Towards Efficient Human Machine Speech Communication: The Speech Graffiti Project
Stefanie Tomko, Thomas K. Harris, Arthur Toth, James Sanders, Alexander Rudnick, Roni Rosenfeld
School of Computer Science, Carnegie Mellon University, Pittsburgh PA 15213

1 Introduction
As the most common mode of human-human interaction, speech can be considered an ideal medium for human-machine interaction. Speech is natural, flexible, and most humans are already fluent in it. Using speech allows users to simultaneously perform other tasks, which may be related to the spoken task (e.g., when pointing) or not. Machine speech also requires modest physical resources and can be scaled down to much smaller and much cheaper form factors than can visual or manual modalities.

Technology now exists for allowing machines to process and respond reliably to basic human speech, and it is currently being used as an interface modality in several commercially available applications, such as dictation systems (e.g., IBM ViaVoice® [18], Dragon NaturallySpeaking® [8], Philips SpeechMagicTM[23]), web browsers (e.g., Conversay Voice SurferTM[7]), and information servers (e.g., HeyAnitaTM [14], 1-800-555-TELL™ [1]). However, we believe that speech would achieve even higher adoption as an interface technology if certain fundamental limitations were addressed, particularly

- Recognition performance
- Language habitability (for users)
- Ease of development (for implementers)

“Natural” interaction with computers has often been cited as a primary benefit of speech. The concept of talking with a machine as fluently and comfortably as with another human being has attracted funding and interest. However, from the user’s perspective, fully natural communication may not be the most desirable option. For instance, Shneiderman [26] suggests that “natural” communication may actually be too lengthy for frequent, experienced users, who expect a system to give them information as quickly as possible. Baber has explored the disadvantages of natural language for automatic speech recognition and cites studies in which users’ natural inclination for talking to computers was to be “short, succinct and task specific; using simple imperative commands and … a restricted vocabulary” [2]. Following these observation, our research focus has been on exploring speech as an efficient input/output modality rather than as a medium for natural communication.

The Speech Graffiti interface is the result of our research into these issues. This paper presents further motivation for creating such an interface, describes its general characteristics, shows examples of its use, summarizes user study results, and describes other aspects of the system such as the development toolkit and a user tutorial.

2 Background

2.1 The concept of a “simple machine”
For our approach to speech interfaces, we distinguish between simple and intelligent machines, and focus our efforts on interfaces to simple machines. We consider intelligent machines to be those that have been programmed with advanced reasoning and problem-solving abilities, whereas with simple machines, high-level intelligent problem solving is performed by the human user. In such cases the machine is only a tool for acquiring necessary information, modifying this information, or issuing instructions to the
desired service. We find speech interaction with simple machines to be interesting because it sidesteps the artificial intelligence problems of natural language in favor of the ubiquitous nature of speech interaction. We believe that in the near future people will be surrounded by hundreds of simple machines with which they will want to interact.

2.2 Speech user interface styles

Although speech recognition technology has made spoken interaction with simple machines feasible, no suitable universal interaction paradigm has been proposed for facilitating effective, efficient and effortless communication with such machines. In general, approaches to speech interfaces for simple machines can be divided into three categories: command-and-control, directed dialog, and natural language. These categories can be differentiated in terms of what users can say to the system and how easy it is for the system to process user input.

2.2.1 Command-and-control

Command-and-control systems severely constrain what a user can say to a machine by limiting its vocabulary to strict, specialized commands. Since such systems do not require overly complicated grammars, these can be the simplest types of systems to design, and can usually offer low speech recognition word-error rates (WER). However, they can be difficult or frustrating for users since, if strict, specialized input is required, users will have to learn a completely new set of commands for each speech interface they come in contact with. Under this paradigm, a user might have to learn five completely different voice commands in order to set the clock time on five separate appliances. While this may not be an unreasonable solution for any application that is used extensively every day (allowing the user to learn the interaction via repeated use), it does not scale up to an environment containing dozens or hundreds of applications that are each used only sporadically.

2.2.2 Directed dialog

Directed dialog (or dialog tree) interfaces use machine-prompted dialogs to guide users to their goals, offering a modest improvement over the touch-tone menu (IVR) interfaces so ubiquitous in telephone-based systems. In directed dialog systems, the user is often forced to listen to a catalog of options, most of which are likely to be irrelevant to their goal. Interactions are slowed by requiring the user to follow a certain path, generally inputting one piece of information at a time. When directed dialog systems allow users to interrupt prompts (barge-in) or to provide multiple pieces of information in their input, frequent users may be able to speed up their interactions by memorizing the appropriate sequence of words to input (as they might with keypresses in an IVR system), but these sequences are not necessarily valid across different applications. Users therefore must learn a separate interface pattern for each new system used and whenever an existing, familiar system is modified. From a designer’s perspective, such systems can be difficult to build because they require being able to break down an activity into the form of a dialog graph; maintenance is difficult because it may require rebalancing the entire tree as new functions are incorporated.

2.2.3 Natural language

Theoretically, in natural language interfaces, users can pose questions and give directives to a system using the same open, complex, conversational language that they would be likely to use when talking to another human about the same task (e.g. “When’s the first flight to New York Monday?” or “Did my stocks go up?”). By giving great freedom to the speaker, the user need not learn specialized commands nor work within a rigid access structure. However, a natural language interface puts a heavy burden on system developers who must incorporate a substantial amount of domain knowledge into what is usually a very complex model of understanding, and who must include all reasonably possible user input in the system’s dictionary and grammar. The large vocabularies and complex grammars necessary for such
systems and the conversational input style they are likely to generate can adversely affect speech recognition accuracy [13, 37].

More importantly, although the inherent naturalness of NL interfaces suggests that they should be quite simple to use, this apparent advantage can at the same time be problematic: *the more natural a system is, the more likely it is for users, particularly novice ones, to overestimate the bounds of, and form unrealistic expectations about such a system* [9, 22]. That is, although the goal of natural language systems is open, flexible communication, there are in practice significant limits to what any current system can understand, in terms of vocabulary, syntax, and functionality, and users will find that input that is acceptable in one natural language interface may be rejected in another. Additionally, as noted above, natural language communication may actually be too verbose for expert users who desire rapid interactions for frequently used applications [26].

2.3 The Speech Graffiti approach

We believe that the optimal style for speech communication with simple machines lies somewhere in the middle between natural language and command-and-control. The Speech Graffiti paradigm is more regular than natural language, yet more flexible than hierarchical menus or strict command-and-control. Speech Graffiti was modeled after two very successful non-speech interaction paradigms: Macintosh-style graphical user interfaces (GUIs) and the Graffiti® writing system for personal digital assistants (PDAs).

