Developing a Sentence Structure Visualising Tool for Language Learning: Practical Challenges and Open Research Opportunities

Maxim Mozgovoy [1,*] and Calkin Suero Montero [2]

[1] School of Computer Science and Engineering, The University of Aizu, Tsuruga, Ikki-machi, Aizu-Wakamatsu, Fukushima, 965-8580 Japan; mozgovoy@u-aizu.ac.jp
[2] Department of Information Technology, Uppsala University, Hus 10, Lägerhyddsvägen 1, Box 337, Uppsala, 751 05 Sweden; calkin.suero.montero@it.uu.se
* Correspondence: mozgovoy@u-aizu.ac.jp

While modern advancements in computing for education arguably can both support and extend existing second language learning practices, designing novel educational activities around emerging technologies is a challenging task from both technical and pedagogical perspectives. The present paper discusses the challenges we faced and the solutions we adopted while designing an instrument for natural language grammar acquisition in language education, based on visual representations that allow learners to freely explore sentences' grammatical structures. We show that even a relatively simple concept rooted in a well-established linguistic theory could not simply be turned into a full-fledged language learning app without numerous compromises due to various technological and pedagogical constraints. The intuitive idea of representing grammar structures with graphical elements proved surprisingly difficult to formalise. It also turned challenging to design a system of visual elements that accurately codify grammar concepts without overwhelming the users. From a conceptual standpoint, our work advances the understanding of how emerging technologies can not only support but also transform second language learning practices by enabling new kinds of learning activities that are not achievable without technological support. From a methodological standpoint, our paper presents a reflective account of the design process behind a practical language learning tool grounded in linguistic theory, highlighting practical compromises and design decisions. We believe our experience can provide valuable insights for CALL researchers, educators, and developers that aim to design innovative language learning applications, educational activities, and tools. Such new kinds of activities and language learning practices open interesting potential research opportunities.

Keywords: computer-assisted language learning; natural language processing; dependency grammar; foreign language acquisition; visualisation.

1. Introduction

Foreign language learning is commonly assisted by technology (Blake, 2017; Godwin-Jones, 2017). The ways in which technology such as computers support foreign language teaching and learning practices have been explored from the perspectives of collaborative learning through computer-mediated communication (Heift & Vyatkina, 2017; Kessler & Bikowski, 2010), synchronous interactions (Bueno Alastuey, 2011), asynchronous activities facilitation (Perveen, 2016), affordances for implementing novel pedagogies (Chapelle, 2019), self-regulated learning and self-efficacy perception (Chen & Hsu, 2020), and providing timely corrective feedback to support grammar awareness (Heift & Vyatkina, 2017), to mention a few. These reports implicitly reflect a well-known framework within the computer-assisted language learning (CALL) community that portrays the distinct roles that a computer may take as a tutor, a tool, or an environment/medium assisting the language learning process. Broadly, within this framework, the tutor provides instruction and feedback, the tool provides access to written and audiovisual learning materials, and the environment/medium facilitates communication and collaboration (Kern, 2006).

The rapid development of mobile technology in recent decades has "brought tutorial CALL back into mainstream" (Hubbard, 2021, p. 19), while CALL research has also focused into exploring the capacity of computers in their role as tools and environments. However, we observe a less explored research area related to the fundamental understanding of how computers could *extend* or *modify* existing foreign language learning. In our opinion, the treatment of computers as merely better tools to reproduce established practices could be narrowed down to the following reasons: a) it is difficult to invent a new type of activity that can be supported by today's technologies; and conversely b) it is difficult to create a technology that would be capable of supporting complex activities such as focused yet flexible conversations (Efimov et al., 2014). Even recent large language model-based instruments are typically employed in traditional learning scenarios, such as conversation practice, writing assistance, and content assessment (Li et al., 2024).

Nevertheless, we find in the literature some examples of activities and systems that *extend* the application area of computers in second language (L2) learning by applying today's state-of-the-art computing technology. For example, Ding et al. (2019) describe an online system able to synthesise sentences using learner's voice, but with the pronunciation corrected to match that of a native speaker's accent. Authors report that hearing correct pronunciation with their own voice indeed helps the learners to improve their pronunciation. Furthermore, Cai and Liu (2018) show how speech recognition technologies can be used to evaluate the pronunciation quality of arbitrary phrases and thus can assist foreign language pronunciation training. Butler (2015) engage children to design computer games that would support their foreign language vocabulary acquisition. Taking a more technology-focused approach, Divekar et al. (2021) present an immersive intelligent CALL (ICALL) environment powered by interactive conversational agents to support Chinese learning as a foreign language. In their preliminary test, the authors report an improvement in the vocabulary, comprehension and conversation skills of the participants using the learning environment. Addressing a pedagogical perspective, Chapelle (2019) discusses reports from the literature indicating how new tools and forms of interaction through technology-mediated environments have fostered new approaches for L2 education

Focusing on L2 grammar, proofreading tools have long been a built-in feature of text processing software such as Microsoft Word, but only recently commercial grammar checking software developers turned to the needs of L2 learners. For instance, the impact of a commercial digital system Grammarly on learner writing is analysed in several research works (Ghufron & Rosyida, 2018; ONeill & Russell, 2019; Qassemzadeh & Soleimani, 2016). Within the CALL research community, the issues of technology-based L2 grammar teaching and learning is discussed by Heift and Vyatkina (2017). They categorize the contribution of technology to L2 grammar instruction from the pedagogical aspects that the technology affords: supporting one-to-one grammar tutoring (tutor CALL), providing pedagogically appropriate error-specific feedback (ICALL), facilitating access to large lexicons/corpora (data-driven learning), and supporting interpersonal exchanges (computer-mediated communication). In particular, the application of advanced technologies to foreign language learning, such as natural language processing through ICALL systems, is widely discussed in the literature (Chapelle & Chung, 2010; Heift & Vyatkina, 2017; Holland et al., 1995; Ziegler et al., 2017).

Our previous work is dedicated to the design of such nontrivial computer-supported activities, focusing on the task of grammar acquisition (Purgina et al., 2020). Our language learning tool WordBricks implements the concept of "visual grammar", allowing the learner to explore grammatical structures, familiarise with new vocabulary and recognise correct syntactic patterns in sentences to support language learning. With WordBricks, we study how L2 grammar learning can be supported through visual activities that encourage the cognitive process of understanding and constructing the underlying syntax of natural language sentences. In the system interface, the learner can combine words, represented with shaped bricks, into phrases that are guaranteed to be syntactically correct. In this way, WordBricks helps learners to understand the logic behind grammar rules, to see how words are combined into phrases, and to figure out what kinds of sentence structures are allowed in the foreign language. The overall *capacity* of the system, i.e., the capability to address diverse linguistic phenomena, positive user experience, and impact on language acquisition, largely depends on the chosen "visual language" or the way we represent grammatical structures on the screen. From this experience we learnt that creating such a visual language representation is a very challenging task, and any possible solution is fraught with drawbacks and compromises.

The goal of this paper is to discuss the ideas and challenges associated with such a "visual language", to represent grammatical structures, that can be applied in computer-mediated grammar learning powered by natural language processing and AI methods (ICALL). Our contribution can be summarised two-fold: our work pushes forward the conceptual understanding of using technology to transform L2 learning practices. For this we expand the idea that visual representations of grammatical structures can serve as a foundation for novel educational activities and can enable learners to explore language in ways that are not feasible otherwise. In addition, our work offers methodological insights for designing and implementing an interactive L2 grammar learning tool, which can inform future efforts to translate theoretical models into usable educational technologies. We will discuss the advantages of having grammar rules represented in a graphical form, the limitations of our approach, possible solutions to unveiled issues, and open challenges. Our broader aim is to show how interactive software systems can be used to design new kinds of language learning activities, not achievable with traditional teaching methods, and to acknowledge significant difficulties of this research agenda.

2. Grammar Visualisation

The task of *grammar visualisation*, as understood in the present work, is to represent the underlying grammatical structure of sentences in a presumably clear and intuitive manner to facilitate efficient learning. Our assumption draws from education reports indicating that visual information can facilitate memory, recall and language learning (Halwani, 2017; Verdi et al., 1997). Grammar visualisation embodies, however, a well-known problem for learners (Park & Warschauer, 2016). Here we present several forms of grammar visualisation for natural language and make a comparison with the grammar of programming language, which inspired the development of our proposed prototype.

2.1. Natural Language Grammar

Grammar acquisition can be seen as an essential goal for effective L2 learning (Rutherford, 2014). Approaches to teaching grammatical structures vary significantly, depending on target learner level, desired degree of formality and specific pedagogical considerations (Donesch-Jezo, 2011; van Lommel et al., 2006). Some educators, such as Stephen Krashen (2003), believe that explicit grammar instruction is unnecessary and even harmful, since grammar acquisition should occur naturally as learners read and talk. However, this stand has risen strong debate and a contrasting research vein that presumes explicit grammar teaching, either as a dedicated activity or as certain bits of information integrated into other learning materials (Macaro & Masterman, 2006). Our approach is based on the assumption that certain grammar-oriented learning activities might be helpful for learners due to the importance of grammar in boosting written language skills (Kempen, 2004), and, therefore, our goal is to make them more efficient by providing additional technological support. The visual structures presented in this work can be potentially used both as extensions of the existing learning materials and as independent learning tools.

The principal challenge of learning grammar is to understand and internalise the rules and constraints governing sentence composition. Most learning materials explain them using both textual descriptions and certain visuals (tables, diagrams, sketches, also found in interactive websites[1]) to facilitate learning. In actual learning practice, grammar rules seem to be often represented with short

[1] See for example, English Grammar Profile (https://www.englishprofile.org/component/grammar), Pearson's Teacher Toolkit (https://www.english.com/gse/teacher-toolkit/user/grammar) and Visual Interactive Syntax Learning (VISL) project, a Danish initiative providing visualization tools and games in several languages (https://visl.sdu.dk/).

textual descriptions and visuals, such as tables and “formulas”. For example, the rules explaining the use of simple past tense in English are described by Azar and Hagen (2005) as shown in Figure 1.

