INTRODUCING ADAPTABILITY TO NATURAL - LANGUAGE SYSTEMS THROUGH USER MODELING

Vasana Jayachamaree Wijesooriya

(138243M)

Degree of Master of Science in Computer Science

Department of Computer Science and Engineering

University of Moratuwa
Sri Lanka

March 2015

INTRODUCING ADAPTABILITY TO NATURAL - LANGUAGE SYSTEMS THROUGH USER MODELING

Vasana Jayachamaree Wijesooriya

(138243M)

Dissertation submitted in partial fulfillment of the requirements for the Degree Master of Science in Computer Science

Department of Computer Science and Engineering

University of Moratuwa
Sri Lanka

March 2015

DECLARATION

I declare that this is my own work and this dissertation does not incorporate without acknowledgement any material previously submitted for a Degree or Diploma in any other University or institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Also, I hereby grant to University of Moratuwa the non-exclusive right to reproduce and distribute my dissertation, in whole or in part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as articles or books).

....

Vasana Jayachamaree Wijesooriya Date

The above candidate has carried out research for the Masters Dissertation under my supervision.

....

Project Supervisor: Dr. Daya Chinthana Wimalasuriya Date

Abstract

A primary way of improving spoken dialogue technology towards incorporating more natural human-computer interaction is by making the systems user-adaptive, which indicates the ability of dialogue systems to conduct user-tailored interaction. Such an adaptation of the system to an individual user can be achieved by building a system's model of the user and exploiting this model to provide responses customized for the individual user. In this case, the system will adapt its part of the dialogue interaction according to the user's goals, plans and beliefs inferred by its user model enabling communication with the user in a manner more convenient for addressing his/ her requirements. Also, the system is required to change its user model dynamically, in order to reflect the specific characteristics of the particular user. The prototype system has been built as an attempt at exploring how dialogue systems can be made user adaptive through the above concept of building and exploiting dynamic models of users.

The system specifically demonstrates how this technology can be harnessed to cater for large-scale business domains such as insurance and banking industries where the call centers can make use of such a system to attend the typically overwhelming amounts of customer queries, saving hugely the employee time and effort.

Keywords: Spoken Dialogue Systems, Natural Language Processing, Dynamic User Modeling, User Adaptation

ACKNOWLEDGEMENT

Firstly, I would like to thank my supervisor Dr. Chinthana Wimalasuriya who had supported and guided me throughout this project to ensure its successful accomplishment. Also, I would like to express my sincere gratitude to Dr. Malaka Walpola for his constant advice and support given to us during course of the project. I am also grateful to Dr. Indika Perera and Dr. Shehan Perera for their invaluable feedback during progress evaluations of the project. Secondly, I would like to thank my colleagues at SriLanka Insurance Corporation for their untiring support extended to me especially during evaluation of the prototype. Further, I should thank my parents, although words fail to express my indebtedness to them for their love, support and encouragement given not only during this project, but throughout my life.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

DEFINITIONS, ACRONYMS AND ABBREVIATIONS

List of abbreviations

Abbreviation	Description
BN	Bayesian Network
DM	Data Management
HCI	Human Computer Interaction
IVR	Interactive Voice Response
NLG	Natural Language Generation
NLU	Natural Language Understanding
SDS	Spoken Dialogue System

List of definitions

Term	Definition
discourse context	(Dynamically changing) scope of the linguistic content and linguistic forms within a conversation(Courtesy: Wikipedia)
dialogue history	The set of preceding dialogs in a conversation
anaphora	Use of an expression of which the interpretation depends upon another (former) expression in context(Courtesy: Wikipedia)
ellipsis	Intentional omission of a word, sentence, or whole section from an utterance without altering its original meaning(Courtesy: Wikipedia)

1. INTRODUCTION

1.1.Chapter Overview

1.1.1. Purpose

Purpose of this chapter is to provide a solid foundation towards embarking on the project by discovering its significance and primary goals as well as formulating a roadmap to be followed in its progression. This also strives to develop a common understanding between the author and the supervisory bodies as to the basic scope of the project.

1.1.2. Scope

The scope of this chapter is to discuss the background, current systems related to the project as well as to explain the proposed approach for the identified problem domain. It will also discuss the main objectives of the project.

1.2.Project Background

For decades, people have been fascinated by the concept of empowering machines with the capability to emulate humans, and achieve the ideal model for human-computer interaction. As speech is the primary mode of interaction among humans, a major goal of research in this area has been to enable machine to emulate human speech, and facilitate more natural spoken interaction between humans and computers. The extent of anticipation for effortless and spontaneous human-computer interaction is depicted by the popular fictitious character HAL in '2001: A Space Odyssey' and many other science fictions envisaging machines capable of human-like speech communication. However, a considerable gap exists between fantasy and real world achievements where speech science and artificial intelligence need much improvement to realize such ambitious inventions.

Interactive Voice Response (IVR) systems are currently one of the prevalent products of speech technology which enable spoken interaction between human users and computer systems. The interaction is typically carried out using isolated word recognition [1] where the human user is restricted to provide short statements/answers and the system may provide detailed information in return. It is evident that the dialogue between user and the system is imbalanced as the system manages the bulk of conversation with only limited user input, which contrasts with human-human communication where the conversation is more or less balanced with both parties playing an active role in the communication. Although the technology has its attractions, IVR systems may not favor the naïve or casual users who are not aware of the criteria of user input expected by the system. Often, it is useful to carry out a less-constrained dialogue between such users and the system where user requirements are learned incrementally through a more balanced conversation [2].

Spoken Dialogue System (SDS) has been defined in the literature as,

> 'Computer systems with which humans interact on a turn-by-turn basis and in which spoken natural language plays an important part in the communication' [3].

The main purpose of a SDS as stated by [2] is to provide an interface between a user and a computer-based application such as a database or expert system. The inherent goal of these systems is to facilitate spoken interaction between human user and the system in a relatively natural manner. However, it is noted that interaction via these systems are carried out within constrained domains due to complexity involved in maintaining human-like conversation by the system.[2][7]

SDSs mainly differ according to the strategy adopted in dialogue management.

1. Finite-state (or graph) based systems –

   These systems maintain system-driven conversation where the user is taken through a dialogue consisting of a sequence of pre-defined steps or stages [4]. These systems typically have high success rates and are similar to IVR systems in their restricted nature[1].

2. Frame based systems –

   These systems provide less-constrained conversation than finite-state systems where the dialogue flow is not pre-defined but depends on user input and pending information the system has to elicit [2]. They typically allow user to provide a range of information in a single utterance [1], [2].

3. Agent based systems –

   These systems adopt mixed-initiative strategy where the dialogue can be both user and system driven. The conversation is hence carried out in a more human-like manner with active and balanced co-operation of both parties during communication [2].Although this approach more closely emulates human-human communication, it also tends to have high error rates due to complexity in processing more variable user input [1].

Some systems use a hybrid approach which is a combination of above techniques to alleviate the drawbacks of individual methods [1].

Adaptability of SDSs:

During human-human communication, people instinctively adapt their own views and beliefs according to views expressed by others leading to dynamic changes in the flow of interaction of each participant. Adaptability of SDSs can be seen as the ability to imitate this instinctive behavior of humans. In order to emulate human-human interaction, ideally, the system should be capable of identifying more abstract attributes of the user such as her plans, goals and beliefs, and then adapt itself to respond to the user accordingly. This mainly involves system's adaptation of its model of the user, while actively participating in the interaction [5].Dynamic adaptation of system's user model may depend on information gathered from current user interactions as well as pre-defined data regarding the user and context. The resulting adaptation of system's part of the conversation involves adapting to the states of above attributes of the particular user and thereby providing a customized service on individual basis. In fact, adaptability can be regarded as a crucial feature to be incorporated in SDSs in order to achieve their ultimate goal of human-like communication.

1.3.Existing Systems

Over the years, researchers have made important contributions through numerous attempts to improve the status of spoken dialogue technology. The Esprit SUNDIAL program [9] sponsored spoken dialogue research in several languages and participants built systems for enquiries on flight and train schedule information.

In the ARISE program [8], participants developed prototypes for train time table information systems using alternative dialogue strategies and also carried out substantial testing on several prototypes.

Many independent initiatives include, the Berkeley Restaurant Project (BeRP) [10] which provided local restaurant information, the AT&T AutoRes system[11] which allowed users to make rental car reservations over the phone, as well as the WAXHOLM system [12] which provided ferry time tables and tourist information for the Stockholm archipelago. An overview of above projects can be found in [1].

Several systems have also been deployed in domains such as call routing, stock quotes, train schedules whether information and travel planning.[1] [13]

1.4.Problem domain

Although speech technology has spread across many commercial domains with large scale industries employing speech-based interactive systems in call centers to provide prompt response to customer queries, there is still a void to be filled in technology before telephone-based customer service can be fully automated. Other than general enquiries, involvement of company staff is often required in responding to many customer enquiries where information should be provided according to individual customer needs. This may concern looking up already stored information of the customer as well as gathering his/her requirement from the on-going interaction. In fact, customers are more pleased with a service focused on their needs individually than one providing general answers in an apathetic tone. With the ever increasing customer bases, automated interactive systems to provide user specific information is hence gradually drawing demand at present.

Author finds that many SDSs have been produced through academic projects as well as commercial products. However, these systems are mainly targeted at domains where the users are typically the general public and the system has no obligation towards identifying individual users. For example, a SDS delivering train/ flight information may provide the general details to all users of the system according to their enquiries, regardless of individual user characteristics. In other words, the system's answer to a specific user query would be the same for all users. Hence, it can be observed that prevalent SDSs lack capability of providing customized information according to individual user characteristics. A theoretical framework for building an adaptive dialogue system for medical domain has been proposed by [5]. However, it has not presented details of an implemented version of the system. [6] presents a significant attempt at user modeling in a system to provide bus information for the public where the user model discriminates among skill level, knowledge level and hastiness of users. Also, as stated in [13] recent efforts in the direction of flexible dialogue systems include the development of in-car navigation, entertainment and communication systems [14] [15] [16]. Still, the context of these applications differs much from the financial and business-oriented environments where the dialogue system should handle enquiries from both existing and potential customers as efficiently and effectively as possible, and also in a manner convenient to individual customers, similar to their interaction with real company staff.

Hence, author observes that introducing 'adaptability' is one of the main concerns for current commercial dialogue systems targeted at business environments. [7] supports this view by revealing that one of the main challenges in 21st century Communication Technology Society is to equip devices with natural interaction capabilities where systems should have dynamic control of the interaction and take the user's personalized needs into consideration.

As described in a previous section, a prime technology of achieving adaptability is by incorporating the user model constructed from already stored user information as well as carrying out dynamic alterations to the model according to immediate user interactions.

1.5.A better approach

From above findings it can be concluded that introducing adaptability in SDSs through user modeling is an important advancement for SDSs which would have promising applications as call center technologies for industries with large customer bases. Author proposes to build an SDS capable of dynamically adapting its user model according to stored data pertaining to the interacting user as well as information gathered from on-going interaction. Author observes that in industries such as banking and insurance, considerable amount of information about the customer is stored in company databases which can be made use by such a system to build a rich user model and facilitate timely response to customer enquiries in an automated yet individually focused manner. Further, the initial user model constructed during first interaction could be stored and reused in latter communications with the same user or similar type of users. The system may also be refined according to the generic user characteristics found in user models. Hence, it can be envisaged that the notion of user modeling could even lead to a system with capabilities beyond that of real staff in efficiently responding with user-tailored information.

1.6.Aim

To research, design, develop and evaluate a prototype for a user-adaptive dialogue system, especially targeted at providing an automated customer-help environment, where the system makes use of a dynamically adaptive user model to respond to user enquiries with information tailored for the individual user.

1.7.Scope

The project exclusively focuses on the analysis of user modeling and their incorporation in SDSs with the aim of exploring how the concept of adaptability can be introduced to SDS by employing this technique. However, detailed research on any other factors or approaches contributing to adaptability of SDSs is not included within scope of this project.

1.8. Objectives

1. Carry out an in-depth literature survey and conduct a literature review to produce a literature review document intended at selecting the most suitable approaches for the analysis, design and development of the proposed system.
2. Design and implement a prototype for the dialogue system to meet the identified requirements.
3. Carry out a critical evaluation of the prototype to verify whether system objectives have been fulfilled.

1.9. Overview of coming chapters

Literature Review:

The chapter provides a review of the extant literature together with an evaluation of existing approaches to determine the most appropriate methodologies and techniques for the proposed system.

Methodology

The chapter presents an outline of how the prototype system is to be implemented, by identifying its primary functionalities and the techniques to be incorporated.

Design:

Chapter will discuss the derivation of system architecture including the process flow and integration of components within the prototype in order to meet the pre-identified requirements and facilitate implementation of system features.

Implementation:

The chapter will discuss in detail the work carried out during implementation phase of the system.

Evaluation:

The chapter presents a detailed description including results and conclusions of both objective and subjective evaluations carried out on the prototype system.

Conclusion:

The chapter provides a summary of overall project activities together with a discussion on achievement of aim and objectives as well as future work regarding the project.

2. LITERATURE REVIEW

2.1. Chapter Overview

2.1.1. Purpose

Purpose of this chapter is to carry out an in-depth analysis of, the techniques and processes related to user modeling in spoken dialogue systems by reviewing the extant literature in this field. The analysis is geared at evaluating the existing methods in order to determine the most suitable approach for the development of the proposed system.

2.1.2. Scope

The scope of this chapter is to provide an overview of SDSs, the importance, related concepts, and methods of user modeling for introducing adaptability to SDSs. It will further explore the prospective techniques in this regard to be employed in the context of building customer help systems for business environments.

2.2. Overview of SDS

In order to get an accurate understanding of the concept of SDS, it is important to realize that its purpose extends beyond merely acting as a speech-based interface between user and a database or expert system.

[7] identifies a speech interface as a software component that enables the user to interact with the system via speech, without generating any abstract concepts for meaning analysis. In contrast, a SDS is described as a speech based system that includes special models for representing and manipulating information at multiple levels of abstraction. A SDS involves artificial realization of a fundamental aspect of human behavior; the human dialogue, by carrying out two tasks, namely, modeling human conversational competence and simulating human conversational behavior, in addition to acting as a

tool for interacting with data, services, resources on computers [18]. Hence, it is evident that spoken natural language plays a crucial role in the functioning of a SDS.

As pointed out in [7], in order to provide a transparent and robust interface with rich interaction capabilities for the user,

> "The system should be intelligent enough to understand the user and his/her incomplete, imprecise, vague and fragmental input, but it should not aim at producing responses on the same level of inferential richness. Rather, it should restrict its responses to those that have clear communicative goals."

In order to explore how these phenomena can be achieved by a SDS, an understanding of internals of a SDS is required. Following section provides a brief overview of the typical architecture of a SDS.

2.3.Components of a SDS

A SDS is composed of several components, which collaborate as a whole to carry out the complex task of communicating with the user. Typically, each component carries out a specific subtask of the overall functionality within a pipeline architecture as shown in figure 2.1 [17].

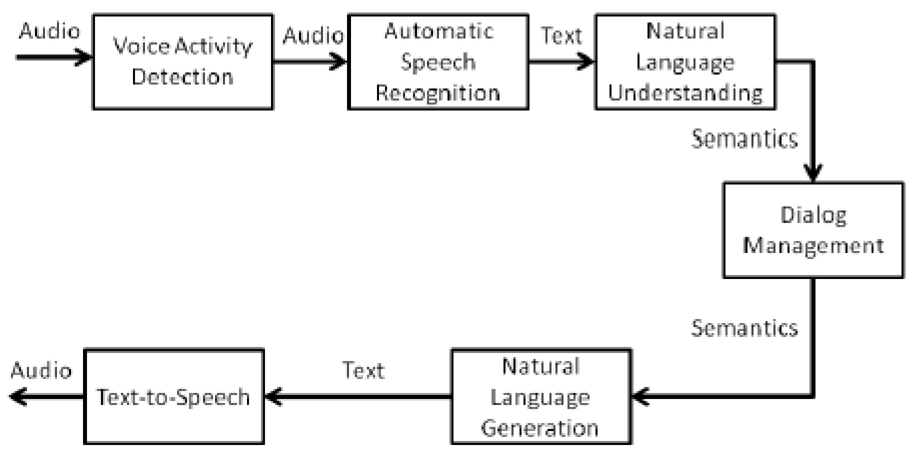


Figure 2.1: The typical pipeline architecture of a SDS [17]

Voice activity detection (VAD) involves accurately capturing user's speech input by distinguishing between speech and non-speech segments of the utterance. The task tends to get more complicated under noisy environments [17]. However, to discuss this phase in detail is not within scope of the project.

Automatic Speech Recognition (ASR) deals with the conversion of input speech utterance, consisting of a sequence of acoustic-phonetic parameters, into a string of words [2]. Specifically, it takes the speech audio data segmented by the VAD and generates the corresponding word-level transcription. ASR makes use of three models, namely, an acoustic model to describe the mapping between audio data and phonemes, a lexicon to describe the mapping between phoneme sequences and words, and a language model to describe the mapping between probable word sequences in a language and the generated words. [17]

Natural language Understanding (NLU) is identified as the analysis of the sequence of words output by the ASR to generate a semantic (meaning) representation of the recognized utterance that can be understood and used by the dialogue management component [2] [17]. For example, output of the NLU for the sequence of words "I NEED TO GO TO THE AIRPORT" would be the semantic frame {destination="THE AIRPORT"} [17].

The **Dialogue Management (DM)** typically carries out the task of taking the semantic representation of the user input produced by the NLU and generating the semantic representation of the system's response [17]. However, in a broader view, it is identified as the heart of the SDS as it is the main component that controls the system-user interaction by managing the dialogue flow while coordinating the activities of other components in the system. It also exploits pre-existing knowledge about the user and domain to facilitate the task at hand, by communicating with external applications such as database or expert systems. [2][13][17]

Natural Language Generation (NLG) module takes the semantic representation of the system's response, output by the DM, to produce the natural language expression of the response. NLG can optionally annotate the expression (surface form) with mark up tags indicating the prosodic variations within the response to be considered by the speech synthesis module. [17]

The **Text-To-Speech (TTS)** or the speech synthesis module takes the natural language output of the NLG and outputs the audio waveform corresponding to its spoken version. This can be achieved by using pre-recorded audio in the case of pre-planned system responses, or employing more flexible TTS systems to facilitate unplanned responses. [17]

2.4.Role of Dialogue management

From above constituents of a SDS, author learns that DM has a major impact on the overall functionality of the system, especially as it acts as the main decision-taker and, hence, the main body controlling the dialogue flow between system and user. As a result, DM stands as the most crucial component requiring attention when designing a SDS.

As identified in the previous section, DM typically accepts a semantic representation of the user's intention and outputs system response at a concept level. Generally, the key roles of a DM include,

1. Retrieving relevant information by connecting to an external knowledge database, based on current user input inferred by NLU as well as the discourse context[1].

2. Asking for further information and/or clarifications in order to have sufficient understanding of the user's request.

[1]Discourse context is acquired by maintaining discourse history.

3. Formulating the next system action at the concept level to output the system's utterance in NLG and TTS modules.
4. Controlling generic conversational mechanisms observed in human-human dialogs to implement more human-like dialogue behavior. [7]

Figure 2.2 illustrates the role of dialogue management with respect to the domain of a restaurant information system.

