UNIVERSITY OF CALGARY

A Daily Scrum Meeting Summarizer for Agile Software Development Teams

By

Shelly Park

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE

DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

CALGARY, ALBERTA

August, 2007

UNIVERSITY OF CALGARY

FACULTY OF GRADUATE STUDIES

The undersigned certify that they have read, and recommended to the Faculty of Graduate Studies for acceptance, a thesis entitled “A Daily Scrum Meeting Summarizer for Agile Software Development Teams” submitted by Shelly Park in partial fulfillment of the requirements of the degree of Master Science.

_____

Supervisor, Dr. Frank Maurer
Department of Computer Science

_____

Supervisor, Dr. Jörg Denzinger
Department of Computer Science

_____

Dr. Christian Jacob
Department of Computer Science

_____

Mr. Ron Murch
Haskayne School of Business
Management Information Systems

_____

Date

Abstract

The goal of this thesis is to create a daily scrum meeting summarizer by experimenting with an existing speech recognition engine and using the transcript to produce a text summary about the meeting. For Agile software development teams, the purpose of daily scrum meetings is to communicate to the team members what is done, what will be done and what problems are encountered during the software development. The thesis presents a method that can improve the accuracy of the transcription by modifying an existing speech recognition engine and improving the coherency of the text through summarization. The proposed method of transcription and summarization can improve by 30% to 50% for the provided sample recordings against a transcript from a generic speech recognition engine. However, the speech recognition technology is still not ready for successful transcription of spontaneous conversational speech and produces too many transcription errors.

Publications

Some contents, ideas, and figures from this thesis have appeared previously in the following peer-reviewed publications:

Park, S., Denzinger, J., Maurer, F., Sharlin, E. (2006) **An Interactive Speech Interface for Summarizing Agile Project Planning Meetings**, Proceedings on Computer-Human Interaction (CHI 2006) Work in Progress Report, pp. 1205-1210 , April 2006, Montréal, Canada

Ablett, R., Park, S., Sharlin, E., Denzinger, J., Maurer, F. (2006) **A Robotic Colleague for Facilitating Collaborative Software Development**, Proceedings on Computer Supported Cooperative Work (CSCW 2006), Interactive Poster, Nov 2006, Banff, Canada

Acknowledgement

I thank my supervisors, Dr. Frank Maurer and Dr. Jörg Denzinger, for their expertise and advice. Their insights and feedback throughout the research have been a great learning experience. I would like to thank Dr. Ehud Sharlin for motivating me to consider a graduate school. I am thankful to the members of the EBE software engineering lab for the help I received over the years.

I would like to thank the University of Calgary and Alberta Learning for providing the scholarships including Queen Elizabeth II Graduate Scholarship, Departmental Graduate Research Scholarship, Alberta Graduate Student Scholarship as well as the Graduate Teaching Assistantship.

Most of all, I would like to express my deepest gratitude to my parents.

Table of Contents

List of Tables

List of Figures

List of Acronyms

ASR	Automatic Speech Recognition engine
EBE	E-Business Software Engineering Lab at the University of Calgary
HMM	Hidden Markov Model
LAN	Local Area Network
LSA	Latent Semantic Analysis
SDK	Software Development Kit
SNR	Signal-to-Noise Ratio
WER	Word Error Rate

1 Introduction

An intelligent transcription and summarization system that can automatically document meeting conversations is helpful to software development teams, especially where dedicating a human resource for the purpose of documentation is difficult. For example, producing documentation about discussions, decisions and problems discussed during the meetings can be useful for future references. Documenting meetings is a highly cumbersome task and burden to the team members because it takes away the resources from the actual development tasks. This research is motivated by the desire to automatically document the verbal conversations held during *daily scrum meetings* in Agile software development teams and help to produce a document about the progress of the project without burdening the development team members' time and effort.

This thesis is part of a larger team project called *Alan* where the goal of the project is to create a robotic colleague for agile software development teams. The purpose of the robot is to be integrated with the lab to help the administrative aspects of software development. In addition, the robot would interact with developers to generate motivation and fun during the software development process. The purpose of the *Alan* project is to maximize the multimodal capabilities in robots and to build a system that can help developers improve their communication and reduce tedious and redundant work that needs to be done by humans.

One manifestation of the *Alan* project is to implement a robot that can participate in the *daily scrum meetings*, which is a type of progress report meeting, and to produce meeting summaries. The robot for this project is given the name *ScrumBot* [AP+06]. Ideally, the ultimate vision is to have the robot participate in meetings much like another human colleague and generate and archive accurate and useful summaries of the meetings for future references. This team project has many research aspects including human-robot interactions, mixed initiative interaction, speech recognition,

summarization, artificial intelligence and computer vision. However, this thesis only covers the transcription and summarization aspect of the *ScrumBot* project.

In this chapter, I will describe the motivation and the scope of my research. But before I describe the research problems and the layout of my thesis, I will briefly describe what Agile software engineering is and the goal of daily scrum meetings.

1.1 Agile Software Engineering

Agile software engineering is a group of software engineering methodologies that share similar beliefs such as close collaboration, frequent delivery and face-to-face communication to improve software development. The *Agile Manifesto* (www.AgileAlliance.org) was drafted in 2001 by a group of 17 software practitioners to outline a set of core beliefs for Agile development; it emphasizes

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

This radical shift in the software engineering process was introduced to offer an alternative engineering process to Tayloristic approaches [Ta91] such as the Waterfall model introduced by W.W. Royce in 1970. The Tayloristic approaches such as the waterfall model emphasize stepping through phases like stepping downward in a waterfall. The phases include requirement analysis, design, implementation, testing, integration and maintenance. The biggest weakness of the waterfall model is that each phase must be perfected before moving on to the next phase, which caused inflexibility within the overall software engineering process. For example, if there is a faulty requirement, it is difficult to go back and fix the requirements during the subsequent phases without incurring a large cost for re-doing the requirement phase. This lack of adaptability in the waterfall model led to Agile development, which encourages

adaptability to the changing needs of the clients and software requirements through iterative processes.

There are many methodologies in the Agile software engineering. To take a few examples, there are eXtreme Programming by Beck [Be04], Scrum by Schwaber [Sc04], Lean development by Poppendieck [PP03], Crystal Clear by Cockburn [Co04] and several more. It is also not uncommon for Agile practitioners to use combinations of these methodologies for their software development.

Today's software development projects are complex and are faced with various changing variables: customer requirements, time, competitions, quality and resources [CH01]. To counteract such changing environmental variables, Agile methodologies emphasize iterative and incremental approaches to delivering software products, which can help respond to unexpected changes during the development process. Cockburn and Highsmith emphasize that in order for the team to respond effectively to changes, to reduce the cost of information exchange and to reduce the time from decision to feedback, the agile team must [CH01]:

- Place people physically closer
- Replace documents with more dialogues and the use of whiteboards
- Improve the team's amicability
- Make the experts part of the team
- Work incrementally

To facilitate better communication and improve the information exchange among developers, *daily scrum meetings* can be held each day where the developers talk about their accomplishments, problems and future plans. The *daily scum meeting* lasts about 15 minutes. Team members can get information about the project's progress and improve the dialogue among team members.

While these conversations can improve rapport among developers and improve the flow of information, they don't always guarantee documentation about the decisions and progress discussed in the meetings for future references. Detailed documentation of these meetings would violate one of the fundamental principles of Agile software development, because an unnecessary amount of effort would be spent on meeting documentation that may not be needed in the future. While close collaboration among developers can eliminate a need for some meeting documentation, it is difficult to obtain this information for those who didn't participate in the meetings. Therefore, there is no reason not to document the progress discussed during these meetings if there is a way to capture the information automatically.

1.2 Research Motivation

In this section, I am going to explain the motivation for my research. Given the above scenario, having an automatic meeting summarizer would be helpful, but the research question is what is actually possible with the current state of technology. One of the motivations for capturing verbal conversations for daily scrum meetings is its highly structured meeting agenda. The meeting is generally a series of monologues where each developer talks about what is finished, what will be finished and what problems are encountered. The topic of the conversation is limited to the software requirements in the current iteration. In the daily scrum meetings, there are no tutorials, design discussions or extensive implementation discussions on how to solve problems. It is simply designed to inform everyone in the software development team of everyone's progress. If there is a need for more meetings, a separate meeting is scheduled outside the daily scrum meetings.

I want to analyze what is actually possible to achieve despite the great hype about the current speech recognition and summarization capabilities given the simple structure of the daily scrum meetings. I want to investigate the biggest impediment in transcribing and summarizing daily scrum meetings, especially its spontaneous conversational aspect. I want to investigate if there are improvements that can be made by modifying

an existing speech recognition engine and whether the structured nature of daily scrum meetings can help the summarization process.

Voice applications such as transcribing and summarizing daily scrum meetings would require mixed-initiative interaction to effectively gather information from the developers. Because it is unlikely that the developers would provide all of the information, the system must take the initiatives to gather the information as needed. Mixed initiative interaction is “a flexible interaction strategy in which each agent (human or computer) contributes what it is best suited at the most appropriate time.”[He99] Mixed-initiative interaction stands between “effective human-computer interaction and multiagent systems” as the system and humans “coordinate their activities” based on who is best suited to lead the interaction for the specified task [He99]. The research question is whether an interaction is possible for spontaneous meeting speeches considering that a computer has no understanding of the actual contents of the meeting. As I will explain later in the thesis, an interaction beyond a rigid question and answer scheme was not possible because the computers cannot adapt to the way humans speak with the current technology. Giving more freedom to the users meant poorer performance for the computer.

One of the requirements for any type of interaction is reasonable transcription success. Without a reasonable transcription of what the user said, it would not only be annoying to ask the users to repeat the same phrases again but the system would be equally incapable in transcribing what the user said the second time. In this thesis, I show evidence that real time interaction is not possible because of the large amount of time required to process the meeting speech.

I have implemented two experimental prototypes to test the performance of the transcription and the summarization. I will describe what worked and the results from the different type of speeches and interactions.

1.3 Research Scope

Here is an outline of the research scope and how I intended to solve the problem.

1. **Identify the current capabilities of speech recognition and summarization technology for spontaneous conversational speech.** I have provided measurements on the effectiveness of speech recognition software and how it works for spontaneous conversational speech. From the experiment, I have outlined what a reasonable expectation of the possible implementation is.
2. **Modify the existing speech recognition engines to work for the daily scrum meetings.** The goal of the project is not to develop an entirely new speech recognition system but to modify an existing speech recognition engine to handle the unique needs of the daily scrum meetings. I identified what was possible to modify and how the modification was made to improve the transcription.
3. **Improve the coherence of the transcribed text.** The goal for the summarizer is to fix the incorrect speech recognition in the transcribed text by replacing elements with a set of coherent sentences. I identified the commonly occurring problems with the transcription and proposed a work around.
4. **Produce a summary of the meeting.** A summarizer determines which points in the transcript are important enough to be included in the summary. By using three level systems categorized by coherence, more meaningful sentences are included in the summary. I show how regular text summarization software is ineffective in summarizing conversational dialog transcripts.
5. **Evaluate the effectiveness of the summary.** The transcription and summarization are evaluated for their effectiveness in producing a coherent and condensed summary of the daily scrum meetings.

1.4 Thesis Structure

Chapter 2 presents a literature survey of the Scrum methodology, automatic speech recognition and text summarization. Chapter 3 analyzes the audio data and the conversational styles in the EBE daily scrum meetings. Chapter 4 presents the

experiments performed on different speech recognition engines. Chapter 5 discusses the results from an extraction based summarization technique using a commercial text summarization tool. Chapter 6 discusses the final implementation and what has been improved. Chapter 7 presents the results and analyzes the success of the implementation. Chapter 8 presents the conclusion and possible future work.

2 Literature Survey

In this chapter, the background information on the Scrum methodology and the daily scrum meeting is described. A brief overview of research being done in the area of speech recognition and summarization are presented.

2.1 Scrum

Scrum is a methodology that helps control the software development process using iterative and incremental practices. Scrum is designed to deal with complex software development problems that behave unpredictably. As Schwaber puts it, complex problems are the ones where "a statistical sample of the operation of these processes will never yield meaningful insight into their underlying mathematical model" or the resulting model would be in such a degree of coarseness that it is irrelevant [Sc04]. The Scrum process is an iterative and incremental process to help developers solve complex software engineering problems.

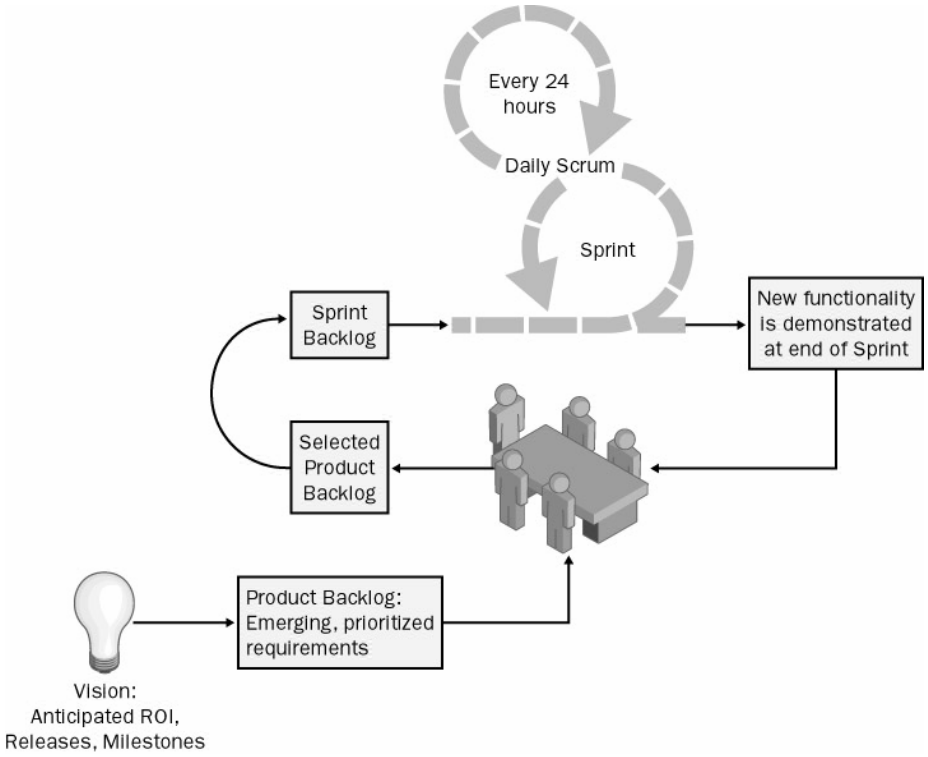


Figure 1: Scrum Process Overview [Sc04]

Figure 1 [taken from Sc04] is a diagram of the overall Scrum process. To start the project, the product owner will have a vision and the necessary budget. The product owner drafts a *Product Backlog*, which is a list of functional and nonfunctional requirements for the project and prioritizes them.

Scrum is composed of a series of Sprints. A *Sprint* is an iteration of 30 consecutive days. Each Sprint starts with a *Sprint planning meeting*, where the product owner and the team meet to discuss what will be finished during the upcoming *Sprint*. During the *Sprint planning meeting*, the development team negotiates with the product owner on the functionalities to be implemented. Once the negotiation is finished, the requirements are locked for 30 days and the product owner cannot change the requirements for the given iteration. Once the planning meeting is over, it is up to the team members to manage their own work to solve the development problems and deliver the product on time.

Scrum is based on the concept of self-organization, which means the team members decide the amount of time required to finish the tasks and how they will coordinate the development of the functionalities. Many face-to-face communications are required to effectively coordinate and collaborate with team members to solve the problems. To keep track of everyone's progress, a 15 minute *daily scrum meeting* is held everyday. At the daily scrum meetings, the team members answer three questions.

- What have you finished on this project since the last daily scrum meeting?
- What do you plan on doing on this project between now and the next daily scrum meeting?
- What impediments stand in the way of you meeting your commitments to this Sprint and this project?

The purpose of the meeting is to synchronize the team members on the progress of the work. If anyone needs to discuss the details of the problems, they schedule extra

meetings with the people involved. The daily scrum meetings can encourage team members to actively seek out help and to clarify ambiguities between the other team member's works. My research focuses specifically on automatically transcribing and summarizing these *daily scrum meetings*.

The Scrum process is rather simple in its concept, but it offers adaptability to changing needs. After each *Sprint*, the team and the product owner are presented with working software to easily confirm the milestones, to verify the implementation to the requirements and to evaluate if the team is on the right track.

The Scrum process is based on an empirical process. An empirical process is especially useful when a problem presented, which is too difficult to define in a model. Ogunnaike and Ray suggest that "it is typical to adopt the defined modeling approach when the underlying mechanisms by which a process operates are reasonably well understood. When the process is too complicated for the defined approach, the empirical approach is the appropriate choice" [OR92]. A well implemented empirical process control must have *visibility*, *inspection*, and *adaptation* [Sc04] and Scrum is a methodology that ensures visibility, inspection and adaptation in the software engineering process to solve difficult software engineering problems.

Agile methodologies rely heavily on the verbal and face-to-face communication. While written documents are important, an agile method emphasizes light documentation or just enough documentation to convey the necessary and important information. Scrum, especially, emphasizes real time communication simply because people respond more efficiently to real time interactions than to reading documents. Unlike documents, a conversation can encourage clarifications and interactions between team members. In addition, solving complex problems requires many hours of discussions between team members to come up with a good solution.

However, verbal communication also means there is no trail of artifacts that can summarize the meetings. In such a case, a person is volunteered to write up a report about the meeting and distribute it to the team. It also means someone's valuable time is spent on writing documents, which could distract the person from fully participating in the development effort. Often, the team will either take notes for themselves or there is no formal report written up about the meeting. Because the information discussed during the meeting is not readily available to those who didn't participate, it contributes to the problem of *visibility* of the project, which is one of the important aspects of empirical processes.

The purpose of our research is to experiment with various technologies to find out whether the current state of technology can accommodate the development of a meeting summarizer for these daily scrum meetings. Some of the current capabilities in automatic speech recognition engines and summarization techniques are presented in the following sections.

2.2 Automatic Speech Recognition

Automatic speech recognition (ASR) is a process where speech signals are automatically converted to corresponding words by computers. Most of the current automatic speech recognition engines are based on Hidden Markov Models (HMM). The theory of HMM was proposed in the late 1960s [BP66, BE67, BS68, BP+70, Ba72] and was implemented for speech recognition in the early 1970s at CMU called DRAGON (which became a commercial product in 1982 called Dragon Naturally Speaking) [Ba75] and at IBM [Je69, BJ75, JBM75, Je76, Ba76, BJM83]. However, limited research occurred due to the lack of tutorials on the theory of HMM until the late 1980s and early 1990s [Ra89].

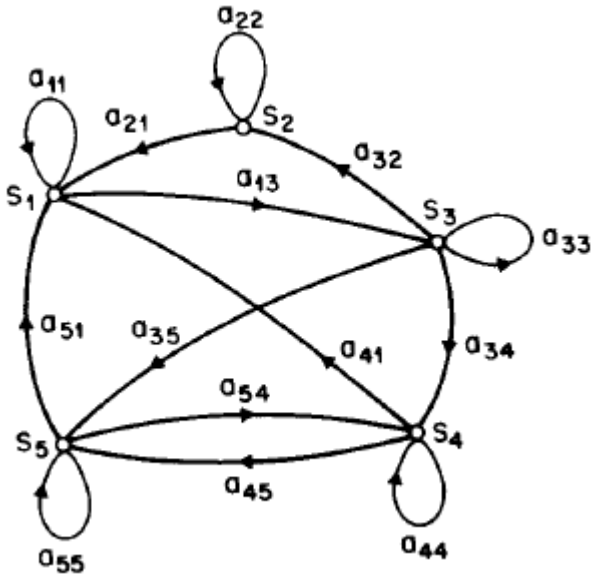


Figure 2: Hidden Markov Chain with five States and state transitions. The states represent the speech features and different transitions can form different word outputs [Ra89]

In a typical speech recognition system that uses HMM, speech feature vectors are extracted from the speech waveform. Then the corresponding words are found using two types of knowledge sources: acoustic knowledge and linguistic knowledge. The HMM is used to find the acoustic features and a stochastic language model is used to represent the linguistic knowledge. In a Hidden Markov model, the states are invisible but the probability distribution of the possible outputs from each state is visible. Using the temporal information, the speech feature vectors are analyzed using the model and produce the best sequence of states that can match the feature vectors [Ra89]. Using the linguistic knowledge, the pattern is matched with words. Figure 2 shows the state transition diagram for a HMM where each state is a speech feature.

With recent advances, speech recognition in controlled situations with clean speech can achieve a high level of accuracy, but the performance degrades drastically in real life situations. Some of the possible interferences can include variability from the speaker and variability within the environment such as background noise, reverberation, different microphones and transmission channels.

Among the several problems mentioned, one of the biggest causes of degradation is a noisy environment. For example, a speech recognition engine that can give 100% accuracy with the clean sampled speech can only produce 30% accuracy in a car traveling at 90km per hour [LB92]. A speech recognition engine that can give 99% accuracy in a quiet environment drops to 50% accuracy in a cafeteria [DB+93].

There are two major causes for degradation of speech recognition in a noisy environment [Ju91, Go95].

- Additive noise such as white noise or background noise can contaminate the speech signal and change the data vectors [BBH92, VM93]
- Speech in a noisy environment can cause articulation variability as the speaker tries to speak over the noise. This is called the Lombard effect [Pi85, HB90, Le89, JA90, 1992, Ch88, HC89].

The degradation is due to the mismatch between the training speech profile and the actual speech from a noise environment. Subsequently, one area of research in speech recognition is on reducing the mismatch between the training and the recognition. For example, Wiener or Kalman filtering can improve the signal-to-noise ratio but not necessarily the intelligibility of the speech or the recognition accuracy due to the change in the overall speech signal [KGG89, MC91, Os89]. There are various techniques including speech parameterization [NS+85, DM80], HMM [Ra89], probability density functions, ANNs (Artificial Neural Networks) [LMP87, HH93], linear discriminate analysis, and parameter estimation [Go95].

The underlying approaches for speech recognition in a noisy environment can be categorized into four types.

- The system is noise independent and uses the same configuration for both noisy and clean speech. It searches for noise resistant features for recognition.

- Use *scheme speech enhancement* where the speech is converted to match the sound produced in the training environment.
- Use *model compensation* or *model adaptation* where the training data is converted to match the noisy environment.
- Use *multi-style training* where speech data is collected from all possible environments. However, collecting data to train every possible noisy environment is not realistic [LMP87].

However, the best solution to dealing with a noisy environment is to use a better microphone. For example, using a headset that can be placed closer to the mouth is better than a desktop microphone placed at an arm's length. Also, using a microphone with noise canceling capability can improve the speech recognition, especially from white noise like wind and mumbling background conversations.

A human-to-human, spontaneously spoken dialogue contains a large amount of variability due to accents, speaking styles and speaking speed. Because the current ASR can only handle small variants in speech, any distortions from accents, styles or speed that are different from the training, which is based on dictation styled speech, can result in poor performance. While expanding the dictionary of the speech recognition engine to handle various speaking modes [OB+96] intuitively seems like a solution, research has shown that it actually confuses the system more [WB+98]. This is because larger search spaces mean more possibilities and a wrong recognition can be the result.

One of the major problems in ASR is the large search spaces. The search space for speech recognition is made up of acoustics, lexicons (legal words) as well as legal word sequences [YH+89]. The increase in the states in the HMM means an increase in the size of networks of states, which can result in a decrease of recognition performance. If we can restrict both the acoustic and linguistic knowledge, better performance can be achieved although fewer words can be recognized [YH+89].

Currently, the most common method is to use a *grammar* to restrict the search space. A *grammar* is a way to restrict the allowable word sequences. A *grammar* contains both syntactic and semantic constraints. The speech recognition can be improved by anticipating certain inputs based on the *grammar*. It can correct the recognized sentence by comparing the output with the anticipated results. For example, Young et al. has created a spoken language dialogue system for Navy ship status. It dynamically generates a grammar based on the previous dialogue from the user to increase the recognition rate [YH+89].

Another area of ASR research is spoken dialogue systems where a user interacts with computer applications such as databases or expert systems using speech. Spoken dialogue systems are computer systems that can interact with humans on a turn-by-turn basis. While the research started in the 1960s in Artificial Intelligence, actual advances started only recently within the last decade. Some of the successful applications are computer-telephony integration and voice portals [MK+02]. The application can range from question and answering systems with simple *yes* and *no* to a larger vocabulary.

Correcting and predicting speech are more difficult for spoken dialogue systems. Beyond the simple command and control type of applications, mismatches can occur due to wrong assumptions by the system and out-of-topic questions by the users. Often it takes repetitive interactions between the machine and the user to reach the correct assumptions. Because error-handling is not sophisticated enough, preventing an error is more important, usually by simplifying the tasks such as reducing the number of possible commands. In this type of application, grammar is used to restrict the possible dialogues that can happen between the machine and humans. The humans are guided by the machine to answer in certain ways that can be recognized by the grammar [LHS05]. This type of application is said to have *system initiated* interaction because the computer controls the interaction.

2.3 Automatic Speech Recognition for Meetings

Recently, there is research being done in the area of automatic speech recognition for spontaneous dialogues for meetings. Because speech recognition for meetings is a very difficult problem to solve, it is even given a metaphor: the “ASR-complete” problem [MB+03]. Meetings contain all the ASR problems of spoken language recognition, such as transcription, microphone technologies, speech overlap detection, utterance segmentation, speaker identification and disfluency detection [MB+03]. Many research groups are actively researching into ASR for meetings. Just to name a few in the U.S., there are Stanford Research Institute, Columbia, Carnegie Mellon University, BBN Technologies and Microsoft. In Europe, there are Sheffield University, the Technical University of Munich, the University of Twente and Brno University [MB+03]. However, even given the active research in this area, the current levels of success are modest at best and still leave a great deal of problems to be solved.

For example, Yu et al. have achieved 40-50% word error rate for their ASR designed for spontaneous conversational dialogues [YC+98]. Morgan et al. tested their ASR for their ICSI weekly development meetings. The best word error rate (WER) achieved is 36% for the 10 minute conversations using high quality close-talking microphones (mostly headsets). The meeting conversation is recorded simultaneously on all microphones using different channels. However, the word error rate doubled to 61.6% for the table top microphones, because of the distance from the speaker and additive noise. Even with close-talking headset microphones, the best word error rate for non-native English speaker was 72% [MB+03].

The poor recognition is blamed on three reasons [WY+01].

- Mismatched and degraded recording conditions (remote, different microphone types)

- Mismatched dictionaries and language models (typically idiosyncratic discussions are highly specialized on a topic of interest for a small group and therefore very different from other existing tasks)
- Mismatched speaking style (informal, sloppy, multiple speakers talking in a conversational style instead of single speakers reading prepared text)

Moore and McCowan have experimented with microphone arrays for overlapping speech during meetings. Microphones can't detect the direction of the speech; therefore, microphone arrays are proposed to detect speech overlap using the variability in the sounds between different microphones. However, this was only tested with a small vocabulary speech recognition corpus (recognizes only numbers) [MM03].

The research into creating ASR for meetings has only just begun. Researchers are creating a large meeting corpus to help evaluate the future ASR implementations and analyzing some of the interesting characteristics of various meeting styles [BMY02]. The results being obtained from their ASR are collected for benchmarking purposes against possible future implementations.

2.4 Automatic Summarization

Maybury defines automatic summarization as "the process of distilling the most important information from a source or set of sources to produce an abridged version for particular users and tasks" [Ma95]. There are *indicative summaries*, which provide an index of documents for more in-depth readings, and *informative summaries* that actually describe the contents of the documents [MRC05]. Summaries can be *topic-related* or *generic* and can also be *extracts* or *abstracts* of the original documents. Summaries can be based on several documents or just a single document [MK+02].

The automatic text summarization research began in the 1950s [Lu59, Ed68]. Luhn attempted to create an abstract for written documents by extracting sentences with high significance measurement. The significance measurement is obtained by simply

counting the word frequency and sentence positions. While the method is unsophisticated, the experiment has shown the possibilities for automatic text summarization and has helped pave the way for the automatic summarization discipline. However, there was a little or no progress in document summarization for several decades until the recent explosion of information in electronic format due to the World Wide Web. Some of the recent research interests involve multi-document summarization and summarization for hand-held devices [Ma01].