Graphical user interfaces became standardized with the development of the Apple Macintosh, which provided a development style guide stressing consistency in the interface look-and-feel of its applications. As a result of such standardization, global ease of use increased. That is, once a user learned the basic set of interactive behaviors (double-clicking, dragging, scrolling, etc.), those behaviors could be transferred to almost any Macintosh application, and when faced with a new Macintosh application, the user was much more likely to know what steps to take to achieve their interaction goals. The success of GUIs in the Macintosh world led to their eventual adoption in most interactive computing environments. We believe that a universal speech interface like Speech Graffiti can have the same benefit. If a speech interface user knows that, for instance, the system will always confirm whatever parts of the user input it understood, or that they can always say *options* to find out what they can talk about at a given point, learning how to use new Speech Graffiti applications should be significantly easier. The existence of a standardized look-and-feel (or more appropriately, “say-and-sound”) is particularly advantageous for spoken interfaces because the input behavior is invisible and must be remembered by the user.

The Graffiti® alphabet for PDAs requires users to slightly modify their handwriting in a standardized way in order to improve recognition performance (fig. 1). Although this requires that users actually invest a modest amount of time in learning the alphabet, the increase in handwriting recognition accuracy compared to that of systems designed to recognize users’ natural handwriting was so significant that the use of Graffiti has been posited as one of the main reasons for the commercial success of the Palm handheld [5]. Similarly, in the Speech Graffiti interface, users are asked to phrase their input in a certain way in order to improve speech recognition accuracy and reduce dialog complexity. Speech Graffiti users need to spend a short amount of time (5-15 minutes) learning the interface, but this is a one-time cost which is amortized by increased recognition accuracy and a GUI-like cross-application standardization. This standardization in turn minimizes the amount of learning with each new application encountered and allows that residual learning to be done “on the job”.

![Graffiti® alphabet](image.png)
3 Related work

Although much research has been conducted on the design of natural language spoken dialog systems, notably including MIT’s Jupiter system [39] and the DARPA Communicator project [35], and far less research has been done on more standardized speech interfaces.

There has been general support in the literature for many of the ideas incorporated into Speech Graffiti. The Telephone Speech Standards Committee has actively promoted the idea of universalizing commands in order to facilitate error recovery and decrease user training requirements (presumably across applications), and studies have been conducted to determine appropriate standard keywords for speech interfaces [11, 27, 31]. Like Speech Graffiti, the speech interface for a handheld computer described in [30] uses auditory icons for efficiency and unobtrusiveness. The use of short output lists and consistent vocabulary in Speech Graffiti matches the need for “continuous representation” in speech interfaces suggested in [19]. The idea of handling dynamic amounts of input (e.g. having a movie name and a theater name in the same utterance vs. splitting those two items into two separate utterances) according to speech recognition performance is used in the TOOT system [20], although the choice to split into separate utterances is system-driven in TOOT while in Speech Graffiti it is user-initiated.

The restriction on the form of Speech Graffiti user input means that it can be considered a subset language – an artificially constructed subset of natural language, designed for a specific purpose (though not necessarily for a specific domain). Sidner and Forlines have investigated the use of a subset language for a home entertainment center [29]. A user study with this language showed that subjects were able to complete all given tasks successfully with the subset language, and that their performance did not decline when performing tasks the following day, demonstrating that users were able to retain their knowledge of the language. Like Speech Graffiti, the subset language used in this study was designed to use simple, common English grammatical structures and a limited vocabulary. Unlike Speech Graffiti however, it was not necessarily designed to be adaptable to different domains, a feature that we believe can help make subset languages even more usable and retainable, and participants in the study had access to visual help while using the system.

Zoltan-Ford [38] investigated the use of restricted languages in both spoken and typed input and found that users generally did not mind having to use a restricted language. In fact, she found that study participants believed that computers naturally require consistency in input, and that even in human-human communication, some amount of adaptation to a partner’s speaking style is necessary.

At Hewlett-Packard, Hinde and Belrose have developed Computer Pidgin Language (CPL), which attempts to address the issue of poor recognition results in speech interfaces [15]. Like Speech Graffiti, CPL must be learned, and its developers compare the learning process to learning Graffiti for improved handwriting recognition. Rather than appropriating concepts from natural language however, CPL is a fully artificial language. CPL vocabulary was created by using genetic algorithms to determine a set of words with low confusability (for example, zeejoy, neefa, and fargoy). These acoustically distinct words can then be mapped to commands, but it appears that this mapping will be largely user-defined and not necessarily transferable across applications.

4 Speech Graffiti: design and general characteristics

Speech Graffiti is designed to provide regular mechanisms for performing interaction universals. Interaction universals are actions which are performed by users at one time or another in nearly all speech user interfaces; the set of universals addressed by Speech Graffiti was derived by analyzing several types of simple machines and application categories prior to developing the Speech Graffiti vocabulary.
These universals include actions involving help/orientation, speech recognition, basic system functions and application-type-specific functions. For this latter category, we recognize that different types of applications can have different interaction needs. For example, a transaction interface must include structures that allow the user to pay, bid, confirm, etc., while a system for controlling a gadget must include structures for giving simple commands (e.g. rewind, toast, turn off, etc.) and setting continuous variables (e.g. changing volume level). Although these structures may vary by application-type, they should be standardized for all applications of the same type.

In general, Speech Graffiti addresses these interaction universals by means of keywords and standard structures. Keywords are appropriate for some interaction universals which involve users’ performing certain, specific actions, while other interactions, such as confirmation and error handling, require a standardized protocol for input and output rather than a single keyword. This section will discuss the specific ways in which Speech Graffiti addresses these interaction universals.

To date, Speech Graffiti has been most fully developed in the domain of information access, and so the descriptions provided here are therefore specialized for that type of application. A representative Speech Graffiti dialog, adapted from Speech Graffiti MovieLine interactions during a user study, is shown in fig. 2. References to line numbers from this dialog will be written as \( n \).

