

Accelerating Constructed Language Design: A Method for LLM-Assisted Conlanging^{*}

Matteo Cristani^{1,*}, Chiara Melloni^{2,†}

¹Dipartimento di Informatica, Università degli Studi di Verona, Strada Le Grazie 15 37134 Verona (Italy)

³Dipartimento di Culture e Civiltà, Università degli Studi di Verona, Via dell'Università, 4, 37129 Verona (Italy)

Abstract

Constructed languages (conlangs) are traditionally developed through iterative and time-intensive design processes that require sustained control over multiple interacting linguistic levels. This paper proposes a structured methodology for the construction of artificial languages using a Large Language Model (LLM) as an assisting agent within a human-guided workflow. Rather than treating the LLM as an autonomous generator, the approach frames it as a constrained collaborator operating under a stepwise prompting protocol, where each phase of language construction—phonology, phonotactics, morphology, lexicon, syntax, and semantics—is explicitly decomposed and validated.

We present a systematic interaction framework based on prompt–response–evaluation cycles, combined with post-selection and consistency checks, allowing the experimenter to maintain full control over design decisions while leveraging the generative capacity of the model. A case study is provided in which a complete constructed language is developed following this methodology. The results indicate a significant reduction in development time compared to traditional practices reported in the literature, while preserving structural coherence and usability of the resulting language.

The paper argues that the effectiveness of the approach lies not in the generative power of the LLM alone, but in the rigor of the interaction protocol. This suggests a broader perspective in which LLMs function as agents within formalized creative processes, enabling accelerated design in domains requiring high structural consistency.

Keywords

Constructed Languages, Large Language Models, Human–AI Collaboration, Language Design Methodology, Prompt Engineering

1. Introduction

Constructed languages are generally developed through iterative design processes, in which phonology, morphology, syntax, semantics, and the lexicon are progressively specified and coordinated into a coherent system [1, 2]. This activity usually requires sustained manual control, repeated revisions, and a significant investment of time [1, 3]. While the literature on constructed languages has extensively discussed typological inspiration, aesthetic goals, and communicative functions [1], to the best of our knowledge comparatively little attention has been devoted to the formalization and acceleration of the construction process itself, particularly through structured computational assistance.

Recent work has begun to explore the use of Large Language Models in this domain. For instance, ConlangCrafter [4] proposes a pipeline-based approach in which language construction is decomposed into sequential modules, including phonological inventory selection, morphological patterning, syntactic structure, and lexicon generation. The system leverages LLMs to produce candidate structures at each stage and aims at maintaining internal coherence across linguistic levels. While this approach represents an important step toward the computational support of conlang design, its primary focus lies in generation quality and modular organization, rather than in the explicit formalization of a human–LLM interaction protocol or in the evaluation of development time and efficiency. In this respect, the

^{*}Corresponding author.

[†]These authors contributed equally.

✉ matteo.cristani@univr.it (M. Cristani); chiara.melloni@univr.it (C. Melloni)

id 0000-0001-5680-0080 (M. Cristani); 0000-0002-4711-2689 (C. Melloni)



© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

Lang.	Cat.	Creator	Span	Time	Ref.	Notes
Esperanto	Aux.	Zamenhof	1873–1887	14y	[5, 6]	Drafts 1873; <i>Unua Libro</i> 1887.
Volapük	Aux.	Schleyer	1879–1880	1y	[7, 8]	Rapid sketch-to-grammar publication.
Interlingua	Aux.	IALA	1937–1951	14y	[9, 10]	Committee-based development.
Lojban	Eng.	LLG	1987–1997	10y	[11, 12]	Large collaborative effort.
Toki Pona	Phil.	S. Lang	2001–2014	13y	[13, 14]	Online drafts to book codification.
Láadan	Phil.	Elgin	1982–1988	6y	[15, 16]	Grammar & dictionary 1988.
Quenya	Art.	Tolkien	1910–1973	63y	[17, 18]	Lifelong evolving system.
Klingon	Fic.	Okrand et al.	1984–1985	1y	[19, 20]	Designed for film use.
Na’vi	Fic.	Frommer	2005–2009	4y	[21, 22]	Developed with film production.
Dothraki	Fic.	D. J. Peterson	2009–2011	<2y	[23, 24, 25]	Rapid staged expansion.

Table 1

Representative conlangs and their initial or complete development spans. Span denotes the interval between first design and first substantial public form, or the durative development of some cases; “Time” is the approximate elapsed duration.

present work complements existing approaches by emphasizing the methodological structure of the interaction and by analyzing its impact on the acceleration of the construction process.