There are two domains that mainly influence the summarization research area. The research in information retrieval has produced extraction techniques based on linguistic analysis and statistics. The abstraction technique is mainly influenced by artificial intelligence, which uses knowledge-based methods for condensing the information. The summarization field has seen a lot of progress in extraction-based methods with results that are comparable to human-generated extractions [Ma98]. However, the abstraction technique hasn't been very successful due to the difficulty in creating a large knowledge base of the world.

A summarization technique can be categorized by the following major classes. The following characteristics are from [HL98].

Input: Characteristics of the source texts	
Source size	Single-document vs. Multi-document – Single document uses single input text. A multi-document summary uses contents from more than one text that are thematically related.
Specificity	Domain-specific vs. General – Domain-specific summarization technique uses knowledge about the content to derive the summary. General summary can handle summary from any domain.

Genre and Scale	Some of the typical genres include newspaper articles, opinions, novels, short stories, non-fiction books, progress reports and business reports. The scale can range from book-length to a few paragraphs.
Output: Characteristics of the summary as a text	
Derivation	Extract vs. Abstract – The extraction technique extracts the texts from the input text. The abstraction technique generates new text from the analysis of the input text.
Coherence	Fluent vs. Disfluent – A fluent summary is composed of grammatical sentences and the sentences are coherent. A disfluent summary is fragmented.
Partiality	Neutral vs. Evaluative – Neutral summary is impartial and unbiased regardless of the opinion presented in the text. A summary can be biased by including opinionated sentences from the input text.
Conventionality	Fixed vs. Floating – A fixed summary is created for specific use and readers. A floating summary uses fixed conventions but makes no assumptions about the readers or the usage.
Purpose: Characteristics of the summary usage	
Audience	Generic vs. Query-oriented – A generic summary gives equal importance to all major themes in the input text. A query-oriented summary produces a summary with only the theme that user has requested.
Usage	Indicative vs. Informative – An indicative summary provides the subject or the domain of the input text only. An informative summary describes the contents of the input text.
Expansiveness	Background vs. Just-the-News – A background summary provides extra information to understand the summary. A just-the-news summary contains just the contents inside the input text.

Table 1: Summarization Categorization [HL98]

Mani and Maybury have categorized the summarization approaches into two classes: *knowledge-rich* approaches and *surface-oriented* approaches [Ma01]. The knowledge-rich approaches use a formal representation to dissemble the original texts into templates. The surface-oriented approaches use weights to extract sentences into the summary. It can be based on (1) position, (2) presence of cue phrases (3) presence of background terms from title, heading or user's query and (4) presence of statistically salient terms [Ma01].

There are several summarization techniques in existence and some researchers have tried combinations to improve the summarization results. The term frequency based extraction was originally proposed by Luhn [Lu59]. The measurement known as *significance factor* is used to rank the sentences in the document. The frequency of the words based on very large corpora of documents is used to determine the significant words. The sentences containing these significant words are considered more important and are ranked by the locations of the words in the sentence. Because certain genres of documents have important sentences in fixed positions - such as the title, the first or the last sentence - we can automatically extract sentences in these locations with a relative confidence that we are extracting the important sentences. Edmundson [Ed68] used cue phrases, title and locations of certain words to extract important sentences.

The word counting method involves giving more importance to sentences containing words that occur frequently. Because the most frequently occurring words are "a" and "the", Salton [Sa88] compiled word lists based on relative frequency. If certain words occur equally often in more than one document such as "a" and "the", these words are not considered important.

Baxendale [Ba58] categorized sentences into two types: bonus phrases and stigma phrases. For example, a bonus phrase is a phrase indicating important contents like "the most important" or "in conclusion". A sentence is given a higher score with more bonus

phrases and a lower score if it contains stigma (or unimportant) phrases. These bonus and stigma phrases must be compiled manually.

2.5 Speech Summarization

A spontaneous spoken dialogue summarization presents a different set of challenges than text summarizations. Mainly because the speech is disfluent; automatic speech recognition is error prone; phrases are either fragmented or contain many grammatical mistakes; and information is sparse. Because attempting a detailed linguistic analysis on the semantics of meeting conversations is a difficult process, various methods of extractions are used instead [WB+98]. While there are considerable research activities in text summarization, there is less work in speech summarization. Most work in speech summarization is in the domain of broadcast news [VR+99, WM99, HF+03], mainly because of the clear speech and clean recordings.

There are two types of summarization in spoken dialogue summarization: feature based approaches and textual approaches. In feature-based approaches, a set of prosodic features such as variations in pitch, loudness, tempo and rhythm are examined. In textual approach, a transcript is produced using ASR and a summary is produced from the text. One of the methods for a textual approach is to use latent semantic analysis (LSA). A contextual-usage of words is extracted by statistical computation of words against a large corpus of text [LFL98]. The word is assigned a weight based on the frequency of the term in the document. This method is used often in text summarization as well [GL01, SJ04].

Some of the previous research on summarizing speech used prosodic features such as pause information and syllable duration [KR05, MRC05] for utterance separations. For example, Koumpis and Renals [KR05] used prosodic features for summarizing voicemail messages to send summaries to mobile devices. Murray et al. used speech and discourse features for extracting the relevant phrases out of meeting conversations, which has shown to be better than just using text transcripts [MR+06]. Zechner [Ze02]

conducted an experiment on the summarization of spoken multiparty dialogues based on maximal marginal relevance (MMR) [CG98], automatic speech disfluency removal, sentence boundary marking and question-answer pair detection. Hori et al. [HHM03] developed an integrated speech summarization for a lecture summarization task, where recognition and summarization are composed into a single finite state transducer. It means the transcription doesn't have to be processed again to produce a summary. Murray et al. used speaker activity, turn-taking and discourse cues in addition to prosodic features to improve the speech summarization, which works better than text summarization from the transcript [MR+06]. However, these techniques are designed for clean dictation speech.

Speech summarization can also work with ASR transcribed texts only, without any prosodic features. For example, Hori and Furui used topic score and linguistic score to build a word set for sentence extraction. Their transcript is produced using the ASR on broadcast news. The summary was produced with a length of about 60-70% of the original. About 72% of the important sentences were extracted [HF00, HF03].

Kikuchi et al. proposed two-stage summarization [KFH03]: important sentences are extracted in the first run and the compaction is then performed in the second run. The importance of a sentence is determined by its linguistic score, significance score and confidence score. During sentence compaction, any sentences that scored relatively low are eliminated. This method is based heavily on grammar corrections. It also means a transcript with too many grammar mistakes will not work very well.

The summarization techniques, no matter the type, work better with human-transcribed text. This is probably because the human-transcribed text contains fewer errors than the ASR transcript. Studies show that the word error rate for the summary tends to be lower than the original transcript produced by the ASR [VR+99, MRC05, ZW00]. Even if the summary doesn't extract all the important sentences, at least speech summarization is

an effective method for reducing word error rate in the spoken dialogue transcripts by the ASR.

2.6 Summarization Evaluation

There are two categories of evaluating text summarization: intrinsic and extrinsic [SG96]. An intrinsic evaluation tests the summarization system. It can evaluate the *coherence* of the summary, *informativeness* of the summary, grammatical mistakes and how the system handles structured texts like tables [MK+02]. To measure *informativeness,* often another *ideal* summary is created for comparison. However, the summary can be equally informative even if it's different from the *ideal* summary. Often an extrinsic evaluation tests the effectiveness of the summary by testing the helpfulness in completing some other task [MK+02]. This method is used often in question-answering and comprehension tasks.

A common method for evaluating extraction based summarization is to compare the extracted summary against the summary being used as an experimental control and determine whether it is better or worse [DDM00]. It can measure the term frequency counts and term weighting [Du91]. A summary containing more of these terms would score higher than a summary containing fewer.

The recall method compares the summary with human-generated extracts. For example, the human-generated summary is considered to contain the "ground truth" and the summary with the most number of these "ground truths" scores highest [GK+99, JB+98]. Jing et al. found that the quality of the machine-generated summary decreases as the summary length increases [JB+98].

2.7 Conversational Speech and Natural Language Understanding

Natural language understanding refers to the computer's understanding of human languages. While understanding written documents has achieved some success over the

last 25 years, understanding conversational dialog is still a difficult area. Mainly, because both automatic speech recognition and natural language understanding is not designed to handle spontaneous conversational dialogue. It is possible to analyze grammatically correct sentences in documents through syntax analyses, but these techniques are useless because conversational dialogues tend to be non-grammatical.

The word *understanding* for a computer is not the same as how it would apply to humans. Rather, it means being able to respond appropriately to queries made using normal human languages [Wh90]. Giving out a correct output by computer doesn't mean it understood the contents of the message. As White puts it, "communication is the exchange of messages that describe the state of a model or change in the state of a model", which means a computer would require contextual information about the relationship between different symbols for representing information.

Spontaneous speech can often have grammar errors: false starts, mixed cases and mixed tenses [Bo44]. An interesting characteristic of spoken dialogues is that even if the person randomly leaves out verbs and nouns, those dialogues still make sense. Cherry has studied how people can still understand the general meaning of the sentence when all grammatical clues are removed from the sentence [Ch75]. For example, "woman, street, crowd, noise, thief, bag, scream, police". In such a case, natural language understanding is about extracting meaning from partially completed phrases and extrapolating the intended meaning.

Lindsay and O'Connell investigated the accuracy of spoken dialogue transcription by humans. Two people were allowed to pause only, but no repetition or replay was allowed. Two other people were allowed to replay as many times as they wanted. However, none of them produced a verbatim transcript, although the summary contained all of the semantic contents [LO94].

Ferber investigated the human ability to catch a slip of the tongue. Ferber asked four people to find a slip of the tongue while listening to a radio discussion without a pause or a replay. None of the slips was found by all four people [Fe91], which means people are not listening to all of the words or listening for grammar mistakes.

2.8 Summary

In this chapter, some of the background information of on the Scrum methodology and research in speech recognition and summarization were presented. In the next chapter, I present the observations from several daily scrum meetings held in the EBE software engineering lab to give background information about the meeting environment and meeting audio qualities.

3 The Daily Scrum Meeting Scenario

The daily scrum meetings of the EBE software engineering lab are the domain for the daily scrum meeting summarizer. Our recordings were obtained from five participants who are working on three different projects. The participants have agreed to provide sample recordings for this research. In this chapter, I present how the daily scrum meetings are held, a brief description of a storyboard, observations about the meeting structures and environmental settings such as background noise problems. I also examined the conversational styles.

3.1 Storyboard

During a Sprint meeting, the requirements for the projects are written down on index cards called *story cards* and they are posted on a board called s*toryboard* (Figure 3). During the Sprint meeting, the developers decide what is realistically possible to finish before the next Sprint meeting. Requirements that cannot be fulfilled are put into the backlog. Each time a requirement on a story card is complete, the card is moved from the *To Do* column to the *Done* column. The board gives a quick overview of the progress being made for each project.

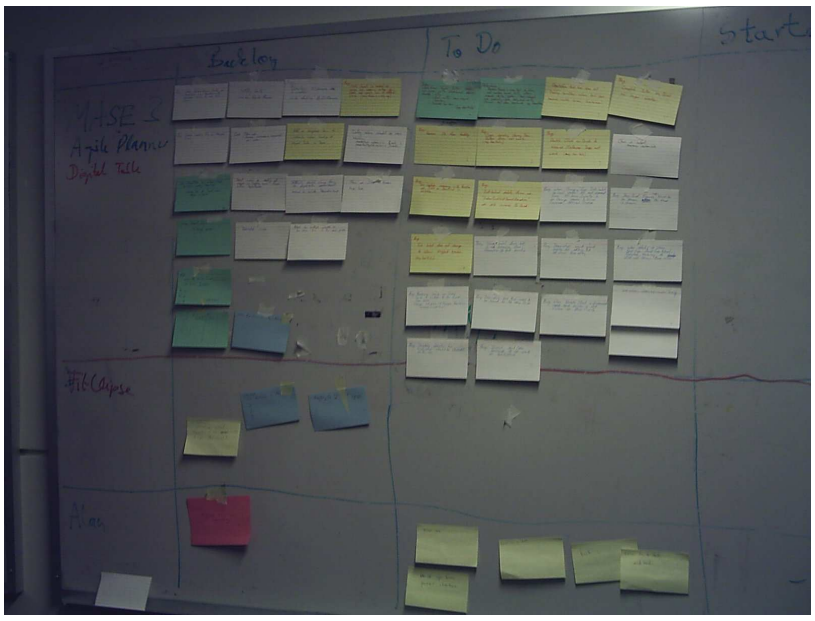


Figure 3: Storyboard

Here is an example of some of the story cards. One bullet point is for one story card.

- Automate Mouse & Key events
- Double click on cards to expand/collapse does not work
- Follow light
- BuildBot – Power Station Loading
- Bug: Mouse pointer is flickering

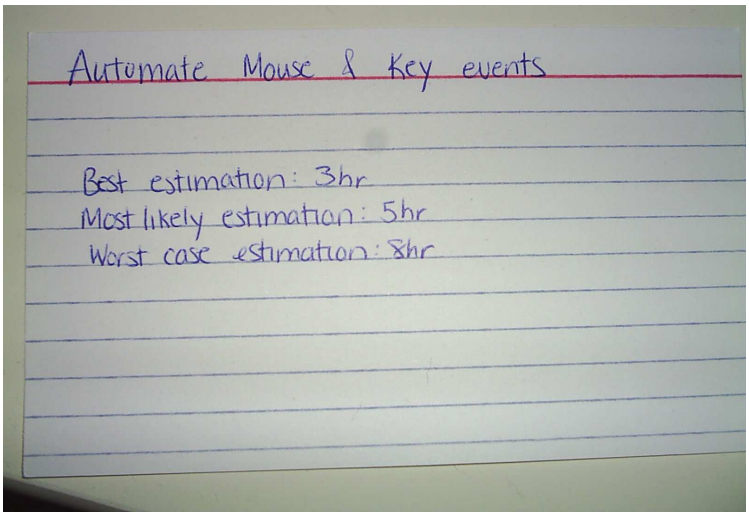


Figure 4: An example story card

Figure 4 shows an example of how these story cards look. The story cards often contain just enough information for the developers to understand, but generally are very vague and difficult for an outsider to understand.

3.2 The Daily Scrum Meeting

The daily scrum meeting is usually held in front of the storyboard, simply because it is easier to move around the story cards between the *To Do* and *Done* columns. While the participants discuss what they did, what they will do and what problems they encountered, the developers would point and move the story cards to clarify their statements. A scrum master would call for the meeting and everyone takes their turn talking about the progress of the project. Figure 5 shows a photograph of developers having a daily scrum meeting.

Figure 5: In Daily Scrum Meetings, developers shuffle story cards and talk about their development progress

Here are my observations made about the daily scrum meetings.

1. **Minimal conversation between the participants** - Usually the scrum master asks questions and the developers answer. The conversation is between the scrum master and the developer and there are very few conversations between the developers. However, when two people are working on the same project, sometimes they would finish each other's sentence. The speeches are usually very short. Monologues are good for the summarizer because disfluency causes problems for the summarizer as thoughts are dispersed across many speakers.
2. **Speech lasts no more than 5 minutes** – Most of the verbal reports to the team lasts about one or two minutes per person. Because of the brevity of the meeting, it is difficult to understand the conversation without the context of the previous progress made in the project or the context of the project. The conversation is also very context-sensitive as people in the meeting are already familiar with each other's work and keep the reports brief. Any tutorials or lengthy discussions are taken outside of the daily scrum meetings. This brevity and context-sensitive aspect makes creating the summarizer very difficult.
3. **The order of discussions on finished tasks, future plans and problems is random** – There is no specific order for the discussions. Randomly, the scrum

master chooses one of the participants to talk. The developer may talk about the finished tasks, problems or future plans in any order he/she wishes. If there is a big issue with the project, the conversation will start with the problem. The developer sometimes may miss out on one of these topics. In this case, the scrum master would ask for some clarifications. Because the computer doesn't understand the meaning of the dialogs, a random order makes summarization very difficult to categorize the spoken phrases into what's done, what will be done and the problems. We have to enforce the user to speak in an order.

3.3 Noisy Environment

As stated in Chapter 3.1 and 3.2, the concept behind daily scrum meetings is very simple. However, the main problem with building a summarizer as described in Chapter 1.2 is the difficulty in making computers adapt to the human world. As stated in the literature survey in Chapter 2.2, background noise is one of the big causes of poor speech recognition accuracy. Because the microphone is at least an arm's length away from everyone in the meeting, it tends to pick up a lot of noise from the surrounding environment. In this section, the noise problems in the audio recordings of the daily scrum meetings are examined.

A real-life graduate computer science lab is bustling with a lot of background noise. Getting clean speech from the lab is quite impossible due to the microphone's distance from the speaker's mouth and various additive background noises such as humming computer fans, people walking, talking, opening and closing of doors.

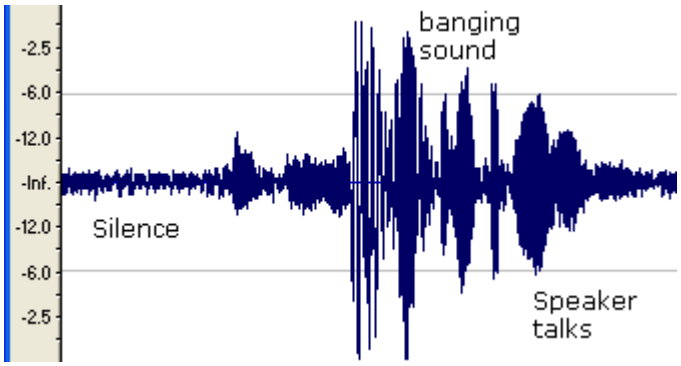


Figure 6: A background noise overlapping the speech

For example, Figure 6 shows a portion of a speech where the first half is recording silence. Then there is an unidentified banging sound in the middle and the speaker continues her talk. The banging sound is most likely a door opening or closing near the meeting area. Any speech embedded in the midst of the banging sound wouldn't be recognizable for the automatic speech recognition engine. But humans are most likely intelligent enough to extrapolate what she was saying despite her voice being drowned out by the banging sound.

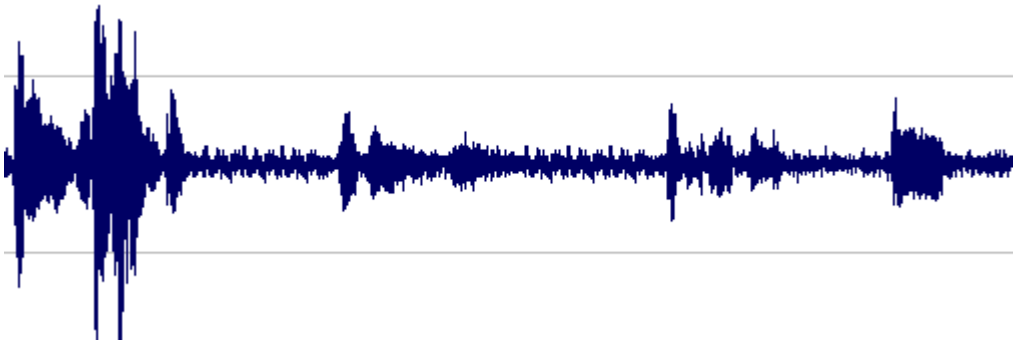


Figure 7: A speech sample of someone talking softer as the time passes

Figure 7 shows an example of a speech that started loud, but as the conversation continued, the voice got softer. Some parts of the speech are so soft that it is indistinguishable from the background noise. Due to the room reverberation, a soft voice, such as shown in this audio file, is difficult for the speech recognition engine to decipher. The only way to solve this problem is getting a headset for each speaker,

which would reduce the additive noise picked up by the microphone along with the voice. However, it would mean extra setup processes and intrusive equipments for recording these very short meetings, which could be a very cumbersome task.

3.4 Conversational Style

As stated in Chapter 2.3, the sentences in spontaneous conversational dialogue are often incomplete, grammatically incorrect, disfluent and contain sloppy pronunciations. Compounded with soft voice and background noise, some of these audio recordings are hard to understand even for humans. Here is an example of a scrum meeting transcribed by a human.

Scrum Master: Then let's get started. What did you do since the last time?

P1: Um. When was the last meeting? Wednesday? So I got…I got some of the initial the structure setup. Um. I'm still working on that. So you know I tried GEF. Going through making sure all these stuffs are setup properly. And ah. I imported some of the videos. Or actually I imported all video but I haven't analyzed yet.

Scrum Master: Um.

P1: That's about all that's happened.

Scrum Master: And what's due the next Thursday?

P1: Um. Thursday I hope some of the GEF, some of the GEF stuff done. Um… Specifically, I want to get the other part we did - one on the screen.

Scrum Master: Ok.

P2: And for me, I… I'm still working on that new project. I'm trying to work on the project. And since I'm going to hold like a presentation for the people in the university lab. To let them be familiar with JUnit and extensions like XMLUnit, Jimmy, JFCUnit, which is connected to…related to swing interface.

Scrum Master: When is it? When is this presentation?

P2: I think in less than a week.

Scrum Master: Let me know when it is.

(The rest of the meeting conversation omitted.)

Several observations can be made about the conversations during the meeting.

1. **No speaker identification** - People rarely call out a person's name. Eye contact is used to indicate who should talk next. Because the audio is being recorded

with only one microphone, it is impossible to identify the speaker simply based on the voice direction and speech characteristics. While there is some research done in speaker recognition [GSR91, GS94, RR95, WC00, WB+02], the techniques aren't sophisticated enough and are unsuitable for real-life situations. In addition, there are no readily available software applications that can be used for speaker identification. The implication is that a human has to manually control the recording application to specify who is currently speaking.

2. **Ambiguous pronouns** - Without the visual information, the conversation can be very cryptic to the reader, especially if the reader doesn't have a prior understanding of the project. For example, people would use pronouns such as "that", "it", "this" or "stuff" to describe a presentation, project names, software names, error descriptions, algorithm names and other various important but hard to pronounce information. Some people are more articulate about what they are trying to say, but most people defer to generic pronouns. Ambiguous speech makes the summarization process difficult because the resulting summary would be equally ambiguous. The system can only extract important sentences but cannot reword ambiguous words as the system doesn't understand the meaning.
3. **Sparse Information** - The conversation only lasts about a minute or two per person and the actual amount of information conveyed is very small. Even though people could talk a lot, most of the conversation can be summarized in one or two point-form sentences. Often the reason for longer conversations is that the speaker wasn't articulate about what he/she was trying to say. An interrogation style conversation poses a difficulty for the ASR because of the disfluency between the speakers. The overall meaning of the dialogue is embedded across multiple sentences and multiple speakers.
4. **Computer jargon** - The dialogues also contain computer jargon such as JUnit, XMLUnit or JFCUnit. The jargon poses special problems for the ASR, because the jargon is usually not in the standard vocabulary list.

5. **Erratic speech speed** - The speed of the speech is also a problem. Unfortunately, the ASR is not designed to handle either the fast or the slow speech. While a slower speed is preferred for accuracy, slower conversation also means the speech might not be very natural. (eg. holding a word for a longer period of time or mumbling through words). As seen in Chapter 4, speech speed other than a slow, dictation style speech decreases the accuracy rate.
6. **Body language** - Understanding body language is also an important aspect of face-to-face meetings. There is very limited research done in interpreting human body language and social context by computers [Pe05, JS05]. The audio recordings alone don't always convey the entire message. People can point to an object instead of saying the object name. They can use body languages to convey messages such as a shrug to convey "I don't know" or nod to convey "yes". Summarizing a face-to-face meeting means a lot of information is not actually spoken. Mixed with ambiguous speech, the resulting summary could be very ambiguous.

A face-to-face meeting creates a more difficult environment for computers to obtain a clean transcript than a distributed environment such as phone conversations where people are more conscience about speaking clearly and not relying on body language. Because listening to conversations in a noisy environment is a second nature to humans, it's not obvious how difficult a problem face-to-face, spontaneous, human conversation can have for computers.

3.5 Conversational Domain

Unlike a generic meeting, daily scrum meetings in the EBE software engineering lab have a very specific topic and style for conversations. This research assumes that, despite the problems presented above, using the domain information about the conversation can help improve the transcription and summarization. Here are the conversational domains.

1. **Time related** - The conversations are about completing tasks. It means the dialogues are time related, but usually not with a specific date. Some of the time-related words are “today”, “tomorrow”, “next Monday”, “coming Tuesday” or “after the meeting”. However, it is rare that people would give a specific time such as “November 1st” or “1:30”.
2. **Story cards** - The conversations are based on the story cards. It is most likely that people would use words that are written on the story cards. However, some of the related words might not be apparent from the story cards, such as reporting what the error message was. Therefore, if we know a speaker is working on a Java program, more vocabularies can be added that are Java-related including Java-specific error messages. Also, story cards do not always contain all of the information required to build the vocabulary list as the contents in the story cards are small.
3. **Finishing and planning tasks** - The conversations are about finishing and planning tasks. Some of these phrases include “done”, “finished”, “I’m still working on it” or “I can get it done”. However, because the task is usually described as “it” or “stuff”, it is not always clear from the audio which story card the person is specifically planning to finish or is already done. It is unfortunate, but without some way of associating the word “it” with the story card, extrapolating what the person meant by “it” or “thing” is difficult and is currently considered outside the scope of this research.

3.6 Summary

In this chapter, I have discussed some of the characteristics observed from the daily scrum meetings in the EBE software engineering lab. Some of the problems identified are the background noise, soft voices, poor pronunciation, erratic speech speed, accents and disfluency. However, the narrow domain gives some predictability to the type of conversations. In the next chapter, I am going to present the experiments performed with the automatic speech recognition engines to determine the capabilities and limitations on transcribing spontaneous conversational dialogues.

4 Speech Recognition Engines

The audio recordings obtained from the meetings suffer from the two most notorious speech recognition problems: additive noise and *Lombard effect*. They suffer from variability in speech speed and disfluency. There is no ASR in existence that can successfully solve these problems yet. Therefore, the purpose of this chapter is to investigate whether a regular speech recognition engine made for dictation speech can be modified to work with spontaneous conversations. In this chapter, I present the tests performed on two ASRs and the rationale for why one of the ASR is chosen over the other. I present some of the limitations of the ASR and how we may improve the speech recognition accuracy for the meeting recordings.

4.1 Automatic Speech Recognition Engines (ASR)

To transcribe the meeting speech, several automatic speech recognition engines were tested. Based on the recommendations from previous research [Fo05], two products were chosen for further investigation: Dragon Naturally Speaking 9.0 [Dra] and Microsoft Speech SDK (SAPI 5.1) [Sa]. Table 2 describes the specifications for the different engines.

	Dragon Naturally Speaking 9.0	Microsoft Speech SDK 5.1
Platform	Windows	Windows
Training	Required	Required
SDK	Not Available	Available
Programmable language	Not programmable	C++, C#, VB.net, VB, Jscript
Cost	Commercial	Free
Accuracy rate claimed by the product developer	99% accuracy rate (1% word error rate) with 300,000 vocabulary size	90% accuracy rate (10% word error rate) with 65,000 vocabulary size
Foreign accents	Yes	No
Grammar	Only possible for US English accent	Yes

Table 2: Automatic Speech Recognition Engine Specifications

A standard Dragon Naturally Speaking software package does not have a SDK (Software Development Kit), which means a real-time interactive application is impossible to build with Dragon. A software development kit provides the programming interface to the software to develop applications using a particular programming language. While it is possible to purchase the SDK at a considerably higher price, it is not considered for the purpose of this research. However, it has an *Auto Transcribe Folder Agent*. A pre-recorded audio file can be dropped into a folder and the dragon engine produces a transcribed text file in a designated directory.

Microsoft Speech provides an SDK in .NET languages or visual basic. Because Microsoft Speech has an SDK, it can be used to build a real-time interactive speech application.

Based on another independent experiment performed, *Dragon Naturally Speaking* software application resulted in a superior recognition rate [Fo05]. Because the experiment was conducted in a quiet environment in a dictation style, another set of experiments was conducted to evaluate the potential for using this ASR for spontaneous speech recognition. The tests were performed regarding distance, noise handling, disfluency and speed.

4.2 Capability Test

The following section contains the data obtained to test the boundaries of the automatic speech recognition capabilities and the suitability of using the software for meeting transcription. The speech is spoken into the microphone at various settings. The success is measured by word error rate (WER), which is calculated by (S+D+I)/N. S is the number of substituted words; D is the number of deleted words; I is the number of inserted words; N is the total number of words in the speech. The higher the value, the poorer the speech recognition it is. A sample speech is spoken five times by one person after about 30 minutes of speech training. The sample speech is created to be

representative of some of the dialogs spoken by the participants in the daily scrum meetings.

