ability to entertain and be sociable. The agent can successfully create a feeling of naturalness and make users enjoy the conversation. The fact that the barkeeper is not strictly task-bound seems to enhance the fun of using the system.

However, rating the SOX components is not straightforward from this evaluation, because the users evaluated the conversational system as a whole.

Therefore, in this section a second evaluation is presented that takes a closer look at the particular benefit of the two components described in the preceding chapters: The small-talk module and the uncertain-answer module. The main assumption is that both components have a significant effect on the perception of the naturalness, likeability and intelligence of the agent, and thus also the fun using the system and the usability of the application.

9.2.1 Evaluation Design

The evaluation described in this section is done in a two-step approach. Firstly, several conversations between the *KomParse* barkeeper (see chapter 3) and several human participants were video recorded. The participants talked to different set-ups of the dialogue system, with the components for small talk and uncertain answers either activated or deactivated. Table 9.6 shows the different set-ups and distributions of modules. If the hypothesis is true that the small-talk module and the uncertain-answer module strongly contribute to the entertaining character and naturalness of the agent, the conversations with activated modules should get better evaluation results than the conversations without activated SOX modules.

Conversation	Small Talk	Uncertain Answers
Conversation 1	-	+
Conversation 2	+	+
Conversation 3	-	-
Conversation 4	+	-

Table 9.6: The Four Different Setups of the Evaluation System

Afterwards, the video-recorded conversations were judged by additional participants. External jurors, people who were not the users who led the conversations with the barkeeper, were asked to watch one recorded video conversation and then to fill out a questionnaire containing judgments regarding the conversation they have seen. The questionnaire contains, amongst others, questions to rate the perceived quality of the dialogue flow and the naturalness of the conversation. The full list of questions is given in table 9.7.

Of course, not only the phenomena that should be rated in the evaluation play a role in the user's judgment process. In conversations, the pure extraction of specific features is just not possible. Many different phenomena become entangled with the features targeted by the evaluation. People might judge the actual topics of the particular conversation, which are boring or coincidentally of special interest to them. They might even rate the physical appearance of the embodied conversational agent or consider missing knowledge in some of its answers. They might not like the system's "funny" answers, or the failure of other system components, such as incorrect anaphora resolution - problems that are not related to the phenomena that should be evaluated at all. The bottleneck of natural-language understanding, for example, can make evaluators rate the system completely negatively although it acts very intelligently if it understands what the user has said. This means that for evaluating the SOX extensions neatly, the whole baseline dialogue system needs to be perfect and without any failures. And even then it is not guaranteed that personal preferences (e.g., for the graphical interface or the conversation domain) will not affect the evaluation. Moreover, all conversation should be similar in content and conversation flow to be comparable.

On the other hand, development of just one conversation script which can then be used to generate four artificial conversations through applying the four different machine set-ups would not help either, because these artificial conversations would not show how the users really try to interact with the system. It is also nearly impossible to create one conversation script that shows the system behavior without being unfair towards the conversations with deactivated SOX extensions. It is obvious beforehand that the small-talk threads cannot be handled by the system set-ups without the extensions. That means parts of the conversation in the script would be condemned to fail for the baseline system right from the start.

Because a conversation script does not come into consideration, the only workaround lies in user-based conversation generation with good

instructions, and the well-considered wording of the questionnaire statements.

The instructions included two tasks: Ordering a drink and finding out information about at least one celebrity. Small talk was not included as a task. Some general instructions to use the dialogue system were given (such as special capitalization rules) to eliminate as many potential problems in the conversation as possible.

Unfortunately, the video-recording is very costly and error-prone. In the end, from the video-recorded conversations only four, one for every set-up, could be used. The others show unfinished conversations due to system crashes or other technical problems, such as a broken communication pipeline to the knowledge bases.

It is well-known that the way in which dialogue systems handle understanding errors influences the perception of task success by users ((Bohus & Rudnicky, 2008), (Skantze, 2003)). According to this observation, the perception of task success is better if an understanding error is not too often explicitly confessed, especially if no rephrase is possible that could be understood. Therefore, the baseline system randomly chooses one of the three strategies "ignoring", "moving to a new topic", and "confessing understanding error" to handle incomprehensible utterances. Thus, the conversations that do not have uncertain answers activated do not solely rely on the not-so-well accepted response of "confessing understanding error" to the user.

9.2.2 Evaluation Results

In the first step of the evaluation, various conversations were video-recorded of which, as mentioned before, four were selected for the second evaluation step. The four final videos differ between four minutes and 30 seconds, and five minutes and 30 seconds. There are 190 turns in total (76 user, 114 system).

In the second step of the evaluation, 32 people watched and evaluated the videos, eight for every video. The age of the participants varied between 21 and 57 years. 18 women, 13 men and one unspecified gender participated. Most of the participants were native German speaker (29), although one participant was native Spanish speaker, another Spanish and Catalan, one Dutch and one Portuguese.

17 participants had no experience with chatbots, 13 reported a little experience with chatbots and two referred to themselves as experienced with chatbots. Ten participants reported having no experience

with computer games and 12 to having a little experience with games. Ten participants were experienced with computer games. By and large, one could say that most participants were neither from the same field of expertise nor very experienced with related technologies.

Table 9.7 shows the mean values of all results of the component evaluation. The best result for every question is printed in bold. Several cases in which more than one conversation gets the best result occur.

The expectation is that the conversations with one of the SOX modules activated (conversation one and conversation four) get better user rates than conversation three with just the baseline dialogue system. If the SOX modules make any essential difference, conversation two, with small talk as well as uncertain answers activated, should get the best results. Conversation three should get the worst results.

For the most parts, the user feedback confirms these expectations. Conversations one and four are rated much better than conversation three. In the total number of most positive results (see figure 9.1), conversations one and four head the table with an equal number (both ten of the most positive results), whereas conversation three only gets four of the most positive values. On the other hand, conversation three has a significantly higher number of the most negative results. From the 23 negative results for the 20 questions, nine are allotted to conversation three, whereas conversations one and four get very few of the most negative results: Conversation one has one negative result, conversation four has four.

Figure 9.1: The Distribution of Most Negative and Most Positive Results

A surprise is the evaluation of conversation two. Against the expectations, the conversation gets many negative results (nine negative results) and only four of the most positive results. Although some of the negative results nevertheless create a neat picture in comparison to the other

Question	conv. 1 -small talk +uncertain	conv. 2 +small talk +uncertain	conv. 3 -small talk -uncertain	conv. 4 +small talk -uncertain
The conversation flowed smoothly.	**-0.25**	-0.75	-2.0	-1.25
The conversation could be the same between two humans.	-1.25	**-0.75**	-2.5	-2.5
I had the feeling the conversation often got stuck.	**1.5**	**1.5**	**1.5**	1.75
I liked the barkeeper's behavior.	**1.5**	-0.25	0.75	1.25
The conversation was certainly fun for the user.	0.5	-1.25	-1.0	**2.0**
The conversation seems interesting.	**-0.5**	-1.5	-1.75	**-0.5**
I would like to talk to the barkeeper myself.	0.75	0.25	-1.0	**1.0**
What the barkeeper said seemed odd sometimes.	**0.75**	1.75	1.25	1.75
I have the feeling the barkeeper understood the user well.	**-0.25**	**-0.25**	-1.5	-1.5
The barkeeper's reactions are appropriate.	**0.75**	-0.25	-0.25	**0.75**
What the barkeeper said creates a nice atmosphere.	0.5	-0.75	0.5	**1.75**
The conversation feels like a real-world conversation.	**-1.0**	-1.75	**-1.0**	-1.25
The barkeeper seems informed about the world.	0.0	**0.5**	**0.5**	**0.5**
The barkeeper behaved naturally (like a real barkeeper).	**0.75**	-1.5	-0.75	-0.25
I would have been bored in place of the user.	**-0.75**	0.75	1.0	-0.5
The barkeeper is a nice person.	0.75	1.25	1.75	**2.0**
The answers of the barkeeper are natural.	0.0	-1.75	-1.0	**0.25**
The barkeeper is a dominant person.	**-1.25**	-0.5	-1.0	-0.75
Given that the barkeeper is a machine, he leads a quite intelligent conversation.	0.75	1.5	0.25	**2.25**
The barkeeper handles topics and dialogue threads very flexible.	0.0	-0.5	-1.25	**1.0**

Table 9.7: The Results of the Post-Test Questionnaire

conversations, some of them are just against the expectations. There can be various reasons for this, of which two are most likely. It is conceivable that the combination of both SOX modules is somehow negative for the conversation. Alternatively, the particular conversation that was used for the evaluation (conversation two) could, due to the number of other factors that can affect the evaluation, contain phenomena that are not acceptable for the participants. At the end of this section I will come back to the two explanations. Firstly, in the following paragraphs we will have a closer look at the results.