Simple present	(a) I ***walk*** to school ***every day***.	verb + ***-ed*** = simple past tense
Simple past	(b) I ***walked*** to school ***yesterday***.	I
		you
Simple present	(c) Ann ***walks*** to school ***every day***.	she / he / it } + *walked (verb +* ***-ed****)*
Simple past	(d) Ann ***walked*** to school ***yesterday***.	we
		they

Figure 1. The simple past tense: using ***-ed****.*

This approach provides great flexibility in describing grammatical phenomena and adjusting explanations to the target learner level. When designing such tables, it is possible to decide what to mention, what to omit, and what kind of rules to consider relevant for the given topic. For instance, Azar and Hagen (2005) also explain in their grammar tables, alongside grammar rules, how to pronounce *-ed* verb endings, even though pronunciation rules belong to phonetics rather than grammar. Unlike textbooks, theoretical academic works on language grammar often use formalised diagrams to describe sentence structure (Mitchell, 1994). In most widespread models, such as phrase structure grammar (Frank, 2004; Gazdar, 1982) and dependency grammar (Bick, 2004; Debusmann, 2000; Nivre, 2003), parsed sentences are usually represented with directed graphs[2], and there are *de facto* established ways to visualise these graphs (see Figure 2).

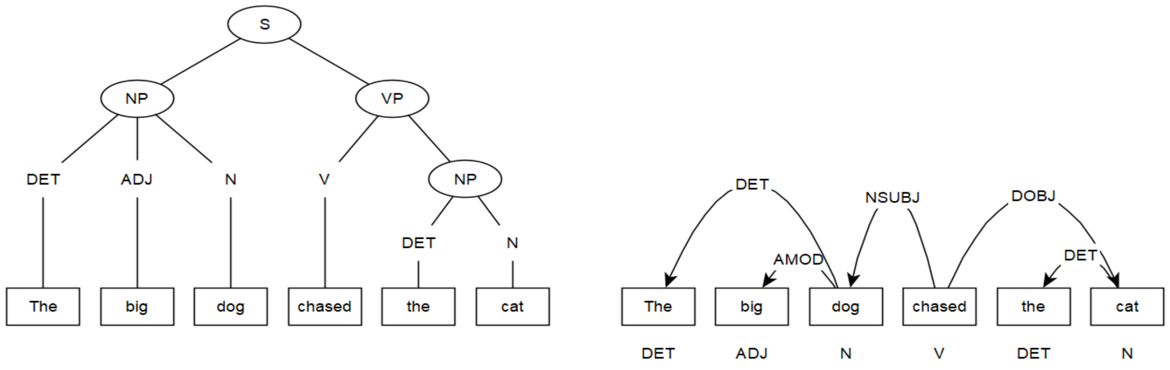
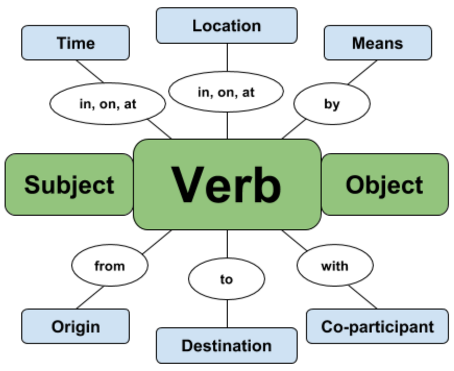


Figure 2. Left) phrase structure tree indicating the function of each word in the sentence: DET determiner, ADJ adjective, N noun, V verb; these parts form the NP noun phrase and VP verb phrase, which form the S sentence. Right) dependency tree of the same sentence indicating how each word is related to one another: ‘The’ determines (DET) ‘dog’ and ‘cat’; ‘big’ modifies (AMOD)’dog’; ‘dog’ is the subject of the verb (NSUBJ) ‘chase’ and ‘cat’ is its direct object (DOBJ).

It is difficult to say why learning materials rarely make use of such graphs or diagrams. It is possible that educators do not see much pedagogical value in them, have difficulties using them in a meaningful way, or are simply satisfied with their current practices. Some authors, however, do rely on structural diagrams to illustrate grammar phenomena. For example, Webb (2019) uses charts to explain structural differences between English and Japanese sentences. The chart shown in Figure 3 illustrates the idea that English sentences generally follow the subject-verb-object (SVO) structure, while Japanese sentences consist of particle-followed elements placed before the main verb in no strictly defined order. Other formalisms, such as Diderichsen’s schema (Diderichsen, 1946), were also proposed to address flexibility of language structures.

[2] Some authors suggest constructing trees with binary branching only (Guevara, 2007; Kayne, 1984); we do not impose such restrictions.



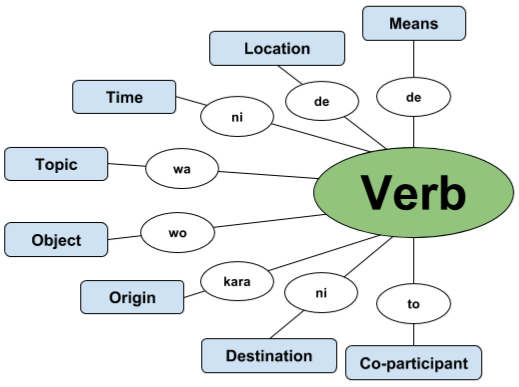


Figure 3. Sentence structure charts of (left) English and (right) Japanese (reprinted with permission from (Webb, 2019)). The English sentence structure generally follows the formula subject-verb-object, whereas the Japanese sentence structure emphasises the verb that is preceded by particles that specify the function of the words that they are accompanying.

Another interesting case is reported by Ebbels (2007). Her work introduces a specialised graphical language (called "shape coding system") for representing sentence structure with diagrams, as shown in Figure 4. The shapes and colours of individual elements in these diagrams are well defined: e.g., ovals represent subjects, rectangles represent objects, hexagons correspond to verb phrases, and semi-circles are reserved for prepositional phrases. In contrast to the English/Japanese chart (Figure 3), shape coding system is used to visualise specific phrases rather than generalized grammatical structures. However, the goal of Ebbels's visuals is to explain grammar by example rather than to analyse the structure of a particular sentence. Ebbels argues that shape coding system helps to teach grammar to schoolchildren with specific language impairments more efficiently. However, it is unclear whether this method would have a positive impact in a conventional classroom.

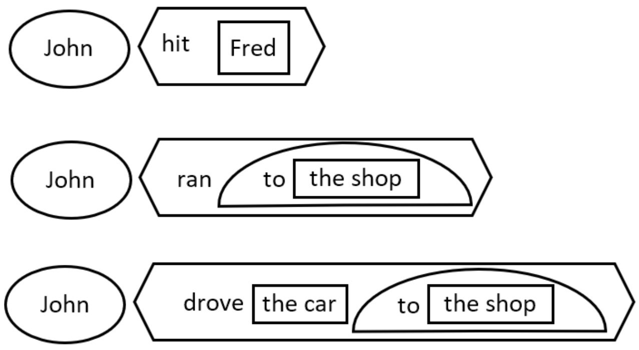
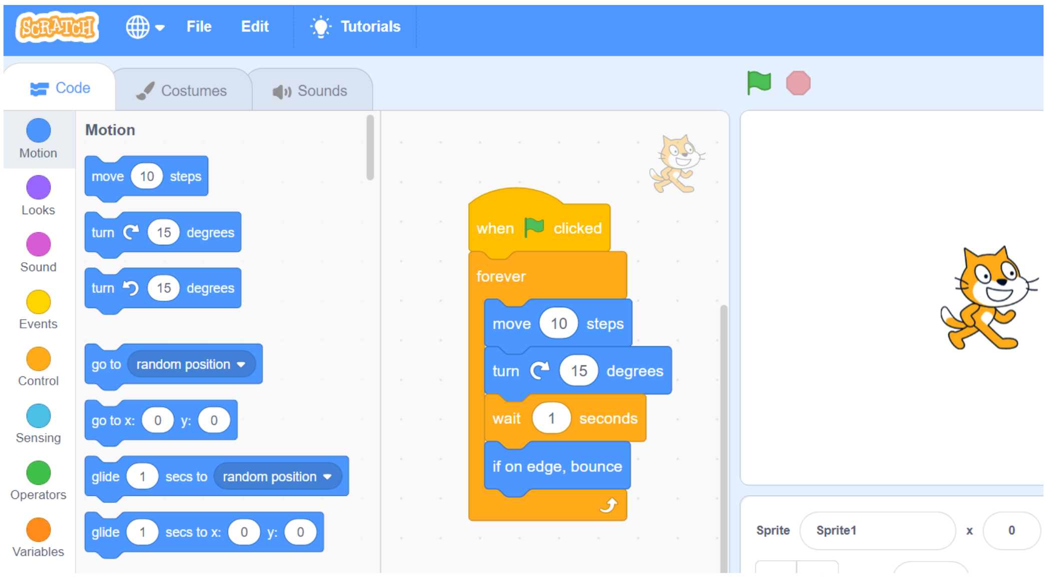


Figure 4. Sentence structure representation with Ebbels's shape coding system: ovals represent subjects ('John'), rectangles represent objects ('the shop'), semi-circles represent prepositional phrases ('to the shop'), hexagons represent verb phrases ('ran to the shop') (adopted from (Ebbels, 2007, p. 72))

2.2. Grammar of Programming Languages

Arguably, the main challenges of programming languages learning are not the same as in case of L2 acquisition. Nevertheless, educators in both contexts must deal with the same topics of syntax, idiomatic expressions, grammar, and so on (Alexandru et al., 2018; Cordy, 2004; Schmidt, 1996; Stefik & Siebert, 2013). These issues are much less pronounced in programming language education because programming languages are specifically designed to be logical, consistent, and simple (whenever possible). Yet, while the formal aspects of programming languages such as syntax and grammar are typically not viewed as especially challenging, many beginners still struggle with them (Kelleher & Pausch, 2005). Therefore, considerable efforts are invested into lowering these entry barriers. One very popular idea of reducing initial complexity of programming is to introduce a specially designed "visual language" consisting of shaped blocks, flowcharts or similar visual elements (Bau et al., 2017; Cook, 2015).