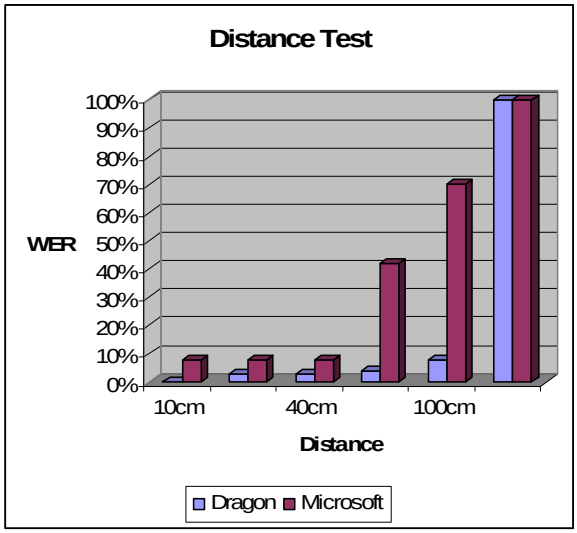
Sample speech: The test server is setup. The project server will be ready by Friday. The project is on schedule. I will double check the configuration as you requested.

4.2.1 Distance Test

The purpose of the distance test is to detect the maximum distance at which the recognition can successfully handle the speech recognition. The microphone is placed at a given distance away from the speaker. Table 3 shows the average word error rate at the given set of distances.

Distance from Microphone	Word Error Rate for Dragon Naturally Speaking	Word Error Rate for Microsoft Speech
10cm	0%	8%
20cm	3%	8%
40cm	3%	8%
60cm	4%	42%
100cm	8%	70%
200cm	100%	100%

Table 3: Distance Test



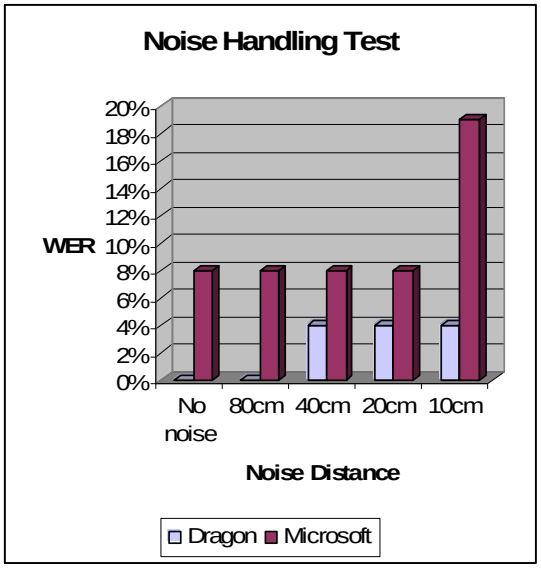
As the distance of the microphone and the speaker increases, the accuracy of the speech recognition decreased (errors increased). The microphone shouldn't be more than about 40cm away, which is about the distance of holding onto a microphone in a comfortable distance. For Dragon, as long as the surrounding environment is quiet, adding a little more distance between the microphone and the speaker doesn't degrade the result too much. However, Microsoft Speech word error rate increases drastically from about 40cm onward.

4.2.2 Noise Handling

The purpose of the noise handling test is to investigate the recognition success even with background noise. A fan noise using a regular household electric fan is placed at a set distance from the microphone. To avoid drowning the speaker's voice with the humming noise, a fan is placed below the microphone and the speaker. The humming noise is not loud enough to be picked up by the microphone, but a regular rotating sound of the blade in the background is distracting enough for the speech recognition engine. The noise placed at given distances and Table 4 shows the average word error rate.

Distance of the noise from the microphone	Word Error Rate for Dragon Naturally Speaking	Word Error Rate for Microsoft Speech
No noise	0%	8%
80cm	0%	8%
40cm	4%	8%
20cm	4%	8%
10cm	4%	19%

Table 4: Noise Handling Test



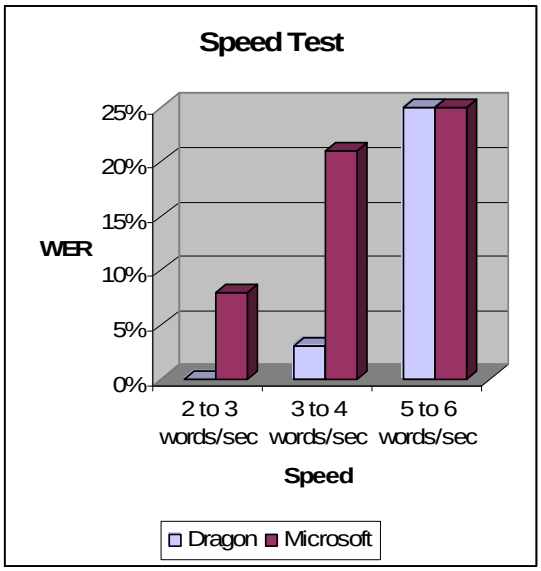
The humming noise doesn't distract the speech recognition too much as long as the sound is not drowning out the speaker's voice. For example, Dragon was able to successfully transcribe all of the sentences when the source of the noise is 80cm away. Microsoft performed equally well with only 8% word error rate. Even when the noise is only 20cm away, Dragon produced 4% word error rate and Microsoft produced 8% word error rate. As long as the speaker can speak louder than the background noise, the speech recognition can work fine.

4.2.3 Speed Test

The purpose of the speed test is to see how fast a person can speak before significant degradation can result. While the speech is spoken fast, the speech is very articulate. Table 5 shows the average word error rate.

The rate of the speech	Word Error Rate for Dragon Naturally Speaking	Word Error Rate for Microsoft Speech
2 to 3 words/sec	0%	8%
3 to 4 words/sec	3%	21%
5 to 6 words/sec	25%	25%

Table 5: Speed Test



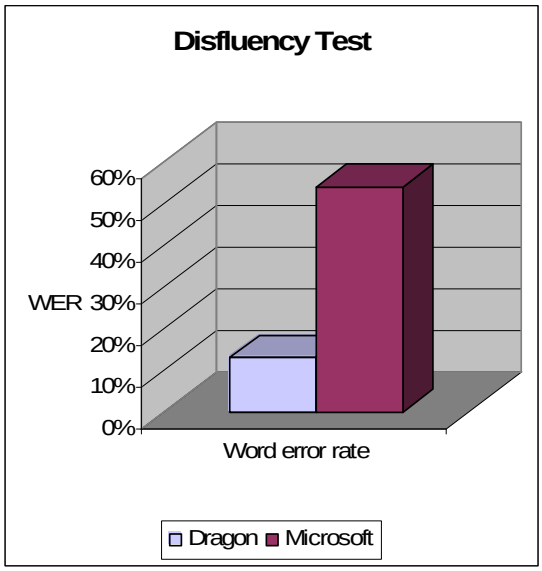
The best speed is about 2 to 3 words per second, which is about the speed of slow dictation speech. Dragon gives 0% word error rate and Microsoft Speech gives 8% word error rate. As the speed nears 5 to 6 words per second, the word error rate increases drastically as both speech recognition engines produced 25% word error rate.

4.2.4 Disfluency Test

"Um" and silence has been added into the speech at the end of every sentence. While there are other distracting words, only "um" is used to simplify the test. However, the speech was spoken in a dictation style with very clean pronunciations. The test attempts to investigate what the transcript looks like when distracting words such as "um" are added in between sentences. The same phrases are recorded five times with additional distracting words and Table 6 shows the average word error rate.

Word Error Rate for Dragon Naturally Speaking	Word Error Rate for Microsoft Speech
13%	54%

Table 6: Disfluency Test



Both engines could not deal with distracting words although Dragon was able to produce more accurate transcriptions. However, more important than the word error rate is the change in the meaning of the sentences. Because the speech recognition engine can't recognize "Um", it randomly chose a word that is close enough such as "I'm" or "not". Sometimes a random replacement for the word "Um" can change the meaning of the sentence completely.

4.2.5 Foreign Accent

The majority of the participants in the meeting have foreign accents and English is a second language. Dragon Naturally Speaking has the capability to handle many accents: US English, UK English, Australian English, Indian English and South East Asian English. However, many of the participants do not belong to any of these accent groups. While it is possible for people with foreign accents to train the Dragon system, they need to repeat several sentences. However, people with US English accent had little difficulty training the ASR. Even with the training, the ASR could not recognize the speeches with foreign accents very well. Microsoft Speech also had difficulty understanding speeches from foreign accents.

4.2.6 Grammar Test

The most common way to reduce the search space for speech recognition is to introduce *grammar* [YH+89]. As stated in Chapter 2.2, *grammar* is a way to reduce the number of possible words or combination of words to increase the performance for the ASR. The grammar has been designed to recognize only the sentences in the sample speech. Table 7 shows the average result of five runs.

Word Error Rate for Dragon Naturally Speaking	Word Error Rate for Microsoft Speech
0%	0%

Table 7: Grammar Test

As shown in Table 7, the accuracy rate is 100% when there are only four sentences to recognize that are in the sample speech.

4.3 Choice of ASR

Given the above analysis, the *Dragon Naturally Speaking* speech recognition engine is chosen for our application. There are advantages and disadvantages for this choice. Despite the disadvantages of using the *Dragon Naturally Speaking* ASR, the system outperforms other engines.

4.3.1 Advantages

- Has a better accuracy rate than the other ASR
- Seems to withstand various untrained situations better than other ASR
- Pre-recorded audio recordings can be transcribed afterward

4.3.2 Disadvantages

- Cannot build an interactive system because of the lack of SDK
- It is not speaker independent. We must train the speech profile for each user who will be using the system
- Cannot create grammars for foreign accent speech profiles

From this section onward, ASR refers to the *Dragon Naturally Speaking* speech recognition engine.

4.4 Transcribing the Real Meeting

As shown in Chapter 4.2, speech recognition can give a relatively high accuracy rate with very clean dictation-styled speech. However, the results in Chapter 4.2 shows that accuracy decreases with distance of the microphone, background noise, disfluency and foreign accents. In the following section, I present the result of transcribing spontaneous conversational speeches recorded from live daily scrum meetings.

The goal of the thesis is to make the transcription work for the speeches directly out of the meetings, which means transcribing these recordings by humans may be hard even for humans. For example, some speeches are very fast or disfluent. While some people may speak clearly, it is not always true with discussions and arguments happening amongst the participants in the meeting. From the several meeting recordings, I have selected five speeches that had relatively long monologues. It is difficult to describe how spontaneous conversational speeches sound like other than it is sloppy, fast, disfluent and contains big variations in the tone. The nature of spontaneous conversational speeches makes transcription hard because these speeches are quite fast and sloppy although humans hardly notice it during the actual meeting.

The recording has been separated by each speaker so we can transcribe the recording using the appropriate speech profile. The speeches with heavy foreign accents are not used and only five samples that had a relatively better audio quality are used for the experiments throughout the thesis. The samples contain both male and female speakers.

The main problem with face-to-face conversational speeches is that the speech seems alright when a human listens to it during the actual meeting. However, because it is easy for humans to decipher these speeches, we don’t realize the amount of problems that these conversational speeches present for the ASR. As the following results show, the

amount of error is huge and almost none of the sentences are recognized correctly. The test results are presented in Table 8. A human transcription is completed to compare the results from the ASR. The matching parts are highlighted to show which parts were successfully transcribed by the ASR. Names are deleted for privacy.

Speech	Human Transcription	Resulting Transcription from ASR
Sample1	I wrote up that script… that I shown you this morning… about the keyword extraction… and **I don't** really have plans anymore… (Faint laugh) Um… Yes.	He links. **I don't** know how and.
Sample 2	(Everyone laughing) Ok. We have (Name) laughing. That's the first sound. Ok. Um. What's it, **Tuesday**? Um. **We did that**. Did I talk, Did I do this experiment with, yeah, three dogs, yeah, experiment with that sound didn't come out right. (Bang) But um… Um. **I want to** get the dog to… …to **look at each person** in turn. Just move the head. Um. I could do that or I could have everyone sitting like this. **I could have everyone sitting at this** kind of thing and have the dog move to predefined points, um, ok, cause.. I've been looking at the way the **energy sound**, **energy level** at ears go and it's random. There's so much noise and what not that you can't move around a lot.	First, they come with **Tuesday** time **we did that**, in his or her to read all that with a crime. **I want to** give a dollar to **look at the person** you are and smaller or mechanical him to do that or **I could have everyone sitting at this** everyone's been disgusting at the dough, if I is for, and I've been looking at a delete **energy sound energy level**. You're still a personal choice and a
Sample 3	Now I'm feeling much better now. I am twenty to half an hour away from having	When are you from having to be in this shift is

	the edit part displayed inside the plugin. I have null pointer exception (mumble) and other minor things. So I should be able to get done by Monday	
Sample 4	Everything's due on Wednesday for (Name)'s course. So I'm trying to get all that stuffs done first. And then basically I have from **Thursday afternoon** until following Wednesday, Wednesday afternoon to get all the (Project Name) stuff done. Um. I started looking at the **videos** from the first couple of iterations. Yeah, But, pretty much everything's on hold right now. (mumble)	Or is the **Thursday afternoon**, only as a friend and a **video** at us and
Sample 5	I tried out few face recognition open source programs. None of them are really working working very nicely. Can't even **recognize** my own face. **So I gave them** about fifty different training data, but still can't find my face in there. So.	This program is none of the in a very I think has been **recognized** by and **so I gave them** about in attaining a I so

Table 8: The transcript from the ASR with original vocabulary

The results in Table 8 shows, the ASR is incapable of transcribing the spontaneous conversational dialogues. The disfluency is too great. The speech is too fragmented and contains too many sloppy pronunciations. As Table 8 shows, the resulting transcripts don't contain enough information to extract useful summaries and much of the content is meaningless. The transcription process also takes between 5 minutes to 1 hour. Sample 4 takes a little over 1 hour to transcribe mainly due to the file size.

The transcribed sentences by the ASR in Table 8 don't make sense semantically due to the wrong words and fragmentations. It would be inappropriate to simply use an extraction technique on the transcribed text to generate summaries. The summaries would have the same mistakes that the transcription had.

4.5 Summary

The results in Table 8 have shown that speech recognition engines do not perform well for spontaneous conversational speeches from the meetings. However, as the experiment in Table 7 shows, *grammar* can improve the speech recognition. Because the speed, disfluency, distance or background noise cannot be controlled, the only way to improve the speech recognition is by reducing the speech recognition domain. However, before we discuss the improvement of the speech recognition transcription process, which is discussed in Chapter 6, we need to perform experiments on the summarization tools. In the next chapter, I present capabilities and limitations for text summarization tools.

5 Summarization

As the literature survey in Chapter 2 shows, most of the summarization techniques are based on extractions. Previous research on summarization software has shown that there are several extraction-based summarization software systems available [Ji05]. To summarize the result from [Ji05], the summarizer software tends to include conversations by a person who talks longer or talks most frequently; the summarizer tends to miss shorter sentences by a person who doesn't speak much although it contained very important contents. The data in [Ji05] shows that results from different summarizers don't have significant differences. Therefore, only one summarizer is used to make the points about why generic, extraction-based summarization software is unsuitable for our purpose in this thesis. In this chapter, I present a summary produced by Microsoft Word on human transcription and ASR transcription. I will compare the results and analyze why the tool is unsuitable.

5.1 Extraction using Human Transcription

Here is the transcript that is previously presented in Chapter 3 as transcribed by a human.

(Previous conversations omitted)

Scrum Master: Ok. Good. You are healthy again.

P1: Now I'm feeling much better now. I am twenty to half an hour away from having the edit part displayed inside the plugin.

Scrum Master: Ok

P1: I have null pointer exception and other minor things. So I should be able to get done by Monday

Scum Master: Twenty minutes?

P1: I gotta get the cards and everything displayed.

Scrum Master: Any obstacles?

P1: Not yet.

Scrum Master: What did you do since Tuesday?

P2: I wrote up that script, that I shown you this morning. about the keyword extraction. and I don't really have plans anymore. Um. Yes.

(The rest of the meeting conversation omitted.)

5.1.1 Test 1: Feed the Entire Transcript

The above transcript is given in its entirety and summary is produced using Microsoft Word summarization capability. Microsoft Word allows the original text to be compacted by a specified amount. For example, 25% *length of the original* means only 25% important sentences from the original text is produced as a summary. Here is the summary produced with the compaction set to 25% of the original transcript.

Scrum Master: Ok. **Scrum Master:** Ok
Scum Master: Twenty minutes?
P1: I gotta get the cards and everything displayed.
Scrum Master: Any obstacles?
about the keyword extraction. Um.

There are two problems with the above summary.

1. It didn't extract the name of the person, who spoke the fifth line. Instead, the fifth line looks like it was spoken by the scrum master.
2. Because the word "scrum master" occurs most frequently in the text, it ended up extracting speeches by the scrum master mostly. However, the scrum master's speech is less important because he/she generally serves as a moderator for the meeting rather than contributing information to the meeting.

5.1.2 Test 2: Separate the Transcript into Individual Participants

As Test 1 shows, script-styled documents are unsuitable for Microsoft Word summarization. To avoid the above problem, the speech from each individual has been transcribed separately and each individual's speech has been compressed to 25% of the original text. I have fed the speech by the participant 1, produced the summary and then fed the speeches by the participant 2. The scrum master's speech has been removed because his/her role is mostly a moderator.

P1: I have null pointer exception and other minor things. I gotta get the cards and everything displayed.
P2: about the keyword extraction. Um.

Here is the result at 50% of the original transcript for each individual.

P1: I am twenty to half an hour away from having the edit part displayed inside the plugin. I have null pointer exception and other minor things. I gotta get the cards and everything displayed.
P2: about the keyword extraction. and I don't really have plans anymore. Um.

Based on the result shown above, the extraction technique works well on manually human transcribed texts. An extraction separated by each user and word length between 25% and 50% of the original speech can catch most of the important sentences.

5.2 Extraction using Automatic Transcription

Here is the same speech transcribed by the ASR separated by each speaker. Even before the summarization happens, the transcript already lacks some of the critical information and contains too many speech recognition errors. Here is the transcript produced by the ASR.

P1: When are you from having to be in this shift is
P2: He links. I don't know how and.

Here is the above transcript summarized by Microsoft Word at 25% summarization setting.

P1:
P2: He links.

The summarizer completely ignored the transcript for P1 and didn't produce any summary. Likewise, due to the speech recognition errors, P2 also has meaningless summary.

Here is the summarization at 50% summarization setting.

P1: When are you from having to be in this shift is
P2: He links.

This time the summarizer decided to put down the entire text for P1 and the same problem from the 25% summary persists for P2. In any case, due to the poor transcript, the summaries also inherited all of the problems that existed in the transcript. There are three reasons why a generic summarization tool will not work for spontaneous conversations transcribed by the ASR.

1. The transcripts produced by the ASR contain too many transcription mistakes. The summaries inherited all of the problems that existed in the transcript, including the gibberish.
2. The summarization software that uses statistical methods gives more importance to the most frequently occurring words. However, spoken dialog is filled with repetitions. It is difficult to determine which words are important simply based on the frequency of the words. Often, the sentence containing computer jargon may have more important contents, but these words may occur less frequently.
3. Unlike written documents, the sentences in the transcript are incomplete and fragmented. The ASR often places the punctuation in wrong places. It is difficult to determine the sentence boundaries based simply on the punctuation.

Because the purpose of these summarization tools is to compress the original text, a summary from a transcript with too many transcription errors is only bound to create an incoherent summary. We need a method to enhance the coherency of the summary if we are to extract some meaningful sentences out of the ASR transcript. The transcript from the ASR is incoherent; therefore the purpose of the summarizer should be *improving the coherency*, not compressing the text.

5.3 Summary

In this chapter, I have shown the limitations with generic summarization tool for ASR generated transcripts. If there are too many transcription mistakes, most of these

mistakes are inherited down to the summary as well. Therefore, the purpose of the summarizer for spontaneous conversational speech should be improving the coherency, rather than compressing the text. In the next chapter, I present how my speech recognition and summarizer are implemented to improve the coherency of the overall summary.

6 Daily Scrum Meeting Summarizers

The following section describes the implemented system to improve the transcription and the summarization for the daily scrum meetings. In this chapter, I am going to introduce two prototypes that were implemented where one is designed for accuracy and the other one for conversational speech. Between the two prototypes, one sacrifices the users' ability to determine how they want to speak and the other sacrifices the accuracy. I will explain how the two prototypes are designed in this chapter. The experimental results and their analysis are shown in Chapter 7.

6.1 Experimental Prototypes

As shown in Chapter 2, several research institutions are pursuing meeting summarizers but are experiencing difficulty in making them work for real life situations. These research prototypes only work with limited scope in restricted lab settings. Likewise, the expectation for my research is also not in producing a successful transcription and summarization tool for the spontaneous conversations in the daily scrum meetings. But rather, an interesting research question is to explore the dilemma that exists in giving the users the freedom in speech style and the effect on the accuracy of the summary. The more conversational the speech gets, the worse the accuracy of the transcription becomes. In other words, the more the computer gives freedom to the users in speech interaction, the harder it becomes to understand the contents.

The purpose of this thesis is to investigate what is possible with the current state of technology. Instead of building an entirely new speech recognition and summarization system, I want to investigate what is possible if I were to use the tools already available. Given the results in Chapters 4 and 5 on the state of speech recognition and summarization tools and the environmental problems stated in Chapter 3, there are several factors contributing to the difficulty in creating a meeting summarizer. The first problem is the difference between conversational speech and dictation speech, as I have emphasized much in Chapter 3 and 4. The accuracy of the transcription decreases

drastically as the speech starts to sound more like a conversation than a dictation. Even the speech recognition software makes no claim about speech recognition success on conversational speech. Second, computers don't actually understand the contents of the conversation. Conversations, even in structured meetings like daily scrum meetings, are too relaxed in structure for computers to follow. A real time interaction is possible only through an interview style interaction where the computer asks a question and the user provides the answer in a carefully controlled interaction.

In this thesis, I decided to perform two experiments. Earlier in the research, I have developed an interview style prototype [PD+06]. It is not possible to operate this system in group settings or in actual meetings because the user is expected to provide a dictation-style speech to the microphone one person at a time after a question is asked by the system. It assumes that the user is only interacting with the computer and it is not designed to be used during actual daily scrum meetings. It had higher transcription and summarization success. However, the first prototype didn't fulfill one of the requirements as we wanted to produce a summarizer for spontaneous conversations recorded during real daily scrum meetings.

With the second prototype, I have attempted to create a system designed to work for real recordings from the daily scrum meetings. The second prototype was designed to work with spontaneous conversational dialogues with no restrictions on the speech. The system had no live interaction with the users as the transcripts are produced after the meeting. The second prototype can handle conversational speech, but had a very low accuracy rate. The improvements I have made can improve the result but fundamentally the improvement is not enough to offset the freedom I have given to the users in their speech style.

6.2 The First Prototype: Dictation style

In this section, I will describe the implementation detail of the first prototype, which is designed for dictation style speech in an interview type of interaction. I will first start

with the ideas underlying the design of the system, then the system architecture and the implementation details.

6.2.1 Underlying Idea

The first prototype was designed to experiment with how the interaction will work with clean dictation speech. Even though the ultimate goal is to analyze how the spontaneous conversations in daily scrum meetings work, the result from this prototype is to be used for comparison purposes with the results from the second prototype. In this prototype, the system asks three questions. After each question, the user is expected to give as clear dictation speech as possible. The user can't deviate from the scenario prescribed by the system.

6.2.2 System architecture

The system is comprised of the speech recognition engine (Dragon), text-to-speech engine (Microsoft SAPI) and the user interface. Figure 8 is the component diagram.

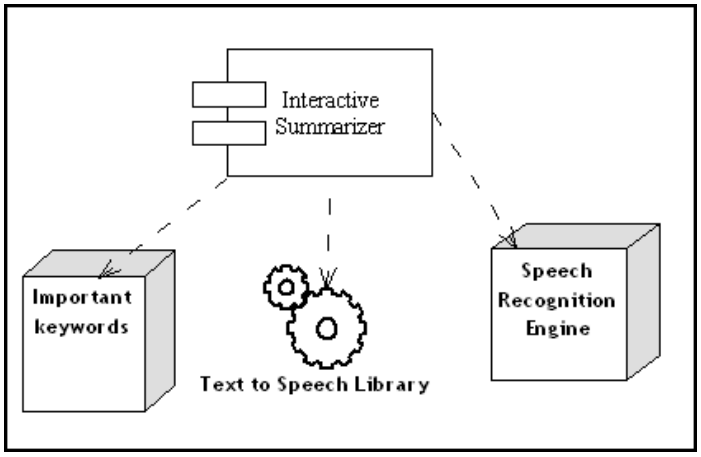


Figure 8: The components of Prototype 1

Figure 9 is the state diagram of how the prototype works.

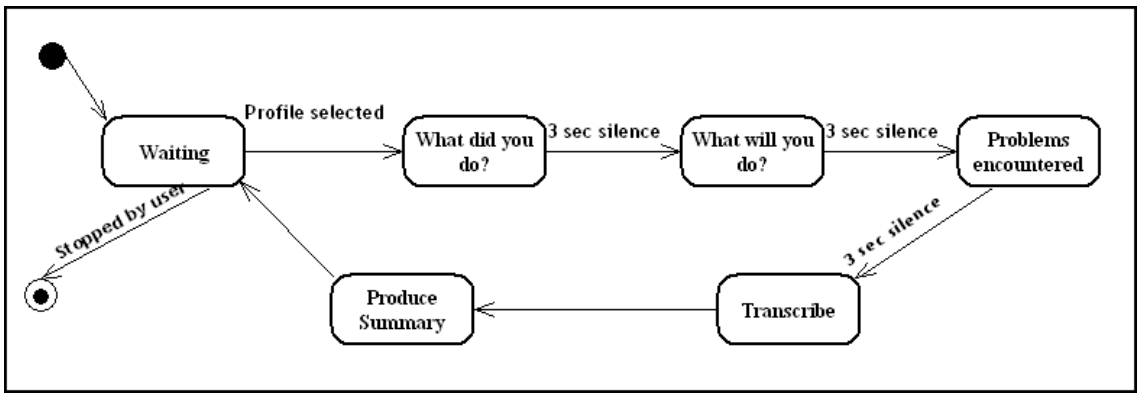


Figure 9: The state diagram of Prototype 1

Figure 10 is the class diagram of the prototype.

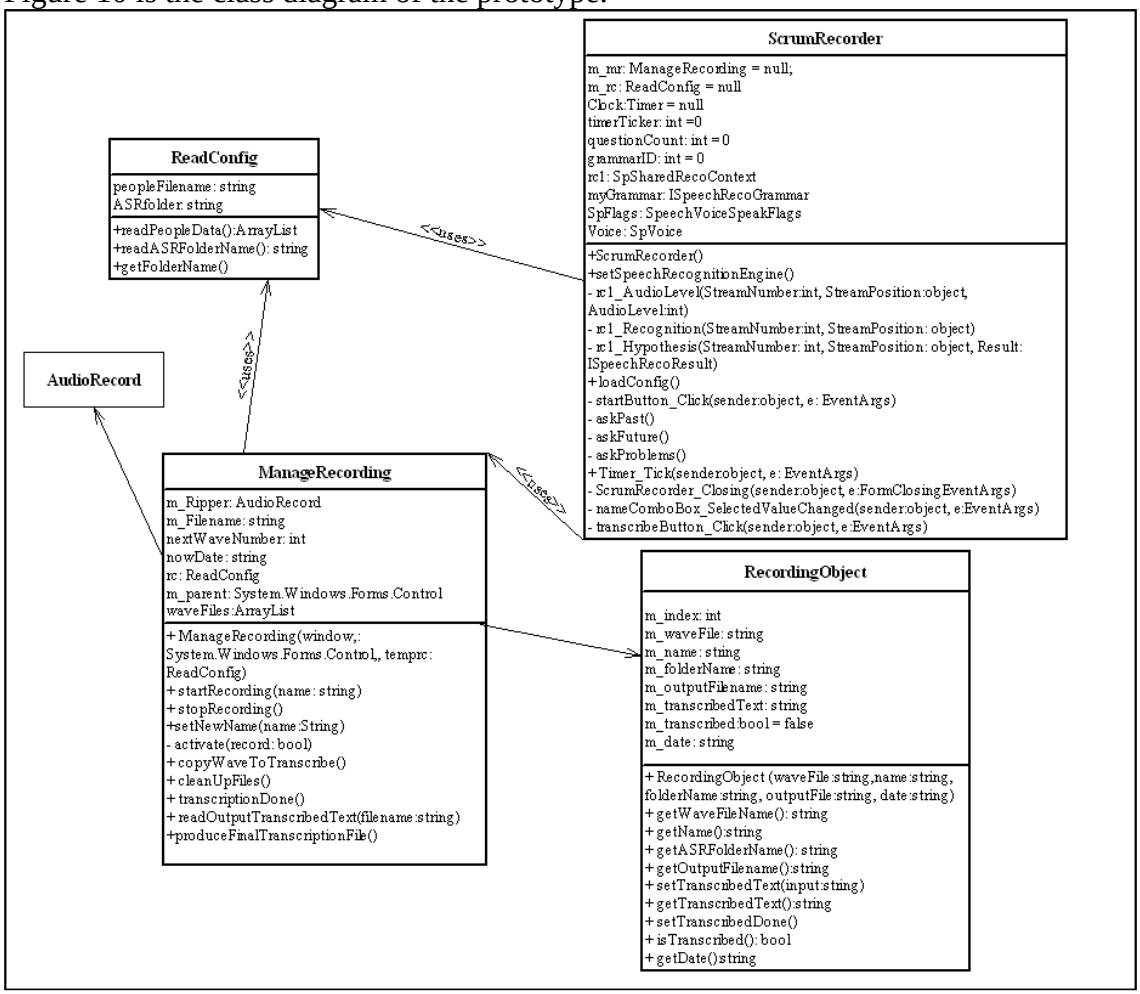


Figure 10: The class diagram of Prototype 1

Figure 11 is the picture of the user interface.

