Allen, J., Ferguson, G., et al. (2006). Chester: Towards a Personal Medication Advisor.
J. Biomedical Informatics 39(5):500-513. Elsevier.
doi:10.1016/j.jbi.2006.02.004 http://elsevier.com/locate/yjbin

# Chester: Towards a Personal Medication Advisor [1]

James Allen [a], George Ferguson [a], Nate Blaylock [b], Donna Byron [c], Nathanael Chambers [d], Myroslava Dzikovska [e], Lucian Galescu [d], Mary Swift [a]

[a] *Department of Computer Science, University of Rochester, Rochester, NY, USA*

[b] *Department of Computational Linguistics, Saarland University, Saarbrücken, Germany*

[c] *Department of Computer Science and Engineering, Ohio State University, Columbus, OH, USA*

[d] *Institute for Human-Machine Cognition, Pensacola, FL, USA*

[e] *Human Communication Research Centre, School of Informatics, University of Edinburgh, Edinburgh, Scotland, UK*

**Abstract**

Dialogue systems for health communication hold out the promise of providing intelligent assistance to patients through natural interfaces that require no training to use. But in order to make the development of such systems cost effective, we must be able to use generic techniques and components which are then specialized as needed to the specific health problem and patient population. In this paper, we describe Chester, a prototype intelligent assistant that interacts with its user via conversational natural spoken language to provide them with information and advice regarding their prescribed medications. Chester builds on our prior experience constructing conversational assistants in other domains. The emphasis of this paper is on the portability of our generic spoken dialogue technology, and presents a case study of the application of these techniques to the development of a dialogue system for health communication.

[1] This material is based upon work supported by Dept. of Education (GAANN) grant no. P200A000306; ONR research grant no. N00014-01-1-1015; DARPA research grant no. F30602-98-2-0133; NSF grant no. EIA-0080124; and a grant from the W. M. Keck Foundation. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the above-mentioned organizations.

## 1 Background and Motivation

Health care is rapidly becoming more elaborate and more expensive. Greater emphasis is being placed on having patients manage their own care in their own homes. Patients need to be able to do such things as manage a complex prescription regimen or use in-home devices to monitor and evaluate their condition. Of course, this is a burden for many patients, who generally have trouble following complex instructions and cannot always interpret those instructions in the changing circumstances of everyday life.

Evidence of these problems is revealed in studies such as those that have shown that regular phone contact by a nurse practitioner significantly improves the quality of life in congestive heart failure patients [1]. Such in-home monitoring by nurse practitioners, however, is not feasible in general: there are simply not enough medical personnel to meet the need, and it would be prohibitively expensive. The long-term goal of this project is to develop automated in-home conversational assistants that can help patients manage their treatment and assist in monitoring their health. The specific project we describe here is Chester, a prototype spoken dialogue system that helps a patient manage their prescriptions.

Devices are coming on to the marketplace that attempt to address this problem. For example, voice prescription labels provide audible label information to help the elderly and the visually or cognitively impaired take their medication correctly. Other devices for in-home monitoring of vital signs and conditions range from glucose meters to cardiac health monitors to a variety of wearable or even implantable sensors.

Such technologies provide some help but fall short in significant ways. In particular, they don't help with the quite common exceptions that happen on a frequent basis. For instance, a talking pill bottle can remind you to take a pill, but can't help you if were out of the house, missed a dose, and don't know whether to take a double dose now or not. A study of voice prescription label technology found that while visually impaired participants found the audible voice labels useful and convenient, they indicated that one improvement would be to support understanding why one was taking a specific medication [2].

Of course, this information is usually available, either on the prescription information sheet (typically lost and printed in type too small for many to read) or online (where the volume of data and the requisite computer skills are often overwhelming). Our solution is to provide patients with a *conversational assistant* that supports access to such information using natural, spoken language. This technology is ideally-suited to home health environments for several reasons. First, it is natural for people, requiring no training in its use. It is efficient,

since it exploits humans' highly developed, built-in mechanisms for spoken interaction. It is also expressive, in that one can express complex, novel ideas using the standard mechanisms of language (and understand same). Spoken conversation is also hands-free, which is very desirable for systems designed to be unobtrusive assistants, for example in a home environment.

For such an approach to be viable, however, we need to be able support *robust* spoken language understanding in order to provide reliable information. There are several aspects to this issue. First, note that the system we describe helps patients and/or their caregivers manage their medical care by providing reminders, answering questions, and engaging in dialogue to collect information for monitoring a patient's current state. The system does not make medical decisions, but rather helps patients follow the instructions that they have been given by their doctors, and provides status reports back to medical support teams. Second, the system provides mechanisms so that the patient can have confidence that the system has correctly understood their questions. One obvious mechanism we use is that the system displays what it heard on the screen for the patient to see. Another would be to always provide a fallback to a human caregiver, for example by pressing "0" on a telephone interface or clicking a button on a screen. Third, the system does not present information that is not explicitly in the patient's instructions from the doctor, or information provided about their medications. For example, if the system says "you can take aspirin," it is because this is explicitly in the doctor's instructions, rather than being a decision that the system makes on its own. And fourth, the system is designed to require a high degree of confidence that it has understood the user's question in order to proceed without any clarification or confirmation from the patient. We believe that these mechanisms can provide the effectiveness of spoken dialogue interaction without excessive risk.

Finally, for such systems to be cost effective, we need *portable* dialogue technology, so that the cost and time to develop a system for a new application is only an incremental cost in adapting generic technology. The TRIPS project is aimed at developing such generic technology, and prototype systems have been developed in a wide range of applications. This paper presents a case study of the issues we faced in adapting our generic system to create a prototype system that helps patients understand and manage their prescribed medications.

### *1.1 The Prescription Compliance Problem*

Several factors are converging to create a major new health care problem in prescription compliance. First, there is the explosion in the number of pharmaceutical therapies and the rate at which they are being prescribed. For

example, over 3.5 billion prescriptions were dispensed in the United States alone in 2004 [3]. This is leading to more patients taking more pills. Second, because of the increased specialization of medical practice and the way patients are assigned to specialists, these prescriptions tend to be assigned by different providers at different times. This can result in complicated (and possibly even unsafe) combinations of prescriptions in a single patient's regimen. Finally, the problem is compounded by the fact that our population is aging. Older patients are both more likely to have multiple conditions that require separate medications, and are often more likely to have the types of conditions that make keeping track of these more challenging.

These trends are combining to leave patients with unwieldy prescription regimens involving multiple medications, each with its own characteristics and requirements. Very quickly, patients are required to spend an inordinate amount of time figuring out what to take and when to take it. Or, worse, they simply give up and take the medication incorrectly (or stop altogether).

The statistics show that non-compliance is a serious problem. One study showed that non-compliance causes 125,000 deaths annually in the U.S. [4]. A review of the literature based on electronic monitoring of drug consumption concluded that mean dose-taking compliance ranged from an already poor 71% (+/- 17%) for once-per-day drugs to just 51% (+/- 20%) for drugs meant to be taken four times per day [5]. The New York Times has labeled non-compliance "the world's other drug problem" [6]. Whatever the reason, it is clear that a significant percentage of patients simply do not (or cannot) comply with their prescribed drug regimen.

### *1.2 Conversational Systems*

Conversational systems enable naive users to interact with complex applications in a natural way using speech, using human dialogue as a model [7]. Such systems integrate many different technologies that have been developed individually: speech recognition, language understanding, planning and reasoning, language generation, text-to-speech synthesis. As these underlying technologies as well as dialogue technology itself have matured, research systems have been developed in numerous domains, while successful systems of more limited flexibility have started to be deployed commercially. At the same time, the focus of dialogue systems research has evolved to include issues of portability, modularity and dynamic configurability [8].

Within this paradigm, we have for some years been developing *conversational assistants*: computer systems that interact with their users using spoken natural language in order to help them solve problems. Starting from corpus

studies gathered using a "Wizard of Oz" setup [9,10], we have built a sequence of prototype collaborative planning systems [11–13]. More recently, we have been extending this work to several new domains including an assistant to a supervisor in a crisis management center [14], a purchasing assistant, an assistant for managing teams of robots engaged in search and rescue[15], and the Medication Advisor described in this paper. A key aspect of our research is that a single set of generic spoken dialogue software components should be used to handle all of the different applications. Our main motivation for this is that we believe it will not be economically feasible to build such complex systems from scratch for each application.