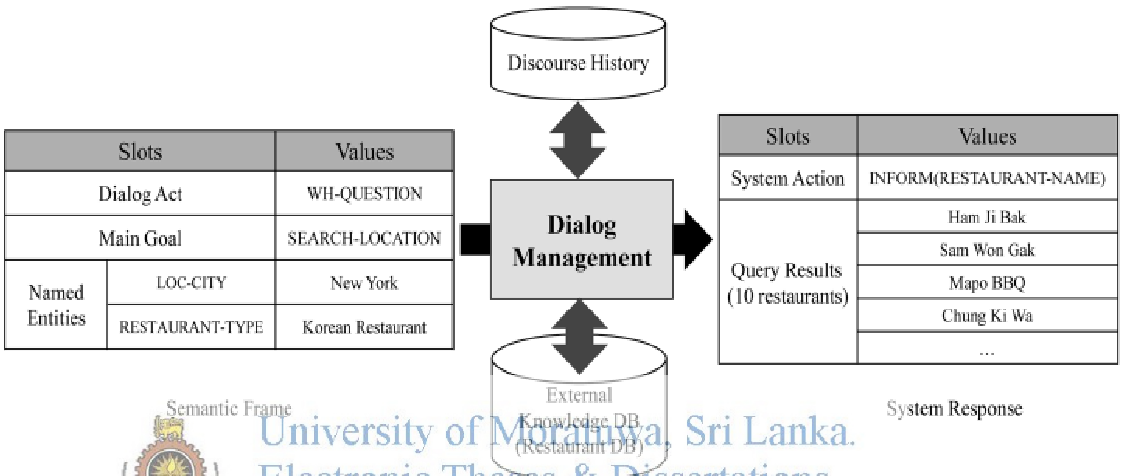


Figure 2.2: Role of dialogue management [7]

The semantic frame consisting of slots and values (left-side) is an instance of meaning representation of user input generated by NLU whereas the system response (right-side) is produced in same format. The DM makes use of discourse context which is built upon discourse history and an external knowledge database (restaurant database in this case) in order to determine correct system action.
Apart from the apparently straight-forward task of producing system response according to user intentions, DM should, in real-world, handle some major problems, one of which is the errors propagated from the preceding components in the serial architecture, namely ASR and NLU modules. For example, errors in ASR may lead to unclear or incomplete user input due to some or all of the words being incorrectly recognized. On the other hand, even in a setting with accurate ASR capabilities, NLU module may not

capture all the correct meanings due to data sparseness or ambiguity. This can cause the system to misunderstand the user and generate inappropriate responses resulting in considerable damage to the quality of system-user interaction. Although increasing accuracy and robustness of ASR and NLU modules is the intuitive solution, achieving perfect ASR and NLU functionality is impractical. Hence, introducing mechanisms within DM to counteract the above additional problems has also been an active research topic related to improving performance of SDSs. [7]

2.4.1. The importance of a user model in DM

User model is regarded in [2] as one type of knowledge among a number of knowledge sources, useful for DM, which are collectively referred to as the dialogue model[2].

It defines user model simply as a model of the user which

"may contain relatively stable information about the user that may be relevant to the dialogue – such as the user's age, gender and preferences, - as well as information that changes over the course of the dialogue, such as user's goals, beliefs and intentions".

With regard to HCI[3] (Human Computer Interaction) in general, [19] points out that systems can be made more usable and useful by providing users with experiences fitting their specific background knowledge and objectives.

It hints at the essence of user modeling by stating that,

"The challenge in an information-rich world is not only to make information available to people at any time, at any place, and in any form, but specifically to say the "right" thing at the "right" time in the "right" way".

Author realizes that achieving this kind of intelligent behavior is the ultimate objective of user modeling in dialogue systems as well.

[2]Other constituents in the dialogue model are dialogue history, task record, world knowledge model, domain model and generic model of conversational competence. [2]

[3]SDS is one particular branch of HCI technologies

[20] identifies that user models are necessary prerequisites for a system capable of exhibiting a wide range of cooperative behavior where the system should consider the user's goals and plans as well as his/her knowledge and misconceptions regarding the domain.

Further, a system's ability to draw assumptions about the user's goals, beliefs and plans is recognized as primarily important by [21], if the system is intended for casual users, especially for the general public.

Following example demonstrates the cooperative dialogue behavior anticipated from such a system [20].

User: Where can I find the nearest gas station?

System: The nearest one which is still open is located …

Here, the system considers underlying intentions of the user's question and provide additional but relevant details in order to achieve the user goal.

[20] identifies several tasks the system should carry out in order to respond in such a cooperative manner. They include,

1. Discovering the presumable plans underlying the user's question
2. Representing them in its knowledge base
3. Examining them for hidden obstacles and
4. Providing information to enable the user to overcome these obstacles.

It is observed that user model is an invaluable entity in accomplishing these complex inference processes [20].According to [21], a user model can facilitate cooperative dialogue behavior by enabling the system to carry out following activities.

1. Take into account the goals and plans for which the user needs some requested information, and supply additional relevant information, if necessary.
2. Take into account what the user probably already knows or does not know about a situation, and thus avoid redundancy and incomprehensibility in its answers and explanations.
3. Detect wrong beliefs of the user and inform the user about them.

Furthermore, [23] identifies the effect of dynamic user modeling where the user model can be modified over the course of interaction to represent the updated state of the current user with respect to his/her goals, plans and beliefs. Such a mechanism is expected to contribute to a more robust, coherent and cooperative dialogue.

Another aspect of cooperative conversational behavior as indicated in [2] is the ability to respond to user queries especially when they are underspecified or ill-formed. Such a situation can also be a result of ASR and NLU errors which challenge the DM functionality as previously mentioned. A user model, in this case, can be a crucial source of information for interpreting user's utterance where it will assist in inferring the plan underlying user's query and responding in terms of the plan, instead of depending only on the user input as it is.

Apart from establishing cooperativeness in dialogue behavior, a user model is also considered an important basis for intelligent dialogue behavior in general. That is, even in less sophisticated dialogue systems, a user model will be required for identifying the objects the user is referring to, analyzing the meaning of implicit or ambiguous user utterances and also for determining the effect a planned dialogue contribution will have on the user [20].

[22] complies with this view by identifying that a user model can serve as a key factor in achieving the following quality attributes within a SDS.

1. Effectiveness – Ensuring that the system reaches the correct decision
2. Efficiency – Reaching the correct decision in an economical way
3. Acceptability – Supporting/expressing the system's decision making in a comprehensible and perhaps also in an agreeable way.

Achieving these qualitative features as well as cooperativeness in SDSs via user modeling mainly requires adaptation of the user model to individual users.

[24] discusses three fundamental properties of a user model within a system aimed at adapting its behavior according to individual users as follows.

1. *individual* in the sense that the system is able to construct different models for each user

2. *dynamic* which can change during the interaction

3. *explicit* which is separable from the rest of the system's knowledge.

Further, it points out the three main ways a user model can assist in adaptation as illustrated in figure 2.3.

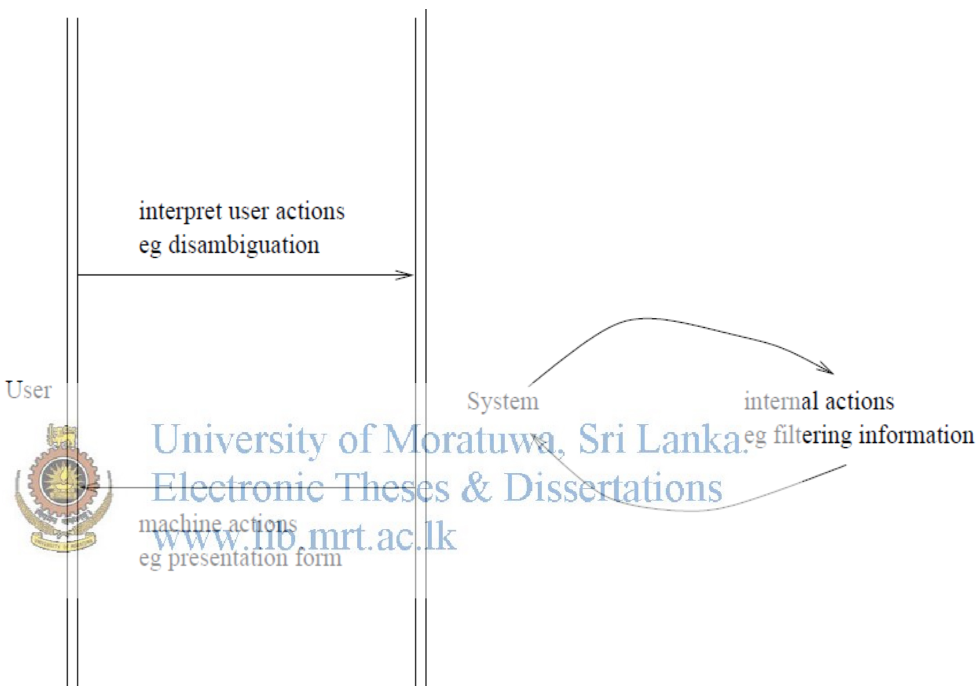


Figure 2.3: Role of the user model in adaptation [24]

As indicated by the upper horizontal line, a user model can assist the system in interpreting user actions[4], especially when the user input is found to be ambiguous or incorrect. As the lower horizontal line indicates, the system's response can also be controlled by a user model to improve the quality of interaction. This may involve adapting the content as well as the presentation form of the response according to user's level of knowledge, competence, use of language etc.

[4] User actions, in the context of SDS, refer to user's speech input to the system.

Apart from directly supporting the system-user interaction in this manner, user model can also facilitate the internal actions of the system as depicted on the right-side of the diagram, especially when it requires filtering information obtained from other sources, on behalf of the individual user. [24]

Therefore dynamic user modeling can be regarded as a key driver of user-adapted interaction in SDSs aimed at providing an improved service to the users. Introduction of adaptability in this manner is also viewed as an important contribution to making man-machine interaction more similar to human communication [21] as it would reflect the inherent capability of humans to adapt to their dialogue partners.

Following section will explore the fundamental aspects that need be considered in the process of user modeling in dialogue systems.

2.5.Overview of user modeling in dialogue systems

[20] distinguishes a user model from the user modeling component in natural language dialogue systems as follows.

> "A *user model* is a knowledge source in a natural-language dialogue system which contains explicit assumptions on all aspects of the user that may be relevant to the dialogue behavior of the system."

On the other hand, the function of the *user modeling component* of a dialogue system is,

> "to incrementally construct a user model; to store, update and delete entries; to maintain the consistency of the model; and to supply other components of the system with assumptions about the user."

According to [21], these assumptions about the user, that a system with a user modeling mechanism should be able to generate, particularly include,

1. The user's goals
2. The plans with which the user wants to reach his/her goals
3. The knowledge (beliefs) of the user about the particular domain.

[25] identifies that a user model[5] developed for this purpose exhibits 4 properties as follows.

1. Separate knowledge base

   Information about the user is contained in a separate module rather than distributed throughout the system

2. Explicit representation

   Information is encoded in a sufficiently expressive representation language facilitating inferences to be made particularly from implicit knowledge about the user.

3. Support for abstraction

   It may describe abstract as well as concrete entities such as classes of users as well as individuals. This will allow individual user models to inherit the more general properties attributed to that particular type of users.

4. Multiple use

   It can be used for several different tasks as it is explicitly represented as a separate module and not highly correlated with the task at hand. This requires representing knowledge in a more general way that does not favor one task over another.

2.5.1. Creation of user models

User model creation mainly involves,

1. acquiring assumptions about the user from a number of knowledge sources and
2. representing these assumptions using an appropriate representation scheme.

2.5.1.1. Knowledge acquisition in a user model

As revealed by [20] and [21], several knowledge sources have been identified for acquiring assumptions about a user including his/her goals, plans and beliefs as follows.

[5]The authors have used the term ‘agent model’ instead of ‘user model’ and regarded the user model as a specialization of agent model.

Default assumptions:

Typically, a system operating in a particular domain can attribute a set of general goals and beliefs to any user of the system, as long as no evidence is found to suggest contradicting assumptions. The system can use these assumptions to construct an initial stereotypical user model at the very beginning of the system-user interaction.

Such default assumptions about users can be based on,

1. The user's general knowledge

   Ex: In a customer-care system that enables customers of an insurance company to enquire about their insurance claims, assumptions can be made about the user's knowledge of insurance terminology to some extent (knowledge of terms such as claim number, policy number, premium etc.).

2. The user's beliefs about a domain

   Ex: Above system may also assume that all users have some basic understanding of the procedure followed for insurance claims by the company[6].

3. The user's goals

   Ex: All users of the above system want to know the progress of their respective claim settlements by the insurance company.

Initial individual user model:

Apart from stereotypical assumptions, a user model constructed during interaction with a user can be stored and retrieved later, to be used as an information source for subsequent interactions with the same user. Such a mechanism can enable the system to have more user-specific assumptions at the start of the dialogue leading to more efficient adaptation to the user. [26] presents one of the popular examples found in the literature for employing initial individual user models in dialogue systems.

However, maintaining such an initial user model can introduce unnecessary overhead especially when there is quite a low probability for a user to consult the same system more than once.

[6] However, in practice, this assumption can mostly be valid for frequent customers or users of the system

Assumptions based on the user's input:

User's input into the system is regarded as the most direct and reliable source of information for deriving assumptions about the user. Transforming user's input into assumptions can vary from being a straightforward task to one that is very difficult. As a result, reliability of inferred assumptions may also vary depending on the complexity of the inference task. According to [25], a system may encounter four types of situations in which goals and plans of the user should be inferred from user input, as listed below in the order of increasing complexity.

1. The user directly states a goal

   Ex: "How do I go to Colombo from here?"

   In this case, the system can directly recognize the user's goal without further inference.

2. The user's goal may be indirectly inferred from the user's utterances

   Ex: "Can you tell me what time it is?"

   Here, the system must infer that the goal behind user's question is to know what the current time is. It is observed that people often make such implicit statements expecting the goals to be indirectly inferred from their utterances.

3. The user has incorrect or incomplete goals and plans

   Ex: "When will the next train to Kandy arrive today?"

   Above question will be invalid in a situation where no train is scheduled for Kandy. System should identify this incorrect goal underlying user's input.

4. The user has multiple goals and plans

   Ex: "Can you tell me what time the next train to the airport departs?"

   [25] identifies that two goals are associated with the above utterance. One is the immediate or communicative goal of the utterance which is, in this case, to find out when the next train leaves. Another is the task goal which is to board the train, as inferred by the utterance.

Apart from the first type, all other situations normally require complex inference mechanisms such as,

1. structure-based (syntactic) inferences
2. content-dependent inferences and
3. Assumptions drawn from linguistic clue words, as identified by [21].

Implicit acquisition of user models in cooperative dialogue systems is also explored by [27] where a number of model acquisition rules falling roughly into three categories (communication, model-based, and human-behavior rules) have been proposed. As indicated by [20], these general inference rules can also be employed in acquiring assumptions from the user input.

Assumptions based on dialogue contributions made by the system:

The dialogue contributions of the system, which are made in the form of answers, questions and commands can also act as an information source to the user model. For instance, after answering a user's question, the system can assume that the user is now aware of the answer, and hence include the assumption in its user model. This is observed to be particularly useful in the case of planning follow-up information and explanation for the user. [20] It will also enable a more succinct dialogue preventing the system from delivering redundant information to the user.

2.5.1.2. Knowledge representation in a user model

A user model needs to formally represent the assumptions about the user, acquired from above sources, in a manner that can facilitate the system-user interaction. A number of mechanisms for representing user models have been introduced in the literature which mainly varies by two factors; complexity of the technique and inference capacity of the resulting user model. As [20] reveals, selection of the appropriate representation technique may depend on the application purpose as well as the degree of expressiveness expected from the system. Following is an overview of some of the techniques as identified by [20].

Use of linear parameters:

This approach is exemplified by [26] which introduces a system (named GRUNDY) that models personality traits of individual users in order to recommend books according to their specific tastes. The system makes use of two linear parameters to represent each personality trait in the user model; one indicating the assumed degree to which the personality trait is attributed to the user and the other indicating the system's level of confidence about the first assumption [21], as illustrated by figure 2.4.

Personality trait	Value	Certainty of assumption
Education	5	900
Seriousness	5	800
Piety	-3	423
Tolerate-sex	5	700
Tolerate-violence	-5	597
Tolerate-suffering	-5	597

Figure 2.4: Representation in the GRUNDY system [21]

Overlay technique:

[28] employs this approach in building a user-tailored front end system for the explanation component of expert systems. The system adjusts the explanation produced by expert system to suit the expertise level of current user, by using only those concepts the user is familiar with. Here, the overlay approach is used to represent user's familiarity with the concepts, where a parameter associated with each concept indicates whether the concept is known/ not-known by the user according to system assumptions, or, whether no information is available in that respect. Another parameter is used to indicate the level of certainty the system has on user's familiarity with each concept. Shown in figure 2.5 is an example of an overlay model. [21]

Concept hierarchy of the system	User model	
	user's knowledge state	certainty of assumption
INFECTIOUS-PROCESS	KNOWN	100
HEAD INFECTION	KNOWN	100
SUBOURAL-INFECTION	NOT-KNOWN	100
OTITIS-MEOIA	NO-INFORMATION	100
SINUSITIS	NO-INFORMATION	100
MENINGITIS	KNOWN	100
BACTERIAL-MENINGITIS	KNOWN	30
MYCOBACTERIUM-TB	NOT-KNOWN	70
VIRUS	NOT-KNOWN	90
FUNGAL-MENINGITIS	NOT-KNOWN	100
MYCOTIC-INFECTION	NOT-KNOWN	100
ENCEPHALITIS	NOT-KNOWN	90
SKIN-INFECTION	KNOWN	100

Figure 2.5: An example of an Overlay model [20]

Partition approach:

A mechanism to represent user's goals and beliefs, with a slightly more expressive power than above methods, is the partition approach. Here, separate partitions are maintained to store system assumptions about each aspect of the user such as user's beliefs about the domain, user's goals and plans as well as user's beliefs about the system's goals and beliefs etc. Each goal or belief contained in a partition is represented using standard concept-based knowledge representation schemes such as predicate calculus or network-like representations. Figure 2.6 is an example of the partition approach which maintains separate partitions to store system's assumptions on user's beliefs and goals. [21]

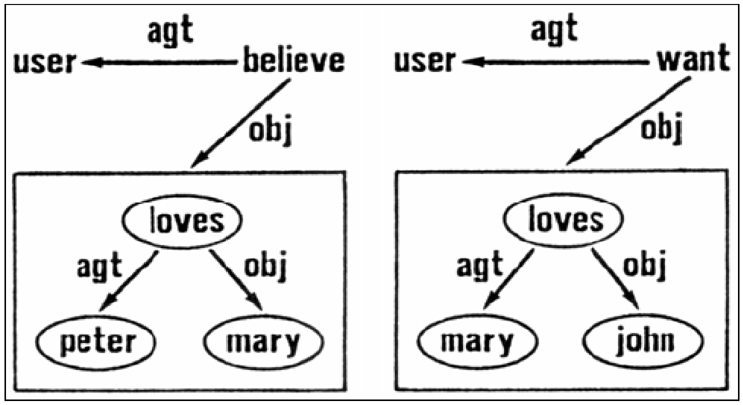


Figure 2.6: A partition representation example

(A partition representation of 'The user believes that Peter loves Mary (user's belief) and wants Mary to love John (user's goal)' [21].)

2.5.2. Approaches for plan recognition in user modeling

Plan recognition is identified as an important part of user modeling which involves inferring about user's goals and plans from his/her intentions, preferences and other observable information. The mechanisms employed for the purpose vary from traditional plan library-based reasoning to stereotype-based techniques.