In particular, we investigate the use of a Large Language Model (LLM) as an assisting agent in the construction of a conlang. The aim is not to delegate the entire process to the model, nor to treat it as an autonomous author, but rather to study a structured human–LLM interaction in which the model operates as a constrained collaborator subject to systematic evaluation. The central idea is that conlang construction can be decomposed into explicitly defined phases, each governed by explicitly specified prompts and controlled through intermediate validation and post-selection by the human experimenter.

The specific thesis defended in this paper is that such a method is effective in generating a constructed language in a substantially shorter time compared to traditional conlanging practices. In addition to this temporal advantage, the method also appears to have practical usefulness, as it supports the rapid production of linguistically structured artifacts that can be tested, revised, and extended in a controlled way.

General overviews of conlang history and practice may be found in [3, 1]. For the purpose of comparing development time, however, the available evidence is uneven: most sources report elapsed spans between first design and first substantial publication or public use, while explicit person-hour estimates are rarely available. Table 1 therefore summarizes representative conlangs by category, together with the most defensible documented span and any qualitative evidence about the scale of effort.

A first point that emerges from the comparison is the strong heterogeneity of conlang development. Some languages intended for rapid deployment or tightly constrained purposes reached an initial public form within about one year, whereas others required a decade or more before a first mature codification, and some artistic languages remained open-ended projects across the creator’s lifetime. A second point is that the documented span depends on the endpoint chosen: first sketch, first grammar, first dictionary, first public performance, or later stabilized form. For this reason, the table should be read as a comparison of *initial documented development spans*, not as a claim about total lifetime work on each language.

Preliminary evidence on the efficiency of the proposed methodology is provided by a set of initial development experiments. In particular, we constructed two test languages aimed at defining the methodology, two additional test languages devoted to its refinement, and one experimental language designed to assess a variant of the approach. These five constructions were completed within a time span of less than three months, with a total effort corresponding to two experimenters working no more than three hours per week each.

Although this evidence is exploratory and does not yet constitute a controlled comparative study, it provides a first indication of the potential of the method to significantly reduce development time. In particular, even under limited and discontinuous working conditions, the methodology enabled the rapid production of multiple linguistically structured artifacts, supporting the claim that a structured human–LLM interaction can accelerate the construction process while maintaining control over the

resulting system.

The possibility of accelerating the construction of a language is not only a matter of efficiency, but also opens up new methodological perspectives for linguistic inquiry. Constructed languages can be used as controlled environments in which hypotheses about the organization of linguistic systems are explicitly implemented and tested. In this sense, a faster construction cycle allows for multiple iterations of design, evaluation, and revision within a limited time frame, thereby supporting an experimental approach to language structure that would otherwise be difficult to sustain in practice.

Preliminary evidence presented in this work suggests that such acceleration is not merely theoretical, but can be realized in concrete settings under relatively modest effort. This reinforces the view that structured human–LLM interaction can transform conlanging from a predominantly long-term and often open-ended activity into a more iterative and reproducible process.

Beyond this, the method enables exploratory applications in historical and comparative linguistics. For instance, it becomes possible to generate structurally plausible intermediate systems between documented stages of a language, or to model candidate structural outcomes of processes such as koiné formation and creolization under explicitly defined constraints. Similarly, one may construct and compare alternative candidate systems that are compatible with hypothesized common ancestries or structural correspondences across language families. These applications are not intended to replace established linguistic methods, but rather to complement them by providing a flexible framework for generating and testing structured hypotheses in a controlled and repeatable manner.

The contribution of the paper is therefore methodological. We propose a systematic pipeline for LLM-assisted language construction, illustrate it through a case study, and discuss the forms of evidence that support its effectiveness. More precisely, the paper offers: (i) a formalized interaction protocol for conlang design with an LLM, (ii) a structured description of the stages of construction, from phonological inventory to semantic and syntactic refinement, and (iii) an argument that the controlled use of an LLM can reduce development time without abandoning linguistic coherence.

The remainder of the paper is organized as follows. Section 2 discusses the relevant background and situates the proposal with respect to both conlanging practice and recent uses of LLMs in structured creative tasks. Section 3 presents the methodology of LLM-assisted language construction. Section 4 describes the implementation of the method in a concrete language-building experiment. Section 5 evaluates the proposed approach, with special attention to efficiency and practical usefulness. Section 6 discusses the broader implications of the method, and Section 7 concludes.