One may ask, *what exactly makes visual programming environments easy for learners?* Among the reasons identified by Bau et al. (2017), two are directly relevant to our subsequent discussion. First, within visual programming environment, such as Scratch (Dorling & White, 2015), the choice of visual blocks is predefined, and one picks a required block from a palette rather than reproduces a certain expression from memory. Such recognition activity imposes a lesser cognitive burden than the one required to recall and type a textual pattern. Second, blocks make the grammar of a programming language simpler, explicit, and visible. Subsequently, block shapes and connectors restrict the manipulations with the code to ensure that only syntactically correct programs are produced (see Figure 5). We could say that this kind of *visual grammar* for programming extends the basic idea of a typical graphical user interface by reducing the cognitive load of a learner during a programming task.



```
when 🏳 clicked
forever
    move 10 steps
    turn ↻ 15 degrees
    wait 1 seconds
    if on edge, bounce
```

Figure 5. Writing programs in Scratch.

An interesting learning theory lens through which we can analyse visual grammars is the widely accepted constructivist theory of education, which suggests that learners actively "construct" knowledge rather than passively acquire it (Hein, 1991). This implies that each learner builds a somewhat unique mental model of a concept he or she acquires, largely on the basis of the previous knowledge and experiences of this particular individual (Vygotsky, 1980, pp. 57, 85). Initially, such mental models are often crude and inaccurate, but they are gradually refined due to exposure to new information. Erroneous understanding of complex concepts is not foreign to this process, so it has been suggested that active measures for developing reasonable models of certain concepts (such as "a processor" or "a memory unit" in case of computer science education) should be taken (Ben-Ari, 2001). Thus, educators must deal not only with the concepts they teach, but also with possible ways the learners interpret (*construe*) them.

One way to take a proactive stance in this process is to provide a *construal* to a learner, as noted by Thompson (2018): *"(t)he objective in creating a construal is to enable the learner to observe the construal's internal workings and to form an understanding through experimentation. <…> The learner constructs understanding through interaction with a model, a construal, that aids them in developing their understanding the phenomena being studied"*. A system of visual blocks serves as such a "construal", helping the learner to build a consistent mental model of a programming language structure, of grammatically correct expressions and the logic behind them. In visual programming language teaching, a common point of concern is knowledge transfer, i.e., the applicability of acquired mental models in the context of conventional text-based languages, such as Python or Java. Classroom studies show that this transition can be performed smoothly, and there is no reason to avoid visual programming instruments (Dorling & White, 2015).

3. WordBricks: Goals and Challenges

The idea of representing programming language grammar with shaped bricks, alongside its benefits outlined above, inspired us to apply the same approach to the context of foreign language education. Our primary point of reference is Scratch programming environment (Dorling & White, 2015), and the main research goal is to find out whether such a system can indeed serve as facilitator of mental models for beginner language learners. Revisiting the question why learning materials rarely use structural diagrams, we can further observe that no visuals, even the most revealing, can serve as a proper construal due to its *non-interactive nature*. A visual may help to understand some of the logic behind a language grammar, but it cannot facilitate experiments with grammatical structures. A learner needs a certain interactive tool to be able, for instance, freely arrange and rearrange word to study and understand which combinations are grammatically correct.

Let us briefly state some considerations that formed our initial vision of WordBricks and thus affected its current shape and functionality. A more detailed account can be found in (Purgina et al., 2020).

We wanted to explore further the capability of visual programming environments, such as Scratch, to produce *syntactically correct* programs only. As mentioned above, programs in Scratch are produced by connecting shaped blocks. These blocks are dragged by the user from a predefined palette and dropped to a working area. The system of shapes and shaped connectors in Scratch is designed in such a way that only legal (i.e., syntactically correct) expressions can be constructed. For example, the diamond-shaped block "and" contains exactly two diamond-shaped connectors. A diamond shape codifies a Boolean (logical true/false) expression, so it is only possible to combine two Boolean expressions with an "`and`" operator, e.g., "`Score > 5 and Level = 1`" (see Figure 6). The result, in turn, will also be a Boolean expression, making it possible to construct longer expressions like "`Score > 5 and Level = 1 and Lives < 3`". In contrast, the "`not`" block has only one connector, so a learner can produce a legal construction like "`not Level = 1`", while illegal constructions like "`Score > 5 not Level = 1`" cannot be created.

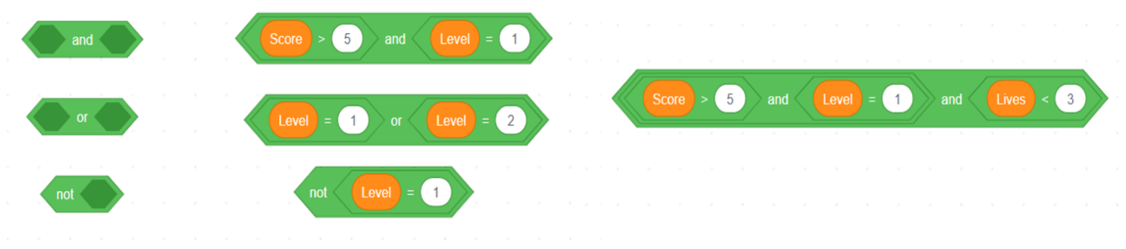


Figure 6. Boolean expressions in Scratch.

This capability of Scratch is noted, but somewhat downplayed in literature, because the primary goal of the learners is to produce code that can do something interesting rather than just be syntactically correct. Thus, most discussion is focused on the semantics of Scratch programs, while the capacity to avoid syntax errors is seen merely as a way to lower the entry barrier to computer programming (Bau et al., 2017).

Our first goal was to study the applicability of the same approach to natural language learning, using computers to extend the way of teaching and learning foreign language grammar. We wondered whether it would be possible to design a *visual natural language grammar* enabling the learners to produce syntactically correct phrases only. While a natural language is much more complex than any programming language, its grammar is still a formal system, having specific well-defined properties (Chomsky, 1965). Therefore, we believe it should be possible to recreate a Scratch-like experience for it. In contrast to Scratch, we treated the capability to produce grammatical phrases as the primary functionality of WordBricks. The issues of style, tone, sentence logic and other higher-level problems the learners face are beyond the scope of our project.

Since we wanted to create a practical tool, suitable both for the classroom and for self-study, we gathered the opinions of educators and learners to build our early prototype and performed some tests in a classroom environment. Early prototypes were tested in distinct settings of a primary school in Ireland (44 students in 5th grade, and 75 students in 3rd grade) and a computer science university in Japan (10 second-year undergraduate students) to understand the spectrum of possible needs and requirements at different educational levels (Ward et al., 2019; Purgina et al., 2017). From this experience we learnt that our respondents had strikingly different opinions on what the primary focus of WordBricks should be:

- Technological:
  - Desktop *and* mobile versions of the system with interfaces tailored for these platforms.
  - Extended gamification features (e.g., levels/stars/scores and other means to engage the user).
  - Online updates to enable quick distribution of new materials.
- Pedagogical:
  - "Restricted" *and* "unrestricted" modes designed either to force the learners to use specific language structures studied in a particular unit, or to let them explore the language freely.
  - Integration of textbook-like learning materials into the system.
  - Extended feedback on user errors.
  - "Demonstration mode" for educators to illustrate key grammatical concepts in the classroom.
- Personalised to individual needs:
  - Support for different foreign languages.
  - "Natural" look of resulting structures enabling easy transition from WordBricks to real texts.

Taking these expressed needs as user requirements and seeing them through the lens of a constructivist learning theory, we iterated through several WordBricks protypes (Sec. **Error! Reference source not found.**), each with its own set of design compromises (Sec. 5) and pedagogical implications (Sec. 6).

4. WordBricks Visual Grammar Approach

4.1. Early Prototype

To illustrate the issues one has to deal with when designing a "visual grammar", let us first consider the approach used in an early WordBricks prototype (shown in the Figure 7).

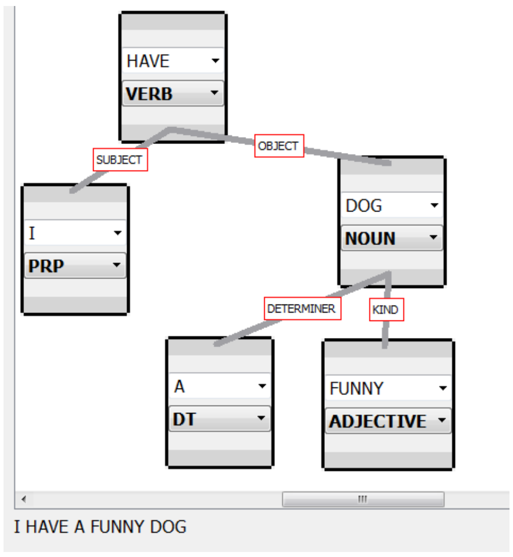


Figure 7. WordBricks early prototype version showing a tree structure with the function of each word in the sentence

In this version, by choosing "New Brick" menu option, the user creates an empty rectangular brick on the canvas. Bricks can be arranged by dragging them with a mouse on the screen. Each brick has an input box where the user can type any word (such as "I" or "DOG"). If the word is recognised by the integrated morphological analyser AOT (Sokirko, 2004), its part of speech is displayed underneath. The user can also choose a different part of speech if multiple options are available (e.g., a word can be either a noun or a verb), and switch between word forms of the same word (e.g., "DOG" can be changed into "DOGS" by using a drop-down menu). This kind of interactive syntax modality is also found in other systems in the CALL literature, including Ljunglöf syntax tree editing system (Ljunglöf, 2011) and MULLE tool by Lange and Ljunglöf (2018). There are also similarities with COMPASS system by Harbusch et al. (2007), which is based on phrase structure rather than dependency grammar visualization. Our system concept design for this prototype differs, however, in that the user is free to enter any text into the prototype's interface and analyse the resulting grammar structure. We consider that in this way the user is better empowered to explore their grammatical knowledge without constraints.