2.5.2.1. Traditional plan-recognition approach:

The typical method followed in the early user modeling systems is to use hand-crafted plan libraries to make inferences based on observed information about users. As identified by [29], the inference process starts with a set of goals that the user is expected to pursue and an observed action by the user. The user's goal needs to be inferred by determining how the observed action contributes to that goal. To accomplish this task, the system makes use of a plan library which provides a set of actions that the user might execute and a set of *recipes* that specify the possible ways of how a user might perform these actions. Accordingly, each recipe includes preconditions, sub goals, effects and goals pertaining to an action. As pointed out by [30], building these plan libraries involves carefully analyzing several instances of the problem and representing the various alternative paths pertaining to the problem that may be encountered and followed during system-user interaction. Although the approach is still widely used, construction of plan libraries in this manner suffers from two shortcomings, collectively known as the *knowledge bottleneck problem*.

They include,

1. Resource-intensiveness of the process of acquiring plan libraries
2. Lack of extendibility or adaptability of plan libraries.

Also, a high amount of uncertainty is involved in user modeling especially when predictions have to be made from implicit user actions or sparse observations about users. Employing plan libraries becomes further problematic as inferring explicit goals and plans is not quite appropriate in this case. [30] proposes predictive statistical models as a promising alternative to alleviate these problems found in the traditional method. The statistical approaches have particularly been useful for inferring based on stereotypes as illustrated in the next section.

2.5.2.2. Stereotype-based approach:

User modeling via stereotypes is a simple but powerful idea first proposed by Rich[26]. The technique is based on the instinctive human reasoning to deduce a large number of assumptions about people from a small amount of information. For example, if someone is a judge, people might hold many assumptions about the person such as that he or she is probably - over forty, well-educated, fairly affluent, honest and well-respected in the community. Although any particular judge may not hold all of these attributes, many of them would have most of these stereotypical characteristics and one tends to assume these qualities of a judge until shown otherwise[26]. Hence, as [24] points out, inference based on stereotypes is intended to be statistically valid. A user-adaptive system can employ this approach to build its initial model of an individual user by establishing a set of default beliefs about the user at a quite early stage in the system-user interaction. [26] presents a well-known example of this technique where the system stereotypes users according to their personality traits (observed from implicit and explicit user input) and use the stereotypically inferred attributes of each user to recommend novels he or she would probably enjoy. Another example is the KNOME system [31], which used stereotypes (such as novice, beginner, intermediate, and expert), to infer the user’s level of expertise in Unix. As [26] states, wide acceptability of this method is due to the fact that ‘although all situations (or people) are not identical, they are all not completely dissimilar either’[7].

This statistical validity of the method suggests that most aspects assumed by the stereotype may be correctly associated with many, if not all, of the users identified by that stereotype. A model of the user derived in such a manner may be refined when more reliable information becomes available about the individual assumptions, perhaps reflecting that a user has many but not all of the aspects from the initial stereotypical inference. Typically, most of the default assumptions regarding the user continue to hold, unless information is acquired later on to indicate that the initial classification was incorrect. [24]

In other words, stereotype-based inference provides a system with a good starting level adaptation to the user, expediting the process of acquiring a more sophisticated model of the user [24].

[24] distinguishes this technique from other knowledge-based reasoning and identifies essential elements of a stereotype as follows.

- triggers – which activate the stereotype

  For example in the KNOME system [31], if the user is identified as an expert in Unix, system will activate the expert-stereotype.

- inferences – stereotype-based predictions about various characteristics which are incorporated into the model of the user [26].

  Ex: Above expert-stereotype would contain many aspects of Unix that an expert user is assumed to know.

[7]Also, a computer system has the potential to be more productive with the use of stereotypes than humans especially due to its emotional immunity towards stereotypes as indicated by [26].

- retraction facility – to deactivate a stereotype when its essential triggers are violated

  Ex: When a stereotype for a programmer has been activated and it is subsequently determined that the user cannot program [24].

Due to the highly statistical nature involved, use of statistical methods is identified as more effective for the construction of stereotypes, compared to the hand-craft approach. With the advances in machine learning and large amounts of electronically available data from various domains, [30] proposes predictive statistical models as a promising alternative to the traditional approach where it can particularly address the knowledge bottle-neck problem and uncertainty associated with user modeling.

Stereotypical inference is facilitated by two main statistical approaches namely, content-based and collaborative, as discussed below.

2.5.2.3. Content-based approach

This method is used when a user's past behavior can be reliably used to predict his/ her future behavior. In other words, it assumes that a user behaves the same way, given a set of circumstances [32]. Hence, this approach is recommended for systems where users tend to exhibit idiosyncratic behavior. Here, data from the user's past behavior is employed to build a predictive model for the user and hence, the system requires collecting large amounts of data from each user to enable the creation of a statistical model. Also, the predictive model, in this case, is built for a specific situation which will not be useful even for the same user, in a different setting (for instance, when the user wants a service different from his/ her past requests). Hence, it is identified that the usefulness of the predictive model depends on the features selected when implementing the approach where features that are too specific will be useful only for repetitive behaviors. [30]

2.5.2.4. Collaborative approach

This method is used when a user can be expected to behave in a similar way to other users. Here, the predictive model for an individual user is built using data from a group of users. This approach is known to be particularly useful when making predictions about a new user or about a known user in a new situation. Hence, it stands out as more advantageous than the content-based approach for catering to both new as well as existing users under various circumstances while reducing the data collection burden for individual users at the same time. However, since the model for a single user is built from observations of many users, the system may not be able to address specific requirements of a user as the predictive model is likely to represent an 'average' user rather than the individual user. To alleviate this problem, it is suggested that the approach should be extended to model the characteristics of groups of users (also known as clusters [32]), in order to make accurate predictions about individual users by matching them to a particular group. [30]

[30] identifies several statistical models that have been used to implement both the content-based and collaborative approach as follows

1. Linear models
2. TFIDF (Term Frequency Inverse Document Frequency)-based models
3. Markov models
4. Neural Networks
5. Classification
6. Rule-induction methods
7. Bayesian Networks

Of the above models, Bayesian Networks is found to support the combination of collaborative and content-based approach. Here, the collaborative approach is used to obtain the conditional probability tables and the initial beliefs of a Bayesian Network whereas the content-based approach is used to update these beliefs when the network is accessed by a user. Hence, it enables the predictive model to overcome the data collection problem in the content-based approach as well as to tailor the collaboratively learned aspects of the model to a single user [30].

2.6.Employment of user modeling for building interactive customer-care systems

[18] identifies three types of task-oriented spoken dialogue systems as follows.

1. Informational
2. Transactional
3. Problem solving

The customer-care systems belong to the informational category out of above types. They enable customers to obtain various kinds of information about their transactions etc. with the particular company. However, as soon as a customer expresses a sophisticated information need, the company staff often has to take over the interaction with the customer to provide the specific information to address his/ her individual requirement. With the ever-increasing customer bases in large-scale business environments, providing such customized information through automated systems has become an important concern in this respect.

Employing an appropriate user modeling mechanism in the construction of such systems would be a promising approach to make them adaptive to individual customers and thereby provide information tailored for their specific needs.

Deciding on the appropriate user modeling approach, in this context, involves identifying the characteristics and behavioral patterns of the above user group with respect to their interaction with the system.

A main characteristic of users within domain specific systems is that their goals of interacting with the system are constrained by the particular application domain. Hence, main user requirements will typically fall into few types of categories which may be easily pre-identified. For example, typical queries from the customers of a bank may include enquiries related to bank balance, taking loans, opening various types of accounts etc which can be intuitively determined in advance.

Also, two main interaction patterns can be observed from users of such a system.

1. Goals of many customer interactions are likely to correspond with one or more of the user requirements that can be pre-identified according to system domain. Hence there is high probability of a user to request one of the services that has already been sought by many other users of the system.

2. There is quite a low probability for a single customer to interact with the system regularly or even more than few times. Hence, it would be more convenient to model the requirements of a user in terms of the types of services requested by them in general, rather than focusing on behavioral patterns of individual customers.

Availability of static information:

Apart from above properties with respect to the nature of system-user interaction, a wealth of static information about the user is often available from the company databases etc, especially for existing customers.

According to the above identified aspects, following user modeling techniques can be suggested as appropriate for this context.

1. Use of stereotype-based user modeling approach

The queries (requesting various informational services) made by users in the past can be used to infer the type of service requested by a particular user from his/ her initial dialogue with the system. The user can hence be stereotyped based on the service requested, enabling his/ her specific plans and goals to be inferred accordingly. Here, employing the collaborative method will be more advantageous where the requirements of an individual user are inferred by referring to the interactions of all users with similar informational needs.

2. Use of an initial user model built from static information available for the user

In most cases, information pertaining to the particular user is already available from databases and such information can be used to build an initial model of the user prior to his/ her interaction with the system. It can act as a valuable source for inferring characteristics of a user (such as knowledge level, beliefs etc.) which are difficult to observe from user input alone.

2.7.Review of existing dialogue system frameworks

A review of existing dialogue system frameworks has been carried out in order to observe the viability of employing already available tools and resources in building the prospective dialogue system. Listed below are some of the state-of-art frameworks introduced by several researchers in the field.

Olympus/Ravenclaw Framework:

Olympus is a complete framework introduced by Carnegie Melon University to facilitate dialogue system development. Its main objective is to provide a sophisticated platform with which dialogue systems researchers can implement and test their ideas, alleviating the need to build complete systems on their own. The Ravenclaw dialogue manager employed by Olympus makes a significant contribution to the framework's architecture where it enforces a clear separation between the domain-specific aspects of the dialogue control logic and the domain-independent characteristics of the dialogue. This allows system developers to focus exclusively on specifying the dialogue task control logic while a rich set of domain-independent conversational skills such as error handling, timing and turn taking are automatically supported and enforced by the Ravenclaw dialogue engine. Olympus also incorporates other dialogue system functionalities such as speech recognition, understanding, generation and synthesis by integrating the relevant modules which have been produced through various research projects. The approach has led to rapid development of mixed-initiative task-oriented dialogue systems where a number of systems operating under multiple domains and interaction styles have been developed using the Olympus framework with several example systems included in the Olympus/Ravenclaw distribution.

CSLU toolkit:

The CSLU toolkit has been introduced by John-Paul Hosom (main developer) at the Center for Spoken language understanding (CSLU) OGI Campus, Oregon Health & Science University. The platform incorporates core dialogue components including speech recognition, natural language understanding, speech synthesis as well as facial animation technologies providing a comprehensive and flexible framework for research and development of spoken-language systems. It has particularly enabled developers having no specialist knowledge of the component technologies (such as speech recognition and natural language processing) to construct, explore and use spoken dialogue systems with relative ease. This is facilitated by the graphically-based authoring environment through which a dialogue system can be constructed by selecting and linking the graphical dialogue objects (each representing a particular function in the dialogue process) into a finite state dialogue model.

OpenDial framework:

Opendial framework introduced by Pierre Lison at University of Oslo is a recent contribution (made available since April 2014) to the field of dialogue system research. The framework has been developed as part of his PhD research [33] on structured probabilistic modeling for dialogue management. Pertaining to the main research goal, the framework demonstrates how robust and adaptive dialogue management can be achieved through a hybrid approach combining both logical and statistical methods of dialogue modeling within a single framework. The framework introduces the notion of probabilistic rules which have been used to encode the primary internal models of the dialogue domain pertaining to NLU, DM and NLU modules. The system employs blackboard architecture and allows interaction of system components through the dialogue state (referred to as the central information hub) to which the system modules are connected. The blackboard architecture of the system specifically allows integration of new modules as well as removal of existing components without disrupting rest of the system functionalities.

Author has experimented with two of the above systems namely the Olympus and OpenDial frameworks. With respect to achieving project's goal i.e. incorporating a user model into the conventional dialogue system, OpenDial framework posed as more advantageous over the Olympus system especially due to its architectural provisions for integrating external modules. Also, setting up of Olympus framework required a number of additional resources in order to gain proper functionality of the system whereas OpenDial was comparably a lightweight program requiring only few additional resources.

3. METHODOLOGY

3.1.Chapter Overview

Purpose of this chapter is to present an outline of how the proposed system is to be implemented, by identifying its primary functionalities and the techniques to be incorporated. The first section highlights the prospective system flow and selection of domain for prototype system while the latter discusses on relevant techniques for achieving the said functionality according to findings from the previous chapter.

3.2.Proposed system

In order to introduce user adaptability to spoken dialogue systems, author proposes to integrate a user modeling component as one of the knowledge sources for DM module as depicted by figure 3.1.

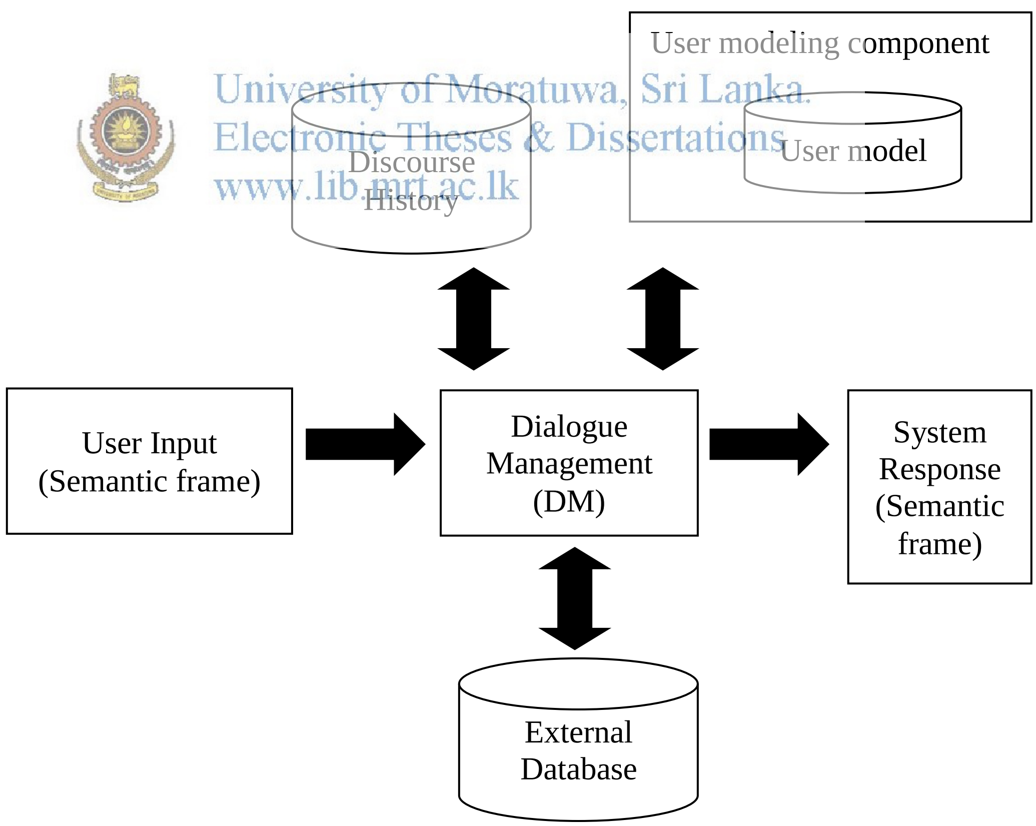


Figure 3.1: Integration of user modeling component in DM

Apart from the discourse history and external database sources employed by DM in the traditional approach, it particularly makes use of the user modeling component in order to determine the correct system response in this case. Input provided by the DM to the user modeling component will be, the stored information pertaining to the user (retrieved from database) as well as the information based on user input which may particularly include user's plans (or short-term goals) explicitly communicated by the user.

The user modeling component will facilitate its user model to infer the user's ultimate (long-term) goals as well as his/ her beliefs and knowledge with respect to the application domain, by using the above static and dynamic information fed by the DM. The assumptions about the user inferred by the user model will be provided back to the DM to facilitate the formulation of user-tailored system response.

Application domain:

Implementation of the prototype system will focus on the domain of providing user-tailored automated customer-care service by an insurance company for its customers.

Following example illustrates the inference of user's long-term goal from his/ her short-term goal (plan) applicable in above domain.

User utterance : What are the available insurance plans for a retired person?

Short-term goal : need information on retirement insurance plans

Stereotype : New insurance plan taker

Long-term goal : Wants to buy an insurance plan (Stereotypically inferred)

3.3.Proposed techniques

Stereotype-based approach will be used by the user modeling component as it is identified in section 2.4 as the most suitable technique to be applied in this domain. Hence, the user modeling component will contain the following functions.

1. Update user model with static information

- The stored user information may be provided by the DM to this unit at the beginning of the interaction with the user. Hence, initial user model can be created using this static information.

2. Trigger stereotype – Assign the statistically inferred stereotype to user model (see section 2.3.2.2)
3. Update stereotype – This involves updating the strength (certainty of correctness) of the inferred stereotype, based on dynamic information obtained during course of the interaction.
4. Retraction facility - To remove/ reassign stereotype (see section 2.3.2.2)
5. Supply DM with assumptions about the user

Accordingly, user model will contain the following information.

1. Static user information
2. Current stereotype of the user
3. Stereotypically inferred assumptions on goals, beliefs and knowledge of the user.

Statistical approach to infer the stereotype for user:

Collaborative method will be used to infer the relevant stereotype as recommended in section 2.4 for this domain.

4. DESIGN

4.1. Chapter Overview

Purpose of this chapter is to derive a detailed and unambiguous design description of the system which will serve to minimize risks and uncertainties encountered at implementation phase. Hence, the chapter first identifies high-level overview of the system including the system context as well as the basic flow of communication between its components. It will then present the design description of each system component in detail and finally discuss on integration of the user modeling component to the conventional dialogue system.

4.2. System Overview

4.2.1. Example use case chosen for the prototype system

As indicated in section 3.2, the prototype system will be targeted at providing an automated customer care environment for customers of a typical insurance company. The scope was narrowed down to address the use case where a customer is interested in obtaining information about insurance policies, possibly with the intention of taking a new policy for the user herself or her familiars. Accordingly, it was identified that the customer enquiry may be directed at one of several goals as shown in figure 4.1.

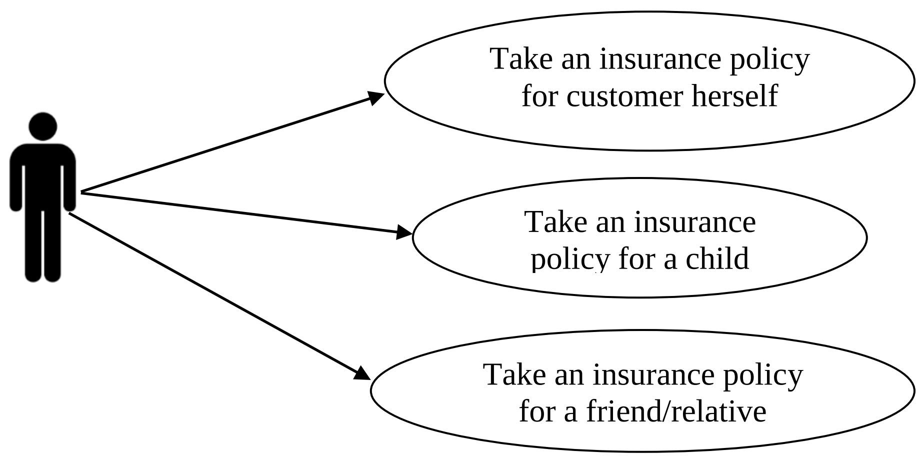


Figure 4.1: Use case chosen for prototype system

Here, the system needs to infer user (customer) goal from input obtained through its dialogue interaction with the user. System should thereby provide necessary information to the user in order to achieve above identified goal in a user tailored manner. In determining information necessary for the user, system (user modeling component in particular) will take into account the unique characteristics (or conditions) of the user which may involve referring to any stored user information and predicting the optimum solution to address his/her need.

