Demonstrating the MUSTE Language Learning Environment

Herbert Lange and **Peter Ljunglöf**
Computer Science and Engineering
University of Gothenburg and Chalmers University of Technology

Abstract

We present a language learning application that relies on grammars to model the learning outcome. Based on this concept we can provide a powerful framework for language learning exercises with an intuitive user interface and a high reliability.

Currently the application aims to augment existing language classes and support students by improving the learner attitude and the general learning outcome. Extensions beyond that scope are promising and likely to be added in the future.

1 Introduction

In this paper we demonstrate MULLE, the MUSTE Language Learning Environment (Lange and Ljunglöf, 2018a). It is a versatile software system that doubles both as an authoring environment for language learning exercises and as a flexible language learning system.

It has an open architecture which makes it adaptable to many different use cases. The main use case we present here is in the context of a traditional language class based on a classic textbook. This limited context facilitates both conceptualization and development.

2 Features

The system we present employs many features to support a positive learning outcome.

The user interface is a system-independent web interface with low overhead that guarantees for intuitive user interactions by using a grammar-backed text editing method that works on the word level instead of on the character level (Ljunglöf, 2011). This method maps editing operations on the surface of a sentence onto modifications on the underlying syntax tree.

The learning process is structured following a schema of lessons and exercises. The whole language learning process is split into several lessons and each lesson consists of several exercises that have to be solved to pass a lesson. The lessons are based on multilingual translation grammars between a source language and the target language. Based on these lesson grammars a large set of exercises can be created, each exercise consisting of two sentences, and the learner's task is to use the abovementioned text editing method to change one of the sentences to make it a proper translation of the other.

The reliance on grammars for modeling the lesson structure as the foundation for the learning process places the approach close to Controlled Natural Languages (Kuhn, 2014) that are well-known for a high reliability for example for transfer-based machine translation. Instead of using the grammars for translation we use them to generate translation exercises but we can provide the same level of reliability (Lange and Ljunglöf, 2018b).

The type of exercises that are generated by our system can be seen as related to Cloze or fill-in-the-blank tests (Taylor, 1953; O'Toole and King, 2011), but much more general. Instead of using corpora to create exercises, we rely on grammars, an idea that also has been explored by (Perez-Beltrachini et al., 2012).

To support the learner motivation we include aspects of gamification. Based on ideas from the Gameflow framework (Sweetser and Wyeth, 2005) we provide *Concentration*, i.e., minimizing the distraction from the task,

Herbert Lange and Peter Ljunglöf 2018. Demonstrating the MUSTE language learning environment. *Proceedings of the 7th Workshop on NLP for Computer Assisted Language Learning at SLTC 2018 (NLP4CALL 2018).* Linköping Electronic Conference Proceedings 152: 41–46.

Challenge by giving a scoring schema, *Control* by providing an intuitive way to modify the sentence, *Clear goals* by providing a lesson structure, and *Immediate feedback* with a color schema to highlight the translation progress.

3 Learner Interaction

Each exercise consists of two sentences in different languages, one language that the user already knows (the *metalanguage*), and the language to be learned (the *object language*). Both sentences differ in some respect, depending on the grammatical features that the lesson is focusing on.

The user interacts with the system by incrementally modifying the object language sentence until it is a correct translation of the metalanguage sentence. The edit operation is based on the work of Ljunglöf (2011).

The editing interaction is done on the word-level, which means that the user is not allowed to enter arbitrary words, phrases or sentences from the keyboard. There are several reasons for this, but one reason is to avoid problems with unknown words and phrases, which is a risk with systems that are supposed to handle free text input (Heift, 2001, section 3). Another reason for disallowing free text input is to make the system accessible for alternative input methods such as mobile phone touch screens.

There are two possible editing operations:

- The user can select (i.e., click, point or otherwise specify) a word (or a phrase) in the text. The system interprets this as a request to either delete the word/phrase, or to replace it with another word (or phrase).
- Alternatively the user can select the space between two words, which is interpreted as a request to insert a new word or phrase.