From a technical perspective, the medication compliance problem seems like a relatively circumscribed domain in which it might be possible to develop an intelligent computer assistant. The major problems involve maintaining the patient's medication schedule (notifying them as appropriate, helping with rescheduling, *etc.*) and answering questions about the schedule and the drugs involved. This paper describes the problems we faced as we built an initial prototype of a system that assist patients in their own home using intuitive and natural communication mechanisms.

## 2 The TRIPS Environment

Our initial prototype of the Medication Advisor is based on the architecture developed for TRIPS, The Rochester Interactive Planning System [16], shown in Figure 1. Components are shown as ovals, and knowledge sources as square boxes. Generic parts are shown with white backgrounds, and the domain specific parts are shown in gray. Broadly speaking, the components are divided among three main categories: Interpretation, Behavior, and Generation, and the knowledge sources often are shared across several categories. The interpretation components involve understanding what the user has said or done, the behavior components manage the systems goals and reasoning, and the generation components construct system contributions to the dialogue. We consider each briefly.

Interpretation: The first stage of interpretation is speech recognition, which draws its vocabulary from the common lexicon for the system, and produces a list of possible interpretations for processing by the Parser. The Parser uses a general lexicon and grammar of spoken dialogue utterances (developed incrementally over several previous applications), and produces a meaning representation expressed in a domain independent semantic representation (the logical form). The interpretation manager receives the parser output and performs contextual interpretation. It resolves referring expressions such as pronouns and definite noun phrases and interacts with the Task Manager which per-

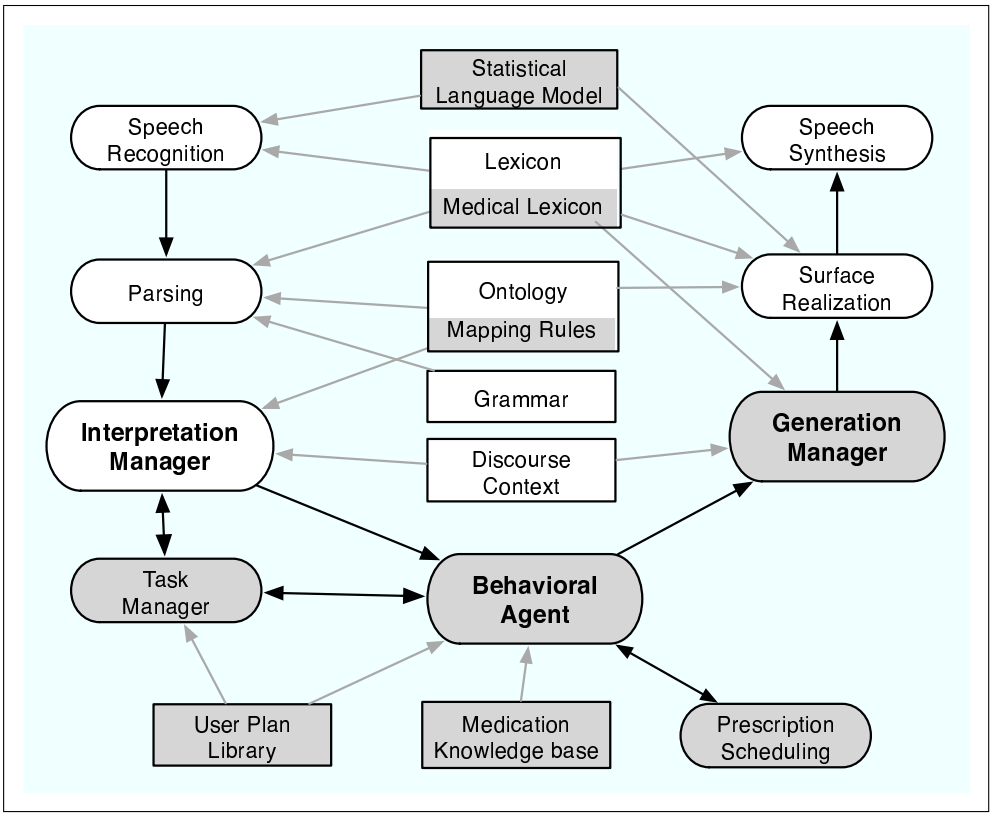


Fig. 1. Chester in the TRIPS Architecture with Domain Specific Aspects in Gray

forms plan and intention recognition. It also applies a set of ontology mapping rules in order to convert the generic semantic representation into the domain specific representation for Chester. It identifies the most likely intended speech act (*c.f.* [17]) in the form of a collaborative problem solving action, and updates the Discourse Context. A more detailed analysis of the problem-solving model, including an example in the Medication Advisor domain, is presented in [18].

Behavior: The Behavioral Agent is the autonomous "heart" of the agent. It takes the collaborative problem solving act produced by the interpretation manager and performs further intention recognition and then plans system behavior based on its own goals and obligations, the user's utterances and actions, and changes in the world state. Actions that require task- and domain-dependent processing are performed by the Task Manager. Actions that involve communication and collaboration with the user are sent to the Generation Manager in the form of communicative acts. Chester contains several domain-specific components, including a medication knowledge base and a rudimentary scheduling algorithm to reason about prescriptions. These domain specific aspects of the system formed the bulk of the work we had to perform in order to construct the system.

Generation: The Generation Manager coordinates planning of the system re-

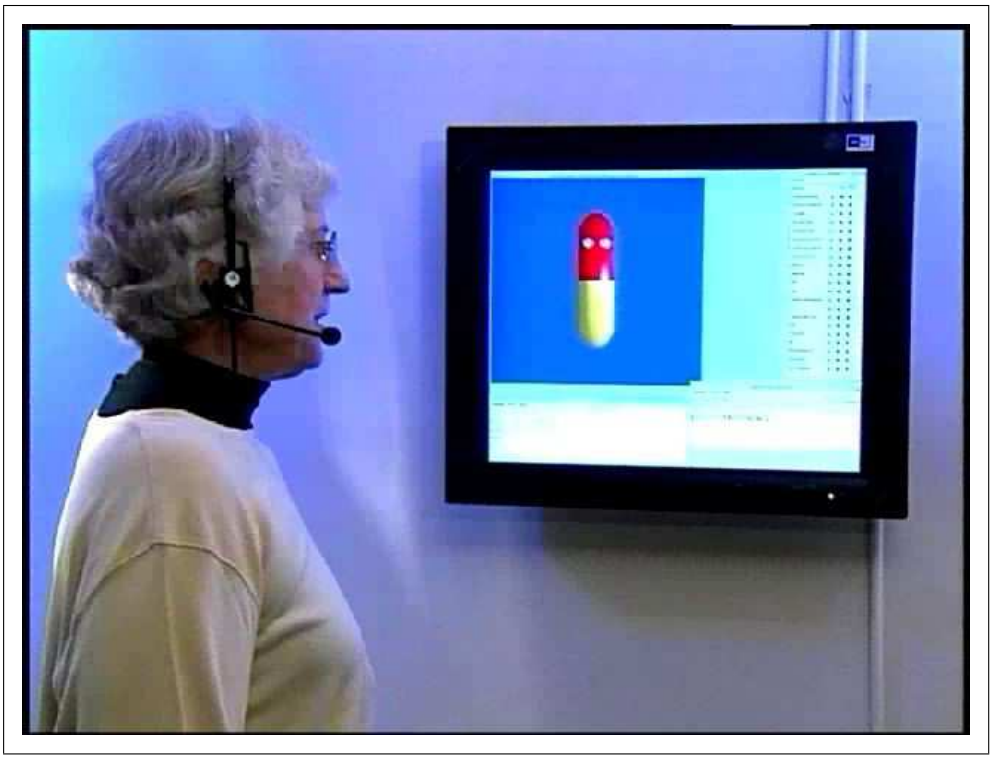

Fig. 2. Screenshot of user talking to Medication Advisor

sponses. Generation is typically divided into content planning, where the content of the system's utterance is planned, and surface realization, where the actual words to be spoken are constructed. Its behavior is driven by discourse obligations (stored in the Discourse Context), and the directives it receives from the Behavioral Agent. The glue between the layers is an abstract model of problem solving in which both user and system contributions to the collaborative task can be expressed. While content planning is domain-specific in TRIPS, the surface realization is performed by an independent surface realization component [19].