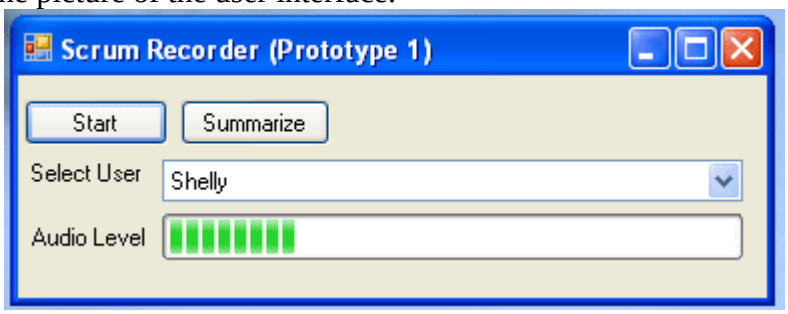


Figure 11: Prototype 1 User Interface

6.2.3 Implementation detail

Microsoft SAPI is used for text-to-speech but Dragon is used for the speech recognition. The system would verbally ask the user the following three questions in a row. What did you do? What will you do? And what problems did you encounter? When there is a three second pause, the system asks the next question. Once all three answers are received, the recordings are transcribed by the speech recognition engine and the transcripts are processed by the system to produce the summary. It extracts the transcribed sentences based on the important keywords to produce the summary.

Instead of relying on the statistical approach for finding important sentences, the extraction is based on a set of pre-defined important keywords. In a statistical approach, a word that occurs most frequently is considered an important word. Therefore, the sentence containing the greatest number of these frequently occurring words is considered an important sentence as seen in Chapter 5. However, in a conversational dialog, people may repeat words or fumble words. Frequency is not a good indicator for word importance. How I came up with the important keywords is explained in Chapter 6.3.6.

The speeches provided for the three questions are transcribed and presented in a summary under the three headings: What's done, What will be done, Problems encountered. Since the computer doesn't understand the meaning of these speeches, it

relies on the user that he/she will provide a clean dictation style speech with the appropriate contents. The user interface is simply a textbox and two buttons for recording and producing the summary.

Because of the lack of semantic understanding of the words spoken by the users, any intelligent interaction is difficult. The system is completely reliant on the user to provide the correct information and a pause is the only way to determine when the user has finished talking.

There are many possibilities for controlling the flow of the interview other than a pause. For example, the keyword can be spoken (for example “Next”) to proceed to the next question. However, the system sometimes recognizes the same keyword during the actual daily scrum meeting speech or misrecognizes a word and abruptly moves on to the next question in mid-sentence. Therefore, the only sure way to determine the flow is silence. A graphical user interface such as a button for proceeding to the next question is also possible, but this would only encourage the user to type up the answer rather than rely on the transcription system that may produce wrong recognitions. Using pause between interview questions is a very rigid interaction, but it gives more control for the system and a better accuracy can be obtained. The results from the first prototype will be shown in Chapter 7.

In the next section, I will explain the second prototype, which is an experimental prototype for the real meeting speeches without the rigid structural interruption from the system.

6.3 The Second Prototype: Conversational Meeting Speech

In this section, I will describe the implementation detail of the second prototype, which is designed for spontaneous conversational style speech. The system does not interact with the users and the transcripts are produced after the meeting is recorded. I will start

with the ideas underlying the design of the system, then the system architecture, the implementation details and the user interface.

6.3.1 Underlying Idea

The speech recognition accuracy is very poor with spontaneous conversational speech. As mentioned in Chapter 4.4, even the human transcriber had difficulty understanding the speeches in the audio recordings. Then the question is how do humans know what the person was trying to say even if he/she didn't hear all of the words? Unlike dictation speeches, listeners do not have to catch all words in conversational speeches to understand. Instead, the listener tries to extrapolate what the speaker was trying to say by picking up the keywords or key phrases. As shown in the research by Lindsay and O'Connell [LO94] and Ferber [Fe91], humans are not listening to all of the words.

Likewise, trying to get a verbatim transcript from an automatic speech recognition engine may be the wrong approach for spontaneous conversational speech. If we were to mimic a human's ability, the transcription process should not be a verbatim transcript of individual words. In other words, if the system hears enough keywords, it should extrapolate the utterance to the intended sentence that makes sense. To simulate this hypothesis, spoken sentences should have simple grammar structures and a smaller conversational domain as complex sentences can create too many guesses on intended meanings.

Even if such a prediction algorithm is implemented, one speech recognizer may not be able to catch all of the phrases in the speech. Therefore, I created two speech recognizers with two different capabilities, cooperating to hear the speech and share the contents to produce a better transcript. One recognizer tries to recognize the predefined phrases, which I call *phrase-based speech recognizer*, and the other should try to recognize all other generic utterances. In this system design, the phrases that could not be recognized by the *phrase-based speech recognizer* may be recognized by the generic

speech recognizer. As shown in Figure 12, the speech is processed by both types of ASR and both try to produce the summary based on their intended strengths.

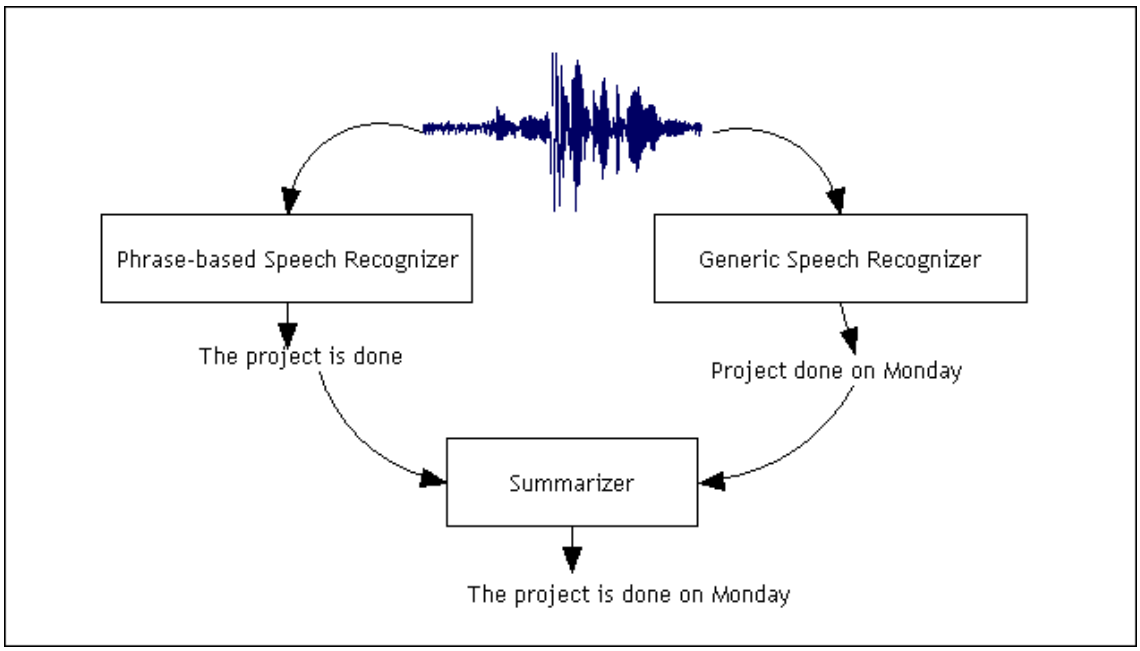


Figure 12: The Phrase-based and Generic Speech Recognizers both produce the transcript for the same speech recording

Before I explain how the *phrase-based speech recognizer* works, I need to explain *grammar*. A *grammar*, first mentioned in Chapter 2.2, is the most common way to reduce the search space for speech recognition [YH+89]. A search space for speech recognition is all of the possible recognizable words and sequence of words. *Grammar* for ASR is not same as the grammar in linguistic sense. Grammar is often used in command and control type of systems. Instead of relying on statistics to determine the words, it uses the *grammar* (which is a command template) to match the speech to the nearest command in the *grammar*. For example, let's suppose the automatic speech recognition is being used for an over-the-phone customer services application. The system could ask for the customer to say the phone number with the area code for further service. Because the system is expecting to hear only nine digits, the grammar

for the speech recognition would only contain single digits from zero to nine. Any speech that might not be a number would be ignored. If a sufficient amount of speech is unrecognizable based on the given grammar, then the system could ask the user to repeat [Ko03]. The same technique is used for voice controlled desktop applications. For example, the user needs to say “File” then “Print” to reach the print options or “Open Notepad” to open the specified application. These systems also use *grammars* to restrict the possible words that the system can recognize. Therefore, even with a relatively high background noise, the commands are still recognized fairly well.

6.3.2 System architecture

The system is composed of the speech recognition engine and the user interface. The user needs to specify two speech recognition profiles: the generic and phrase-based. The user interface is used to drop the recordings into the appropriate speech recognition folders. Once the transcripts are available in the predefined folder, the summarizer combines the transcripts and performs post-processing on the transcribed sentences to produce the summary. Figure 13 is the component diagram of the system.

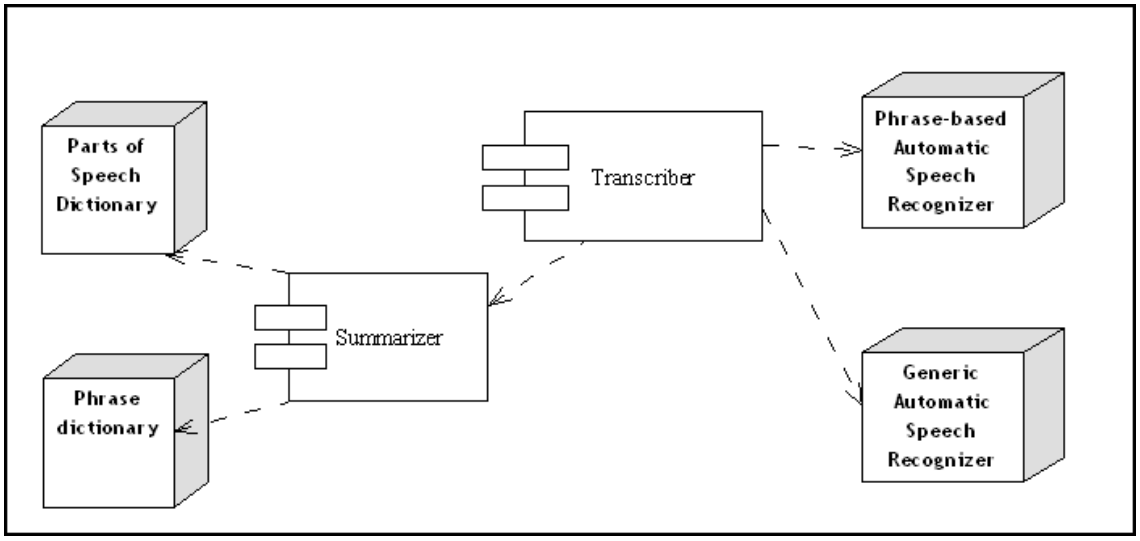


Figure 13: The components of Prototype 2

Figure 14 is the state diagram of the system.

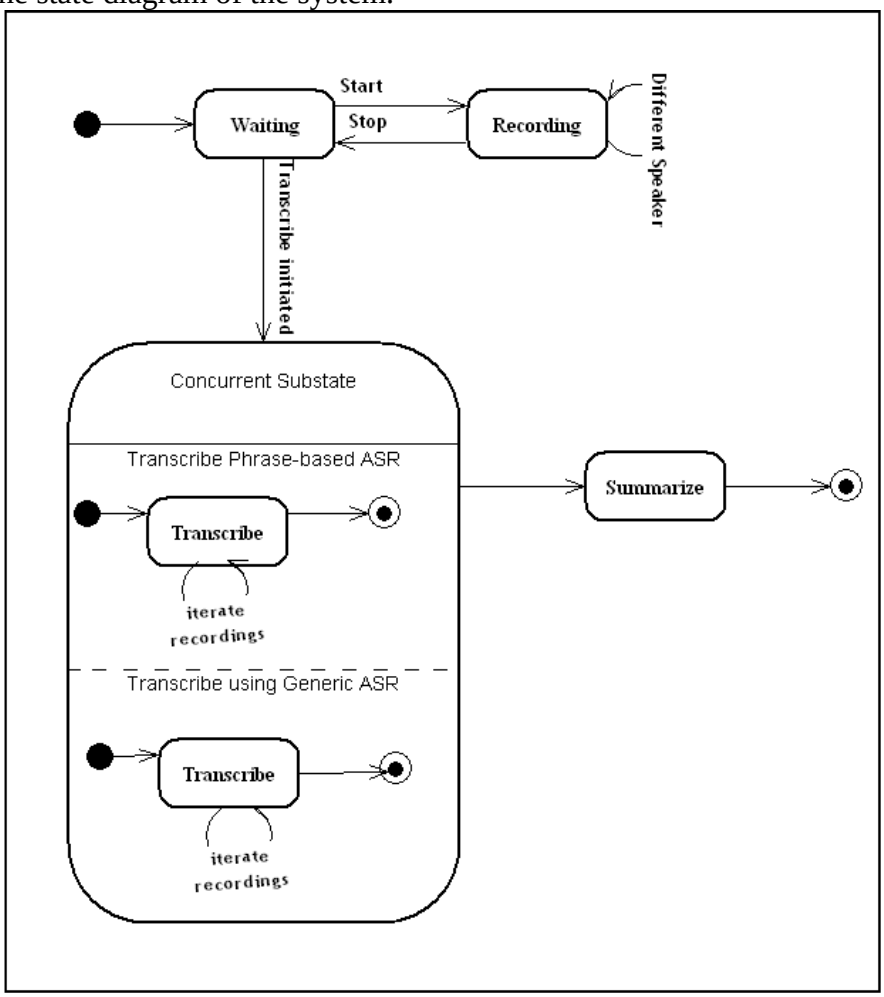


Figure 14: The state diagram of Prototype 2

Figure 15 is the class diagram of the system.

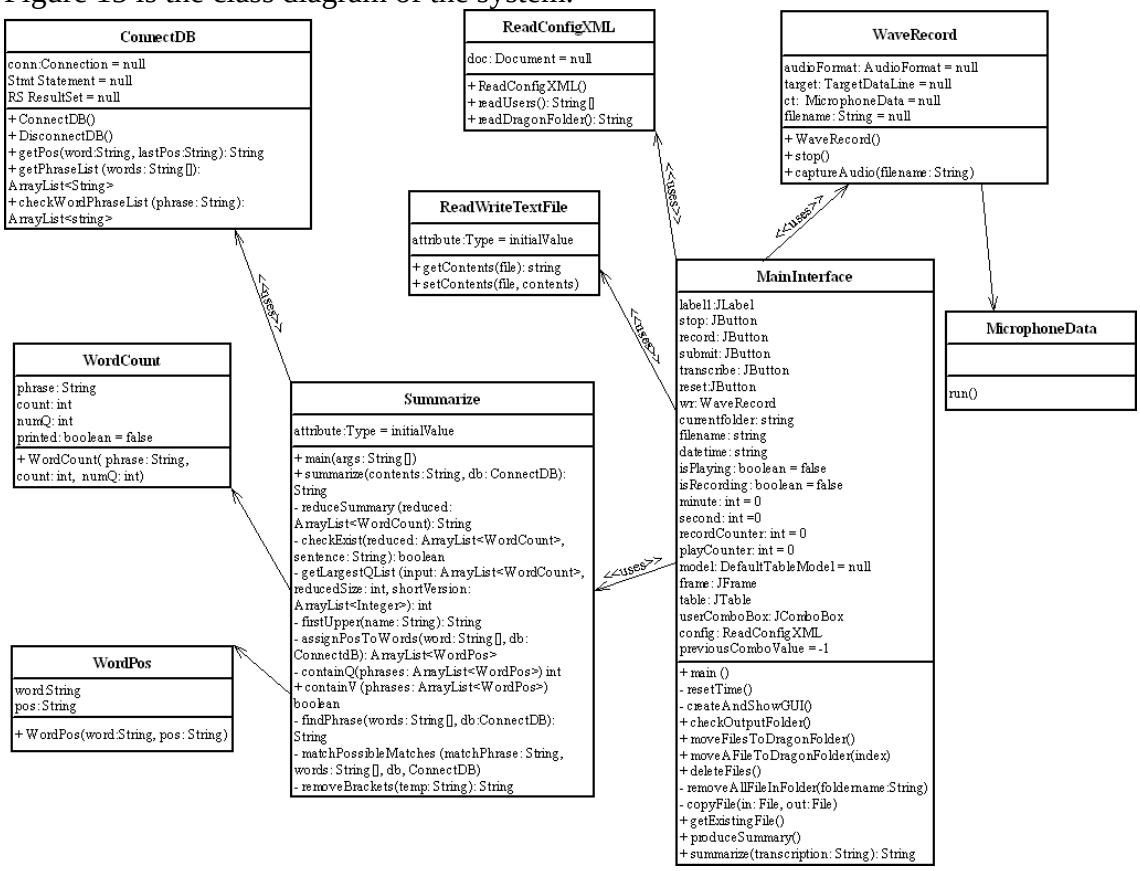


Figure 15: The class diagram of Prototype 2

6.3.3 Implementation detail: Phrase-based Recognizer

In this section, I am going to explain how the *phrase-based recognizer* is implemented. The purpose of the *phrase-based recognizer* is to match the speech with the closest pre-defined phrase. By restricting the possible recognizable phrases, there is a higher chance that the recognized speech may be more coherent and closer to the intended meaning.

As shown in the experiment in Chapter 4.2 and the results in Table 7, a *grammar* is the best way to improve speech recognition. The standard way people use grammar is to

give a series of individual words that can be recognized. In the command-and-control type of speech recognition applications, each entry in the grammar has a command. For example, an individual word such as “open”, “file” or “print”. Instead, I wanted to experiment by having the grammar hold series of phrases, thus reducing the chance of the speech recognizer mixing wrong words together.

To build the *grammar*, we need to have two lists in the ASR: a vocabulary list for the acoustic knowledge and a grammar list for the possible phrases. The original 400,000 vocabulary entries that came with the ASR are used for the vocabulary list. The terminology *grammar* for ASR does not have the same meaning as the grammar in linguistic sense. *Grammar* refers to a set of allowable words (or sequence of words) that the ASR can recognize. You can define more elaborate *grammar* using context-free grammar formulation, but it is rare that conversational speeches would follow some strict rules as disfluency can often distract the sentence flow. Because conversational speeches would follow much lose grammar structure, I have stayed with simply specifying a list of likely spoken phrases. The following paragraphs describe how the grammar is built.

Suppose G_1 is a *grammar* where

G_1 = {“what did you do today”, “I have a presentation”}

If the spoken phrase was closer to “what did you do today”, even if there are a few poorly recognized words, the recognizer will output “what did you do today” instead of “I have a presentation”. If the speech is sufficiently different from either of the two entries, it will ignore the speech and does not produce any output.

The entries in the grammar can contain partial sentences as a separate entry. For example, let’s suppose G_2 is a grammar that contains two entries.

G_2 = {“The project will be finished”, “by next Monday”}

If the above two phrases in G_2 are spoken together, a newly recognized sentence can be formed: “The project will be finished by next Monday”.

To build the *phrase-based speech recognizer*, several meetings are analyzed and the following types of phrases are built into the grammar. The phrases given to the *phrase-based speech recognizer* can be found in Appendix C.

1. **The phrases express due dates.** (Eg. “Everything’s due on Monday”, “I won’t start anything until Monday”, “I can get that done by Monday”, “Let’s have a meeting afterward” or “shown you this morning”)
2. **The phrases express completion of a task.** (Eg. “Then we are done”, “I’ve got some recordings”, “Next step for me would be”, “I tried a few face recognition” or “On the same machine it works”)
3. **The phrases express difficulties.** (Eg. “The error keeps coming out”, “haven’t changed the package”, “it says it can’t find the class” or “It’s not working”, “will not execute”)
4. **The phrases express errors and exception messages** (Eg. “IO exception, “data format exception”, “general security exception”, “compile error” or “parser error”)
5. **The phrases can be questions.** (Eg. “what do you think”, “what did you do”, “what happened” or “Do we really need the directory”)
6. **The phrases describe details about the project.** (Eg. “about this keyword extraction”, “JBoss server shouldn’t execute column fixtures”, “working on fitness” or “working on Alan”)
7. **The phrases are based on the story cards.** The phrases written on the story cards are not proper sentences (Chapter 3.1). It is also unlikely that people will articulate the contents written on the story cards in verbatim form. The user has

to make a few guesses about some of the possible ways the story cards can be expressed. For example, if the story card has “Add mouse pointer”, the entries in the grammar should be some variations. For example, “Adding mouse pointer is next item”, “Pointer is implemented” or “Can’t add the mouse pointer”, etc.

Because the system is being built using the recordings from the EBE software engineering lab, the recognizable word domain is set to Java, Fitness, Agile planner, Alan project and MASE. In total, about 600 phrases were manually typed to create the grammar for the *phrase-based speech recognizer*. The ASR allows building the grammar for US English accents only. When there are more story cards, more phrases should be generated into the grammar. However, there is a risk that more phrases will mean a larger search space. This method will become less effective with the increase in the search space.

6.3.4 Implementation detail: Generic Recognizer

As mentioned in the beginning of Chapter 6, there are two types of speech recognizers. Because the *phrase-based speech recognizer* only contains about 600 phrases, a generic speech recognizer is used to catch all other generic utterances.

The original ASR comes with over 400,000 recognizable terms. Larger vocabularies mean more possible combinations of words, which mean the probability of word error is also higher. To build the generic speech recognizer about 3,800 computer jargon terms and domain-specific words are manually entered. Some of these words include “runtime environment”, “system call”, “SQL exception”, “MySQL”, “look and feel”, “compile time”, “unsupported file exception”, “arithmetic exception”, “preliminary design”, “EJB”, “Visual studio”, “Eclipse”, “UML”, etc. The words are compiled around Java, testing and the Alan project, which are some of the projects that the participants were working on.

As mentioned earlier, a larger search space means more possibilities. Because there are already too many words in the vocabulary list, the newly entered 3,800 words are not always recognized and some other similar sounding words could be recognized instead. There are many closely sounding words, especially in a vocabulary list with 400,000 plus 3,800 computer jargon terms. Often, humans use semantics to distinguish differences between the different meanings but closely sounding words. However, computers are less effective in distinguishing the semantics of words, especially with unstructured phrases in conversational speeches.

6.3.5 Purpose of the Summarizer

In the previous sections, I have so far described how the two different types of ASR were implemented. Once the speeches are transcribed, post-processing is required to extract important sentences and extrapolate the intended meanings for the incoherent recognitions. The purpose of my summarizer is to perform the post processing on these transcripts to produce more meaningful sentences. The goal of the summarizer should be to address the following problems.

1. **Determine sentence boundaries**. The ASR puts commas and periods in the wrong places or omits them completely. Some utterances are not even a sentence. The summarizer should pick out only the utterances that are proper sentences. This is accomplished using a *parts-of-speech dictionary*. An utterance is considered a sentence only if it contains at least one verb.
2. **Incoherent sentences**. Due to wrong words and repeated words, the transcribed text can be incoherent. The summarizer would try to reword these utterances to make more sense. This is achieved using a *sentence- dictionary*.

In the following sections, I am going to explain the *parts-of-speech dictionary* and how it is used for finding the sentence boundaries and important sentences. Then the *sentence-dictionary* is introduced, which is used for placing the transcribed text into more coherent sentences. I am going to explain how these two components are used to fix the transcripts and produce a more coherent summary.

6.3.6 Implementation detail: Parts-of-Speech Dictionary

A computer relies on lexical analysis such as the location of punctuation to determine the sentence boundaries, mainly because of the lack of semantic understanding of the utterances. The ASR is designed to put punctuation whenever it encounters a pause in the speech, but people pause in mid-sentence or may not pause long enough at the end of a sentence. In a conversational dialog, the disfluency causes a thought to be broken among many utterances. A pause is not always a good indicator for determining what constitutes a complete thought.

While spoken dialogues don't follow a strict grammar, they still follow some loose grammar structures. For example, a sentence should at least have a verb. The sentences in conversational dialogues have relatively simple grammar structures, so an assumption can be made that a sentence will have at least one verb, which can be used to determine what constitutes a sentence. To find the verb in the sentence, a parts-of-speech dictionary is formulated.

A MySQL database is used with two columns: one column for the word and another column for the parts-of-speech. Table 9 contains the symbols used for the parts-of-speech.

The original file for the dictionary was obtained from [Mo]. The symbols in Table 9 are specified in the [Mo] file, but *Q* is defined by me. Except for the irregular verbs, [Mo] only contains the infinitive form of the verbs. Therefore, all verbs are given additional entries for present tenses and past tenses by adding inflections such as "ing" and "ed" at the end of the words. Because nouns also contain only the singular forms, additional entries are given for the plural forms by adding inflections such as "s" and "es" at the end.

Computer jargon terms are obtained from [Fo] and they are given the symbol *Q* in the dictionary. In addition, the dictionary has been manually adjusted to denote 11,609 important keywords. These manually compiled keywords range across computer jargon, words from story cards and some frequently occurring words that may signify importance.

A word can have multiple parts-of-speech based on how and where the word is used. Therefore, an entry could have multiple symbols assigned to it. For example, “compromise” can behave as both noun and a verb. Therefore, the parts of speech column for “compromise” would say “NV”.

Symbol	Parts of Speech
N	Noun
P	Plural
H	Noun Phrase
V	Verb
T	Transitive Verb
I	Intransitive Verb
A	Adjective
V	Adverb
C	Conjunction
P	Preposition
!	Interjection
R	Pronoun
D	Definite Article
I	Indefinite Article
O	Nominative
Q	Important word for summarization

Table 9: The parts-of-speech Notation

In total, over 458,000 words are added to the parts-of-speech dictionary. The dictionary is large enough to contain most of the frequently used words in the English language. If a word is not included in the database, then it is most likely a noun (eg. name of a person, name of a project, etc.)

The purpose of the *parts-of-speech* dictionary is to find the sentence boundaries and important sentences. The transcribed texts from both the *phrase-based ASR* and the generic ASR are parsed into sentences using a period, semicolon, colon, dash and question mark. Then each word in the sentence is tagged with parts-of-speech. Any word acting as a verb is tagged with *V, I* or *T*. If the sentence contains at least one verb, the sentence is kept. Otherwise, the sentence is discarded.

Once all utterances that may not be sentences are discarded, the remaining sentences are ranked based on their importance. If the sentence contains any word with *Q* in the parts-of-speech dictionary, the sentence is given an importance ranking score of one point per appearance of the keyword. For example, “code” is considered an important word in the parts-of-speech dictionary; therefore, the sentence with “code” will get one score per appearance of “code”. The sentence “The code is ready” will receive one importance score because it has one appearance of the word “code”. The importance score is used in determining which sentences should be included in the summary when the summary exceeds the word limit.