4.2.2. System Constraints

As integration of user modeling component in a dialogue system is the main focus of the project, prototype system will constitute of only the core internal components of a dialogue system namely, NLU, DM and NLG modules which would provide an adequate setting to accomplish the task at hand. However, it is acknowledged that a complete SDS would necessarily consist of speech recognition and text-to-speech synthesis, in addition to above modules.

4.2.3. High-level overview of system functionality

System concerns introduction of a dialogue system capable of providing user-specific system responses, and the development will mainly focus on integration of a user modeling component to a dialogue system in order to accomplish the task. Figure 4.2 presents the basic, high-level functionality of the expected system.

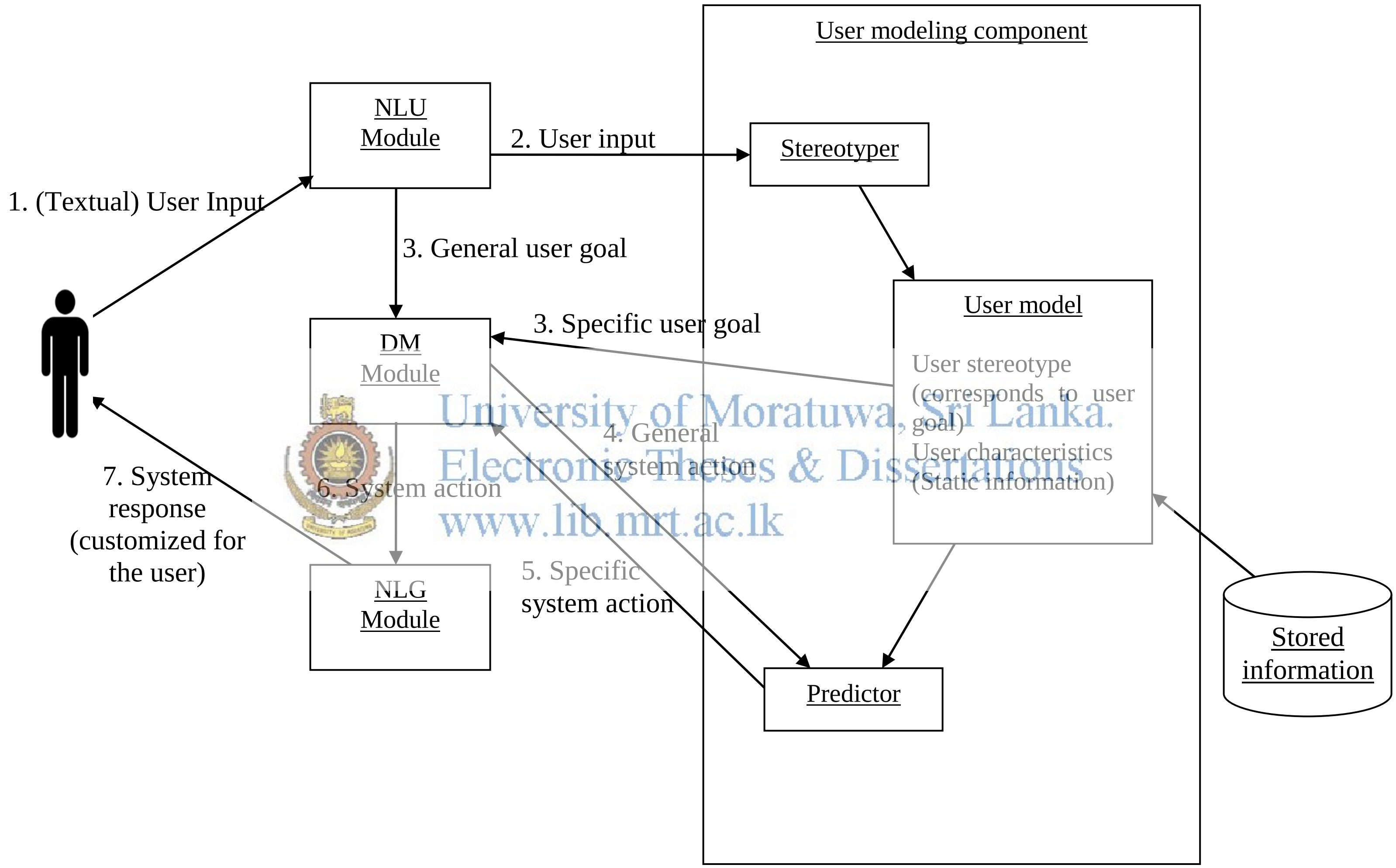


Figure 4.1: System Overview

The user input (in textual form) is fed to the NLU module which determines user action inferred by the input. Here, the user input is also passed on to the user modeling component where it is processed by the stereotyper sub-module to identify the user stereotype. Subsequent user inputs may also be used to refine/verify the initial stereotype which is then stored in the user model. User stereotype will correspond to the specific user action or user goal determined using collaborative method[8]. Apart from stereotype, user model may store any static information about the user available from databases etc. While the NLU module delivers the general user goal to DM module, user modeling component will provide more specific goal intended by the user to DM module.

DM module then chooses the system action appropriate for the user goal in hand. However it also consults the user modeling component by providing it with the system action to be performed in general. The Predictor sub-module within user modeling component considers above system action together with the specific user characteristics in the user model to formulate a more refined user-specific system action which is offered back to DM module.

System action thus obtained is passed from DM to NLG module to produce the natural language representation of the system action which is then presented to the user as system response. The system response is hence user-tailored to address both explicit and implicit goals of the user inferred by direct user input as well as the specific user characteristics.

Deviating slightly from the approach suggested in Methodology chapter, user modeling component communicates with NLU module (in addition to DM module) in order to assist in identifying the specific user goal from direct user input. This design decision was regarded necessary for intended functionality of the prototype system.

[8]Refer section 2.3.2.4 for a description of the method

4.3.Detailed System Design

4.3.1. Design of the basic dialogue system

The basic dialogue system (to which the user modeling component will be integrated) will be constructed as the first step in prototype development. It may include general capabilities to handle user enquiries relating to use case identified in section 4.2.1. Blackboard architecture, a widely used technique in dialogue system development, will be employed as it provides a flexible structure to the system with less complexity in addition and communication between system components. Here, the blackboard refers to the system (dialogue) state, while the modules such as NLU, DM and NLG are attached to the blackboard.

In event-driven blackboard architecture, key mode of communication between components is via change in the dialogue state. A change in dialogue state (an event) is notified to all system modules and the appropriate module(s) will choose to execute relevant actions, further modifying the dialogue state. The updated state will in turn cause the next affected module(s) to act upon it and so on, until the dialogue state is stabilized (with no further components to be activated). Component execution will typically start from NLU module (albeit SR module in a SDS) and continue to DM and NLG modules (may further extend to Speech synthesis module in a SDS) respectively. This mechanism particularly contributes to module independence which allows maintaining a flexible workflow within the dialogue system process as well as convenience in addition/ removal of modules to the system.

In accordance with these observations, the basic dialogue system will be implemented on top of the OpenDial framework[9] which facilitate building dialogue systems conforming to the above event-driven blackboard architecture.

[9]The framework (OpenDial toolkit) has been introduced by Pierre Lison in 2014 as part of his PhD thesis on structured probabilistic modeling for dialogue management ([33]).

OpenDial framework

In the openDial framework, main modules within the dialogue system are each composed of a rule-structured model which is primarily, a set of (probabilistic) rules to be executed upon activating the particular module and its trigger condition (changes in dialogue state which instantiate above set of rules).

The dialogue state is maintained through a set of state variables which are updated by system modules as appropriate. State variable(s) triggering the execution of a module is/are called its trigger variable(s). Update of any of the trigger variables of a module within dialogue state will cause the set of module-specific rules to be executed leading to another update of certain state variables within dialogue state (any of which may act as trigger variable(s) to another module).

OpenDial framework introduces 4 main state variables maintained within dialogue state as shown in table 4.1.

Variable name	Symbol (used byOpenDial)	Updated by	Triggered module	Description
User utterance	u_u	User input (captured through user interface)	NLU	stores latest user input to the system
User action	a_u	NLU module	DM	stores the user action (user goal) inferred by user utterance.
System action	a_m	DM module	NLG	stores the system action to be performed to cater to user goal stored in user action variable.
System utterance	u_m	NLG module	None (output as system response through user interface)	stores the system output to the user according to the state of system action variable.

Table 4.1: State variables in OpenDial framework

Accordingly, figure 4.3 illustrates how the main modules corporate to change the dialogue state through the workflow of basic dialogue system, taking for example, an instance of the use case selected for the prototype system.

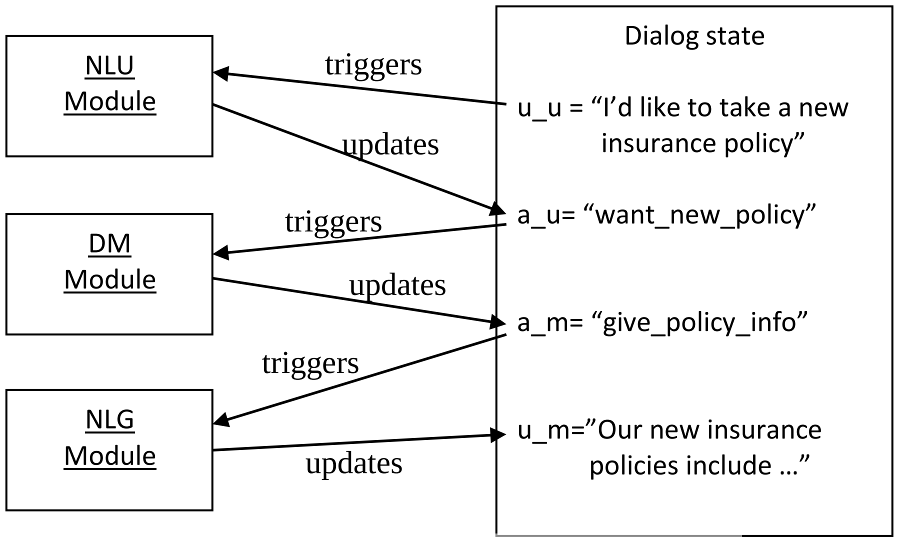


Figure 4.2: Workflow of basic dialog system

In addition to above modules which contribute to core functionality of the system, OpenDial also facilitates integration of external components, including speech recognition and text-to-speech modules, to enable construction of comprehensive spoken dialogue systems. The task is made simpler by blackboard architecture of the system where external modules can be easily plugged in and out of the system without disrupting rest of the system process. Integration is done by making available the dialogue state, to be monitored and updated as appropriate by external modules, in a similar way to above main modules.

4.3.2. Design of the user modeling component

User modeling component will primarily consist of the user model together with two other utility components namely the Stereotyper and Predictor sub-modules.

Design of the user model

As shown in figure 4.2.3, user model contains the dynamically generated user stereotype as well as stored information about the user retrieved from database. Considering the use case selected, the user model for a particular user will hold one of the following stereotypes.

1. Want_Policy_For_Self – User is in want of taking an insurance policy for himself/herself.
2. Want_Policy_For_Child – User is in want of taking a policy on behalf of his/her child.
3. Want_Policy_For_Other – User is interested in helping a friend/relative to take a policy.
4. Want_General_Info–User simply wants information about insurance policies

Fourth stereotype is in fact a generalization of the former stereotypes and the user model resort to holding this stereotype when the goal behind user's seeking information is not recognized by the system.

User model will further be populated with attributes (static information) shown in table 4.2 pertaining to the user. Some values are translated to categorical form as shown in table 4.2 to facilitate in the inference process.

Attribute	**Range of values**
Age	less_than_35/ between_35_and_50/ more_than_50
Gender	male/ female
Marital status	married/ unmarried
Risk level (Life risk level according to profession)	high/ low/ medium
Salary level	high/ low/ medium

Table 4.2: Static user attributes stored in user model

Design of Stereotyper sub-module

Stereotyper sub-module is assigned with the task of deducing user stereotype from user input to the system. As collaborative method has been identified as the appropriate inference technique, the user stereotype need be determined by comparing user input against a set of (pre-categorized) past enquiries of users. Hence, supervised learning will be used to train a model with a comprehensive set of already available user enquiry data and infer the current user stereotype (which would be one of the stereotypes mentioned in previous section) by consulting the trained model with current user input.

Here, the categorization process is in fact a text classification problem, and hence, the naïve bayes classifier for text classification will be employed for the purpose.

The stereotyper will also implement a mechanism to refine or clarify the initially construed stereotype in case the initial user input is not adequate for a firm conclusion.

Flow diagram shown in appendix A illustrates how the stereotyper will function in determining the current user stereotype.

As shown in the diagram, in case the probability of inferred stereotype is below threshold value, user will be asked to repeat or rephrase his/her enquiry in order to confirm (or retract from) the initial (unconfirmed) stereotype.

Design of Predictor sub-module

Predictor sub-module carries out the task of formulating system response (system action) that will be most appropriate/ productive in addressing the user's goal according to specific needs of the individual user. As indicated in figure 4.2.3, it makes use of available static information about the particular user and, based on the general system action proposed by the DM module, derives the user-specific system action to be performed. Bayesian networks (BN) will be employed as the inference mechanism in this case where the inference will be drawn from user specific information acting as initial (parent) nodes in the network.

Two main general system actions have been identified to be considered within prototype system as shown in table 4.3.

General system action	Description
Give_PolicyInfo_For_Self	This is chosen when the user's goal is identified as to take an insurance policy for user himself.
Give_PolicyInfo_For_Child	This is chosen when the user's goal is identified as to take an insurance policy on behalf of a child of the user.

Table 4.3: Main general system actions within prototype system

Accordingly, two BNs will be constructed within prototype system to provide user-tailored system responses with respect to above main general system actions as shown in figure 4.4 and figure 4.5.

BN for the general system action – Give_PolicyInfo_For_Self

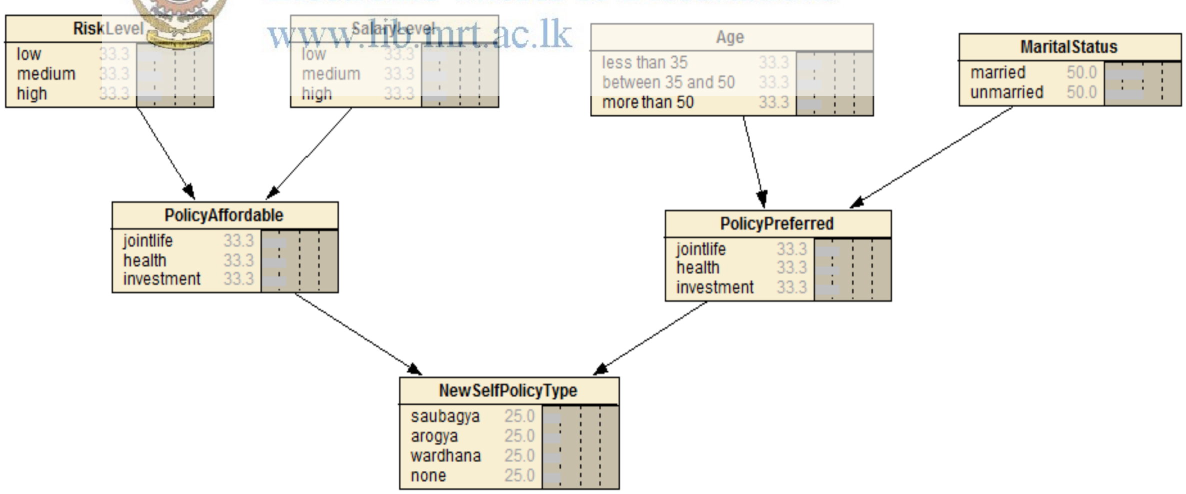


Figure 4.4: BN for Give_PolicyInfo_For_Self

Here, the BN makes use of 4 items of static information regarding the user. The type of policy that is affordable for the user is inferred from his/her risk level and salary level whereas the policy most likely to be preferred by the user is determined from age and marital status. The most viable policy for the user (which will be recommended by user-specific system action) is decided by considering above two policy types.

BN for the general system action – Give_PolicyInfo_For_Child

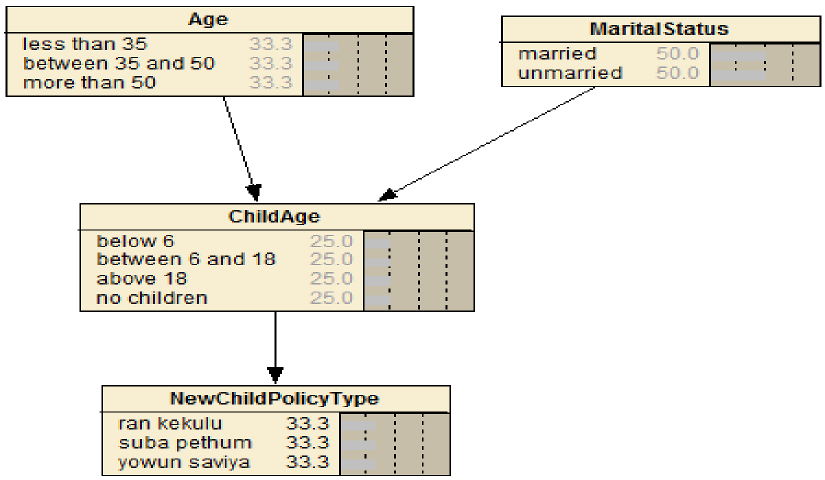


Figure 4.5: BN for Give_PolicyInfo_For_Child

BN in this case employs age and marital status of the user to infer age range of the child, and thereby, deduce the appropriate child insurance policy viable for the user.

Sequence diagram

Sequence diagram for the user modeling component is as shown in figure 4.7.

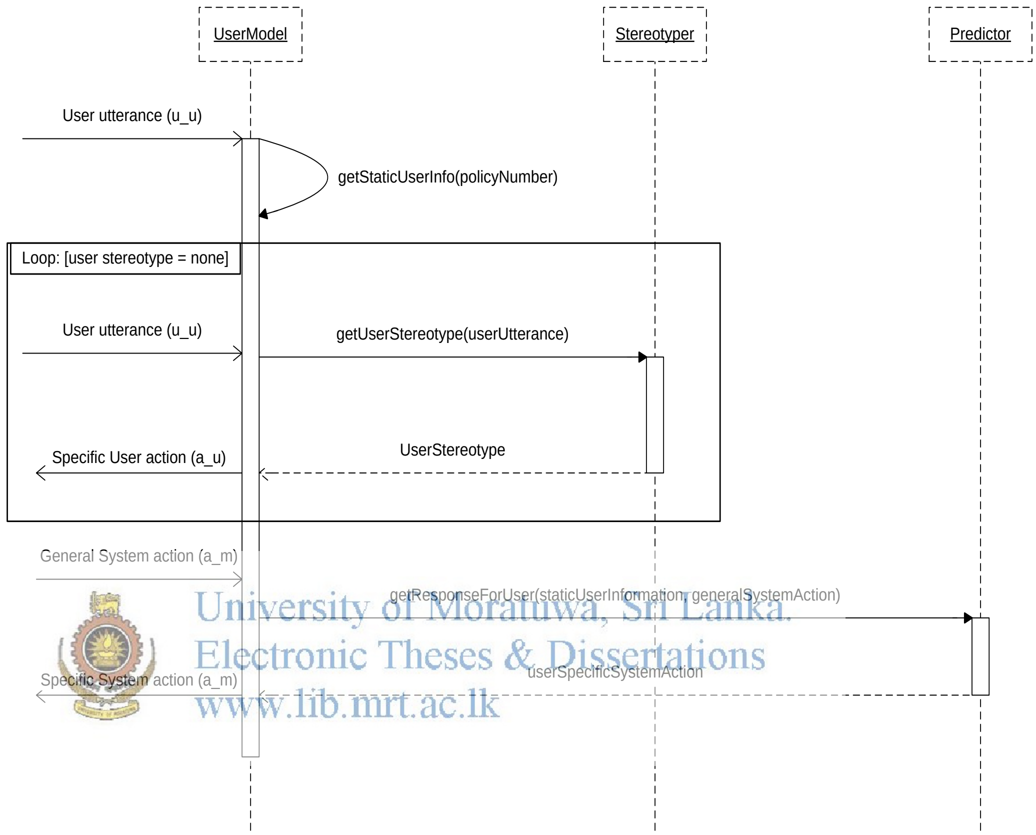


Figure 4.6: Sequence diagram for User modeling component

4.3.3. Integration of user modeling component to basic dialogue system

Integration of user modeling component is done by using the facilities provided within OpenDial framework to plug-in external components to the system. The user modeling component will hence contribute to the functionality of the system by communicating with each internal module through monitoring and updating the relevant state variables in the dialogue state. Figure 4.8 illustrates how the user modeling component extends the functionality of basic dialogue system described in section 4.3.1.

