論文 / 著書情報
Article / Book Information

Title	Speech processing tools - An introduction to interoperability
Authors	Christoph Draxler, Toomas Altosaar, Sadaoki Furui, Mark Liberman, Peter Wittenburg
Citation	Proc. INTERSPEECH 2011, , , pp. 3229-3232,
Pub. date	2011, 9
Copyright	(c) 2011 International Speech Communication Association, ISCA
DOI	http://dx.doi.org/

Speech Processing Tools – An Introduction to Interoperability

Christoph Draxler[1], *Toomas Altosaar*[2], *Sadaoki Furui*[3], *Mark Liberman*[4], *Peter Wittenburg*[5]

[1]Institute of Phonetics and Speech Processing, LMU Munich, Germany
[2]Aalto University School of Science and Technology, Espoo, Finland
[3]Dept. of Computer Science, Tokyo Institute of Technology, Japan
[4]Dept. of Linguistics, University of Pennsylvania, Philadelphia PA, USA
[5]Max Planck Institute for Psycholinguistics, Nijmegen, The Netherlands

draxler@phonetik.uni-muenchen.de, Toomas.Altosaar@hut.fi, furui@cs.titech.ac.jp, myl@cis.upenn.edu, Peter.Wittenburg@mpi.nl

Abstract

Research and development in the field of spoken language depends critically on the existence of software tools. A large range of excellent tools have been developed and are widely used today. Most tools were developed by individuals who recognized the need for a given tool, had the necessary conceptual and programming skills, and were deeply rooted in the application field, namely spoken language.

Excellent tools are a prerequisite to research. However, tool developers rarely receive academic recognition for their efforts. Journals, conferences and funding agencies are interested in the results of the work on a research question while the tools developed to achieve these results are of less interest. This makes it difficult to publish articles on tools, and it is next to impossible to obtain funding for their development.

The Interspeech 2011 special event on speech processing tools aims to provide a forum for tool developers to improve their academic visibility and thus enhance their motivation to continue developing the software needed by the community.

This year, the focus is on *interoperability* – how can different tools be integrated into the workflow, data be exchanged between tools, queries work across tools, and a consistent user interface be achieved.

1. Introduction

Historically, the development of software tools for speech processing can be divided into three phases. In the pre-PC era, processing power, memory and network bandwidth were limited, and only very few institutions could afford computers. Furthermore, only trained operators could use the software which in general was highly specialized. In this era, the mainframe was the dominant system architecture and software was often designed to reuse existing modules and to share data since resources were limited.

The PC era saw an exponential increase in processing power and available memory, and a high degree of standardization for hardware, programming languages and graphical user interfaces. A typical desktop computer allowed processing hitherto unknown amounts of data while standardization resulted in software being developed for large audiences. Researchers, technology developers and the general audience now had access to sophisticated software tools for processing speech, ranging from simple audio recording to annotation editors, signal processing, speech recognition and speech synthesis systems. As a side effect, the "old" ideals of tools sharing resources and data became less and less important.

In the current Internet era, corpus and data sizes have outgrown the capabilities of individual workstations.

Resources are now increasingly made available on the web, and web services provide the necessary processing power. This will create a fundamental change in the way research is performed: researchers will expect to have their data, as well as their favorite working environment and tools, accessible in a virtual workplace in a browser.

During the transition of the pre-PC era to the post-PC era, several software systems were developed for operating on discrete-time signals - and more specifically - on speech. They represent real cases where software development for signals and speech was funded and reported through Ph.D. theses, reports, and deliverables that nearly every speech researcher is aware of, has used, or is still even using today.

Today's speech researchers may not be aware that early software tools played an integral role in speech research with far reaching effects. For example, the TIMIT corpus was processed and annotated on a software system called SPIRE [1] that ran on Symbolics Lisp machines and was freely available to universities via a license. SPIRE itself was influenced by earlier work by Gary Kopec who formulated a representation for discrete-time signals in software [2]. Kopec extended his theoretical work to cover both signal representation and processing, as well as a system called SDB [3] that was used for the storage and management of signals and their properties. This latter work was used especially for speech processing research and applications. Funded work on other speech-specific software tools also existed, e.g., QuickSig [4] that was applied to speech corpus annotation and speech processing, as well as StanisLAX that noted the need for advanced tools and architectures for speech processing environments [5].