2. Background and Related Work

LLMs in interactive artifact construction. A first line of related work concerns the use of large language models in interactive settings where the quality of the artifact depends strongly on the structure of the prompt and on iterative refinement. Prompt engineering has increasingly been described not as a collection of ad hoc tricks, but as a systematic methodology for shaping model behavior through task specification, context control, exemplification, reasoning scaffolds, and revision cycles [26]. In particular, chain-of-thought prompting showed that explicitly eliciting intermediate steps can improve performance on multi-step tasks [27], while more recent work on meta-prompting has argued that effective interaction often requires detailed descriptions, explicit context specification, and step-by-step reasoning templates [28]. This line of work is directly relevant to the present paper because conlang construction is not a one-shot generation problem: it requires constrained artifact production, intermediate validation, and repeated reformulation of prompts as the specification evolves.

LLMs in linguistic and language-centered tasks. A second line of work concerns the use of LLMs in linguistic tasks more broadly. Recent surveys characterize LLMs as general-purpose language technologies whose domain performance depends on capabilities such as instruction following, reasoning, and structured generation [29]. In applied linguistic settings, LLMs have been studied for multilingual and low-resource translation, where they show both promise and important limitations. For example,

large-scale evaluation of multilingual machine translation found that GPT-4 surpassed a strong supervised baseline in a substantial subset of directions, while still lagging behind commercial systems in many low-resource settings [30]. Other work has proposed explicitly instructional or textbook-based prompting strategies for low-resource translation, showing measurable gains over zero-shot baselines [31]. Beyond translation, language-centered generation tasks now include forms of linguistic annotation and representation building, such as the automatic production of interlinear glossed text from speech [32]. Taken together, these studies suggest that LLMs can support structured linguistic generation when the task is decomposed and guided, which is precisely the kind of setting required for language construction.

Computational approaches to constructed language design. A third and more specific strand concerns computational approaches to conlang construction itself. Here the literature is still limited. One recent direction is procedural generation for games: Cai and Martens present an initial generator for fictional languages intended for game worlds, organized around phonology, morphology, and syntax, and explicitly describe the area as underexplored [33]. A more recent and much closer antecedent to the present work is ConlangCrafter, which proposes an end-to-end multi-hop LLM pipeline for conlang creation. Their system decomposes language design into modular stages such as phonology, morphology, syntax, lexicon generation, and translation, and combines randomness with self-refinement to promote diversity and internal consistency [4]. The present paper is closest to this third strand, but differs from it in emphasis: rather than pursuing full automation, it focuses on a human-LLM interaction protocol in which construction proceeds through validated substeps, persistent artifacts, and explicit acceptance or rejection decisions.

3. Methodology: LLM-Assisted Language Construction

The method proposed in this paper is based on a controlled interaction between a human experimenter and an LLM. The core assumption is that the construction of a conlang can be decomposed into a sequence of relatively well-defined tasks, and that each of these tasks can be addressed through prompting under explicit constraints. The role of the LLM is to generate structured options, propose organized solutions, and assist in checking consistency, whereas the role of the human experimenter is to define objectives, evaluate outcomes, refine constraints, and make final decisions.

3.1. Design Principles

The methodology is based on a structured interaction protocol between the experimenter and a Large Language Model, in which language construction is organized as a sequence of controlled operations. At a top level, the method is characterized by the following four principles:

- **Phase decomposition.** The construction of the language is divided into a sequence of phases corresponding to major linguistic levels, each further subdivided into explicit substeps.
- **Prompt-criterion duality.** Each substep is governed by a pair consisting of a prompt, which specifies the generative task assigned to the model, and a criterion, which is used by the experimenter to evaluate, refine, or reject the output.
- **Artifact persistence.** The results obtained at each step are externalized, stored, and reintroduced in subsequent phases, so that the construction process is cumulative and subject to consistency checks across stages.
- **Human-controlled validation.** The experimenter performs systematic post-selection and correction, maintaining final authority over all decisions and ensuring that the resulting language remains coherent and aligned with the intended design.

On the basis of these principles, the construction process is not treated as a single generative task, but as an accumulative procedure in which linguistic structures are progressively specified, validated, and fixed. The Large Language Model is therefore not used as an autonomous generator, but as a constrained component within a methodological framework that separates generation, evaluation, and consolidation of results.

Figure 1 provides a global view of the construction pipeline and summarizes the detailed operations associated with each stage.

Step 0: Experimental setup The process begins with the definition of the project scope and the organization of interactions, typically structured as one phase per working unit or session. At this stage, explicit rules governing interaction with the LLM are established, emphasizing a non-deferential and critically controlled use of the model, with clear human authority over decisions. The overall construction pipeline is initialized, and the sequence of steps that will guide the development of the language is defined.