In our early prototype, bricks arrangement across the horizontal axis forms the resulting sentence. In the example shown in Figure 7, the resulting structure is "I have a funny dog" because "I" is the leftmost brick, "DOG" is the rightmost brick, and all other bricks are located between them. Finally, the user can connect a pair of bricks with a line. If such connection is deemed valid by the system, a connection type ("subject", "object", etc.) is displayed. Otherwise, the connection is shown as greyed out. Connection type is decided based on word compatibility according to a handcrafted set of grammar rules. However simple, this early prototype has certain practical merits. It allows the user to freely experiment with grammar, rearranging words and linking different bricks to see what kinds of connection can be established. In addition, its integrated morphological analyser helps to explore available options. For example, when the user fails to link the word "*she*" as an object, there is always a chance to try out other word forms, including the correct type "*her*".

Nevertheless, this early prototype did not reach a formal testing stage since preliminary demonstrations to a small group of undergraduate students provided sufficient insight to determine that the approach was unlikely to yield fruitful results. The largest difficulty for the user in this early system implementation was to understand the principles of brick linkage as there is no scaffolding in forming syntactically correct sentences. Why, for instance, "funny" should be connected to "dog"? Commonly used in theoretical works (see Figure 2), tree-like grammar structures are not obvious for beginners. Furthermore, it was far from being obvious whether the learners needed to understand parse trees, and which linguistic theory should be used to form them. (See, for instance, (Vinther, 2004) for a discussion on the use of parsers as pedagogical tools for L2 instruction). Another obvious downside of this prototype was the identical outlook of all bricks in the system.

This outcome informed the next developmental iteration of our system. In the subsequent revisions, we had to disable the free user input functionality. Although this functionality worked well with generic brick shapes and links, a more elaborate system of shapes and connectors was needed to visually emulate the richness of grammatical structures. This elaborated system was introduced in the updated prototype, for which more advanced text processing capabilities were required. The free input functionality will likely be reintroduced into the system for future research.

4.2. Scratch-inspired Prototype

For the next version of the system, we agreed on few basic principles to guide our design and development process:

An established grammatical theory should be used to determine brick linkage principles. The process of selecting a suitable grammar framework is important within ICALL (Schulze & Penner, 2008). We decided to follow the rules of dependency grammar (Debusmann, 2000) primarily due to the simplicity of their visual representation: words are linked directly with other words, and there are no additional structures that might clutter the working area. The principles of word linkage are clearly described in literature (Marneffe & Manning, 2008), and their logic is easy to follow in most cases. Sometimes we simplify the resulting structure by providing "compound" bricks representing logically inseparable syntactical constructions (called "*nuclei*" in Tesnière's original work on dependency grammars (Tesnière, 2015)). For example, we might provide a brick "`has been doing`" instead of forcing the user to combine the separate bricks "`has`", "`been`", and "`doing`" into a single structure.

Bricks should be shaped in a way that allows the user to see possible connections immediately. There should be no "what-if" scenarios where it is necessary to try to establish a connection to see whether it is admissible or not. This supports the visual "descriptive power" of the grammatical framework we selected (Schulze & Penner, 2008)

The resulting structure should be clearly readable as a sentence. Words should be arranged in a left-to-right order, to support the learner's grammar acquisition (Schulze & Penner, 2008). Two-dimensional arrangements (left-to-right, top-to-bottom), similar to book pages, are also allowed.

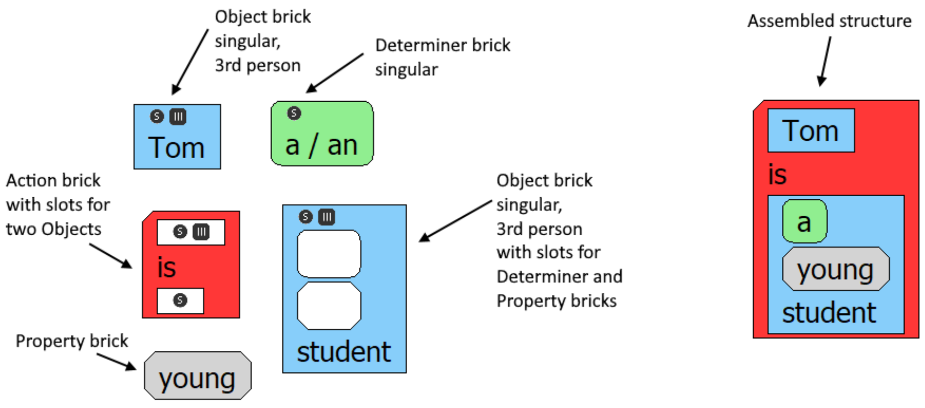


Figure 8. Explicit representation of grammar rules with brick shapes in the WordBricks Scratch-inspired prototype version

Adherence to these principles should produce sentences that look more like Scratch programs. Bricks arrangements in Scratch can be read just like real computer programs, left to right, top to bottom, and the connectors of each brick are clearly visible, so the learner does not have to link two bricks to see whether they match. In other words, this approach can be seen as an experiment in the most explicit way of visualising language structures, where all relevant grammar phenomena are encoded in brick shapes, colours, and connectors (see Figure 8). If we treat it as a case study rather than a practical system, it can let us see the implications our decisions have for actual use and learning, so that we can address identified issues in subsequent solutions.

The requirement of "natural" sentence arrangement in combination with the reliance on dependency grammar is not always possible to meet, we learnt. Scratch-like bricks are strictly hierarchical: if two elements represent a "head-dependent" relationship, they should physically be arranged within a single graphical entity. For instance, if a word w_1 appears before w_2 in a sentence, all dependent words of w_1 will also appear before all dependent words of w_2. It means that Scratch-like visualisation can only represent *projective* dependencies. To give an example, let us consider a sentence "`A hearing on the issue is scheduled today.`" Here "`hearing`" is a head word for the structure "`on the issue`", and thus "`on the issue`" will have to be inserted into a certain connector of "`hearing`". However, if we rearrange the sentence as "`A hearing is scheduled on the issue today`", it would be impossible to represent it in a Scratch-like format. Such non-projective dependencies are not common in English, but appear in 25-27% of structures in languages like Czech or German (Havelka, 2007).

More apparent issues were related to high flexibility of grammar rules that exist in natural languages. It turned out that nearly every single word might appear in contexts requiring distinct shape/colour encoding. For example, in English many nouns, e.g., "`book`", can also serve as verbs and adjectives (or modifiers). If we have separate brick shapes for nouns, verbs, and adjectives, we will have to create three separate bricks for every such word. Connectors are also flexible: the structure "`I am`" may continue with a noun ("`a learner`") or an adjective ("`young`"). Therefore, a placeholder for the object of the verb "am" can be both a noun-shaped and an adjective-shaped. Furthermore, part of speech is not the only linguistic attribute that has to be encoded in a connector: for example, a possible subject of the verb "have" is a personal pronoun in a non-singular, 3rd-person form.

Thus, an attempt to represent every possible shape-connector combination of a word with a separate brick on a palette would produce a long and confusing set of alternatives. The user will not be able to simply drag a word into a working area; instead, it will be necessary to choose a specific brick associated with the word, required in the given context. It is also worth mentioning that incompatible connectors should have different visual representations. For instance, a connector for singular 3rd-person pronoun should not look exactly like the one for plural 2nd-person pronoun, and so on. Given the abundance of grammatical attributes and their combinations, a visual system might require an impractically large amount of distinct connector shapes, even for a relatively small number of unique words.

It is interesting to note that Scratch developers apparently faced the same difficulties, albeit to a much lesser extent. For example, the "if" construction of Scratch programming language may have an optional "else" clause, meaning that the set of connectors of "if" is not fixed. Scratch developers opted for two independent blocks in this case (see Figure 9, left). The system of Scratch types is also not perfectly described with connector shapes. Scratch has a dedicated diamond-shaped connector type for Boolean expressions (it can be seen in an "if" block). However, a Boolean expression can also be assigned to any variable, i.e., inserted into a round-shaped connector (see Figure 9, right). This operation is valid since Scratch types are implicitly convertible to each other; still, connector shape does not match the block shape to be inserted, which might be confusing for the user.

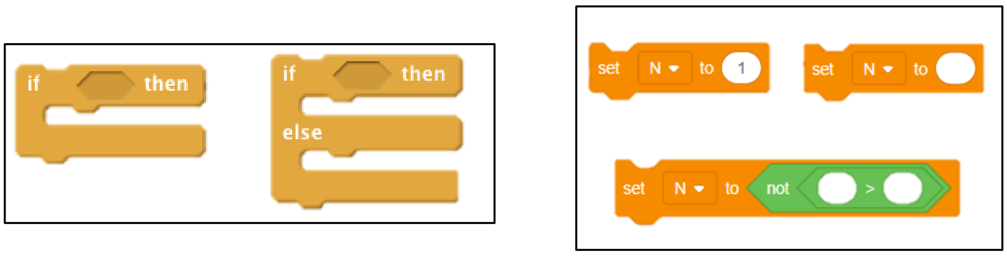


Figure 9. Design decisions compromises found in Scratch – the hexagonal connector with a Boolean expression can be inserted in a round-shaped connector.

It was quite clear from the experiences and lessons learnt with the Scratch-inspired prototype that strict adherence to the principles of strong typing and graphical representation of distinct grammar structures led to overly complicated blocks and required considerable investments into user interface design, all of which likely resulted in increased users' cognitive load that we observed when the users interacted with the system (Purgina et al., 2017). This would defy the goal of facilitating the learning of natural language grammar. We also understood that even in the case of a simple programming language it is reasonable to cut some corners and provide a solution that visualises certain aspects of the grammar while encoding others in a non-visual form. These insights led us to the next developmental iteration of the system, a mobile prototype.

4.3. Mobile Prototype

When producing a mobile application, where screen size does not allow to show much auxiliary information, uncluttering the visualisation of grammar items becomes fundamental. Therefore, the design of visual elements (including bricks) must be simplified, which is a common strategy in mobile development. For our prototype we decided to bind bricks' shapes and their corresponding connectors to universal coarse part of speech tags (Petrov et al., 2012). If particles, punctuation marks and unknown/foreign words are excluded, only ten different word types remain: nouns, verbs, adjectives, adverbs, pronouns, determiners and articles, prepositions, postpositions, numerals, and conjunctions. Connector sets are predefined for each brick, so each word might still have several associated bricks, having differences in shapes and associated connectors.