### *2.1 System Demonstration*

The Medication Advisor prototype appears to the user as shown in Figure 2. In addition to displaying the system's understanding of the user's speech and the conversation thus far (at the bottom of the screen), the system is embodied as a graphical avatar in the shape of a pill. This could perform a few actions to convey the system's state, for example, idle, attentive, thinking, and speaking. The thinking and attentive states proved most valuable in indicating when Chester required extra processing time and expected the user to wait before moving on. While the actions and visualization of the avatar were very simple, they provided invaluable information and a natural focus of attention for the users.

| | | |
|---|---|---|
| Patient: | Can I take an aspirin? | (1) |
| Chester: | No, you are taking Celebrex and Celebrex interacts with aspirin. | (2) |
| Patient: | Oh. Could Celebrex cause headaches? | (3) |
| Chester: | No, headaches are not an expected side-effect of Celebrex. | (4) |
| Chester: | Do you have a headache? | (5) |
| Patient: | Yes. | (6) |
| Chester: | How severe is your headache on a scale of one to five. | (7) |
| Patient: | One. | (8) |
| Chester: | Your doctor says you can take Tylenol for mild headaches. | (9) |
| Patient: | Ok, I'll try that. | (10) |
| | Why am I taking Celebrex? | (11) |
| Chester: | You are taking Celebrex to help with your arthritis. | (12) |
| Patient: | Should I take one now? | (13) |
| Chester: | Yes, you are taking Celebrex every morning. | (14) |
| Patient: | Is there anything else I need to take now. | (15) |
| Chester: | Yes, you should take your Prinivil. | (16) |
| Patient: | Which one is that? | (17) |
| Chester: | It's the yellow one. [shows picture] | (18) |
| Patient: | Could that be causing my headaches? | (19) |
| Chester: | Yes, headaches are an expected side-effect of Prinivil. | (20) |
| Chester: | Tylenol should help. | (21) |
| Patient: | Ok, thanks. | (22) |
| Chester: | You're very welcome. | (23) |

Fig. 3. Demonstration dialogue for Chester

Figure 3 shows a dialogue which we often use when demonstrating the system. This dialogue, designed in collaboration with our medical collaborators, was created to showcase the capabilities of the system and the potential for addressing the compliance problem. The user is an elderly woman, who is taking a small number of commonly prescribed medications. She wakes up with a headache and decides to ask Chester for help.

The user starts by asking a simple question:

Patient: Can I take an aspirin? (1)

The system interprets this as either a yes-no question about the ability to take aspirin, or as a suggestion to take aspirin (now, *i.e.*, at the time of the utterance). In either case, the system must first verify whether taking aspirin is possible. Note that it must identify that the user is asking about whether it is advisable given their medical situation, not whether they could physically swallow a pill. All of these decisions require the capability to recognize the intentions underlying utterances (see Section 3.4). Once the correct intention is identified, Chester cross-references its knowledge about the user's prescriptions with its knowledge base of drug interactions and side-effects. Since this search turns up an interaction with one of the prescribed medications, the system can immediately answer that taking aspirin is not permissible:

Chester: No, you are taking Celebrex and Celebrex interacts with aspirin. (2)

The user goes on to ask another question:

Patient: Oh. Could Celebrex cause headaches? (3)

At first glance, this is yet another yes-no question. The system could simply look up the answer using its knowledge bases and tell the user the answer. However, it is here that we begin to see the difference between a simple natural language front-end to a database query system and a true conversational assistant. As part of interpreting the user's utterance, the system again attempts to recognize the intention behind the utterance—that is, why she said what she did. In this case, the question about headaches adds evidence to the hypothesis that the user is trying to treat a symptom (headache) with an over-the-counter remedy (aspirin). This hypothesis regarding the user's intention was initially created during the processing of utterance (1), but at that time there was not yet sufficient evidence for the system to take action based on it. With the additional evidence from (4), the system is reasonably confident that the user has a headache that they are trying to treat. As part of its response to this type of user action, it consults its knowledge bases to determine whether such a situation requires action on its part. It turns out that, in this scenario, one of the other medications that the patient is prescribed for can result in severe headaches, and that if these occur, it is a serious condition that should be reported to the doctor immediately.

So, to summarize, the literal answer to the question about Celebrex causing headaches is no. However, the system decides to take the initiative in the conversation and resolve the issue about the user's headache, since it is potentially serious. The result is the following utterances:

Chester: No, headaches are not an expected side-effect of Celebrex. (4)
Chester: Do you have a headache? (5)

After the user answers yes, Chester seeks to clarify the severity of the headache.

Chester: How severe is your headache on a scale of one to five? (7)
Patient: One. (8)
Chester: Your doctor says you can take Tylenol for mild headaches. (9)
Patient: Ok, I'll try that. (10)

The form of the question in (7) was suggested to us by our medical colleagues. The information used in utterance (9) comes from explicit instructions from the doctor associated with the prescription (*i.e.*, if the headache had been severe, Chester would have recommended the the patient call her doctor immediately).

With the headache issue resolved, the system returns the initiative in the conversation to the user, who decides to pursue her original line of questioning regarding Celebrex. The remainder of the dialogue illustrates the sorts of questions that might reasonably arise in a Medication Advisor context.

| | |
|---|---|
| Patient: Why am I taking Celebrex? | (11) |
| Chester: You are taking Celebrex to help with your arthritis. | (12) |

The reason for taking a medication is an important aspect of the extended notion of "prescription" required to support this kind of dialogue.

Another capability that we wanted to emphasize is the system's ability to reason about when the medications are supposed to be taken (as opposed to the "why" emphasized thus far in the dialogue). This is an ideal problem for AI-based scheduling technology to help people by giving reminders and answering questions about when they should take their pills (see Section 4.5). So, for example, the user continues:

| | |
|---|---|
| Patient: Should I take one now? | (13) |
| Chester: Yes, you are taking Celebrex every morning. | (14) |
| Patient: Is there anything else I need to take now? | (15) |
| Chester: Yes, you should take your Prinivil. | (16) |

Th user continues with a request to help identify the pill, and Chester responds by showing a picture from its medication knowledge base.

| | |
|---|---|
| Patient: Which one is that? | (17) |
| Chester: It's the yellow one. [shows picture] | (18) |

In the rest of the dialogue, the user follows up on her earlier headache question and then completes the dialogue.

Although the system is fully implemented, it is far from a robust, complete product suitable for fielding in real situations. Specifically, its knowledge base contains information on only 10 medications, it has no user adaptation in its speech recognition that would be required for high-accuracy recognition, it does not have a full range of common user goals that would be typical in real use of the system, and attaining sufficient linguistic coverage will require further extensions to the lexicon, grammar and ontology. It does, however, provide a demonstration of potential feasibility of such technology, and we are currently working to resolve the problems above to produce a system sufficiently capable to support controlled experiments in usability and medical effectiveness.

In the rest of this paper, we describe the technical challenges we faced in building this initial version of Chester.

## 3 Customization of the Interpretation Processes

One of the challenges facing developers of a dialogue system in new domains is building the various models of language needed for effective processing. The typical way to address this problem these days is by collecting a corpus of training data. But this is not a workable strategy for most dialogue applications. It is typically very expensive to collect and annotate large amounts of dialogue relevant to a system that doesn't exist yet. We have been pursuing a different approach that involves using generic language models, including statistical n-gram models (*c.f.* [20–22]) trained from a variety of other dialogue domains, a generic lexicon that at the time covered approximately three thousand of the common core English words (lemmas; standard morphological variations are generated automatically in the lexicon), and a comprehensive grammar of spoken dialogue that was and still is under continual development using data from multiple domains. In addition, we have generic components for discourse processing [23], including speech act identification [24,25], reference resolution [26,27] and ellipsis processing (*c.f.* [28,29]). This section describes what we needed to do to adapt the generic components to produce an effective interpretation system for Chester.