1.1. Workflow

The typical phases of an end-to-end workflow in speech processing, along with the associated data types and a selection of well-known tools are shown in fig. 1.

Clearly, proceeding through the workflow involves the use of many different tools, but even more it requires handling different types of data: signal data (audio, video, sensor), graphics data, program code and plain, mark-up or styled text.

Presently, every tool in the workflow uses its own native data format. Some tools provide import- and export functions,

28 – 31 August 2011, Florence, Italy

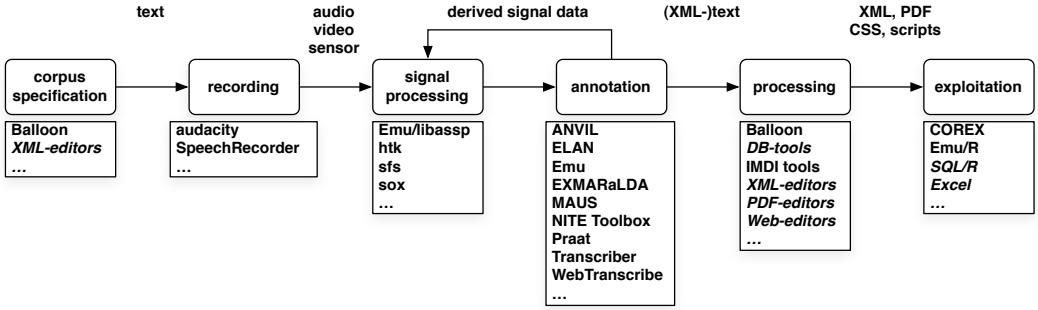

Figure 1: *Speech processing workflow and a selection of tools*

but significant amounts of data conversion and reformatting is still necessary. A global data model for the speech processing workflow, based on the concept of a corpus, has been proposed by [6]. Such a data model will be immediately useful for web-based and collaborative work, but transformation costs, both in terms of converting data and in terms of adapting software to access data in the global data model, are high.

1.2. Tools workshops and conferences

In recent years there have been a number of initiatives to disseminate knowledge about existing tools, to share development efforts, and to bring together researchers and tool developers to define the requirements for future tools. These initiatives include, e.g., a workshop on standards for phonological corpora organized by U. Gut in Augsburg in 2009 and the workshop on 'New Tools and Methods for Very-Large-Scale Phonetics Research' organized by the Phonetics Laboratory of UPenn in 2011, and others, e.g., the European CLARIN project [7].

As a result, informal cooperation between tool developers and discussion groups have been established. For example, the effort of the developers of ELAN, ANVIL and EXMARaLDA to define and implement a common data exchange format [8], the language archive technology (LAT) discussion groups hosted by the Max Planck Institute of Psycholinguistics in Nijmegen, the newsletters and discussion forums of LDC, ELRA and other speech resource providers, now exist.

However, contributions to these initiatives do not result in publications, let alone high-impact publications. In the last 10 years, only one issue of 'Speech Communication' has been devoted to tools (Vol. 22, Numbers 1-2, 2001). The Romans knew what this means: *Quod non est in actis non est in mundo*.

On the other hand, tools are mentioned in many publications. Table 1 summarizes the results of a simple Google search for a given tool in the body of web-accessible PDF documents of large speech-related conferences[1]:

1.3. Interoperability

Encyclopedia Britannica defines interoperability in the context of computer science as *the ability of the user of one member of a group of disparate systems (all having the same functionality) to work with any of the systems of the group with equal ease*.

