Multilingual language learning with Grammatical Framework

Adrian De Lon[1,2] **Inari Listenmaa**[3]
[1]AI4REASON, Prague, Czechia
[2]Mathematical Logic Group, University of Bonn, Bonn, Germany
[3]Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg, Gothenburg, Sweden
adelon@uni-bonn.de inari@chalmers.se

Abstract

We present Palaver, a multilingual language-learning tool built with Haskell and Grammatical Framework. Shared abstract representations yield the phrases being learned, their translations, interlinear glosses, and alternative answer choices. Progress is tracked through lesson completion and spaced repetition, independent of the metalanguage. The prototype includes demonstration courses in seven languages.

1 Introduction

A multilingual language-learning tool should let learners combine their *target language* with their preferred supporting language of instruction (the *metalanguage*). The metalanguage need not be the learner's first language. For instance, a learner who is fluent in their second language, and is learning another language in the same language family, might prefer to learn the new vocabulary via their second language. In the acquisition of a third language, a typologically close second language can be a stronger source of syntactic transfer than the first language. Bardel and Falk (2007) attribute this to its status as a second language, Rothman (2011) to typological proximity.

We use *Grammatical Framework* (GF) (Ranta, 2011) to organize multilingual language-learning material in a compositional manner. Course material is represented as typed abstract syntax in GF. Each abstract tree can be realized through different concrete syntaxes, which can be used for generating expressions in the target language, glosses, and translations. For example, an abstract greeting is mapped to concrete realizations such as *Hello* or *Hej*. Figure 1 shows how a shared abstract representation can act like an interlingua. GF can parse

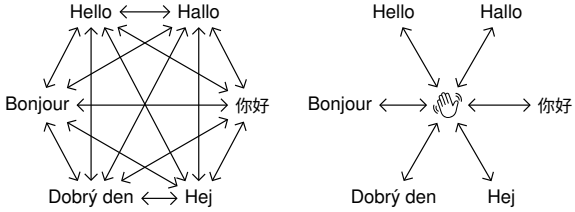


Figure 1: Pairwise mappings of greetings (left) vs. mapping through a shared abstract representation, here symbolized by a waving hand (right).

a concrete expression into one or more abstract trees, allowing the same grammar to support both presentation and answer checking. Shared material can be supplemented by material that deals with aspects that are specific to a language or a language family, such as particular grammatical features.

We also explore the integration of interlinear text into learning material, using morphological analyses that GF's *Resource Grammar Library* (RGL) (Ranta, 2009) provides. The glossing component of our tool builds on the custom resource grammar of our experimental glossing tool (De Lon and Listenmaa, 2026), which generates parallel interlinear lines from a shared grammatical analysis.

In comparison to earlier language-learning tools based on GF (Section 2), Palaver[1] adds interlinear glosses with lexical glosses in the metalanguage (Section 4), cloze distractors built from replacement of a subtree with another of the same type (Section 4.3), exercise states and review schedules that persist across changes of the metalanguage (Sections 5.3 and 5.4), and speech synthesis (Section 6).

2 Related work

Previous language-learning tools built with GF are the closest related work. Ljunglöf (2011) de-

This work is licensed under a Creative Commons Attribution 4.0 International Licence. Licence details: http://creativecommons.org/licenses/by/4.0/.

[1]https://github.com/adelon/palaver

Proceedings of the 15th Workshop on Natural Language Processing for Computer Assisted Language Learning (NLP4CALL 2026)
pages 111–118, November 19–20, 2026, University of Tartu Library

scribed how syntax trees can be edited through grammar-preserving operations on their textual realizations. MULLE (Lange and Ljunglöf, 2018b) and the MUSTE environment (Lange and Ljunglöf, 2018a) used this editing approach for translation exercises generated from lesson grammars. Abolahrar (2011) developed a multilingual translation quiz that generated questions and checked answers using GF.

Beyond GF, grammatical resources have been used both to generate exercises and to analyze learner responses. OAHPA! (Antonsen et al., 2009) used finite-state morphology and a Constraint Grammar parser to generate North Sámi exercise material, analyze learner responses, and provide grammatical feedback. GramEx (Perez-Beltrachini et al., 2012) used grammar-based generation to produce exercises targeting specified grammatical phenomena with controlled syntax and vocabulary.