### *3.1 Language and Pronunciation Modeling*

Two important components of a speech recognition system are the language model and the pronunciation dictionary (also called lexical model). In our previous work on spoken dialogue applications, we devised a technique for quickly building language models for new task domains that has delivered very good performance [30]. Pronunciations were drawn from the 120,000-word general-purpose CMU dictionary [31], which provided satisfactory coverage for our applications. Unfortunately, for domains like the Medication Advisor, this may not always be possible, as general purpose dictionaries by their nature tend to have limited coverage of specialized lexicons, whereas domain-specific dictionaries may not even exist.

The seriousness of the problem is compounded by the fact that, even if a canonical pronunciation can be obtained, it is not safe to assume that it will be known and used by uninformed users of the system. In previous research it has been shown that adding non-canonical pronunciations in the lexical model helps improve the speech recognizer's performance [32]. We expect that the same will be true in the Medication Advisor domain. Moreover, we propose that these non-canonical pronunciations – as well as pronunciations for out-of-dictionary words – can be generated automatically.

| | |
|---|---|
| In dictionary | AA S T IY AA P ER OW S IH S |
| Automatic (1) | AA S T IY AA P ER OW S AH S |
| | AO S T IY AA P ER OW S IH S |
| | AO S T IY AA P ER OW S AH S |
| Automatic (2) | AO S T IY OW P AH R OW S IH S |
| | AA S T IY OW P AH R OW S AH S |

Fig. 4. Alternative pronunciations obtained automatically for the word OSTEOPOROSIS.

For illustration, in Figure 4, under the heading "Automatic (1)", we give examples of pronunciations for a medical term in our lexicon, obtained automatically with a domain-independent grapheme-to-phoneme converter [33]. We expect that general grapho-phonotactic knowledge gleaned out of a general purpose dictionary will be suboptimal for the medical domain. One way to improve the quality of our models is to learn automatically domain-specific constraints from medical dictionaries that give guides to pronunciation (*e.g.*, [34]). For example, the alternative pronunciations listed in Figure 4 under the heading "Alternative (2)" are obtained by running the same converter on the string "OSS-tee-oh-pa-ROW-sis", which is the pronunciation indicated by the National Cancer Institute on their website.

Additional improvements may be obtained by associating probabilities to alternative pronunciations and adapting them dynamically; this way we can personalize the pronunciation dictionary to specific users and correct the possible errors introduced by the grapheme-to-phoneme converter.

### *3.2 Adapting the Lexicon and Ontology*

The TRIPS grammar and lexicon provide broad coverage for a variety of constructions common in practical dialogue including questions, imperatives, and fragmentary utterances such as noun or prepositional phrase fragments given as answers to questions. To handle dialogue in the Medication Advisor domain, we augmented our lexicon with names for medications, symptoms, and medical conditions, as well as with previously missing domain-independent senses for existing words, such as a `Consume` sense of *take* to handle utterances such as "Can I take an aspirin", and similarly for *have*, as in "I like to have an antacid before bedtime".

The domain-independent semantic representation (a logical form) produced by the parser is designed to be suitable for discourse processing, and it is used by components handling reference resolution and dialogue management as input for further processing. The logical form representation is based on

a domain-independent language ontology. The reasoning components use a domain-specific ontology optimized for reasoning in the medication domain, using specialized concepts as described in Section 4. To connect the two, we use a set of mappings between the ontologies and a transform engine that converts the syntax of the logical form to the domain-specific syntax used by the reasoners. These mappings tailor the generic logical form to the needs of the reasoning components built for a particular domain.

The ontology mapping facility also allows us to improve parsing speed and accuracy on in-domain utterances. A specialization algorithm automatically propagates domain-specific selectional restrictions into the lexicon for entries that have domain-specific mappings, tightening the constraints on words relevant to the domain. Word senses with domain-specific mappings are preferred, allowing the parser to find a correct interpretation faster. This is especially helpful for words such as *take*, which have a large number of senses, resulting in a high degree of ambiguity. In the Medication Advisor domain, the `Consume` sense of *take* as in (1) is strongly preferred and will be considered first, as opposed to the `Transport` sense of *take*. Our lexicon specialization significantly boosts parsing speed and accuracy compared to the generic parser, as discussed in Section 5.1.

### *3.3 Interpretation and Reference Resolution*

We found two key challenges for discourse interpretation in the Medication Advisor domain. One arose from an increased occurrence of reference to kinds rather than individuals or sets. Medicinal substances are of course frequently mentioned in this domain, and often they must be interpreted as kinds. For example, in utterance (2) above, "Why am I taking Celebrex?" the word *Celebrex* refers to the kind, not to any particular instance of Celebrex. Interpreting references to kinds required us to extend to the grammar, the logical form language, and the reference resolution algorithms.

The second interpretation challenge in this domain was that the language used to speak about medical conditions and treatments abounds with metonymy (mentioning one item in a conventional way to refer to a related item). For example, in "I need to know when to take my prescription," *my prescription* is a metonymic reference to the medication associated with the speaker's prescription (since one can't literally take a prescription). This metonymic phrasing must be coerced back to the correct argument type for reasoning. The back-end reasoners do not know how to place a `PRESCRIPTION` object in a `TAKE-MEDICINE` event, and therefore the dosing schedule that the user is requesting cannot be created. Rather, a `MEDICATION` object is expected by the back-end planner. This situation creates an asymmetry between the represen-

tations built for the language-processing front-end and the back-end reasoning system. We want to allow the parser to accept "take my prescription" as an acceptable sentence, even though it violates the argument restrictions in place in the back-end. We cannot solve this problem by removing all restrictions from the parser, because we still want it to reject truly unacceptable combinations, such as "*I need to know when to take my shoe."

Our solution was to create a set of inference rules that specify the relationships between certain object types that are commonly used in metonymic constructions in this domain. Using the inference rules, the interpretation of "my prescription" in "I need to know when to take my prescription," is transformed into "the medication of my prescription." The inference rule adds both the relationship (*medication-of*) and the additional hidden argument to the interpretation of the sentence, so that the reasoner in charge of building the dosing schedule receives a `MEDICATION` argument as expected.

### *3.4 Plan and Intention Recognition*

A vital step of dialogue understanding in the Medication Advisor domain is intention recognition. Intention and plan recognition go beyond *what* was said and look at *why* it was said. What did the user hope to accomplish by uttering this? What prompted him to utter it?

Intention recognition is critical since similar language patterns can be based on different underlying user goals and need to be handled in very different ways. Take, for example, the following two utterances, the second of which is from the example dialogue in Figure 3:

(1) "Can I take a Celebrex?"
(2) "Can I take an aspirin?"

While these utterances are identical in structure, the user's *underlying goals* behind these utterances can be very different. Let us assume, for this example, that we know that the user has a prescription for Celebrex, but does not have a prescription for aspirin. Because we know that the user has a prescription for Celebrex, we can infer that the intention behind the first utterance is that he is likely querying about a previously-defined medication schedule. For the second utterance, however, there must be some other motivation for asking about taking the medication. It may turn out, as it does in our sample dialogue, that the user has a headache, and desires to take aspirin to get rid of the symptoms. This intention requires a very different response from the system than that of the first utterance.

To capture the user's intentions, we identify two aspects of intention: the

*collaborative problem solving act* between the user and the system, and the *domain goal/plan*. We consider each of these briefly.

Communicative intentions capture how each agent tries to individually affect the joint problem-solving state. Together, the system and user do things like adopting objectives, formulating and choosing recipes (prototypical plans), and creating and executing solutions. The details of our collaborative problem solving model can be found in [18,35] While the collaborative problem-solving model is domain independent, it utilizes a domain-dependent plan library which provides the system with knowledge of domain-level objectives, recipes, and resources [36].