To get a better understanding of the significance and variance of the results, *Anova* tests have been done between several groups of conversations. These are:

Small Talk One test comparing all values of conversations including the small talk model (conversation two and four; 16 items) against all values of conversations without the small-talk module (conversation one and three; 16 items).

Uncertain Answer One test comparing all 16 values of the conversations one and two (including the uncertain-answer module) against all 16 values of the conversations three and four (excluding the uncertain-answer module)

Baseline One test comparing the baseline system (conversation three; eight items) against all values of the other conversations (conversation one, two and four; 24 items)

Table 9.8 shows the results of the Anova tests with a significance threshold of 10%.

Utterance Understanding and Generation Some evaluation results show very clear differences between the conversational set-ups, such as in the evaluation of the dialogue system component for *natural-language understanding*. The NLU capabilities of the system are rated *significantly better* with the activated uncertain-answer module. Both conversations excluding uncertain answers get an average value of -1.5 for the statement "I have the feeling the barkeeper understood the user well", whereas the conversation with uncertain answers activated both get -0.25 (see figure 9.2). The p-value of the Anova test is 0.0246.

	Statement	P-Value
Uncertain Answer	I have the feeling the barkeeper understood the user well.	0.0246
	The conversation flowed smoothly.	0.0343
	The conversation could be the same between two humans.	0.0330
	What the barkeeper said creates a nice atmosphere.	0.0425
Small Talk	Given that the barkeeper is a machine he leads a quite intelligent behavior.	0.0451
Baseline	I would like to talk to the barkeeper myself.	0.0998
	The barkeeper handles topics and dialogue threads very flexibly.	0.0256
	The conversation flowed smoothly.	0.0423

Table 9.8: The Results of the Anova Tests Within 10%

Although all of these values are generally negative, the values are much better with uncertain answers than without. These findings support the assumption that the strategies in the uncertain-answer module can successfully create an impression of understanding.

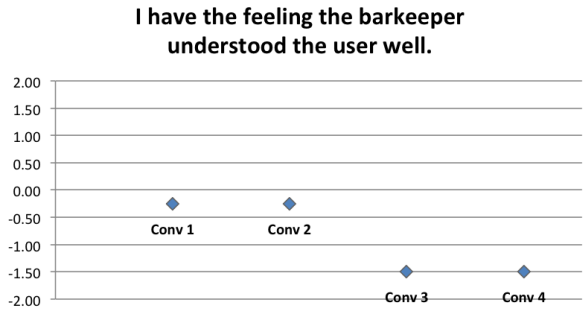

Figure 9.2: The mean values for NLU

Both conversations with small talk activated get very high results for the statement "What the barkeeper said seemed odd sometimes" (the average value is 1.75). That result is remarkably higher than for the conversations without small talk (0.75 and 1.25 respectively). This indicates that the small-talk module frequently generates strange answers. However, the statement "The barkeeper's reactions are appropriate" gets good results for conversation one (0.75) and four (0.75). This combination for conversation four could mean that the generated small-talk utterances are sometimes odd but still appropriate. Odd does not necessarily mean negative. On the other hand, conversation two, which could have supported this conclusion, again gets a negative result with -0.25 (see discussion below). Conversation three gets the same value (-0.25).

Intelligence Small-talk capabilities seem to have a huge effect on the perception of the *machine's intelligence*. The statement "Given that the barkeeper is a machine, he leads a quite intelligent conversation" is much more highly supported for the conversations including small talk than those without. The two average values are 1.5 (+small talk +uncertain answer) and even 2.25 (+small talk -uncertain answer), which are really good results. Both conversations without small talk get only 0.75 and 0.25. Conversation three gets the worst result.

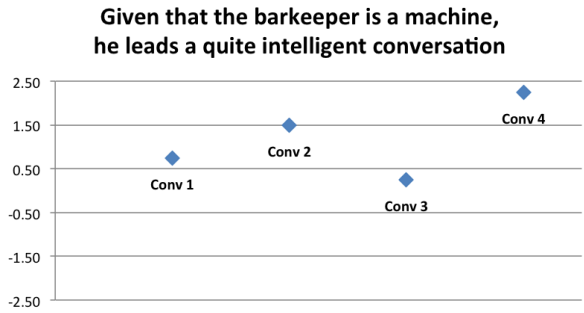

Figure 9.3: The Mean Values for Intelligence

The second measure for intelligence, "The barkeeper seems informed about the world" shows no clear results. All conversations except conversation one were rated 0.5, while conversation one received a neutral average value.

Likabilty & Attitude Towards the Conversation More people judged the barkeeper to be a *dominant person*, when small talk was activated. That makes sense because the system changes dialogue threads more frequently and comes up with its own conversation threads more often if started in small-talk mode. However, the average values are all positive. Most people did not find that the barkeeper was dominant, leading to average values of -0.5 and -0.75 for the small-talk conversations and -1.25 and -0.75 for the conversations without small talk. These results might indicate that it is still safe to activate small talk, since it will probably not affect user satisfaction.

The other statements regarding the agent's *likability* and the conversation are generally positive. The statement "What the barkeeper said creates a nice atmosphere" gets a very good result in conversation four (1.75), whereas both conversations without small talk activated get just 0.5. Conversation two does not meet expectations again, since it receives the only negative result of -0.75. The Anova test gives a p-value of 0.0425 for this statement if compared between the uncertain-answer and the small-talk conversations, which could indicate that the uncertain-answers module creates a nicer atmosphere. However, comparison is complicated because of the two very different values for small-talk conversations (see figure 9.4).

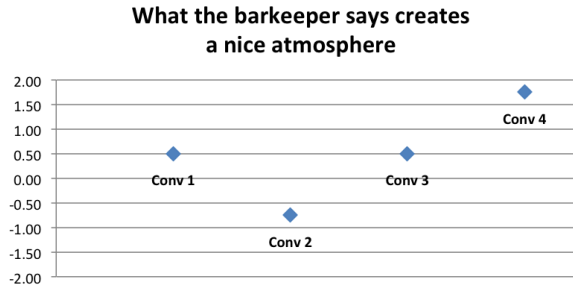

Figure 9.4: The Mean Values for Conversation Atmosphere

The two other statements in this class do not show consistent perception. "The barkeeper is a nice person", the most general statement regarding the *likability* of the agent gets no negative values. Again, conversation four has a very good result with an average value of 2.0. Astonishingly, conversation three gets also a good result (1.75), followed

by conversation two (1.25) and at last conversation one with an average value of 0.75.

"I liked the barkeeper's behavior" has the most positive result in conversation one (without small talk but with uncertain answers): 1.5. This value is directly followed by conversation four (1.25). Conversation three nevertheless gets an average value of 0.75. Only conversation two has a negative result, with the average value of -0.25.

Naturalness Regarding the *naturalness* of the conversation, the uncertain answer module seems to play an important role. Conversations three and four without uncertain answers, both have very negative results, with an average value of -2.5 for the statement "The conversation could be the same between two humans", whereas conversations one and two get -1.25 and -0.75 respectively. The Anova p-value for the two classes is 0.033.

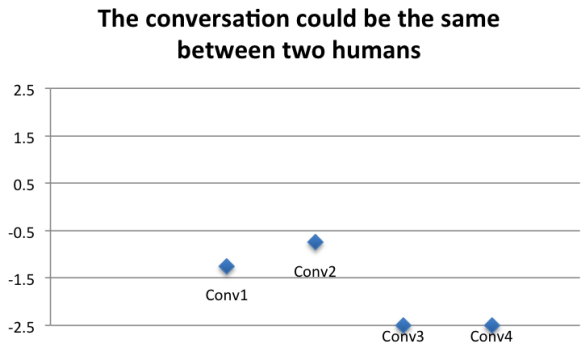

Figure 9.5: The Mean Values for Naturalness

This observation is supported by the results to the statement "The barkeeper behaves naturally", but only for conversation one. While conversation one has a good result yielding an average value of 0.75, and conversations three and four both have negative results (-0.75 and -0.25), conversation two gets the very poor result of -1.5, even though it should have a good value to confirm the hypothesis. This is one of the cases in which the evaluation of conversation two shows very different results from what was expected. For the statement "The conversation feels like a real-world conversation" all reported values are negative. However, they are much closer together, so it is difficult to determine any differences.