When the user performs an editing operation, the system searches for similar sentences according to the grammar, and presents them in a menu. The user can select one of the suggestions, or they can reject the suggestions by selecting something else.

The suggestions that are presented are always grammatically correct according to the lesson grammar. This is done by parsing the original sentence, then modifying the syntax trees while keeping them correct according to the grammar, and then linearising the modified trees.

The system tries to be intelligent in the way that it knows which tree nodes are modified, and since it knows which surface words these nodes are responsible for, it can designate each modified sentence to a specific selection of the surface sentence.

3.1 An Illustrative Example

In this example we assume that the metalanguage is English (meaning that the user already knows English), and the object language is Latin (i.e., the language that the user is learning). All screenshots are found as figures 2–5, in appendix A.

The exercise consists of translating the English sentence "*many kings love Paris*" into Latin. As a starting point we have the Latin sentence "*rex librum legit*", meaning "*a king reads a book*" (or "*the king reads the book*", since Latin doesn't make a difference between definite and indefinite form).

Figure 2 shows how the exercise screen looks at the start. Note that the words that already match each other ("*king*" vs "*rex*") are highlighted in green.

Now we have to select something in the Latin sentence to modify. We start with selecting the verb "*legit*" (eng. *read*), and the system shows a menu of possible verbs to replace with. We select the correct verb "*amat*" (eng. *love*), and the Latin sentence changes. Now two words are highlighted because they are matching with the English sentence. (See figure 3).

Second we decide to insert a determiner corresponding to the English word "*many*". We click in front of the first word and the system displays a menu with different determiners. After selecting the word "*multi*", the sentence changes, and there are three highlighted words. (See figure 4).

Note that the inflection form of *rex* changes to *reges*, because the number of the determiner changed from singular to plural. Also note that *amat* changes to *amant* for exactly the same reason.

Finally, we change the noun "*librum*" (eng. *book*) into the proper name "*Lutetiam*" (*Paris*), and the exercise is solved. (See figure 5).

4 Under The Hood

In this section we give a very brief explanation of how the system works under the hood.

A lesson is defined by a grammar that is bilingual, in the sense that both languages share a common syntactic representation. This language-independent syntax is called *abstract syntax*, and for each language there is a mapping from the abstract syntax to the *concrete syntax* for that language. This mapping is called *linearisation*, and its inverse is called *parsing*.

Since languages are inherently ambiguous, several different syntax trees can linearise to the same string. Therefore, we represent each sentence as the set of all its parse trees. The goal of an exercise is to make the object language sentence a translation of the meta-language sentence, and the system tests that by checking if the first sentence has at least one parse tree in common with the second sentence.

4.1 Populating The Menus

Text editing in this system consists of the user selecting modifications of the sentence from one of its menus. The menus are populated like this:

1. First we collect the linearisations for syntax trees that are similar to some parse tree of the original sentence.

2. For each modified linearisation, we decide which words are changed from the original sentence. The affected words (or spaces between words) in the original sentence are called the *selection*.

3. Every selection has a corresponding menu, to which we add the modified linearisation.

The main problem in this procedure is how to find similar syntax trees. We use an idea similar to adjunction in TAG, Tree Adjoining Grammar (Joshi and Schabes, 1997), where we "cut out" a clique of nodes from the tree and replace them with another clique so that the new tree is still grammatical. These connected nodes that we cut out are similar to the auxiliary trees in TAG, as are the ones we put back into the tree. To reduce the number of menu items, we also filter out all similar trees that can be reached in two smaller steps.

4.2 The Grammar Formalism

The grammar formalism that we use in our implementation is Grammatical Framework (Ranta, 2011), because it has very good support for multilingualism, abstract and concrete syntax, and an extensive Resource Grammar Library for up to 30 languages (Ranta, 2009).

Note that all algorithms, and the implementation of the system as a whole, are independent of the grammars and the meta- and object languages. This means that the only thing we have to do to make the system work between e.g. Swedish and French, is to change the bilingual grammar.

5 Lesson Authoring

Traditionally, a language class relies on a textbook which provides the learner with a sequence of lessons, each consisting of a text fragment, a vocabulary list and some exercises to be solved on paper. This approach tends to be inflexible and unappealing to students, especially concerning translation exercises. We remedy this drawback by providing flexibility to this kind of exercise and use a game-like computer system to present them to the learner.