4.1 Vocabulary

The vocabulary of a Speech Graffiti application consists of two parts: a set of universal keywords and an application-

---

**User1** Start over.
**System1** STARTING OVER.

**U2** Theater is the Manor, title is Abandon.
**S2** THE CINEMAGIC MANOR THEATRE, ABANDON.

**U3** What are the show times?
**S3** 2 show times: 1:25, 5:30.

**U4** Start over.

**U5** Theater is Maxi Saver, what is the address?
**S5** CARMIKE MAXI SAVER 12 THEATER: 2001 MOUNTAIN VIEW DR. IN WEST MIFFLIN.

**U6** Start over.

**U7** Options.

**U8** More.

**U9** More.

**U10** More.

**U11** Area options.

**U12** Area is Monroeville.
**S12** MONROEVILLE.

**U13** What are the theaters?
**S13** SHOWCASE CINEMAS PITTSBURGH EAST.

**U14** Theater is Showcase Cinemas Pittsburgh East.
**S14** SHOWCASE CINEMAS PITTSBURGH EAST.

**U15** What are the titles?
**S15** 11 titles: My Big Fat Greek Wedding, Abandon, Brown Sugar, [...].

**U16** Genre is comedy.

**U17** Where am I?
**S17** AREA IS MONROEVILLE, THEATER IS SHOWCASE CINEMAS PITTSBURGH EAST, WHAT ARE THE TITLES?, GENRE IS COMEDY.

**U18** What are the titles?
**S18** 5 titles: My Big Fat Greek Wedding, Brown Sugar, Jonah - A Veggie Tales Movie, [...].

**U19** Title is My Big Fat Greek Wedding.

**U20** What are the show times?
**S20** 4 show times: 3:45, 5:45, 7:45, 9:45.

**U21** Start over. Theater is Pleasant Hills.

**U22** Location is Pleasant Hills.

**U23** What is theater?
**S23** CARMIKE SOUTHLAND 9.

**U24** Theater is Southland Nine.

**U25** What is movie?

---

*figure 2. Sample Speech Graffiti MovieLine dialog.*
specific lexicon. The keywords, which will be discussed in the following sections, are summarized in Table 1. Some synonyms are also admitted, such as \textit{where were we?} for \textit{where was I?}

4.1.1 Application-specific lexicon

The size and contents of the application-specific lexicon are naturally determined by the functionality and complexity of each application, and will generally be quite a bit larger than the Speech Graffiti keyword set. Implementationally, in order to improve recognition accuracy, word strings that nearly always occur together in the Speech Graffiti interface are considered to be one word in the systems’ language models. This includes many open class items like movie titles and theaters (\textit{e.g.} \textit{My_Big_Fat_Greek_Wedding, Carmike_Village_Ten}). Because of its standardized structure, nearly all of Speech Graffiti’s keyword and syntactic phrase units are linked as word units as well (\textit{e.g.} \textit{scratch	extunderscore that, show	extunderscore time	extunderscore is}).

The current Speech Graffiti MovieLine lexicon includes around 450 “Speech Graffiti words,” approximately 100 of which are movie titles. Not counting movie titles, the Speech Graffiti MovieLine lexicon is composed of about 265 unique, ordinary English words. By comparison, a natural language system created to access the same movie information database contains around 690 words. The Speech Graffiti FlightLine lexicon contains about 260 Speech Graffiti words.

4.2 System output

In many speech interface situations, no visual display is available, so extra care must be given to the design of audio output to ensure that the system is able to convey information and express concepts to the user clearly and accurately. We believe however, and others have noted as well [21, 30], that it should not be necessary to always present information in the most verbose manner possible – and indeed, doing so would be a violation of the Gricean maxim that conversational contributions should be no more and no less informative than necessary [10]. Unnecessary verbosity and repetition in a system can become tiring; since we propose Speech Graffiti as a standard interface for systems that people might interact with several times a day, this effect is multiplied. Furthermore, one of the proposed advantages of Speech Graffiti is to facilitate efficient communication by allowing direct access to tasks that are too cumbersome for prompt- and menu-driven systems; using output that is too verbose could negate the effects of this strategy. Speech Graffiti implements its non-verbose output strategy via terse confirmation, segmented lists, and auditory icons.

4.3 Design details

4.3.1 Keywords

A single user utterance can contain one or more phrases, plus optional keywords that can also be used alone. For Speech Graffiti to be a truly universal interface, it must use a small set of words that non-technical users will feel comfortable with. Therefore, we have attempted to restrict our list of keywords to simple, everyday words and phrases rather than more technical terms like “execute.” Our original selection of keywords was based largely on our own intuitions about which words had simple,
unambiguous meanings and were at the same time relatively acoustically distinct. We later conducted a web-based user study to investigate the appropriateness of our keyword choices and to solicit possible alternative keywords from users [27]. Some of our original keyword choices (e.g. start over and repeat) performed well in the study, while others were replaced as a result (for example, options replaced now what? as the keyword for asking what can be said at any point in an interaction).

It is also desirable to keep the number of Speech Graffiti keywords small, in order to minimize the effort required to learn and retain them. Currently the system incorporates about seven main keywords plus another six navigation keywords, as shown in Table 1. Each keyword will be discussed individually where appropriate in this section.

4.3.2 Querying and specification

4.3.2.1 Phrases

The primary action in the information access domain is a database query. With Speech Graffiti, database queries are constructed by joining together phrases composed of slot+value pairs. Each user utterance can contain any number of phrases. Phrases are order-independent and each one can be used either for specifying constraints or querying. The use of slot+value phrases simplifies the work of the parser and roughly conforms to natural speaking patterns.

In our current implementations we have restricted the specification-phrase syntax to <slotname> is <value>, as in {U14} and the query-phrase syntax to what is the <slotname>?, as in {U15}. In order to reduce the command-and-control feel of Speech Graffiti however, these user input structures have been influenced by natural language. For instance, common synonyms can be accepted in many situations (e.g. movie and title both represent the same slot in a movie information system), and plural forms are accepted wherever they would naturally be used (e.g. what are the theaters? is equivalent to what is the theater?).

It is worth emphasizing that our current choice to restrict phrases to this simple syntactic format is not driven by limitations of the parsing technology. The Phoenix parser we use can accept far more elaborate grammars, and in fact we have used such flexible grammars in many other dialog applications at Carnegie Mellon. Rather, our choice is based on the arguments made earlier in support of a structured, simplified interaction language, and the many benefits it brings: increased recognition and understanding accuracy, improved system transparency (including clear lexical, syntactic and functional boundaries), and dramatically reduced development time. While we firmly believe that a semi-structured, semi-natural-language interface style will ultimately become ubiquitous in human-machine speech communication, we do not expect our current design choices to necessarily be the optimal ones. Our goal is to assess different designs in terms of the benefits they bring vs. their cognitive cost.