To obtain the domain knowledge for the Medication Advisor domain, we analyzed a set a questions from an online medication FAQ. Some representative objectives found in the FAQ include evaluating possible drug interactions, taking over-the-counter drugs, managing medication schedules, querying medication schedules, stopping undesirable symptoms, assessing side-effects, and defining medical terminology. A subset of these goals were encoded, together with corresponding recipes, to be used with the plan/intention recognition system.

We analyze here, the first few exchanges of the sample dialogue in Figure 3 in order to show how plan and intention recognition proceed in the system. We reprint part of the dialogue here for the reader's convenience.

| | |
|---|---|
| Patient: Can I take an aspirin? | (1) |
| Chester: No, you are taking Celebrex and Celebrex interacts with aspirin. | (2) |
| Patient: Oh. Could Celebrex cause headaches? | (3) |
| Chester: No, headaches are not an expected side-effect of Celebrex. | (4) |
| Do you have a headache? | (5) |

The user starts with utterance (1), which the system must interpret. Intention recognition sees that the user does not have a prescription for aspirin, but that aspirin is an over-the-counter drug. The communicative intention inferred is that the user and system jointly evaluate the domain plan of the user taking an over-the-counter drug, namely aspirin. With this intention now known, the rest of the system (back-end reasoners) determines that it is not a good idea for the user to take aspirin, since she is taking Celebrex and aspirin and Celebrex interact. This is generated and becomes utterance (2) by the system.

The user does not seem to have intended that the system understand *why* he wants to take aspirin. However, through plan recognition, the system realizes that taking an over-the-counter drug is not an end goal, or an objective unto itself (*c.f.* [37]). Rather, it is a means towards a higher-level objective. However, there is not yet enough information for the system to make an inference.

The user then makes utterance (3). Intention recognition recognizes this as an intention to identify some fact in the situation, specifically, if headaches are a side-effect of Celebrex. Back-end reasoners find that this is not the case, resulting in the system's utterance (4).

At the same time, utterance (3) gives plan recognition sufficient information to make a guess about the user's higher-level goals. In the system's plan library, the plan on taking an over-the-counter medication is part of a larger objective to relieve a symptom. The system knows that aspirin can relieve headaches and infers that the user's higher-level objective is possibly relieving a headache. However, the system is not sure about this inference, so it produces utterance (5) to confirm it. Such plan recognition enables the system to be helpful (*c.f.* [38]). Later in the exchange the system, based on this inference, is able to take initiative and suggest that the user take Tylenol for her headache.

Chester is able to perform intention recognition effectively because of the limited number of goals we have encoded. We are currently developing techniques for effective and efficient intention recognition over large plan libraries by combining statistical and symbolic techniques ([39]).

## 4 Building the Reasoning Components

The bulk of the development effort in Chester involved developing the medical knowledge base and the reasoning engines required for the task. Such components clearly differ dramatically from task to task and thus inevitably will remain domain dependent. The goal of our work, however, is that once such components are built for a domain, creating the dialogue system on top of it is relatively low effort. Furthermore, many of the domain-dependent reasoners are built using generic technology originally developed for other domains.

### *4.1 Knowledge Acquisition*

In the Medication Advisor prototype, a crucial component is the system's knowledge base of prescription drugs and over-the-counter medications. The preliminary knowledge base was built using a process that combined automatic knowledge extraction and manual correction of the extracted knowledge.

Our starting point for the knowledge base was the public MedlinePlus Health Information web site [40] provided by the National Library of Medicine and the National Institutes of Health. This site provides reference information for thousands of drugs for both the U.S. and Canada. Since the site was designed

for the average web user, most of the information is described using English sentences and the main task during knowledge acquisition was to extract the relevant information from this English.

We needed to extract three classes of information from each page: substance to brand name mappings, dosing schedules, and substance effects. We used a combination of HTML extraction rules and simple text patterns to extract the brand names and dosing information. Unfortunately, the third type of knowledge, substance effects, is not as simple. Our automated system was helpful in determining when a drug interaction was listed, but pulling the semantics out of the natural English was difficult and left to the human researcher.

### *4.2 Representing Knowledge About Drugs*

One of the main functions of the Medication Advisor knowledge base is to represent drug interactions and their effects. However, classifying drug interactions proved to be a difficult problem. The following examples show the ambiguity that arises from MedlinePlus's descriptions of drug interactions (all taken from its description of "Antihistamines (Systemic)" [41]):

(1) Troleandomycin—Use of these medicines with astemizole or terfenadine may cause heart problems
(2) If you are now taking, or have taken within the past 2 weeks, any of the MAO inhibitors, the side effects of the antihistamines, such as drowsiness and dryness of mouth, may become more severe
(3) Make sure you know how you react to the antihistamine you are taking before you drive, use machines, or do anything else that could be dangerous if you are not alert.
(4) Antihistamines are used to relieve or prevent the symptoms of hay fever and other types of allergy

The richness of these descriptions made their representation a challenging research problem as well as an important application domain.

Example (1) is representative of the most prevalent class of interactions listed in MedlinePlus. It is a drug-to-drug interaction that specifies the substances that interact and what effect results. Obviously, drugs interact with other drugs. However, the temporal constraints on such an interaction are both ambiguous and undefined (contrast with (2)). In the development of the knowledge base we made several assumptions about the correct interpretation of these constraints. This is one of several areas where a more formal oversight process would be required before the system was deployed.

Example (3) shows that we need to represent substance interactions with other "events" instead of just other substances. In this case, the event (or act) of operating machinery should be avoided when taking antihistamines. Finally, example (4) shows a case that is neither drug-to-drug nor drug-to-event, it describes the effects of a substance on a symptom.

Without going into the details, we represented these and a wide range of similar statements using three basic event expressions: *having* a condition, *taking* a substance, and *doing* an activity. A *history-of* operator allows us to describe past events. Thus for example (2) we get:

```
(history-of (taking MAO-inhibitor) (weeks 2))
```

It is worth noting that such qualitative relationships between events are much more frequent in MedLinePlus than any kind of simple metric constraints.

### *4.3 Drug Effects and Interactions*

With all activities represented as events, and a semantics for time periods adequately defined, the next step was to represent the effects of a substance when taken within different contexts. The schematic form of these rules can be paraphrased as: "Under conditions $C$ with timing $T$, the prescription constraint $P$ applies. For each side-effect, the effect $E$ will occur with frequency $F$. Further, if it is observed, then it indicates a problem of severity $S$ and the patient's response should be $R$." The condition $C$ is the drug interaction, such as `(AND (taking MAO-inhibitor) (doing machinery))`. The prescription constraint $P$ has a variety of options, but the most common is `doctors-permission`, meaning "when prescribed by a doctor."

The key representational challenge here was the representation of *effects*. The common example of aspirin would lead us to a conclusion that substances lessen the degree of a side-effect (as aspirin lessens pain). However this is only one in a set of many types of effects. The following are a few examples from the MedlinePlus database that we needed to represent:

(1) Although not all of these side effects may occur, if they do occur they may need medical attention. . . cough; diarrhea; difficulty swallowing; dizziness; fast heartbeat; fever; headache. . .
(2) (acetaminophen taken while having) liver disease—The chance of serious side effects may be increased.
(3) Acetaminophen is used to relieve pain and reduce fever.
(4) Salicylates can make this condition (Gout) worse and can also lessen the effects of some medicines used to treat gout.

Again, without going into the details, we introduced a number of attributes of events, such as *severity-of*, *chance-of*, and *effectiveness-of*. Drug effect (or interaction) rules are then formulated in terms of *increase* or *decrease* of these attributes. Thus the effect in example (2) is:

```
(increase (chance-of (having serious-side-effects)))
```

and that of example (4) is:

```
(decrease (effects-of (taking medicines-used-to-treat-gout)))
```

### *4.4 Prescribed Use and Dosing*

After formalizing the representation for drug interactions and side-effects, the final task was to represent the section of MedlinePlus which describes proper dosing for the drug substances. The schematic form of these rules can be paraphrased as "Do $A$ at rate $R$ with restrictions $X$ for purposes of effect $E$." The action $A$ is typically *taking* a medication, the rate $R$ is something like "1 to 2 mg every eight to twelve hours," and the intended effect $E$ is as described above. Where appropriate, we also encoded an action to be performed when a dose was missed (for example "take now or skip if close to next" or "contact doctor"). These rules form the core knowledge of the system about when and how the patient's medications should be taken.