To be able to do this we have to transform the information available in the textbook into a set of lesson grammars. The process of creating a lesson grammar from a textbook lesson consists of three steps. These steps should be automated as much as possible, but at the moment require some human intervention. The steps are the following:

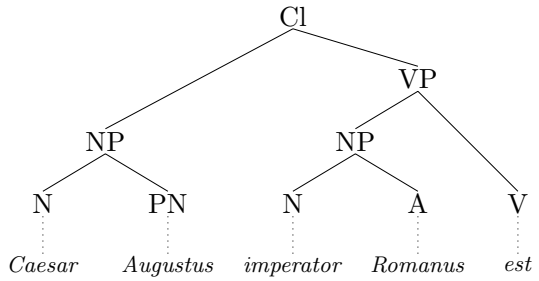
(a) Adapt a lexicon from the textbook lesson, which usually is given as an explicit vocabulary list. Available lexical and morphological resources can be reused.

(b) Create syntax trees for all sentences in the text. This can be done manually

Augustus	(m)	name of an emperor
Caesar	(m)	name of an emperor (later used as title)
imperator, -oris	(m)	emperor
Romanus, -a, -um		Roman
est		(he/she/it) is

```
PN  ::=  "Augustus"
N   ::=  "Caesar"  |  "imperator"
A   ::=  "Romanus"
V   ::=  "est"
```

(a) Formalize the vocabulary from the textbook (top) as a Grammar lexicon (bottom)



(b) Create a syntax tree to cover the sentence

```
NP  ::=  N PN  |  A N
VP  ::=  V NP
Cl  ::=  NP VP
S   ::=  Cl
```

(c) Derive a grammar from the syntax trees

Figure 1: The steps to derive a grammar from a sentence

or semi-automatically by parsing the sentences with an extensive grammar like the ones available in the Resource Grammar Library extended with the new lexicon. In case of several analyses, the correct, i.e. desired, analysis has to be selected manually.

(c) Create a new grammar describing precisely the trees from the previous steps. For that the rules can be read off the inner nodes of the trees. Usually this grammar will be over-generating. Several methods can be used to reduce the grammar, e.g. by merging several rules to one. These grammars can be implemented by using a subset of a Resource Grammar from the Resource Grammar Library.

An example of this process can be seen in Figure 1. The last two steps can profit from having access to the Resource Grammar Library. It makes it easy to add additional languages to a lesson or exchange one language for another, thanks to a high level of abstraction.

The grammar which we get as a result from this process can be used in our application to generate translation exercises based on the content of a syllabus and in the context of a language course. In the end each lesson is covered by one grammar which is specific to exactly the same vocabulary and syntactic complexity of this lesson. This means that the content of the exercises generated from this lesson grammar should already be familiar to a student from the classroom.

To create exercises within a lesson, pairs of sentences have to be selected. These sentences have to be covered by the lesson grammar and it should be within the current abilities of the learner to transform one of the sentences in a way that makes it a proper translation of the other one.

So in conclusion we can say, that a lesson in MULLE consists both of a multilingual grammar including both the meta- and the object language and a set of exercises, i.e. pairs of sentences covered by this grammar. To finish a lesson a subset of these exercises have to be solved.

6 Discussion

The current focus is on supporting existing language classes in a closed classroom setting. This focus is not new and has already some history within the use of machine translation technology for language learning (Richmond, 1994). This is for most languages still the most common way to teach and learn them. However, depending on the language and the context, different kinds of language competence can be the goal of the language classes. Sometimes just translation competence is required while in other circumstances extensive communicative competence is the ultimate goal.

Especially historic languages belong to the first category while most of the modern languages belong to the second. The framework we present here has relevant properties that make it especially suitable for historic languages but it can also be adapted to support language learners that aim for more than just translation competence.

