Utter Command:
Human-Machine Linguistics, Human-Machine Grammar, and a New Interface

Kimberly Patch, Boston Voice Users Group, July 12, 2005

Slightly modified September 15, 2005

SLIDE 0

Utter Command: Human-machine linguistics, Human-Machine Grammar, and a new interface

As you've heard, I'm here to talk about a new speech interface. Actually, I'm going to talk about three related things — human-machine linguistics, Human-Machine Grammar, and the new interface — Utter Command. I'll talk for about half an hour and then we'll have a 20 minute demonstration of the interface, which should leave plenty of time for questions.

Several years ago, when Dragon Systems was still Dragon Systems, I talked here about the speech macros I used and Joel Gould, the architect of Dragon Dictate and NaturallySpeaking, asked me a key question: how do you remember your commands.

My answer was honest, but not very good. I said I probably remember only about a third of the commands I make, and I make new commands sometimes without realizing I already have the same function in a macro that is worded differently. At the time I didn't have a better answer.

I do now.

The short answer is it is important to use a vocabulary and grammar designed for human-computer communications. Using the large body of grammar and words that developed naturally for communications among people doesn't work in a human-computer context.

SLIDE 1 — HML, HMG, Utter Command

To fully explain, I'm going to talk about the concept behind the better answer: human machine linguistics, then I'll talk about an implementation of that concept — Human-Machine Grammar — and then I'll demonstrate Utter Command, the first in a series of software products that uses a speech interface based on the Human-Machine Grammar. Utter Command works with the NaturallySpeaking speech engine.

So — at first glance, the concept of human machine linguistics might seem a bit strange. Why would speaking to a computer be any different from speaking to a person?

To explain, I'm going to ask you to think about three things: bicycles, talking, and apple peelers.

First, bicycles. Picture how difficult it might be to ride a bicycle that has been changed so that when you turn the handle bars to the right the wheel turns to the left and vice versa.

Now picture how difficult it might be to ride a unicycle.

I need a show of hands: who thinks the bicycle would be more difficult to learn? Who thinks the unicycle would be more difficult to learn?

It usually seems to people that the unicycle would be more difficult. A unicycle is difficult, but many people can learn it given time, effort, and perhaps youth. The changed bicycle, however, is nearly impossible to ride.

This is because, although the bike seems more familiar, the changed aspect goes directly against the way the body works. We're used to coordinating the way we move in a set way — this is an advantage for most things, but makes catching your balance by turning your hands one way while looking and leaning another way extremely difficult.

Going against the way you instinctively do things is a good way to see how difficult something really is.

The point is that the difficulty of a given task depends on how it dovetails with how we are accustomed to doing things.

This relates to human-machine linguistics because the human brain has specific abilities when it comes to manipulating words and grammar, and it is important when building a speech interface to take these into consideration and — metaphorically speaking — make sure the handlebar and wheel turn the same way.

SLIDE 2 — Human Machine Linguistics

Human-machine linguistics leverages the way the human brain processes language to make spoken communications with machines fast and efficient.

OK, now were going to talk about talking.

What words might you need to talk to someone in any given situation and on any given topic? There are many possibilities.

Now picture a conversation with a computer. What might you need to say to the computer? Here there are many fewer possibilities.

Even if I don't know what you want to do on the computer I can make a pretty good guess — whatever it is, it is likely to have something to do with bringing up a file and controlling aspects of a program.

The point I'm trying to make here is that you can much more easily predict the words you might use to communicate with a computer than you can predict the words you might use to communicate with a person.

Here's a typical English dictionary — it probably contains all the words you need to have a conversation in English with any given person.

When you're dictating text to a computer you also need access to a regular dictionary's worth of words. When you're using speech commands to control the computer, however, you don't need nearly as many possibilities.

I mentioned Human-Machine Grammar at the beginning of this talk. When talking to computers it is important to have a grammar designed for human-computer communications. Here's the Human-Machine Grammar dictionary — it contains all the words you need to control the computer. It's much, much smaller than the dictionary that contains words designed for humans to communicate with each other.

OK, now I want you to look at things yet another way.

Here's an apple peeler. You can see fairly easily how it works because it is bound by the laws of physics. You can see all the parts, and many of the parts are familiar. There is a pointy end for holding the apple, a crank for turning, and it's easy to see the relationship between the crank and the spiral screw.