### *4.5 Medication Scheduling*

The scheduling component of the Medication Advisor is used to generate relatively optimal, safe, and dynamically adaptive schedules given necessary information about the patient's prescriptions. Input to the scheduler consists of two types of information:

(1) Information regarding prescribed medications (name, dosage and rate, *etc.*);
(2) Information describing constraints on taking the medication, including timing constraints (*e.g.*, "take before bed"), activity constraints (*e.g.*, "do not take with food"), and drug interactions, including with over-the-counter drugs.

Additional scheduler input comes in the form of events such as the patient missing a dose.

From this information, the scheduler generates a schedule taking into account all known constraints. If a new prescription is added, the system should adjust its schedule to satisfy the new constraints if needed. In addition, some of the constraints are "soft," such as "It is preferable to take Medicine A at least 3 hours before a meal." The system should be able to find the best schedule which satisfies as many soft constraints as possible. Finally, when some dynamic changes happen, the system should be able to adjust the schedule to adapt to the new conditions. The remainder of this section briefly describes our initial approach to these requirements.

First, we needed a temporal representation that facilitated the generation of a quantitative schedule from a set of mixed qualitative and quantitative constraints. After considering several options, we settled on a quantized representation that divided each day into half-hour units. By quantizing time this way, the number of possibilities the scheduler must consider is dramatically reduced, and the final schedule is easy for people to understand and follow.

An important aspect of the temporal content of prescriptions is that information is often expressed relative to prototypical events, such as breakfast, lunch, dinner, mealtime, bedtime, and so on. For example, "medicine A must be taken within 3 hours of lunch time," "medicine B can be taken 6 hours before bedtime," and so on. To capture this, we chose a constraint-based representation of events. Each primitive constraint relates the time of taking the medication to the prototypical events. We should note this is not sufficient to capture the complete range of dosing information seen in MedlinePlus (and represented in the knowledge base as described above). For example, some medications are prescribed to be taken "as needed for pain." Others involve taking varying amounts depending on other circumstances. In any case, our representation can handle a fairly wide range of dosings, in particular those that are possible to schedule definitively.

Given this representation, it was clear that part of the prescription scheduling problem could be formulated as a constraint-satisfaction problem (as reviewed in [42], for example). Other aspects, however, look more like an optimization problem. For instance, a good prescription regimen would tend to minimize the number of times a day that medication needs to be taken. In this initial implementation, Chester could verify that a certain prescription regimen was consistent with the stated prescriptions, but was not required to generate optimal schedules. We are currently working on developing richer algorithms for prescription scheduling that will be used in the next version of the system.

Finally, another challenge for the scheduling system is that it must be able to adjust the schedule in light of certain changes, such as the patient forgetting to take a medicine at the right time. The appropriate repair strategy is constrained by requirements specified in MedlinePlus, such as:

"Medicine A should be taken at once if missed, unless it is less than 2 hours until the next dose, in which case you should skip the dose."

The original Chester scheduler could verify some changes proposed by the user, but did not solve this problem in any generality. This is also a challenging area for future research (as described, for example, in [43]).

## 5 Component Evaluations

We have performed a number of component evaluations that indicate the plausibility of being able to construct a robust, usable dialogue system from generic components. In this section we briefly summarize some of these component evaluations, and provide pointers to where these results have been presented in more detail.

### *5.1 Evaluating Parsing and Domain Customization*

To assess portability, we evaluated the coverage of our generic grammar on a set of five human-human dialogues in the Medication Advisor domain after adding missing lexical entries, but not extending the grammar. The parser produced correct full sentence syntactic and semantic representations for 70% of the 294 utterances, judged for correctness by trained annotators (inter-annotator reliability in our crisis management domain has a kappa score of 0.79 [44]). Our exact match evaluation requires not only syntactic constituent accuracy (the standard for statistical parser evaluations [45]) but also correct word sense disambiguation and predicate-argument relation assignment, which are key for effective interpretation. Note that this score represents a lower bound on parser coverage, because the correct parse may be ranked as a second or third alternative. This result shows the generality of the grammar and lexicon, as we obtain 70% complete and correct sentence interpretations before we have put any effort into extending the grammar for the new domain. Our previous research on other domains has shown that additional development significantly improves parser performance, reducing the error rate by at least 50%.

[46] evaluated the effectiveness of our domain specialization technique that uses mapping rules to convert the generic semantic representations to domain-specific concepts. Our Medication Advisor domain model contained 182 classes corresponding to domain-specific concepts. Mapping the generic parser ontology to the new domain required 95 straightforward rules. We then compared parsing speed and accuracy for specialized and non-specialized versions of

the grammar on a speech lattice parsing task in which the parser must select the best sequence of words from 50 possible interpretations output by the speech recognizer. The use of the domain mapping rules resulted in a 1.7 times speedup in the parser and 50% error rate reduction, dramatic improvements for a small amount of effort. To further improve parser performance, we are enhancing our search and pruning algorithm [47] and developing techniques for augmenting our deep parsing with statistical methods [48,49].

Note that the goal of our system is accurate intention recognition, not accurate whole-sentence linguistic analyses. Our discourse processing techniques work well starting from a set of meaningful fragments as long as the fragment interpretations are semantically accurate. The critical capability for the system (as for people) is to robustly make sense of what was heard and identify the intentions. Also important is the ability to identify when an utterance is not understood sufficiently well and to produce a clarification or confirmation question to resolve the problem.

### *5.2 Evaluating Intention Recognition*

The main barrier to using plan and intention recognition within dialogue systems is that they are intractable in the general case [37]. Systems which do utilize intention recognition (*e.g.*, [50]) tend to only use a small number of possible goals, as we did in the Chester demonstration system. Recent work in probabilistic goal recognition (*e.g.*, [51]) has made goal recognition more tractable, but at a sacrifice of representational power (typically, representation of parameter values, among other things). These techniques would be inadequate for a full Medication Advisor system.

To recognize intentions more generally without sacrificing expressive power, we have developed techniques that use a combination of statistical and symbolic techniques. The general idea is to use a fast statistical recognizer to quickly identify an n-best set of possible intentions, and then use these to focus the reasoning in a slower symbolic intention recognizer. We have created a statistical goal recognizer which runs in time linear to the number of possible goals, making it fast and scalable. At the same time, it is also able to predict goal parameter values, as well as make partial predictions for the cases where it is not yet sure enough to make a full prediction [39]. We have evaluated this recognizer in two diverse domains. In an emergency planning domain with 10 possible goal schemas, using simulation-derived data, we were able to get 96.4% precision and 67.5% recall for 3-best prediction, with the recognizer converging on the correct answer after only having observed a little more than half of the actions. In an operating system domain (Linux) with 19 possible goal schemas, and based on actual user sessions, the recognizer achieved 68.8%

precision and 38.7% recall for the 3-best prediction. An in-depth discussion of the different results can be found in [39].

### *5.3 Evaluating Pronunciation Modeling*

In Section 3.1, we mentioned how difficult it is to find pronunciations for words in the Medication Advisor domain and suggested that automatic techniques could prove helpful. In order to assess the magnitude of the problem, we collected all the words in the seven MedlinePlus HTML files used for knowledge acquisition. There were 1357 unique words, of which 529 (39%) were missing from the CMU Dictionary.

As explained above, our proposed solution is to use a grapheme-to-phoneme converter to obtain pronunciations for out-of-dictionary words. Although our joint n-gram model for bi-directional grapheme to phoneme conversion has demonstrated excellent performance both on random words [33] and on more difficult, specialized cases like names [52], we expect the lack of domain-specific training to make the Medication Advisor domain difficult to handle.