These modifications allow to build a much cleaner user interface (see Figure 10). However, this prototype loses one of the primary features of the previous version: shapes no longer represent the full set of meaningful word attributes. For instance, both bricks "`I`" and "`he`" have the same shape, matching the subject connector of the brick "`has`". Thus, the user has all reasons to presume that these bricks can be connected to each other. To scaffold the learning in this protype, when such an attempt is made, we show an error message explaining the problem. Error messages are not adaptive, but can be potentially extended with levels of mediation, corresponding to student needs (such levelled feedback is discussed, e.g., by Nicholas et al. (2024)). The system also provides another optional capability: once the user starts dragging a brick, all compatible connectors are highlighted. This immediate feedback mechanism makes it instantly clear what kind of constructions are allowed in the current situation, working also as a scaffolding strategy, which is important "in enhancing the benefits of CALL" (Rezaee et al., 2015).

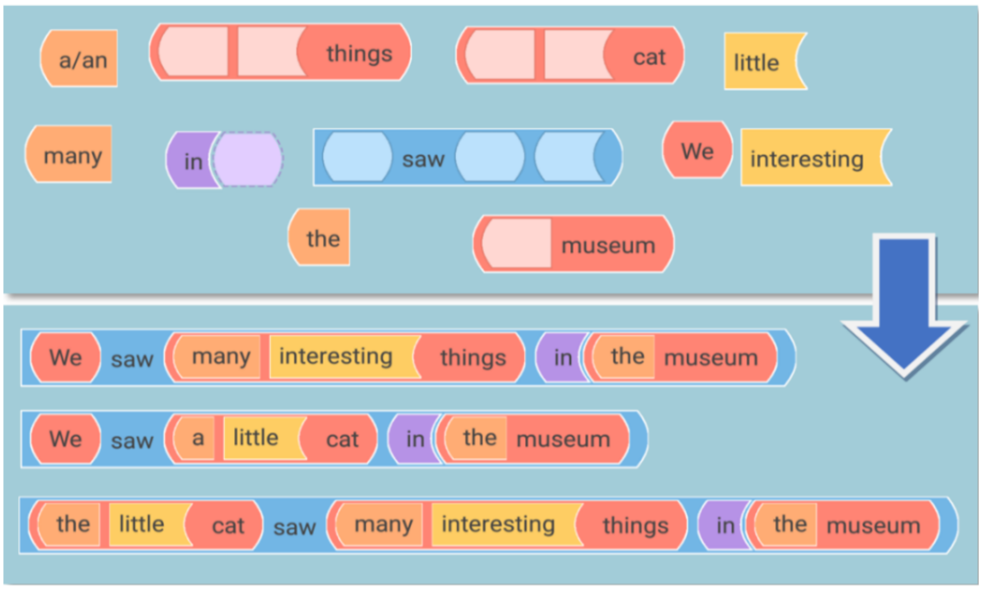

a/an
things
cat
little
many
in
saw
We
interesting
the
museum
We saw many interesting things in the museum
We saw a little cat in the museum
the little cat saw many interesting things in the museum

Figure 10. WordBricks mobile device prototype version – uncluttered facilitation of syntax exploration.

The mobile version of the system was positively evaluated both by students and teachers (Purgina et al., 2020). Its simplified interface, originally designed to overcome the limitations of small mobile screens, provides intuitive and streamlined user experience. These benefits, however, come at a cost of reduced visual clues. The students do not always see that two bricks cannot be linked together, receiving feedback only after attempting to form a phrase.

5. Discussion on Development Challenges

We can see from the experiences with the experimental WordBricks prototypes that designing a consistent and user-friendly visual representation of a natural language grammar is not an easy task. Every decision has a flip side: a parse tree-like representation does not help the user to identify word order and word relations; a Scratch-like representation is not suitable for a mobile device, does not support non-projective constructions, and provides excessive details that increase the cognitive load of the user; while a cleaner mobile visualisation hides important word properties thus making word linking rules opaque.

The mentioned difficulties are well known to the authors of visual programming languages. Bau et al. (2017) summarise them as an answer to the question *"why don't professionals program with block interfaces?"* In brief, the following points are mentioned:

A) High *viscosity*: even small edits of the program are hard to make, as they may require rearrangement of several blocks.

B) Low *density*: blocks take more space than text, thus only a small part of the program is visible on the screen.
C) Search and navigation in a visual program are challenging (palette navigation is an issue, too).
D) Source control is difficult to organise.

Clearly, the issues of viscosity, density, and search/navigation are relevant for our case as well (the fourth issue of *source code versioning* is not applicable here). The same work (Bau et al., 2017) examines existing approaches to address some of these disadvantages. In a nutshell, most improvements are based on the idea of a "hybrid" visual/text interface. It can be either a "text block input mode" that allows to retrieve a block from the palette by typing its contents on the keyboard, or a parallel text/block representation of the program. In this mode the user types the code as a text, but sees its visualisation in real time, which helps to spot structural errors immediately.

The idea of building a graphical representation of a user-typed sentence can be also useful in language learning, and one of WordBricks versions, in fact, supported this (Purgina & Mozgovoy, 2017). However, we can conclude that there is no established method to resolve the mentioned challenges, and the only viable approach is to implement partial solutions, improving the overall user experience based on the pedagogical aims to achieve. Let us outline some available options:

A) **Designing a system of brick and connector shapes.** As discussed above, in the ideal case, the look of the brick should determine the list of matching connectors, enabling the user to judge which constructions are considered legal. At the same time, it is desirable to minimise the number of graphical elements both to save screen space (which is especially important for a mobile app) and to reduce visual clutter. Reducing clutter is not just an aesthetic goal: a clean interface allows the users focus on their primary learning goals rather than on visual elements and their underlying meaning. Probably, the only visual means that do not consume space are shape and colour. In practice, most visual systems either use only one of these instruments or bind a specific colour to a specific shape, so that if there is a *yellow diamond* in the system, there will be no *blue diamond* or *green diamond* (which removes obstacles for colour-blind users). Since each brick in the system must have a certain shape and colour, it is only possible to connect shapes/colours with a certain grammatical attribute that exists in every possible word. A natural choice would be either a part of speech (noun, verb, etc.) or a grammatical role (subject, object, etc.). Other attributes (number, person, tense, gender, etc.) must be represented with optional visual elements. We made such an attempt in our Scratch-inspired prototype. Figure 8 shows how word number (singular/plural) is shown with a circular icon, while word person ($1^{st}/2^{nd}/3^{rd}$) is shown with a rectangular icon. Each connector can also be accompanied with optional icon sets, representing compatible dependent words. Unfortunately, such icon-based coding system can become complex and unwieldy in practice.

B) **Supporting brick variations and brick search capability.** Dealing with numerous word bricks, having many variations and different shapes and connectors, can be challenging for the user. A simple word such as "book" can be present in the system as a noun, an adjective, and a verb, and even within a specific part of speech, variations in connectors can be considerable (e.g., "green book" vs. "book of recipes"). Placing all possible brick variations on the same palette is not scalable. While there are no perfect solutions for this problem, some variation of a hybrid visual/text input is usually implemented (Bau et al., 2017). The user could filter down the list of options by typing a word or a word part, corresponding to the required brick. The choice between connector sets is trickier, though, because not all of them are equally common, and too many available options might be confusing. One possible solution is implemented in Scratch, where all unary mathematical operators are contained inside a single block "`abs`". If the user needs, for example, a square root function, it is necessary to place the block "`abs`" in the working area, and then switch it to "`sqrt`" (see Figure 11).

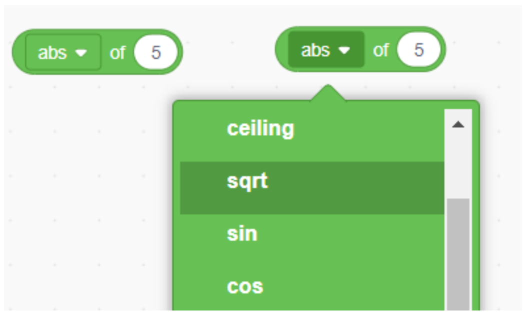


Figure 11. Configurable blocks in Scratch.

C) **Handling non-projective structures.** We consider support to non-projective structures to be a feature of low importance for our system. Even in languages other than English, non-projective structures are typically found in relatively advanced grammatical structures, not very relevant for beginners. Nevertheless, one possible way to handle them is to switch to a *node-based* rather than *block-based* visual representation. This approach is widely used in a variety of professional process modelling systems, such as Simulink or Orange. It can also be found in LabVIEW for LEGO Mindstorms, a visual programming language for designing computer-controlled LEGO robots (see Figure 12). This system combines sequential block linking with the capability of connecting any compatible blocks with "wires". Such "wire" connections can be also added to the conventional Scratch-like interface.

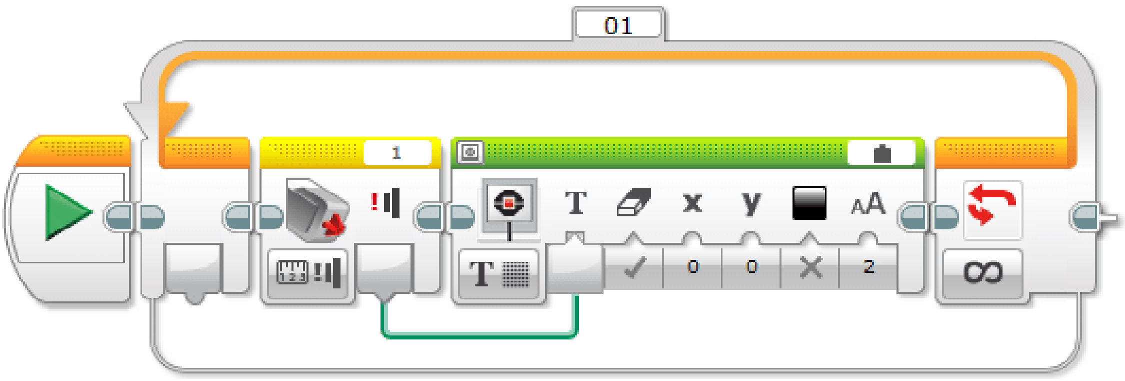


Figure 12. "Wire" connections in LabVIEW interface for programing LEGO robots. Taking this as inspiration, sequential blocks linking could be used to handle visual grammar representation of non-projective structures

Summing up, educational tools such as WordBricks can be further developed in a number of ways, reflecting various practical use scenarios. However, the primary challenges might still relate to the original task of *designing a reasonable visual representation of a natural language grammar.*

6. Discussion on Linguistic and Pedagogical Challenges

The challenges outlined in the previous section are mostly reduced to the task of striking the right balance between the functionality, usability, and the amount of information shown in the interface. There is no "right" solution because this balance depends on the system's platform, learner's level, target language, specific pedagogical goals, and other similar factors (Slavuj et al., 2017). While experimenting with our system, we faced difficulties in terms of the linguistic representation and pedagogical aspects that the system could support. These difficulties are even harder than the functionality and usability challenges already discussed, since "striking the right balance" approach is not sufficient here. Some of them can still be resolved with partial solutions, while others require additional investigation. Let us outline these challenges.