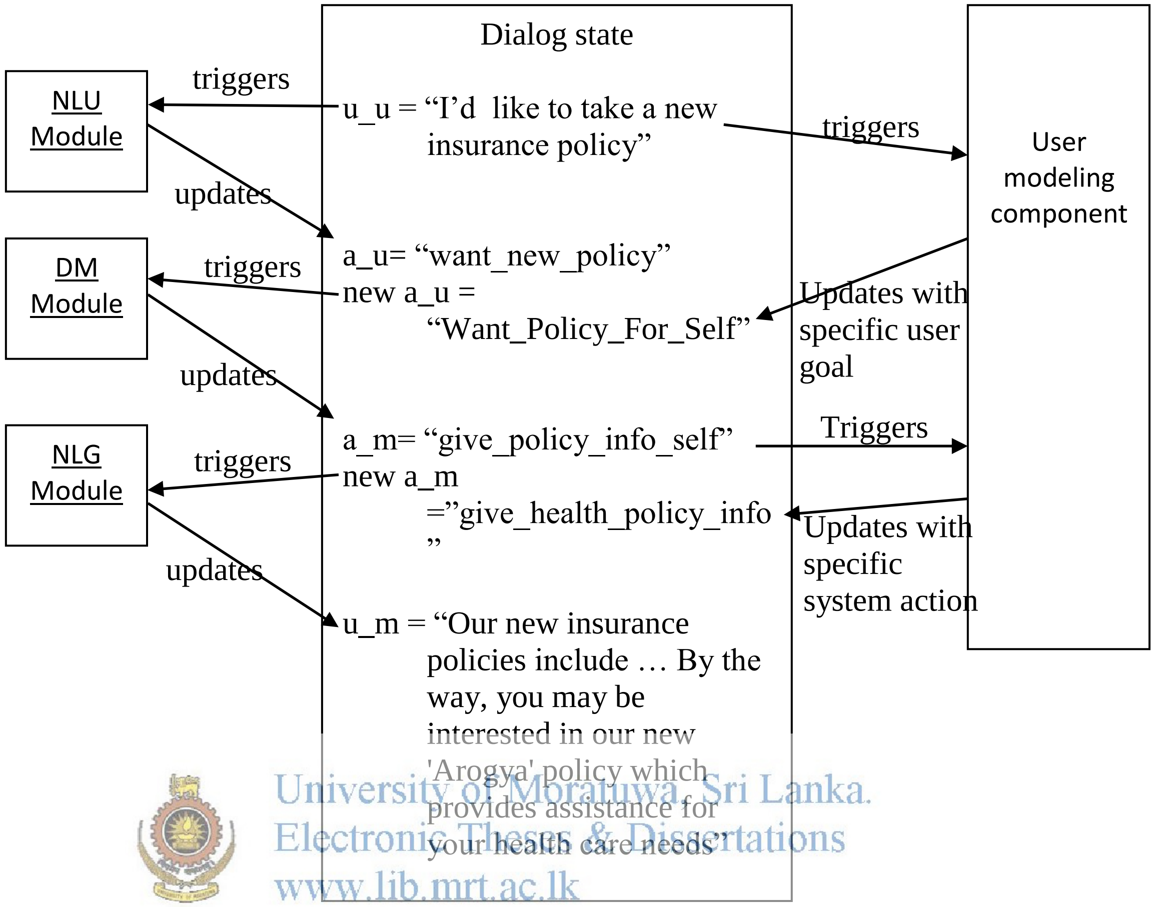


Figure 4.7: Example workflow of prototype system

In the example case shown in figure 4.7, while the basic dialogue system decides that the user wants information about new policies, user modeling component determines that the user wants an insurance policy specifically for herself which is conveyed as specific user goal to the dialogue system. Upon receiving the consequent general system action as to give out information about insurance policies (suitable for persons in general), user modeling component uncovers that a health insurance policy may suit the user best (considering various user-specific factors) and causes the system action to change accordingly. System response will thereby include user tailored information to effectively guide the user in selecting an insurance policy most appropriate for his/her needs.

5. IMPLEMENTATION

5.1. Chapter Overview

Implementation of the proposed system has been carried out with respect to the design of each system component and their integration determined at the previous stage. Accordingly, this chapter discusses on the key aspects of prototype implementation by explaining the development steps followed while carrying out the task. Several third party resources have been utilized in the process in order to incorporate state-of-art techniques to the system as well as ensure its timely completion.

5.1.1. Manipulation of third party resources

Resource	Available at:
OpenDial Framework (Version 0.95)	http://opendial.googlecode.com/svn/downloads/
Weka 3 API	http://www.cs.waikato.ac.nz/ml/weka/downloading.html
Netica Java API (Netica-J)	https://www.norsys.com/netica-j.html#download

Table 5.1: Manipulation of third party resources

5.2. Implementation of System components

Development of main system components has been carried out in four stages as follows.

1) Implementation of basic dialogue system
2) Implementation of Stereotyper (User modeling sub-component)
3) Implementation of Predictor (User modeling sub-component)
4) Integration of user modeling component to basic dialog system

5.2.1. Implementation of basic dialogue system

As identified in Design chapter, basic dialogue system has been built on top of the OpenDial framework which is an open source Java-based platform for dialogue system creation. Hence, conforming to OpenDial standard for dialogue domain construction, an XML-encoded dialog domain specification was created as relevant for the prototype system.

Dialogue domain specification for prototype system:

Additional State variables:

Apart from the system defined state variables such as u_u, a_u, a_m and u_m (refer section 4.3.1), a user-defined state variable 'current_question' was added to the dialogue state in order to have a system initiated and continuous conversation with the user as follows.

```
<initialstate>
	<variable id="current_question">
		<value>howHelp</value>
	</variable>
</initialstate>
```


At the start of the dialogue, current_question is set to the initial question from the system (e.g. How may I help you?) which triggers subsequent model (below) to update system action (a_m) to ask current question from user.

```
<model trigger="current_question">
	<rule>
		<case>
			<effect>
				<set var="a_m" value="Ask({current_question})"/>
			</effect>
		</case>
	</rule>
</model>
```

Basic NLU module:

NLU module for basic system is implemented to convert user utterance to the appropriate user action. This is carried out via the rudimentary technique of deciding user action according to keywords found in user utterance. For example, if user utterance contains phrases such as ‘for…son’ or ‘for…daughter’, system concludes the user action (user goal) as ‘wanting a child insurance policy’ (Want_Policy_For_Child).

Above example of NLU logic is implemented in OpenDial’s XML notation as follows.

```
<model trigger="u_u">
        <!-- Rule to determine user action -->
        <rule>
                <case>
                <condition>
                        <if var="current_question" value="howHelp"/>
                                <or>
                                        <if var="u_u" value="for * son" relation="in"/>
                                        <if var="u_u" value="for * daughter" relation="in"/>
                                </or>
                </condition>
                <effect>
                        <set var="a_u" value="wantPolicyForChild"/>
                </effect>
                </case>
        …
                …
                …
        </rule>

</model>
```


As shown above, NLU module is triggered by user utterance (u_u) and user’s goal is determined specifically by user’s answer given to initial system’s question.

Basic DM module:

DM module determines system action appropriate for the user goal (user action) as shown in following extract from DM specification in the system.

```
<model trigger="a_u">
        <!-- Rule to determine system action -->
                <rule>
                <case>
                        <condition>
                        <ifvar="a_u"value="wantPolicyForChild"/>
                        </condition>
                        <effect>
                        <setvar="a_m"value="givePolicyInfoChild"/>
                        </effect>
                </case>
                        …
                        …
                </rule>
</model>
```

The module is triggered by user action (a_u) and the system action (a_m) is set to 'give out child policy information' when user action is identified as 'wanting a child insurance policy' in above example.

Basic NLG module:

This produces the system utterance corresponding to the chosen system action and is implemented using XML notation as in above modules.

```
<model trigger="a_m">
        <rule>
                <case>
                        <condition>
                                <if var="a_m" value="Ask(howHelp)"/>
                        </condition>
                        <effect util="1">
                                <set var="u_m" value="How may I help you? />
                        </effect>
                </case>
                …
                …
```

However, while formulating the system response, the module also updates 'current_question' in order to ask the next question from user when appropriate.

```xml
<case>
        <condition operator="or">
                <if var="a_m" value="givePolicyInfoChild"/>
                <if var="a_m" value="givePolicyInfoSelf"/>
                <if var="a_m" value="givePolicyInfoOther"/>
        </condition>
        <effect>
                <set var="current_question" value="answerGood"/>
        </effect>
</case>
```

As shown above, when the system action is formulated as to give out relevant policy information (child, self or other), current_question is updated as 'answerGood' which causes the system to ask user whether he/she is satisfied with provided information (i.e. whether the system answer is good enough for the user).

Complete domain specification for basic dialogue system is presented in Appendix B.

Functionality of basic dialogue system:

Snapshot of OpenDial GUI shown in figure 5.1 illustrates the functionality of basic dialogue system constructed from above dialogue domain specification.

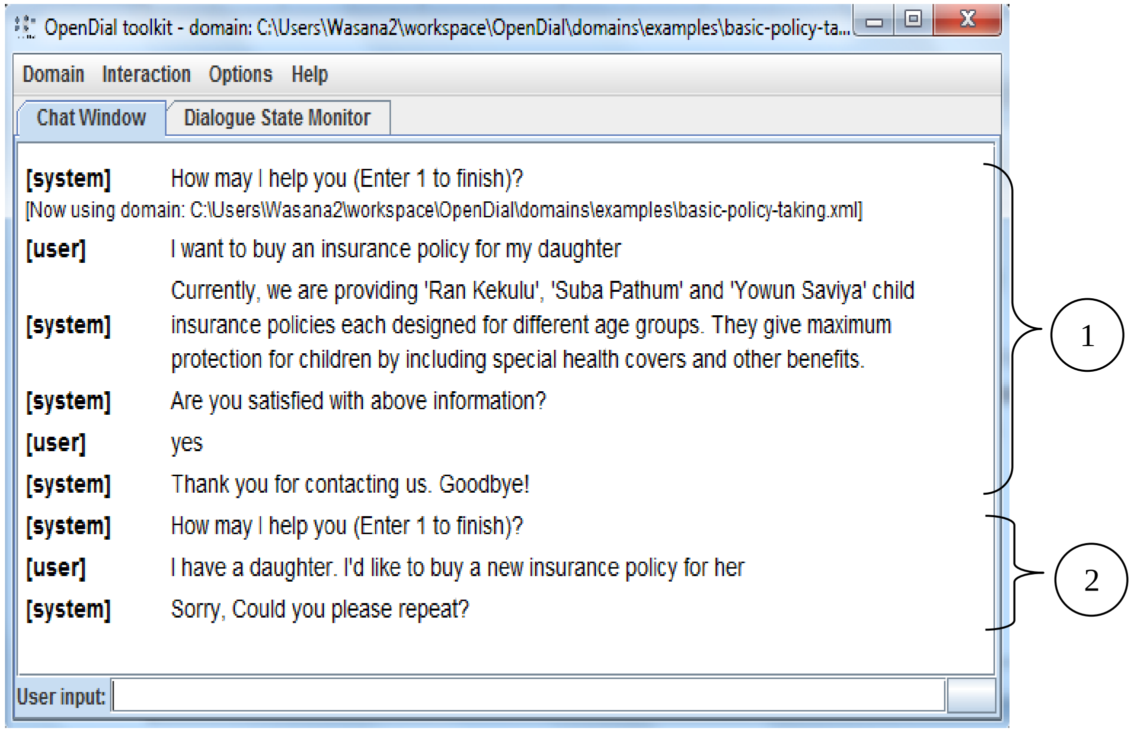


Figure 5.1: Functionality of basic dialogue system

In the example dialogue shown in figure 5.1, the system was able to provide general information regarding child insurance policies to meet the user goal in first conversation, while in the second conversation, the system was not able to identify the user goal from user utterance.

5.2.2. Implementation of User modeling component

Implementation of Stereotyper sub-component

Purpose of Stereotyper is to output the relevant user stereotype (user goal) that can be inferred from user utterance. This is accomplished by consulting a model trained via supervised learning with already available user enquiry data. A dataset of 200 user enquiry records was used for training the model which contained 50 matching user enquiry records for each of the four pre-identified stereotypes (refer section 4.3.2).

A naïve Bayes classifier was build and used for inferring the user stereotype by employing the algorithms provided by Weka 3 API for supervised learning.

A technique for verifying the initially inferred stereotype was also implemented as indicated in section 4.3.2, where the stereotype will be stored until further confirmation from user in case the probability of inferred stereotype is below a threshold value of 0.8.

Implementation of Predictor sub-component

Task of Predictor sub-module is to deduce the user-specific system action that would be most helpful for achieving the user goal. The system action proposed by predictor hence corresponds to the suggestion of most relevant insurance policy for the user (along with general information provided by basic dialogue system) which can meet the user-specific needs. As indicated in section 4.3.2, Bayesian networks have been used to infer user-specific system action for each of the two main general system actions within prototype system.

The two Bayesian networks were implemented using the facilities provided by Netica-J API for Bayesian inference. Implementation was carried out according to their design proposed in the previous chapter.

Accordingly, for general system action 'Give_PolicyInfo_For_Self', Bayesian network will suggest one of the following insurance policies as the most suitable for a user according to his/her specific characteristics.

Saubagya	A joint life insurance policy that can be taken for both user and his/her spouse
Arogya	A health insurance policy
Wardhana	A life insurance policy targeted at investors

Table 5.2: Insurance policy types for grown-up people

For general system action 'Give_PolicyInfo_For_Child', one of the following child insurance policies are suggested through Bayesian inference.

Ran Kekulu	Suitable for children below 6 years of age
SubaPathum	Includes special benefits for children between 6 – 18 years
YowunSaviya	For young people above 18 years

Table 5.3: Insurance policy types for children

In addition to the core classes within the user modeling component, the classes 'Customer' and 'DBConnector' were constructed to access the static user information (described in 4.3.2) stored in a MySql database.

5.2.3. Integration of User modeling component to basic dialogue system

User modeling component is integrated to the basic dialogue system by using the facilities provided by OpenDial to integrate external modules to the system. This is accomplished by implementing the Module interface of OpenDial which particularly makes available the current dialogue state to the externally attached module.

The Module interface is implemented by the class UserModeler which is attached to the system as a synchronous module[10] in order to trigger module's internal processing at each update of the dialogue state. This is carried out mainly by implementing 'trigger' method of Module interface which is activated at each state update with the current dialogue state and a list of recently updated state variables passed to it as method arguments.

Following extract of UserModeler class shows how the user utterance (u_u) is accessed by the module.

```
public void trigger(DialogueState state, Collection<String>updatedVars) {
        if (updatedVars.contains("u_u") && state.hasChanceNode("u_u")
        String userUtterance =
        state.queryProb("u_u").toDiscrete().getBest().getValue("u_u").toString();
                ...
        }
        ...
}
```

[10]OpenDial allows integrating both synchronous and asynchronous modules to the system.

UserModeler module also re-updates the dialogue state with the specific user and system actions inferred by the Stereotyper and Predictor components respectively.

Following code snippet shows how the user action (a_u) is updated with user stereotype within the UserModeler class.

```
String stereotype = userSterotyper.getStereotype(userUtterance);
assignment = new Assignment("a_u", stereotype);
try       {
state.addToState(assignment);
} catch (DialException e)
{         e.printStackTrace();      }
```

Domain specification initially constructed for the basic dialogue system is also changed to facilitate integration of the user modeling component. Accordingly, the specification is changed to:

1) Ask the user to enter his/her policy number from which the stored user information can be retrieved for user-specific inference.
2) Include a mechanism to refine the user stereotype when the specific user action cannot be inferred. This is carried out by directly asking the user for whom the new insurance policy would be taken.

Snapshots of OpenDial GUI from figure 5.2 to figure 5.4 illustrates how the functionality of basic dialogue system is extended with integration of user modeling component.

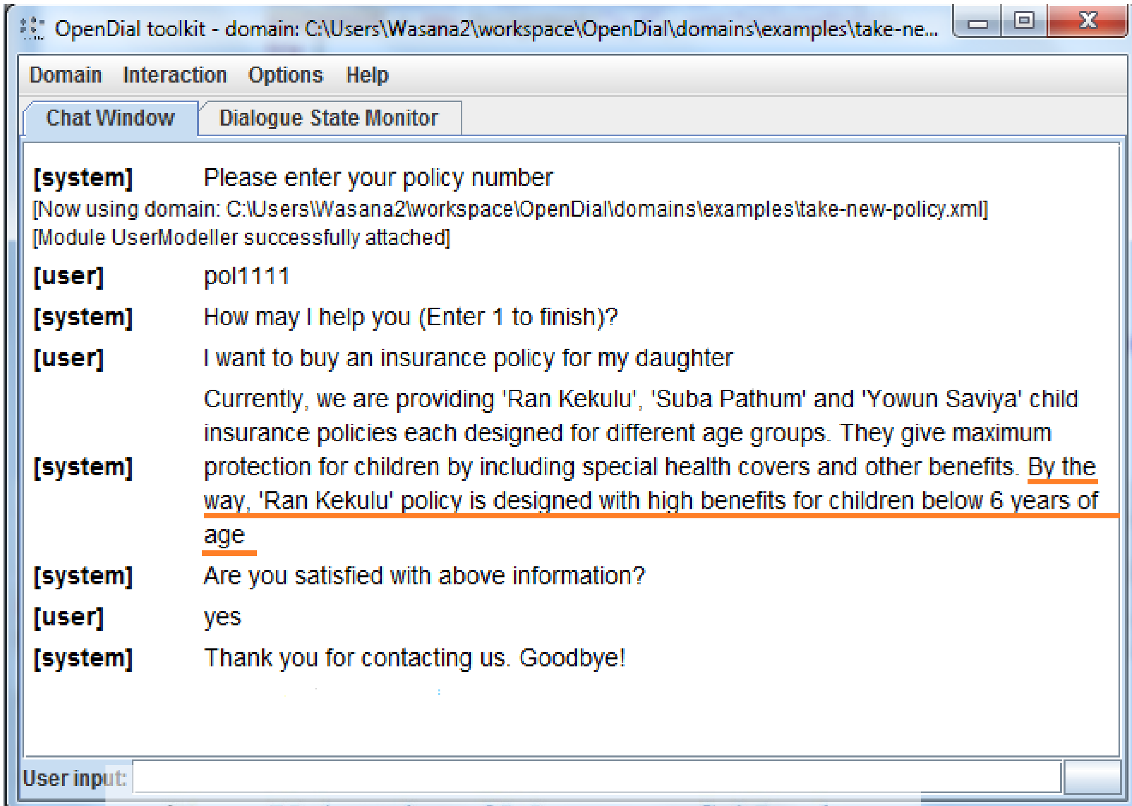


Figure 5.2: Functionality of prototype system – giving out user-tailored information

In the example shown in figure 5.2, apart from giving general information about child insurance policies, system considers the specific user characteristics to form relevant extra information (shown under-lined) that may help user in achieving his/her goal.