The English Wikipedia features several definitions of *interoperability*, ranging from very general to domain-specific, e.g.,

[1]The Google search expression was: 'allintext: "SpeechRecorder" LREC OR Interspeech OR ICASSP OR Eurospeech OR ICSLP site:.org filetype:pdf' query date was 23 March 2011.

Table 1: *Google query hits for tool names in the PDF documents of major speech-related conferences*

name	type	count
htk	speech recognition toolbox	2,760
Praat	annotation editor and processor	1,100
Sphinx	speech recognition toolbox	882
NXT	corpus toolbox	343
ELAN	annotation editor	222
ESPS	signal processing toolbox	230
ANVIL	annotation editor	226
Emu	annotation editor and processor	201
WaveSurfer	signal processor and viewer	151
Audacity	signal editor	64
EXMARaLDA	annotation editor	44
MAUS	forced aligner	41
SpeechRecorder	audio recording tool	10

software interoperability. In the context of speech processing tools, the following definition from the IEEE Glossary is useful: interoperability is *the ability of two or more systems or components to exchange information and to use the information that has been exchanged*. The article distinguishes *syntactic* from *semantic* interoperability and has a paragraph on *software interoperability*. The article also emphasizes the importance of standardization.

2. Standardization

With large amounts of text becoming available on the Internet, and with the growing number of well-defined speech and text corpora, standardization has received increased attention. Several standards bodies have taken up the task of standardizing terminology, methods and tools for speech and text processing. These bodies include institutions such as the International Standards Organisation (ISO), the World Wide Web Consortium (W3C) and national standards bodies, interest groups such as the Text Encoding Initiative (TEI) and the ISOCat data category registry, and industry consortia such as the MPEG forum.

3. Semantic interoperability

Among spoken language researchers and technology developers there is now a general consensus about the entities involved in speech processing.

Speech is a *time-dependent physical signal*, it is processed

using digital signal processing methods and algorithms, and it is described (transcribed, segmented, annotated, etc.) using *textual* representations. A *speech corpus* is a structured collection of speech, possibly derived signal data, and textual data [9].

A *segment* is a tuple consisting of an anchor, an optional extension, and a label. In time-dependent segments, the anchor is a time-point, the extension is a duration. In time-independent segments the anchor is a unique identifier. The label describes the segment contents, commonly in free-form or structured text.

A *tier* holds segments of a common type. Within a tier, segments are ordered, e.g., according to the time points or a sequence index. Constraints govern the organization of segments in a tier: segments may overlap, there may be gaps, etc.

For a given collection of speech and text data, there may exist many tiers. In general, tiers relate to each other by some relationship which can be named and which specifies quantitative constraints.

The *schema* specifies the tiers and the relationships between the tiers. A schema not only allows checking the formal integrity of corpus data, but it supports semantically meaningful cross-tier queries and may also be used to build flexible query and visualization tools.

The Annotation Graph formalism provides a formal framework for speech processing, notably the annotation of speech [10]. However, it lacks a schema description. Many speech processing tools implicitly implement a schema in which the tiers are independent of each other. Other tools have implicit or explicit 1:n relationships, only Emu has explicit 1:n and n:m relationships [11].

3.1. Data model overview

A data model is a formal framework describing a collection of data. The *hierarchical* data model organizes data in 1:n relationships: there is a single root element, only one path from the root to any given element, and the relationships are not named. The *network* data model allows more than one root element and named 1:n relationships; different paths are distinguished by name. The query languages in both data models are navigational, i.e., searching for a specific element is achieved by traversing the hierarchy or network.

The *relational* data model organizes data in relational tables with attributes, relationships are expressed via the relational join operator based on comparisons between attributes of relational tables; in principle, relationships may exist between any two relational tables. The query language is declarative, i.e., queries are formulated using relational algebra or calculus, and the concept of navigation is alien to the relational model.

The *object-oriented* data model distinguishes two basic types of relationship: *is-a* and *uses*. The is-a-relationship organizes all data in a hierarchy (*single inheritance*) or a network (*multiple inheritance*), the uses-relationship is named and may link arbitrary elements. Object-oriented query languages are navigational.