The only remarkable thing is that conversations one and three (without small talk) are rated a little better (both -1.0) than conversations four (-1.25) and two. Again conversation two gets very poor results with an average value of -1.75, to which one explanation has already been postulated.

One piece of evidence might be that the users also explicitly rated the "Answers of the barkeeper are natural" very negatively for conversation two (-1.75) although all other conversations were rated quite differently. Conversation three also gets a negative average value of -1.0, conversation four has a positive value of 0.25 and conversation one is rated neutrally (0.0).

Fun to Use The results for *fun to use* are expected to show the benefit of the two SOX components. In general this assumption is confirmed: In two out of three statements conversation three gets the most negative results. The most negative value for the statement "The conversation seems interesting", which received negative results for all conversations, is assigned to conversation three, with an average value of -1.75. Conversations one and two are rated better with both having a value of -0.5. However, conversation two again also has the negative result of -1.5.

A similar result is found for the statement "The conversation was certainly fun for the user". Conversations one and four have positive results, especially conversation four, which gets an average value if 2.0. This could mean that having small talk activated enhances the fun of using the system. Unfortunately, conversation two, the other dialogue with small talk activated, which could have supported this observation, again has a negative result of -1.25. Conversation one is the only other conversation to get another positive value, which is 0.5. Conversation three was negativity rated, with an average value of -1.0. Again, conversation two cannot be used to prove the benefit of the small-talk module. This might be caused by users heavily disliking parts of the conversation.

The statement "I would have been bored in place of the user" is once again mostly reinforced by the participants watching conversation three (average value: 1.0). In contrast, conversation one and four get better values: -0.75 and -0.5 respectively. Conversation two was judged negatively again with an average value of 0.75.

Summing up, the results for *fun to use* are complicated to understand. On the one hand, small talk seems to play an important role, but this result is not backed up by the values for conversation two. The Anova tests cannot provide significant p-values for any of the statements. In

general, conversations one and four, which include one SOX component each, have the best values.

Conversation Flow The three statements "The conversation flowed smoothly", "I had the feeling the conversation often got stuck", and "The barkeeper handles topics and dialogue threads very flexibly" should indicate the perception of the *conversation flow*. These statements have rather negative values, especially the statement "The conversation flowed smoothly", with values from -0.25 (conversation one) to -0.75 (conversation two), -1.25 (conversation four) and right down to -2.0 for conversation 3. Although the values are all negative, a smallish difference can be seen between the conversations including uncertain answers and the conversations excluding uncertain answers: Conversations one and two get fewer negative values than conversation three and four. The Anova p-value for the comparison of the classes is 0.0343.

However, this observation cannot be confirmed by the values for "I had the feeling the conversation often got stuck", which do not differ very much at all. Conversation one, two and three all get an average value of 1.5. Conversation four has 1.75.

The results for the last statement "The barkeeper handles topics and dialogue threads very flexibly." are generally slightly better. Conversation four with small talk activated gets an average value of 1.0, which supports the expectations. The small-talk module enables more topics and a very flexible handling of dialogue threads. However, this observation is not mirrored by the value for conversation two, which gets an average result of -0.5. Conversation three gets an even worse result of -1.25. Conversation one is rated neutral. However, the baseline against the other conversations' Anova test results in a p-value of 0.0256.

In a nutshell one can say that from three mostly negative results in the category of *conversation flow*, two are assigned to conversation three. The integration of both SOX components seems to make a difference in the perception of the conversation flow, since two statements get a significant p-value in the Anova test, testing the baseline against the rest.

The statement "I would like to talk to the barkeeper myself" indicates the overall usability of the system. The responses to this statement indicate the relationship between the above-described measures (utterance understanding, utterance generation, intelligence, likability, naturalness, fun to use, and conversation flow) and the benefit of the application. The

goal is not only to find out if the given measures are rated better with the SOX components or without, but also how this is perceived by the users from a usability point of view. The question is: *Do the users like the application better?*

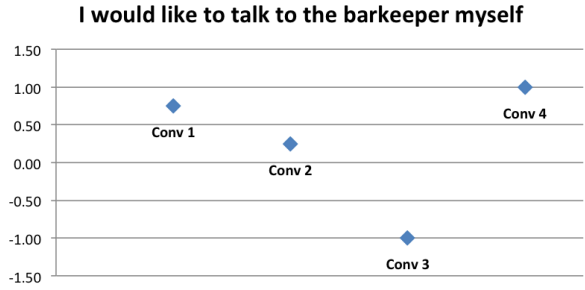

Figure 9.6: The Mean Values for the Willingness to Use the System

The feedback for this statement is positive for all conversations which include SOX components (see figure 9.6): Conversation one has an average value of 0.75, conversation four gets the best results (1.0) and even conversation two, which gets so many negative results, has an average value of 0.25. The only conversation that gets negative results for this statement is conversation three, the one without any SOX component: -1.0. The p-value in the Anova test comparing the baseline against the other conversations is 0.0998. These values confirm the assumption that the SOX components provide a clear benefit for applications such as the conversational agent.

Keeping in mind that conversation two also gets a positive value for the statement "I would like to talk to the barkeeper myself" and the fact that the conversation four, which includes small talk, gets the best result for this statement, it seems particularly odd, that conversation two gets so many negative results. Picking up the two explanations from the beginning of this section (either that the combination of both SOX components makes the conversation worse or that something is completely displeasing in conversation two), the results seem to favor the second interpretation. There are several details which indicate that, in conversation two, the barkeeper said things which spoil the results in the evaluation. One strong piece of evidence is the negative results for the statements measuring the "naturalness" of the system, especially

for “The answers of the barkeeper are natural.”. The participants who watched conversation 2 rated the system with an overall value of -1.75. This is a remarkable difference to the values for the other conversations (0.0, -1.0, 0.25). How they judged the statement “The barkeeper behaves naturally” was similar.

In addition to the statements, the questionnaire gave the participants the opportunity to explain in open-end questions why a conversation did not feel natural and what the main differences were. These values are very useful for understanding the problems with conversation two. For example, one participant mentions the following part of the conversation as particularly unsatisfactory:

USER: I like the drink!

AGENT: Actually, I don’t care about drinks.

This answer from the barkeeper is produced by the uncertain-answer module, because the user’s compliment was not understood. Although this answer fits in the conversation perfectly, it seem to make no sense in conversation with a barkeeper. That may indicate that the conversation contains problems on a higher pragmatic level.

Another participant states that the barkeeper compliments the user largely to “suck up to the user”. Actually, the small-talk module in this conversation initiates one compliment thread in the course of the conversation. In conversation four, which also includes small talk, no compliment thread is initialized. A selection of possible small-talk threads is made at random. The user in conversation two, on the other hand, pays two compliments to the agent. Although the user’s objection is not easily comprehend, it is clear that something is unpleasant. Maybe a further check on the effect of compliments in conversation is needed.

9.3 Conclusion

This chapter presents the results of two evaluations. The first evaluation is about the usability of the *KomParse* conversational agent barkeeper application, which is the test environment for the research and technology described in this thesis. The second evaluation tries to get a deeper insight into the benefits of the two main technological novelities used for integration of social talk in dialgue systems: the SOX components for handling out-of-domain utterances and the SOX component encapsulating the social talk model.

The first evaluation (section 9.1) was carried out through a field test with the barkeeper agent in the virtual online game. Participants talked to the barkeeper agent in a virtual bar and afterwards filled out a questionnaire regarding the usability of the system. This evaluation shows a very positive attitude towards the barkeeper agent. System usability is rated very positively, especially the entertaining functionality. The participants certify that the agent is a clear benefit for the online game application.

The goal of the second evaluation (section 9.2) was to find out how the usability feedback of the first evaluation is related to the social talk research and technology presented in this thesis. The evaluation followed a two-step approach. Firstly, different conversations with four different setups of the dialogue system were video-recorded. The participants talked to the barkeeper agent again, but the SOX components (small talk and uncertain answers) were either activated or deactivated. Secondly, the videos of these conversations were watched and rated by several other participants. Rating was done by a questionnaire, focusing on the measures of intelligence, naturalness, likability, fun to use, utterance understanding, utterance generation, and dialogue flow. Thereby, the special benefit and effect of the single components were measured. Although this is complicated to show because single components and phenomena cannot be cleanly singularized in a dialogue, the results encourage the conclusion that the SOX components are important originators of the mentioned measures and the overall user satisfaction. There is a clear distinction between the willingness of the participants who watched the conversations with SOX components to use the system, and the willingness of those who watched conversations without. Social talk capabilities seem to play an important role regarding the fun to use the system and other measures such as naturalness, conversation flow and natural-language understanding. However, some of the video-recordings could not be completed due to technical problems. More dialogues are needed to further confirm the evaluation observations. Future work will therefore focus on the recording of more conversations with the different set-ups of the dialogue system.