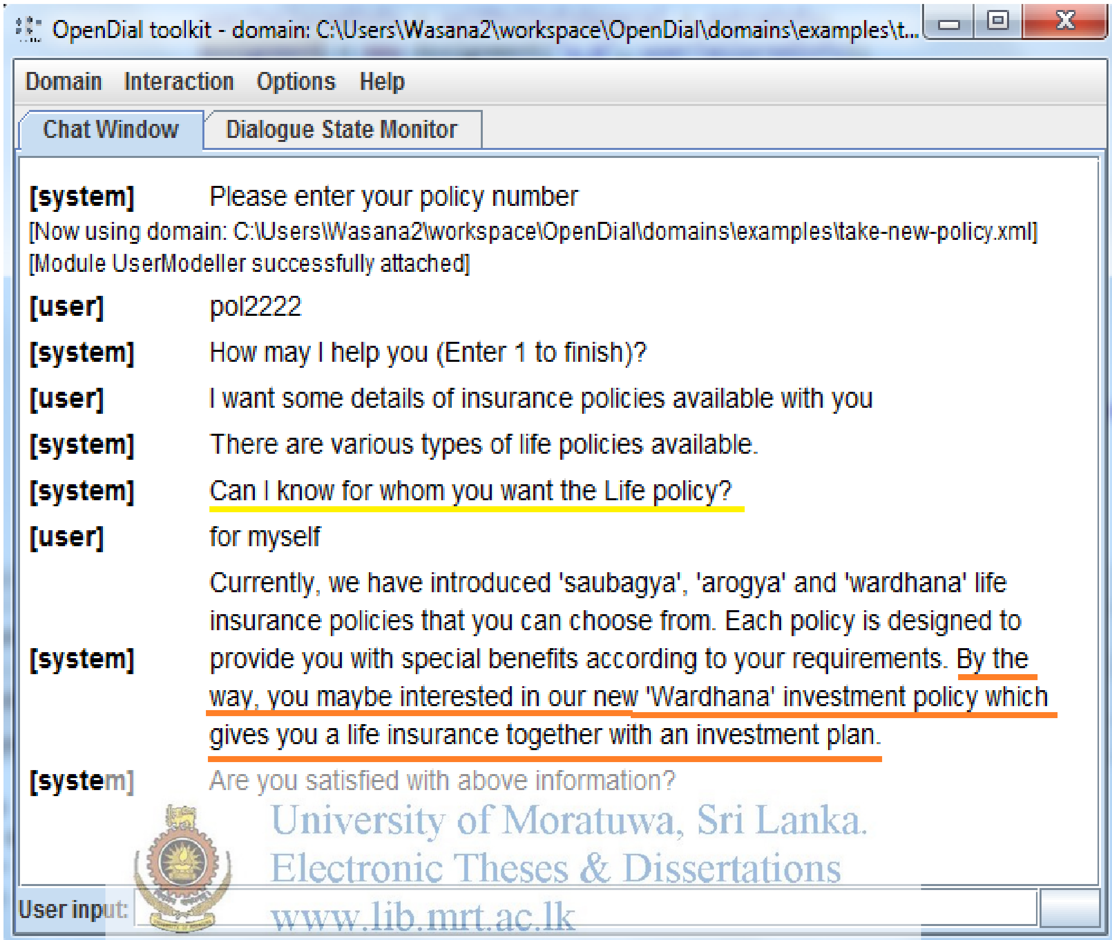


Figure 5.3: Functionality of prototype system - refining initial user stereotype

Figure 5.3 shows how the initially learned stereotype is refined through the dialogue where the user is asked to provide further details about his/her enquiry.

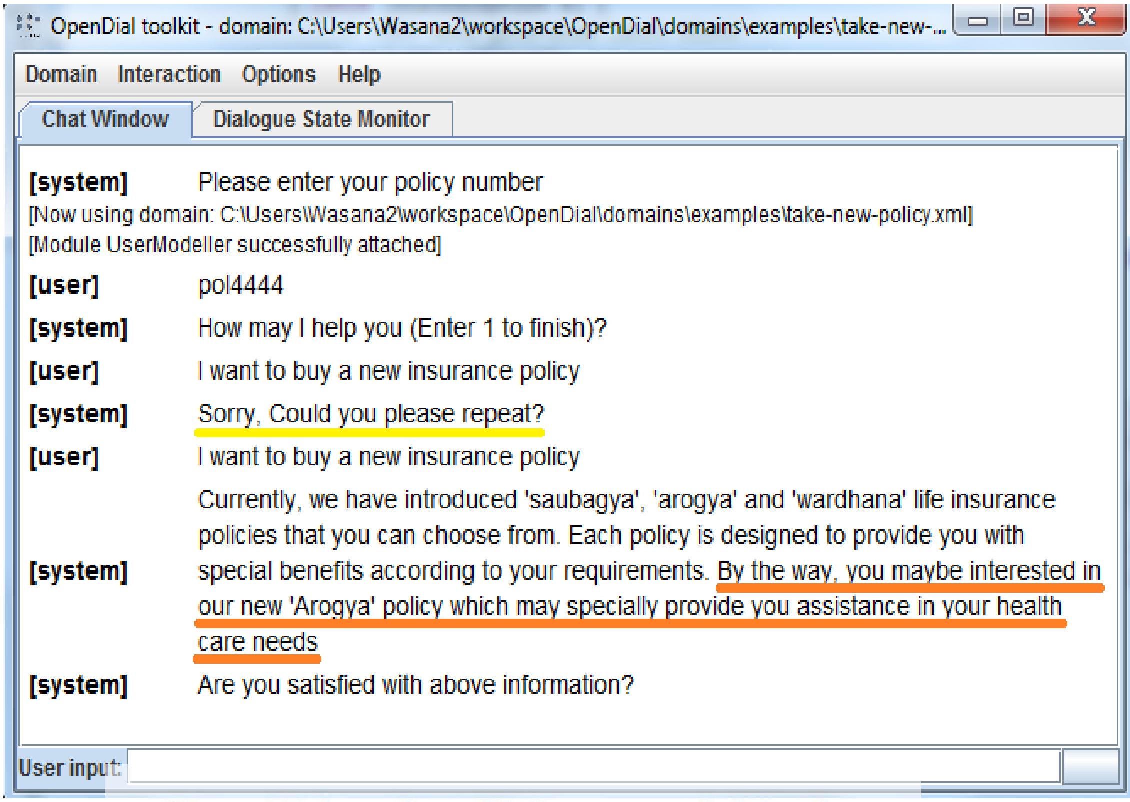


Figure 5.4: Functionality of prototype system - verifying user stereotype

Example dialogue shown in figure 5.4 illustrates how the user action is verified by the Stereotyper module. Here, the initial stereotype ('Want_Policy_For_Self') which has been inferred with probability below threshold value is confirmed by further user input.

6. EVALUATION

6.1. Chapter Overview

Evaluation of the prototype has been carried out by following a comprehensive evaluation scheme which covered both objective and subjective evaluation of the system. Objective evaluation was conducted in order to obtain a measurable indication of system performance compared to a baseline dialogue system. Subjective evaluation was used to evaluate user opinion from a set of subjects familiar with the domain. The chapter primarily discusses evaluation procedures followed, results obtained as well as interpretation and conclusion based on the evaluation results.

6.2. Purpose of evaluation

Main purpose of evaluation of the prototype system is to observe the contribution of its user model (user modeling component) to the core dialogue system. As indicated by [34], such an evaluation of user model in a dialogue system should be able to demonstrate a measurable effect on performance of the system due to consulting this model which will be an indicator of accuracy of the user model. However, as [34] further notes, evaluation of user models in dialogue systems with respect to existing systems is not a feasible task as the generally accepted guidelines for such an evaluation have not yet been formulated, and many researchers have conducted evaluations of similar systems according to the specific capabilities of their systems. In the light of this information, it is concluded that comparing the performance of prototype system with that of a conventional (hand-crafted) dialogue system built for the same domain/requirements would be more apt for achieving the purpose of evaluation.

6.3. Experimental Set-up

Version of the prototype system prior to integrating user modeling component is used as the conventional dialogue system for carrying out the aforementioned comparative evaluation. The dialogue domain specification for the conventional system is hand-crafted according to the same requirements addressed by the prototype system.

Both objective and subjective evaluations were carried out through user trials involving 15 subjects chosen from employees of an insurance company who were well-acquainted with the domain chosen for the prototype. The subjects were first briefed about how to interact with the systems (prototype and conventional) and each was given a set of 5 user profiles each of which had a specific goal to be achieved through the system.

An example user profile is as follows.

Policy number: pol1111
Age: 34
Gender: Female
Marital Status: married
Profession: Cashier
Salary level: Low
Risk level: low
Goal if interaction: Want to buy an insurance policy for my son

The subjects were asked to read each user profile provided to them and then interact with the two systems (one at a time) in order to achieve the user's goal. In case of prototype system, the subjects were asked to enter policy number when prompted whereas policy number was not required for the conventional system.

The subjects were encouraged to interact with the system in a natural manner similar to how they would interact via normal speech. No time limit was set to complete the task as it was more important to receive the information offered by the system(which is the system response to address user goal) and after the interaction, subjects were asked to answer a brief questionnaire to assist in subjective evaluation of the prototype system.

6.4.Objective evaluation

Objective evaluation has been carried out by defining a set of objective metrics which have been frequently employed in dialogue system evaluation including [33]. The metrics are directed at measuring the quality and efficiency of the dialogues carried out by the subjects.

Objective metrics:

M1: Average number of repetition requests per dialogue:

Number of repetitions in a dialogue would correspond to the number of failed attempts at identifying the user goal. A repetition occurs when the system explicitly asks the user to repeat (re-enter) his/her request. In this case, user may simply repeat the request or rephrase it to make it more understandable to the system.

M2: Average number of user rejections per dialogue:

A user rejection occurs when the system response conflicts with user goal. An example is the system giving information about policies suitable for the user when the user has asked about policies that can be taken for his/her child. Information provided by the system is not what the user has expected and hence rejected by the user.

M3: Average number of user turns per dialogue:

Number of user turns is also an indication of earlier metrics as the user turns would naturally increase with number of repetitions and user rejections. However, the prototype system includes a mechanism for refining the user stereotype which may contribute to additional user turns. Hence, user turns was also considered as a separate objective metric.

M4: Task completion rate:

As mentioned in previous section, the dialogues were not time constrained and the subjects were allowed to interact with the system until the task was complete (in both the prototype and conventional systems). However the task completion rate was observed with respect to the appropriateness of responses provided by the two systems. In case of prototype system, task was flagged as successfully completed if the system was able to give user-tailored extra information appropriate for the user and considered as not fully completed if the system was only able to provide general information according to user request. However, in case of conventional system, the task was flagged complete if the appropriate general information was provided by the system (as its purpose is necessarily to provide general information).

Results of Objective evaluation

Results obtained against objective metrics (M1 to M4) for both the prototype and conventional systems are depicted by the graphs shown in figure 6.1 to figure 6.4.

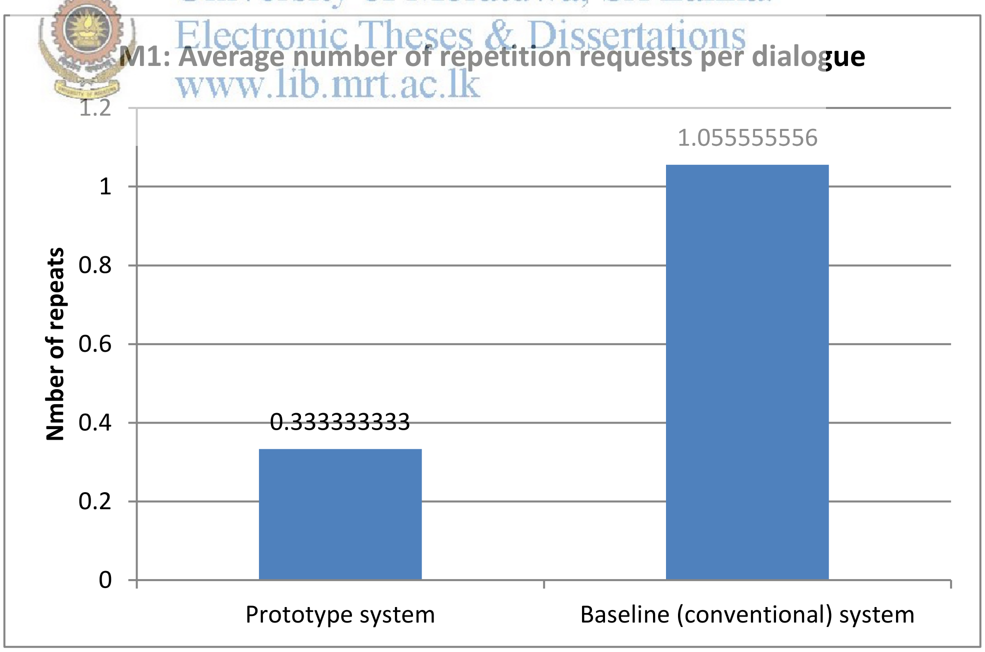


Figure 6.1: Average number of repetition requests per dialogue

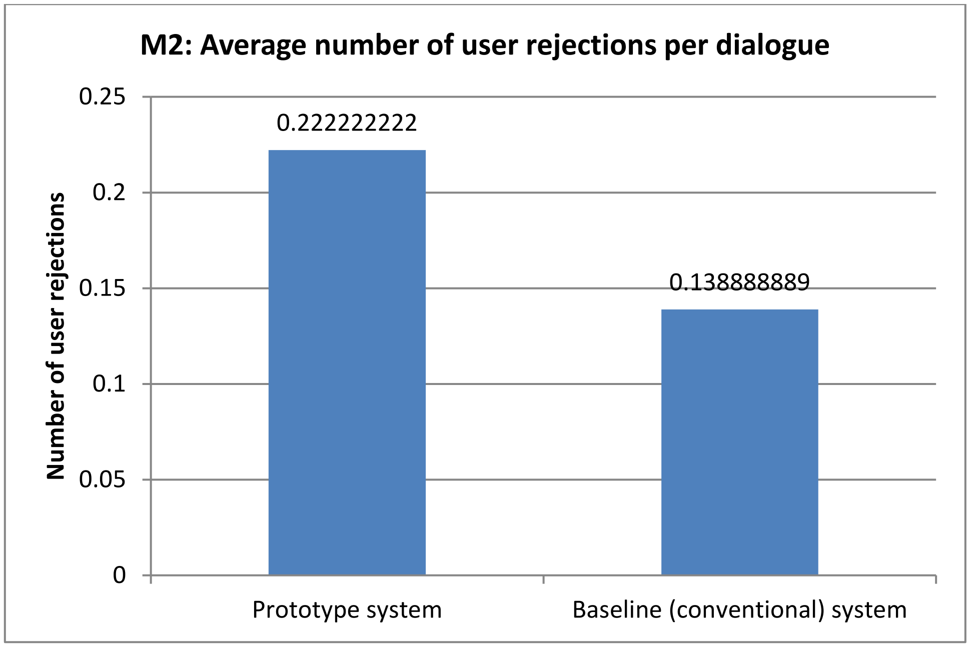


Figure 6.2: Average number of user rejections per dialogue

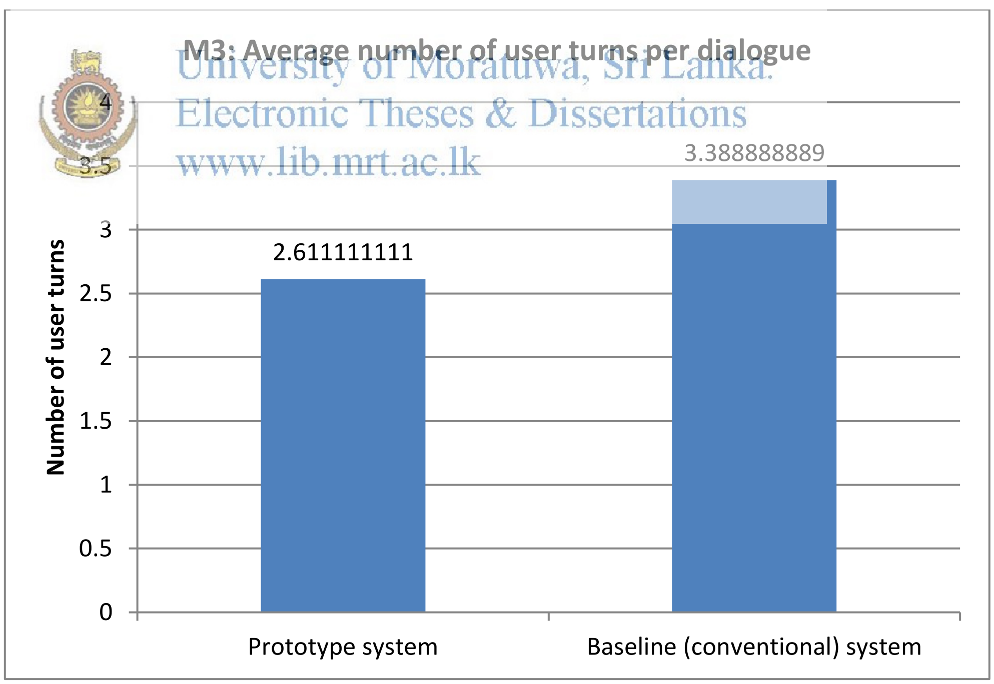


Figure 6.3: Average number of user turns per dialogue

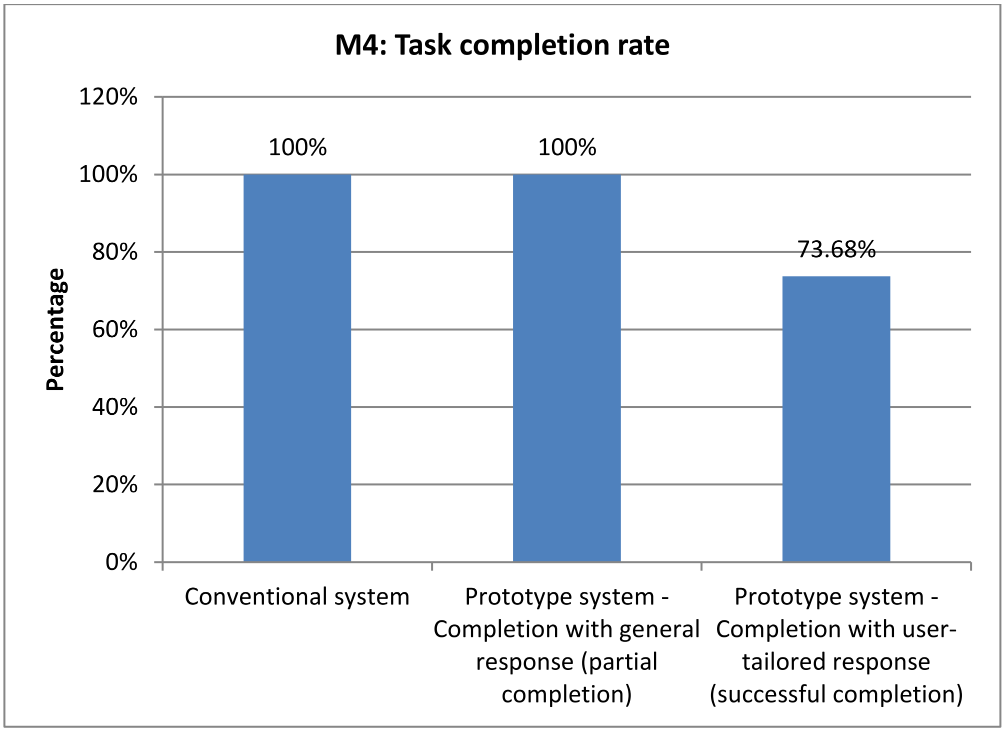


Figure 6.4: Task completion rate

Discussion of Objective evaluation

A marked increase in the average number of repeats per dialogue can be observed within conventional system while the prototype system has performed relatively better with lower number of repeats. The result suggests that prototype system is more competent in identifying the user goal with minimal user input/ turns. This can be mainly attributed to the statistical technique employed by the user model to identify the user stereotype (user goal) i.e. by consulting similar requests made previously by the users to identify current user request. Also, the prototype system applies a technique to verify the initial stereotype (in case specific user goal is not inferred) which contributes to reducing the number of user repeats.