Other tools have also used linguistic analysis to support reading and exercises based on texts. GLOSSER-RuG (Nerbonne and Smit, 1996) was designed to help Dutch speakers read French: for a selected word, it showed a morphological analysis, a bilingual dictionary entry, and corpus examples. Revita (Katinskaia et al., 2018) generated exercises from learner-selected texts and selected exercises based on previous responses. LARA (Akhlaghi et al., 2019) supported reading with translations, audio, and personalized concordances.

We build directly on decades of grammar engineering work that has gone into the RGL and on ideas implemented in earlier GF-based language-learning tools.

3 Compositional multilingual authoring

3.1 Abstract and concrete syntax

In GF, an abstract syntax declares categories and typed functions for building trees, which represent language-independent meanings. A *concrete syntax* maps these meanings to strings through *linearization* definitions. Different concrete syntaxes can linearize the same abstract tree into different languages. Concrete syntax can use records and tables indexed by grammatical features, such as case, number, or tense (Ranta and Listenmaa, 2023). This allows syntax rules to select inflected forms in context and makes it possible to associate those forms with grammatical labels in a separate concrete syntax for glossing (Section 4). GF's Resource Grammar Library offers language-specific morphology and syntax as a reusable library. Our implementation builds on it with an *application grammar* for the course material.

With this grammatical approach, we can build expressions from reusable components. For example, the noun *house*, the adjective *small*, and adjectival predication can combine to express *the house is small*. The grammar can also represent communicative purposes or situational phrases, such as greeting someone or apologizing, without prescribing a shared sentence structure. For example, an apology might be expressed by a single word in one language and a full sentence in another. Thus, abstract syntax can represent both the parts of a compositional expression and a communicative act as a whole.

Sharing abstract syntax does not mean that every language must have identical lessons. We aim to share most vocabulary, basic communicative functions, and short texts across languages. Explanations and exercises about grammatical features that are specific to a language or language family can be implemented without sharing. For instance, a course teaching English should not have a lesson on the illative case, so it makes sense to realize such a lesson only for languages that have the respective feature, such as Finnish. Each course can freely combine shared material, realized through each language's concrete syntax, with material developed for specific features of the target language.

At run time, a course combines the *target concrete syntax*, which produces the expression being learned, with the *metalanguage concrete syntax*, which produces translations and lexical glosses. The user interface is localized through a separate layer and kept in sync with the metalanguage.

3.2 Resource reuse

Let U be the number of shareable units in a course inventory and L the number of languages with realizations of that inventory. A unit may be a lexical concept, a communicative phrase, or a sentence tree used in a lesson. The shared abstract inventory requires $O(U)$ authored objects. Defining a realization of each unit in each language requires at most $O(LU)$ language-specific entries, totalling $O(U + LU) = O(LU)$. Realizations in different languages are connected only through the

shared abstract trees, so these resources can generate courses for all metalanguage–target-language pairs without maintaining an explicit translation layer for each pair. Authoring and maintaining the same inventory independently for each ordered language pair would instead require up to $O(L^2U)$ entries.

These orders refer to the number of authored units, which does not necessarily correspond to effort or development time. This number excludes grammatical resource development, language-specific curricula, and contrastive explanations. For example, a Finnish illative exercise can share its tree across metalanguages, but its explanation still needs to be written for the intended learners. Beyond reuse, generating realizations from trees instead of storing them as fixed strings provides composition, contextual inflection, parsing, and coordinated generation of forms and annotations, as the story example below illustrates. Grammatical information can also help with disambiguation, for example for speech synthesis. Consider the pronunciation of *read* in *I read books*, which depends on its tense.

In our prototype, authoring requires direct creation of abstract syntax trees as symbolic expressions. A parse-and-select interface or a visual tree editor could make this easier.

4 Learner-oriented interlinear glosses

Interlinear glossed text, as used in linguistics, often includes a segmented object-language line, a morpheme-by-morpheme gloss, grammatical category labels, and a free translation. The Leipzig Glossing Rules (Comrie et al., 2015) are a commonly used convention for alignment, morpheme boundaries, and grammatical labels. Interlinear translations have a long history in language teaching (Bonilla Carvajal, 2026). In language learning, more lightweight annotations are common, often using the inflection of the metalanguage instead of elaborate chains of grammatical labels.