4.3.2.2 Context

Speech Graffiti can be set to retain or discard context depending on the requirements of individual applications. If context is turned off, parsed phrases are discarded after each query command. If context is retained, all parsed phrases since the last clearing of context are used to produce a database query string. Fig. 3 shows an example of context retention. When the user queries show times in their third utterance, context has not been cleared, so the system returns all three movies (from the previous query) and show times for the Manor theater.

| Theater is the Manor. |
| THE CINEMAGIC MANOR THEATRE. |
| What are the movies? |
| 3 TITLES: CHICAGO, DAREDEVIL, DOWN WITH LOVE. |
| What are the show times? |
| THE CINEMAGIC MANOR THEATRE. 3 TITLES, 15 SHOW TIMES: CHICAGO: 5:30, DAREDEVIL: 4:45, 7:30, […] |
| Genre is action, go ahead. |
| ACTION. 1 TITLE, 3 SHOW TIMES: DAREDEVIL: 4:45, 7:30, 9:50. |

Figure 3. Sample MovieLine dialog illustrating context retention.
Context is cleared using the start over keyword, or individual slots may be overwritten via re-specification. Our current implementation allows a slot to take only a single value at a time, so restating a slot with a new value overrides any previous value of that slot. This behavior will have to be altered for certain other domains in which slots are allowed to take multiple values. For example, in a pizza-ordering system, the “topping” slot should be allowed to hold more than one value at a time.

Speech Graffiti will execute a query immediately upon processing an utterance containing a query phrase. If, after hearing the response, the user would like to re-issue the same query (either with the same exact slots and values or after having re-specified some slot(s)), the go ahead keyword is used, as in the fourth user utterance in fig. 3.

In some applications it may be appropriate for users to request a complete set of information rather than querying a specific slot. For instance, in the FlightLine application, a user might want to know all the pertinent information about a flight that matches the user’s constraints. Rather than ask the user to issue a query phrase for each slot, the speaker can simply use the go ahead keyword to effectively query all unspecified slots, as shown in fig 4. This approach is most appropriate for applications using simple databases containing a single table. With a single table, “complete information” is easily defined as all of the fields for all of the records matching the user’s constraints. With more than one table, the join function required by the query renders the notion of “complete information” ill-defined; sorting this out would require problem solving and heuristics that we consider outside the bounds of Speech Graffiti’s basic purpose. For instance, our MovieLine application contains three tables: one for movie information, one for theater information, and one for show times. Without a filtering heuristic, a user requesting complete information for the movie The Matrix would end up hearing complete theater information—address, neighborhood and phone number—for each of the theaters where the movie is playing.

### 4.3.3 List presentation

In database applications, information that is returned to the user often takes the form of a list. We have implemented standard Speech Graffiti structures for the output and navigation of simple and complex lists, as determined by the number of query phrases issued (see fig. 5).

In keeping with our philosophy of presenting just as much information as is useful, our general strategy is to output information in small, manageable chunks. Therefore, in simple lists, we output three items at a time {S18}, or four if a split would result in a one-item orphan chunk {S20}. In complex lists, such as those which occur in the FlightLine output, the long, summary information for each item is also split into chunks. The notation {...} in our text examples {S8, S15} represents an auditory icon played at the end of a chunk to indicate that the list continues beyond the current chunk. {...} is currently implemented as a brief, three-beep signal intended to suggest the written punctuation for ellipsis (...).
The initial list chunk is prepended with a header indicating the size of the entire list, e.g. 11 TITLES {S15}. If the queried data is not available in the database, Speech Graffiti returns the string SORRY, THAT INFORMATION DOESN’T APPEAR TO MATCH ANYTHING IN OUR DATABASE.

4.3.4 List navigation
Speech Graffiti includes a suite of keywords for navigating through lists: more, next, previous, first, last and stop. The more keyword is used to access additional information of the same type, i.e. the next chunk at the same level of information. In complex lists or instances where the speaker has used go ahead to retrieve complete record information, the user can jump to the next item in the list (as opposed to the next chunk for the initial item) by saying next (in simple lists this keyword functions the same as more). This can be thought of graphically as navigating a two dimensional table, with more continuing horizontally and next continuing vertically. Previous returns the previous chunk in the list, first returns the initial chunk, and last returns the final chunk in the list. Each navigation keyword can be followed by an integer which allows the user to customize the size of the list returned. For example, last six would return the six items at the tail end of the list.

Splitting complex output into chunks not only helps to avoid information overload, but also enables the repeat keyword to act on current, smaller segments of information that the user might be interested in hearing again. If information were not chunked this way, the system would have to repeat an entire set of results from the very beginning. Interaction

4.3.5 Turn-taking
Speech Graffiti responds to each user input with a terse, standardized confirmation {S2, S14}; the user can then correct this item if necessary or continue on with their input. The repeat keyword replays the last system utterance in both normal and list-navigation modes.

4.3.6 Question answering
As discussed in section 4.3.2.1, queries are formed using the what is <slotname>? structure. Our earliest Speech Graffiti implementations included a terminator keyword which would signal that the user’s command was complete and ready to be executed by the system (e.g. theater is the Manor, what are the movies? go!). This eased the processing burden, since the speech recognition component simply needed to spot a terminator and then pass along the recognized string to the parser. The terminator keyword also increased the flexibility of phrase order: users could state a query phrase and then add specification phrases in subsequent utterances before sending the complete command to the system. However, we found that users had difficulty remembering to use the keyword. Once a query phrase (e.g. what are the movies?) has been uttered, users naturally expect the system to provide an answer, and the system has been refined to accommodate this expectation. As discussed in above in section 4.3.2.2, the go ahead keyword is used to re-execute a query phrase stored in context from a previous utterance.

4.3.7 Session management
Each session begins with a recorded introduction to the system (fig. 6); experienced users can barge-in on this introduction and start their interaction. When Speech Graffiti recognizes goodbye, it returns GOODBYE!, but the system remains active in case the input was misrecognized. If the user wants to continue, they can simply speak again; if not, they can just hang up. Since the information access applications are currently all telephone-based, sessions are initiated by the user calling the system.

WELCOME TO THE SPEECH GRAFFITI MOVIELINE. REMEMBER TO USE SPEECH GRAFFITI WHEN YOU’RE TALKING TO THE SYSTEM. IF YOU NEED HELP AT ANY POINT, JUST SAY “HELP.”