6.3.7 Implementation detail: Sentence-Dictionary

Even if the sentence boundaries are identified and the sentence importance is ranked, the actual contents of the sentences may be incoherent. While the *phrase-based ASR* could produce more coherent phrases than the generic ASR, it still puts punctuation in the wrong places resulting in run-on or fragmented sentences. I have contemplated discarding incoherent transcripts, but as you will see in Chapter 7, discarding all of the incoherent sentences may result in a blank summary due to the difficulty in transcribing

conversational dialogs. To salvage some utterances from the transcript, the system has to perform an intelligent extrapolation to guess what the person might have said.
Because the domain of the summarizer is very small, a *sentence-dictionary* was developed. It contains some of the likely sentences that people would speak. By using the knowledge about the domain, about 600 pre-defined phrases were created for the database. The sentences in the *sentence-dictionary* are listed in Appendix D.

As the experiment by [Ch75] has shown in Chapter 2.7, humans can intelligently extrapolate what the sentence might have said just from a series of keywords. Likewise, the summarizer should be able to extrapolate what the speech might be saying by matching the series of keywords. The *sentence-dictionary* contains a relation between a sentence and series of keywords. If all of the keywords are found in the utterance, the system should replace the utterance with the given sentence. Currently, the order of keyword appearance in the sentence does not matter. The decision is based on the assumption that the sentences are simple and that the order of keywords will not impact the intended meaning of the utterance. Let's suppose there are four entries in the *sentence-dictionary*.

Sentence-Dictionary = { (I, am, finished) -> I am finished,
(project, finished) -> The project is finished,
(due date, Monday) -> The due date is Monday
(test, all, file, under, directory) -> We tested all files under the directory
}

Let's suppose that we encounter a transcribed text that says "We, you know, tested all of the files under that directory". It contains the keywords, "test", "all", "file", "under", and "directory", which matches with the entry in the *sentence-dictionary*, "*we tested all files under the directory*". Since the transcribed text contains all of these five keywords, the text is replaced with the *sentence-dictionary* entry. Instead of "We, you know, tested

all of the files under that directory", the summary would say "We tested all files under the directory". As a tie breaker, if there are multiple entries with the same number of keyword matches, the entry with the longest sentence is chosen. While they probably contain similar contents, it assumes that the longer entry has more details that could improve the summary compared to the shorter entry. This assumption could be wrong depending on the actual contents of the sentence, most of the entries are vague enough that it would not change the meaning too much.

Creating the *sentence-dictionary* was a very labor intensive process. The dictionary had to be developed manually as computers do not know which word must be a keyword. In order for this method to work, a very large human-processed knowledge base had to be formed. Because the computer cannot understand the semantics of the sentence, it relies on this human processed dictionary.

Because the *sentence-dictionary* contains about 600 sentences, not all utterances can be matched. In such a case, a verbatim copy of the transcribed text is included in the summary until the word limit of the summary is reached.

6.3.8 Producing the Summary

At this point in the summarization process, the transcripts from both types of speech recognizer are combined and the sentence boundaries are identified. The sentences are given rankings using the importance score, and utterances are replaced with more coherent sentences from the *sentence-dictionary*.

The sentences are categorized into three pools: (1) the sentences replaced by the *sentence-dictionary*, (2) the sentences that are considered important by the parts-of-speech dictionary but not replaced by the *sentence-dictionary* and (3) the rest of the leftover sentences. Figure 16 shows the three types of sentences and how these sentences are picked to be included in the summary in decreasing importance order until a 100 word limit is reached. Having more entries from Level 1 (sentences from the

sentence-dictionary) tend to be more coherent as the sentences in the other levels may be incoherent.

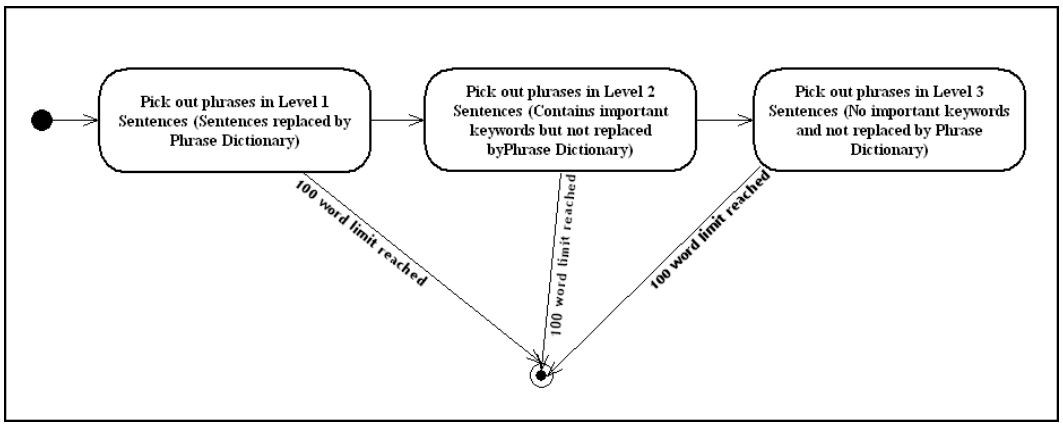


Figure 16: Rank the sentences and produce a summary

The summary is given 100 words limit per speech. From our observations, people were able to convey most of their messages within about 100 words. Anything more than 100 words comprised incoherent utterances. Once the summarizer picked out the sentences to 100 words, the sentences are laid out in the order they were spoken.

6.3.9 User Interface

Finally, the user interface allows the user to record and arrange the audio files to produce the transcript and the summary. The main purpose of the user interface (Figure 17) is to help with speaker identification. The user interface allows the human controller to select who the current speaker is while the recording is in progress.

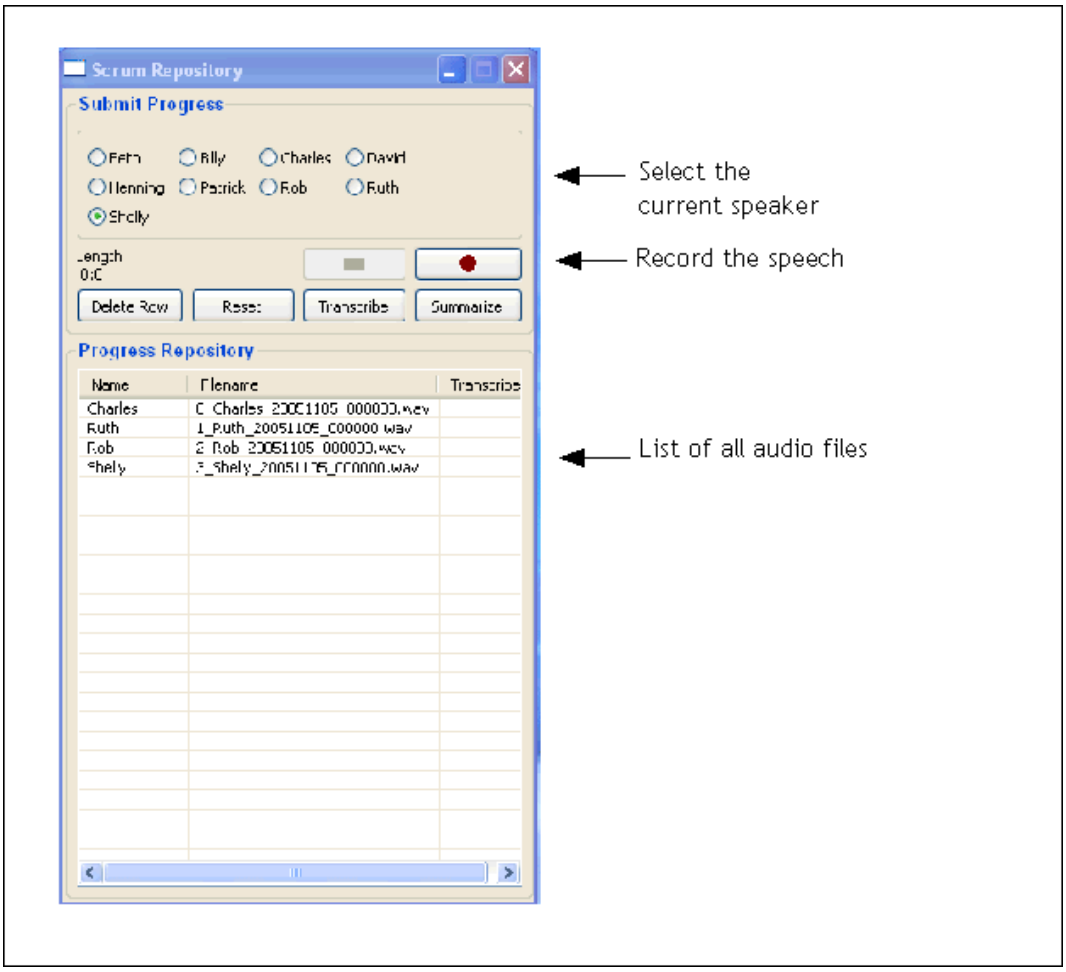


Figure 17: Prototype 2 User interface

The information about the intended users is specified in the *config.xml* file. Here is how a *config.xml* file looks. It is located in the same folder as the executable program.

```
<scrum>
      <dragonfolder>C:\\Documents and Settings\\shelly\\My
Documents\\DragonAutoTranscribefolder</dragonfolder>
      <user>Beth</user>
      <user>Billy</user>
      <user>Charles</user>
      <user>David</user>
      <user>Henning</user>
      <user>Patrick</user>
      <user>Rob</user>
      <user>Ruth</user>
      <user>Shelly</user>
</scrum>
```

The *dragonfolder* specifies the path to the root folder where all the audio files should reside. The root folder should contain folders called *Storage, Output, Temporary* and two folders for each speaker (Eg. Shelly and Shelly2). Using the Dragon's *AutoTranscribe Agent* interface, the speech profile must be mapped with the speaker's folders. For example, the 'Shelly' folder is mapped to the generic ASR profile and 'Shelly2' is mapped to the phrase-based ASR profile.

Audio files recorded using any other recording devices can be deposited into the *Temporary* folder directly and they will appear on the user interface table in real time. However, the file name should follow a naming convention: number denoting the speaker's turn, underscore, name of the speech profile to use, underscore, today's date and time, ".wav" for the file extension.

When the *transcribe* button is pressed, the wave files are automatically transferred to the designated user's folders as specified in *config.xml*. The ASR agents will automatically pick up the audio files one at a time and deposit the corresponding text files into the *Storage* folder. Depending on the file size of the wave file, the transcription can take anywhere from 5 minutes to 1 hour, which is one of the reasons why real-time interaction is impossible using Dragon. Each audio file will have two transcripts in the *Storage* folder: one from the phrase-based ASR and other from the generic ASR. The interface will indicate *Done* beside each recording after the transcript has been produced. When the *Summarize* button is pressed, the two transcripts per user in the *Storage* folder are read and processed according to the algorithm specified in this chapter.

6.4 Summary

In this chapter, I have described two prototypes. The first prototype uses dictation style speech and simply relies on the assumption that the transcripts have very little transcription errors. The purpose of the second prototype is to work with conversational

speeches recorded from the real daily scrum meetings. For the second prototype, I described how two speech recognizers are developed and the implementation details of the summarizer. The goal of the summarizer is to find the sentence boundaries, rank the sentences and replace the utterances with the entries in the *sentence-dictionary*. In the next chapter, I will present the results from the summary, measure the improvements and analyze the limitations of the methods presented in Chapter 6.

7 Analysis

In this chapter, I present the results of the summaries produced from the prototypes presented in Chapter 6. First, I explain how I obtained the samples and describe the characteristics of these recordings. I then present the resulting transcripts and the final summaries. Finally, I will compare the results from the two prototypes as well as with the human-generated summary and explain strengths and weaknesses of the strategies presented.

7.1 Obtaining the Sample Recordings

The three recordings were obtained from real daily scrum meetings. Each meeting lasted about 10 to 15 minutes. However during the time of this research, which is between September 2005 and April 2007, some participants were no longer available and several speeches had to be removed from the samples due to the difficulty in obtaining the speech recognition training files. In addition, speeches with heavy foreign accents were also eliminated. The recordings with too many dialogs between speakers are also eliminated due to the difficulty in transcribing speeches that span across multiple speakers. In the end, five sample speeches are chosen. These speeches have few or no interruptions during the speech and the user carries out a monologue for about 2 to 3 minutes. Some of these recordings had additional conversations that followed after, but were eliminated due to the rapid dialogs occurring between people. As these are conversational speeches, all of the problems stated in Chapter 3 are present. To obtain the dictation speech, these five speeches are re-recorded using one person's voice in a dictation style. Generally, the spontaneous conversational speeches are much faster than the dictation speeches. The dictation speeches are intentionally spoken slowly with emphasis on every word for clearer speech.

These recordings were obtained using two different microphones. The dictation speeches were obtained using a Logitech noise canceling desktop microphone (980240-0403). Although a wireless Bluetooth microphone (Logitech Mobile Traveler Headset

and D-Link USB Bluetooth Adapter) has been tried, the sound quality is not good enough to produce a decent transcription. The meeting speeches are obtained from the robot (Aibo ERS-7), but there is no quality difference between the Logitech microphone and the robot's microphone. The speech recognition engines are trained for about 30 minutes by reading the default training text.

7.2 Transcription Result

Table 10 contains the transcription results from all of the transcription types presented in Chapter 6. The human transcription is added as the standard for comparison purposes. The transcript from the first prototype is based on a dictation style speech. The second prototype produced two types of transcripts: the phrase-based and generic. As shown in Table 10, the error rate is quite high for the transcripts produced by the second prototype because they are based on spontaneous conversational speech. The correctly matched words are highlighted in bold. The correctly matched words in the second prototype transcripts are highlighted in bold in the human generated transcript for easier comparison. If nothing has been produced by the speech recognizer, it is denoted using "(none)". For the dictation speech, the questions asked by the computer are enclosed in square brackets.

Sample	Transcription Type	Transcript
Sample 1	Human generated transcript	I wrote up that script, **that I shown you this morning**, about the **keyword extraction**. And **I don't really have plans** anymore… (Faint laugh) Um. Yes.
	Transcript from the generic ASR (Second prototype)	He links. **I don't** know how and.
	Transcript from the phrase-based ASR (Second prototype)	**I shown you this morning keyword extraction, I don't have plans.**

	Dictation speech (First prototype)	[What did you do?] **I wrote up that script that I shown you this morning about the keyword extraction.** [What will you do?] **I don't really have plans anymore** on **yes** [What problems did you encounter?]
Sample 2	Human generated transcript	(Everyone laughing) Ok. We have (Name) laughing. That's the **first** sound. **Ok**. Um. **What's it, Tuesday**? Um. **We did that**. Did I talk, Did I do this experiment with, yeah, three dogs, yeah, experiment with that sound didn't come out right. (Bang) But um… Um. I want to get the dog to… …to look at each person in turn. Just move the head. Um. **I could do that** or I could have everyone sitting like this. **I could have everyone sitting** at this kind of thing and have the dog move to predefined points, um, **ok,** cause. **I've been looking at the way** the **energy sound**, **energy level** at ears go and it's random. **There's so much noise** and what not that you can't move around a lot.
	Transcript from the generic ASR (Second prototype)	**First,** they come with **Tuesday** time **we did that**, in his or her to read all that with a crime. I want to give a dollar to look at the person you are and smaller or mechanical him to do that or **I could have everyone sitting** at this **everyone's** been disgusting at the dough, if I is for, and **I've been looking at** a delete **energy sound energy level.** You're still a personal choice and a
	Transcript from the phrase-based ASR (Second prototype)	Fine **We did that Okay I can do that There's so much noise What's it** You are motivated **Okay okay**
	Dictation speech (First prototype)	[What did you do?] **Okay we have** nothing**. That's the first to sound okay** on **what's it Tuesday** on. **We did** ask, **did I talk** the night duties **experiment with** now**, three dogs experiment with that sound didn't come out right**. [What will you do?] **But** on Palm **I want to get the dog to**

		to look at each person in time. Just moved ahead on **I could do that** war. **I could have everyone sitting like this**. **I could have everyone sitting at this kind of thing**, and half **the dog move to predefined points** on okay he **cause I've been looking at the way**. **The energy sound energy level.** [What problems did you encounter?] At year's goal, **and it's random there is is so much noise and whatnot that you can't move around** on lots
Sample 3	Human generated transcript	Now I'm feeling much better now. I am twenty to half an hour away from having the edit part displayed inside the plugin. I have null pointer exception and other minor things. So I should be able to get done by Monday
	Transcript from the generic ASR (Second prototype)	When are you from having to be in this shift is
	Transcript from the phrase-based ASR (Second prototype)	(None)
	Dictation speech (First prototype)	[What did you do?] **Now I'm feeling much better now. I am 20 two half an hour away from having the** eddied Bharata **displayed inside a plug-in**. [What will you do?] **So I should be able to get done by Monday** [What problems did you encounter?] **I have null pointer exception, and other minor things.**
Sample 4	Human generated transcript	Everything's due on Wednesday for (Name)'s course. So I'm trying to get all that stuffs done first. And then basically I have from **Thursday afternoon** until following Wednesday, Wednesday afternoon to get all the (Project Name) stuff done. Um. I started looking at the videos from the first couple of iterations. Yeah, But, pretty much everything's on

		hold right now.
	Transcript from the generic ASR (Second prototype)	Or is the **Thursday afternoon**, only as a friend and a video at us and
	Transcript from the phrase-based ASR (Second prototype)	(None)
	Dictation speech (First prototype)	[What did you do?] **Everything is due on Wednesday for** someone's chorus. **So I'm trying to get all that stuff's done first and then basically. I have from Thursday afternoon until following Wednesday. Wednesday afternoon to get all the project stuff done on. I started looking at the videos from the first couple of iterations**. [What will you do?] **but pretty much everything is on hold right now** [What problems did you encounter?]
Sample 5	Human generated transcript	I tried out few face recognition open source programs. **None of them** are really working working very nicely. **Can't even recognize my own face. So I gave them about fifty different training data**, but still can't find my face in there. So.
	Transcript from the generic ASR (Second prototype)	This program is **none of the** in a very I think has been **recognized** by and **so I gave them** about in attaining a I so
	Transcript from the phrase-based ASR (Second prototype)	**None of them Can't even recognize my face I gave about 50 different training data**
	Dictation speech (First prototype)	[What did you do?] **I tried out a few face recognition open source programs.** [What will you do?]

		[What problems did you encounter?] **None of them are really working working very nicely Can't even recognize my own face. So I gave them about 50 different training data, but still can't find my face in** their toll

Table 10: The transcript produced by the different prototypes

With sample 3 and 4, the *phrase-based speech recognizer* didn't produce any output. This means the speech didn't have enough prosodic features to confidently match it with any given phrases, and the ASR has completely ignored the speech. There are many factors causing the ASR to ignore the speech, including the background noise as described in Chapter 3.3 and a mismatch in prosodic features such as variations in loudness and speed.

7.2.1 Dictation versus Spontaneous Conversations

From the observation of Table 10, there is clearly a difference in recognition accuracy between the dictation speech and the conversational speech. The accuracy rate is much higher if the user consciously pronounces all words clearly and with little variation in tempo or loudness of the speech. The words in dictation speeches match better with the acoustic knowledge in the speech recognition engine, therefore a better transcript can be produced. However, with a spontaneous conversation, there is a high degree of variation in the sound and the tempo of the speech; therefore, the mismatch between the spoken words and the acoustic knowledge in the speech recognition engine can occur, resulting in a high error rate.

As mentioned in Chapter 2, researchers are trying to find a way to match the conversational speech with the acoustic knowledge in the speech recognition engine, but so far with less success. Success depends much on the original speech style. Even if the speech is from a conversational dialog, a slower and cleaner monologue style speech has a better chance of getting a better transcript. For example, sample 3 and 4 are much faster than the other samples and produced more wrong and empty transcript contents.

The strategy shown in Chapter 6 demonstrates that using two types of speech recognition engine may produce a better transcript, because the phrases that could not be recognized by one may be recognized by the other. As shown in Table 10, the strategy is working because more phrases are obtained using this strategy, thus there are more phrases to work with. However, even using this strategy, there are misrecognized words embedded among the correctly recognized words distorting the actual meaning of the speech.

7.2.2 Limitations of the Second Prototype Speech Recognition

The strengths of using both the *phrase-based speech recognizer* and the generic speech recognizer for transcribing the spontaneous conversational speech is that it (1) improves the number of phrases that are recognized, and (2) recognizes the entire phrase rather than just one word at a time. However, the design also presents some limitations:

1. **Not all phrases are available in the grammar.** People will likely speak phrases that are more than what is available in the "*grammar*". The 600 phrases in Appendix C present a very small number of phrases compared to what can be spoken during the meetings.
2. **The *phrase-based speech recognizer* is slightly better than the generic speech recognizer, but still error prone.** The *phrase-based speech recognizer* can improve the coherency of the transcript, but exact accuracy is still difficult to obtain. For example, the speaker said "I wrote up that script that I shown you this morning, about the keyword extraction". The generic speech recognizer produced "He links". But the *phrase-based recognizer* can produce "I shown you this morning keyword extraction". The *phrase-based recognizer* was able to produce a better transcript.
3. **The *phrase-based speech recognizer* is not immune from variations in the speech:** Even if the spoken phrase is available in the phrase list, sloppy pronunciations, variations in speed or in loudness may cause the ASR to misrecognize the phrase or completely ignore it. For example, some phrases in

sample 3 and 4 are available in the phrase list, but the ASR decided to ignore it because it didn't match enough prosodic features.

4. **Punctuation:** The speech recognizers are having trouble with putting punctuations in the right places. As shown in Table 10, none of the output had any punctuation. It also shows that people do not pause long enough between the sentences in spontaneous conversations.
5. **One incorrectly recognized word can change the meaning.** The *phrase-based speech recognizer* can mistakenly match the utterance with the wrong sentence. For example, the speaker could say "I am not finished". But because the recognizer couldn't catch "not", it could match the speech with "I am finished". The meaning of the utterance has completely changed. The *phrase-based speech recognizer* is still not immune from sloppy pronunciation.
6. **Match with the wrong sentence.** If the *phrase-based recognizer* cannot match the speech with any grammar entry, the best action is to ignore the unrecognizable phrase rather than producing completely wrong transcript. However, it may try to match it with the closest grammar entry. For example, "It will be done" is in the phrase list, but the person said "It will be taken down". The ASR can mistakenly match the phrase with "It will be done", if there are enough matches of prosodic features. The meaning of the message has changed completely.

7.3 Summarization result

In the following section, I present the result of the summarization. Table 11 shows the summary for the five samples presented in Table 10. The human generated summary is presented for comparison purpose. The first prototype simply extracts sentences with important keywords. The second prototype replaces the phrases, ranks sentences and extracts the sentences as presented in Chapter 6.

Sample	Summary Type	Summary
Sample 1	Human generated	- I wrote up that script about the keyword extraction - I don't really have plans
	First prototype	What is done: **-** I wrote up that script that I shown you this morning about the keyword extraction. What will be done: - I don't really have plans anymore on yes Problems:
	Second prototype	- I don't really have plans.
Sample 2	Human generated	- Did I do this experiment with, yeah, three dogs - I want to get the dog to look at each person in turn - I could have everyone sitting - Have the dog move to predefined points - I've been looking at the way energy sound, energy level at ears - There's so much noise
	First prototype	What is done: -That's the first to sound okay on what's it Tuesday on. **-** We did ask, did I talk the night duties experiment with now, three dogs experiment with that sound didn't come out right**.** What will be done: - But on Palm I want to get the dog to to look at each person in time. - Just moved ahead on I could do that war**.** - I could have everyone sitting at this kind of thing**,** and half the dog move to predefined points on okay he cause I've been looking at the way**.** - The energy sound energy level.

		Problems: - At year's goal, and it's random there is is so much noise and whatnot that you can't move around on lots
	Second prototype	- There's so much noise. - What did I do. - The work is okay and I can get an okay result. - We have come on Tuesday and we tried the test. - We tried where you read all the text within a time. - It looks at a person and the head. - I want to give it all up.
Sample 3	Human generated	- I am twenty to half an hour away from having the edit part displayed inside the plugin - I should be able to get done by Monday
	First prototype	What is done: - Now I'm feeling much better now. - I am 20 two half an hour away from having the eddied Bharata displayed inside a plug-in. What will be done: - So I should be able to get done by Monday Problems: - I have null pointer exception, and other minor things.
	Second prototype	(None)
Sample 4	Human generated	- I have Thursday afternoon until following Wednesday, Wednesday afternoon to get all the (Project Name) stuff done. - I started looking at the videos from the first couple of iterations

	First prototype	What is done: - So I'm trying to get all that stuff's done first and then basically. - Wednesday afternoon to get all the project stuff done on. - I started looking at the videos from the first couple of iterations. What will be done: - but pretty much everything is on hold right now Problems:
	Second prototype	(None)
Sample 5	Human generated	- I tried out few face recognition open source programs. - None of them are really working working very nicely. - Can't even recognize my own face. - So I gave them about fifty different training data, but still can't find my face in there.
	First prototype	What is done: - I tried out a few face recognition open source programs. What will be done: Problems: - None of them are really working working very nicely Can't even recognize my own face. - So I gave them about 50 different training data, but still can't find my face in their toll
	Second prototype	I tried out few face recognition open source programs. None of them are really working working very nicely. Can't even recognize my own face. So I gave them about fifty different training data, but still can't find my face in there. So.

Table 11: The summary results for the sample speeches by the different prototypes

Samples 3 and 4 didn’t produce any summary for the second prototype, which means the sentences in the transcripts are discarded either because there are no important

sentences in the transcript or they may have been discarded because they were not considered a sentence.

In much the same way the first prototype depends on the users to provide a clean dictation speech for the transcription, much of the success for the summarization also depends on the user. If the user provides wrong information for the three categories, there is no way the system would know the difference. It is simply extracting the important sentences from the transcripts that are already categorized into three questions.

For the second prototype, the summary is more coherent than the transcripts (see Table 10); therefore, the goal of the summarizer is fulfilled. Because the spontaneous conversational speech creates a higher error rate during the transcription, the summarization's role is to reduce member of the errors as much as it can. While there are many ways this summary can be improved, I believe this is probably the best result that can be obtained considering the amount of word errors that existed in the transcript (See Table 10 for comparison). The summary is the result of multiple layers of guesses done by the summarizer due to the errors and vagueness in the transcripts. Unfortunately, without a good transcript, producing a good summary is very difficult. The result is analyzed in Chapter 7.5.

7.4 Analysis of the Summarization Result

As mentioned in Chapter 2, there are many evaluation methods for measuring the quality of a summary. These methods include comparing the summary with human-generated summaries or giving numerical scores for specific types of syntactical or grammatical occurrences. The daily scrum meeting summarizer for the second prototype is not an extraction-based summarizer, as the sentences in the summary are not a verbatim copy of the transcript. Therefore, another method is required to measure the quality of the summary. In this chapter, I will analyze the summary presented in Table 11.

7.4.1 Analysis Methodology

There is no doubt that a human can produce a better summary than the result in Table 11 by re-wording all of the sloppy sentences and condensing the sentences. The performance of the machine summarization is nowhere near the quality that a human can produce. Therefore this evaluation is not about how well the summary can be produced compared to a human, but the amount of improvement that the summarizer was able to make. The evaluation is measured by the following factors.

- **Measurement 1: Speech recognition word error rate:** For the first prototype, this measurement evaluates the word error rate for the dictation speech. For the second prototype, this measurement evaluates the improvement of using two speech recognizers. As presented in Chapter 2, the word error rate for speech recognition is calculated as (S+D+I)/N. S is the number of substituted words; D is the number of deleted words; I is the number of inserted words; N is the total number of words in the speech.
- **Measurement 2: Semantic Deviation**: Count the number of clauses in the summary that deviate from the meanings in the original speech. To obtain the measurement value, I added up the number of mistakenly summarized sentences.
- **Measurement 3: Missing sentences**: Count the number of important clauses that are deemed important by a human summarizer, but didn't get included in the summary.
- **Measurement 4: Incoherent sentences:** Count the number of clauses that are incoherent.

7.4.2 Analysis Result

Here is the analysis on the above four factors for the summary presented in Table 11. The five sample speeches presented throughout the thesis were analyzed and averaged. Tables 12 and 13 show the results of the analysis based on the four factors. The *Generic speech recognition WER* and *Measurement 1: Combined Speech Recognition WER* are measured from Table 10. *Measurements 2, 3,* and *4* are obtained by comparing the

results in Table 11. Table 12 contains the analysis result for the summaries produced from the first prototype.

