QCon

Please evaluate my talk via the mobile app!

An Unseen Interface

Creating Speech-driven UI For Your App That Makes Users Happy :D

by Halle Winkler, @politepix http://www.politepix.com

What is a speech-driven UI?

A speech-driven UI uses either speech recognition as an input method, speech synthesis as an information source for the user, or both together.

...but it can also be multi-modal.

How does speech recognition work?

The elements of speech recognition are:

1. An acoustic model
2. A lexicon
3. A language model (probability) or grammar (ruleset for states)
4. A decoder

What kind of apps benefit from speech UIs?

Large Vocabulary Tasks: server, built-in vocabulary (UITextView, Android.speech, Nuance, AT&T, iSpeech)

- Tasks in which free-form dictation is useful
- Tasks which relate specifically to language

Command and control tasks: offline, you generally define vocabulary (OpenEars or other CMU Sphinx or Julius implementations, some Android.speech devices and OSes)

- Interfaces where the user is looking somewhere else
- Interfaces where speech provides a new input or output
- Interfaces that are more fun with speech
- Interfaces where it's easier to speak than type
- Interfaces where it's easier to listen than read
- Interfaces where a heavy obstacle is removed

Why offline?

The interface is always available to your user

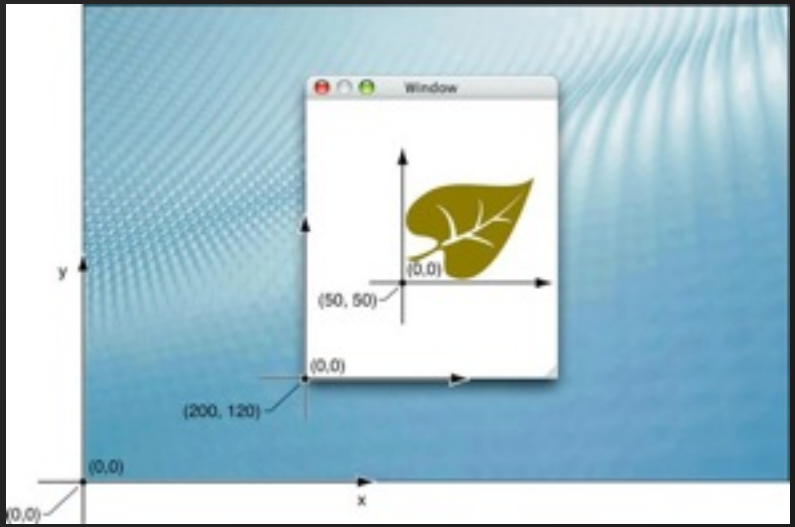
Speed is as fast or faster as a network API – and it's quantifiable!

Interface design and implementation is simpler and more predictable without an asynchronous network dependency

The user is not giving away any of their data

How is a speech UI different from a visual UI?

What are the dimensions on which a visual UI is rendered?



What are the dimensions on which a speech UI is rendered?

A speech UI is rendered on the dimension of time. People value their time exquisitely.

Do people understand each other perfectly all the time? Why not?

Accents
Lack of shared vocabulary/Dialect
Noise
Distractions
Interruptions
Hearing difficulties
Distance
Language errors

Human speech interactions have frequent comprehension faults
Emotional intelligence makes us incredibly fault-tolerant

Automated speech recognition is subject to all the same issues as human speech recognition, but **without the emotional intelligence**

We have to stack the deck in our (users') favor.

Short is good.

Don't bite off more than you can chew – small (read "fast") steps forward means small (read "fast") steps backwards

Use keyword detection to launch events

Switch between small vocabularies that each relate to one domain
This results in accuracy, speed, and a large vocabulary!

Short is bad.

Phonemes are the smallest unit of speech
Words with few of them have a lot of rhymes
Contextless rhyming is our enemy
Medium-sized, crunchy granola words are our friends

My app, my rules

Some apps need to recognize words
or phrases in ways that can be
expressed by rules.

Or be flexible

Some apps need to do probability-based detection
There are probability-based language models for
expressing this such as ARPA models

Out of vocabulary

Your app also has to behave well when people aren't speaking to it!

Mic distance and vocabulary

The more distance, the less vocabulary

Test, test, test.

And obtain appropriate test material.

Case study 1: Recipe App

A natural implementation of offline speech recognition

What are our interface considerations?

- What are we buying with our time? Hands-free operation, moving locus
- Hands-free doesn't mean eyes-free! We can provide visual info
- Operational distance is pretty far
- Instead of NLP, offline grammar
- Secret weapon: we know all the words in a recipe in advance
- Fault tolerance: one level of complexity, don't confirm; return!
- Challenges: noise, moving locus, reflection, competing speech

Case study 2: Marco Polo

A dialog management tag game: one user checks in a single location and the other user receives volume-based speech feedback about their proximity to the target when they say “Marco”

UX Considerations

- What are we buying with our time: play!
- For a single word, language model is fast and sufficient
- Acoustic environment and OOV semi-important
- This is a single-mode interface – an actual dialog manager
- Extra development time should be put into increasing voice dynamic range

Case study 3: TalkCheater

An app to whisper sweet presentation notes in your ear

UX Considerations

- What are we buying? Eye contact, moving locus, enhanced human capabilities
- Is this a speech recognition app?
- Does this have a visual or a touch interface?
- The body is the interface
- Fault tolerance, always important but most important in a high-value scenario
- Volume
- Speaking speed of synthesized speech

Talk to me @politepix and the OpenEars forums. I will tell you all the things.

QCon

Please evaluate my talk via the mobile app!