The relevant properties for historic languages are:

- Tackling data sparseness with a grammar-based approach
- Combination of traditional and modern methods of teaching to improve learner motivation
- Both flexible and reliable exercises with high level of control over content

For modern languages additional exercises can be added for the future. These exercises can include among others:

- Morphology exercises to train word forms and agreement
- Graphical exercises containing image description tasks
- Listening exercises

The first kind of exercises can be created by temporarily relaxing grammatical constraints which can be done automatically given grammars in a suitable formalism like Grammatical Framework. The other two exercise types are possible because a sufficiently expressive grammar formalism can not only describe string languages, but can as well express procedures for picture generation or search terms for audio samples in a uniform way.

A pilot evaluation already showed interest in this kind of application both among teachers and students which leads to a concrete plan for the near future. This includes first a full evaluation followed by the extensions sketched here.

References

Trude Heift. 2001. Intelligent Language Tutoring Systems for Grammar Practice. *Zeitschrift für Interkulturellen Fremdsprachenunterricht*, 6(2).

Aravind K. Joshi and Yves Schabes. 1997. Tree-adjoining grammars. In Grzegorz Rozenberg and Arto Salomaa, editors, *Handbook of Formal Languages. Vol 3: Beyond Words*, chapter 2, pages 69–123. Springer-Verlag.

Tobias Kuhn. 2014. A Survey and Classification of Controlled Natural Languages. *Computational Linguistics*, 40(1):121–170.

Herbert Lange and Peter Ljunglöf. 2018a. 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 (NLPTEA '18)*, pages 108–112, Melbourn. Australia. Association for Computational Linguistics.

Herbert Lange and Peter Ljunglöf. 2018b. Putting Control into Language Learning. In *SIGCNL 2018*, Maynooth, Ireland.

Peter Ljunglöf. 2011. Editing Syntax Trees on the Surface. In *Nodalida'11: 18th Nordic Conference of Computational Linguistics*, Rīga, Latvia.

J.M. O'Toole and R.A.R. King. 2011. The deceptive mean: Conceptual scoring of cloze entries differentially advantages more able readers. *Language Testing*, 28(1):127–144.

Laura Perez-Beltrachini, Claire Gardent, and German Kruszewski. 2012. Generating grammar exercises. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, NAACL HLT '12, pages 147–156, Stroudsburg, PA, USA. 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 Publications, Stanford.

Ian M. Richmond. 1994. Doing It Backwards: Using Translation Software To Teach Target-Language Grammaticality. *Computer Assisted Language Learning*, 7(1):65–78.

Penelope Sweetser and Peta Wyeth. 2005. GameFlow: A Model for Evaluating Player Enjoyment in Games. *Computers in Entertainment (CIE)*, 3(3):3–3.

Wilson L. Taylor. 1953. "cloze procedure": A new tool for measuring readability. *Journalism Quarterly*, 30(4):415–433.

A Screenshots From An Exercise Session

See section 3.1 for a deeper explanation of the screenshots in this appendix.

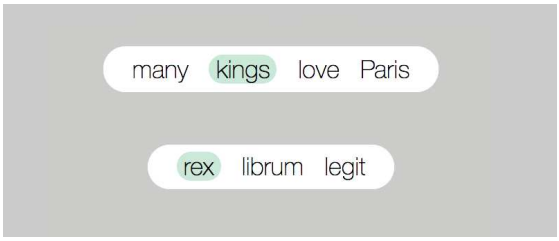


Figure 2: Beginning of the exercise – the lower Latin sentence means *a king reads a book*