There is some empirical evidence that supports using glosses for learning vocabulary (Yanagisawa et al., 2020; Boers, 2022), though evidence on interlinear glossing specifically, as well as evidence on using glosses for grammar learning, is limited or mixed. In one study with 48 Spanish-speaking novice learners of English, those who read a text with interlinear translations translated more of its words, both immediately and eight days later, than those who read it with two other gloss layouts (Bonilla Carvajal, 2025). We use interlinear glosses to make grammatical information available in a compact form. Their usefulness may depend on the learner's familiarity with the notation and linguistic terminology in general.

4.1 Glosses as parallel generation from a shared grammatical analysis

In GF, interlinear glosses can be realized through parallel generation from a shared grammatical analysis (De Lon and Listenmaa, 2026). In that glossing tool, an input sentence is parsed, candidate trees are ranked heuristically, and the first tree whose relinearization aligns with the input words is selected. Parallel concrete syntaxes then generate the object-language forms, morpheme segmentation, glosses, and a diagnostic translation. In our learning tool, the course author supplies the abstract tree directly. Thus the analysis is fixed, without requiring selection heuristics. The author also chooses exercise targets and alternative subtrees (e.g. as distractors for cloze answers), turning the analysis into learning material. Preparing a text requires grammar coverage of its vocabulary and constructions, a tree for each sentence, and exercise specifications where applicable.

The target concrete syntax linearizes the tree into a complete sentence. A dedicated glossing concrete syntax linearizes the same abstract tree into aligned cells. Each cell contains the surface form, a segmentation into morphemes, a lexical gloss, and grammatical labels. Usually the cells are aligned word for word, but lexicalized multiword expressions can be combined into one cell. Our tool linearizes each lexical item in the metalanguage concrete syntax to obtain its lexical gloss. For instance, the lexical gloss displayed below a Finnish word can change from English to German or Swedish without changing the Finnish sentence or its grammatical analysis. Grammatical labels such as ILL.SG belong to the shared glossing concrete syntax and are currently the same for every metalanguage.

The glossing resource grammars extend the existing RGL grammars with morpheme boundaries and grammatical labels. They use the same *smart paradigms*, building inflection tables from compact specifications. Segmentation rules, adapted from the glossing tool, are defined per inflection paradigm, so a new word of an already covered

paradigm needs no further glossing work. Its glosses are derived from the abstract tree.

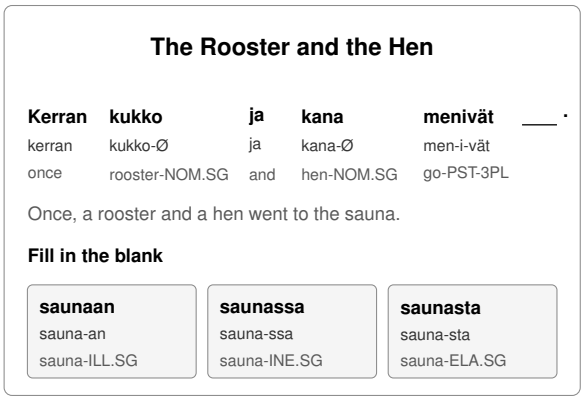


Figure 2: Schematic story interface with a morpheme-segmented line, glosses, a metalanguage translation, and a case-form exercise. Ø marks a zero morpheme. A correct answer advances the story to the next sentence.

Figure 2 illustrates this for the Finnish sentence *Kerran kukko ja kana menivät saunaan. Kukko* and *kana* are given the English lexical glosses *rooster* and *hen* and are both labelled with NOM.SG. The verb *menivät* is given the lexical gloss *go* and PST-3PL; *saunaan* is given the lexical gloss *sauna* and ILL.SG. The English realization of the underlying abstract tree (Figure 3) is *Once, a rooster and a hen went to the sauna.*

The interlinear gloss follows the word order of the sentence in the target language, whereas the translation realizes the sentence via the normal concrete syntax of the metalanguage. For example, Finnish *saunaan* and its glosses use the glossing resources for the Finnish illative case, whereas the English translation uses *to the sauna*.

4.2 Adapting the gloss to the learner

The morpheme-segmented line, as illustrated in Figure 2, shows learners how the words are formed. Glosses can become richer as a learner progresses through the course. Early on, some grammatical labels might be omitted. For example, when a learner has only seen verbs in the present tense, there is not yet a reason to mark this explicitly until the distinction is introduced in the course. This is consistent with the Leipzig Glossing Rules, which allow the amount of detail to depend on the reader's background and the purpose of the gloss. Hiding a label does not remove the information from the underlying analysis in GF. The tool could still use this information in exercises or display it in a later lesson.