The Merriam-Webster medical dictionary, accessible from the MedlinePlus website, supplies a definition and pronunciation for only 267 of the 529 words. This is further proof that even specialized dictionaries fail to cover a large proportion of the medical terms, especially proprietary names (of the 262 words not included, about 200 are brand names). The 267 words include 68 proprietary drug names and 184 non-proprietary drug names and drug types; the majority of the other 15 words are disease names.

We created our reference set by transcribing the pronunciations using the CMU Dictionary phoneme set and conventions, including multiple pronunciations where given. We then used our general-purpose grapheme-to-phoneme converter to generate pronunciations for all the test words, and compared the hypothesized pronunciations to those in the reference set. The results for both the top hypothesis (1-best) and top 10 hypotheses (10-best) are shown in Figure 5. The bottom part of the figure shows word accuracies (full pronunciation is correctly hypothesized) and the top displays phoneme error rates (summing over substitutions, deletion and insertion rates). We include results that disregard confusions between schwa and other vowels, which account for the majority of the errors – a typical type of error in grapheme-to-phoneme conversion, but one that a recognizer can recover from easily if given enough context.

Overall, the word accuracy is rather low, especially for the best hypothesis (22-25%). Note that the same model's performance on general vocabulary exceeds 70% word accuracy [33]. 10-best performance (40-63% word accuracy)

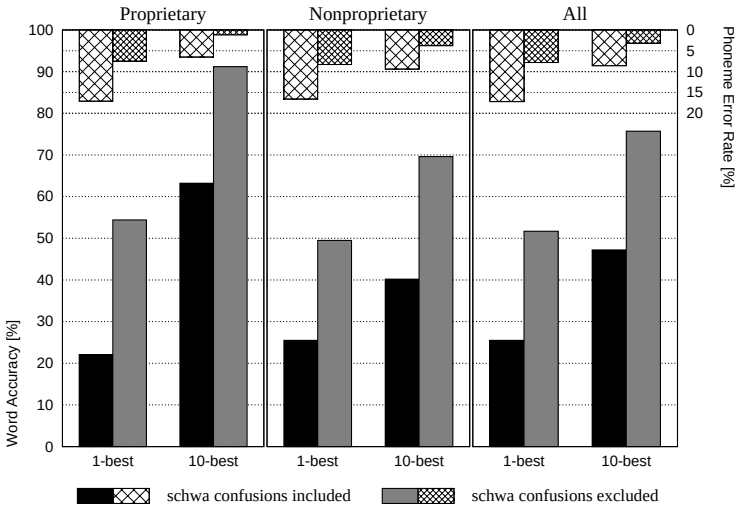


Fig. 5. Grapheme-to-phoneme conversion results on words in the Medication Advisor domain.

is markedly improved, which indicates that adding domain-specific data to the training of the model would prove very helpful.

When we compared the results for proprietary drug names and non-proprietary drug names we were surprised to find that the former were somewhat easier to pronounce. In part, this is probably due to the manufacturers' desire to find easy to remember names—hence easy to pronounce—for their drugs, though it is often the case that in their quest for uniqueness, they break the grapho-phonotactic rules of English. Again, a more detailed analysis of the errors for non-proprietary drugs reveals that many errors are systematic and could be easily avoided with a small amount of domain-specific data.

These results in no way should be taken as indications of likely recognition results on these words, given that many factors contribute to a speech recognizer's performance. However, they do provide hope that including automatically generated pronunciations in a dictionary to be used by a speech recognizer has significant potential. Only a direct evaluation of continuously spoken utterances from real speakers will give us precise clues as to how much.

## 6 Conclusions and Future Work

The Chester system suggests a promising avenue of research addressing the important societal problem of helping people manage their medications. As we have described, it also presents a variety of challenges to our current understanding of how to build such systems. As we hoped, the main effort in building the initial Chester system involved designing and implementing the back-end medication reasoning systems. The generic components actually worked rea-

sonably well on this new domain out of the box without modification. And the domain specialization techniques proved to be effective at improving the accuracy and efficiency of the parser.

We believe that the Medication Advisor is just the tip of the iceberg in terms of providing a conversational assistant that can help people take care of their health in their homes. Ultimately, we hope to expand the Medication Advisor into a more general "Personal Medical Assistant," which will integrate the information provided by the various technologies and provide a personalized point of contact for the residents of the home. The goal is not to replace doctors, nurses, or pharmacists. Rather, we want to provide systems that can help people better manage *their* part of their health care, and connect them to health care providers, family members, and the broader community as appropriate. We are in the early going, of course, but the need is great and growing, and the future looks very interesting.

Before tackling these more ambitious goals, however, we plan to demonstrate that robust, usable systems can be effective in simple, but real, applications. At present, we are exploring possible options with collaborators in the University of Rochester Medical School. We are looking at problems where effective techniques are known but they are too expensive (in terms of personnel) to implement. Besides the prescription compliance problem, we are also investigating systems that help patients with congestive heart failure and a system for advising on diet.

## References


[1] M. Naylor, D. Brooten, R. Campbell, G. Maislin, K. McCauley, , J. Schwartz, Transi-tional care of older adults hospitalized with heart failure: A randomized, control trial, Journal of American Geriatric Society 52 (2004) 675–684.

[2] J. B. Engelhardt, R. Allnat, A. Mario, J. Gao, An evaluation of the functionality and acceptability of the voice prescription label, Journal of Visual Impariment and Blindness 95 (11) (2001) 702–706.

[3] D. Mendelson, The Health Strategies Consultancy, Follow the pill: Understanding the u.s. commercial pharmaceutical supply chain, Report 7296, The Henry J. Kaiser Family Foundation (14 March 2005).

[4] D. Smith, Compliance packaging: A patient education tool, American Pharmacy NS29 (2).

[5] A. J. Claxton, J. Cramer, C. Pierce, A systematic review of the associations between dose regimens and medication compliance, Clinincal Therapeutics 23 (8) (2001) 1296–1310.

[6] A. Zuger, The 'other' drug problem: Forgetting to take them, The New York Times June 2 (1998) Section F, Page 1, Column 2.

[7] M. F. McTear, Spoken dialogue technology: enabling the conversational user interface, ACM Computing Surveys 34 (1) (2002) 90–169.

[8] J. Glass, S. Seneff, Flexible and personalizable mixed-initiative dialogue systems, in: A. I. Rudnicky, C. L. Sidner (Eds.), HLT-NAACL 2003 Workshop: Research Directions in Dialogue Processing, Association for Computational Linguistics, Edmonton, Alberta, Canada, 2003, pp. 19–21.

[9] D. Gross, J. F. Allen, D. R. Traum, The TRAINS-91 dialogues, TRAINS Technical Note 92-1, Department of Computer Science, University of Rochester, Rochester, NY, 14627 (June 1993).

[10] P. A. Heeman, J. F. Allen, The TRAINS-93 dialogues, TRAINS Technical Note 94-2, Department of Computer Science, University of Rochester, Rochester, NY (March 1995).

[11] J. F. Allen, L. K. Schubert, G. Ferguson, P. Heeman, C. H. Hwang, T. Kato, M. Light, N. G. Martin, B. W. Miller, M. Poesio, D. R. Traum, The TRAINS project: A case study in defining a conversational planning agent, Journal of Experimental and Theoretical AI 7 (1995) 7–48, also available as TRAINS Technical Note 93-4, Department of Computer Science, University of Rochester.

[12] G. Ferguson, J. Allen, B. Miller, TRAINS-95: Towards a mixed-initiative planning assistant, in: B. Drabble (Ed.), Proceedings of the Third Conference on Artificial Intelligence Planning Systems (AIPS-96), Edinburgh, Scotland, 1996, pp. 70–77.

[13] G. Ferguson, J. F. Allen, TRIPS: An integrated intelligent problem-solving assistant, in: Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98), Madison, WI, 1998, pp. 567–573.

[14] A. J. Stent, The monroe corpus, Tech. Rep. TR728 and TN99-2, Department of Computer Science, University of Rochester (Mar. 2000).