Mathematically, the above data models can be seen as *graphs* with different directionality, quantity and naming constraints.

3.2. Data models for speech processing

The data types relevant for speech processing are highly structured: data streams, sequences, trees and network structures are common.

Generally speaking, the relational data model is an abstract representation of data; any structure can be represented, but this inflates the number of relational tables and results in lengthy queries. However, due to the simplicity of the data model and the small number of basic data types, relational models of speech corpora will always be very similar, facilitating the exchange of data and the reuse of code.

The object-oriented data model is a natural choice for representing the richness of speech processing data. However, due to the unlimited possibilities of constructing complex data types out of other data types, no two object-oriented models of speech corpora will look the same, and hence the exchange of data and the reuse of code is heavily affected.

4. Syntactic interoperability

Syntactic interoperability must be divided into at least four levels: the basic encoding level, the data format level, the implementation level, and the programming interface or service level.

4.1. Encoding level

The encoding level comprises the digital representation of the basic data types: signal data and text data. Many different audio, video and sensor data formats have been defined, they are often tool- or platform-specific. For uncompressed data and open data formats, this does not pose a problem – in general, data can be converted without loss from one format to another.

For text data, Unicode is becoming the de facto standard character set. However, the transition from older character sets and encodings to Unicode is still in progress. Unfortunately, some tools do not yet support Unicode.

Although interoperability on the encoding level is feasible – in general, doing a round trip is possible – it often requires a lot of manual work and can be quite time-consuming.

4.2. Data format level

The data format level comprises the organisation of the data as it is being used by an application or a web service; it is what is commonly known as an application dependent file format, e.g., a Praat TextGrid or an ELAN annotation file. Only a minority of the file formats in use by today's tools are formally defined with the specifications being publicly available.

Some tools require files to be in a given storage location, others do not. None of the common tools stores its data in a DBMS or has an interface to a database system.

Since many different file formats have become de facto standards most tools provide import and export functionality. However, most work on individual files or directories but cannot process entire corpora. Converting from one file format to another normally leads to loss of information, and in general, round-tripping is not possible.

4.3. Implementation level

The implementation level relates to the programming languages and libraries used to implement a given speech processing application. Naturally, technology advances, applications become available or software becomes obsolete. Interoperability is a moving target.

Currently, the programming languages most commonly used for the implementation of speech processing tools are strongly typed and compiled languages like Java, C++ and C, and interpreted script languages like perl, tcl and python, as well as JavaScript. Besides these, application or domain specific languages are used, e.g., the built-in script language of Matlab, the

statistics language R, or the database query language SQL.

Every language comes with its own syntax, execution model, basic data types and with its specific programming libraries. Some languages provide features to include precompiled modules, thereby allowing the reuse of existing code.

4.4. Programming interface or service level

Many program libraries are accessible via public interfaces. Wrapper classes take care of the conversions necessary on the data format level and thus enable the exchange of data. Such interfaces are common in high level applications, e.g., spreadsheet programs and statistics packages.

Carrying the concept of client-server computing one step further is *cloud computing*. Here, a user establishes a virtual workspace, where he or she has access to all the required data and tools. The tools are implemented as web services, and they are run by some service provider somewhere on the net.

For basic tasks such as text processing or standardized workflows such as web link management or financial services, and especially social networks, cloud computing has become very popular and successful. For language research, especially text corpus work, virtual workplaces have started to appear [12]. For speech processing, work on web services and the infrastructure necessary to establish virtual workspaces has begun [13, 7].

5. Users' view on interoperability

Interoperability has another facet, not covered by the definitions given above: *the users' view*. A user works with a given software tool for many reasons: suitability, availability, joy of use, familiarity, etc. and by using a tool he or she wants to get work done. In an ideal world, the tool should reflect the users' knowledge, skills and best practices in the application domain, i.e., support the user by using the data, algorithms, workflow and terminology of the field. In reality, things are often the other way around: tools impose a specific world view upon the user, so that switching from one tool to the other means moving from one mini-environment to the next.