Analysis Factor	Sample 1	Sample 2	Sample 3	Sample 4	Sample 5
Measurement 1: Speech Recognition WER	1/24	26/124	2/42	2/57	2/42
Measurement 2: Semantic deviation	0/2	4/7	1/4	0/4	0/3
Measurement 3: Missing sentences	0/2	1/6	0/2	0/2	0/4
Measurement 4: Incoherent sentences	0/2	5/7	1/4	0/4	0/3

Table 12: The result of the Daily Scrum Meeting Summarizer (Prototype 1)

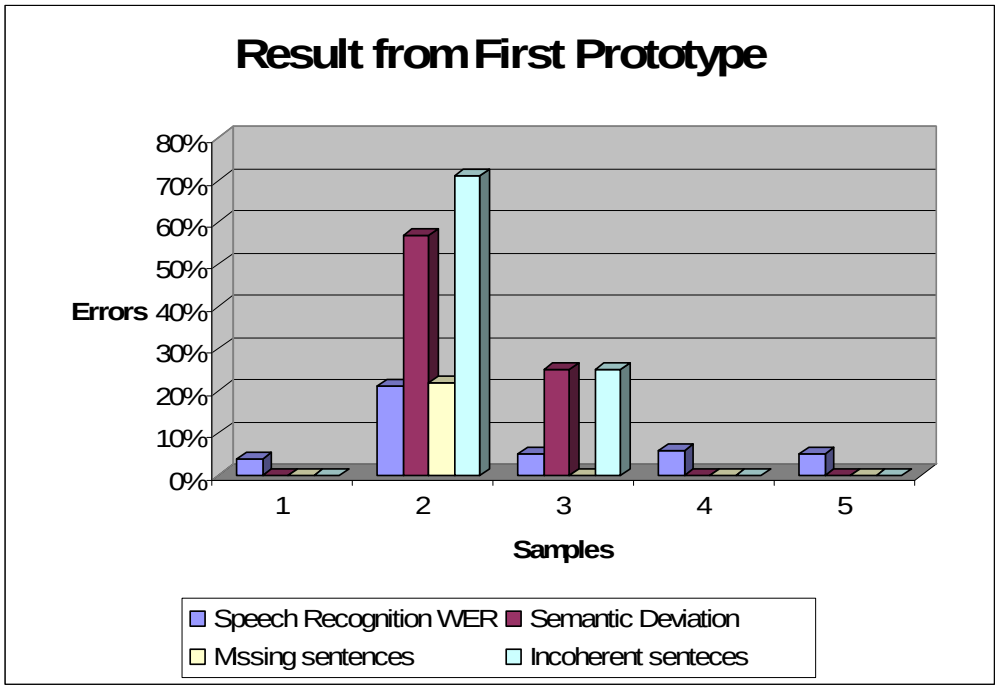


Figure 18: A graph of the result from the Prototype 1

To summarize the results in Table 12, a dictation speech produces a transcript with much higher accuracy rate. Therefore, extracting sentences from an already highly organized transcript produces a much better summary. If the user consciously provides the correct information and a dictation speech, the summary has very few errors. The summaries produced a similar result as the human-generated summary. However, this result was possible because of the highly controlled nature of the way the speeches were obtained. In a real daily scrum meeting scenario, these types of perfect speeches are impossible to obtain.

As shown in Table 12, there are huge variations in the result anywhere from 0% to 100% error. Especially with Sample 2, it has a higher error rate. Sample 2 is a longer speech with a lot of disfluencies, which means a sentence is not properly finished and the speaker would often jump from one topic to the next without finishing his/her sentence. As I have shown in Chapter 4, disfluency and variations in the speech speed can cause speech recognition degradation and Sample 2 just has more of these qualities than the other speeches.

Table 13 contains the analysis results for the summaries produced from the second prototype. The generic speech recognition

Analysis Factor	Sample 1	Sample 2	Sample 3	Sample 4	Sample 5
Generic speech recognition only WER	27/24	159/124	53/42	68/57	52/42
Measurement 1: Combined Speech Recognition WER	16/24	115/124	42/42	57/57	41/42
Measurement 2: Semantic Deviation	0/1	3/7	0/0	0/0	0/0
Measurement 3: Missing sentences	1/2	6/6	2/2	2/2	0/4
Measurement 4: Incoherent sentences	0/1	0/7	0/0	0/0	0/0

Table 13: The result of the Daily Scrum Meeting Summarizer (Prototype 2)

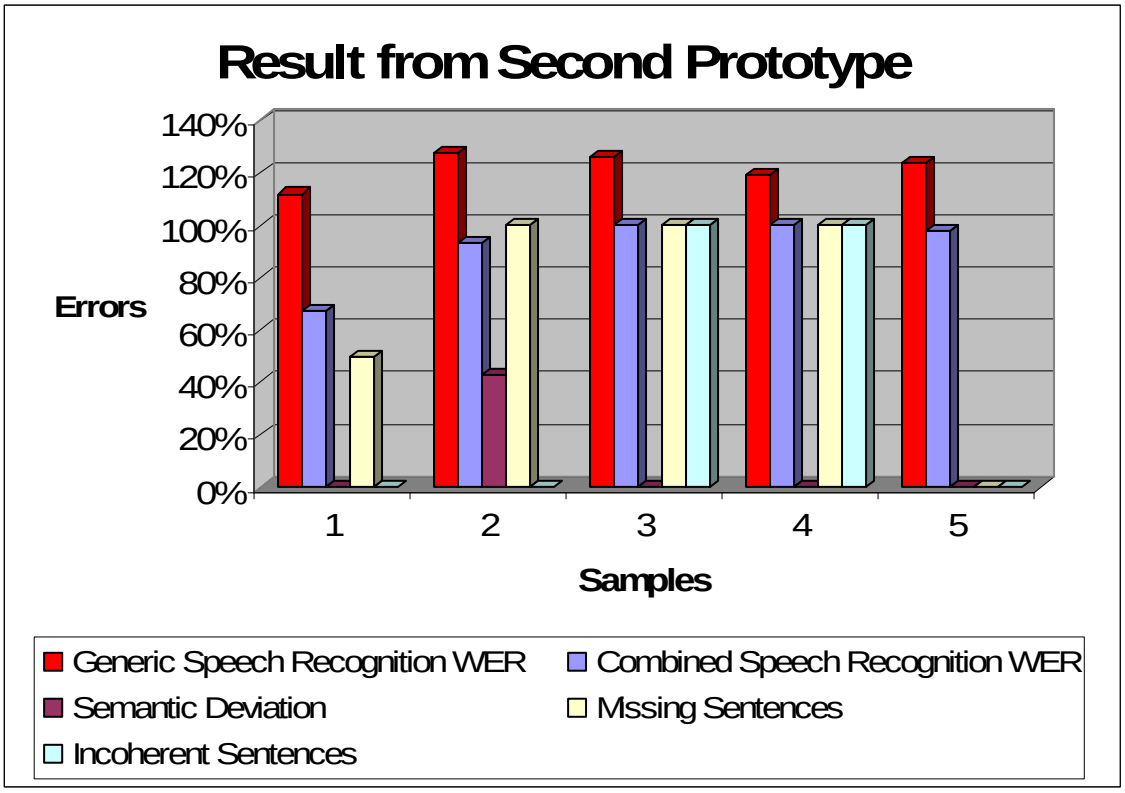


Figure 19: A graph of the result from the Prototype 2

To summarize the result in Table 13, combining the phrase-based and the generic speech recognizer can improve accuracy by an average of 30%. The summarizer can additionally improve about 30 to 50%. However, if the speech recognition engine can't recognize anything at all, then the entire system doesn't work. For example, Sample 3 and 4 produced empty transcripts and thus empty summary is produced.

The word error rate for the transcript is approximately 90%. Producing a summary based on 10% of the correctly recognized transcript text doesn't yield a significantly valid summary. Approximately half of the summary is either deviating from the original speech or incoherent.

Based on these results, I conclude that modifying a generic speech recognition engine can slightly improve the speech recognition, but the improvement is not significant enough to be useful. Overall, more research is needed on the fundamental theories of speech recognition, especially in the area of spontaneous conversational speech.

7.5 Summarization on Human Generated Transcript

The results in Table 13 show that the main cause of the problems is due to the poor speech recognition of spontaneous conversational dialogues. To measure the effectiveness of the summarizer for the second prototype alone, a human transcript is used for the summarizer. The human transcripts of the same five samples are put through the summarizer manually to produce the results in Table 14.

Sample	Summary
Sample 1	-I wrote up that script. -I shown you this morning.
Sample 2	-Ok That's the first sound. -Did I talk, Did I do this experiment with, yeah, three dogs, yeah, experiment with that sound didn't come out right. -Um I want to get the dog to to look at each person in turn. -Just move the head.

	-I could have everyone sitting like this. -um I could do that or I could have. -The dog could move to predefined points. -I could have everyone sitting at this kind of thing and have the to um ok cause. -It's random.
Sample 3	- Have the edit displayed. - I am twenty to half an hour away from having the part inside the plugin.
Sample 4	- I'm trying to get all that stuff done. - so I'm get all that first. - I am done (with the current task).
Sample 5	- I tried out few open source programs. - I tried face recognition. - None of them are working.

Table 14: The summary produced by the second prototype using the human generated transcript

The results in Table 14 are compared with the human-generated summary presented in Table 11. The results are presented in Table 15.

Analysis Factor	Sample 1	Sample 2	Sample 3	Sample 4	Sample 5
Measurement 2: Semantic deviation	0/2	2/9	0/2	1/3	0/3
Measurement 3: Missing sentences	1/2	2/9	1/2	2/2	2/4
Measurement 4: Incoherent sentences	0/2	4/5	0/2	1/3	0/3

Table 15: The result of the summarizer (Prototype 2) using the human generated transcript

The summarizer works much better with human-transcribed text in the most part. Because the sentences are generally cleaner in the human generated transcript, there are fewer incoherent sentences or wrong sentences. In general, the summarizer can catch

half of the important sentences (54% error rate) and contains minimal incoherent sentences (16% error rate).

7.6 Interpretation

There are several limitations and improvements for the daily scrum meeting summarizer.

1. **Combined Speech Recognition can improve the transcription for spontaneous conversational speech:** The combination of *phrase-based speech recognition* and generic speech recognition can produce improved result compared to just the generic speech recognition. However, it only works for a very small recognizable word and topic domain and produces about an average of 30% improvement.
2. **The generic speech recognition engine alone does not work for spontaneous conversational dialogue**: A spontaneous conversational dialogue is fundamentally different from dictation speech. Even with my modifications, the improvement is minimal. The speech recognizer for spontaneous conversational dialogue should be able to handle sloppy sentences, fast speech, abrupt pauses and the Lombard effect.
3. **The phrase-based speech recognition works only within a small domain:** The success of the phrase-based speech recognition is based on the small number of repeatable phrases. If there are too many phrases, the accuracy decreases. If people speak phrases that are not in the grammar, either the *phrase-based ASR* will ignore the speech or match the utterance with a wrong phrase.
4. **A *sentence-dictionary* can improve the coherence but its capability is limited by the number of its entries.** A greater number of entries in the *sentence-dictionary* can produce more coherent and relevant sentences in the summary. If there aren't enough sentences to choose from, the summarizer will choose the closest but a wrong sentence or will copy the incoherent sentence directly from the transcribed text.
5. **Speaker Independence**: Currently, the system cannot be used by people who didn't train a speech profile. It is impractical to force users to train the speech

recognition engine. It is also difficult to specify who the speaker is when the dialogue is rapidly changing between two or more people. The speech recognition engine should be able to transcribe the text using a generic speech profile.

6. **If there are too many grammatical errors, the system is ineffective in fixing the errors.** The transcripts from the spontaneous conversations contain too many incoherent sentences so that it is almost impossible to fix the grammar mistakes in the transcript. When the error rate in the transcript is too high, the summarizer has to extrapolate the meaning and replace the sentence with the closest sentence.
7. **The overall technology is not ready for daily scrum meetings:** While this research has shown that it is possible to improve the transcription and summarization, the error rate is still too high for a serious usage in the real daily scrum meetings.

7.7 Impact on Daily Scrum Meetings

The developers are uncomfortable with the idea of their conversations being recorded and having intrusive recording devices can actually disturb the meetings. In order to make the developers more comfortable with being recorded, the recording process must be unobtrusive. While the first prototype was able to achieve better accuracy, this kind of dictation-style interview with the computer is simply impractical in real life. The users simply are not comfortable with the idea of being interviewed by a computer and the computer is not smart enough to make the interview process more comfortable for the users. However, the second prototype is also impractical due to the high error rate. While giving more freedom to the users with very little intrusion can make the developers feel more comfortable, the resulting transcript contains too many errors to be of much use. In conclusion, further research is required to balance out the interview and the user freedom. In addition, the system needs to become more intelligent and actually try to understand what the user has said so that it can anticipate the type of interaction.

7.8 Summary

In this chapter, I have presented the result of the summarizer performance in finding the important sentences and correcting the incoherency in the transcripts. The summarizer is compared by using both the ASR transcripts and human-generated transcripts. Most of the problems are caused by poor speech recognition. To summarize this chapter, the summarizer has shown some improvement but the error rate is still too high to be used as a tool for a real life scenarios. In the next chapter, I will summarize the thesis by presenting the conclusion.

8 Conclusion

I conclude the thesis by summarizing the research contributions. I revisit the motivation for the research, the research problems and how I solved the problem. I conclude the thesis with possible future work.

8.1 Research Problems

The underlying motivation for starting the research was to help improve the verbal documentation process during the daily scrum meetings by creating an automatic transcriber and summarizer. The goal of the thesis was to experiment with the modification of existing speech recognition engines to transcribe and summarize spontaneous conversational speeches. In the following section, I have re-stated the five research problems first mentioned in Chapter 1 and I describe the solutions that I have proposed.

1. **Identify the current capabilities of speech recognition and summarization software for spontaneous conversational speech.** At current capability levels, speech recognition is quite successful with clean dictation speech. However, the recognition accuracy decreases when the speech starts to sound more like conversational speech and when there is an increase in background noise.
2. **Modify the speech recognition engines for spontaneous conversational dialogue in daily scrum meetings.** The domain for speech recognition can be controlled using a grammar. Combining two types of speech recognition – the phrase-based and the generic ASR - can improve speech recognition accuracy by about average of 30%. Restricting the domain for speech recognition can increase the success of correct speech recognition.
3. **Improve the coherence of the transcribed text.** Wrong recognitions can lead to incoherent sentences. As a worst scenario, even one wrong word in a critical location of a sentence can entirely change the meaning of the sentence. There are two ways to improve coherence: the *phrase-based speech recognition* and *sentence dictionary*. The phrase-based speech recognition approach pre-

processes the speech to match the nearest pre-defined phrases. The *sentence dictionary* approach post-processes the transcribed text to match the pre-defined sentences. The procedure can re-word and condense conversational styled sentences.

4. **Produce a summary of the meeting.** Because the transcripts from the spontaneous conversations have a high word error rate, post-processing is required to make more sense of the phrases. Using the *sentence dictionary* to replace incoherent sentences and ranking the sentences by the occurrence of important keywords can improve the coherency of the summary.
5. **Evaluate the effectiveness of the summary.** The transcription and summarization should be evaluated for its effectiveness in producing a coherent and condensed summary. The summary is evaluated on four factors: combined speech recognition word error rate, missing sentences, wrong sentences and incoherent sentences.

8.2 Thesis Contribution

This thesis makes the following contributions to the problems stated in the previous section.

1. **Combining two different types of speech recognizer engines is better.** As the saying goes, two heads are better than one. The *phrase-based speech recognizer* only listens for a set of repeatable phrases. The rest of the speech is handled by the generic speech recognition engine. While the *phrase-based speech recognition* can mistakenly recognize wrong phrases, it can handle spontaneous conversational speech better for the given set of phrases. The text from the phrase-based speech recognition is more coherent.
2. **Post processing the transcribed text using a *sentence dictionary* can improve the readability.** Because conversational dialogues are sloppy, the transcribed text should be re-worded to eliminate dialogue-styled phrases. The *sentence dictionary* can replace the transcribed text with pre-formatted grammatically correct sentences. The post processing can improve the coherence of the

transcribed text by 30 to 50%. While it is possible that the utterance can be replaced with the wrong sentence in the *sentence dictionary*, there are greater benefits. However, because most of the sentences in the *sentence dictionary* are vague, the resulting sentences in the summary can also be vague.

3. **Ranking the sentences by importance can produce a more coherent summary.** Each speech is given up to a limit of 100 words and the sentences with the highest importance scores are included until the word limit is reached. Because the sentences that are replaced by the *sentence-dictionary* are given the highest score, more coherent sentences are likely to be included in the summary.
4. **The summary should be evaluated based on the number of sentences that deviate from the semantics of the summary, missing sentences and incoherent sentences.** For spontaneous conversational dialogues, the word error rate is not as important as the overall message that has been conveyed in the summary. Counting the number of wrong, missing or incoherent sentences are better indicators on the accuracy of the summary.

8.3 Future Work

The transcription and summarization technology is still too unsophisticated to produce a useful summary for daily scrum meetings. There is still a lot of research required for such a project to succeed. Here are the kinds of research that need to be improved before the *ScrumBot* project can become a reality.

1. **Automatic Speech Recognition for Spontaneous Conversational Speech**: Conversational speech is different from dictation speech. More research is required to reduce the word error rate for spontaneous conversational dialogue speech recognition. Without a better speech recognition engine, it is difficult to produce a good summary.
2. **Interaction with humans**: Some of the speeches in the meetings are so vague that a better interaction method should be examined so that the system can ask the user for clarifications. One method is to restrict the possible ways a person can provide the answers, rather than work with spontaneous conversational

speech. As shown in prototype one, a better speech recognition result can be obtained even with no special system design if the human just provides an audio format that the system can work with. The interaction method would teach the user how to speak with the system until the system obtains the answer in the format that it can work with. However, it also means the system needs to have a large knowledge base about what to ask and how to categorize the user's answers. Interrupting humans in the middle of their conversations requires extensive research into mixed initiative interaction. For example, when is the appropriate time for interruption during the meeting?

3. **Speaker identification**: The machine should be able to match the current speaker and any other participants in the meeting automatically. It is very important for matching the speaker with the correct speech profile.
4. **Computer Vision**: Face-to-face meetings are more than just speech. The system should be able to identify the speakers using visual information such as face recognition. It should be able to understand some body language such as a nod or shake and be able to follow where the person is pointing.
5. **Digital Story cards**: It is difficult for the system to understand human writing. The story cards and any other documents must be stored in a digital format so that the information is also equally easily accessible by the system.
6. **Multiple speech recognition**: Rather than having two types of speech recognition, these speech recognition engines can behave like a group of agents and they can cooperate to improve their performance. Each of these speech recognizers is responsible for more specialized contents or for a specific story card.

8.4 Conclusion

This thesis presents an experiment performed on transcribing and summarizing daily scrum meetings by modifying a generic speech recognition engine and summarizing the transcript based on the key phrases. The experiments have shown that it is possible to

modify the existing technologies to improve the speech recognition by an average of 30% and the summarization can improve the coherency by 30 to 50%.

The vision of creating a meeting summarizer using a robot still requires a lot more research, especially in the area of automatic speech recognition of spontaneous conversational speech. A lot more technical improvements are needed for the speech recognition, integrating artificial intelligence, human-robot interaction and computer vision. With the current level of technology, it is not feasible to create a summarizer that can improve the overall software engineering processes. I predict we are still at least ten years away from having this vision to become a reality. However, as this thesis has shown, small improvements are possible even with the current state of the technology and I believe that small incremental steps in the next decade can make the overall goal possible.

9 References

[AP+06] Ablett, R., Park, S., Sharlin, E., Denzinger, J., Maurer, F. (2006) A Robotic Colleague for Facilitating Collaborative Software Development, Proceedings on Computer Supported Cooperative Work (CSCW 2006), Interactive Poster, Nov 2006, Banff, Canada

[Ba58] Baxendale, B. (1958) Machine-Made Index for Technical Literature – An Experiment, *IBM Journal*, October, 354-361

[Ba72] Baum, L. (1972) An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov process, *Inequalities*, Vol 3, pp 1-8, 1972

[Ba75] Baker, J. (1975) The dragon system – An overview, *IEEE Acoustic, Speech, and Signal Processing*, Vol. ASSP-23, No.1, pp.24-29

[Ba76] Bakis, R. (1976) Continuous speech word recognition via centi-second acoustic states, *Proc ASA Meeting*, Washington, DC

[Be04] Beck, K. (2004) *Extreme Programming Explained: Embrace Change, 2nd Edition*, Addison Wesley

[Bo44] Bodmer, F. (1944) *The Loom of Language*, W.W. Norton & Company, New York

[BBH92] Bateman, D., Bye, D., Hunt, M. (1992) Spectral contrast normalization and other techniques for speech recognition in noise, *Proc. IEEE Internat. Conf. Acoust. Speech Signal Process*, San Francisco, USA, March 1992, Vol I, pp 241-244

[BE67] Baum, L., Egon, A. (1967) An inequality with applications to statistical estimation for probabilistic functions of a Markov process and to a model for ecology, *Bull. Amer.Meteorol.Soc.*, Vol. 73, pp. 360-363

[BJ75] Bahl, L., Jelinek, F. (1975) Decoding for channels with insertions, deletions, and substations with applications to speech recognition, *IEEE Trans. Informat. Theory*, Vol. IT-21, pp. 404-411

[BJM83] Bahl, L., Jelinek, F., Mercer, R. (1983) A maximum likelihood approach to continuous speech recognition, *IEEE Trans. Pattern Anal. Machine Intell.*, Vol. PAMI-5, pp. 179-190

[BMY02] Burger, S., MacLaren, V., Yu, H. (2002) The ISL Meeting Corpus: The Impact of Meeting Type on Speech Style, ICSLP 2002, Denver, Colorado, USA, pp. 1-4

[BP66] Baum, L., Petrie, T. (1966) Statistical inference for probabilistic functions of finite state Markov chains, *Ann. Math Stat*., Vol. 37, pp. 1554-1563

[BP+70] Baum, L., Petrie, T., Soules, G., Weiss, N. (1970) A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains, *Ann. Math. Stat.*, Vol. 41, No 1, pp. 164-171

[BS68] Baum, L., Sell, G. (1968) Growth functions for transformation on manifolds, *Pac. J. Math*, Vol 27, No 2, pp. 211-227

[Ch75] Cherry, C., (1975) *On Human Communication*, MIT Press, Cambridge, Mass.

[Ch88] Chen, Y. (1988) Cepstral domain talker stress compensation for robust speech recognition, *IEEE Trans. Acoust. Speech signal Process*, Vol. ASSP-36, pp. 433-439

[Co04] Cockburn, A. (2004) *Crystal Clear: A Human Powered Methodology for Small Teams (The Agile Software Development Series),* Addison-Wesley Professional

[CG98] Carbonell, J., Goldstein, J. (1998) The use of MMR, diversity-based reranking for reordering documents and producing summaries, *SIGIR 1998,* pp. 335-336

[CH01] Cockburn, A., Highsmith, J. (2001) Agile software development: the people factor, Vol. 34, Iss. 11, pp. 131-133

[Dra] Dragon Naturally Speaking 9.0 http://www.nuance.com/naturallyspeaking/

[Du91] Dumais, S. (1991) Improving the retrieval of information from external sources, *Behavior Research Methods, Instruments & Computers*, Vol. 23, Iss. 2, pp.229-236

[DB+93] Das, S., Bakis, R., Nadas, A., Nahamoo, D., Picheny, M. (1993) Influence of background noise and microphone on the performance of the IBM TANGORA speech recognition system", *Proc. IEEE Internet Conf Acoust. Speech Signal Processes 1993*, Vol II, pp 71-74

[DDM00] Donaway, R., Drummey, K., Mather, L. (2000) A Comparison of Ranking Produced by Summarization Evaluation Measures, *Proceedings of the Workshop on Automatic Summarization*, ANLP 2000, Seattle, Washington, pp. 69-78

[DM80] Davis, S., Mermelstein, P. (1980) Comparison of parametric representation for monosyllabic word recognition in continuously spoken sentences, *IEEE Trans. Acoust. Speech Signal Process*, Vol. ASSP-28, No. 4, pp. 357-366

[Ed68] Edmundson, H (1968) New Methods in Automatic Extraction, *Journal of the ACM*, 16(2), 264-285

[Fe91] Ferber, R. (1991) Slip of the tongue or slip of the ear? On the perception and transcription of naturalistic slips of the tongue, *Journal of Psycholinguistic Research*, Vol. 20, pp. 105-122

[Fo] FOLDOC – Computing Dictionary www.foldoc.org

[Fo05] Fox, B. (2005) Investigation and Testing of Text-to-Voice Software for an Automated Dictation Device, CPSC Undergrad Project Report, University of Calgary, Dec. 2005

[Go95] Gong, Y. (1995) Speech recognition in noisy environment: A survey, *Speech Communication*, Vol 16, Iss 3, pp. 261-291

[GK+99] Goldstein, J., Kantrowitz, M., Mittal, V., Carbonell, J. (1999) Summarizing text documents: Sentence selection and evaluation metrics. *Proceedings of the ACM SIGIR*, pp. 121-128

[GL01] Gong, Y., Liu, X. (2001) Generic text summarization using relevance measure and latent semantic analysis, *SIGIR 2001*, pp. 19-25

[He99] Hearst, M. (1999) Mixed-initiative interaction, *IEEE Intelligent Systems*, Sep-Oct 1999, pp. 14-23

[HB90] Hansen, J., Bria, O. (1990) Lombard effect compensation for robust automatic speech recognition in noise, *Internat. Conf. Speech and Language Process*, Nov 1990, pp. 1125-1128

[HC89] Hansen, J., Clements, M. (1989) Iterative speech enhancement with spectral constraints, *Proc. IEEE Internat. Conf. Acoust. Speech Signal Process,* Dallas, April 1987, pp. 189-192

[HF00] Hori, C., Furui, S. (2000) Automatic speech summarization based on word significance and linguistic likelihood, *International Conference on Acoustic, Speech, Signal Processing*, ICASSP 2000, Istanbul, pp. 1579-1582

[HF03] Hori, C., Furui, S. (2003) A new approach to automatic speech summarization, *IEEE Transactions on Multimedia,* Vol. 5, Iss. 3, pp. 368-378

[HF+03] Hori, C., Furui, R., Malkin, R., Yu, H., Waibel, A. (2003) A statistical approach for automatic speech summarization, *Journal on Applied Signal Processing*, Vol. 2, pp.128-139

[HHM03] Hori, T., Hori, C., Minami, Y. (2003) Speech summarization using weighted finite-state transducers, *Proceedings of 8th European Conference on Speech Communication and Technology*, Geneva, Switzerland, September 2003

[HH93] Hush, D., Horne, B. (1993) Progress in supervised neural networks, *IEEE Signal Processing Mag.*, pp. 8-39

[HL98] Hovy, E., Lin, C. (1998) Automated Text Summarization and the Summarist System, Proceedings of a workshop held at Baltimore, Maryland, *Association for Computational Linguisitcs,* pp.197-214

[GS94] Gish, H., Schmidt, M. (1994) Text-independent speaker identification, *Signal Processing Magazine*, Vol. 11, Iss. 4, pp. 18-32

[GSR91] Gish, H., Siu, M., Rohlicek, R. (1991) Segregation of speakers for speech recognition and speaker identification, *Acoustics, Speech, and Signal Processing*, 1991, ICASSP 1991, Toronto, Canada, pp. 873-876

[Je69] Jelinek, F. (1969) A fast sequential decoding algorithm using a stack, *IBM J. Res. Develop,* Vol 13, pp. 675-685

[Je76] Jelinek, F. (1976) Continuous speech recognition by statistical methods, *Proc IEEE*, Vol. 64, pp. 532-536

[Ju91] Juang, B. (1991) Speech recognition in adverse environment, *Computer Speech and Language*, Vol. 5, pp 275-194

[JA90] Junqua, J., Anglade, Y. (1990) Acoustic and perceptual studies of Lombard speech: Application to isolated words automatic speech recognition, *Proc. IEEE Internat. Conf. Acoustic Speech Signal Process*, Albuquerque, NM, 1990, pp. 841-844

[JBM75] Jelinek, F., Bahl, L., Mercer, L. (1975) Design of a linguistic statistical decoder for the recognition of continuous speech, *IEEE Trans. Informat. Theory*, Vol. IT-21, pp. 250-256

[JB+98] Jing, H., Barzilay, R., McKeown, K., Elhadad, M. (1998) Summarization evaluation methods experiments and analysis, *AAAI Intelligent Text Summarization Workshop*, Stanford, CA, Mar. 1998, pp. 60-68

[Ji05] Ji, M. (2005) Text summarization tool evaluation: A feasibility study for generation meeting summaries, CPSC undergrad project report, University of Calgary, Dec. 2005