Figure 6. Text of session introduction.
4.3.8 Undo
See section 4.5.2 (correct ASR errors) below. Slots can also be cleared (i.e. set to no value) using the value anything, as in <slotname> is anything. The entire context can be cleared using the start over keyword.

4.3.9 Help primitives
4.3.9.1 What can the machine do?
Currently this is addressed by the options keyword, which returns a list of slots the user and system can talk about. A slightly different functionality could be provided by a keyword like what can you do? which should return a list of high-level application functions. This is probably more appropriate for multi-function applications, which we have not yet implemented in the information access domain. An example might be a movie information system which would provide information about movies and show times and also sell tickets.

4.3.9.2 What can I say?
As noted above, the options keyword returns a list of what can be said at any given point in the interaction. If used by itself {U7}, options returns list of available slots. If used as <slotname> options {U11}, a list of values that can be paired with <slotname> is returned. If the values that a particular slot can take make up a standard class or make too long of a list to be enumerated efficiently (even when split into chunks), the response to <slotname> options can be a description of the possible values instead. For instance, the system response to show time options is SHOW TIME CAN BE A TIME SUCH AS SEVEN THIRTY, OR A RANGE LIKE AFTER NINE O'CLOCK.

4.3.9.3 I need help
Currently, the help keyword allows the user to get help on the use of keywords and the basic form of Speech Graffiti input. Initially, help returns an example of how to talk to the system, plus a short list of appropriate keywords for either general use or list navigation, depending on where the user is in the interaction. On the second consecutive call, help returns this information with a more detailed explanation.

4.3.10 Speech channel primitives
4.3.10.1 Detecting automatic speech recognition (ASR) errors
Errors occur when the Phoenix parser cannot completely parse input passed to it. This may occur either because of a misrecognition of the users’ speech input or because the user simply did not speak within the Speech Graffiti grammar. Speech Graffiti uses a succinct confirmation strategy in which the system only confirms that input which it has understood. By responding this way, the system does not distinguish between different types of errors which may have occurred. If an error occurs in which user input is misinterpreted as acceptable input that does not match what the user said (e.g. the user says area is Monroeville and the system hears area is Squirrel Hill), the user can recognize the error from the explicit confirmation.

In general, Speech Graffiti confirms each entered, parsable-as-recognized phrase with a paraphrase of its value [S2, S12, S14]. If the system receives an utterance that cannot be fully parsed due to either type of error, it prepends its confirmation of any understood/parsable phrases with an auditory icon {S21}. This icon is represented as {CONF!} (for confusion) in our text and is currently implemented as an error-signifying beep. The system will respond with only {CONF!} if no part of the input was understood. On a third consecutive input that contains no parsable information, Speech Graffiti responds with the more verbose {CONF!} I'M SORRY, I'M HAVING TROUBLE UNDERSTANDING YOU. TRY AGAIN.
4.3.10.2 Correcting ASR errors

*Scratch that* is the primary Speech Graffiti keyword for correcting errors, although other strategies can be used as well. If used independently, *scratch that* clears a user’s *previous* utterance. If used in conjunction with other input, *scratch that* clears all preceding input from the *same* utterance, thereby allowing users to self-correct disfluencies. As noted in section 4.3.2.2, re-specifying a slot will override any value already stored there, so corrections can also be made this way. In the most extreme case, a user could opt to say *start over* and re-attempt their command from the beginning.

The phrasal structure of Speech Graffiti also helps to mitigate the effects of errors and reduce the amount of duplicated or unnecessarily re-confirmed information in subsequent utterances. That is, although expert users might enter several specification phrases and a query phrase in a single utterance, a user experiencing recognition problems can enter one phrase at a time, making sure it is successfully confirmed before moving on to the next phrase. This of course slows down the overall interaction, but can be used as a fallback strategy when the system’s recognition rate is low (for instance, if the user is speaking in a noisy room or has a strong accent).

4.3.10.3 End of speech

Speech Graffiti plays a quiet beep when Sphinx has determined the endpoint of an utterance and has started to process this input.

5 System architecture

The Speech Graffiti implementation is modular, with its various components residing on multiple machines spanning two platforms (Linux and Windows NT) (fig. 7). The dialog manager consists of an application-independent Speech Graffiti engine and an application-specific domain manager. The Speech Graffiti engine calls on the Phoenix parser [36], and the domain manager interacts with a commercial database package. These components together constitute a stand-alone, text-based version of the system, which can be developed and tested independently of the speech recognition, speech synthesis and telephony control components.

In our current setup, speech recognition is performed by the CMU Sphinx-II engine [17], using acoustic models based on Speech Graffiti applications. Statistical language models for the Speech Graffiti system were created using the CMU/Cambridge SLM Toolkit [6]. Speech synthesis is performed via unit-selection-based, limited-domain synthesis using the Festival system [3, 4]. All components are integrated using a Visual Basic framework and a socket interface where needed. Some Speech Graffiti systems access static Postgres databases; the MovieLine interacts with an Oracle database that is updated

![Figure 7. Speech Graffiti system architecture.](image-url)
semi-automatically from the internet via a series of Perl scripts.

6 Application generator

One of the acknowledged impediments to the widespread use of speech interfaces is the *portability problem*, namely the considerable amount of labor, expertise and data needed to develop such interfaces in new domains. During the early and mid 1990s, in the aftermath of the ATIS project [24], it was often estimated that several person-years of effort would be needed to build ATIS-like systems. Furthermore, at least some of the people involved in such an effort would have to be seasoned speech researchers with considerable experience in building such systems. Additionally, tens of thousands of utterances would need to be collected in Wizard-of-Oz style experiments in order to create a sufficiently robust grammar.

Speech Graffiti’s semi-structured interaction increases speech recognition accuracy and obviates the need for such vast in-domain data collection. The unified structure of Speech Graffiti-compliant interfaces also makes possible the automatic generation of new interfaces from a terse high-level specification. We have created a toolkit comprising all the necessary programs and files to create and run Speech Graffiti information access applications. Together, these components

- generate code for the domain manager which properly accesses the database;
- generate a grammar file for the Phoenix parser which both enforces Speech Graffiti interaction style and is consistent with the database content (*e.g.* column names, data types and named entities in open data types);
- generate a language model and pronunciation dictionary for the Sphinx speech recognition system which are consistent with the grammar; and
- properly cross-link these various knowledge sources so that multiple generated Speech Graffiti applications do not interfere with each other’s operation.