On an abstract level, speech processing can be seen as searching for data according to some conditions and then manipulating this data. Query languages range from very general to highly application- or tool-specific, and they can be written as text, constructed using a wizard, or by interactive exploration of the data itself. In fact, currently every tool implements its own query language, and often this query language directly reflects the data structures of the tool.

An alternative is to separate the surface form of the query from the underlying query evaluation mechanism. This underlying mechanism reflects the global domain data model and the formal properties of this model, e.g., relational tables or graphs, whereas the surface form is determined by the user interface which is tool or task specific.

This approach encourages tool developers to provide more than one user interface for queries, or to develop application independent query interfaces that can access different query evaluation mechanisms. Furthermore, this approach allows the user to select the query input method that best suits his or her needs, and even to switch between input modes.

6. Conclusion

The field of speech and language processing is alive and productive. There is a general consensus about the application domain; formal frameworks have been proposed which underlie and guide the development of tools and systems. Excellent tools have been developed and are widely available – in fact, the large number of tools mirrors the richness of the field.

Interoperability is a key issue. This may be a bold statement, but semantic interoperability is not the problem, it's the data format level interoperability in combination with disagreement on the extent of a global data model. Thus, a common minimal core data model encompassing the concept of a corpus is needed, and this data model must be supported by all tools – [6, 8] are first examples of how this ambitious goal can be reached. In the future we need to go even further.

7. References

[1] R. H. Kassel, "A users guide to SPIRE," MIT Research Laboratory of Electronics, Cambridge, MA, Tech. Rep., 1986.

[2] G. Kopec, "The representation of discrete-time signals and systems in programs," Ph.D. dissertation, MIT Ph. D. thesis, Cambridge, MA, 1980.

[3] ——, "The signal database system SDB," in *Proc. of ICASSP*, San Diego, CA, 1984.

[4] M. Karjalainen, T. Altosaar, P. Alku, L. Lehtinen, and S. Helle, "Speech processing in the object-oriented DSP environment QuickSig," in *Proc. of Eurospeech*, Paris, 1989, pp. 1450–1453.

[5] E. Junqua and H. Wakita, "StanisLAX: an artificial laboratory for the development of cooperative speech processing algorithms," *Engineering Applications of Artificial Intelligence*, vol. Volume 4, pp. 23–33, 1991.

[6] C. Draxler and K. Jänsch, "Wikispeech – a content management system for speech databases," in *Proc. of Interspeech*, Brisbane, 2008, pp. 1646–1649.

[7] "CLARIN Common Language Resources and Technology Infrastructure: www.clarin.eu."

[8] T. Schmidt, S. Duncan, O. Ehmer, J. Hoyt, M. Kipp, D. Loehr, M. Magnusson, T. Rose, and H. Sloetjes, "An exchange format for multimodal annotations," in *Multimodal Corpora*, ser. Lecture Notes in Computer Science, vol. 5509. Springer Verlag, 2009, pp. 207–221.

[9] C. Draxler, *Korpusbasierte Sprachverarbeitung - eine Einführung*, ser. narr Studienbücher. Tübingen: Gunter Narr Verlag, 2008.

[10] S. Bird and M. Liberman, "A Formal Framework for Linguistic Annotation," *Speech Communication*, vol. 33, no. 1,2, pp. 23–60, 2001.

[11] S. Cassidy and J. Harrington, "Multi-level annotation in the emu speech database management system," *Speech Communication*, no. 33, pp. 61–77, 2001.

[12] R. Gleim, P. Warner, and A. Mehler, "eHumanities Desktop - an architecture for flexible annotation in iconographic research," in *Proc. of Intl. Conference on Web Information Systems and Technologies (WEBIST'10)*, Valencia, 2010.

[13] P. Nguyen, "Techware: Speech recognition software and resources on the web," *IEEE Signal Processing Magazine*, vol. 26, no. 3, pp. 102–105, 2009.