[JS05] Jaimes, A., Sebe, N. (2005) Multimodal Human Computer Interaction: A Survey, *IEEE International Workshop on Human Computer Interaction*, Beijing, China, Oct. 2005

[Ko03] Kotelly, B. (2003) *The art and business of Speech Recognition: Creating the noble voice*, Addison-Wesley

[KFH03] Kikuchi, T., Furui, S., Hori, C. (2003) Two-stage automatic speech summarization by sentence extraction and compaction, *Workshop on Spontaneous Speech Processing and Recognition*

[KGG89] Koo, B., Gibson, J., Gray, S. (1989) Filtering of colored noise for speech enhancement and coding, *Proc. IEEE Internat. Conf. Acoust. Speech Signal Process*, Glasgow, Scotland, May 1989, pp 349-352

[KR05] Koumpis, K., Renals, S. (2005) Automatic summarization of voicemail messages using lexical and prosodic features, *ACM Transactions on Speech and Language Processing*, Vol. 2, pp. 1-24

[Le89] Lecomte, I. (1989) Car noise processing for speech input, *Proc IEEE Internat. Conf. Acoust. Speech Signal Process*, Glasgow, Scotland, May 1989, pp. 512-515

[Lu59] Luhn, H. (1959) The Automatic Creation of Literature Abstracts, *IBM Journal of Research and Development*, 159-165

[LB92] Lockwood, P., Boudy, J. (1992) Experiments with a Non-linear Spectral Subtractor (NSS), Hidden Markov Models and the projection, for robust speech recognition in cars, *Speech Communication*, Vol 11, No. 2-3, pp. 215-228

[LFL98] Landauer, T., Foltz, P., Laham, D. (1998) Introduction to Latent Semantic Analysis, *Discourse Processes 1998*, Vol. 25, pp. 259-284

[LMP87] Lippmann, R., Martin, E., Paul, D. (1987) Multi-style training for robust isolated-word speech recognition, *Proc. IEEE Internat. Conf. Acoust. Speech Signal Process*, Dallas, TX, April 1987, pp. 705-708

[LHS05] Litman, D., Hirschberg, J., Swerts, M. (2005) Characterizing and Predicting Corrections in Spoken Dialogue Systems, *Computational Linguistics*, Vol. 32, Iss. 3, pp. 417 – 438

[LO94] Lindsay, J., O'Connell, D. (1994) How do transcribers deal with audio recordings of spoken discourse?, *Behavioral Science*, Vol. 24, Iss. 2, March 1995

[Ma95] Maybury, M. (1995) Generating summaries from event data, *Information Processing Management*, Vol. 31, Iss. 5, pp. 735-751

[Ma98] Marcu, D. (1998) Improving Summarization through Rhetorical Parsing Tuning. *Proceedings of the COLING-ACL Workshiop on Very Large Corpora*, Montreal, Canada

[Ma01] Mani, I. (2001) Recent Developments in Text Summarization, *Proceedings from* CIKM, Nov 2001, Atlanta, Georgia, USA

[Mo] Moby Part-of-Speech, http://aspell.sourceforge.net/wl/

[MC91] Mokbel, C., Chollet, G. (1991) Speech recognition in adverse environments: speech enhancement and spectral transformations, *Proc. IEEE Internat. Conf. Acoust. Speech Signal Process*, 1991, pp. 925-928

[MK+02] Mani, I., Klein, G., House, D., Hirschman, L., Firmin, T., Sundheim, B. (2002) SUMMAC: a text summarization evaluation, *Natural Language Engineering*, Cambridge University Press, Vol 8, Iss. 1, March 2002

[MM03] Moore, D., McCowan, I. (2003) Microphone array speech recognition: experiments on overlapping speech in meetings, *Acoustics, Speech and Signal Processing*, ICASSP 2003, Vol.5, pp. 497-500

[MRC05] Murray, G., Renals, S., Carletta, J. (2005) Extractive summarization of meeting recordings, *Proceedings of the 9th European Conference on Speech Communication and Technology*, Lisbon, Portugal

[MR+06] Murray, G., Renals, S., Carletta, J., Moore, J. (2006) Incorporating Speaker and Discourse Features into Speech Summarization, *Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL*, New York, June 2006, pp. 367-374,

[NS+85] Nocerino, N., Soong, F., Rabiner, L., Klatt, D. (1985) Comparative study of several distortion measures for speech recognition, *Proc. IEEE International Conf. Acoustic Speech Signal Process*, 1985, pp. 25-28

[Os89] O'Shaughnessy, D. (1989) Enhancing speech degraded by additive noise or interfering speakers, *IEEE Communication Magazine*, Feb 1989, pp. 46-52

[OB+96] Ostendorf, M., Byrne, B., Bacchiani, M., Finke, M., Gunawardana, A., Ross, K., Roweis, S., Shriberg, E., Talkin, D., Waibel, A., Wheatley, b., Zeppenfeld, T. (1996) Systematic Variations in Pronunciation via a Language-Dependent Hidden Speaking Mode, ICSLP 1996, Philadelphia, USA

[OR92] Ogunnaike B., Ray, W.H (1992) *Process Dynamics, Modeling and Control*, Oxford University Press, p 364 in Schwaber,K. (2004) *Agile Project Management with Scrum*, Microsoft Press

[Pe05] Pentland, A. (2005) Socially aware computation and communication, *IEEE Computer*, Vol.38, No. 3, pp. 33-40

[Pi85] Pisoni, D. (1985) Some acoustic-phonetic correlates of speech produced in noise, *Proc IEEE Internat. Conf. Acoust. Speech Signal Process,* Tampa, March 1985, pp .1581-1584

[PD+06] Park, S., Denzinger, J., Maurer, F., Sharlin, E. (2006) An Interactive Speech Interface for Summarizing Agile Project Planning Meetings, Proceedings on Computer-Human Interaction (CHI 2006) Work in Progress Report, pp. 1205-1210 , April 2006, Montréal, Canada

[PP03] Poppendieck, M., Poppendieck, T. (2003) *Lean Software Development: An Agile Toolkit for Software Development Managers*, Addison Wesley

[Ra89] Rabiner, L. (1989) A tutorial on hidden Markov models and selected applications in speech recognition, *Proc. IEEE*, Vol. 77, No. 2, pp. 257-285

[RR95] Reynolds, D., Rose, R. (1995) Robust text-independent speaker identification using Gaussian mixture speaker models, *Speech and Audio Processing*, Vol. 3, Iss. 1, pp. 72-83

[Sa] Microsoft Speech SDK 5.1 www.microsoft.com/speech/download/sdk51/

[Sa88] Salton, G. (1988) *Automatic Text Processing*, Reading, MA:Addison-Wesley

[Sc04] Schwaber, K. (2004) *Agile Project Management with Scrum*, Microsoft Press

[SG96] Sparch-Jones, K., Galliers, J. (1996) *Evaluating Natural Language Processing Systems: An Analysis and Review: Lecture Notes in Artificial Intelligence,* Iss. 1083, Springer-Verlag

[SJ04] Steinberger, J., Jezek, K. (2004) Using latent semantic analysis in text summarization and summary evaluation, *ISIM 2004*, pp.93-100

[Ta91] Taylor, F. (1911) *The Principles of Scientific Management*, http://melbecon.unimelb.edu.au/het/taylor/sciman.htm

[VR+99] Valenza, R., Robinson, T., Hickey, M., Tucker, R. (1999) Summarization of spoken audio through information extraction, *ESCA Workshop on Accessing Information in Spoken Audio*, pp.111-116

[VM93] Vaseghi, S., Milner, B. (1993) Noise-adaptive hidden Markov models based on Wiener filters, *Proc European Conf Speech Technology,* Berlin, 1993, Vol. II, pp. 1023-1026

[Wh90] White, G. (1990) Natural Language Understanding and Speech Recognition, *Communications of the ACM*, Vol.33, No.8, August 1990

[WB+98] Waibel, A., Bett, M., Stiefelhagen, R., Meeting Browser: Tracking and summarizing meetings, *DARPA Broadcast News*

[WC00] Wan, V., Campbell, W. (2000) Support vector machines for speaker verification and identification, *Neural Networks for Signal Processing*, Sydney, Australia, pp.775-784

[WM99] Witbrock, M., Mittal, V. (1999) Ultra-summarization: A statistical approach to generating highly condensed non-extractive summaries, in *SIGIR*, pp. 315-316

[WY+01] Waibel, A., Yu, H., Westphal, M., Soltau, H., Schultz, T., Schaaf, T., Pan, Y., Metze, F., Bett, M. (2001) Advances in meeting recognition, *Human Language Technology Conference* 2001, San Diego, pp. 1-3

[YH+89] Young, S., Hauptmann, A., Ward, W., Werner, P. (1989) High level knowledge source in usable speech recognition systems, *Communications of the ACM*, Vol. 32, Iss. 2, pp. 183-193

[Ze02] Zechner, K. (2002) Automatic summarization of open-domain multiparty dialogues in diverse genres, *Computational Linguistics*, Vol. 28, No.4, pp. 447- 485

[ZW00] Zechner, K., Waibel, A. (2000) Minimizing word error rate in textual summaries of spoken language, *NAACL 2000*

10 Appendix A: Ethics Approval

UNIVERSITY OF CALGARY

MEMO

CONJOINT FACULTIES RESEARCH ETHICS BOARD
c/o Research Services
Main Floor, Energy Resources Research Building
3512 - 33 Street N.W., Calgary, Alberta T2L 1Y7
Telephone: (403) 220-3782
Fax: (403) 289 0693
Email: bonnie.scherrer@ucalgary.ca
Wednesday, February 01, 2006

To: **Shelly S. Park**
Computer Science

From: Dr. Janice P. Dickin, Chair
Conjoint Faculties Research Ethics Board (CFREB)

Re: **Certification of Institutional Ethics Review:** ALAN: A Robotic Companion for Agile Teams

The above named research protocol has been granted ethical approval by the Conjoint Faculties Research Ethics Board for the University of Calgary.

Enclosed are the original, and one copy, of a signed **Certification of Institutional Ethics Review**. Please make note of the conditions stated on the Certification. A copy has been sent to your supervisor as well as to the Chair of your Department/Faculty Research Ethics Committee. In the event the research is funded, you should notify the sponsor of the research and provide them with a copy for their records. The Conjoint Faculties Research Ethics Board will retain a copy of the clearance on your file.

Please note, an annual/progress/final report must be filed with the CFREB twelve months from the date on your ethics clearance. A form for this purpose has been created, and may be found on the "Ethics" website, http://www.ucalgary.ca/UofC/research/html/ethics/reports.html

In closing let me take this opportunity to wish you the best of luck in your research endeavor.

Sincerely,

Bonnie Scherrer
For:
Janice Dickin, Ph.D., LLB., Faculty of Communication and Culture and
Chair, Conjoint Faculties Research Ethics Board

Enclosures(2)
cc: Chair, Department/Faculty Research Ethics Committee
Supervisor: Frank Maurer

UNIVERSITY OF CALGARY

CERTIFICATION OF INSTITUTIONAL ETHICS REVIEW

This is to certify that the Conjoint Faculties Research Ethics Board at the University of Calgary has examined the following research proposal and found the proposed research involving human subjects to be in accordance with University of Calgary Guidelines and the Tri-Council Policy Statement on *"Ethical Conduct in Research Using Human Subjects"*. This form and accompanying letter constitute the Certification of Institutional Ethics Review.

File no:	**4689**
Applicant(s):	**Shelly S. Park** Ruth M. Ablett Frank Maurer Joerg Denzinger Ehud Sharlin
Department:	**Computer Science**
Project Title:	**ALAN: A Robotic Companion for Agile Teams**
Sponsor (if applicable):	

Restrictions:

This Certification is subject to the following conditions:

1. Approval is granted only for the project and purposes described in the application.
2. Any modifications to the authorized protocol must be submitted to the Chair, Conjoint Faculties Research Ethics Board for approval.
3. A progress report must be submitted 12 months from the date of this Certification, and should provide the expected completion date for the project.
4. Written notification must be sent to the Board when the project is complete or terminated.

Janice Dickin

Janice Dickin, Ph.D, LLB,
Chair
Conjoint Faculties Research Ethics Board

1 February 2006

Date:

Distribution: (1) Applicant, (2) Supervisor (if applicable), (3) Chair, Department/Faculty Research Ethics Committee, (4) Sponsor, (5) Conjoint Faculties Research Ethics Board (6) Research Services.

11 Appendix B: Glossary

Abstraction summarization: Summarize by rewording the original text.

Acoustic knowledge: a set of data and instructions on how to process auditory signals or sound.

Acoustic features: refers to the computational features from a sampled speech signal that makes the signal unique.

Agile software engineering: A set of evolutionary software engineering methodologies that emphasize iteration, real time communication and adaptability to changing requirements.

Agile Manifesto: A principle underlying agile software development. It emphasizes four factors: individual and interactions over processes and tools; working software over comprehensive documentation; customer collaboration over contract negotiation; responding to change over following a plan.

Artificial Neural Network: An interconnected group of nodes much like a network of neurons in the human brain that can be used as a mathematical model for computation. The cost function determines which route the system should take in the network.

Automatic Speech Recognition (ASR): A process of converting a speech signal to words.

Computer-Telephony integration: A technology that allows integration of a telephone and a computer. For example, call information display, fast dial, call routing or call center phone control.

Crystal Clear: An agile methodology proposed by Cockburn. It emphasizes being sensitive to people issues and better communication among developers.

Daily Scrum Meeting: A daily progress report meeting in Scrum methodology. The participants talk about what they did, what they will get done and what problems they encountered.

Disfluency: Irregularities and breaks in utterances such as repeated words, repeated syllables, unrecognizable utterances or filler words such as “uh”.

eXtreme Programming (XP): An agile methodology proposed by Kent Beck. It emphasizes “embracing change”, courage, testing and pair programming.

Expert System: Also known as a knowledge-based system. It is a computer system that contains subject-specific knowledge and performs a specific set of tasks.

Extraction summarization: Produces a summary of the original text by extracting important sentences in verbatim.

Hidden Markov Model (HMM): Composed of a network of states with transition probabilities. It is a process that can find the paths and the nodes by using only the observable parameters. Used for pattern recognition such as speech recognition.

Indicative summary: The summary only identifies the subject or the domain of the original text.

Inflection: A variation in the form of a word. In the English language, it is done usually by adding a prefix or a suffix.

Informative summary: The summary provides the contents of the original text

Grammar: a set of rules for a speech recognition engine to follow when transcribing the speech signal.

Lexical Analysis: Analyzing the sentence based on vocabularies

Linear Discriminate Analysis: a broad set of statistical techniques that tries to categorize a set of objects into groups based on the object’s features.

Linguistic knowledge: a set of knowledge on how to use the language such as pronunciations, semantic knowledge, grammatical knowledge and when to use the words.

Local Area Network (LAN): A computer network covering a small area like a home or an office.

Latent Semantic Analysis (LSA): a method for extracting a summary by analyzing the contextual meaning of words using statistical computations applied to a large corpus of text.

Lean development: An agile methodology proposed by Poppendieck that emphasizes eliminating waste, minimizing inventory and doing it right the first time.

Lombard effect: A variation in the overall vocal intensity when a speaker tries to speak over noisy environment.

Mixed-initiative interaction: A flexible interaction strategy where each agent (human or computer) contributes to the collaboration when it is best suited and most appropriate to do so.

Multimodal interaction: An interaction strategy that provides the user with multiple modes of interfacing with the system including audio (speech) or video (gesture, body motion, facial expressions)

.NET languages: .NET is a framework that can run on top of the MS Windows operating system. It provides the developer with pre-coded class libraries and runtime environments so that programmers don't have to worry about different CPUs or versions of the operating system. Some of the .NET languages include C#, VB.NET or J#.

Parameter Estimation: A process which we know the finite number of unknown parameters and we would like to find the best estimate of these parameters.

Probability Density Function: a function that represents a probability distribution of sample points by "smoothing" out the discrete points into a continuous distribution

Product Backlog: The requirements for software development that will not be implemented in the current iteration.

Prosodic feature: A collective term describing variation in pitch, loudness, tempo and rhythm in speech. It is called Intonation in the English language.

Reverberation: reflected sounds in a room. Eg. sound in auditorium

Scrum methodology: An agile methodology introduced by Ken Schwaber. It emphasizes a 30-day sprint and frequent communication through daily scrum meetings.

Signal to Noise Ratio (SNR): a ratio that compares the level of a desired signal with the level of background noise

Software Development Kit (SDK): tools that allow a computer programmer to create applications

Speech feature vector: a vector containing speech features from an audio signal

Speech parameterization: a method for extracting relevant information from speech signals so that different speech sounds can be recognized.

Spontaneous conversational speech: A type of speech that people use when they are having live spontaneous conversations with other people.

Sprint: An iteration consisting of 30 days in the Scrum methodology

Sprint planning meeting: A meeting with developers and stakeholders to determine what functionalities should be implemented during the current Sprint.

Stochastic language model: a model depicting the probability of sequence of words

Stochastic process: Assumes that the process can produce many possible outcomes although there are more probable outcome. There are some random factors such as time that can change the outcome.

Tayloristic approach: a management style based on task-oriented optimization. Eg. assembly line in manufacturing factory.

Transcription: a process of converting spoken words into written language.

Transmission channels: a communication means of transmitting signals.

Voice portal: Also referred to as vortal where a service can be reached by telephone. It often offers information such as weather, stock quotes and sport scores.

Waterfall model: A sequential software development model that steps through phases including requirements, design, implementation and testing like a waterfall.

White noise: a signal that contains all combination of different frequencies in sound

Wiener and Kalman filtering: methods for filtering out noise based on statistical approach

Word Error Rate (WER): computed as follows: (S+D+I)/ N where S is number of substitutions; D is number of deletions; I is the number of insertions; N is the total number of words in the original speech. The higher the value, the poorer the speech recognition it is.

12 Appendix C: Phrases in the Phrase-based ASR

About the keyword extraction
About the keyword extractions
activation exception
Actually I got it to work
Actually we got it to work
Aibo to look at whoever's talking
Aibo to walk
All the test files and fixture files are in the directory
All the tutorials are done
already bound exception
And then run again
and what not
Anybody running into trouble that they can't solve
Anything important
Anything that implements that interface can easily replace this interface
application exception
Apply the methods
arithmetic exception
As a general rule
AWT exception
backing store exception
Bad
bad location exception
Because we commented the line
Before you have to switch
but I haven't analyzed it yet
But I think his voice is very low
But there's some exception
But you define what you need
But you said
By the way
Can we assume that project file is there
Can we discuss in the next meeting
Can we discuss this now
Can we discuss this on Friday
Can we discuss this on Monday
Can we discuss this on Saturday
Can we discuss this on Sunday
Can we discuss this on Thursday
Can we discuss this on Tuesday
Can we discuss this on Wednesdsay
Can we discuss this tomorrow
Can't even recognize my own face
certificate exception
class cast exception
class not found exception
client is creating the error
clone not supported exception
data format exception
destroy failed exception
Did I talk about it
Do I do this experiment with Aibo
Don't know
everyone is working
Everything's due next week
Everything's due on Friday
Everything's due on Monday
Everything's due on Saturday
Everything's due on Sunday
Everything's due on Thursday
Everything's due on Tuesday
Everything's due on Wednesday
Everything's due tomorrow
exception
Experiment with that sound
fine
Fixture doesn't have to be on the server
Follow where the sound is
font format exception
Friday
general security exception
get project names
Getting the correct transcript
Good
Good luck
Got what we expect
having the edit part displayed
He doesn't have a prior understanding of the work
Here is the brief introduction
How are you
How do we do that
I am
I am available for testing
I am done
I am getting the correct transcript

I am half an hour away
I am motivated
I analyzed
I believed
I brought all the video imported
I brought this
I can do that
I can get that done
I can get that done by Friday
I can get that done by Monday
I can get that done by next month
I can get that done by next week
I can get that done by Saturday
I can get that done by Sunday
I can get that done by Thursday
I can get that done by Tuesday
I can get that done by Wednesday
I can get the color
I can guarantee
I can make it through
I can provide the information
I can read it locally
I can show the color to green color or the red color
I could do that
I could have everyone sitting at this kind of thing
I could have everyone sitting like this
I didn't know how
I don't care
I don't get it
I don't have a prior understanding of the work
I don't have the plan
I don't know
I don't know why
I don't really have plans anymore
I don't think we have to separate
I forget everything until something pops up on my computer
I forgot everything
I give
I got
I got the script going
I guess
I guess I didn't know how
I had to know
I have bad feeling about this
I have carefully examined
I have changed the email address
I have different versions
I have encountered
I have finished
I have from
I have located
I have not analyzed yet
I have not done a whole lot
I have not finished yet
I have to decide
I have to know
I have to perform
I have tried different versions
I haven't analyzed yet
I haven't done a whole lot
I haven't finished yet
I hope
I imported some of the video
I imported them
I just like to
I know
I love doing this
I must know
I need to do
I need to perform
I promise
I set it up
I set it up as recommended
I setup
I shown you last night
I shown you last week
I shown you this afternoon
I shown you this afternoon
I shown you this evening
I shown you this morning
I shown you three days ago
I shown you two days ago
I shown you yesterday
I sorted some problems about EJB
I started looking at some of the videos
I think
I think actually follow sound would be better.
I think I'm just going to move onto
I think in less than a month
I think in less than a week

I think in less than a year
I took your words
I tried
I tried CMU Sphinx
I tried different versions
I tried Dragon Naturally Speaking
I tried Microsoft Speech SDK
I tried open CV
I tried out few face recognition
I understand
I want to
I want to get the dog to look at each person in turn
I want to get the other parts done
I want to get the other parts we did on the screen
I want to try out on the different machine
I want to try out on the same machine
I will work on it
I wish
I wish I could
I won't start anything by Friday
I won't start anything by Monday
I won't start anything by Saturday
I won't start anything by Sunday
I won't start anything by Thursday
I won't start anything by Tuesday
I won't start anything by Wednesday
I wouldn't waste my time
I wrote a letter
I wrote up that script
I wrote up that script
I'd like to stay
If asynchronous persistor uses our interface
If connector doesn't make sense we drop it
If not, get rid of them
If we are handling works on server
If we run it in our server
I'll assure you
I'll do that
I'll like to
I'll try to update the output
I'll try to update the output stream
illegal argument exception
illustrates
I'm
I'm a little concerned about
I'm alright
I'm also researching
I'm feeling good
I'm feeling much better now
I'm going to be spending
I'm going to hold a presentation
I'm going to hold a presentation
I'm going to hold a presentation for the people in lab
I'm going to hold a seminar
I'm going to start with a brief introduction
I'm going to try it out
I'm going to try that one out
I'm going to try that one out
I'm not getting any
I'm reading out
I'm ready to go
I'm researching about
I'm still working on it
I'm still working on that
I'm still working on that
I'm still working on the project
I'm trying to do that
I'm trying to do that
I'm trying to get all that stuffs done
I'm trying to work on the project
I'm working
I'm working on
I'm working on the new project
I'm working with
In the scheduler
instantiation exception
interrupted exception
IO exception
Is it the project directory
Is it tomorrow
Is it tomorrow or next week
Is it yesterday
It can automatically find out
It can done
It can still be useful
it compiled
It didn't come out right
It doesn't have to be
It has a higher pitch
It is able to follow

It is available for testing
it is important to
it is important to realize
It is impossible
It is located
It is not automated
It is not possible
It is not required to implement it
It is possible
It isn't required to implement it
It pumps out a new error
It says
It says it can't find a class
It says it can't find a classpath
It was working yesterday
It wasn't long ago
It will be packaged
It's a general rule
It's automated
It's much easier to use our synchronous persistor
It's not there
It's on Friday
It's on Monday
It's on Saturday
It's on Sunday
It's on Thursday
It's on Tuesday
It's on Wednesday
It's only you can only put in certain hours
It's possible
It's random
It's so good
It's terrible
it's too early to say
It's very complicated
I've been busy
I've been busy with course work
I've been looking at the energy sound level
I've been looking at the way
I've been looking for
I've got some recordings
I've got some recordings
JBoss server shouldn't execute column fixture
Just move the head
last owner exception
Let them be familiar with JUnit
Let's change it
let's do it
Let's get down to business
Let's get started
let's go
let's have a look
Let's have a look
let's reintroduce
let's see
Let's see if it works
let's see if it works
Let's see if this works
Like remote connection
Maybe I should get some help
Mime type parse exception
Monday
naming exception
Next step for me would be
No
no such method exception
None of them are working
noninvertible transform exception
not owner exception
Not yet
Now it should actually work
null pointer exception
Okay
On separate machine it doesn't
on the fitness page
On the same machine it works
on the webpage
out of memory error
parse exception
parser configuration exception
parser error
Pretty much everything's on hold
printer exception
privileged action exception
Right now initialization has been torn down
runtime exception
Saturday
SAX exception
Say what you mean
She doesn't have a prior understanding of the work

So but then we have to make sure we copy the files into that directory
So for example
So I got some of the initial structure setup
So I should be able to get done by Friday
so I should be able to get done by Monday
So I should be able to get done by next month
So I should be able to get done by next week
So I should be able to get done by Saturday
So I should be able to get done by Sunday
So I should be able to get done by Thursday
So I should be able to get done by tomorrow
So I should be able to get done by Tuesday
So I should be able to get done by Wednesday
So I should be able to get done in about a month
So next time is on Friday
So next time is on Monday
So next time is on Saturday
So next time is on Sunday
So next time is on Thursday
So next time is on Thursday
So next time is on Tuesday
So next time is on Wednesday
So that's done
so we are meeting tomorrow
So will it always be that directory
So you can get all that done
So you can get all that done by Friday
So you can get all that done by Monday
So you can get all that done by Saturday
So you can get all that done by Sunday
So you can get all that done by Thursday
So you can get all that done by Tuesday
So you can get all that done by Wednesday
Still call them
Sunday
Talking too fast
tested all files under the directory
Thank you
Thanks
That is different from ordinary one
That is the only reason
that the people are working on
That was easy
That was hard
That's a totally different presentation
that's a waste of my time
that's a waste of time
That's about all that's happened
That's about all that's happened
That's fine
That's my goal for next
That's my goal for next
That's not the worst problem
That's right
That's the first sound
That's the fixture
That's the other one
That's the other one
The client is creating the error
The concern that we have with that thing is
the dog move to predefined paths
the dog move to predefined points
The error keep coming out
The noise is kind of blended in
The package is in the directory
Then let's get started
Then we are done
There are many large files
There are many small files
There are two ways to implement
There is a new problem
There's a very easy way out
There's so much noise
They are automated
They are available for testing
They are different
They are motivated
They are same
They don't have a prior understanding of the work
They have to decide
They were different
This computer has been locked

This is a central componenet
this is common knowledge
This is impossible
This is not possible
This is server
This thing is causing too much problem
thread death
Thursday
to get the dog to not fall off the table
To see if either there are any difference
Tuesday
unsupported flavor exception
we analyzed
we are able to follow
We are done for today
We are done then
We are motivated
We are now getting the live context
we ask
We assume there is something in the directory
we can include
we can only include session beans
We can only include session beans
we can specify
We cleaned up
We did that
We don't need any of them
We don't need to throw that exception
We found no evidence
we found that
we got
We have Agile Planner
we have asynchronous persistor
We have him laughing
We have someone laughing
We have synchronous persistor
we have to separate
we haven't changed the package
We haven't changed the package
We haven't changed the package
we hope
we look
we may not get
we need
we need directory
we plan to
we should consider
We should get something else
we test all
we test all things under the directory
We think
we use
we wait
we want to have
We want to have another thing
We want to have the iteration meeting
we will be using
we will conduct
we will discuss
Wednesday
what are the current activities
what are they doing
what are you doing
what are you doing here
what are you going to do with it
what are you saying
what can
what can I do
what can they do
what can they see
what can you do
what changes
what changes are they making
What did you do
What did you do since the last meeting
What did you do since the last time
what do I need
what do they need to do next
what do you think
What does it say
What happened
what happened
What happens
what have you done
what if I can't
what is
what objects are they using
What should begin that project directory
what will they do next
what will you do next
whatever it is
what's going on
what's important is

What's it
What's the concrete next step
what's the plan for the next week?
what's the plan for tomorrow
What's to the next Friday
What's to the next Monday
What's to the next Saturday
What's to the next Sunday
What's to the next Thursday
What's to the next Tuesday
What's to the next Wednesday
When I do the fixture code in the fitness engine
When is it
When is the meeting
When was the last meeting
whenever it is
Where is it
wherever it is
which files
Which one do we need
Which version
Why are there many
will accumulate
will allow for
will be
will be analyzed
will be conducted
will be different
will be executed on
will be included
will be noted
will be packaged
will be recorded
will be used
will have
will help ensure
will list
will not accumulate
will not be analyzed
will not be conducted
will not be different
will not be executed on
will not be included
will not be noted
will not be packaged
will not be recorded
will not be used
will not differ
will not have
will not help ensure
will not list
will provide guidance for
will provide guidance for
will remain
will remain
will work
Working on Alan
Working on Digital tabletop
Working on Fitness
Working on Mase
working on speech recognition
working on summarization
working on testing
working on the analysis
working on the report
working on the survey
Working on thesis
Would it be better for the stability
Yeah
Yes
You are motivated
You can think about that
You can't move around a lot
You finish it
You have the interface
You know
You know I tried refreshing
You shouldn't have multiple tasks
You start a task
you've got the latest up to date information

13 Appendix D: Sentences in the Sentence-Dictionary

The first column is the replacement sentence and the subsequent words are the keywords.