The application-specific variables are collected for insertion into the various components via an XML document. Application developers can either create this XML document from scratch using the Speech Graffiti Document Type Definition (DTD) as a template, or they can utilize the Speech Graffiti Web Application Generator. The Web Application Generator is a web-based program that lets the developer fill out a series of web forms, from which an appropriate XML document is derived. Regardless of whether the developer uses the web interface or manually codes the XML document, the Application Generator (in the form of a Perl script) is used to insert the application-specific information from the XML file into all of the components discussed above. Fig. 8 shows a schematic of this process. The user

![Figure 8. Speech Graffiti application generation process.](image-url)
is referred to [33] for more details on the Application Generator.

It should be noted however that Speech Graffiti should be considered more of an interaction style than a programming schema such as VoiceXML [34] or SALT [25]. Although those systems allow users to develop speech applications in a more standard and modular way, they are generally unconcerned with standardizing the actual user interaction. Thus, it is possible that Speech Graffiti and VoiceXML and/or SALT can be used together, although we have not yet explored this possibility.

7 Evaluation

In this section, we present results from user studies that have demonstrated that the Speech Graffiti approach is habitable – that is, users can learn it and apply their learning to complete tasks successfully. Furthermore, we have found that Speech Graffiti promotes greater user satisfaction and more efficient interactions compared to natural language interfaces.

7.1 Assessing Speech Graffiti habitability and retention

Our earliest studies were largely concerned with testing our various language design choices to inform improvements to the interface. For instance, as noted in section 4.3.1, we conducted a survey to determine the appropriateness of various keyword choices [27]. Some of our original keywords, like start over and repeat, were retained based on these results, while others were changed accordingly (e.g. now what? was changed to options). Design changes were also made as a result of our initial user studies [28]. For instance, in keeping with the <slotname> is <valuename> specification syntax, we originally required query phrases to follow a parallel form: <slotname> is what? However, this inversion of natural syntax proved too cumbersome for users. Also, as noted in section 4.3.6, these early studies showed us that users expected the system to respond with an answer once a query phrase was issued, and that it was too difficult to remember to always issue a separate, explicit execution command like go!.

7.2 Speech Graffiti vs. natural language

Our most comprehensive evaluation of Speech Graffiti to date has been a user study comparing Speech Graffiti and natural language interfaces [32]. Our goal was to determine whether users would prefer a more efficient yet structured interaction over one that was more natural, but perhaps less efficient. In addition, we also set out to compare various subjective and objective measures, including user satisfaction and task completion rates and times, across the two interfaces. In this study, participants made a series of queries to a movie information database, using either a Speech Graffiti interface (SG-ML) or a natural language interface (NL-ML). After completing a set of tasks and an evaluation questionnaire, participants repeated the process with the other system. System presentation order was balanced.

7.2.1 Participants

Twenty-three users (12 female, 11 male) accessed the systems via telephone in our lab. Most were undergraduate students from Carnegie Mellon University, resulting in a limited range of ages represented. None had any prior experience with either of the two movie systems or interfaces, and all users were native speakers of American English. About half the users had computer science and/or engineering (CSE) backgrounds, and similarly about half reported that they did computer programming “fairly often” or “very frequently.”

7.2.2 Training

Users learned Speech Graffiti concepts prior to use during a brief, self-paced, web-based tutorial session. Speech Graffiti training sessions were balanced between tutorials using examples from the MovieLine
and tutorials using examples from a database that provided simulated airline flight information. Regardless of training domain, most users spent ten to fifteen minutes on the Speech Graffiti tutorial.

A side effect of the Speech Graffiti-specific training is that in addition to teaching users the concepts of the language, it also familiarizes users with the more general task of speaking to a computer over the phone. To balance this effect for users of the natural language system, which is otherwise intended to be a walk-up-and-use interface, participants engaged in a brief natural language “familiarization session” in which they were simply instructed to call the system and try it out. To match the in-domain/out-of-domain variable used in the Speech Graffiti tutorials, half of the natural language familiarization sessions used the NL-MovieLine and half used MIT’s Jupiter natural language system for weather information [39]. Users typically spent about five minutes exploring the natural language systems during the familiarization session.

7.2.3 Tasks
Upon completion of the training session for a specific system, each user was asked to call that system and attempt a set of eight tasks (e.g. “list what’s playing at the Squirrel Hill Theater,” “find out & write down what the ratings are for the movies showing at the Oaks Theater”). Participant compensation included task completion bonuses to encourage users to perform each task in earnest. Regardless of which system they were working with, all users were given the same eight tasks for their first interactions and a different set of eight tasks for their interactions with the second system.

7.2.4 Assessment
After interacting with a system, each participant completed a user satisfaction questionnaire scoring 34 subjective-response items on a 7-point Likert scale. This questionnaire was based on the Subjective Assessment of Speech System Interfaces (SASSI) project [16], which sorts a number of subjective user satisfaction statements (e.g. “I always knew what to say to the system” and “the system makes few errors”) into six relevant factors: system response accuracy, habitability, cognitive demand, annoyance, likeability and speed. User satisfaction scores were calculated for each factor and an overall score was calculated by averaging the responses to the appropriate component statements.\(^1\) In addition to the Likert scale items, users were also asked a few direct comparison questions, such as “which of the two systems did you prefer?” For objective comparison of the two interfaces, we measured overall task completion, time- and turns-to-completion, and word- and understanding-error rates.

7.2.5 User satisfaction
After using both systems, 17 out of the 23 subjects (74%) stated that they preferred the Speech Graffiti system to the natural language interface. Mean scores for subjective user satisfaction assessments were significantly higher for Speech Graffiti overall and in each of the six user satisfaction factors, as shown in fig. 9 (by one-sided, paired t-tests: overall t = 3.20 df = 22, p < 0.003; system response accuracy t = 3.36, df = 22, p < 0.002; likeability t = 2.62 df = 22, p < 0.008; cognitive demand t = 2.39, df = 22, p < 0.02; annoyance t = 1.94, df = 22, p < 0.04; habitability t = 2.51, df = 22, p < 0.01; speed t = 5.74, df = 22, p < 0.001).