10 Conclusion

This thesis addresses the research goal of enabling social talk in dialogue systems. Dialogue systems have become part of many everyday technologies and applications. They are embedded into a lot of different applications such as a banking hotlines, journey planner systems, smart phones, or websites. Although many of these applications are confronted with social talk, particularly the ones that have personality features, and although the necessity of social talk is well known and observed in a variety of research, existing solutions to social-talk integration are poor. The few solutions which exist are often based on either uncontrollable, external chatbot components or ad-hoc solutions without much knowledge.

This thesis presents a set of extensions to dialogue systems that enable social talk based on well-found empirical and theoretical work and new approaches to essential research questions. The main research areas covered in this thesis are:

Social Talk How can we model social talk in an abstract way well-founded by theoretical groundwork?

Error Handling How can we handle understanding errors caused by social out-of-domain utterances in a way that is appropriate for entertaining applications?

Dialogue Management How can graph-based dialogue management become powerful enough to allow for multiple interwoven conversation threads?

Evaluation How do people actually perceive the integration of social talk and how is their opinion related to the usability of an embedding application?

Moreover, there are several other necessary research foci which are closely related to these research areas. The thesis therefore also deals

with challenges in the field of natural-language understanding, which are *dialogue-act recognition*, *domain classification*, and *topic detection* for single utterances.

The thesis provides extensions to existing dialogue systems as a technological solution for all of the mentioned research questions. The extensions are packaged into a toolkit called SOX (SOcial talk eXtensions). SOX offers several components that can be deployed to dialogue systems as a bundle, but also each on its own. The components concern many different aspects of dialogue-system architectures such as natural-language interpretation and dialogue management.

Although SOX aims at being usable with a lot of different dialogue systems, a simple switch to SOX is not possible. Knowledge about the architectures of dialogue systems is necessary, because the extensions have to be applied to several parts of an existing dialogue system. Therefore, chapter 2 starts with a description of dialogue systems, possible architectures, their components, benefits, and drawbacks. We have seen that there are many different ways to implement dialogue systems differing, for example, in the development of the dialogue manager.

Chapter 3 describes the dialogue system that was used as the test bed for integration of the SOX modules for this research. The dialogue system originates from the research project *KomParse*, in which two conversational agents, a furniture seller and a barkeeper, were developed for a virtual online game. The barkeeper virtual agent is the test bed for the usability evaluation described in chapter 9.

The *KomParse* dialogue system is a graph-based system that includes several knowledge sources and frames to achieve further flexibility. Edges in the graph contain conditions over knowledge base results or user input. For further abstraction, conditions over user input are expressed as conditions over dialogue acts. Natural-language understanding first analyzes the linguistic information in the incoming utterance and than tries to detect the correct dialogue act.

In chapter 4 existing research work related to the focal research points of this thesis is presented. This includes the state of the art in dialogue management, error handling, dialogue act recognition, domain recognition, and social talk for conversational agents. Although the chapter shows that social talk is an important aspect of conversation with dialogue systems, the best attempts in relevant fields of research reveal a lack of solutions and experience. Several authors have, for example, integrated social talk into their conversational agents, but nearly all of

the existing approaches are just ad-hoc solutions without any knowledge about the field of social talk.

Chapter 5 summarizes research results and technological solutions for extending the necessary aspects of *natural-language understanding*. A new approach to dialogue-act recognition combining syntactic and semantic relations is also presented, as well as a new minimally supervised approach to in-domain classification based on a novel data-driven approach to topic recognition. The components encapsulating the particular methods and research results are a dialogue-act recognizer, a domain classifier, and a topic recognizer, which are helper components in the SOX toolkit and can also be used on their own. Evaluation results for the dialogue-act recognizer and the domain classifier are presented.

Chapter 6 introduces the first of the three main research areas and describes an approach to *dialogue management* that modifies graph-based dialogue management to use conversation threads as basic, constitutive elements. This is an important step for the integration of social talk, because it tackles the problem that one cannot know beforehand when, how, and how often a user may initiate social talk. Moreover, social talk is often interwoven with other talk. Thus *multi-threading support* is a crucial feature for social-talk integration. Conversation threads encapsulate parts of the overall conversation graph belonging to one abstract conversation goal. This is a novel way to realize dialogue graphs, which is motivated by discourse analysis and also solves the problem of common graph-based approaches which are not flexible enough to jump from one state to a state in another part of the graph outside the given order. The approach is comparable to plan- and goal-based dialogue management. Conversation threads can be seen to include possible conversation sequences belonging to one abstract goal. However, no component is needed to translate a goal into dialogue moves, because graphs already encode dialogue moves inside the conversation threads. The chapter also describes the results of the thread-selection algorithm evaluation.

Chapter 7 is dedicated to the second main research focus and one of the two main SOX components: *social talk*. The chapter describes empirical ground-work regarding social talk (or "small talk") from social science and linguistics. A new set of dialogue acts for social talk is developed. Existing dialogue-act sets lack important parts of social acts. The dialogue acts are used to annotate a dialogue corpus, and inter-annotator agreement is presented. In addition, a model of possible social dialogue threads containing communication patterns for social talk is learned from the annotated dialogues. The resulting model is part of the social-talk

SOX component, which is one of the integral parts for extending social talk in dialogue systems.

The second integral part of the social-talk extension and the third main research focus is the *intelligent handling of out-of-domain utterances*, which is described in chapter 8. If a dialogue system enables social talk it will definitely be confronted with many utterances that cause understanding errors, because they are out of the knowledge domains of the system, so-called "out-of-domain utterances". Therefore, the integration of social talk and intelligent strategies to handle understanding errors caused by out-of-domain utterances are a necessary combination. The chapter first focuses on strategies from human-human interaction, which hide understanding problems. Because data for hidden understanding errors does not exist, several sources for strategies are used, such as behavior from hearing-impaired people described in psychology. These strategies are part of the second main SOX component: the *uncertain-answer module*. The module handles incoming out-of-domain utterances by applying one of the strategies and using as much linguistic information as possible, such as recognized topics, to generate a reaction.

An *evaluation* of the test bed dialogue system including the SOX components is described in 9. The chapter contains two evaluations, one regarding the overall usability of a conversational-agent application, which includes SOX. The second evaluation aims at the perception of the two main SOX components regarding measures such as *naturalness* and their relationship to system usability. In general one can say that both evaluations show that the participants liked the application a lot. In particular, the fun they had while using the application seems to be related to the system's social ability. Both SOX components seem to be important for the usability of an embedding application.

10.1 Future Research

There are many areas for interesting future investigation. One future research plan includes more experiments for cross-checking some of the evaluation results. This especially concerns the dialogue-act recognition approach described in chapter 5 and the SOX-component evaluation described in chapter 9. While the dialogue-act recognition approach described in chapter 5 is indeed used as the dialogue-act classification solution for the overall running system, the evaluation given in chapter 5 was done before some of the other extensions were developed and therefore contains only a subset of all possible dialogue acts. The system was

running with the full set of small talk and other talk dialogue acts for the evaluations in chapter 9. However, as described in chapter 3, the dialogue-act recognition uses a hybrid approach incorporating the classifier on the one hand and a rule-based approach on the other. Because it would be interesting to know how classifier and rule-based system would be evaluated with the full set of dialogue acts, further investigation is intended on the approach of linguistic dialogue-act recognition and its evaluation in new applications and domains.

Another important point is to carry out a cross-check of the second evaluation described in section 9.2. A better understanding of the values for conversation two presented in this evaluation is needed. Although conversation two includes all of the described extensions, the evaluated conversation got surprisingly poor results in the evaluation, in parts. This is particularly astonishing because the two other conversations one and four, which include just parts of the extensions, got much better values than the baseline system. In the overall rating conversation two is still better than the baseline system, but cannot beat conversations one and four. A new evaluation using more video-recorded conversations with the different system's set-ups may give a clue to whether the supposition is true that the conversation two described in chapter 9 accidentally contained some phenomena or utterances from the system that were unacceptable to the users. The cross-check could suspend the unlikely interpretation that the combination of all extensions may turn out to be counterproductive.