Step 1: Phonology The first linguistic layer concerns the construction of the phonological system. This involves selecting phonemic inventories and exploring phonotactic constraints, followed by the generation and evaluation of basic phonological structures. Through iterative validation procedures, including contrast testing, the system converges toward a stable set of admissible forms. Prosodic features are also defined at this stage.

Step 2: Orthography Once the phonological layer is stabilized, an orthographic system is introduced. This step defines the unit of representation (e.g., phoneme, syllable, or morpheme) and establishes a consistent mapping between linguistic structures and their written forms. The resulting writing system provides a formal interface for representing the language and is specified in alignment with the phonological constraints previously defined.

Step 3: Morphology (first construction phase) The morphological system is then constructed by determining the overall typological profile of the language and specifying its core grammatical categories. Functional elements and morphological processes are progressively introduced, allowing the formation of a minimal working lexicon. These elements are validated through paradigm construction and consistency checks, leading to a provisional stabilization of morphological structures.

Step 4: Syntax Building on the morphological layer, the syntactic structure of the language is defined. This includes the specification of word order, constituent organization, and core syntactic configurations. Mechanisms governing argument structure, clause formation, and recursion are introduced and tested through controlled sentence construction. The integration of morphology into syntax is carefully managed to ensure coherence.

Step 5: Morpho-syntactic refinement At this stage, interactions between morphology and syntax are revisited and refined. Morphological paradigms are adjusted based on their syntactic behavior, and agreement systems or case marking strategies are fine-tuned. The lexicon is expanded in response to syntactic needs, and inconsistencies at the interface between levels are resolved.

Step 6: Semantics The semantic layer introduces the interpretation of linguistic structures. Core semantic categories are defined, and lexical items are aligned with their intended meanings. Compositional mechanisms are specified to ensure that meaning is systematically derived from structure, while ambiguity and scope relations are identified and controlled.

Step 7: Lexical system construction The lexicon is then expanded and organized into coherent semantic domains. This stage ensures that lexical growth remains consistent with previously defined morphological, syntactic, and semantic constraints. Phenomena such as synonymy, polysemy, and lexical gaps are monitored and regulated, and lexical items are integrated into the broader grammatical system.

Step 8: Systematic testing and validation Following the construction of the core system, the language undergoes systematic testing through controlled tasks involving both production and inter-

pretation. These tests are designed to reveal inconsistencies, ambiguities, or underspecified areas across all linguistic levels.

Step 9: Targeted revision and refinement Based on the results of testing, targeted revisions are performed. Specific issues are addressed at the appropriate level, and interfaces between components are adjusted as needed. The lexicon may be further expanded or refined in a controlled manner, and modified elements are revalidated to ensure system-wide consistency.

Step 10: Artifact consolidation The final stage consists in consolidating all validated components into a stable and coherent system. This involves resolving any remaining conflicts, performing global consistency checks, and preparing the language resources for reuse and extension. The process concludes with the definitive fixation and versioning of all artifacts.

3.2. Interaction Protocol

The practical workflow follows a structured cycle based on the interaction schema introduced above. For each substep, the experimenter first specifies a methodological description, defining the local objective, the available artifacts, and the relevant constraints. On this basis, a prompt is formulated and submitted to the Large Language Model, which produces a candidate output. The output is then evaluated according to explicitly defined criteria, assessing its adequacy with respect to the requirements of the substep.

The outcome of this evaluation determines a decision. In the case of rejection, a revised prompt is formulated, incorporating the issues identified during evaluation, and the generation step is iterated. In the case of acceptance, the result is fixed and stored as an artifact, becoming part of the persistent state of the construction and serving as input for subsequent substeps and phases.

This protocol has two main advantages. On the one hand, it reduces the cognitive burden associated with managing multiple interacting linguistic components, by isolating decisions within well-defined substeps. On the other hand, it produces a documented chain of intermediate artifacts and decisions, making the process easier to analyse, revise, and reproduce.

Methodological workflow. The overall construction process is organized as a sequence of phases and steps, each producing validated artifacts that are reused in subsequent stages.