Now think about the graphical user interface — when you really think about it, there's a reason for all the complaints about computer software design. It is less inherently obvious how software works than something mechanical like the apple peeler works. This is because software lacks physical constraints that can give you clues about how it can be used.

There are conventions, however. Once you get used to the way computers and graphical user interfaces like Windows work you know how to try out a program to learn what you can do — click the menus at the top of the window to find commands and dialog boxes that give you more commands, right-click on objects and different parts of the screen to find more menus.

In the case of the apple peeler, the physical properties of the machine's parts are true constraints — the machine simply can't do anything that goes against these.

In the case of the graphical user interface the constraints are artificial — the conventions of menus and clicking.

Now let's look at the speech interface in this light. Speech evolved as a means of communication among people and works very well most of the time. The constraints of this type of speech have to do with the way the human brain works. Because we use speech more or less effortlessly, these constraints are difficult to see.

This is a problem only because a speech interface designed for communications between a person and a computer is necessarily different from the speech interface we use among people. People can adapt to each other and to the task at hand — time-saving jargon happens naturally — while computers cannot.

So we're starting with a difficult problem. It gets worse. Speech as a computer interface doesn't map very well to the existing graphical user interface. Speaking steps that are designed to be carried out by the keyboard and mouse becomes tedious very quickly.

In saying all this I'm trying to prove to you logically that the speech interface is different enough that it takes some planning to use.

Now we've come back around to the short answer — simply taking the large body of grammar and words that developed naturally for communications among people and expecting it to work in human-computer communications is not nearly enough.

The differences between human-human and human-computer communications have made the current speech interface a bad fit, and this bad fit is the reason many people who use speech recognition software as it exists today say that it is difficult to use.

It doesn't have to be this way.

Slide 3 — Keys to a successful speech interface

Making it easier, however, means using wording that dovetails with instinct, like in the bicycle example, using a small, easy-to-remember vocabulary, because that's all that's required when you're commanding a computer, and providing an overall framework of constraints, or grammar rules, because the workings of a speech interface are not as obvious as an apple peeler.

This is the goal of Human-Machine Grammar.

SLIDE 4 — Human-Machine Grammar

Human-Machine Grammar is a system of words and rules designed to allow humans to communicate commands to computers. It takes into consideration that humans have a large natural language vocabulary that has evolved over thousands of years and that we use seemingly effortlessly, while computers do not yet have the capacity to understand the meaning of speech — every computer speech command must be mapped exactly to an action ahead of time.

Human-Machine Grammar is made up of a relatively succinct dictionary of words that are used in commands according to a concise set of grammar rules.

It also takes into consideration that while language seems easy for humans, different phrasings encompass a considerable span of cognitive effort. Human-Machine Grammar is designed to limit cognitive effort in order to free up as much of the brain as possible so you can more easily concentrate on the tasks you are using the computer to do.

The system is relatively easy for humans to learn, and computers can respond to the commands without having to decode natural language or be loaded down with very large sets of synonymous commands.

This grammar system has been informed by 4 fairly different books:

SLIDE 5 — Books That Have Informed Human-Machine Grammar

Words and Rules, by Stephen Pinker
Linked, by Albert-Laszlo Barabasi
The Psychology of Everyday Things, by Donald Norman
and *The Humane Interface*, by Jef Raskin

SLIDE 6 — Books — Relevant Points Words and Rules

Words and Rules points out how highly ordered language is, how difficult it should be to learn, and how natural it is for humans to learn. The research cited in the book is informed by MRI studies that watch blood flow as people think about words.

Here are some relevant highlights:

Human speech is the most difficult thing we do — our brains contain the largest number of neural connections that they ever will when we are toddlers learning speech.

Human speech has a lot to do with patterns, and it involves rules that are built into the way our brains work.

So — go with the general flow of human language in commanding a computer, and things seem easy. Go against that flow, and commands become much more difficult to remember and use.

SLIDE 7 — Books — Relevant Points plus Linked

Linked delves into the structure of the many networks in the world — the Internet, social networks, and relationships among words in a vocabulary, just to name a few.

The key piece of information from this book is the more connected a network, the easier it is to get from one point to another — the six degrees of separation concept.