4.3 Glosses in exercise material

In the story example, the rooster and hen went *to the sauna*. In the abstract tree of this sentence (Figure 3), the relation `Destination_Rel` specifies that the sauna is the destination of their movement. To generate distractors, we replace this relation with alternatives specifying that the movement takes place *in the sauna* or starts *from the sauna*. These alternatives have the same type in the abstract grammar, so the resulting tree is still welltyped in GF. Finnish expresses this distinction through noun inflection. Linearizing the original and modified trees yields *saunaan* 'to the sauna' (illative), *saunassa* 'in the sauna' (inessive), and *saunasta* 'from the sauna' (elative), together with their glosses. These forms become the answer choices for the cloze.

The task in Figure 2 is to fill the blank to express the meaning given by the English translation, i.e. to select the correct Finnish form expressing movement *to the sauna*. The distractors are welltyped and grammatical, but their meaning is incorrect in this context. A correct choice restores the cell and allows the learner to continue with the story; an incorrect choice leads to another attempt.

Just replacing a subtree with another of the same type does not by itself yield a usable cloze test. The subtree must be chosen with care to yield a sensible test. The current construction requires every distractor to change the same single aligned cell and to yield a distinct answer form: distractors that change several cells, disrupt alignment, or coincide in surface form are excluded. A distractor must also differ from the correct answer in its translation, since the translation is the cue.

In Figure 2, the glossed choices help the learner in the interpretation of the forms and their labels, so success does not demonstrate unaided recall. In particular, this question does not test if the learner knows the word *sauna*. Story lessons can also include vocabulary questions with unglossed answer options, requiring the learner to recognize the expression that matches the meaning cue.

5 Exercises and learning progress

Our tool currently supports vocabulary matching, multiple-choice clozes, typed-answer exercises, flashcards, story exercises, guided interactions, and Chinese character writing. GF partici-

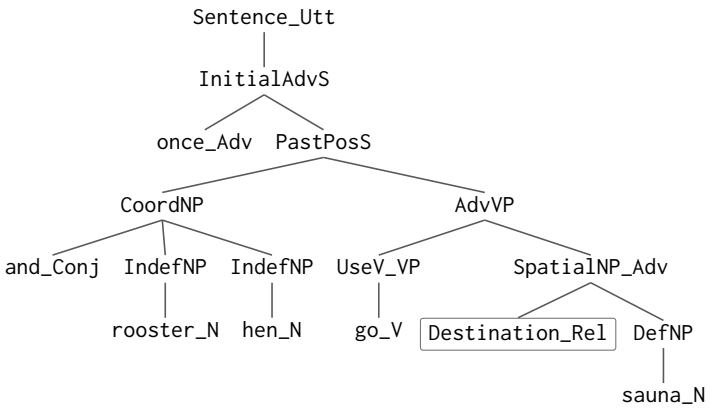


Figure 3: Abstract tree of the sentence in Figure 2. The distractors replace the boxed `Destination_Rel` with `StaticLocation_Rel` and `Source_Rel`, which Finnish realizes as the inessive *saunassa* and the elative *saunasta*.

pates at different points in these exercises. It realizes lexical items and sentences for matching and flashcards, derives cloze spans and context-sensitive choices from trees, and parses typed answers in the target language.

Chinese character-writing exercises use the Hanzi Writer library (Chanin et al., 2026) for stroke input and feedback. The writing lesson asks learners to trace the outlines of the twelve characters in the eight basic phrases of the first lesson.

5.1 Guided interactions

Guided interactions ask the learner to complete a task in incremental steps, with feedback and optional help. For instance, to express *the house is red* in Swedish, the learner can supply the definite subject *huset* (*house*-DEF) and the agreeing adjective *rött* in either order, or type the whole sentence *huset är rött*. The learner can revise an incorrect answer, request hints, or ask the tool to demonstrate a step or the remaining solution. Our demo lessons also include a worked example, in which the tool demonstrates every step, and a task in which the learner first chooses between the adjective forms *kall* and *kallt* and then completes the sentence.

