

WeSay, A Tool for Collaborating on Dictionaries with Non-Linguists

From Payap Language Software

Reviewed by Ross Perlin, *Himalayan Languages Project & University of Bern*

1. OVERVIEW. WeSay is a free, open-source software program for Windows and Linux, co-produced by Payap Language Software (based at Payap University in Chiang Mai, Thailand), SIL Papua New Guinea, and SIL International.¹ The program envisions members of two major user groups collaborating on a dictionary through WeSay—*advisors*, typically linguists operating with a fair amount of computer experience and decent equipment; and *users*, native speaker community members who have at least some access to and comfort level with a computer. For a variety of reasons, it may not always be possible to assemble such user-advisor pairs, but the scenario is increasingly common as speakers of less common languages gain access to and become more familiar with information technology, and as more collaborative approaches to language documentation become popular. Judging by the projects listed on Language Depot, WeSay’s online data repository, dictionary projects around the world now appear to be making use of WeSay.

The creators of WeSay have stressed the software’s simplicity, its task-based approach, the low training load needed, its potential for community participation, and its ability to run on “inexpensive, rugged, low-power hardware” (such as the computers distributed by the One Laptop Per Child program). On all accounts, they have done an impressive job and created something rare: a piece of language documentation designed to a significant degree around the needs of community members (or others just beginning in the field of language documentation). At the same time, advisors still play a critical role—setting up and maintaining projects is not entirely straightforward, and the ultimate added value from WeSay is likely to come from its use in conjunction with a Fieldworks Language Explorer (FLEX) dictionary project. The advisor’s role is to handle all of the “back end” administrative and technical details so that users can focus on the language work. WeSay is not—at least at the time of this writing—a tool for communities to create their own dictionaries, although it points to the kind of web-based, mobile solutions that might make this possible.

2. GETTING STARTED. After running the WeSay installer on the advisor’s computer, the first step is to create and configure a new project, either from scratch or from a FLEX lexicon, imported via a Lexicon Interchange FormAT (LIFT) file.² For online collaboration—and this is WeSay at its most useful—the advisor will then need to set up a new project on the Language Depot and “push” the lexicon to that site: this is the location in “the cloud,” administered by the WeSay developers, where collaboration on the project is

¹ http://www.wesay.org/wiki/Main_Page; see also Albright & Hatton 2008.

² <http://www.wesay.org/wiki/LIFT>

effectively stored. Installing WeSay and downloading the particular dictionary project onto the computers of native speaker collaborators is a process best completed in person with each computer's user. WeSay also requires .Net Framework 3.5 service pack 1 (which in turn requires Microsoft Installer 3.1)—although both are free, these take up considerably more disk space than WeSay itself.

WeSay has a “front end” where users and advisors can work on specific tasks like word gathering or adding example sentences, and a “back end,” a configuration tool where the advisor can configure settings and tasks, as well as see what changes to the dictionary have been synced by different users.

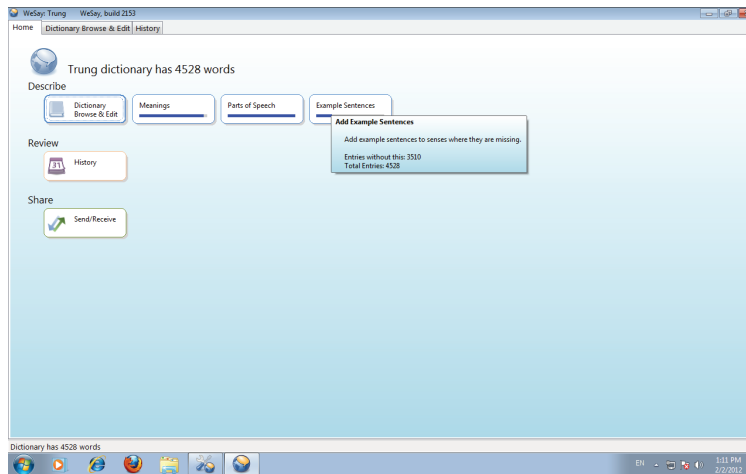


FIGURE 1. An example of WeSay’s “front end,” the home page users will see.