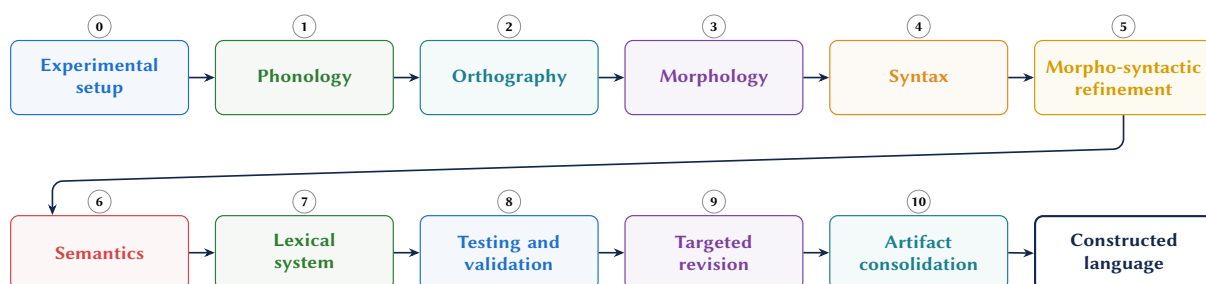


Figure 1: Global schema of the language construction pipeline. The process is organized as a sequence of stages, from experimental setup to artifact consolidation, leading to a coherent constructed language. The numbered nodes, from 0 to 10, are the steps of the pipeline and are traversed in that order, the second row continuing the first; the unnumbered node on the right is not a step but the resulting artifact. The same five principles govern every step: human authority and critical stance, explicit criteria and evaluation at each step, artifact persistence and traceability, iterative improvement and feedback loops, and coherence, consistency and usability.

At a global level, the methodology follows the workflow in Figure 1 that is a detailed version of the general schema presented in Figure 2. In each substep, instead, the methodology is implemented as illustrated in figure 3.

① **Experimental setup.** Define the project and organize the interactions, with one phase per chat or working unit; establish the interaction rules with the LLM (non-deference, critical stance, human authority); initialize the pipeline and define the sequence of steps.

- ① **Phonology.** Select the vowel and consonant inventories; define phonotactic constraints exploratively; generate and filter disyllabic structures on the basis of euphony; validate contrasts through minimal pairs; formalize phonotactic rules and admissible combinations; fix the phonological artifacts (vowels, consonants, syllables, rules); store the phonetic and phonotactic specifications; define the prosodic rules (stress, tone, and the like).
 - ② **Orthography.** Select the orthographic unit (phoneme, syllable, or morpheme); map linguistic units onto orthographic representations; define the writing system.
 - ③ **Morphology (first phase).** Define the global morphological typology; specify the core grammatical categories; construct functional morphemes and words; design inflectional and derivational morphology; generate a minimal working lexicon; validate via paradigms and consistency checks; fix and store the artifacts provisionally.
 - ④ **Syntax.** Define the global word order; specify constituent structure and the core syntactic configurations; define argument structure and complement marking; integrate the morphological markers; build clause-linking devices and discourse markers; define recursion and clause combination rules; validate via controlled sentence generation and parsing.
 - ⑤ **Morpho-syntactic refinement.** Revise the morphological paradigms on the basis of syntactic behavior; adjust agreement, case marking, and functional distribution; expand the lexicon in interaction with syntax; resolve and consolidate the morpho-syntactic interfaces.
 - ⑥ **Semantics.** Define the core semantic categories (tense, aspect, modality, quantification); align lexical items with their intended meanings; specify compositional semantics across structures; identify and control ambiguity and scope relations.
 - ⑦ **Lexical system construction.** Organize lexical fields and semantic domains; expand the lexicon under semantic and morpho-syntactic constraints; control synonymy, polysemy, and lexical gaps; verify lexical consistency across categories; integrate lexical items into the existing structures.
 - ⑧ **Systematic testing and validation.** Apply the language to controlled tasks (translation, generation, interpretation); perform bidirectional testing, in production and in parsing; identify inconsistencies, ambiguities, and underspecification across morphology, syntax, and semantics.
 - ⑨ **Targeted revision and refinement.** Correct the issues detected in morphology, syntax, and semantics; adjust the interfaces between linguistic levels; expand and refine the lexicon in a controlled manner; re-validate the modified components.
 - ⑩ **Artifact consolidation.** Fix all validated components; perform consistency checks across phases; resolve conflicts between previously defined artifacts; prepare reusable and extensible language resources; apply a final fixation and versioning of all components.
-
- ① **Method (substep definition).** States the goal of the substep, the inputs and artifacts available, the expected output, and the constraints and guidelines that the output must respect.
 - ② **Prompt (initial).** The instruction sent to the LLM in order to generate the desired output.
 - ③ **Criterion.** Evaluation of the output against explicit criteria: completeness, coherence, alignment with the constraints, and overall adequacy.
 - ④ **Decision.** On the basis of the evaluation the output is either rejected, and the substep returns to the prompting stage, or accepted, and the result is consolidated.
 - ⑤ **Iterative prompt.** A revised prompt, formulated on the issues identified during the evaluation, from which the LLM generates a new output.
 - ⑥ **Store.** The accepted output is stored as an artifact and becomes an input for subsequent substeps or phases.