---

\(^1\) Some component statements are reversal items whose values were inverted for analysis, so that high scores in all categories are considered good.
All of the mean SG-ML scores except for annoyance and habitability are positive (i.e. > 4), while the NL-ML did not generate positive mean ratings in any category. The SG-ML’s lowest user satisfaction rating was in the habitability category, which involves factors related to knowing what to say to the system – a not unexpected issue with a subset language interface. For individual users, all those and only those who stated they preferred the NL-ML to the SG-ML gave the natural language system higher overall subjective ratings. Participants confirmed our suspicions that programmers and users with CSE backgrounds might be more amenable to the Speech Graffiti approach. In all categories, CSE/programmer subjects gave the SG-ML higher user satisfaction ratings, although the differences were significant in fewer than half of the categories.

We also compared users’ subjective assessments of the Speech Graffiti MovieLine based on whether they had used the tutorial system based on that same system or on the SG FlightLine, and found that training domain had a negligible effect on satisfaction ratings.

7.2.6 Objective assessments

7.2.6.1 Task completion

Task completion did not differ significantly for the two interfaces. In total, just over two thirds of the tasks were successfully completed with each system: 67.4% for the NL-ML and 67.9% for the SG-ML. Participants completed on average 5.2 tasks with the NL-ML and 5.4 tasks with the SG-ML (fig. 10). As with user satisfaction, users with CSE or programming background generally completed more tasks in the SG-ML system than non-CSE/programming users, but again the difference was not statistically significant. Training domain had no significant effect on task completion for either system: users who trained on the SG-ML completed an average of 5.45 tasks correctly, while users who trained on the FlightLine system completed an average of 5.42 tasks correctly. Considered along with the lack of difference in user satisfaction ratings, this indicates that users are largely able to transfer concepts from one Speech Graffiti application to another, and that although application-specific training systems might be used if available, they are not absolutely necessary for reasonable performance.

7.2.6.2 Time-to-completion.

To account for incomplete tasks when comparing the interfaces, we ordered the task completion measures (times or turn counts) for each system, leaving all in-completes at the end of the list as if they had been completed in “infinite time,” and compared the medians.
For completed tasks, the average time users spent on each SG-ML task was lower than for the NL-ML system, though not significantly: 67.9 versus 71.3 seconds. Considering incomplete tasks, the SG-ML performed better than the NL-ML, with a median time of 81.5 seconds, compared to 103 seconds.

### 7.2.6.3 Turns-to-completion
For completed tasks, the average number of turns users took for each SG-ML task was significantly higher than for the NL-ML system: 8.2 versus 3.8 ($F=26.4$, $p<0.01$). Considering incomplete tasks, the median SG-ML turns-to-completion rate was twice that of the NL-ML: 10 versus 5. This reflects the short-turn, one-concept-at-a-time style adopted by most users. Speech Graffiti flexibly supports both short and long turns, by allowing any number of phrases to be concatenated together or spoken in isolation. Users’ choices in this regard are often influenced by the prevailing recognition accuracy.

### 7.2.6.4 Word-error rate
The SG-ML had an overall word-error rate (WER) of 35.1%, compared to 51.2% for the NL-ML. When calculated for each user, WER ranged from 7.8% to 71.2% (mean 35.0%, median 30.0%) for the SG-ML and from 31.2% to 78.6% (mean 50.3%, median 48.9%) for the NL-ML. The six users with the highest SG-ML WER were the same ones who preferred the NL-ML system, and four of them were also the only users in the study whose NL-ML error rate was lower than their SG-ML error rate. This suggests, not surprisingly, that WER is strongly related to user preference.

To further explore this correlation, we plotted WER against users’ overall subjective assessments of each system, with the results shown in fig. 11. There is a significant, moderate correlation between WER and user satisfaction for Speech Graffiti ($r=-0.66$, $p<0.01$), but no similar correlation for the NL-ML system ($r=0.26$).

### 7.2.6.5 Understanding error
Word-error rate may not be the most useful measure of system performance for many spoken dialogue systems. Because of grammar redundancies, systems are often able to “understand” an utterance correctly even when some individual words are misrecognized. Understanding Error Rate (UER) may therefore provide a more accurate picture of the error rate that a user experiences. For this analysis, we only made a preliminary attempt at assessing UER. These error rates were hand-scored, and as such represent an approximation of actual UER. For both systems, we calculated UER based on an entire user utterance rather than individual concepts in that utterance. SG-ML UER for each user ranged from 2.9% to 65.5% (mean 26.6%, median 21.1%). The average change per user from WER to understanding-error for the SG-ML interface was $-29.2\%$. The NL-ML understanding-error rates differed little from the NL-ML WER rates. UER per user ranged from 31.4% to 80.0% (mean 50.7%, median 48.5%). The average change per user from NL-ML WER was $+0.8\%$.

### 7.2.6.6 Grammaticality
Independently of its performance compared to the natural language system, we were interested in assessing the habitability of Speech Graffiti: how easy was it for users to speak within the Speech Graffiti
Users' grammaticality tended to increase over time. For each subject, we compared the grammaticality of utterances from the first half of their session with that of utterances in the second half. All but four participants increased their grammaticality in the second half of their Speech Graffiti session, with an average relative improvement of 12.4%. A REML analysis showed this difference to be significant, F = 7.54, p < 0.02. Only one of the users who exhibited a decrease in grammaticality over time was from the group that preferred the natural language interface. However, although members of that group did tend to increase their grammaticality later in their interactions, none of their second-half grammaticality scores were above 80%. A more thorough longitudinal analysis over multiple sessions is needed to further assess changes in grammaticality over time.

The lowest individual grammaticality scores belonged to four of the six participants who preferred the natural language MovieLine interface to the Speech Graffiti one, which suggests that proficiency with the language is very important for its acceptance. Indeed, we found a moderate, significant correlation between grammaticality and user satisfaction for Speech Graffiti (a cursory analysis found no similar correlation for the natural language interface). The six users who preferred the natural language MovieLine generated 45.4% of the ungrammaticalities, further supporting the idea of language proficiency as a major factor for system acceptance.

Users’ grammaticality tended to increase over time. For each subject, we compared the grammaticality of utterances from the first half of their session with that of utterances in the second half. All but four participants increased their grammaticality in the second half of their Speech Graffiti session, with an average relative improvement of 12.4%. A REML analysis showed this difference to be significant, F = 7.54, p < 0.02. Only one of the users who exhibited a decrease in grammaticality over time was from the group that preferred the natural language interface. However, although members of that group did tend to increase their grammaticality later in their interactions, none of their second-half grammaticality scores were above 80%. A more thorough longitudinal analysis over multiple sessions is needed to further assess changes in grammaticality over time.