Another planned future work is the further testing of the interoperability of the suggested extensions. The toolkit is developed to be as independent from dialogue system's architectures as possible, but integrated and tested only with the extended finite-state dialogue manager described in chapter 3. Plans to use the toolkit with other dialogue managers already exist. For exhaustive testing it should optimally be applied to several dialogue systems with different kinds of dialogue management approaches such as an information-state update model or a plan-based system. Even interoperability testing with a probabilistic dialogue manager is planned. The goal is a toolkit which is easy to understand, usable for several kinds of dialogue systems and straightforward to integrate into new applications, while it also provides enough interfaces on all necessary levels to enable communication with the mother system. Optimally, it would provide the pre-defined social-talk content as well as an easy way to extend the system with social talk desired by the developer of a particular system.

Bibliography

Adolphs, P., Benz, A., Bertomeu, N., Cheng, X., Klüwer, T., Krifka, M., et al. (2011, 10). Conversational Agents in a Virtual World. In *Proceedings of the 34th annual german conference on artificial intelligence.* Springer.

Adolphs, P., Cheng, X., Klüwer, T., Uszkoreit, H., & Xu, F. (2010). Question Answering Biographic Information and Social Network Powered by the Semantic Web. In *Proceedings of LREC 2010.* Malta.

Allan, J. (Ed.). (2002). *Topic detection and tracking: event-based information organization.* Norwell, MA, USA: Kluwer Academic Publishers.

Allen, J., Chambers, N., Ferguson, G., Galescu, L., Jung, H., Swift, M., et al. (2007). PLOW: a collaborative task learning agent. In *Proceedings of the 22nd national conference on artificial intelligence - volume 2* (pp. 1514–1519). AAAI Press.

Allen, J., & Core, M. (1997). *DAMSL: Dialogue Act Markup in Several Layers (Draft 2.1).*

Allen, J., Ferguson, G., & Stent, A. (2001). An architecture for more realistic conversational systems. In *Iui '01: Proceedings of the 6th international conference on intelligent user interfaces* (pp. 1–8). New York, NY, USA: ACM.

Allen, J., Manshadi, M., Dzikovska, M., & Swift, M. (2007). Deep linguistic processing for spoken dialogue systems. In *Deeplp '07: Proceedings of the workshop on deep linguistic processing* (pp. 49–56). Morristown, NJ, USA: Association for Computational Linguistics.

Allen, J., & Perrault, C. R. (1980). Analyzing Intention in Utterances. *Artificial Intelligence*, *15*, 143–178.

Allen, J. F., Ferguson, G., Miller, B. W., Ringger, E. K., & Sikorski, T. (2000, July 25). Dialogue Systems: From Theory to Practice in TRAINS-96. In *Handbook of natural language processing* (1st ed.,

pp. 347–376). Marcel Dekker. Hardcover.

Allen, J. F., Miller, B. W., Ringger, E. K., & Sikorski, T. (1996). A robust system for natural spoken dialogue. In *Proceedings of the 34th annual meeting on association for computational linguistics* (pp. 62–70). Morristown, NJ, USA: Association for Computational Linguistics.

Andernach, T. (1996). A Machine Learning Approach to the Classification of Dialogue Utterances. *Computing Resource Repository, cmp-lg/9607022*.

Aust, H., Oerder, M., Seide, F., & Steinbiss, V. (1995, November). The Philips Automatic Train Timetable Information System. *Speech Commun.*, *17*(3-4), 249–262.

Austin, J. L. (1975). *How to do things with words*. Cambridge, Mass.: Harvard University Press.

Baker, C. F., Fillmore, C. J., & Lowe, J. B. (1998). The Berkeley FrameNet Project. In *Proceedings of the 17th international conference on computational linguistics* (pp. 86–90). Morristown, NJ, USA: Association for Computational Linguistics.

Bertomeu, N. (2012). Finding optimal presentation sequences for a conversational Recommender System. In *Proceedings of the international conference on information processing and management of uncertainty in knowledge-based systems (ipmu-12).* Catania, Sicilia: Springer Verlag.

Bertomeu, N., & Benz, A. (2009). Annotation of Joint Projects and Information States in Human-NPC Dialogues. In *Proceedings of the first international conference on corpus linguistics (CILC-09).* Murcia, Spain.

Bickmore, T. (1999). *A Computational Model of Small Talk.* Retrieved from `http://web.media.mit.edu/~bickmore/Mas962b/`

Bickmore, T. (2003). *Relational Agents: Effecting Change through Human-Computer Relationships*. Unpublished doctoral dissertation, Massachusetts Institute of Technology.

Bickmore, T. W., & Cassell, J. (2000). "How about this weather?" Social Dialog with Embodied Conversational Agents. In *Proceedings of the aaai fall symposium on socially intelligent agents.*

Bickmore, T. W., & Cassell, J. (2001). Relational agents: a model and implementation of building user trust. In *Chi* (p. 396-403).

Bickmore, T. W., & Picard, R. W. (2005). Establishing and maintaining long-term human-computer relationships. *ACM Transactions on Computer-human Interaction*, *12*, 293–327.

Bircher-Müller, U. (1997). *Der schwerhörige Patient.* Quintessenz, MMV-Medizin-Verlag.

Blascovich, J. (2002). A theoretical model of social influence for increasing the utility of collaborative virtual environments. In *Proceedings of the 4th international conference on collaborative virtual environments* (pp. 25–30). New York, NY, USA: ACM.

Bobbert, D., & Wolska, M. (2007). Dialog OS: An extensible platform for teaching spoken dialogue systems. In *Decalog '07: Workshop on the semantics and pragmatics of dialogue.* Rovereto, Italy.

Bohus, D. (2007). *Error awareness and recovery in conversational spoken language interfaces.* Unpublished doctoral dissertation, Pittsburgh, PA, USA. (AAI3277260)

Bohus, D., & Rudnicky, A. I. (2003). RavenClaw: Dialog Management Using Hierarchical Task Decomposition and an Expectation Agenda. In *Interspeech.* ISCA.

Bohus, D., & Rudnicky, A. I. (2005). Sorry, I didn't catch that! An investigation of non-understanding errors and recovery strategies. In *Carnegie mellon university research showcases.* Computer Science Department.

Bohus, D., & Rudnicky, A. I. (2008). Sorry, I Didn't Catch That! In L. Dybkjær, W. Minker, & N. Ide (Eds.), *Recent trends in discourse and dialogue* (Vol. 39, p. 123-154). Springer Netherlands.

Bohus, D., & Rudnicky, A. I. (2009). The RavenClaw dialog management framework: Architecture and systems. *Computer Speech & Language*, *23*(3), 332 - 361.

Bratman, M. E., Israel, D. J., & Pollack, M. E. (1988). Plans And Resource-Bounded Practical Reasoning. *Philosophy and AI: Essays at the interface*, 1–22.

Brinker, K., & Sager, S. (1989). *Linguistische Gesprächsanalyse.* Schmidt.

Bui, T., Rajman, M., & Melichar, M. (2004, September). Rapid Dialogue Prototyping Methodology. In P. Sojka, I. Kopecek, & K. Pala (Eds.), *Proceedings of the 7th international conference on text, speech dialogue (tsd)* (Vol. 3206/2, pp. 579–586). Berlin Heidelberg New York: Springer Verlag.

Bunt, H. (2011, April). Multifunctionality in dialogue. *Comput. Speech Lang.*, *25*, 222–245.

Bunt, H., Alexandersson, J., Carletta, J., Choe, J.-W., Fang, A. C., Hasida, K., et al. (2010, may). Towards an ISO Standard for Dialogue Act Annotation. In N. Calzolari (Ed.), *Proceedings of the*

seventh conference on international language resources and evaluation (lrec'10). Valletta, Malta: European Language Resources Association (ELRA).

Carletta, J., & Isard, A. (1996). *HCRC Dialogue Structure Coding Manual* (Tech. Rep.). Centre, University of Edinburgh.

Cassell, J., Bickmore, T., Billinghurst, M., Campbell, L., Chang, K., Vilhjálmsson, H., et al. (1999). Embodiment in Conversational Interfaces: REA. In *Proceedings of the sigchi conference on human factors in computing systems: the chi is the limit* (pp. 520–527). New York, NY, USA: ACM.

Clark, H. (1996). *Using Language*. Cambridge University Press. Paperback.