This “version control” capability used in WeSay, called Chorus, is one of its most important features, enabling a geographically scattered group of users to collaborate at the same time without reduplicating each others’ work. For the system to work, all collaborators have to click the “Send/Receive” button fairly often, at a minimum before beginning and after finishing a few hours of work. A division of labor would probably still come in handy. One of the more critical, yet opaque, features is how WeSay handles “conflicts” between different edits with its “best guess” algorithm. Of course, the advisor can always resolve the conflict manually, but this can be challenging as WeSay’s “merge conflict notes” are not always clear.

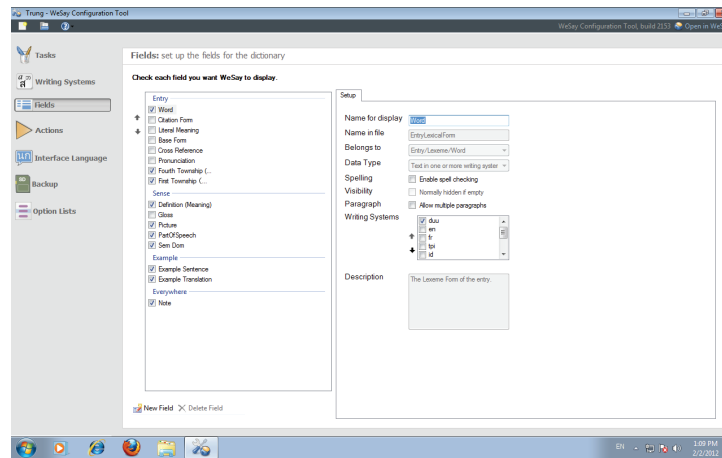


FIGURE 4. A look at fields the advisor can configure in the “back end.”

In principle, the WeSay interface is available in some 18 languages besides English, and is easily translated into others, although many individual terms seem to have been left untranslated and may pose difficulties for (or merely confuse) those users without an English background. It is worth noting that, despite the best efforts of the developers to accommodate users with little computer knowledge, WeSay does rely on some intuitive computing conventions such as scrolling over boxes, highlighting items, and clicking below a field to reveal hidden, dependent fields (such as translation fields for an example sentence). All of these may pose difficulties for certain users.

For general dictionary editing, “Dictionary Browse & Edit” is probably the most useful window, in particular its handy search function that works by looking for approximate matches, allowing some leeway for orthographies that are not yet well known or well established. Searches can be conducted in any of the languages defined for the project (WeSay has no difficulty handling lexicons in more than two languages, though it seems to privilege unidirectionality, working from the language of study.)

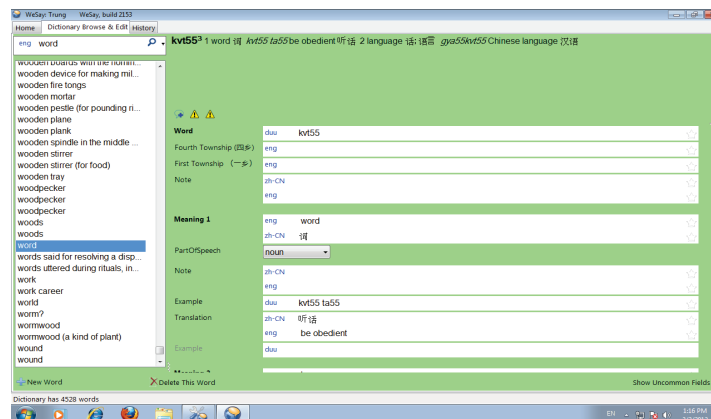


FIGURE 5. An example of the “Dictionary Browse & Edit” window in WeSay’s “front end.”

Collaboration without internet access, although not ideal, is certainly possible. On the one hand, WeSay automatically saves changes to a user’s computer, and can send them using the “Send/Receive” function the next time the user has internet access. On the other hand, the dictionary can be downloaded to a USB drive on one computer and then synced with the version of the dictionary on another machine. In terms of material output for different purposes, WeSay offers several options: sending by e-mail, exporting to SFM, opening in Lexique Pro, and creating a PDF (a basic but decent option, and a quick way to demonstrate results). WeSay automatically saves a project in LIFT format—making it theoretically very straightforward to go back and forth between WeSay and FLEx—although it seems best to minimize this, since problems can crop up when mapping the rich categories and features of FLEx onto the much greater simplicity of WeSay.