Linguistic and grammatical perspectives. Probably, the biggest challenges are connected to the fact that every formal system of rules (such as Scratch or WordBricks) can only ensure compliance with these rules, while the goal of L2 acquisition or developing programming skills is much wider and more nuanced. There is a traditional classification of language phenomena into "phonetics", "morphology", "grammar", "semantics" and so on, but in real pedagogical practice the borders between these areas are very blurry. This is not a significant issue for the teachers: while working on a particular aspect of grammar, for example, a teacher can briefly explain certain semantic subtlety or correct learner's pronunciation. However, within a formal system we have to clearly identify which phenomena should be addressed by the system, and which should be considered outside the scope. Poor choice might make the system nearly useless if both the teachers and the learners see how many faulty sentences it allows to produce, and how many correct sentences are not allowed. Choosing the scope is especially challenging because not all linguistic phenomena are handled equally well with today's technology (Schulze & Penner, 2008). For example, spelling and morphological analysis is handled more reliably than semantics and style.

One simple example of a choice to be made in terms of scope is shown in the Figure 10. In English, the use of an indefinite article requires the choice between one of its two forms (a / an). Should WordBricks provide separate bricks for "`a`" and "`an`", and how their shapes should look like? We decided to keep one brick "`a / an`", which reflects the identical grammatical properties of both forms. However, once the brick is linked to another brick, the system automatically displays the appropriate form according to the next word in the sentence. Another relatively simple predicament occurs when certain words comprise a semantic unity, but their internal relations might be confusing for the learner, such as proper names (*John Doe*) or verb tenses formed with auxiliary words (*have been doing*), for instance. Such entities, corresponding to Tesnière's nuclei (Tesnière, 2015, p. 46), can be directly represented with individual bricks, and there is little pedagogical reason to separate them into words. Hence, for the learner it means that they do not need to think how to connect *John* with *Doe*, because a "*John Doe*" brick is provided.

Pedagogical perspectives. A more challenging task is to decide the amount of information needed for the learners of the given level and stage. Depending on the pedagogical goals, grammar books deliberately simplify explanations for the sake of beginners' needs, but it is hard to simplify bricks without undesirable consequences (e.g., allowing to create ungrammatical sentences), so certain trade-offs are inevitable as we have observed during the transition from early prototype to Scratch-inspired prototype. For example, in English the same noun form can be freely used as a subject or an object. Thus, it is tempting to utilise the same connector type for these roles. In contrast, pronouns do not work in this way: one cannot use "he" or "she" as objects; an object case ("him", "her") is required. The most logical approach would be to distinguish nominative and object cases and to provide different connector types for them. However, it means that every noun must be represented with two bricks: a "nominative case noun" brick and an "object case noun" brick. This can be confusing both in terms of user interface, and in terms of the brick-encoded grammar: beginners, most likely, do not need knowledge of such subtleties at the early stage of their learning. One possible way to remedy this situation is to introduce an *auto-conversion mechanism*, allowing the brick to be converted to a certain related brick with desired properties. This would allow the users to modify grammatical attributes of words using WordBricks' interface (e.g., forming a plural form or a certain verb tense of the given brick). It might be especially helpful for languages with rich morphology, where producing a correct word form is a challenge itself (e.g., in agglutinative languages such as Finnish (Bertram et al., 2000)).

When transitioning from a Scratch-inspired prototype to a mobile prototype we could also perceive pedagogical trade-offs: from a richer set of shapes to represent the full set of meaningful word attributes to a streamlined mobile-friendly design that avoids cognitive overload. Yet, this simplification may limit learners' ability to visually grasp the complexity and richness of grammatical structures, potentially reducing opportunities for deeper conceptual engagement with the language. On the other hand, improved usability and reduced cognitive load can be especially important for younger learners when accessing the tool on smaller screens. The choice then reflects the tension between instruction goals and technology affordance especially when adapting the Anonymous tool for different platforms and learner contexts. Akin to choosing a suitable grammar book, this choice would be based on learners' needs (e.g., level and ability), learning objectives, and instructional approach, for instance.

Another important aspect to consider when developing visual grammar systems for language learning is the amount and timeliness of the feedback provided (Choi, 2016; Heift & Hegelheimer, 2017; Lavolette et al., 2015). Consider, for example, the problem of distinguishing the sentences *making no sense* from *ungrammatical* sentences. In computer programming languages this border is very well defined: Scratch programs are considered syntactically correct, and if the output is wrong, it should be attributed to flaws at the semantics level. In natural language, a sentence like *I will arrive there yesterday* is syntactically correct, since *yesterday* is an adverb, which can be used as a verb modifier. Being correct, it, however, makes no sense. Should such a sentence be deemed correctly by a system like WordBricks? How could the system provide active feedback to the user? These questions become especially complicated if we observe that many entry-level grammar books list certain restrictions as if they were grammar rules: for instance, *"we do not use* ***the*** *with names of people ('Helen', 'Helen Taylor', etc.)"* (Murphy, 2012). However, more advanced guides provide examples where such usage is admissible: *"that's not* ***the*** *Stephen Fraser I went to school with"* (Hewings, 2013). Similarly, entry-level grammar books state that words like *yet*, *already*, and *just* are used with the present perfect tense, and thus can help to choose between the past and the present perfect tenses (Murphy, 2012). However, these rules reflect the semantics of verb tenses and usage of words like *just* rather than syntactic requirements.

The rationale for this choice becomes clear if we consider that it is very unlikely that a beginner would have a practical need to use *the* with a personal name, so it is safer to state that such usage is not allowed. Similarly, the structure *verb (present perfect) + just/yet/already* is a common pattern, so it makes sense to study it as a stable grammar structure. Furthermore, the flexibility of natural language grammar allows beginners to inadvertently create constructions that should formally be considered admissible, but in practice require quite counter-intuitive interpretations. Classic *garden path* sentences like *"The old man the boat"* (relying on the meaning of *man* as "operate") are good examples of such situations (Guo, 2016). To provide active and constructive feedback in relation to these issues, certain design choices can be made within the current WordBricks framework. For example, we can create a brick for present perfect tense verbs, so it includes a compulsory element for an adverb, limited to *just* or *already*, or remove the determiner connector from the bricks representing people's names. Other situations might need more in-depth analysis of semantic properties of a particular word in a particular context, in order to provide supportive feedback. Alternatively, the system can be used within a framework of carefully designed lessons, associated with predefined *admissible* constructions. This approach can be pedagogically sound, but further research is needed to determine how well it agrees with the goal of *making construals*.

Pedagogical decisions are also embedded in the rules accompanying individual exercises. The process of creating rules and exercises requires considerable effort, which can be possibly reduced by deriving them automatically from text corpora (Zanetti et al., 2021) and/or relying on existing formalisms in CALL such as a Grammatical Framework (Ranta, 2004). The deliberate choice of handcrafted rules and exercises in Anonymous reflects our wish to fine tune the system for the students of different levels and needs[3]. However, integration of brick-based visualisations with existing CALL instruments can be a valuable research direction.

7. Future Outlooks

[3] In our classroom experiments using the mobile prototype of the system (21 undergraduate students participated. 10 students used Anonymous, 11 students used standard teaching materials (Purgina et al., 2020)), we tried to reproduce several sections from Azar and Hagen (2005), to follow the teacher's lesson plans. The number of bricks and rules were stripped down to illustrate specific grammatical phenomena addressed in respective sections of the book.

There is no doubt that computational technologies hold a firm ground in L2 acquisition. However, this term covers a whole spectrum of diverse software instruments, used in daily teaching and learning practice. These instruments greatly vary in scope, goals, popularity, and effectiveness. Comprehensive surveys by Golonka et al. (2014) and Zhang and Zou (2022) examine the use of technologies, ranging from course management systems and interactive white boards to electronic dictionaries and speech recognition tools. Arguably, most instruments and technologies reviewed by the authors are designed either for general-purpose use (smartphones, social networks) or for wide educational setting (e-portfolios, course management systems). Thus, the question "how to create efficient tools aimed specifically at L2 learners" is still open. If we presume that there are certain tasks and scenarios specific for foreign language acquisition, we should expect them to benefit from specifically designed technological solutions.

Advancements in certain areas of natural language processing, on the one hand, show possible examples of such technologies, e.g., through ICALL. Learner-centric grammar checking tools (such as Grammarly) and automated speech recognition methods are already shown to be of value for the learners. It seems reasonable to presume that this list will grow in the future. On the other hand, evidently, designing such technology-driven learner-centric systems is not an easy task, and very few of them reach the stage of maturity and wide adoption. It is beyond the scope of our paper to analyse the reasons contributing to this situation, but our case study reveals certain factors that make advancements in this area challenging.

Our research addressed a relatively narrow and well-defined problem domain: supporting natural language grammar acquisition via "visual language", highlighting the properties of grammatical structures. In particular, we consider important to reflect word order, head/dependent relationships, verb government, and (where possible) word attributes like grammatical number, gender, and case. Allowing learners to explore sentence structures visually and interactively is of importance especially when the target language differs from their native language in syntactical composition. The inner organisation of natural language grammar is reasonably well understood, and there are established linguistic theories able to describe grammatical rules in a formal way, helpful for representation in a computer (Schulze & Penner, 2008). There are also numerous well-known "visual language environments", used both in industrial programming and in computer science education. Still, despite such reasonably firm ground many issues are open to the interpretation of software system designers, and optimal decisions depend on numerous technological, psychological, and pedagogical aspects that require further study.