This translates to the command vocabulary used by Human-Machine Grammar as well. The smaller and more connected a vocabulary, the easier it is to think of the words and structure you need for a given command.

SLIDE 8 — Books — Relevant Points plus the Psychology of Everyday Things

The Psychology of Everyday Things contains many insights about the interfaces all around us.

It points out how important visual cues are, and it hammers home the point that when people repeatedly make mistakes that seem stupid — like pushing the wrong side of a door or turning on the wrong stove burner — the root cause is almost always a design flaw.

The same is true of speech. A speech interface that is difficult to use points to a design flaw.

SLIDE 9 — Books — Relevant Points plus The Humane Interface

The Humane Interface contains many insights about computer interfaces.

It stresses how easy it is to trip someone up by messing with habits, and it points out the weaknesses of today's graphical user interface. It's ridiculous, when you think about it, to have to do a different series of clicks or a different key combination to carry out the same function — like adding the date — depending on which program you happen to be in.

This inconsistency means that once something becomes habit you'll start making mistakes like pressing the key combination to paste the date in Word when you are in Eudora, precisely because you're not thinking about it.

The graphical user interface also involves babysitting the computer through a multitude of steps — how many steps should you have to go through to access a particular file, folder or Web site? How much does it slow you down to have to click through five nested folders to get to a particular file? How many steps should you have to go through to send an email? 12, 7, 3?

The great thing about the speech interface is that it can address some of the inherent inadequacies of the graphical user interface — but only if it is designed to do so.

This is the thinking behind Human-Machine Grammar.

The important elements of language are words, context, and word order. Human-Machine Grammar leverages all three.

We balance several considerations when choosing the words that make up the human-machine dictionary, and when putting these words together to form commands.

Here are the basic human-machine command building guidelines:

SLIDE 10 — Command-Building Guidelines

We match the words used for a command as closely as possible with what the command does, and we use words the user sees on the screen when we can. This makes commands easier to remember.

We keep in mind that the ease of saying a command is always important, but becomes even more important the more often a command is used. In contrast, the ease of remembering a command is always important, but becomes even more important for commands that are not frequently used.

We use one-word commands very sparingly because they are apt to get mixed up with text. Beyond one word, however, we keep the number of words used in any given command to a minimum, because this makes them easier to remember, say and combine.

One tenet of Human-Machine Grammar is there are no synonyms. This makes the vocabulary smaller, makes it easier to remember and predict commands, and makes it possible to combine commands, which speeds everything up.

Another important way to keep vocabulary to a minimum is not using a new word when an existing one will do. You'll hear more about this shortly.

Our words and rules result in consistent word patterns across the entire set of commands.

Here a few specific examples to illustrate some of these rules:

SLIDE 11 — No synonyms

There are no synonyms in the human-machine dictionary.

The word "This", for instance, refers to something that is highlighted or on the clipboard. It is the only word that carries these meanings. If you want to say a command that does a single action to a selection, like **"This Cut"**, or **"This Bold"** you know you'll use this word.

The word "Back" and only back refers to moving something in the direction to the left of the cursor. **"Word Back 3"**, for instance, moves the word nearest the cursor 3 words to the left.

"Forward" and only forward refers to moving something in the direction to the right of the cursor. **"Graph Forward 2",** for instance, moves the paragraph nearest the cursor down two paragraphs.

SLIDE 12 — Conserve words by using more than one meaning: "Top"

Context comes into consideration because it makes it possible to use command words in several different ways. This also helps keep the vocabulary small and easy to remember.

In the human-machine dictionary, like regular dictionaries, the same word can have more than one meaning.

"Top", for instance, refers to the beginning of a document — the command **"Go Top"** puts the cursor at the beginning of a document.

"Top" also refers to the portion of a word, line, paragraph or document that lies before the cursor. For example **"Graph Top"**, selects the portion of a paragraph that is before the cursor and **"Doc Top"** selects from the cursor to the beginning of the document.

SLIDE 13 — Conserve words by using more than one meaning: "Numbers"

Another good example is numbers — numbers can refer to hitting a key a number of times, like **"Backspace 3"** or selecting a number of objects, like **"3 lines"**.

The numbers 1 to 100 also indicates several types of absolute measures. **"Volume 50"**, for instance, adjusts the computer's speaker to its middle volume.