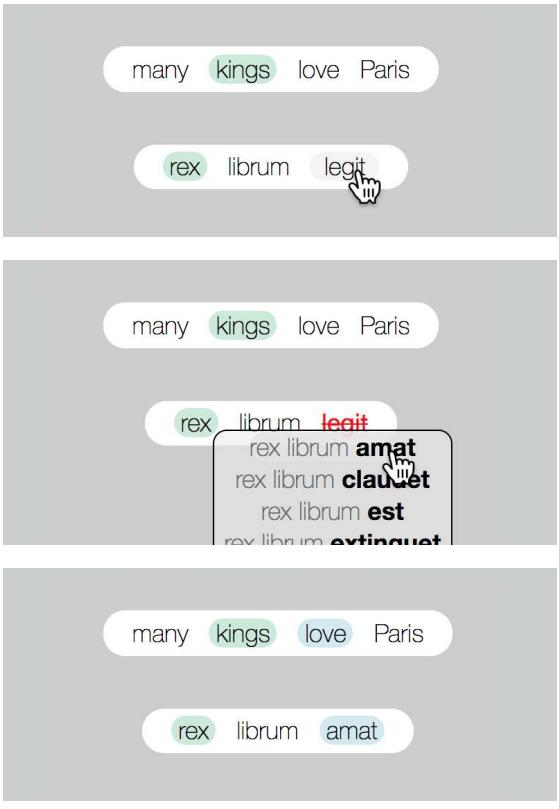


Figure 3: Replacing the verb *legit* (eng. *read*) with *amat* (eng. *love*)

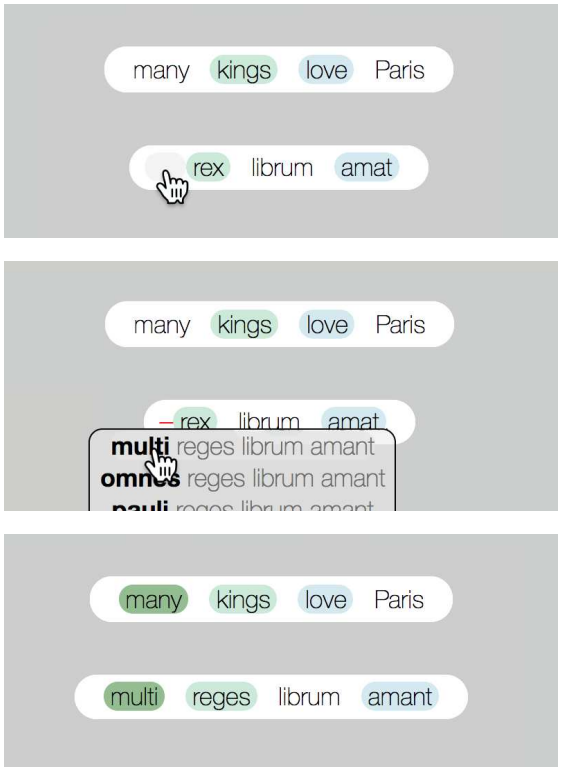


Figure 4: Inserting the determiner *multi* (eng. *many*) before *rex* (eng. *king*)

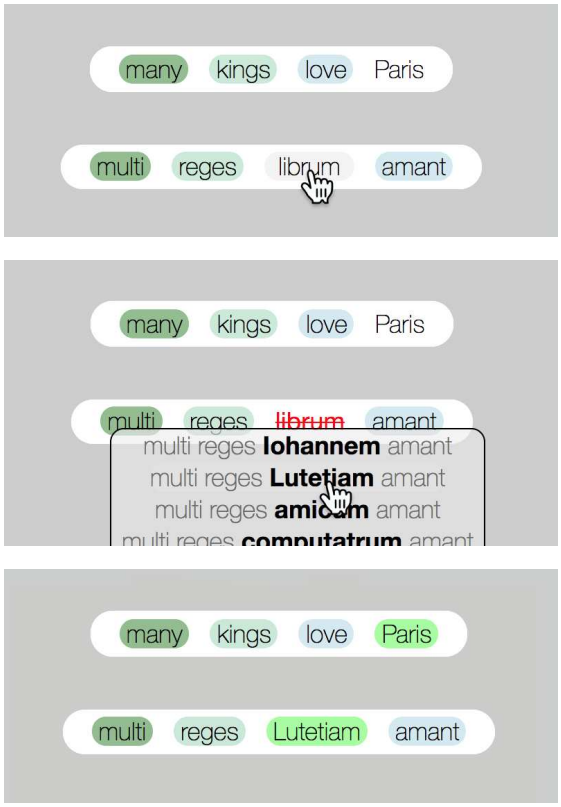


Figure 5: Replacing the noun *librum* (eng. *book*) with *Lutetiam* (eng. *Paris*)