The principal open challenge for our goals can be formulated as follows. We need a clean, consistent, and easy to understand visual representation of a natural language grammar. This representation should allow the learners to construct grammatically correct sentences, providing immediate visual feedback. The learners should be able to understand what words and phrases can be connected and why.

This challenge in practice reduces into specific decisions to make, such as how to represent grammatical attributes, how to deal with non-projective structures, how to represent acceptable variations. Some of possible solutions can be adopted from existing tools. For example, most programming environments assist textual input with advanced autocomplete features.

From a more general perspective, our project can be seen as an attempt to implement a "virtual lab"-like environment for natural language, where the learners are encouraged to grow language intuition by playing with words and phrases, exploring the possibilities and testing ideas. Such labs are well known in fields like physics, electronics engineering, or chemistry, and bring this concept to language education could be an exciting research goal.

8. Conclusion

We have discussed challenges faced and lessons learnt over the process of developing a software tool for natural grammar acquisition, based on a visual grammar formalism. We believe that our resulting system WordBricks is able to represent many basic grammatical structures, but not all of them look equally intuitive in the form of visual blocks. Thus, it was important for us to examine the scope where "visual grammar" approach can be helpful for the learners. We consider basic computer programming education to be a success story in this direction: there is general agreement that visual programming is beneficial at a certain stage of learning, although there is also an understanding that such systems are inherently limited and not designed for intermediate and advanced students. We suggest that an efficient natural language learning instrument has to be supported both from technological and from pedagogical sides. Software developers have to understand the needs of teachers and learners, as well as the tasks that objectively arise in the course of L2 acquisition. In our experience, engaging designers, developers, teachers and students in a co-creation process towards understanding the needs and requirements the technology needs to fulfil, yields the most suitable results as seen in the latest WordBricks prototype. However, course material designers similarly have to understand the affordances and limitations of today's technologies and decide how they can be used in the most efficient manner for L2 learning. Our current prototypes are flexible and adaptable to particular curricular needs, which could represent an interesting venue for future exploration, taking into account the pedagogical goals of instructors and the level of L2 sensitivity of the learners. This sensitivity could include the level of metalinguistic knowledge, i.e., conscious grammatical knowledge, of the learner, which would result in varied technological solutions suited according to the target language to acquire as well as the native language of the learners (Butler, 2002; Golonka et al., 2014; Roehr & Gánem-Gutiérrez, 2009). Further investigations on how our current prototypes, i.e., with and without explicit scaffolding, support learners' metalinguistic knowledge of the target language could also represent an interesting aspect to examine.

References

Alexandru, C. V., Merchante, J. J., Panichella, S., Proksch, S., Gall, H. C., & Robles, G. (2018). On the usage of pythonic idioms. In E. G. Boix (Ed.), *Proceedings of the 2018 ACM SIGPLAN International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software.* ACM. https://doi.org/10.1145/3276954.3276960

Azar, B., & Hagen, S. (2005). *Basic English Grammar, 3rd Ed*. Pearson Longman.

Bau, D., Gray, J., Kelleher, C., Sheldon, J., & Turbak, F. (2017). Learnable programming: Blocks and beyond. *Communications of the ACM*, *60*(6), 72–80.

Ben-Ari, M. (2001). Constructivism in Computer Science Education. *Journal of Computers in Mathematics and Science Teaching*, *20*(1), 45–73.

Bertram, R., Laine, M., & Virkkala, M. M. (2000). The role of derivational morphology in vocabulary acquisition: Get by with a little help from my morpheme friends. *Scandinavian Journal of Psychology*, *41*(4), 287–296. https://doi.org/10.1111/1467-9450.00201

Bick, E. (2004). Grammar for fun: IT-based grammar learning with VISL. In P. J. Henriksen (Ed.), *CALL for the Nordic Languages* (pp. 49–64). http://beta.visl.sdu.dk/pdf/call2004.pdf

Blake, R. (2017). Technologies for teaching and learning L2 speaking. In C. A. Chapelle & S. Sauro (Eds.), *Blackwell handbooks in linguistics. The handbook of technology and second language teaching and learning* (1st ed., pp. 107–117). Wiley Blackwell.

Bueno Alastuey, M. C. (2011). Perceived benefits and drawbacks of synchronous voice-based computer-mediated communication in the foreign language classroom. *Computer Assisted Language Learning*, *24*(5), 419–432. https://doi.org/10.1080/09588221.2011.574639

Butler, Y. G. (2002). Second Language Learners' Theories on the Use of English Articles. *Studies in Second Language Acquisition*, *24*(3), 451–480. https://doi.org/10.1017/S0272263102003042

Butler, Y. G. (2015). The use of computer games as foreign language learning tasks for digital natives. *System*, *54*, 91–102. https://doi.org/10.1016/j.system.2014.10.010

Cai, J., & Liu, Y. (2018). Research on English pronunciation training based on intelligent speech recognition. *International Journal of Speech Technology*, *21*(3), 633–640. https://doi.org/10.1007/s10772-018-9523-8

Chapelle, C. A. (2019). Technology-Mediated Language Learning. In J. W. Schwieter & A. Benati (Eds.), *The Cambridge Handbook of Language Learning* (pp. 575–596). Cambridge University Press. https://doi.org/10.1017/9781108333603.025

Chapelle, C. A., & Chung, Y.-R. (2010). The promise of NLP and speech processing technologies in language assessment. *Language Testing*, *27*(3), 301–315. https://doi.org/10.1177/0265532210364405

Chen, Y. L., & Hsu, C. C. (2020). Self-regulated mobile game-based English learning in a virtual reality environment. *Computers & Education*, 154, 103910. https://doi.org/10.1016/j.compedu.2020.103910

Choi, I.-C. (2016). Efficacy of an ICALL tutoring system and process-oriented corrective feedback. *Computer Assisted Language Learning*, *29*(2), 334–364. https://doi.org/10.1080/09588221.2014.960941

Chomsky, N. (1965). *Aspects of the theory of syntax*. MIT Press.

Cook, D. D. (2015). Flowgorithm: Principles for teaching introductory programming using flowcharts. In *Proc. American Society of Engineering Education Pacific Southwest Conf. (ASEE/PSW)* (158-167).

Cordy, J. R. (2004). TXL-a language for programming language tools and applications. *Electronic Notes in Theoretical Computer Science*, *110*, 3–31.

Debusmann, R. (2000). An introduction to dependency grammar. *Hausarbeit Fur Das Hauptseminar Dependenzgrammatik SoSe*, *99*, 1–16.

Diderichsen, P. (1946). *Elementær dansk grammatik*. Gyldendal.

Ding, S., Liberatore, C., Sonsaat, S., Lučić, I., Silpachai, A., Zhao, G., Chukharev-Hudilainen, E., Levis, J., & Gutierrez-Osuna, R. (2019). Golden speaker builder – An interactive tool for pronunciation training. *Speech Communication*, *115*, 51–66. https://doi.org/10.1016/j.specom.2019.10.005

Divekar, R. R., Drozdal, J., Chabot, S., Zhou, Y., Su, H., Chen, Y., Zhu, H., Hendler, J. A., & Braasch, J. (2021). Foreign language acquisition via artificial intelligence and extended reality: Design and evaluation. *Computer Assisted Language Learning*, 1–29. https://doi.org/10.1080/09588221.2021.1879162

Donesch-Jezo, E. (2011). The role of output and feedback in second language acquisition: A classroom-based study of grammar acquisition by adult English language learners. *Journal of Estonian and Finno-Ugric Linguistics*, *2*(2), 9–28. https://doi.org/10.12697/jeful.2011.2.2.01

Dorling, M., & White, D. (2015). Scratch: A Way to Logo and Python. In A. Decker, K. Eiselt, C. Alphonce, & J. Tims (Eds.), *Proceedings of the 46th ACM Technical Symposium on Computer Science Education* (pp. 191–196). ACM. https://doi.org/10.1145/2676723.2677256

Ebbels, S. (2007). Teaching grammar to school-aged children with specific language impairment using shape coding. *Child Language Teaching and Therapy*, *23*(1), 67–93.

Efimov, R., Mozgovoy, M., & Brine, J. (2014). CALL for Open Experiments. In *Proceedings of International Conference on Computer Supported Education (CSEDU 2014, Barcelona, Spain*.

Frank, R. (2004). *Phrase structure composition and syntactic dependencies*. MIT.

Gazdar, G. (1982). Phrase Structure Grammar. In *The Nature of Syntactic Representation* (pp. 131–186). Springer, Dordrecht. https://doi.org/10.1007/978-94-009-7707-5_5

Ghufron, M. A., & Rosyida, F. (2018). The role of Grammarly in assessing English as a Foreign Language (EFL) writing. *Lingua Cultura*, *12*(4), 395–403.

Godwin-Jones, R. (2017). Smartphones and language learning. *Language Learning & Technology*, *21*(2), 3–17.

Golonka, E. M., Bowles, A. R., Frank, V. M., Richardson, D. L., & Freynik, S. (2014). Technologies for foreign language learning: a review of technology types and their effectiveness. *Computer Assisted Language Learning*, *27*(1), 70–105.

Guevara, E. (2007). Binary branching and linguistic theory: lel 2007 example. *Lingue E Linguaggio*(VI.2), 1–11. https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=c0cd5b520df3e599d5d7d60d68bbb3b1601766d2

Guo, J. (2016). *Google's new artificial intelligence can't understand these sentences. Can you?* https://www.washingtonpost.com/news/wonk/wp/2016/05/18/googles-new-artificial-intelligence-cant-understand-these-sentences-can-you

Halwani, N. (2017). Visual Aids and Multimedia in Second Language Acquisition. *English Language Teaching*, *10*(6), 53–59. https://eric.ed.gov/?id=ej1143525

Harbusch, K., Breugel, C., Koch, U., & Kempen, G [G.]. (2007). Interactive sentence combining and paraphrasing in support of integrated writing and grammar instruction: A new application area for natural language sentence … In *11th Euopean Workshop in Natural Language Generation (ENLG07)* (pp. 65–68). https://pure.mpg.de/pubman/faces/viewitemoverviewpage.jsp?itemid=item_59671

Havelka, J. (2007). Beyond projectivity: Multilingual evaluation of constraints and measures on non-projective structures. In *45th Annual Meeting of the Association of Computational Linguistics*.