Cohen, P. (1997). Dialogue Modeling. In R. Cole, J. Mariani, H. Uszkoreit, A. Zaenen, & V. Zue (Eds.), *Survey in the state of the art in human language technology.* Cambridge University Press.

Constantinides, P. C., Hansma, S., Tchou, C., Rudnicky, A. I., & Rudnicky, E. I. (1998). A Schema Based Approach To Dialog Control. In *Proceedings of the international conference on spoken language processing* (pp. 409–412).

Copestake, A. (2007). Semantic composition with (robust) minimal recursion semantics. In *Proceedings of the workshop on deep linguistic processing* (pp. 73–80). Stroudsburg, PA, USA: Association for Computational Linguistics.

Coupland, J., Coupland, N., & Robinson, J. D. (1992). "How are you?": Negotiating phatic communion. *Language in Society*, *21*(02), 207–230.

Denecke, M. (2002). Rapid prototyping for spoken dialogue systems. In *Proceedings of the 19th international conference on computational linguistics - volume 1* (pp. 1–7). Stroudsburg, PA, USA: Association for Computational Linguistics.

Dybkjær, L., Hemsen, H., & Minker, W. (Eds.). (2007). *Evaluation of Text and Speech Systems* (Vol. 37). Springer.

Endrass, B., Rehm, M., & Andre, E. (2011). Planning Small Talk Behavior with Cultural Influences for Multiagent Systems. *Computer Speech and Language*, *25*(2), 158 - 174.

Firby, R. J. (1994). Task Networks for Controlling Continuous Processes. In *In proceedings of the second international conference on ai planning systems* (pp. 49–54).

Fujita, Y., Takeuchi, S., Kawanami, H., Matsui, T., Saruwatari, H., & Shikano, K. (2011, October). Out-of-Task Utterance Detection

Based on Bag-of-Words Using Automatic Speech Recognition Results. In *Proceedings of the 2011 asia-pacific signal and information processing association annual summit and conference.* APSIPA.

Gebhard, P., Kipp, M., Klesen, M., & Rist, T. (2003). Authoring Scenes for Adaptive, Interactive Performances. In *Proc. of the second international joint conference on autonomous agents and multiagent systems (aamas'03).*

Goddeau, D., Meng, H., Polifroni, J., Seneff, S., & Busayapongchaiy, S. (1996). A Form-Based Dialogue Manager For Spoken Language Applications. In *In proc. icslp* (pp. 701–704).

Goffman, E. (1967). *Interaction Ritual: Essays on Face-to-Face Behavior.* Garden City, New York: Anchor Books, Doubleday & Company, Inc.

Goffman, E. (1971). *Relations in public; Microstudies of the public order.* Basic Books.

Grice, H. P. (1975). Logic and conversation. In P. Cole & J. L. Morgan (Eds.), *Syntax and semantics* (Vol. 3). New York: Academic Press.

Gustafson, J., Bell, L., Boye, J., Lindström, A., & Wirén, M. (2004). The NICE fairy-tale game system. In *Proceedings of the 5th sigdial workshop on discourse and dialogue at hlt-naacl 2004.*

Harel, D. (1987, June). Statecharts: A visual formalism for complex systems. *Sci. Comput. Program.*, *8*(3), 231–274.

Harel, D. (1988, May). On visual formalisms. *Commun. ACM*, *31*, 514–530.

Heeman, P. A., Yang, F., Kun, A. L., & Shyrokov, A. (2005). Conventions in human-human multi-threaded dialogues: a preliminary study. In *Proceedings of the 10th international conference on intelligent user interfaces* (pp. 293–295). New York, NY, USA: ACM.

Henderson, M., Matheson, C., & Oberlander, J. (2012). Recovering from Non-Understanding Errors in a Conversational Dialogue System. In *Proceedings of the 16th workshop of the semantics and pragmatics of dialogue (seinedial).*

Holly, W. (1979). *Imagearbeit in Gesprächen. Zur linguistischen Beschreibung des Beziehungsaspekts.* Tübingen: Niemeyer.

Isbister, K., Nakanishi, H., Ishida, T., & Nass, C. (2000). Helper Agent: Designing An Assistant for Human-Human Interaction in a Virtual Meeting Space. In (pp. 57–64).

Iurgel, I. (2006). Cyranus - An Authoring Tool for Interactive Edutainment Applications. In *Lecture notes in computer science* (Vol. 3942, pp. 577–580). Springer.

Jokinen, K., & McTear, M. F. (2009). *Spoken Dialogue Systems*. Morgan & Claypool Publishers.

Jurafsky, D., Schriberg, E., & Biasca, D. (1997). *Switchboard SWBD-DAMSL Shall-Discourse-Function Annotation Coders Manual.* Retrieved from `http://stripe.colorado.edu/~jurafsky/manual.august.html`

Jurafsky, D., Shriberg, E., Fox, B., & Curl, T. (1998). *Lexical, Prosodic, and Syntactic Cues for Dialog Acts.*

Kamp, H. (1981). A Theory of Truth and Semantic Representation. In J. A. G. Groenendijk, T. M. V. Janssen, & M. B. J. Stokhof (Eds.), *Formal methods in the study of language* (Vol. 1, pp. 277–322). Amsterdam: Mathematisch Centrum.

Kamp, H., & Reyle, U. (1993). *From Discourse to Logic: Introduction to Model Theoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory* (No. Bd. 1). Kluwer.

Keizer, S., Akker, R. op den, & Nijholt, A. (2002). Dialogue act recognition with Bayesian networks for Dutch dialogues. In *Proceedings of the 3rd sigdial workshop on discourse and dialogue* (pp. 88–94). Morristown, NJ, USA: Association for Computational Linguistics.

Keizer, S., & Akker, R. o. d. (2006). Dialogue act recognition under uncertainty using bayesian networks. *Nat. Lang. Eng.*, *13*(4).

Klesen, M., Kipp, M., Gebhard, P., & Rist, T. (2003). Staging Exhibitions: Methods and tools for modelling narrative structure to produce interactive performances with virtual actors.

Klüwer, T. (2009). RMRSBot - Using Linguistic Information to Enrich a Chatbot. In *Intelligent virtual agents. 9th international conference, proceedings.* Springer Berlin / Heidelberg.

Klüwer, T. (2011a, 7). From Chatbots to Dialogue Systems. In D. Perez-Marton & I. Pascual-Nieto (Eds.), *Conversational agents and natural language interaction: Techniques and effective practices* (p. 1-22). IGI Global Publishing Group.

Klüwer, T. (2011b). "I like your shirt" - Dialogue Acts for Enabling Social Talk in Conversational Agents. In *Proceedings of the 11th international conference on intelligent virtual agents.* Springer.

Klüwer, T. (2012). A Multi-threading Extension to State-based Dialogue Management. In S. Brown-Schmidt, J. Ginzburg, & S. Larsson (Eds.), *Proceedings of semdial 2012 (seinedial): The 16th workshop on the semantics and pragmatics of dialogue.* n/a.

Klüwer, T., Adolphs, P., Xu, F., & Uszkoreit, H. (2011). A Dialogue System for Conversational NPCs. In *Paralinguistic information*

and its integration in spoken dialogue systems. Springer.

Klüwer, T., Adolphs, P., Xu, F., & Uszkoreit, H. (2012, forthcoming). Evaluation of the KomParse Conversational Non-Player Characters in a Commercial Virtual World. In *Proceedings of the eighth international conference on language resources and evaluation conference (lrec).*

Klüwer, T., Adolphs, P., Xu, F., Uszkoreit, H., & Cheng, X. (2010). Talking NPCs in a Virtual Game World. In *Proceedings of the system demonstrations section at acl 2010.*

Klüwer, T., Uszkoreit, H., & Xu, F. (2010). Using Syntactic and Semantic based Relations for Dialogue Act Recognition. In *Coling 2010: Posters* (pp. 570–578). Beijing, China: Coling 2010 Organizing Committee.

Komatani, K., & Kawahara, T. (2000). Generating effective confirmation and guidance using two-level confidence measures for dialogue systems. In *Interspeech* (p. 648). ISCA.

Kopp, S., Gesellensetter, L., Krämer, N., & Wachsmuth, I. (2005). A conversational agent as museum guide – design and evaluation of a real-world application. In *Proc. of intelligent virtual agents (iva 2005)* (Vol. 3661, pp. 329–343). Springer.