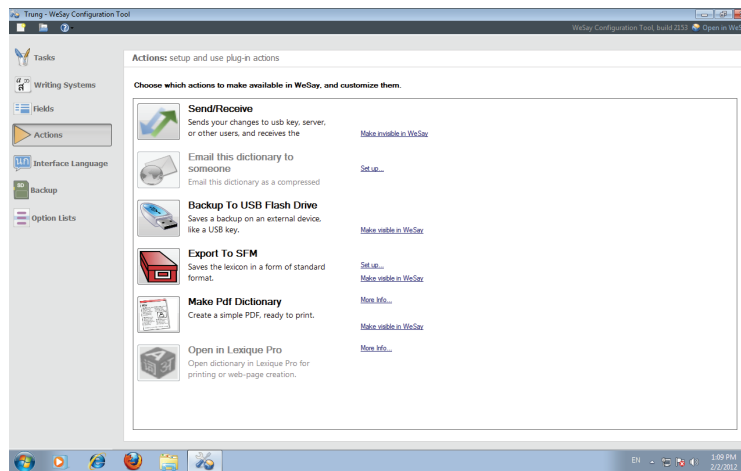


FIGURE 6. Options available for outputting and sharing a dictionary project.

3. LIMITATIONS. WeSay is no substitute for in-person collaboration between linguists and native speakers, and the advisor can expect to spend a fair amount of time “cleaning up” the material input by users and resolving conflicts where multiple users are involved. “Running” the back end will require a certain amount of technical (and English) fluency, so WeSay will probably not serve well at present for communities operating their own dictionary projects. A somewhat established orthography and keyboard input system is presumed, as is some knowledge of a language of wider communication and its metalanguage (“word,” “sentence,” “part of speech,” etc.). Advisors should not count on the ability to resolve all issues remotely, however—for instance, a user switching computers will find it difficult to “move” WeSay. The WeSay developers (who also run wesay@googlegroups.com) appear to be prompt and helpful in trying to resolve issues—as in my own experience with the program, in which I found the Chorus version control system to be particularly finicky, prone to errors and bugs.

4. SUMMARY OF WESAY.

- Primary function: Dictionary collaboration with non-linguists, especially community members with limited computer experience and limited access to high-performance computing equipment.
- Pros: Enables remote collaboration with community members; customizable for particular projects (word gathering, image gathering, etc.), not limited to compilation of whole dictionaries; allows for just-in-time training of collaborators and step-by-step work; genuinely easy, intuitive interface; many actions handled for the user, such as backing up and file management; impressive version control and online collaboration capabilities; decent integration with FLEx and variety of output methods; search tool operates with approximate match
- Cons: May require significant upfront investment and maintenance from linguist-advisor; sometimes still too complex for less experienced users; occasional bugginess, especially with Chorus version control system; opaque merge conflict and error reports
- Platforms: Windows 2000, XP, Vista, Windows 7, Linux
- Open Source: WeSay officially supports Ubuntu Linux distributions and can be installed for Linux using the APT tool or the Synaptic package manager
- Cost: Free
- Reviewed Version: WeSay 0.8.2153 (current “stable” releases begin with 0.8; “dev” releases with new features begin with 0.9)
- Application Size: 31.8 MB for entire directory
- Documentation: Manual available through WeSay website at this wiki URL: http://wesay.org/wiki/Help_And_Contact
- Requirements: Net 3.5 SP1, Microsoft Installer 3.1 (or later)
- Download To download: <http://projects.palaso.org/repositories/entry/wesay-doc/WeSay%20Documentation%20Printable.pdf>

REFERENCES

- Albright, Eric & John Hatton. 2008. WeSay, A tool for engaging native speakers in dictionary building. In D. Victoria Rau & Margaret Florey (eds.), *Documenting and revitalizing Austronesian languages*, 189–201. *Language Documentation & Conservation* Special Publication 1. Honolulu: University of Hawai’i Press. <http://hdl.handle.net/10125/1368>
- Moe, Ron. 2006. Dictionary development process. SIL International. <http://www.sil.org/computing/ddp>