Average number of user rejections is considerably low in both systems, with the prototype system producing a higher maximum value of 2 user rejections per dialogue. As user rejections occur when the user goal has been identified incorrectly, the result indicates that the statistical inference technique may need further improvement for accurate determination of user goal. In fact, the prototype system used only 200 data records of user requests for learning the statistical model which may be scarcely sufficient for the purpose of accurate inference. Conventional system, on the other hand, has a low user rejection rate as it has a low tendency to identify the user goal incorrectly. This is due to the interesting fact that although it may demand a high number of repeats, the conventional system uses hand-crafted rules to identify user goal which can be more robust with respect to accuracy than statistical methods.

In line with average number of repeats per dialogue, prototype system was able to perform with comparatively lower average of user turns per dialogue than that of the conventional system. This indicates that the mechanism used by prototype system to refine the user stereotype has a lower impact on number of user turns than effect due to user repeats.

Both the prototype system and conventional system have achieved 100% task completion rate with respect to responding to the user with basic policy information appropriate for his/her need. However, the prototype system exhibits a lower task completion rate when the task was considered as giving out user-tailored extra information (together with general information) as system response. This was due to the fact that in some cases the system was uncertain about the specific user conditions when formulating user-tailored response. For example, when a user asks about child insurance policies and the database records indicate that user is unmarried, the system has refrained from providing extra information due to contradictory user conditions. However, a satisfactory level of around 74% task completion rate has been achieved by the prototype system.

6.5.Subjective evaluation

Subjective evaluation was based on the answers given by the aforementioned subjects to the questionnaire given to them after completing their interaction with the two systems. The questionnaire has been derived by referring to the set of questions used by [33] in the evaluation of his system. Accordingly, the questionnaire contained five questions aimed at evaluating user opinion about several aspects relating to performance of the two systems. Five alternative answers were provided along with each question for the subject to choose from, in order to facilitate the analysis process.

Results of subjective evaluation

Graphs shown in figure 6.5 to figure 6.9 shows the results obtained for each question (Q1 to Q5) for both the prototype and conventional systems.

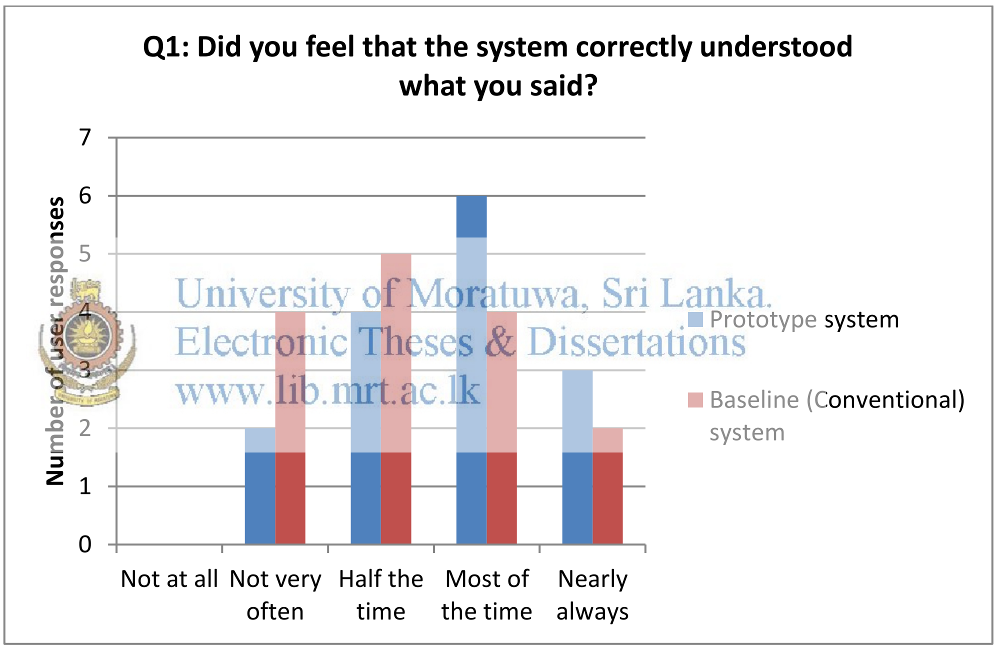


Figure 6.5: Results of subjective evaluation - Question 1

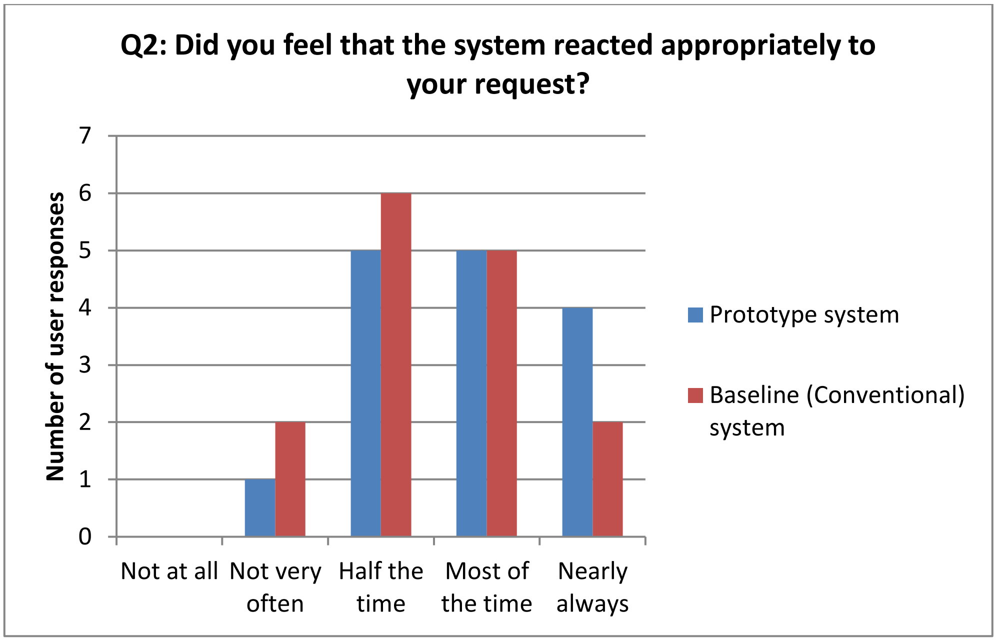


Figure 6.6: Results of subjective evaluation - Question 2

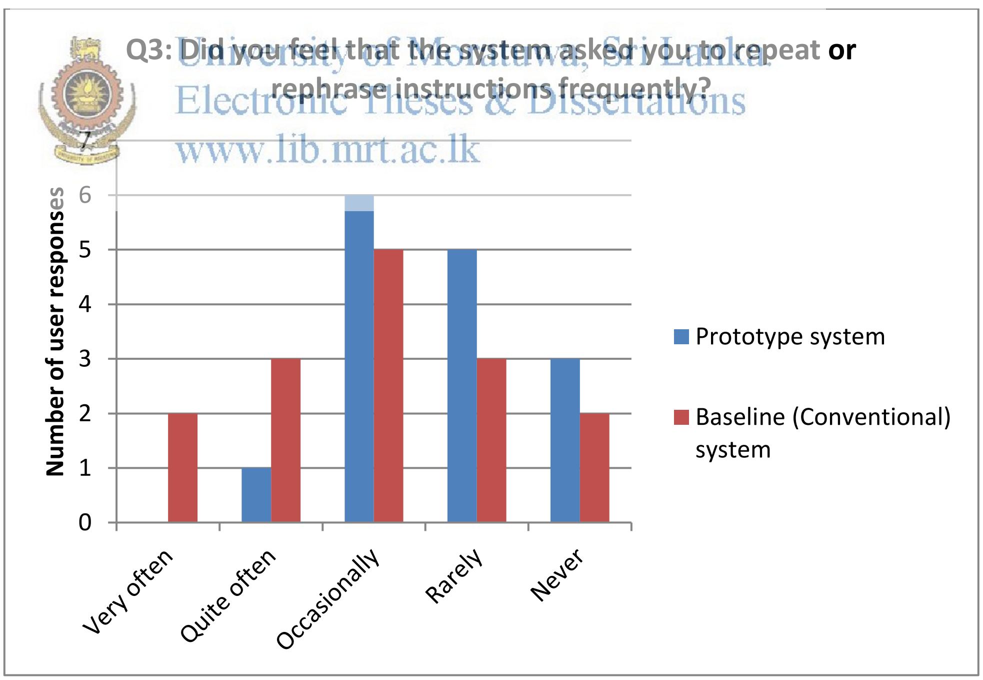


Figure 6.7: Results of subjective evaluation - Question 3

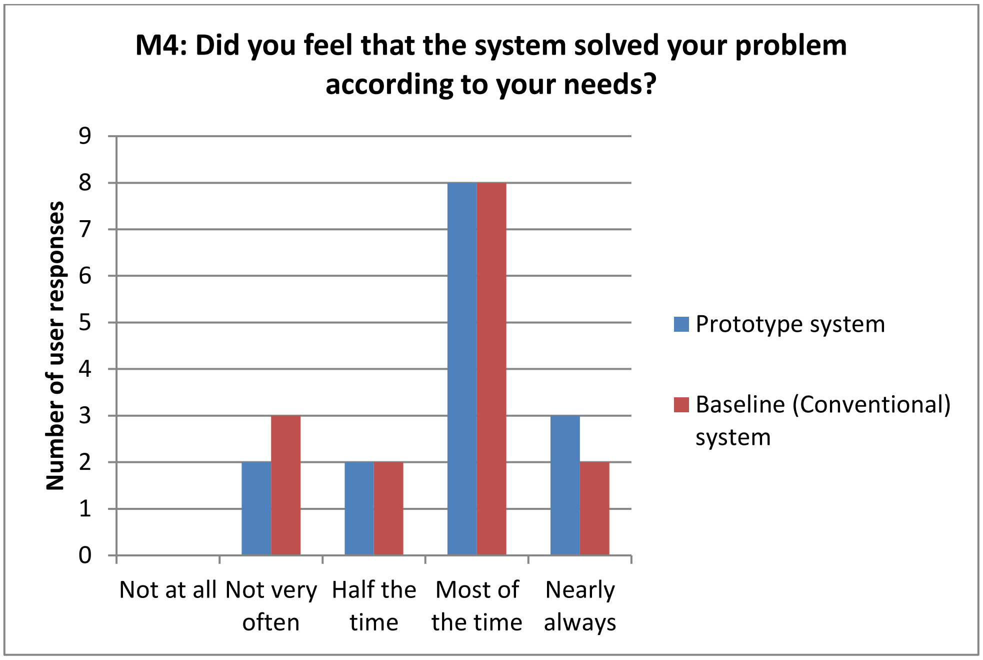


Figure 6.8: Results of subjective evaluation - Question 4

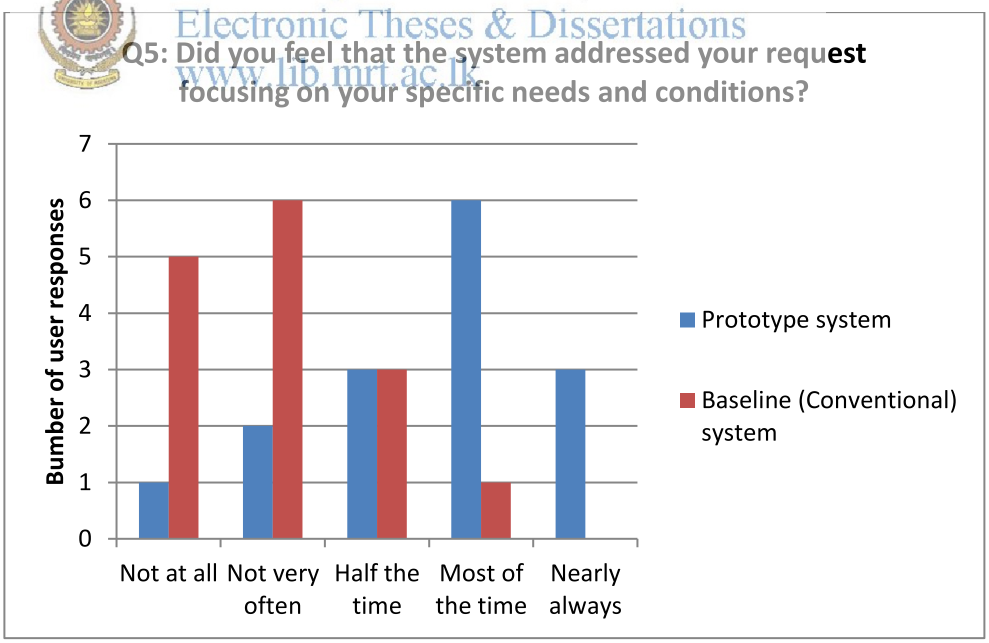


Figure 6.9: Results of subjective evaluation - Question 5

Discussion of subjective evaluation

For each question asked, the prototype system has gained more favorable feedbacks than for the conventional system. Particularly, the user responses for Q1 and Q2 indicates that the prototype system is viewed as more competent in understanding the user goal while the results obtained for questions Q3 to Q5 shows that prototype system outsmarts the conventional system in addressing user's specific needs with as little user input/turns as possible.

6.6.Discussion of overall results

Results of both objective and subjective evaluations suggest that the prototype system stand out as the more viable solution for attending the user requests in an efficient as well as user specific manner. Although a comparatively higher user rejection rate was observed for the prototype system during objective evaluation, this has not been much reflected in the subjective evaluation where the results have been obtained highly in favor of the prototype system. Also, objective evaluation revealed that the prototype system exhibited lower task completion rate compared to conventional system. However, it has had no considerable effect on the subjective evaluation. The high favorability for the prototype system by the subjects may be attributed to the distinct capabilities provided by the user model in the prototype system for identifying the user goal as well as formulating the user-specific response. While the objective evaluation pointed out a few weaknesses in the prototype system, it is observed that results of the subjective evaluation regarded the prototype system as the more convenient solution to address the problem domain.

7. CONCLUSION

7.1.Chapter Overview

Preceding chapters illustrated on the work that has been carried out from the point of inception of the problem domain to the accomplishment of the identified solution. With the overall completion of the project author was able to affirm the viability of the approach as well as to introduce an implemented version of the solution. Purpose of this chapter is to draw final conclusions for the project by reviewing the work done with respect to achieving its ultimate goals and covering a discussion of future work related to the project.

7.2.Retrospection on the overall project

Making dialogue systems user adaptive has been the focus of attention of many researchers in natural language systems, as such a system will cater an ideal solution to a plethora of communicative needs in today's world. One such goal of automated communication is to attend to the overwhelming amounts of customer enquiries occurring within companies with large customer bases such as banking and insurance industries. Goal of the project was to explore the viability of applying user modeling techniques in dialogue systems in order to enable user-adaptive communication especially within above domains.

Accordingly, a comprehensive literature review was carried out by first exploring the architecture and concepts in current natural language systems and then extending the research towards identifying the current advancements and techniques related to user modeling in dialogue systems. Dialogue management component in a dialogue system was found to be the key candidate to which a user model can be incorporated for achieving the goal of user-adaptation within the system. An in-depth review of the extant user modeling techniques was also conducted which included discussions on acquisition, representation as well as plan recognition approaches applicable for the task of user modeling. It was identified that statistical inference played a key role in addressing uncertainty and complexity associated in user-adaptive interaction and stereotype-based method together with collaborative approach posed as more favorable techniques to be manipulated by such a system targeted at providing automated customer-care service within the pre-identified domain.

Methodology for building the prototype system was drawn mainly based on findings from the literature review and a detailed design description for each system component as well as integration of user modeling components was subsequently derived.

The basic dialogue system of the prototype application was implemented using OpenDial, a state-of-art dialogue system introduced in recent times. Development and integration of the user modeling component was also done without deviating from the formulated design.

Finally, an evaluation of the prototype system was conducted in order to assess both the quantitative and qualitative contribution of user modeling component to the basic dialogue system. This has been accomplished by carrying out an objective evaluation on a set of performance metrics as well as subjective evaluation of the system through analysis of user opinion. Prototype system was compared to the baseline (conventional) dialogue system where considerable improvement was shown by the prototype system in both cases.

7.3.Achievement of project objectives

The initially defined project objectives were duly fulfilled during course of the project and a brief overview of how they were addressed within each phase is as follows.

Objective	**Fulfillment of objective**
Carry out an in-depth literature survey and conduct a literature review to produce a literature review document intended at selecting the most suitable approaches for	The objective had been achieved to a great extent at Literature review phase where all the identified areas were covered.

the analysis, design and development of the proposed system.	
Design and implement a prototype for the dialogue system to meet the identified requirements.	Objective was fulfilled during Design and implementation phases of the project respectively.
Carry out a critical evaluation of the prototype to verify whether system objectives have been fulfilled.	The objective has been achieved during evaluation phase of the project.

Table 7.1 : Achievement of project objectives

7.4.Achievement of project aim

The aim of the project as defined at the initial phase was;

To research, design, develop and evaluate a prototype for a user-adaptive dialogue system, especially targeted at providing an automated customer-help environment, where the system makes use of a dynamically adaptive user model to respond to user enquiries with information tailored for the individual user.

The project can be considered successful in accomplishing this primary aim especially within the time and other constraints imposed on an academic project. Author was able to design and develop the proposed system and produce a baseline product to fulfill this aim. Author understands that in order to better address real world needs, the system need be further enhanced which probably exceeds the scope of the project. Nevertheless, fulfillment of project aim has essentially provided a promising ground to achieve a more practical solution through further improvements.