This design is inspired by Heeren et al. (2010), who specify valid sequences of solution steps for mathematical exercises through rewrite strategies. We use finite strategies built from sequences and choices of steps. After each accepted or demonstrated step, the remaining strategy determines which steps are available and when the task is complete. GF supplies the surface forms, choices, model sentences, and the linguistic examples in hints. Hints and explanations are authored separately. Answers consisting of individual forms are checked against the forms generated by GF, while whole sentences are checked by parsing. The tool stores a timestamped record of each submitted answer, hint request, and demonstrated step, together with the metalanguage in use. These records provide process data showing how the learner reached a solution and whether it was correct.

5.2 Answer checking

In exercises where the learner is asked to type out a sentence, an answer is accepted when one of its analyses is the intended abstract tree. This allows the tool to accept different surface forms that GF maps to the same tree, without requiring course authors to list them all as hard-coded strings. Parsing also distinguishes three kinds of rejected answers: an answer that the grammar recognizes with a different tree, an answer that seems incomplete, and an answer that the grammar does not cover. Valid responses outside the exercise grammar, or acceptable paraphrases represented by another tree, may be rejected. The metalanguage cue must therefore supply enough context to identify the intended analysis. For example, an English *you* cue alone may not specify the politeness level required in a language with a pronominal address distinction such as French or German.

We also apply language-specific tolerance when checking typed input, such as automatically correcting missing diacritics. For Swedish, *en rav* is accepted as a translation of *a fox*, with a warning that the correct form is *en räv*.

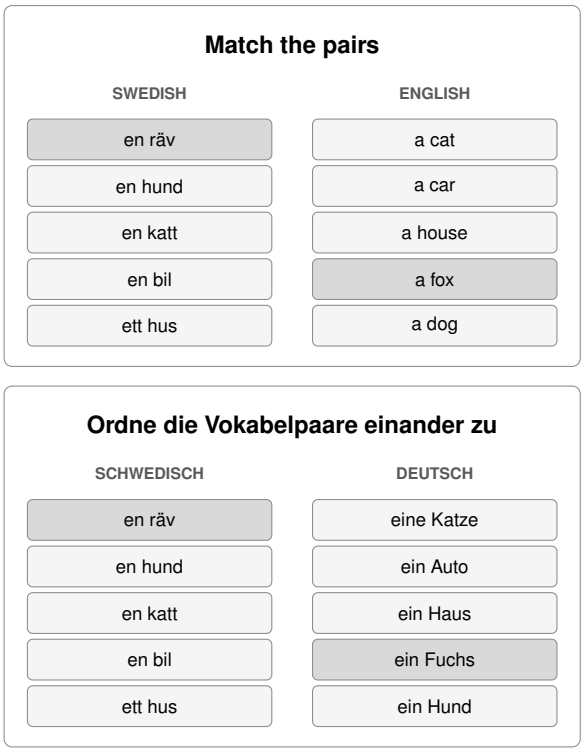


Figure 4: The same Swedish vocabulary exercise with English (top) and German (bottom) as metalanguage. Learning progress is preserved when the interface and translations change.

5.3 Preserving progress across metalanguages

A learner can change the metalanguage during an exercise without losing progress, as shown in Figure 4. The tool stores an exercise attempt as a reference to the exercise, a random seed, and a state that does not depend on the metalanguage. When the learner changes metalanguage, the tool regenerates the same exercise from the seed in the new metalanguage and restores its state. Lesson completion and exercise results are likewise stored independently of the metalanguage.

5.4 Spaced review

Progress is tracked through lesson completion and review schedules for vocabulary, grammar, and character-writing targets. We schedule reviews with FSRS-6 (Ye, 2026), which builds on the memory models of Ye et al. (2022) and Su et al. (2023). For each learner, we maintain separate review schedules for each learning objective in the target language and each kind of task. Tasks differ in the response they require, such as recognition or production, and in the direction of recall: for example, from a meaning cue to an expression, or from an expression to its meaning. For instance, recalling Swedish *Hej* from a greeting cue tests vocabulary recall, while choosing between the articles *en* and *ett* for *räv* is intended to test knowledge of its grammatical gender.

Reviews share a schedule when they practise the same objective through the same kind of response and direction of recall. Recalling *Hej* from English *Hello* and later from German *Hallo* contributes to the same history.

For automatically graded questions, the first answer determines the rating: an incorrect answer maps to *Again* and a correct answer to *Good*. Flashcards use the learner's self-rating after revealing the answer.

