Wiki-to-Speech for Mobile Presentations

John Graves
AUT University
John.graves@aut.ac.nz

This quality assured paper appeared at the 2nd annual conference of Computing and Information Technology Research and Education New Zealand (CITRENZ2011) incorporating the *24th Annual Conference of the National Advisory Committee on Computing Qualifications*, Rotorua, New Zealand, July 6-8. Samuel Mann and Michael Verhaart (Eds).

Abstract
This paper provides a status update, conceptual framework and demonstration of a novel text-to-speech presentation tool: Wiki-to-Speech. Aiming to leverage a combination of the World Wide Web's hypertext links, Google's key word search capability, Wikipedia's collaboratively created reference material and the emerging technology of "Avatar-quality" text-to-speech, Wiki-to-Speech introduces a system for collaboratively creating voice-over presentations driven by web-based scripts, including responses to post-presentation questions. The tool's open source development is the focus of the research behind the project, but the tool itself may be of value and interest to the field of educational technology, especially given the free availability of cross platform (Mac, Windows, Linux) and mobile (Android) versions.

Keywords
Text-to-speech, mobile learning

Introduction
For Clay Shirky (2010), Wikipedia marks the beginning of an age of global, collaborative knowledge sharing. The numbers are remarkable. Wikipedia's monthly report card[1] shows 400 million unique users access over 15 billion pages per month. While a few educational institutions have created new outlets for academic knowledge via the internet, notably MIT with

[1] http://stats.wikimedia.org/reportcard/

2000 courses on their Open Courseware site[2], the inputs to these systems are not open. Perhaps computing and IT education could benefit from new, collaborative systems for gathering inputs to a collective, on-line store of learning materials modeled on Wikipedia. Considering the impact of one individual dedicated to sharing knowledge on-line highlights the potential here. Salman Khan made 2300 educational YouTube videos which have been viewed over 50 million times[3]. Free, on-demand access evidently has great appeal for learners. *How* to develop such systems is one research question. *Whether* to develop such systems is another question. This paper addresses the subject from a Design Science perspective: *how does the design of such a system evolve as we attempt to build a viable prototype?*

Methodology

Gregor and Jones (2007) identify eight components in the anatomy of a design theory, of which this paper only touches on the first four: purpose, constructs, form and mutability. With continued research and development, the other, more substantive, components should be fleshed out, but we begin here with a skeleton.

Purpose

From a teaching and learning perspective, computing and IT education should be sequenced and better organised than a Wikipedia article or, more to the point, StackOverflow[4] (the "language-independent collaboratively edited question and answer site for programmers"). Collaboration and searchable content alone are just sufficient to create Vygotsky's (1978) zone of proximal development, where instructional content *may* be found which *may* reflect greater knowledge or experience which *may* add value for a learner. Wikipedia and StackOverflow exist because they can sometimes be used to find what is wanted (answers), even though what is found may not be what is actually needed (knowledge).

Knowledge constructs

To add more structure to the flow and order of ideas requires a script, conventionally delivered in the form of a written textbook or a lecture. In an educational setting, a learner is usually expected to engage with the resulting structure, most often by posing and answering questions about it.

Form

So, in simplest terms, if a lesson is a script and learning the lesson involves dialogue, a Design Science approach to creating a computer-based system which matches those basic requirements suggests starting with a scripted dialogue system.

Artifact mutability

This won't solve the whole knowledge transfer problem, but it is one place to start. We can then work at refining the concept and clarifying the design and implementation through a series of iterations.

First iteration: dialogue interface options

Computers will become better teachers as the capabilities for dialogue improve. On the input side, as things stand today, the performance of general purpose

[2] http://ocw.mit.edu/index.htm

[3] http://www.khanacademy.org/

[4] http://stackoverflow.com/

voice recognition systems has been inadequate to displace mouse and keyboard input, except for disabled users. The use of gesture has increased with multi-touch devices, displacing mouse input (while arguably degrading keyboard input). Possibilities for touchless interaction are being developed, such as Microsoft Kinect[5]. Thus, it is not difficult to imagine future learning systems which may watch and listen and respond to touch all at once. On the output side, the recording and playback of voice has been optimized in the Speex codec[6] and vendors now offer a wide variety of high quality text-to-speech products in languages from Arabic to Turkish. The net result of this first exploration of the current solution space suggests a multi-lingual text-to-speech, mouse or touch-based dialogue system.