7.5.Future work

The prototype system has been successful in demonstrating the potential in applying user modeling for dialogue systems to enable user-tailored interaction especially for customers of large-scale business organizations. The prototype provides an implementation of the appropriate user modeling approach and the techniques that can be followed in building a comprehensive system targeted for above domain. However, it is in essence a nascent application built using the prevalent inference techniques as its main purpose was to introduce the user modeling component to the conventional dialogue system. In an era of rapidly and frequently refined machine learning techniques, the system may be further enhanced by manipulating state-of-art learning technologies to accomplish the inference tasks within the user modeling component.

Also, manipulation of dynamic user input by the prototype system is at a rudimentary level with the use of dynamic information being restricted to merely identifying the user's goal. It is observed that dynamic behavior of users may be utilized to a great extent in determining implicit characteristics of the user that can be considered in formulating the user-specific response. For example, as shown by [6], skill level, knowledge level and degree of hastiness can be inferred by the dynamic user input and the system response can be adapted to meet these 'soft' requirements of the user in addition to addressing the main user goal.

Further, speech recognition and synthesis modules should also be integrated to the prototype system in order to enable spoken dialogue functionalities within the system.

REFERENCES

[1] J. R. Glass, "Challenges for Spoken Dialogue Systems," in *Proceedings of the 1999 IEEE Automatic Speech Recognition and Understanding Workshop (ASRY 1999)*, 1999.

[2] M. F. McTear, "Spoken dialogue technology: enabling the conversational user interface," *ACM Computing Surveys*, vol. 34, no. 1, pp. 90-169, March 2002.

[3] D. Gibbon, R. K. Moore, and R. Winski, "Assessment of Interactive Systems," in *Handbook of Standards and Resources for Spoken Language Systems*, ch. 13. Ratzlow-Druck, Berlin: de Walter Gruyter& Co., 1997.

[4] S. Kamboj. (2009, Feb.). Spoken Dialogue Systems. [Online]. Available: http://www.eecis.udel.edu/~kamboj/presentations/sigaI.sds.ppt

[5] I. Azzini, D. Falavigna, R. Gretter, G. Lanzola, and M. Orlandi, "First Steps toward an Adaptive Spoken Dialogue System in Medical Domain" in*Seventh European Conference on Speech Communication and Technology (INTERSPEECH 2001)*, 2001, pp. 1327-1330.

[6] K. Komatani, F. Adachi, S. Ueno, T. Kawahara, and H. G. Okuno, "Flexible Spoken Dialogue System based on User Models and Dynamic Generation of VoiceXML Scripts" in *Fourth SIGdial Workshop on Discourse and Dialogue*, 2003, pp. 87-96.

[7] K. Jokinen, "Natural interaction in spoken dialogue systems," University of Helsinki and University of Art and Design Helsinki, Finland.

[8] E. D.Os, L. Boves, L. Camel, and P. Baggia, "Overview of the ARISE project," in *Proceedings of the Sixth European Conference on Speech Communication and Technology(Eurospeech 1999)*, 1999, pp. 1527-1530.

[9] J. Peckham, "A New Generation of Spoken Dialogue Systems: Results and Lessons from the SUNDIAL Project," in *Proceedings of the Third European Conference on Speech Communication and Technology (Eurospeech 1993)*, 1993, pp. 33-40.

[10] D. Jurafsky et al, "The Berkeley Restaurant project," in *Proceedings of the Third International Conference on Spoken Language Processing (ICSLP 1994)*, 1994, pp. 2139-2142.

[11] S. M. Marcus, D. W. Brown, R. G. Goldberg, M. S. Schoeffler, W. R. Wetzel, R. R. Rosinski, “Prompt Constrained Natural Language-Evolving the next generation of telephony services,”in *Proceedings of the Fourth International Conference on Spoken Language Processing (ICSLP 1996)*, 1996, pp. 857-860.

[12] M. Blomberg et al, “An Experimental Dialogue system: Waxholm,” in *Proceedings of the Third European Conference on Speech Communication and Technology (Eurospeech 1993)*, 1993, pp. 1867-1870.

[13] C. Lee, S. Jung, K. Kim, D. Lee, and G. G. Lee, “Recent approaches to dialogue management for spoken dialogue systems,” *Journal of Computing Science and Engineering*, vol. 4, no. 1, pp. 1-22, March 2010.

[14] W. Minker, U. Haiber, P. Heisterkamp, and S. Scheible, “SENECA spoken language dialogue system,”*Speech Communication*, vol. 43, no. 1-2, pp. 89−102, June 2004.

[15] O. Lemon, K. Georgila, J. Henderson, and M. Stuttle, “An ISU Dialogue System Exhibiting Reinforcement Learning of Dialogue Policies: Generic Slot-Filling in the TALK Incar System,” In *Proceedings of the Eleventh Conference of the European Chapter of the Association of Computational Linguistics*, 2006, pp. 119–122.

[16] F. Weng et al, “CHAT: A Conversational Helper for Automotive Tasks,’ In *Proceedings of the Ninth International Conference on Spoken Language Processing (Interspeech 2006 – ICSLP)*, 2006, pp. 1061−1064.

[17] A. Raux, “Flexible turn-taking for spoken dialogue systems,” Ph.D. dissertation, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 2008.

[18] M. McTear. (2012, Feb.). Spoken dialogue technology: achievements and challenges. University of Ulster. [Online]. Available: www.paulmckevitt.com/imagineering-quarter/slides/mctear.ppt

[19] G. Fischer. “User modeling in human-computer interaction,” *User Modeling and User-Adapted Interaction*, vol. 11, No. 1-2, pp. 65-86, 2001.

[20] A.Kobsa and W. Wahlster, “User models in dialogue systems,” in *User Models in Dialogue Systems*. Berlin: Springer, 1989, pp. 4-34.

[21] A. Kobza, “User modeling in dialogue systems: potentials and hazards,” *AI & Society*, vol. 4, no. 3, pp. 214-240, 1990.

[22] K. S. Jones, "Realism about user modeling," in *User Models in Dialogue Systems*, A. Kobsa and W. Wahlster. Berlin: Springer, 1989, pp. 341-363.

[23] E. Schuster, "Establishing the relationship between discourse models and user models," *Computational Linguistics*, vol. 14, no. 3, pp. 82-85, September 1988.

[24] J. Kay, "User modeling for adaptation," in *User Interfaces for All: Concepts, Methods, and Tools*, C. Stephanidis, Ed. New Jersey: Lawrence Erlbaum Associates, Inc., 2001, pp. 271-293.

[25] R. Kass and T.Finin, "Modeling the user in natural language systems," *Computational Linguistics*, vol. 14, no. 3, pp. 5-22, 1988.

[26] E. Rich, "User modeling via stereotypes," *Cognitive Science*, vol. 3, no. 4, pp. 329-354, October 1979.

[27] R. kass, "Implicit acquisition of user models in cooperative advisory systems," Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA, Tech. Rep. MS-CIS-87-05, 1987.

[28] D. Sleeman and J. S. Brown, Eds., *Intelligent tutoring systems*, New York, NY: Academic Press, 1982.

[29] S. Carberry, "Techniques for plan recognition," *User modeling and user-adapted interaction*, vol. 11, no. 1-2, pp. 31-48, 2001.

[30] I. Zukerman, D. W. Albrecht, "Predictive statistical methods for user modeling," *User modeling and user-adapted interaction*, vol. 11, no. 1-2, pp. 5-18, 2001.

[31] D. Chin, ''KNOME: modeling what the user knows in UC,'' in *User models in dialogue systems*, ed. by A Kobsa and W Wahlster, Springer-Verlag, Berlin, 1989, pp. 74-107.

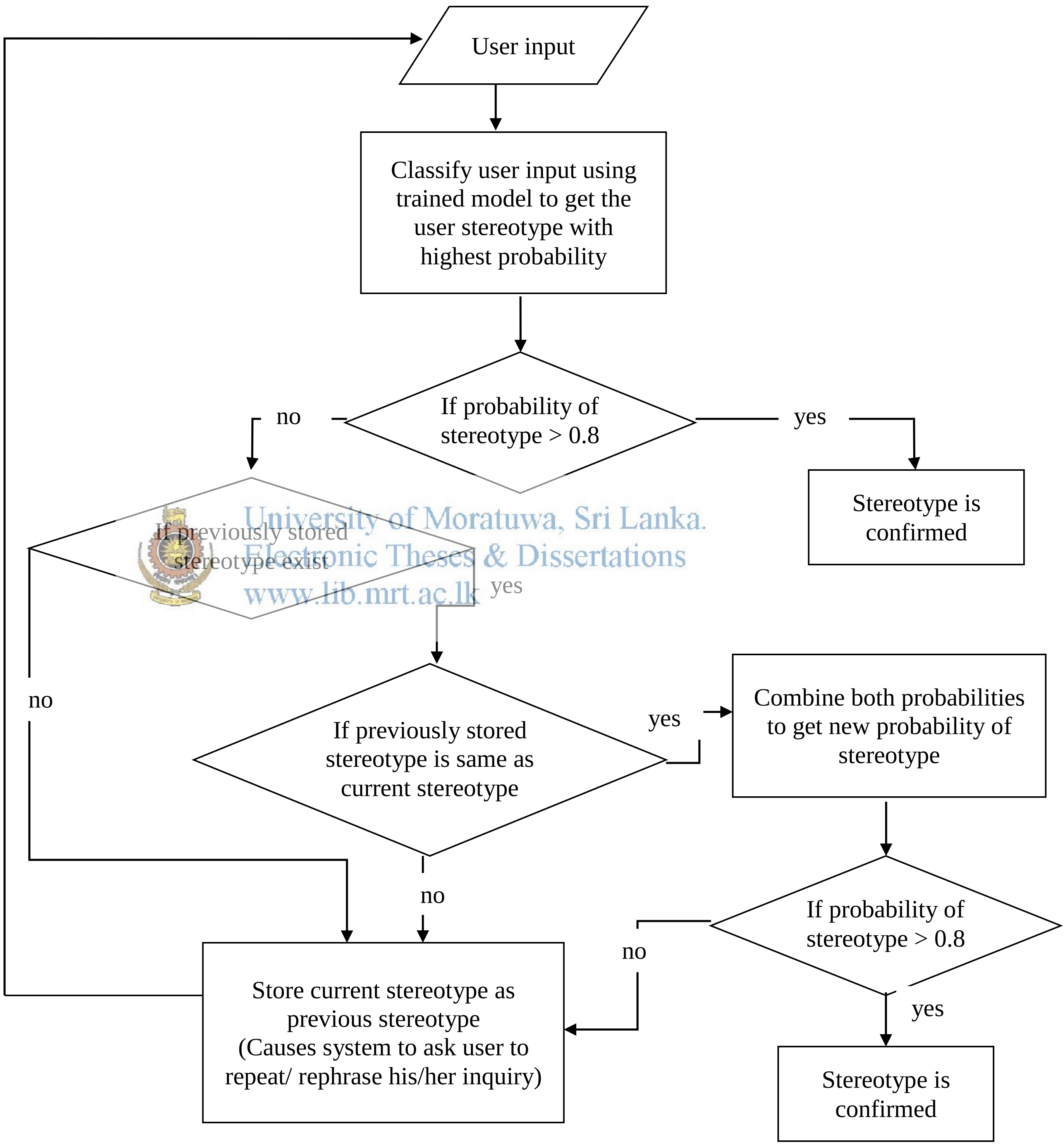
[32] P. Johansson, "User modeling in dialogue systems," Department of Computer and Information Science, Link¨oping university, Link¨oping, Sweden, 2002.

[33] P. Lison, "Structured probabilistic modeling for dialogue management," Ph.D. dissertation, Department of Informatics, Faculty of mathematics and natural sciences, University of Oslo, Oslo, Norway, 2014.

[34] I. Zukerman and D. Litman, "Natural language processing and user modeling: Synergies and limitations," *User modeling and user-adapted interaction*, vol. 11, no, 1-2, pp. 129-158, 2001.

APPENDICES

Appendix A



Appendix B

Dialogue Domain Specification for basic dialogue system

```
<?xmlversion="1.0"encoding="UTF-8"?>

<!-- domain for new policy taking --->
<domain>
        <initialstate>
                <variableid="current_question">
                        <value>howHelp</value>
                </variable>
        </initialstate>

        <!-- Action selection -->
        <modeltrigger="current_question">
                <rule>
                        <case>
                                <effect>
                                        <setvar="a_m"value="Ask({current_question})"/>
                                </effect>
                        </case>
                </rule>
        </model>

        <modeltrigger="u_u">
        <!-- rule to identify user action -->
        <rule>
        <case>
        <condition>
        <ifvar="u_u"value="1.0"/>
        </condition>
        <effect>
                                <setvar="a_u"value="conclude"/>
                        </effect>
                </case>
        <case>
        <condition>
        <ifvar="current_question"value="howHelp"/>
        <or>
                                <ifvar="u_u"value="for * son"relation="in"/>
                                <ifvar="u_u"value="for * daughter"relation="in"/>
                                </or>
        </condition>
        <effect>
                                <setvar="a_u"value="wantPolicyForChild"/>
                        </effect>
        </case>
        <case>
        <condition>
        <ifvar="current_question"value="howHelp"/>
        <or>
```

```
                              <ifvar="u_u"value="want to take another *
                              policy"relation="in"/>
                             <ifvar="u_u"value="want to take * for myself"relation="in"/>
                    </or>
</condition>
<effect>
                    <setvar="a_u"value="wantPolicyForSelf"/>
          </effect>
</case>
<case>
<condition>
<ifvar="current_question"value="howHelp"/>
<or>
                             <ifvar="u_u"value="* friend* wants a *
                             policy"relation="in"/>
                             <ifvar="u_u"value="* relative* wants a *
                             policy"relation="in"/>
                    </or>
</condition>
<effect>
                    <setvar="a_u"value="wantPolicyForOther"/>
          </effect>
</case>
<case>
<condition>
<ifvar="current_question"value="howHelp"/>
<or>
                             <ifvar="u_u"value="about * policies"relation="in"/>
                             <ifvar="u_u"value="what are * policies"relation="in"/>
                    </or>
</condition>
<effect>
                    <setvar="a_u"value="wantPolicyInfoGeneral"/>
          </effect>
</case>
<case>
<condition>
          <ifvar="current_question"value="howHelp"/>
</condition>
<effect>
                    <setvar="a_u"value="None"/>
          </effect>
          </case>
     <case>
<condition>
<ifvar="current_question"value="answerGood"/>
<or>
                    <ifvar="u_u"value="yes"/>
                    <ifvar="u_u"value="ya"relation="in"/>
                    </or>
</condition>
<effect>
```

```
                    <setvar="a_u"value="conclude"/>
                </effect>
            </case>
            <case>
<condition>
<ifvar="current_question"value="answerGood"/>
<or>
                    <ifvar="u_u"value="no"relation="in"/>
                    <ifvar="u_u"value="nope"relation="in"/>
                    </or>
</condition>
<effect>
                    <setvar="a_u"value="startOver"/>
                </effect>
            </case>
            <case>
<condition>
<ifvar="current_question"value="answerGood"/>
</condition>
<effect>
                    <setvar="a_u"value="None"/>
                </effect>
            </case>
<case>
                <effect>
                    <setvar="a_u"value="None"/>
                </effect>
            </case>
</rule>
</model>

<modeltrigger="a_u">
<!-- Rule to determine system action -->
<rule>
<case>
<condition>
<ifvar="a_u"value="wantPolicyForChild"/>
</condition>
<effect>
<setvar="a_m"value="givePolicyInfoChild"/>
</effect>
</case>
<case>
<condition>
<ifvar="a_u"value="wantPolicyForSelf"/>
</condition>
<effect>
<setvar="a_m"value="givePolicyInfoSelf"/>
</effect>
</case>
<case>
<condition>
```

```
<ifvar="a_u"value="wantPolicyForOther"/>
</condition>
<effect>
<setvar="a_m"value="givePolicyInfoOther"/>
</effect>
</case>
<case>
<condition>
<ifvar="a_u"value="wantPolicyInfoGeneral"/>
</condition>
<effect>
<setvar="a_m"value="givePolicyInfoGeneral"/>
</effect>
</case>
<case>
<condition>
<ifvar="a_u"value="conclude"/>
</condition>
<effect>
<setvar="a_m"value="conclude"/>
</effect>
</case>
<case>
<condition>
<ifvar="a_u"value="startOver"/>
</condition>
<effect>
<setvar="a_m"value="startOver"/>
</effect>
</case>
<case>
<effect>
<setvar="a_m"value="askRepeat"/>
</effect>
</case>
</rule>
</model>

<modeltrigger="a_m">
        <!-- Rule to change the current question when a given slot is confirmed -->
        <rule>
                <case>
                        <conditionoperator="or">
                        <ifvar="a_m"value="givePolicyInfoGeneral"/>
                                <ifvar="a_m"value="givePolicyInfoChild"/>
                                <ifvar="a_m"value="givePolicyInfoSelf"/>
                                <ifvar="a_m"value="givePolicyInfoOther"/>
                        </condition>
                        <effect>
                                <setvar="current_question"value="answerGood"/>
                        </effect>
                </case>
```

```
        <case>
                <condition>
                        <ifvar="a_m"value="startOver"/>
                </condition>
                <effect>
                        <setvar="current_question"value="howHelp"/>
                </effect>
        </case>
        <case>
                <condition>
                        <ifvar="a_m"value="conclude"/>
                </condition>
                <effect>
                        <setvar="current_question"value="howHelp"/>
                </effect>
        </case>
</rule>

<!--  basic NLG model -->
<rule>
        <case>
                <condition>
                        <ifvar="a_m"value="Ask(howHelp)"/>
                </condition>
                <effectutil="1">
                        <setvar="u_m"value="How may I help you (Enter 1 to
                        finish)?"/>
                </effect>
        </case>
        <case>
                <condition>
                        <ifvar="a_m"value="Ask(answerGood)"/>
                </condition>
                <effectutil="1">
                        <setvar="u_m"value="Are you satisfied with above
                        information? "/>
                </effect>
        </case>
        <case>
                <condition>
                        <ifvar="a_m"value="conclude"/>
                </condition>
                <effectutil="1">
                        <setvar="u_m"
                                value="Thank you for contacting us. Goodbye!"/>
                </effect>
        </case>
        <case>
                <condition>
                        <ifvar="a_m"value="askRepeat"/>
                </condition>
                <effectutil="1">
```

```
                <setvar="u_m"value="Sorry, Could you please repeat?"/>
            </effect>
        </case>
        <case>
            <condition>
                <ifvar="a_m"value="givePolicyInfoChild"/>
            </condition>
            <effectutil="1">
                <setvar="u_m"value="Currently, we are providing 'Ran
                Kekulu', 'SubaPathum' and 'YowunSaviya' child insurance
                policies each designed for different age groups. They give
                maximum protection for children by including special health
                covers and other benefits. "/>

            </effect>
        </case>
        <case>
            <condition>
                <ifvar="a_m"value="givePolicyInfoSelf"/>
            </condition>
            <effectutil="1">
                <setvar="u_m"value="Currently, we have introduced
                'saubagya', 'arogya' and 'wardhana' life insurance policies
                that you can choose from. Each policy is designed to provide
                you with special benefits according to your requirements"/>
            </effect>
        </case>
        <case>
            <condition>
                <ifvar="a_m"value="givePolicyInfoOther"/>
            </condition>
            <effectutil="1">
                <setvar="u_m"value="There are three types of general life
                policies available."/>
            </effect>
        </case>
        <case>
            <condition>
                <ifvar="a_m"value="givePolicyInfoGeneral"/>
            </condition>
            <effectutil="1">
                <setvar="u_m"value="There are various types of life policies
                available."/>
            </effect>
        </case>
    </rule>
  </model>
</domain>
```