Koulouri, T., & Lauria, S. (2009). Exploring miscommunication and collaborative behaviour in human-robot interaction. In *Proceedings of the sigdial 2009 conference: The 10th annual meeting of the special interest group on discourse and dialogue* (pp. 111–119). Stroudsburg, PA, USA: Association for Computational Linguistics.

Krahmer, E., Swerts, M., Theune, M., & Weegels, M. (2001, March). Error detection in spoken human-machine interaction. *International Journal of Speech Technology*, *4*(1), 19–29.

Krug, E., & Claußen, W. (1949). *Charakter und Schwerhörigkeit* (No. Bd. 1). Ed. Harmsen.

Kruijff, G.-J. M., Lison, P., Benjamin, T., Jacobsson, H., Zender, H., & Kruijff-Korbayová, I. (2010). Situated Dialogue Processing for Human-Robot Interaction. In H. I. Christensen, G.-J. M. Kruijff, & J. L. Wyatt (Eds.), *Cognitive systems* (Vol. 8, pp. 311–364). Berlin/Heidelberg, Germany: Springer Verlag.

Lane, I. R., Kawahara, T., Matsui, T., & Nakamura, S. (2007). Out-of-Domain Utterance Detection Using Classification Confidences of Multiple Topics. *IEEE Transactions on Audio, Speech & Language Processing*, *15*(1), 150-161.

Lapata, M., & Lascarides, A. (2004). Inferring Sentence-Internal Temporal Relations. In *Proceedings of the north american chapter of the association for computational linguistics* (pp. 153–160).

Larsson, S., & Traum, D. (2000). Information state and dialogue management in the TRINDI dialogue move engine toolkit. *Natural Language Engineering*, *6*, 323–340.

Lee, C., Jung, S., Lee, D., & Lee, G. G. (2007). Example-based error recovery strategy for spoken dialog system. In S. Furui & T. Kawahara (Eds.), *Asru* (p. 538-543). IEEE.

Lemon, O., & al. et. (2003). Managing Dialogue Interaction: A Multi-Layered Approach. In *In proceedings of the 4th sigdial workshop on discourse and dialogue* (pp. 168–177).

Lemon, O., Gruenstein, A., Battle, A., & Peters, S. (2002). Multi-tasking and collaborative activities in dialogue systems. In *Proceedings of the 3rd sigdial workshop on discourse and dialogue - volume 2* (pp. 113–124). Stroudsburg, PA, USA: Association for Computational Linguistics.

Lesh, N., Marks, J., Rich, C., & Sidner, C. L. (2004). "Man-Computer Symbiosis" Revisited: Achieving Natural Communication and Collaboration with Computers. *IEICE Transactions*, *87-D*(6), 1290-1298.

Leuski, A., Patel, R., & Traum, D. (2006). Building effective question answering characters. In *In proceedings of the 7th sigdial workshop on discourse and dialogue* (pp. 18–27).

Malinowski, B. (1949). The Meaning of Meaning: A Study of Influence of Language Upon Thought and of the Science of Symbolism. In (10th edition ed., p. 296-336). New York: Harcourt, Brace and World.

Mann, W. C., & Thompson, S. A. (1988). Rhetorical structure theory: Toward a functional theory of text organization. *Text*, *8*(3), 243–281.

Marneffe, M. C. de, & Manning, C. D. (2008). The Stanford Typed Dependencies Representation. In *Coling 2008: Proceedings of the workshop on cross-framework and cross-domain parser evaluation.* Manchester, UK.

McTear, M. (1998). Modelling Spoken Dialogues with State Transition Diagrams: Experiences with the CSLU Toolkit. In *Icslp-98* (Vol. 4, pp. 1223–1226). Sydney.

Mehlman, G. U. (2009). *SceneMaker 3 - An Interpreter for Parallel Processes Modeling Behavior of Interactive Virtual Characters.* Unpublished master's thesis, Saarland University, Faculty of Natural Sciences and Technology I Department of Computer Science.

Möller, S. (2002). A new taxonomy for the quality of telephone services based on spoken dialogue systems. In *Proceedings of the 3rd sigdial workshop on discourse and dialogue - volume 2* (pp. 142–153). Stroudsburg, PA, USA: Association for Computational Linguistics.

Möller, S. (2005). *Quality of Telephone-Based Spoken Dialogue Systems* (1. Aufl. ed.). Berlin, Heidelberg: Springer.

Nakano, M., Hasegawa, Y., Funakoshi, K., Takeuchi, J., Torii, T., Nakadai, K., et al. (2011, March). A multi-expert model for dialogue and behavior control of conversational robots and agents. *Know.-Based Syst.*, *24*(2), 248–256.

Nass, C., Steuer, J., & Tauber, E. R. (1994). Computers are social actors. In *Computer human interaction.*

Neumann, G., & Schmeier, S. (2002). Shallow Natural Language Technology and Text Mining. *Künstliche Intelligenz. The German Artificial Intelligence Journal.*

Palmer, M., Gildea, D., & Kingsbury, P. (2005). The Proposition Bank: An Annotated Corpus of Semantic Roles. *Comput. Linguist.*, *31*(1), 71–106.

Patel, R., Leuski, A., & Traum, D. (2006, August). Dealing with Out of Domain Questions in Virtual Characters. In *Proceedings of the 6th international conference on intelligent virtual agents.* Marina del Rey, CA.

Peckham, J. (1993). A new generation of spoken dialogue systems: results and lessons from the sundial project. In *Eurospeech'93.*

Pedersen, T., Patwardhan, S., & Michelizzi, J. (2004). WordNet::Similarity: measuring the relatedness of concepts. In *Demonstration papers at hlt-naacl 2004* (pp. 38–41). Stroudsburg, PA, USA: Association for Computational Linguistics.

Polanyi, L. (1996). The Linguistic Structure of Discourse. In *Tilburg university, technical report csli-96-200.* (pp. 141–178).

Poppe, R., Truong, K. P., & Heylen, D. (2011). Backchannels: Quantity, Type and Timing Matters. In H. H. Vilhjálmsson, S. Kopp, S. Marsella, & K. R. Thórisson (Eds.), *Intelligent virtual agents* (Vol. 6895, pp. 228–239). Springer.

Prasad, R., Dinesh, N., Lee, A., Miltsakaki, E., Robaldo, L., Joshi, A., et al. (2008). The Penn Discourse TreeBank 2.0. In *In proceedings*

of lrec.

Reinhart, T. (1982). Pragmatics and Linguistics: An Analysis of Sentence Topics. *Philosophica*, *27*, 53–94.

Reiter, E., & Dale, R. (1997, March). Building applied natural language generation systems. *Nat. Lang. Eng.*, *3*(1), 57–87.

Resnik, P. (1995). Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th international joint conference on artificial intelligence - volume 1* (pp. 448–453). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Rich, C., & Sidner, C. L. (1998, February). COLLAGEN: A Collaboration Manager for Software Interface Agents. *User Modeling and User-Adapted Interaction*, *8*(3-4), 315–350.

Rosé, C. P., Di Eugenio, B., Levin, L. S., & Carol. (1995). Discourse processing of dialogues with multiple threads. In *Proceedings of the 33rd annual meeting on association for computational linguistics* (pp. 31–38). Morristown, NJ, USA: Association for Computational Linguistics.

San-Segundo, R., Pellom, B., Ward, W., & Pardo, J. M. (2000). Confidence measures for dialogue management in the CU Communicator system. In *Proceedings of the acoustics, speech, and signal processing, 2000. on ieee international conference - volume 02* (pp. II1237–II1240). Washington, DC, USA: IEEE Computer Society.

Savy, R. (2010, may). Pr.A.Ti.D: A Coding Scheme for Pragmatic Annotation of Dialogues. In N. C. C. Chair) et al. (Eds.), *Proceedings of the seventh international conference on language resources and evaluation (lrec'10).* Valletta, Malta: European Language Resources Association (ELRA).

Schank, G. (1981). *Untersuchungen zum Ablauf natürlicher Dialoge.* M. Hueber.

Schank, R. C. (1977). Rules and Topics in Conversation. *Cognitive Science*, *1*(4), 421-441.

Schegloff, E. A. (2007). *Sequence Organization in Interaction: Volume 1: A Primer in Conversation Analysis.* Cambridge University Press.

Schegloff, E. A., & Sacks, H. (1973). Opening up closings. *Semiotica*, *8*(4), 289–327.

Schlangen, D. (2004). Causes and strategies for requesting clarification in dialogue. In (pp. 136–143).