Second iteration: scripting options

Scripting a dialogue should be as simple as possible. XML and HTML are, by definition, markup languages which clutter the words of a script with surrounding markup tags. AIML[7], the Artificial Intelligence Markup Language, which was designed to produce a form of computer-based dialogue, provides a system for template-based responses to matched patterns, as in:

```
<category>
 <pattern>WHAT ARE YOU</pattern>
 <template>
  <think>
   <set name="topic">Me</set>
  </think>
  I am the latest result in artificial intelligence, which can reproduce the capabilities of the human brain with greater speed and accuracy.
 </template>
</category>
```

Interaction with an AIML-based chatbot is typically shallow (see the Chatterbox Challenge[8] for examples). Contributors to Wikipedia write in Wikitext[9], "a simplified alternative/intermediate to HTML." Once the difficulty of writing directly in HTML has been accepted and the utility of authoring script content in a simplified alternative format is examined, the minimum syntax required appears to be a separation/distinction between statements and choices on the one hand and answers and responses on the other hand, yielding a structure described by this schematic (where the Answers and Responses are optional elements):

```
Question/Statement
Answer 1 <separator><action> Response 1
Answer 2 <separator><action> Response 2
...
<blank line>
```

Thus, with optional answers and responses in this Question/Answer/Response (Q/A/R) structure, a monologue can be scripted as a series of statements; a multiple choice question can be created by placing a

[5] http://www.xbox.com/en-NZ/kinect

[6] http://www.speex.org/

[7] http://www.alicebot.org/aiml.html

[8] http://www.chatterboxchallenge.com/

[9] http://en.wikipedia.org/wiki/Wiki_markup

<separator> after each choice; and a *responsive* multiple choice question can be created by adding a response after each <separator>, as in the following example (where a single semi-colon is the <separator> and a second semi-colon signals advance-to-next):

```
Sun Microsystems released Java in: 1995 ;; Yes. James Gosling developed it. 2007 ; No. In 2007 Java was relicensed under the GNU General Public License.
```

Again, the idea here is to try to minimize markup, leaving the words of the script as unfettered as possible (compare the GIFT microformat[10]).

A scripted, responsive choice question offers an antidote to the branching bush of choices encountered while surfing web pages. The continual introduction of new hypertext options and choices can make it difficult to learn from Wikipedia without getting distracted. Wolfram (2002) points out how a small rule change in a recursive program can yield dramatically different results. In this case, by introducing choices which *don't* go anywhere, instead of endlessly branching hypertext choices, we get pathways through the correct responses. "Wrong" answers become opportunities to make mistakes and learn without penalty. In any case, these ideas motivate this second iteration.

Third iteration: dealing with questions

A one-way (author-to-audience), asynchronous presentation tool is improved by adding an ability to respond to anticipated questions. The answers must already be provided by the script, so the task for the pattern matching rule is to route the flow of dialogue to the appropriate point in the sequence. This is very similar to the function of an index in the back of a book. To achieve this, the script syntax is extended to include:

```
[input];
```

Labels for points in the script:

```
[Part 1: History of Java]
```

And the definition of rules:

```
[[Part 1: History of Java]]
example="[When] was Java created?"
```

The pattern matching approach seen in AIML can be applied here. Any question containing the word "when" would be routed to the section labelled [Part 1: History of Java].

Once an input box appears in the interface, other new opportunities present themselves including capturing of input values to use to fill slots in a template, commands (such as opening a new script, quitting) and search.

Templates work with in-line script variables, such as:

```
Hello ${first_name}
```

With question rules for filling in the values:

[10] http://microformats.org/wiki/gift

```
[_ask]
[[_first_name]]
reply="Sorry, what is your name?"
```

Given an input, search naturally arises as an alternative to the state of all-the-rules-have-failed-so-now-what. Rather than give up, it makes sense to try matching the input first in the labels (so an input of "history" or "java" will land at the start of the sequence labeled [Part 1: History of Java]) and proceed to full-text search only if the label search also fails.