Completing a writing lesson makes each character available in a separate handwriting review schedule. These reviews hide the outline and ask for a character at a specified position in a phrase, using the phrase's pīnyīn and translation as cues. A separate concrete syntax linearizes the phrase trees as pīnyīn. After writing or revealing the answer, the learner rates recall as *Again*, *Hard*, *Good*, or *Easy*. Only this self-rating determines the schedule update. Stroke errors are recorded but do not currently impact the scheduling.

Changing the metalanguage preserves the review history and schedule, and the metalanguage is recorded as review context. This design assumes that learners are reasonably comfortable in each metalanguage that they choose. The language of a cue may nevertheless affect performance. Combining evidence across metalanguages is a modelling decision whose pædagogical appropriateness needs to be tested.

6 Speech synthesis and implementation details

Exercises with audio support have a *Listen* button which provides a pronunciation generated by Chatterbox (Resemble AI, 2026), an open-source voice model. The text input is the linearization of the exercise's tree in the target concrete syntax, so the audio matches the displayed text. Audio is available for flashcards, cloze sentences, model answers of typed exercises, and story sentences. The synthesized audio is cached as files and can also be generated in advance.

Palaver is implemented as a web application in Haskell, using a combination of Twain, Lucid 2, and HTMX 4. It loads the compiled grammars

Target	Further lessons	Metalanguages
Czech	–	all others
English	glossed story	German, French
Finnish	glossed, segmented story	all others
French	–	all others
German	nouns	all but Finnish
Mandarin	character writing	all others
Swedish	nouns, sentences	all but Finnish
	guided interactions	English

Table 1: Demonstration courses. Every course starts with a lesson of eight basic phrases. All seven languages are available as interface languages.

with GF's Haskell runtime, which provides linearization and parsing, and stores learner data in SQLite.

7 Prototype scope and limitations

Our prototype is an architecture demonstration. It has a small set of lessons and supports Czech, English, Finnish, French, German, Swedish, and Mandarin Chinese (Table 1). Courses will require significant revisions and extensions to actually become useful for language learning. There are also still some architectural limitations, such as alignment restrictions and lack of support for alternate answer trees. Authoring material that works across languages requires care, for example to ensure that cues make important distinctions clear. Speech synthesis currently runs only on macOS and does not support all languages. Voice input is also missing and would be a natural future feature. Guided interactions are available for Swedish with English as the metalanguage and do not integrate with spaced review yet. Interface localization uses a simple string-based approach, independent of GF. Exercises with typed answers still have limited diagnostics. We do not make any claims about practical authoring efficiency or learning outcomes.

8 Conclusion

We have presented Palaver, a language-learning tool built around shared abstract representations in GF. Concrete syntaxes realize course material as sentences, translations, and interlinear glosses, while the same trees support exercise generation and answer checking. Learners can change the metalanguage without losing exercise progress or starting a separate review history. Shared course material can be reused across different languages and supplemented with language-specific content.

Acknowledgements

This work was supported by the EU through the ERC grant *NextReason* (doi:10.3030/101200949). We thank the reviewers for their comments.

AI Disclosure

We used Claude Opus 5.5 for spellchecking and routine editing. Speech synthesis in our tool uses Chatterbox. Our application grammar and the Haskell web app include agent-assisted code.

References

Elnaz Abolahrar. 2011. Multilingual grammar-based language training: Computational methods and tools. Master of science thesis in software engineering and technology, Chalmers University of Technology, Göteborg, Sweden, September. Department of Computer Science and Engineering, Division of Language Technology.

Elham Akhlaghi, Branislav Bédi, Matt Butterweck, Cathy Chua, Johanna Gerlach, Hanieh Habibi, Junta Ikeda, Manny Rayner, Sabina Sestigiani, and Ghil'ad Zuckermann. 2019. Overview of LARA: A learning and reading assistant. In *8th ISCA Workshop on Speech and Language Technology in Education (SLaTE 2019)*, pages 99–103.

Lene Antonsen, Saara Huhmarniemi, and Trond Trosterud. 2009. Interactive pedagogical programs based on constraint grammar. In *Proceedings of the 17th Nordic Conference of Computational Linguistics (NODALIDA 2009)*, pages 10–17, Odense, Denmark. Northern European Association for Language Technology (NEALT).

Camilla Bardel and Ylva Falk. 2007. The role of the second language in third language acquisition: The case of Germanic syntax. *Second Language Research*, 23(4):459–484.