Schneider, K. (1988). *Small Talk: Analysing Phatic Discourse.* Unpublished doctoral dissertation, Philipps-Universitat, Marburg, Germany.

Schreiber, A. (2000). *Auswirkungen einer Schwerhörigkeit auf die Psyche.* Talk manuscript on webpage. (Available online at `http://www.ohrenseite.de/eb/eb_almuth_schreiber.pdf`; visited on June 8th 2012.)

Schröder, M., Charfuelan, M., Pammi, S., & Steiner, I. (2011). Open source voice creation toolkit for the MARY TTS Platform. In *Proceedings of interspeech 2011.* ISCA.

Schuler, K. K. (2005). *Verbnet: a broad-coverage, comprehensive verb lexicon.* Unpublished doctoral dissertation, Philadelphia, PA, USA.

Searle, J. R. (1969). *Speech acts : an essay in the philosophy of language* [Book]. Cambridge University Press, London.

Shriberg, E., Dhillon, R., Bhagat, S., Ang, J., & Carvey, H. (2004). The ICSI Meeting Recorder Dialog Act (MRDA) Corpus. In *Proc. of the 5th sigdial workshop on discourse and dialogue* (pp. 97–100).

Shyrokov, A., Kun, A., & Heeman, P. (2007). Experimental Modeling of Human-Human Multi-Threaded Dialogues in the Presence of a Manual-Visual Task. In *Proceedings of the sigdial 2007.* Antwerp, Belgium.

Sinclair, J., & Coulthard, M. (1975). *Towards an analysis of discourse: the English used by teachers and pupils.* Oxford University Press.

Skantze, G. (2003). Exploring Human Error Handling Strategies: Implications for Spoken Dialogue Systems. *Speech Communication*, *45(3)*, 325-341.

Spierling, U., Weiß, S. A., & Müller, W. (2006). Towards accessible authoring tools for interactive storytelling. In *Proceedings of the third international conference on technologies for interactive digital storytelling and entertainment* (pp. 169–180). Berlin, Heidelberg: Springer-Verlag.

Sporleder, C., & Lascarides, A. (2008). Using Automatically Labelled Examples to Classify Rhetorical Relations: A Critical Assessment. *Natural Language Engineering*, *14*(3), 369–416.

Spranz-Fogasy, T., & Spiegel, C. (2001). Aufbau und Abfolge von Gesprächsphasen. In K. Brinker, G. Antos, W. Heinemann, & S. Sager (Eds.), *Text- und gesprächslinguistik* (p. 1241-1252). Berlin/New York.

Stent, A. (2000). Rhetorical structure in dialog. In *Proceedings of the first international conference on natural language generation - volume 14* (pp. 247–252). Stroudsburg, PA, USA: Association for Computational Linguistics.

Stolcke, A., Ries, K., Coccaro, N., Shriberg, E., Bates, R., Jurafsky, D., et al. (2000). Dialogue Act Modeling for Automatic Tagging and Recognition of Conversational Speech. *Computational Linguistics*, *26*, 339–373.

Subba, R., & Di Eugenio, B. (2009). An effective discourse parser that uses rich linguistic information. In *Naacl '09: Proceedings of human language technologies: The 2009 annual conference of the north american chapter of the association for computational linguistics* (pp. 566–574). Morristown, NJ, USA: Association for Computational Linguistics.

Suchanek, F., Kasneci, G., & Weikum, G. (2007). YAGO: A Core of Semantic Knowledge - Unifying WordNet and Wikipedia. In *Proc. of www 2007* (pp. 697–706). Banff, Canada.

Surendran, D., & Levow, G.-A. (2006). Dialog Act Tagging with Support Vector Machines and Hidden Markov Models. In *Interspeech.*

Tonelli, S., Riccardi, G., Prasad, R., & Joshi, A. K. (2010). Annotation of Discourse Relations for Conversational Spoken Dialogs. In N. Calzolari et al. (Eds.), *Lrec.* European Language Resources Association.

Uszkoreit, H., Xu, F., Liu, W., Steffen, J., Aslan, I., Liu, J., et al. (2007). A Successful Field Test of a Mobile and Multilingual Information Service System COMPASS2008. In *Proceedings of hci international 2007, 12th international conference on human-computer interaction.*

Verbree, A., Rienks, R., & Heylen, D. (2006). Dialogue-act tagging using smart feature selection: results on multiple corpora. In B. Raorke (Ed.), *First international ieee workshop on spoken language technology slt 2006.* Palm Beach: IEEE Computer Society.

Wahlster, W. (2000). *Verbmobil: Foundations of Speech-to-Speech Translation.* Springer.

Wahlster, W. (2006). *SmartKom: Foundations of Multimodal Dialogue Systems.* Springer.

Walker, M. A., Litman, D. J., Kamm, C. A., & Abella, A. (1997). PARADISE: a framework for evaluating spoken dialogue agents. In *Proceedings of the eighth conference on european chapter of the association for computational linguistics* (pp. 271–280). Morristown, NJ, USA: Association for Computational Linguistics.

Wallace, R., & Bush, N. (2001). *Artificial Intelligence Markup Language (AIML) Version 1.0.1 (2001).* (Unpublished A.L.I.C.E. AI Foundation Working Draft (rev 006))

Webb, N., & Liu, T. (2008). Investigating the Portability of Corpus-Derived Cue Phrases for Dialogue Act Classification. In *Proceedings of the 22nd international conference on computational linguistics (coling 2008)* (pp. 977–984). Manchester, UK.

Weizenbaum, J. (1966, January). ELIZA - a computer program for the study of natural language communication between man and machine. *Commun. ACM*, *9*(1), 36–45.

Wong, W., Cavedon, L., Thangarajah, J., & Padgham, L. (2012). Goal-driven approach to open-ended dialogue management using BDI agents. In *Proceedings of the 11th international conference on autonomous agents and multiagent systems - volume 3* (pp. 1187–1188). Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems.

Xu, F., Adolphs, P., Uszkoreit, H., Cheng, X., & Li, H. (2009). Gossip Galore: A Conversational Web Agent for Collecting and Sharing Pop Trivia. In *Proceedings of ICAART 2009.* Porto, Portugal.

Xu, F., Uszkoreit, H., & Li, H. (2007, June). A Seed-driven Bottom-up Machine Learning Framework for Extracting Relations of Various Complexity. In *Proceedings of the 45th annual meeting of the association of computational linguistics* (pp. 584–591). Prague, Czech Republic: Association for Computational Linguistics.

Yang, F., Heeman, P. A., & Kun, A. (2008). Switching to real-time tasks in multi-tasking dialogue. In *Proceedings of the 22nd international conference on computational linguistics - volume 1* (pp. 1025–1032). Stroudsburg, PA, USA: Association for Computational Linguistics.

Young, S., Gasic, M., Thomson, B., & Williams, J. (2013). POMDP-based Statistical Spoken Dialogue Systems: a Review. *Proc IEEE*, *N/A*, N/A.

Zheng, F., & Webb, G. I. (2006). Efficient lazy elimination for averaged one-dependence estimators. In *Icml* (p. 1113-1120).

Zimmermann, M., Liu, Y., Shriberg, E., & Stolcke, A. (2005). Toward joint segmentation and classification of dialog acts in multiparty meetings. In *Proc. multimodal interaction and related machine learning algorithms workshop (mlmi–05* (p. 187).

Zollo, T. (1999). A study of human dialogue strategies in the presence of speech recognition errors. In *Psychological models of communications in collaborative systemes* (p. 132-139). AAAI Press.

Small talk capabilities are an important but very challenging extension to dialogue systems. Small talk (or "social talk") refers to a kind of conversation, which does not focus on the exchange of information, but on the negotiation of social roles and situations. The goal of this thesis is to provide knowledge, processes and structures that can be used by dialogue systems to satisfactorily participate in social conversations. For this purpose the thesis presents research in the areas of natural-language understanding, dialogue management and error handling. Nine new models of social talk based on a data analysis of small talk conversations are described. The functionally-motivated and content-abstract models can be used for small talk conversations on various topics. The basic elements of the models consist of dialogue acts for social talk newly developed on basis of social science theory. The thesis also presents some conversation strategies for the treatment of so-called "out-of-domain" (OoD) utterances that can be used to avoid errors in the input understanding of dialogue systems. Additionally, the thesis describes a new extension to dialogue management that flexibly manages interwoven dialogue threads. The small talk models as well as the strategies for handling OoD utterances are encoded as computational dialogue threads.

ISBN 978-3-86223-173-7