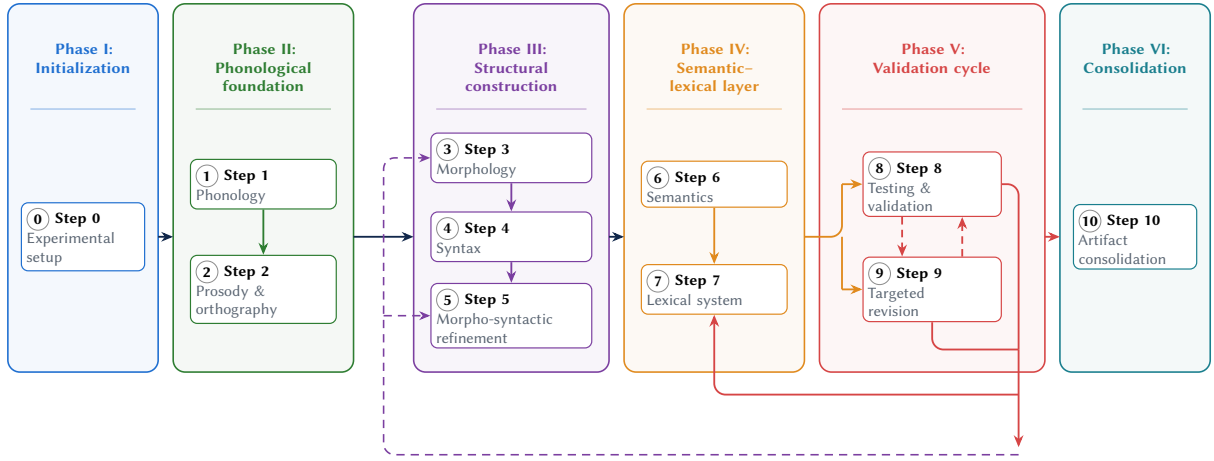


Figure 2: General schema of the language construction pipeline divided in different phases.

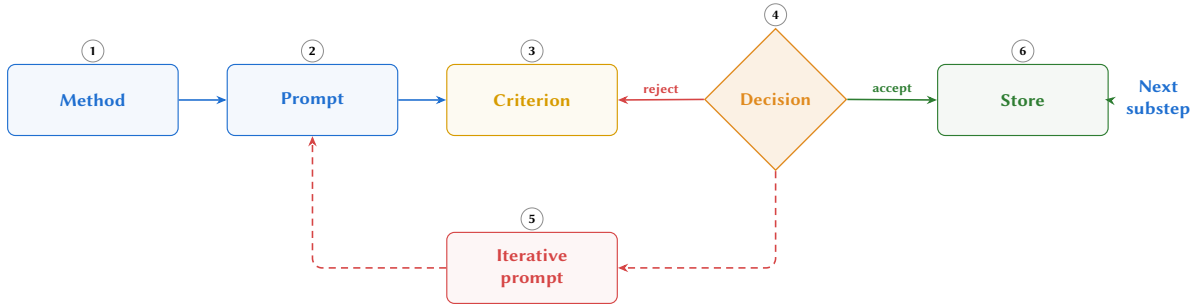


Figure 3: Schema of a substep in the proposed methodology. Each substep is governed by a method definition, followed by a prompt to the model, an evaluation phase based on explicit criteria, and a decision step. Rejected outputs trigger an iterative prompt, while accepted outputs are stored and reused in subsequent phases.

Figure 3 illustrates the internal structure of a substep within the proposed methodology. Each substep begins with a methodological specification, which defines the objective, constraints, and expected form of the output. This is followed by a prompt issued to the Large Language Model, which produces a candidate output. The output is then evaluated according to explicitly defined criteria.

On the basis of this evaluation, a decision is made. In the case of rejection, the process returns to the prompting stage through an iterative refinement of the instruction. In the case of acceptance, the result is consolidated and stored as an artifact, becoming part of the persistent state of the construction. This iterative and selective mechanism ensures both local correctness at each step and global coherence across the entire language construction process.