[15] N. Chambers, J. Allen, L. Galescu, H. Jung, A dialogue-based approach to multi-robot team control, in: Proceedings of the 3rd International Multi-Robot Systems Workshop, Washington D.C., 2005, pp. 257–262.

[16] J. Allen, G. Ferguson, A. Stent, An architecture for more realistic conversational systems, in: Proceedings of the Conference on Intelligent User Interfaces (IUI-2001), Santa Fe, NM, 2001, pp. 1–8.

[17] J. R. Searle, Speech Acts: An essay in the philosophy of language, Cambridge University Press, Cambridge, England, 1969.

[18] J. Allen, N. Blaylock, G. Ferguson, A problem-solving model for collaborative agents, in: Proceedings of the Conference on Autonomous Agents and Multi-Agent Systems (AAMAS-02), Bologna, Italy, 2002, pp. 774–781.

[19] N. Chambers, Real-time stochastic language generation for dialogue systems, in: Proceedings of the European Workshop for Natural Language Generation (ENLG-2005), Aberdeen, Scotland, 2005, pp. 40–48.

[20] C. D. Manning, H. Schütze, Foundations of Statistical Natural Language Processing, MIT Press, Cambridge, MA, 1999, Ch. 6, pp. 191–228.

[21] D. Jurafsky, J. H. Martin, Speech and Language Processing, Prentice-Hall, Upper Saddle River, NJ, 2000, Ch. 6, pp. 191–234.

[22] X. Huang, A. Acero, H.-W. Hon, Spoken Language Processing, Prentice-Hall, Upper Saddle River, NJ, 2001.

[23] J. Allen, D. Byron, M. Dzikovska, G. Ferguson, L. Galescu, , A. Stent, An architecture for a generic dialogue shell, Journal of Natural Language Engineering 6 (3) (2000) 1–16, special issue on Best Practices in Spoken Language Dialogue Systems Engineering.

[24] J. F. Allen, C. R. Perrault, Analyzing intention in utterances, Artificial Intelligence 15 (3) (1980) 143–178.

[25] J. F. Allen, Natural Language Processing, 2nd Edition, Benjamin/Cummings, Redwood City, CA, 1995, Ch. 17.9, pp. 567–570.

[26] D. K. Byron, Resolving pronominal reference to abstract entities, in: Proceedings of the Fortieth Annual Meeting of the Association for Computational Linguistics (ACL-2002), 2002, pp. 80–87.

[27] D. K. Byron, Resolving pronominal reference to abstract entities, Ph.D. thesis, University of Rochester Computer Science Dept., also available as URCS Technical Report 815 (2004).

[28] D. Hardt, An empirical approach to VP ellipsis, Computational Linguistics 23 (4) (1997) 525–541.

[29] M. Poesio, Semantic analysis, in: R. Dale, H. Moisl, H. Somers (Eds.), Handbook of Natural Language Processing, Marcel Dekker, New York, NY, 2000, pp. 93–122.

[30] L. Galescu, E. Ringger, J. Allen, Rapid language model development for new task domains, in: Proceedings of the First International Conference on Language Resources and Evaluation (LREC), Granada, Spain, 1998, pp. 807–812.

[31] R. L. Weide, The CMU pronunciation dictionary, release 0.6, http://www.speech.cs.cmu.edu/cgi-bin/cmudict (1998).

[32] H. Strik, C. Cucchiarini, Modeling pronunciation variation for ASR: A survey of the literature, Speech Communication 29 (1999) 225–246.

[33] L. Galescu, J. Allen, Bi-directional conversion between graphemes and phonemes using a joint n-gram model, in: Proceedings of the 4th ISCA Tutorial and Research Workshop on Speech Synthesis, Atholl Palace Hotel, Perthshire, Scotland, 2001, pp. 103–108.

[34] National Cancer Institute, Dictionary of cancer terms, http://www.cancer.gov/dictionary.

[35] N. Blaylock, J. Allen, G. Ferguson, Managing communicative intentions with collaborative problem solving, in: J. van Kuppevelt, R. W. Smith (Eds.), Current and New Directions in Discourse and Dialogue, Kluwer, Dordrecht, 2003, pp. 63–84.

[36] M. Pollack, Inferring domain plans in question-answering, Tech. Rep. MS-CIS-86-40 LINC LAB 14, University of Pennsylvania, phD thesis (May 1986).

[37] H. Kautz, A circumscriptive theory of plan recognition, in: P. R. Cohen, J. Morgan, M. Pollack (Eds.), Intentions in Communication, MIT Press, Cambridge, MA, 1990, pp. 105–134.

[38] J. Allen, Recognizing intentions from natural language utterances, in: M. Brady, R. C. Berwick (Eds.), Computational Models of Discourse, MIT Press, 1983, pp. 107–166.

[39] N. Blaylock, J. Allen, Recognizing instantiated goals using statistical methods, in: G. Kaminka (Ed.), Workshop on Modeling Others from Observations (MOO-2005), Edinburgh, 2005, pp. 79–86.

[40] National Library of Medicine and National Institutes of Health, Medlineplus drug information, http://www.nlm.nih.gov/medlineplus/druginformation.html.

[41] National Library of Medicine and National Institutes of Health, Antihistamines (systemic), http://www.nlm.nih.gov/medlineplus/druginfo/uspdi/202060.html.

[42] V. Kumar, Algorithms for constraint-satisfaction problems: A survey, AI Magazine 13 (1) (1992) 32–44.

[43] M. E. Pollack, L. Brown, D. Colbry, C. E. McCarthy, C. Orosz, B. Peintner, S. Ramakrishnan, I. Tsamardinos, Autominder: An intelligent cognitive orthotic system for people with memory impairment, Robotics and Autonomous Systems 44 (2003) 273–282.

[44] M. Swift, M. Dzikovska, J. Tetreault, J. Allen, Semi-automatic syntactic and semantic corpus annotation with a deep parser, in: Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC'04), Vol. 4, Lisbon, Portugal, 2004, pp. 1463–1466.

[45] E. Black, S. Abney, D. Flickenger, C. Gdaniec, R. Grishman, P. Harrison, D. Hindle, R. Ingria, F. Jelinek, J. Klavans, M. Liberman, M. Marcus, S. Roukos, B. Santorini, T. Strzalkowski, A procedure for quantitatively comparing syntactic coverage of English grammars, in: Proceedings of the DARPA Speech and Natural Language Workshop, 1991, pp. 306–311.

[46] M. O. Dzikovska, M. D. Swift, J. F. Allen, Integrating linguistic and domain knowledge for spoken dialogue systems in multiple domains, in: Proceedings

of Workshop on Knowledge and Reasoning in Practical Dialogue Systems at the 18th International Joint Conference on Artificial Intelligence (IJCAI-2003), Acapulco, Mexico, 2003, pp. 25–35.

[47] M. Dzikovska, C. Rosé, TFLEX: speeding up deep parsing with strategic pruning, in: Proceedings of the 9th International Workshop on Parsing Technologies (IWPT-05), 2005, pp. 194–195.

[48] M. Swift, J. Allen, D. Gildea, Skeletons in the parser: Using a shallow parser to improve deep parsing, in: Proceedings of the 20th International Conference on Computational Linguistics (COLING-04), Vol. 1, 2004, pp. 383–389.

[49] M. Elsner, M. Swift, J. Allen, D. Gildea, Online statistics for a unification-based dialogue parser, in: Proceedings of the 9th International Workshop on Parsing Technologies (IWPT-05), 2005, pp. 198–199.

[50] J. Chu-Carroll, S. Carberry, Conflict resolution in collaborative planning dialogues, International Journal of Human-Computer Studies 53 (6) (2000) 969–1015.

[51] D. W. Albrecht, I. Zukerman, A. E. Nicholson, Bayesian models for keyhole plan recognition in an adventure game, User Modeling and User-Adapted Interaction 8 (1998) 5–47.

[52] L. Galescu, J. Allen, Name pronunciation with a joint n-gram model for bi-directional grapheme-to-phoneme conversion, in: Proceedings of the 7th International Conference on Spoken Language Processing (ICSLP'2002), Denver, Colorado, 2002.