With these additions, the system begins to offer considerable flexibility in what it can say and how it can respond verbally. With just text on the screen, however, it is still not much to look at and gives the wrong impression. Users can see and read text for themselves. When that text is spoken by the computer, they immediately see the benefit for blind people, but not for themselves.

Fourth iteration: voice-over
With the text-to-speech engine running in one application, the user's web browser can be opened to a particular web page. This is an interesting development because it means whatever the web has to offer, it can now be discussed and explained using a voice-over, before, during or after access to the website. Both the web address and the voice-over are part of the script:

```
Here is the official website for
[http://python.org/ Python];;

Note that links for downloads and
documentation are on the left.
```

Fifth iteration: collaborative web-based authoring
Creating a text-to-speech script requires only a simple text editor, so the capabilities of web-based wiki, blog and etherpad[11] systems can be enlisted to allow for the authoring to take place collaboratively on-line. The software reading the script from a web page, now christened Wiki-to-Speech, has to parse away the surrounding HTML, but this is facilitated by adopting a convention that scripts are tagged as preformatted text:

```
<pre>
This is a Wiki-to-Speech script.
</pre>
```

Sixth iteration: web-browser-only viewing
Having to download a separate application is an obstacle to expanded use. Once a script has been authored, the text-to-speech can be pre-rendered and served up as audio files.

Seventh iteration: PowerPoint/OpenOffice authoring
If audio can be served up, so can images. This iteration explores the potential of a presentation software-based workflow for creating scripts. By using the Speaker Notes feature of OpenOffice (and the fact that PowerPoint can save to the OpenOffice XML-based format), a script-building tool is developed to extract the speaker notes from each slide and place them into a Wiki-to-Speech script file (plain .txt). That script then runs through the text-to-speech engine to pre-render audio files, one per slide. When these pieces line up

[11] For example, http://ietherpad.com/

with image files generated from the PowerPoint/OpenOffice slides, the result is a presentation which "delivers itself."

Eighth iteration: Dropbox as a web server

Dropbox[12] is an on-line file sharing system. Files in the Public folder are just that: public. Right click on one and Dropbox provides a public link. Given the link to the first slide of a Wiki-to-Speech presentation, any web browser capable of playing sound files can fetch and display each slide with the corresponding voice-over. All that is required to publish a talk is to save it in that Public folder (on the local system; it is automatically synced by Dropbox with the cloud server and from there to the world).

Ninth iteration: going mobile

Currently, the Webkit browsers on mobile devices may not play audio embedded in HTML files. On the other hand, the latest Android devices (2.2 and above) offer up a choice of high quality text-to-speech voices for on-the-fly conversion of a Wiki-to-Speech script into spoken audio output. Relative to video, this is a very low bandwidth solution for delivering multimedia to a mobile device.

Conclusion

This paper explored one way in which currently available technology could be used to build toward a Wikipedia-like resource of on-line learning materials. The resulting prototype system reveals how some rather remarkable capabilities are now freely available to educators. For example, text entered into an etherpad can be fetched and spoken by a mobile smart phone. Many classroom presentations begin and end as a slideshow seen only by the students in the room. By combining Wiki-to-Speech[13] with Dropbox or other freely available on-line resources, such classroom presentations could be seen and heard by anyone with a web browser or an internet-enabled mobile device. From a theoretical perspective, further research is needed to find testable propositions, justificatory knowledge, and principles of implementation. Meanwhile, the current research provides an expository instantiation: a prototype artifact for on-going experimentation.

Acknowledgements

This paper would not have been possible without the encouragement of my faculty advisors for which I am most grateful.

References and Citations

Gregor, S. and Jones, D. (2007) The anatomy of a design theory. *Journal of the Association for Information System 8(5), 312-335.*

Shirky, C. (2010) Cognitive surplus: creativity and generosity in a connected age: Penguin Press.

Vygotsky, L.S. (1978) Mind and society: the development of higher psychological processes. Cambridge, MA: Harvard University Press.

Wolfram, S. (2002) A new kind of science: Wolfram Media.

[12] http://www.dropbox.com

[13] http://wikitospeech.org