A key property of the proposed methodology is its strictly sequential organization across linguistic levels. Each phase is designed to stabilize a specific component of the language before introducing dependencies with subsequent levels. This prevents premature interference among partially defined structures and allows inconsistencies to be identified and resolved locally.

In this perspective, the Large Language Model is not employed as a single-shot generator of a complete language, but as a controlled assistant embedded within a stepwise construction process. The resulting system emerges through the progressive accumulation and validation of artifacts, rather than from unconstrained generation.

4. Case Study

4.1. Overview of the Constructed Language

The language developed in this case study results from the application of the methodology described in the previous section. The construction process was guided by explicit constraints aimed at ensuring phonological accessibility, structural regularity, and morphological transparency, while maintaining a controlled and reproducible workflow.

The constructed language developed in this case study will be referred to as LABORON. The choice of the name reflects the experimental nature of the methodology, evoking the idea of a linguistic “laboratory” in which structures are systematically generated, tested, and refined. At the same time, the form of the name complies with the phonological and morphological constraints of the language itself, following a regular CV-based structure and adopting the nominal coda used for lexical categorization.

Design principles. The construction process was guided by the following operational constraints:

- **Phonetic accessibility:** phonemes and syllable structures are selected to be easily pronounceable by speakers of Romance, Germanic, and Slavic languages.
- **Regularity:** the system minimizes irregularities across phonology, morphology, and lexicon.
- **Compositionality:** word structure encodes grammatical function in a predictable way.
- **Economy:** a restricted and optimized set of syllables is used to generate the lexicon.

Phonological structure. The phonological system is based on a predominantly CV syllabic structure, with single vowels allowed only in initial position. A limited set of consonantal codas is reserved for marking lexical categories. The admissible syllables were selected through a ranking procedure based on cross-linguistic ease of pronunciation, followed by euphony-based filtering and contrast validation, and filtered to eliminate combinations that could be difficult or ambiguous for the target speaker groups (see Step 1).

Prosody. The stress system is fully regular: stress falls on the penultimate syllable in polysyllabic words, while monosyllables carry inherent stress. Mid-open vowels are restricted to stressed positions, and each word contains exactly one lexical stress. These constraints were introduced to ensure deterministic stress assignment and reduce ambiguity during parsing tasks (see Step 1).

Lexical organization. A central design decision was to encode grammatical categories directly in word length, eliminating the need for additional morphological marking. Primitive lexical items follow fixed syllabic templates whereby word length encodes part-of-speech (derived in Step 3 and refined in Step 5):

- nouns (3 syllables);
- adjectives (4);
- verbs (5);
- adverbs (6).

This mechanism reduces the need for additional morphological marking and contributes to the transparency of the system.

Functional system. Functional elements are primarily realized as disyllabic forms and include demonstratives, quantifiers, and adpositions. In particular, adpositions are organized in bidirectional pairs, where each form can function as a preposition or postposition depending on syntactic configuration. Careful selection ensures the absence of phonological collisions among functional words.

Morphological strategy. The language adopts a hybrid morphological strategy, combining analytic expression of grammatical relations with structural encoding of lexical categories through syllable count and coda markers. This strategy provides both clarity and expressive flexibility (see Steps 3–5).

Emergent properties. The resulting system exhibits a high degree of structural regularity and predictability, although these properties depend on the controlled constraints imposed during construction. Pronunciation is largely deterministic, grammatical categories are transparently encoded, and the lexicon is generated within a constrained and optimized phonological space. These properties contribute to learnability and make the language suitable for controlled experimental use (validated in Step 7 and consolidated in Step 9).

4.2. Methodological Remarks

The construction of this language illustrates the effectiveness of the proposed methodology. Each component was developed through iterative substeps involving generation, evaluation, and validation, as described in Figure 3. The progressive fixation of artifacts ensured consistency across phases, while the controlled use of prompts allowed the exploration of alternatives without compromising coherence.

In particular, the use of explicit constraints at each stage (e.g., syllable ranking, phonotactic filtering, fixed-length lexical templates) demonstrates how the interaction protocol enables the systematic construction of a language without relying on unconstrained generative processes. This supports the claim that the methodology can produce linguistically structured artifacts in a controlled and reproducible manner.

4.3. Illustrative Lexical Examples

The following examples are consistent with the class-marking codas adopted for nonfunctional words: nouns end in **-n**, adjectives in **-l**, verbs in **-r**, and adverbs in **-s**. Functional words do not take class-marking codas.