SLIDE 14 — Word Pairs

We use existing word pairs when we can — this takes advantage of our instinctive knowledge that pairs carry related meanings. It also helps makes the vocabulary concise and easy to remember.

Back and forward are a pair. We also frequently use "On" and "Off". For example, **"Speech On"** and **"Speech Off"** turn the microphone on and off. Another common pair is "Before" and "After" — **"5 Before"** moves the cursor 5 words to the left, while **"5 After"** moves the cursor 5 words to the right.

SLIDE 15 — What do you call the word before the cursor?

In a few cases we've had to come up with words not usually needed — for instance, what do you call the word before the cursor? This is something you don't usually have to specify until you find yourself commanding a computer using speech. You could use a phrase, but if you have a single word for it, and if that word is paired nicely with the word for the word following the cursor, it will be easier to remember and use. In these cases we have followed the way people naturally adjust language to fit a situation.

To select three words before the cursor, for instance, you would say **"3 Befores"**, and to select three words after the cursor, **"3 Afters"**.

Using words this way makes for a more concise set of basic commands. Keeping commands concise is an important rule because when you become more advanced you'll find yourself combining commands — doing two, three, or even four things with one spoken command phrase.

SLIDE 16 Combined Commands

This is a huge time-saver — combining a pair of steps into one step equals a 100 percent increase in efficiency. In general, if you don't have to think between steps, there's no reason to have separate steps.

One of the most important rules of Human-Machine Grammar is that the command steps carried out by combined commands always follow the order of events.

This rule cuts out a lot of alternate wording possibilities and establishes a pattern, making it much easier to remember or guess how a command would be worded.

For example, **"3 Lines Bold"** selects, then Bolds the three lines below the cursor, **"3 Graphs Cut"** selects, then cuts the three paragraphs below the cursor.

Another thing we pay attention to is feedback. When you use the mouse to do an action that involves several separate steps, like selecting a paragraph, cutting the paragraph, moving the cursor to another location, then pasting the paragraph, you by default follow exactly what is happening.

We construct our commands to make sure that when you, for instance, select, cut, move and paste text using a single command, you're able to follow the action by seeing the text highlighted in its original location before it is cut, then highlighted after it is pasted in the new location. It's important that this kind of feedback not become annoying, however, so it happens quickly. We also use audio feedback in a few instances.

As you watch the demonstration of Utter Command make sure to notice the types of words, their order, and the resulting patterns. Also be on the lookout for feedback.

Now I'll tell you a little more about Redstart Systems and Utter Command.

SLIDE 17 — Redstart Systems

Redstart Systems is a company of people who use speech recognition — we think the key to making good software is using it.

Utter Command is our first product, and the foundation of a set of speech recognition interface products that will use the human-machine dictionary and grammar rules.

SLIDE 18 — Utter Command

Utter Command provides all you need to control every aspect of a computer using speech. In many cases it allows you to control the computer faster than you can using the keyboard and mouse.

Utter Command is now in alpha testing. We expect that it will be available in the fall. It will initially be sold from the Redstart Systems Web site. We're also setting up a program to train Utter Command trainers.

This initial edition of Utter Command runs in conjunction with NaturallySpeaking Professional, meaning you must have NaturallySpeaking Pro in order to use Utter Command.

The Utter Command manual includes a reference section that documents every command, and a lesson section that takes you through many of the commands. The lesson section includes self-guided tours, or lists of commands that you can say to the computer and learn by watching what happens.

The Human-Machine Grammar is included in the manual and will also be posted on the Redstart Systems Web site. The Human-Machine Grammar dictionary contains all the words and rules used in Utter Command as well as all the words we use internally. The dictionary is an active document, and we consider it one of our most important jobs to keep it updated.

We'd also like to invite anyone who wishes to do so to use the Human-Machine Grammar system of words and rules in writing custom macros. There's a place on the site to ask about wording if you'd like advice on how to word commands based on Human-Machine Grammar. We're also open to suggestions when new words are needed.

SLIDE 19 — Conclusion

So, to sum things up, human-machine linguistics informs Human-Machine Grammar, which underlies Utter Command software. The result is speech that fits the way humans and machines work together.

Now for the Utter Command demo — again, as you watch, be sure to notice related words, word order, and patterns. One last note — all Utter Command commands work globally — meaning they work in all programs.