

Issue Statement
Steven Abney

If the relevant concern is the scientific study of language, as opposed to technological applications of that study, then the most pressing agenda item by far is the creation of a universal linguistic database, comparable to the Human Genome Project. We require (ideally) complete digital documentation for (ideally) every human language. The best form for language documentation is an open question, but it encompasses at least a large corpus of naturally occurring text (minimally a million words, or the entire corpus in the case of extinct languages), morphological and syntactic annotation of that corpus, grammar, and lexicon. Grammar and lexicon should be completely interlinked with the corpus. Every word of the corpus should have a pointer into the lexicon, and every syntactic annotation should have a pointer into the grammar, and instead of the usual example sentences within the grammar and lexicon, there should be pointers into the corpus.

In constructing such a universal database, the first priority is obviously endangered languages. There are at least two sorts of endangerment: languages that are not being learned by the next generation, and languages that are already extinct, but whose extant documentation is not widely available and in danger of loss.

A first, small but essential step is the creation of a universal viewer/editor that handles any language and runs on any platform. One would have thought that many such editors had already been implemented, but if so, I have been unable to find them. The closest things that exist are tools that SIL has produced. However, they do not run on multiple platforms; they generally run only under Microsoft Windows. And though they are designed to be extensible to any language, they do not appear to be universal in the sense of working out of the box with any language.

The only practical way to achieve platform-independence is by using Java. The only practical way to achieve universality at the character level is Unicode. For universality in rendering, the tool cannot depend on fonts in the environment (which vary widely from user to user and almost never cover all of Unicode), but must be distributed with its own fonts. Java fortunately uses Unicode natively, but the higher-level text interfaces do not work correctly with TrueType fonts that are generally available for coverage of more unusual scripts. A tool can however be constructed that uses the lower-level interfaces.

On the input side, special keyboards and input methods are appropriate for people working with single languages, but for a universal editor, there also needs to be a universal Romanization using seven-bit ASCII.