7.2.7 Summary of user study results

- Users rated the Speech Graffiti significantly better than a natural language interface to the same database in terms of system response accuracy, likeability, cognitive demand, annoyance, habitability, and speed.
- Participants successfully completed approximately the same number of tasks with each system (about two-thirds of the assigned tasks).
- Speech Graffiti users often took more turns to complete tasks than natural language interface users, but they completed tasks in slightly less time.
- In Speech Graffiti, there is a significant correlation between low word error rate and high user satisfaction, whereas no similar correlation developed for the natural language interface. This suggests that even if WER can be reduced for natural language systems, users may still have other difficulties with such systems, such as the efficiency of the interaction.

8 Future work

The results of our user studies have shown that, compared to users of a natural language speech interface, Speech Graffiti users had higher levels of user satisfaction, lower task completion times, and similar task completion rates, at a lower overall system development cost. We also found that task success and user satisfaction with Speech Graffiti were significantly correlated with grammaticality (how often users spoke within the grammar). This indicates that it is very important to help users speak within the grammatical bounds of voice user interfaces (particularly subset language ones). However, even after training, some users had difficulty speaking within a restricted grammar. In our comparative experiment, six of 23 participants preferred the natural language system. The experience of these six users provides a snapshot of frustrating interaction. In the Speech Graffiti system, they accounted for the highest word- and concept-error rates, the lowest task completion rates, and the four lowest grammaticality rates. (These
users also accounted for the four lowest task completion rates for the natural language system, which suggests that working with speech interfaces in general may pose problems for some users.) One defining characteristic of these six participants was that all but one of them belonged to the group of thirteen study participants who did not have computer programming backgrounds.

Based on our results to date, we plan to refine the Speech Graffiti system to improve the user experience and increase interaction efficiency for all users. However, we will specifically consider the experience of the six NL-prefering participants in our improvements. One interesting feature of the user input collected in this study was that when users spoke to the natural language system, their inputs distilled into nearly 600 grammatical patterns. However, when users were ungrammatical in the Speech Graffiti interface and spoke natural language to it rather than using the restricted syntax, their input distilled into less than 100 patterns. Noticeably absent from this input were conversational, non-topic items such as “could you please tell me” or “I would like to know about.” This suggests that simply knowing that they are speaking to a restricted-language system is enough to affect the types of input users provide to a system.

Our planned approach for future work in the information access domain is to implement a system of intelligent shaping help and adaptivity with the goal of increasing interaction efficiency. We plan to exploit the fact that users naturally restrict their input to some degree when told they are interacting with a simple, restricted language system. The goal is to create a system which can understand input that is less than conversational, but broader than strict Speech Graffiti. Since interaction at the Speech Graffiti level is expected to be less error-prone and more efficient, system prompts can then be used to shape user input to match this more efficient style. We propose that the implementation of such a shaping scheme can virtually eliminate the pre-use training time that is currently required to learn the Speech Graffiti system. This system benefits both long-term users, who will learn strategies for making their interactions more efficient, and one-time users, who should be able to complete tasks using the expanded language without necessarily having to learn the Speech Graffiti style.

In addition to helping first-time users learn the basics of the system, we also plan to implement adaptive strategies that can help novice users become experts and have even more efficient interactions. Such strategies might include making suggestions about more advanced keyword use, such as using the <slotname> is anything construction to clear individual slots or combining navigation keywords with integers to customize the length of query result sets. The system’s own interaction style could change for experts as well. As part of our shaping strategy, input confirmations will echo the <slotname> is <valuename> format of Speech Graffiti input to be more lexically entraining than repeating only the value. However, once a user has achieved proficiency with the <slotname> is <valuename> format, confirmations could probably switch back to value only to make the interactions faster and less repetitive.

8.1 Appliance control

As a framework for investigating the application of Speech Graffiti principles in the appliance-control domain, we built the Speech Graffiti Personal Universal Controller (SG-PUC). Its specification language and communications protocol effectively separate the SG-PUC from the appliances that it controls, enabling mobile and universal speech-based appliance control. The development of interfaces to numerous appliances and the results of user studies (described in [12]) have demonstrated the usefulness of the SG-PUC, indicating that a high-quality, low-cost human-appliance speech interface can be largely appliance-agnostic. As with in information domain, the use of a universal control language provides the benefit of clear, unambiguous semantics and low input perplexity. These factors translate into a more robust system, with fewer errors than functionally equivalent natural language speech interfaces.
Another potential area for experimenting with the Speech Graffiti approach is in the interactive guidance domain, in which the system leads the user through a series of steps to complete a task, such as repairing a mechanical part or baking a cake. Another area would be transaction systems, such as those that would allow users to make restaurant reservations or purchase movie tickets. Even in the information access domain, Speech Graffiti functionality could be expanded to include the addition, modification, and deletion of database records.

9 Conclusion

We have found Speech Graffiti to be a promising step in increasing the efficiency of human-machine speech interaction. Our system was designed to make recognition more reliable by regularizing the interaction style, and the lower word- and concept-error rates generated in our comparison study verify this approach. User study results also demonstrated that speakers can use Speech Graffiti well enough to complete tasks successfully and prefer the system to a less efficient natural language interface. However, our studies also demonstrated that learning and using Speech Graffiti successfully can still be challenging for some users. Our future research directions are aimed at reducing this challenge, opening the possibility for Speech Graffiti–like systems to be integrated into a variety of publicly-accessible applications.

Information access applications provide perhaps the greatest opportunity for Speech Graffiti systems. Requiring only a telephone for access, they generally access text databases which easily support mappings to Speech Graffiti slots and values. Transaction systems would be the natural next extension to such systems. The implementation of other types of systems like gadget control and interactive guidance introduces an interesting area of research questions on the idea of skill and learning transference, not just cross-domain, but cross-form.

As a modality, speech interaction is celebrated for its accessibility, portability, and ease of use. It is usually an extremely efficient mode of communication for human-human interaction. However, the current state of the art in speech and language processing and artificial intelligence does not allow for equally efficient human-computer speech communication. Speech Graffiti offers a step towards improving spoken human-computer interaction efficiency.

10 Acknowledgements

This work was supported by grants from Pittsburgh Digital Greenhouse, a National Defense Science and Engineering Graduate Fellowship and Grant No. N66001-99-1-8905 from the Space and Naval Warfare Systems Center, San Diego. The content of the information in this publication does not necessarily reflect the position or the policy of the US Government, and no official endorsement should be inferred.

11 References

1. 1-800-555-TELL™, http://www.1-800-555-tell.com/