Heift, T., & Hegelheimer, V. (2017). Computer-assisted corrective feedback and language learning. *Corrective Feedback in Second Language Teaching and Learning: Research, Theory, Applications, Implications*, *66*, 51–65.

Heift, T., & Vyatkina, N. (2017). Technologies for Teaching and Learning L2 Grammar. In *The Handbook of Technology and Second Language Teaching and Learning* (pp. 26–44). John Wiley & Sons, Ltd. https://doi.org/10.1002/9781118914069.ch3

Hein, G. (1991). *Constructivist learning theory*. Institute for Inquiry. https://www.exploratorium.edu/education/ifi/constructivist-learning

Hewings, M. (2013). *Advanced grammar in use: A self-study reference and practice book for advanced learners of English* (3rd ed). Cambridge University Press.

Holland, V. M., Kaplan, J. D., & Sams, M. R. (1995). *Intelligent language tutors: Theory shaping technology*. Erlbaum.

Hubbard, P. (2021). An invitation to CALL. *Foundations of Computer Assisted Language Learning*.

Kayne, R. S. (1984). *Connectedness and binary branching*. *Studies in Generative Grammar: Vol. 16*. Foris Publications. https://doi.org/10.1515/9783111682228

Kelleher, C., & Pausch, R. (2005). Lowering the barriers to programming: A Taxonomy of Programming Environments and Languages for Novice Programmers. *ACM Computing Surveys (CSUR)*, *37*(2), 83–137.

Kempen, G [Gerard] (2004). Interactive visualization of syntactic structure assembly for grammar-intensive first-and second-language instruction. In *InSTIL/ICALL 2004 Symposium on NLP and Speech Technologies in advanced language learning systems.* Symposium conducted at the meeting of University of Venice.

Kern, R. (2006). Perspectives on Technology in Learning and Teaching Languages. *TESOL Quarterly*, *40*(1), 183. https://doi.org/10.2307/40264516

Kessler, G., & Bikowski, D. (2010). Developing collaborative autonomous learning abilities in computer mediated language learning: Attention to meaning among students in wiki space. *Computer Assisted Language Learning*, *23*(1), 41–58. https://doi.org/10.1080/09588220903467335

Krashen, S. D. (2003). *Explorations in language acquisition and use*. Heinemann Portsmouth, NH.

Lange, H., & Ljunglöf, P. (2018). 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*.

Lavolette, E., Polio, C., & Kahng, J. (2015). The accuracy of computer-assisted feedback and students' responses to it. *Language, Learning & Technology*, *19*(2), 50–68.

Li, B., Lowell, V. L., Wang, C., & Li, X. (2024). A systematic review of the first year of publications on ChatGPT and language education: Examining research on ChatGPT's use in language learning and teaching. Computers and Education: Artificial Intelligence, 7, 100266, https://doi.org/10.1016/j.caeai.2024.100266.

Ljunglöf, P. (2011). Editing syntax trees on the surface. In *Proceedings of the 18th Nordic Conference of Computational Linguistics (NODALIDA 2011)*.

Macaro, E., & Masterman, L. (2006). Does intensive explicit grammar instruction make all the difference? *Language Teaching Research*, *10*(3), 297–327. https://doi.org/10.1191/1362168806lr197oa

Marneffe, M.-C. de, & Manning, C. D. (2008). *Stanford typed dependencies manual*. Stanford University.

Mitchell, D. (1994). Sentence parsing. In M. A. Gernsbacher (Ed.), *Handbook of psycholinguistics* (pp. 375–409). Academic Press.

Murphy, R. (2012). *English Grammar in Use, 4th Ed*. Cambridge University Press.

Nicholas, A., Blake, J., Perkins, J., & Mozgovoy, M. (2024). Evaluating the Effectiveness of a Computerised dynamic Assessment of L2 English Email Requests. *Computer Assisted Language Learning*, 1-33. https://doi.org/10.1080/09588221.2024.2374775

Nivre, J. (2003). An efficient algorithm for projective dependency parsing. In *Proceedings of the eighth international conference on parsing technologies*.

ONeill, R., & Russell, A. (2019). Stop! Grammar time: University students' perceptions of the automated feedback program Grammarly. *Australasian Journal of Educational Technology*, *35*(1). https://doi.org/10.14742/ajet.3795

Park, Y., & Warschauer, M. (2016). Syntactic enhancement and second language literacy: An experimental study. *Language Learning & Technology*, *20*(3), 180–199.

Perveen, A. (2016). Synchronous and Asynchronous E-Language Learning: A Case Study of Virtual University of Pakistan. *Open Praxis*, *8*(1). https://doi.org/10.5944/openpraxis.8.1.212

Petrov, S., Das, D., & McDonald, R. (2012). A Universal Part-of-Speech Tagset. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)* (pp. 2089–2096). European Language Resources Association (ELRA).

Purgina, M., & Mozgovoy, M. (2017). Visualizing Sentence Parse Trees with WordBricks. In *3rd IEEE International Conference on Cybernetics (CYBCONF).* Symposium conducted at the meeting of IEEE.

Purgina, M., Mozgovoy, M., & Blake, J. (2020). WordBricks: Mobile Technology and Visual Grammar Formalism for Gamification of Natural Language Grammar Acquisition. *Journal of Educational Computing Research*, *58*(1), 126–159. https://doi.org/10.1177/0735633119833010

Purgina, M., Mozgovoy, M., & Ward, M. (2017). Learning Language Grammar with Interactive Exercises in the Classroom and Beyond. In *Proceedings of the 9th International Conference on Computer Supported Education*.

Qassemzadeh, A., & Soleimani, H. (2016). The Impact of Feedback Provision by Grammarly Software and Teachers on Learning Passive Structures by Iranian EFL Learners. *Theory and Practice in Language Studies*, *6*(9), 1884. https://doi.org/10.17507/tpls.0609.23

Ranta, A. (2004). Grammatical Framework. *Journal of Functional Programming*, *14*(2), 145–189. https://doi.org/10.1017/S0956796803004738

Rezaee, A. A., Marefat, H., & Saeedakhtar, A. (2015). Symmetrical and asymmetrical scaffolding of L2 collocations in the context of concordancing. *Computer Assisted Language Learning*, *28*(6), 532–549. https://doi.org/10.1080/09588221.2014.889712

Roehr, K., & Gánem-Gutiérrez, G. A. (2009). The status of metalinguistic knowledge in instructed adult L2 learning. *Language Awareness*, *18*(2), 165–181. https://doi.org/10.1080/09658410902855854

Rutherford, W. E. (2014). *Second Language Grammar: Learning and Teaching*. *Applied Linguistics and Language Study*. Routledge.

Schmidt, D. A. (1996). Programming language semantics. *ACM Computing Surveys (CSUR)*, *28*(1), 265–267.

Schulze, M., & Penner, N. (2008). Construction grammar in ICALL. *Computer Assisted Language Learning*, *21*(5), 427–440. https://doi.org/10.1080/09588220802447727

Slavuj, V., Meštrović, A., & Kovačić, B. (2017). Adaptivity in educational systems for language learning: a review. *Computer Assisted Language Learning*, *30*(1-2), 64–90. https://doi.org/10.1080/09588221.2016.1242502

Sokirko, A. (2004). Morphological modules on the website www.aot.ru (in Russian). In *Proceedings of the Dialog'04 International Conference* (pp. 559–564).

Stefik, A., & Siebert, S. (2013). An empirical investigation into programming language syntax. *ACM Transactions on Computing Education (TOCE)*, *13*(4), 1–40.

Tesnière, L. (2015). *Elements of structural syntax*. John Benjamins Publishing Company.

Thompson, E. (2018). Teaching computational reasoning through construals. *Education and Self Development*, *13*(3), 40–52.

van Lommel, S., Laenen, A., & d'Ydewalle, G. (2006). Foreign-grammar acquisition while watching subtitled television programmes. *The British Journal of Educational Psychology*, *76*(Pt 2), 243–258. https://doi.org/10.1348/000709905X38946

Verdi, M. P., Johnson, J. T., Stock, W. A., Kulhavy, R. W., & Whitman-Ahern, P. (1997). Organized Spatial Displays and Texts. *The Journal of Experimental Education*, *65*(4), 303–317. https://doi.org/10.1080/00220973.1997.10806606

Vinther, J. (2004). Can Parsers be a Legitimate Pedagogical Tool? *Computer Assisted Language Learning*, *17*(3-4), 267–288. https://doi.org/10.1080/0958822042000319584

Vygotsky, L. S. (1980). *Mind in society: The development of higher psychological processes*. Harvard University Press.

M. Ward, M. Mozgovoy, M. Purgina. Can WordBricks Make Learning Irish More Engaging For Students? *International Journal of Game-Based Learning*, 2019, vol. 9(2), pp. 20-39.

Webb, R. (2019). *Japanese Sentence Structure: The Ultimate Beginner's Guide*. https://8020japanese.com/japanese-sentence-structure/

Zanetti, A., Volodina, E., & Graën, J. (2021). Automatic Generation of Exercises for Second Language Learning from Parallel Corpus Data. *International Journal of*

TESOL Studies. Advance online publication. https://doi.org/10.46451/ijts.2021.06.05

Zhang, R., & Zou, D. (2022). Types, purposes, and effectiveness of state-of-the-art technologies for second and foreign language learning. *Computer Assisted Language Learning*, *35*(4), 696–742. https://doi.org/10.1080/09588221.2020.1744666

Ziegler, N., Meurers, D., Rebuschat, P., Ruiz, S., Moreno-Vega, J. L., Chinkina, M., Li, W., & Grey, S. (2017). Interdisciplinary Research at the Intersection of CALL, NLP, and SLA: Methodological Implications From an Input Enhancement Project. *Language Learning*, *67*(S1), 209–231. https://doi.org/10.1111/lang.12227