**Project Factor**

**Searchable Speech to Text Framework**

School of Applied Technology

Industry Associate Professor

Jeremy Hajek

**Abstract**

This project, Factor, developed an elastic-cloud based framework for parsing text from standard video codecs using existing opensource tools. The purpose was to generate text transcripts of video that can be marked up by users and exported to different digital/print formats--even the printing of custom made books. This project goal was designed to give libraries control over data and information once again and make them the central repository of knowledge-print and digital.

**Introduction and background of project**  
  
 Compared to Engineering or Literature, Information Technology as a discreet field is comparatively young. A very young field and much like an infant, textbook data and the information covered by IT is rapidly changing-evolving As we speak textbooks from even a few years ago are of only historical or novel use. Libraries have the option of investing multiple thousands of dollars in new IT textbook collections each year, but are hesitant to do so knowing the quick changes in IT/technology. The question and the driving force for this project is, “If all the new IT data is not in textbook form, then where is it?” Our realization was that it is contained in video. It is in lectures, conference presentations, YouTube and how-to videos spread across universities and the internet.

The Factor project proposes to create a framework for taking the text transcripts out of video and presenting them to the user for their custom use. Giving them access to complete text transcripts opens a world of possibilities. With outcomes such as custom digital and print works, subtitles, transcripts for ADA, as well as archival and searchable options. Imagine what you could do if you had the text transcripts of an entire department for an entire semester?

**Project Details**

Factor seeks to be a framework and provides an end to end solution. There are 6 components of the projects.

1. **Project Gallium -** 
   1. This is the application that uses our Speech to text engine CMU Sphinx. This is a Carnage Mellon opensource speech to text engine project. No need to reinvent the wheel but to build on top of their success. In our tests worked very well for simple spoken English (See documentation below). Accuracy increased when source input was of higher quality (meaning better quality microphones, high quality sample rate capture, and slower speaking) Through the translation worked well – we ultimately decided to add a human element to the speech-to-text conversion.
2. **Project Window Paine**
   1. This part is the companion piece to the first project. Inspired by a presentation I saw at Slate 2012 from two IT staff members from Southwest Missouri Baptist College. They were making use of Google’s Closed Captioning services for this same purpose. As this has advantages there are disadvantages as well. Window Paine was designed as an HMTL5 video player (that way it plays on all laptops, tablets, and mobile) that syncs the machine rendered transcript with the timestamp of the video. Asynchronously allowing multiple users to “collaborate” and fix the transcript as they watch the source video. Time did not permit but we have a plan for a “gamification” system (ala stackexchange.com) as well to award points, reputation, and privileges to those who participate most in the editing and updating of the transcript.
3. **Project SH-3**
   1. This project handled the portion of the project dealing with data storage and retrieval. Seeing as the nature of this project as dealing with large amounts of text, timestamps, and document revisions a standard Relational database was deemed not to be the best fit. Something flexible such as a document store (noSQL) such as CouchDB which uses noSQL (HTTP requests via javascript and AJAX – in place of SQL) can be clustered or another idea researched was an XML store or XML database such as eXist DB or BaseX DB using xPath and xQuery on an xml document – which would make sense as our data is primarily XML.
4. **Project Transformer**
   1. Project uses SH-3 and allows for a user to take the corrected transcript and “mark” it up using a simple drag and drop application. Using the DocBook 5 XML standard that allows for a standardized method of creating books in XML format. The transcripts that come from project Gallium (the Speech to text engine project 1) does not give us any context (paragraphs, pages, subject titles, headers, etc. etc.) Project Transformer allows a user to mark up the text and give it context they want so that the transcript can be used in other formats such as HTML, PDF, or ePub (used on e-readers). These documents could even be stored in a virtual library or printed on demand, shared with others, or printed in a library.
5. **Project Ponce De Leon**
   1. The final portion of the application would be to use an elastic search engine such a Lucene and Solr. The idea would we to integrate a “Google” style search across all the transcripts you have collected and allow the user to either jump directly to the subject matter and video via the timestamp or “collect” these custom snippets of text and mark them up in Project Transformer as a “custom” book
6. **Project Luna**
   1. Elastic rendering engine built on top of the private cloud software Eucalyptus. This project manages the metadata of “video” source submission. It also controls the launching of worker instances to handle Speech to text conversion jobs in a linear scale—not exponential – thereby saving cost by reducing the need for standing hardware.