Frank Boers. 2022. Glossing and vocabulary learning. *Language Teaching*, 55(1):1–23.

Camilo Andrés Bonilla Carvajal. 2025. Interlinear translations reduce cognitive load on EFL vocabulary acquisition. *Íkala, Revista de Lenguaje y Cultura*, 30(1):1–23.

Camilo Andrés Bonilla Carvajal. 2026. Interlinear translations for language teaching: The principles, practice, and results for the last five hundred years. *Historiographia Linguistica*.

David Chanin et al. 2026. Hanzi Writer. `https://hanziwriter.org/`.

Bernard Comrie, Martin Haspelmath, and Balthasar Bickel. 2015. The Leipzig Glossing Rules: Conventions for interlinear morpheme-by-morpheme

glosses. Revised version of February 2008; last change May 31, 2015.

Adrian De Lon and Inari Listenmaa. 2026. Multilingual interlinear glossing in the Grammatical Framework. In *Proceedings of the 22nd Conference on Natural Language Processing (KONVENS 2026)*. To appear.

Bastiaan Heeren, Johan Jeuring, and Alex Gerdes. 2010. Specifying rewrite strategies for interactive exercises. *Mathematics in Computer Science*, 3(3):349–370.

Anisia Katinskaia, Javad Nouri, and Roman Yangarber. 2018. Revita: a language-learning platform at the intersection of ITS and CALL. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Herbert Lange and Peter Ljunglöf. 2018a. Demonstrating the MUSTE language learning environment. In *Proceedings of the 7th workshop on NLP for Computer Assisted Language Learning*, pages 41–46, Stockholm, Sweden. LiU Electronic Press.

Herbert Lange and Peter Ljunglöf. 2018b. MULLE: A grammar-based Latin language learning tool to supplement the classroom setting. In *Proceedings of the 5th Workshop on Natural Language Processing Techniques for Educational Applications*, pages 108–112, Melbourne, Australia. Association for Computational Linguistics.

Peter Ljunglöf. 2011. Editing syntax trees on the surface. In *Proceedings of the 18th Nordic Conference of Computational Linguistics (NODALIDA 2011)*, pages 138–145, Riga, Latvia. Northern European Association for Language Technology (NEALT).

John Nerbonne and Petra Smit. 1996. GLOSSER-RuG: in support of reading. In *COLING 1996 Volume 2: The 16th International Conference on Computational Linguistics*, pages 830–835.

Laura Perez-Beltrachini, Claire Gardent, and German Kruszewski. 2012. Generating grammar exercises. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, pages 147–156, Montréal, Canada. Association for Computational Linguistics.

Aarne Ranta. 2009. The GF Resource Grammar Library. *Linguistic Issues in Language Technology*, 2(2).

Aarne Ranta. 2011. *Grammatical Framework: Programming with Multilingual Grammars*. CSLI Studies in Computational Linguistics. CSLI Publications, Stanford.

Aarne Ranta and Inari Listenmaa. 2023. Grammatical Framework: From interlingual semantics to morphophonemic processes. In Arvi Hurskainen, Kimmo Koskenniemi, and Tommi Pirinen, editors, *Rule-Based Language Technology*, volume 2 of *NEALT Monograph Series*, pages 9–48. Northern European Association for Language Technology.

Resemble AI. 2026. Chatterbox-TTS. `https://github.com/resemble-ai/chatterbox`.

Jason Rothman. 2011. L3 syntactic transfer selectivity and typological determinacy: The typological primacy model. *Second Language Research*, 27(1):107–127.

Jingyong Su, Junyao Ye, Liqiang Nie, Yilong Cao, and Yongyong Chen. 2023. Optimizing spaced repetition schedule by capturing the dynamics of memory. *IEEE Transactions on Knowledge and Data Engineering*, 35(10):10085–10097.

Akifumi Yanagisawa, Stuart Webb, and Takumi Uchihara. 2020. How do different forms of glossing contribute to L2 vocabulary learning from reading? A meta-regression analysis. *Studies in Second Language Acquisition*, 42(2):411–438.

Junyao Ye. 2026. The algorithm: FSRS-6. GitHub wiki. Accessed 2026-09-11.

Junyao Ye, Jingyong Su, and Yilong Cao. 2022. A stochastic shortest path algorithm for optimizing spaced repetition scheduling. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4381–4390, New York, NY, USA. Association for Computing Machinery.