3D rendering performance is a key quality requirement, 3D, rendering, performance, quality, requirement
About the keyword extraction, about, keyword, extraction
Adheres to bluetooth 1.2 standard, bluetooth, 1.2, standard
After the system startup for 10 minutes, system, start, 10, minute,
Agile method promotes iterative development, agile, promote, iterative
Aibo to look at whoever's talking, aibo, look at, talking
All files will play perfectly in WMP, file, play, wmp
All the test files and fixture files are in the directory, test file, fixture file, in, directory,
All the tutorials are done, tutorial, done ,
Allows you to connect to a wide variety of devices without cables, devices, connect, without, cables,
Allows you to easily send data wirelessly, wirelessly, send, data
An icon is in the lower right of the desktop, icon, desktop, lower, right,
Anybody running into trouble that they can't solve?, running into trouble, anybody, they can't solve
Asked for the administrator password, ask, administrator, password
Attempted System restore, attempt, system restore ,
Bluetooth enabled networking products are capable of data transfer among devices, bluetooth, networking, capable, data transfer, devices
Can access the computer remotely, access, computer, remotely
Can we discuss in the next meeting, can we, discuss, next, meeting,
Can we discuss this now, can we, discuss, now
Can we discuss this on Friday, can we, discuss, friday
Can we discuss this on Monday, can we, discuss, monday
Can we discuss this on Saturday, can we, discuss, saturday
Can we discuss this on Sunday, can we, discuss, sunday
Can we discuss this on Thursday, can we, discuss, thursday
Can we discuss this on Tuesday, can we, discuss, tuesday
Can we discuss this on Wednesday, can we, discuss, wednesday
Can we discuss this tomorrow, can we, discuss, tomorrow
Cannot start Windows Firewall service, cannot, start, Windows, Firewall, service
Can't connect to the internet, can't, connect, internet
Can't establish the connection to the internet, can't, connection, internet
Can't even recognize my own face, can't even, recognize, face, my,
Can't figure it out, figure, out, can't
Checked CPU usage, check, cpu usage ,
Client is creating the error, client, is creating, error
Computer boot into safe mode, computer, boot, safe mode
Could have done on every Wednesday, could have, every, Wednesday
Could not find the name of the program, could, not, find, name, program
Data can be intercepted , data, can, intercepted
Decided to compromise by caching the data into the session, compromise, decide, caching, session,

Deliver wireless Personal Area Network connection for your computer, wireless, Personal Area Network ,
Demand better technology, demand, better, technology
Demonstrated motion capture technology, demonstrate, motion, capture
Developed using Java2D and Java3D, Java2D, Java3D ,
Discuss the current strategies for testing, strategy, testing ,
Do an experiment with Aibo, experiment, Aibo ,
Do not have a USB drive, do not have, USB, drive
Do not have the CD, do, not, have, CD,
Downloaded a screensaver, screensaver, download ,
Everyone is working together well, everyone, working ,
Everything's due next week, everything's, due, next week
Everything's due on Friday, everything's, due, friday
Everything's due on Monday, everything's, due, monday
Everything's due on Thursday, everything's, due, thursday
Everything's due on Tuesday, everything's, due, tuesday
Everything's due on Wednesday, everything's, due, wednesday
Everything's due tomorrow, everything's, due, tomorrow
Fails to find Intel matrix storage manager, fail, setup, Intel matrix storage
Failure to test will accumulate performance issues, failure, accumulate, performance, issue,
Figure out where the voice location is and move, voice, location, move, location,
Fixture doesn't have to be on the server, Fixture, doesn't, have to, server,
Found and read the floppy, found, read, floppy
Getting the blue screen of death, blue screen of death
Goals were to implement a virtual machine, virtual machine, goal, implement
GUI is done, gui, done ,
GUI is not done, gui, not, done
Had some successes in agile Extreme Programming, agile, success, extreme programming
Had to tweak a little, tweak
"Has ""Write Once, Run Anywhere""", Write Once, Run anywhere ,
Has a quick and affordable delivery, quick, affordable, delivery
Has automated video editing tool, automated, video, editing, tool,
Has Dell computer, Dell, computer ,
Has installed file sharing software, file, sharing, software
Has multiple configurations for different platforms, configuration, different, platform
Has no internal CD or DVD drive, CD, DVD, drive, no,
Has processor designs that have multiple computing engines, processor, computing, engine, design,
Has the infrastructure and technology, infrastructure, technology ,
Has to store the data in the session, session, data, store
Have a problem with Windows, problem, have, window
Have established connection to the internet, connection, internet ,
Have the edit displayed, edit, displayed ,
Have tried lots of anti spyware software, anti spyware, lots, tried
Have tried to install some codecs, install, codec ,
He doesn't have a prior understanding of the work, he, doesn't have, prior, understanding, work
Here is the brief introduction, here, brief, introduction
How are you?, how are you
I am available for testing, I, available, testing
I am done (with the current task), I, done ,
I am frustrated, frustrated, I ,
I am getting application error, get, application, error

I am getting error messages, error message, get ,
I am getting the correct transcript, getting, correct, transcript
I am trying the beta version, beta, try ,
I can get all that done, can, get, done, all, I
I can get done by Friday, I, can, get, done, Friday
I can get done by Monday, I, can, get, done, Monday
I can get done by next month, I, can, get, done, next month
I can get done by next week, I, can, get, done, next week
I can get done by Thursday, I, can, get, done, Thursday
I can get done by Tuesday, I, can, get, done, Tuesday
I can get done by Wednesday, I, can, get, done, Wednesday
I can make it through, can, make, through
I can provide the information, can, provide, information
I can read it locally, read, locally ,
I can't reinstall Windows, can't, reinstall, window
I checked the config, check, config ,
I checked the configuration, check, configuration ,
I connected the pendrive, connect, pendrive ,
I could have everyone sitting like this, everyone, sitting, like, this,
I couldn't get rid of the spyware, spyware, rid, couldn't
I didn't check the config, didnt' check, config ,
I didn't check the configuration, didn't check, configuration ,
I didn't know how, didn't know, how ,
I don't care, don't, care ,
I don't get it, don't, get, it
I don't have a prior understanding of the work, I, don't have, prior, understanding, work
I don't have plans anymore, I, have, plans, don't,
I don't have the plan, don't, have, plan
I don't know, I don't know
I don't know how to do it, don't, know, how
I don't know why, don't, know, why
I don't really have plans anymore, I don't, have, plans
I don't think we have to separate, don't think, have to, separate
I forget everything, forget, everything ,
I forgot everything, forgot, everything ,
I got the script going, script, going ,
I guess I didn't know how, I guess, I didn't know, how
I have bad feeling about this, have, bad feeling, about
I have been looking at a sound level, look, sound, level
I have carefully examined, I, carefully, examined
I have changed the e-mail address, changed, email, address
I have debugged the program, debug, program ,
I have different versions, different, versions, have
I have finished it, finished, I ,
I have no other option but to restart and lose all data, no, option, restart, lose, data
I have not analyzed yet, have, not, analyzed
I have not done a whole lot, have, not, done, lot,
I have not finished it yet, not, finished ,
I have spent few hours making a simple prototype, prototype, hour, making
I have to decide, I, have to, decide
I have to know, have to, know ,
I have to perform it, perform, have to ,

I have tried different versions, have tried, different, version
I haven't analyzed yet, haven't, analyzed ,
I haven't done a whole lot, haven't, done, lot
I haven't finished yet, haven't, finished ,
I imported some of the video, imported, some, video
I imported them, imported
I need to perform it, perform, need ,
I need to work on writing the thesis, writing, thesis, work on
I set it up (the system), set, up ,
I set it up as recommended, set it up, recommended ,
I setup the server, setup, I, server
I should be able to get done by Friday, should, able, get done, by Friday,
I should be able to get done by Monday, should, able, get done, by Monday,
I should be able to get done by next month, should, able, get done, next month,
I should be able to get done by next week, should, able, get done, next week,
I should be able to get done by Thursday, should, able, get done, by Thursday,
I should be able to get done by tomorrow, should, able, get done, by tomorrow,
I should be able to get done by Tuesday, should, able, get done, by Tuesday,
I should be able to get done by Wednesday, should, able, get done, by Wednesday,
I should be able to get done in about a month, should, able, get done, in, month
I shown you this afternoon, shown, afternoon ,
I shown you this morning, shown, morning ,
I shown you yesterday, shown, yesterday ,
I sorted problems with EJB, ejb, sorted, problem
I started looking at some of the videos, start, look, video
I think following the sound would be better, following, sound, better
I think I am going to move onto the next, going to, move onto ,
I think in less than a month, less than a month
I think in less than a week, less than a week
I think in less than a year, less than a year
I took your words, took, your, words
I tried CMU Sphinx, CMU, Sphinx ,
I tried different versions, tried, different, versions
I tried Dragon Naturally Speaking, tried, Dragon Naturally Speaking ,
I tried face recognition, face recognition, tried ,
I tried face recognition open source programs, face, recognition, open, source,
I tried Microsoft Speech, tried, Microsoft, Speech
I tried open CV, CV, open ,
I tried out few open source programs, open, source, program
I want to do 'proof of concept', proof, of, concept
I want to get the other parts we did on the screen, parts we did, on, screen, want,
I want to give it all up, give, it, all, up,
I want to know how to solve the problem, know, solve ,
I want to try out on the different machine, try out, different machine ,
I want to try out on the same machine, try out, same machine ,
I will check the config, will check, config ,
I will check the configuration, will check, configuration ,
I wish I could do as requested, I wish, I could ,
I won't start anything by Friday, won't, start, friday
I won't start anything by Monday, won't, start, monday
I won't start anything by Saturday, won't, start, saturday
I won't start anything by Sunday, won't, start, sunday

I won't start anything by Thursday, won't, start, thursday
I won't start anything by Wednesday, won't, start, wednesday
I wouldn't waste my time, wouldn't, waste, time
I wrote the letter, wrote, letter ,
I wrote up that script, wrote, script ,
I wrote up that script about the keyword extraction, wrote, script, about, keyword, extraction
Ideas and advices on how to improve the coding, advice, coding, idea
I'll try to update the output, update, output, try
I'm a little concerned, little, concerned ,
I'm feeling better, I'm, feeling, better
I'm feeling good, feeling, good ,
I'm going to hold a presentation, going to, hold, presentation
I'm going to hold a seminar, going to, hold, seminar
I'm going to start with a brief introduction, going to start, brief introduction ,
I'm going to try it out, going to, try ,
I'm going to try that one out, going to, try, that
I'm reading out, reading, out ,
I'm ready to go, ready, go ,
I'm researching, I'm, researching ,
I'm researching about it, researching, about ,
I'm still working on it, still, working, on, it,
I'm still working on that, still, working, that
I'm still working on the project, still, working on, project
I'm trying to demo my application, try, demo, application
I'm trying to do that, trying, do, that
I'm trying to get all that stuff done, trying to, stuff, done
I'm trying to work on the project, trying, project, work on
I'm working on it now, I'm, working ,
I'm working on the new project, working, new project ,
Important in computer games, computer game, important ,
Important in information visualization, information, visualization, important
Important in medical imaging, medical, imaging, important
Installed a new hard drive, install, hard drive ,
Installed a new mouse, install, mouse ,
Introduced most powerful graphics processor, processor, introduce, graphic, powerful,
Introduced performance breakthrough in wireless connectivity, wireless, performance, breakthrough
Is it tomorrow or next week?, is it, tomorrow, next week
Is it tomorrow?, is it, tomorrow ,
Is it yesterday?, is it, yesterday ,
It can automatically find out, can, automatically, find out
It can follow the sound, follow, sound ,
It can get the correct transcript, get, correct, transcript
It can still be useful, still, can, useful
It can't handle everything, handle, can't, everything
It compiled, compiled
It doesn't have to be, doesn't, have, to be
It has a higher pitch, higher pitch
It has activation exception, activation, exception ,
It has already bound exception, already, bound, exception
It has application exception, application, exception ,
It has arithmetic exception, arithmetic, exception ,

It has backing store exception, backing, store, exception
It has bad location exception, bad, location, exception
It has certificate exception, certificate, exception ,
It has class cast exception, class, cast, exception
It has class not found exception, class, not, found, exception,
It has clone not supported exception, clone, not, supported, exception,
It has data format exception, data, format, exception
It has destroy failed exception, destroy, failed, exception
It has font format exception, font, format, exception
It has general security exception, general, security, exception
It has illegal argument exception, illegal, argument, exception
It has interrupted exception, interrupted, exception ,
It has invalid midi data exception, invalid, midi, data, exception,
It has IO exception, IO, exception ,
It has naming exception, naming, exception ,
It has no such method exception, no such method, exception ,
It has null pointer exception, null, pointer, exception
It has out of memory exception, out of memory, exception ,
It has parse exception, parse, exception ,
It has parser configuration exception, parser, exception, configuration
It has parser error, parser, error ,
It has printer exception, printer, exception ,
It has privileged action exception, privileged, exception, action
It has runtime exception, runtime, exception ,
It has the mechanical device to do that, mechanical, do that ,
It has thread death, thread, death ,
It has unsupported flavor exception, unsupported, flavor, exception
It have located, I, located ,
It is a collection of latest technical gadgets and software, collection, latest, gadgets, software,
It is a general place to write down the findings and research, general place, research, findings, write,
It is a versatile Bluetooth adapter for PCs, Bluetooth, pc, adapter
It is able to follow, able to, follow ,
It is almost complete, almost, complete ,
It is available for testing, available, testing ,
It is cached in the session to ease the load, cached, session, load, ease,
It is complete, complete
It is executed by a person's voice, execute, person's, voice
It is getting page fault, page fault, get ,
It is ideal for heterogeneous computing, heterogeneous, computing ,
It is important to do it right, important
It is important to realize it, important, realize ,
It is meant to exploit a new generation of technology, exploit, new, technology
It is not automated, not, automated ,
It is not complete, complete, not ,
It is not possible, not possible
It is only a short term solution, short, term, solution
It is out of memory, out of memory
It is part of the continuous integration process, continuous, integration, process
It is released as Free Open Source software, release, open source, free
It is released through GNU General Public License, release, public, license
It is secure, secure

It is sort of done, done, sort ,
It is trying to come up with different ideas, different, ideas, come up
It is very compact and lightweight, compact, lightweight ,
It looks at a person and the head, look, person, head
It says Insert the disk labeled, insert, disk, labelled, say,
It says it can't find a class, can't find, class ,
It says it can't find a classpath, can't find, classpath ,
It says the password is invalid, password, invalid ,
It supplies interfaces to render 3D objects, interface, 3D ,
It supports automated performance testing, automated, performance, testing
It uses Java3D, Jave3D
It uses JUnit, JUnit
It was not working yesterday, work, not, yesterday
It was working yesterday, work, yesterday ,
It wasn't long ago, wasn't, long, ago
It will accumulate, will, accumulate ,
It will be allowed, will, allowed ,
It will be analyzed, will, analyzed ,
It will be conducted, will, conducted ,
It will be executed, will, executed ,
It will be included, will, included ,
It will be listed, will, listed ,
It will be noted, will, noted ,
It will be packaged, will, packaged ,
It will be recorded, will, recorded ,
It will compile, will, compile ,
It will not accumulate, will, not, accumulate
It will not be analyzed, will, not, analyzed
It will not be conducted, will, not, conducted
It will not be executed, will, not, executed
It will not be included, will, not, included
It will not be listed, will, not, listed
It will not be noted, will, not, noted
It will not be packaged, will, not, packaged
It will not be recorded, will, not, recorded
It will not be used, will, not, used
It will not compile, will, not, compile
It will not differ, will, not, differ
It will not have it, will, not, have
It will not list, will, not, list
It will work, will, work ,
It works with wireless microphone, work, wireless, microphone
It's a general rule, general rule
It's automated, automated
It's on Friday, on, friday ,
It's on Monday, on, monday ,
It's on Saturday, on, saturday ,
It's on Sunday, on, sunday ,
It's on Thursday, on, thursday ,
It's on Tuesday, on, tuesday ,
It's on Wednesday, on, wednesday ,
It's possible, it's, possible ,

It's random, random
It's so good, so, good ,
It's talking too fast, talk, too, fast
It's terrible, terrible
It's too early to say, early, to say ,
It's very complicated, complicated
I've been busy, busy
I've been busy with course work, busy, course ,
I've been looking at the sound level, looking at, sound level ,
I've got some recording, got, some, recording
J2EE is for enterprise applications, J2EE, enterprise application ,
J2ME is for mobile applications, J2ME, mobile application ,
Java had multiple configuration built for different platforms, Java, configuration, different, platform,
Java is available under General Public License, Java, General Public License ,
Java is available without a charge, java, available, charge, without,
Java is controlled through the Java Community Process, Java, control, Java, Community, Process
java is popular, java, popular ,
Java was called Oak, java, oak ,
JBoss server shouldn't execute column fixture, JBoss, shouldn’t execute, column fixture
JRE is a subset of JDK, JRE, JDK, subset
Keeping the log for the research progress, research, progress, log
Large data centers use virtualization, data center, virtualization ,
Let them be familiar with JUnit, Junit, familiar, let them
Let's do it, Let's, do ,
Let's get down to business, get down, business ,
Let's get started, let's, started ,
Let's have a look, let's, look ,
Let's reintroduce, let's, reintroduce ,
Let's see if it works, let's see, if, works
Let's see if this works, let's see, if, this, works,
"Links to a tree structure, scene graph", scene graph, tree, structure
Little is described about performance testing in agile environment, little, describe, performance, testing, agile
Located the drivers, locate, driver ,
Logged onto the server from home, log, server, from, home,
Looked at several of the postings, look, several, posting
Looked at the display property settings, look, display, property, setting,
Maybe I should get some help, I should, get, help
Need some new ideas, need, new, idea
Need to have Windows reinstalled, windows, reinstall ,
Need to make sure the idea could be done before I start, make sure, done, could, done,
Need to review the video, review, video ,
Need to talk about it with someone, need, talk ,
Need to test everything, test, everything ,
None of them are working, none of them, working ,
Nothing is wrong so far, nothing, wrong ,
Nothing seems to help, nothing, seem, help
"On the same machine, it works", on, same machine, it, work,

Partitioned and formatted it, partition, format ,
Performance problems should be exposed, performance, problem, exposed
Plan to cache some data, cache, data ,
Popup blocker is not working, popup, blocker, not, working,
Pretty much everything's on hold, everything, on hold ,
Pricing is comparable, pricing, comparable ,
Problem occur when the drive is pulled out without going through the tray bar icon, problem, pulled out, without going, tray bar icon,
Promote frequent and timely treatment of system issues, frequent, timely, treatment, issues,
Random error messages appearing, random, error, message, appear,
Realized importance of conducting early and frequent performance investigations, early, frequent, conduct, performance, test
Removed the indicated entries, indicated, entry, remove
Scan for virus, scan, virus ,
Scan the files for worm or virus infection, scan, file, infected, virus, worm
Screen goes black after logo displays, screen, black, logo, display,
Security is an issue when data is transmitted without the wires, security, data, transmitted, without, wires
Seems to work well enough, work, well enough ,
She doesn't have a prior understanding of the work, she, doesn't have, prior, understanding, work
So next time is on Friday, next time, Friday ,
So next time is on Monday, next time, Monday ,
So next time is on Saturday, next time, Saturday ,
So next time is on Sunday, next time, Sunday ,
So next time is on Thursday, next time, Thursday ,
So next time is on Tuesday, next time, Tuesday ,
So next time is on Wednesday, next time, Wednesday ,
System restore is done, system restore, done ,
Takes advantage of the manufacturing technology, advantage, technology, manufacturing
Test automation plays a vital role in agile projects, test automation, agile, project, role,
Thank you, thank you
Thanks, thanks
That was easy, easy
That was hard, hard
That's a totally different presentation, totally, different, presentation
That's a waste of time, waste, of, time
That's about all that's happened, all, happened, about, that's,
That's my goal for next, my goal, next ,
That's right, that's, right ,
That's the first sound, first sound
That's the other one, other, one, that's
The adapter plugs to any USB port, adapter, plug, USB port
The application uses Flickr API, Flickr, API ,
The client is creating the error, client, creating, error
The code actually does what it is supposed to do, code, what, supposed to do
The code is working, code, working ,
The code works, code, works ,
The complete source is available, source, available ,
The computer is on a workgroup, computer, workgroup ,
The CPU is at 100 percent, 100 percent, cpu ,
The CPU is at 100%, 100%, cpu ,

The dog could move to predefined points, dog, move, predefined, points,
The errors keep coming out, error, keep, coming
The extraction technique is not working, extraction, technique, working, not,
The extraction technique is working, extraction, technique, working
The fields and icons are grayed out, field, icon, grayed out
The icons are grayed out, icon, grayed out ,
The laptop is small, laptop, small ,
The last known good configuration is loaded, last known, configuration, load
The logon process is terminated unexpectedly, logon, process, terminated, unexpectedly,
The memory could not be read, memory, read, could, not,
The metrics is based on the results of a survey, metrics, based, survey, result,
The noise is blended in, noise, blended ,
The notebook has a hard disk, notebook, hard disk
The package is in the directory, package, directory ,
The performance expectation should be tested, performance, expectation, test
The performance matches the speed of the fastest supercomputer, performance, match, speed, fastest, supercomputer
The performance must be tested across the entire software development cycle, performance, test, software , development, cycle
The performance testing needs to be test-driven, performance, test, automated
The player is recently upgraded, player, upgrade ,
The plug and play makes installation simple and easy, plug and play, installation ,
The problem goes away but comes back after a while, problem, go, away, come, back
The project requires Flickr, Flickr, project ,
The punctuations are the hardest, punctuation, hardest ,
The purpose of the paper is to describe a tool, paper, describe, tool
The result is always same, result, always same,
The result is too long, too long
The security is configurable, security, configurable ,
The summarization technique is having a difficulty distinguishing differences in usage, summarization, difficulty, distinguishing, difference,
The summarization technique is not having a difficulty distinguishing difference in usage, summarization, difficulty, distinguishing, difference, not
The system is complete, system, complete ,
The system is designed to handle some rough dialogs, system, handle, dialog
The system is in beta version, beta, is ,
The system is infected with virus, system, infect, virus
The system is not complete, system, complete, not
The task is impossible to do, impossible
The task is possible to do if done right, it, possible ,
The technology will be detailed in a paper to be presented, detail, paper, present
The tool can test execution time of frame rate, execution time, frame, rate
The tool can test execution time of loading and unloading objects, execution time, tool, test, loading, unloading
The tool can test execution time to build a scene, tool, test, execution, time, scene
The tool is for specific aspects of system performance, aspect, system, performance, specific,
The work is okay and I can get an okay result, okay, I can, that
Then let's get started, let's, get, started
There is a new problem, new, problem ,
There is no look and feel yet, look and feel, no, yet
There's so much noise, so much, noise ,
There's some error, error, some ,

There's some exception, exception, some ,
They are available for testing, available, testing, they
They are different, they, different ,
They are same, they, same ,
They don't have a prior understanding of the work, they, don't have, prior, understanding, work
This is impossible, impossible, this ,
This is not possible, not possible, this ,
This is one of the most impressive technical achievement, impressive, technical, achievement
Ths dog could move to predefined paths, dog, move, predefined, paths,
To get the dog not to fall off the table, dog, not, fall off, table,
To see if there is difference, to see, if, difference
Today is Friday, today is, friday ,
Today is Monday, today is, monday ,
Today is Saturday, today is, saturday ,
Today is Sunday, today is, sunday ,
Today is Thursday, today is, thursday ,
Today is Tuesday, today is, tuesday ,
Today is Wednesday, today is, wednesday ,
Tried another monitor and same result, tried, another, monitor
Tried anti spyware, anti, spyware, try
Tried changing the settings, changing, setup ,
Tried formatting the hard drive, try, format, hard drive
Tried many different files, tried, different, files
Tried many different variations, tried, different, variations
Try to change the settings, try, change, setting
Use the bioinformatics project to evaluate the effectiveness, bioinformatics, project, evaluate, effectiveness,
Used for bioinformatics project, bioinformatics, project ,
Used to be able to play avi files, used to, able, play, avi,
Uses the enhanced 128-bit data encryption, 128, bit, encryption, data,
Want to stop the auto update, want, stop, update
We are able to follow, able to, follow, we
We are done, we are, done ,
We are launching a site on the Internet, site, launch, internet
We can include it, we, can, include
We can only include session beans, we, can, include, session, beans
We found no evidence, we, found, no, evidence,
We got it to work, got it, work ,
We have come on Tuesday and we tried the test, come, Tuesday, we did that
We have to separate, we, have, separate
We haven't changed the package, haven't, changed, package
We introduced Scrum, scrum, introduce ,
We tested all files under the directory, tested, files, under, directory,
We tested all things under the directory, tested, all, under, directory,
We tried where you read all the text within a time, you, read, all, time, with
Web browser has the ability to run applet, applet, browser, run
What are the current activities?, what, current, activities
What are they doing?, what, are they, doing
What are you doing here?, what, are you, doing, here,
What are you doing?, what, are you, doing
What are you saying?, what, are you, saying
What can I do?, what, can, I, do,

What can they do?, what, can, they, do,
What can they see?, what, can, they, see,
What can you do?, what, can, you, do,
What changes are they making?, what, changes, making
What did I do, what, did, you, do,
What did you do since the last meeting?, what, did you do, since, last meeting,
What did you do since the last time?, what, did, since, last time,
What have you done?, what, have you, done
What if I can't?, what, I can't, if
Whatever it is, whatever it is
What's the next step?, what's, next step ,
What's the plan for the next week?, what's, plan, next week
What's the plan for tomorrow?, what's, plan, tomorrow
What's the plan?, what's, plan ,
What's to the next Friday?, what's, to, next Friday
What's to the next Monday?, what's, to, next Monday
What's to the next Saturday, what's, to, next Saturday
What's to the next Sunday, what's, to, next Sunday
What's to the next Thursday, what's, to, next Thursday
What's to the next Tuesday, what's, to, next Tuesday
What's to the next Wednesday, what's, to, next Wednesday
When is it?, when is it
When is the meeting?, when is, meeting ,
When was the last meeting?, when, last meeting ,
whenever it is, whenever it is
Where is it?, where is it
wherever it is, wherever it is
Will be finished soon, finish, soon ,
Will be used widely for computers, use, computer ,
Will contain movies, contain, movies ,
Will contain movies and tv shows, contain, movies, tv
Will demonstrate an experimental computer chip, demonstrate, experimental, chip
Working on Alan, working on, alan ,
Working on Digital tabletop, working on, tabletop ,
Working on Fitness, working on, fitness ,
Working on Mase, working on, Mase ,
Working on Speech recognition, working on, speech recognition ,
Working on summarization, working on, summarization ,
Working on survey, working on, survey ,
Working on testing, working on, testing ,
Working on the analysis, working on, analysis ,
Working on the report, working on, report ,
Working on the thesis, working on, thesis ,
Write a test case before the code, test case, write, code
Write program with few defects, write, few, defect
You can get all that done, can, get, done, all,
You can get done by Friday, you, can, get, done, Friday
You can get done by Monday, you, can, get, done, Monday
You can get done by Saturday, you, can, get, done, Saturday
You can get done by Sunday, you, can, get, done, Sunday
You can get done by Thursday, you, can, get, done, Thursday
You can get done by Tuesday, you, can, get, done, Tuesday

You can get done by Wednesday, you, can, get, done, Wednesday
You can't move around a lot, can't, move around, lot
You finish the task, you, finish, task
You shouldn't have multiple tasks, shouldn't have, multiple, task
You start a task, you, start, task