Noun	Meaning
<i>kalonan</i>	person / man
<i>katelen</i>	water
<i>padaren</i>	house
<i>kamolan</i>	food
<i>masilan</i>	sun
<i>satiran</i>	road
<i>tamakan</i>	hand
<i>vatanen</i>	language

Table 2

Some example nouns in LABORON.

Interpretation of the examples. The lexical items and sentences reported above illustrate the structural properties induced by the construction methodology. First, the examples confirm the strict correspondence between lexical category, syllable count, and class-marking coda, showing that the morphological constraints are systematically enforced across the lexicon. No violations of these constraints are observed in the generated forms.

Adjective	Meaning
<i>pakalemal</i>	big
<i>masilenal</i>	small
<i>kaperimal</i>	good
<i>tadorunal</i>	bad
<i>panarimal</i>	new
<i>katelumal</i>	old

Table 3

Example adjectives in LABORON.

Verb	Meaning
<i>patorimakar</i>	to see
<i>kaderufitor</i>	to eat
<i>makalepusar</i>	to go
<i>tavolinamer</i>	to speak
<i>petulokasir</i>	to take

Table 4

Illustrative verbs.

Adverb	Meaning
<i>pakalematanas</i>	quickly
<i>masilenatoras</i>	slowly
<i>katorimakaras</i>	well
<i>padarimakaras</i>	strongly

Table 5

Some adverbs in LABORON.

Disyllable	Function
<i>lata</i>	article (class + singular marker)
<i>pana</i>	demonstrative (proximal: "this")
<i>pena</i>	demonstrative (medial: "that")
<i>puna</i>	demonstrative (distal: "that, far")
<i>hana</i>	quantifier ("all")
<i>heni</i>	quantifier ("some")
<i>tari</i>	preposition ("to, toward")

Table 6

Disyllabic functional elements used in the illustrative sentences.

Example sentence	Translation
<i>lata kalonan patorimakar lata kamolan.</i>	The person sees the food.
<i>lata kalonan pakalemal patorimakar lata kamolan.</i>	The big person sees the food.
<i>pana kalonan kaderufitor lata kamolan.</i>	This person eats the food.
<i>lata kalonan kaderufitor lata kamolan pakalematanas.</i>	The person eats the food quickly.
<i>lata kalonan makalepusar tari lata padaren.</i>	The person goes to the house.
<i>hana kalonan patorimakar lata masilan.</i>	All people see the sun.
<i>heni kalonan kaderufitor lata kamolan.</i>	Some people eat the food.
<i>pana kalonan pakalemal tavolinamer lata vatanen katorimakaras.</i>	This big person speaks the language well.
<i>puna padaren tadorunal.</i>	That house is bad.
<i>pena kalonan panarimal makalepusar tari lata satiran pakalematanas.</i>	That new person goes toward the road quickly.

Table 7

Illustrative sentences in LABORON.

Second, the phonological regularity of the system is reflected in the uniform CV-based structure of all lexical items, with no occurrence of complex consonant clusters or irregular syllabic patterns. This demonstrates that the phonotactic filtering applied during construction effectively restricts the admissible space of forms.

Third, the examples show that the language supports compositional combination at the syntactic level. The sentences exhibit consistent word order, appropriate use of functional elements, and predictable interaction between lexical categories. In particular, modifiers, arguments, and adverbial elements integrate without requiring additional morphological adjustment, confirming the transparency of the design.

Finally, these examples provide evidence that the methodology produces not only isolated lexical items, but a coherent system capable of supporting basic communicative structures. The correctness of the examples is not the result of post-hoc editing, but of the iterative validation process embedded in the construction protocol, which enforces local constraints at each substep and ensures global consistency across the language.

5. Evaluation

Recent work has begun to address the problem of evaluating constructed languages from a methodological perspective. In particular, [34] argues that conlang evaluation must be grounded in the intended purpose of the language, and that no single set of criteria can be universally applied across different types of constructed languages. Evaluation is therefore inherently goal-dependent, and typically involves dimensions such as internal coherence, learnability, typological plausibility, and satisfaction of design objectives.

The approach proposed in this paper differs from existing frameworks in a fundamental way. Rather than treating evaluation as a post-hoc analysis of a completed language, it integrates evaluation directly into the construction process. Each substep of the methodology includes an explicit evaluation phase, governed by predefined criteria, followed by a decision step that determines whether the generated artifact is accepted, refined, or rejected. In this sense, evaluation is not a separate stage, but a continuous and constitutive component of the methodology.

This integrated perspective provides two main advantages. First, it allows inconsistencies and inadequacies to be detected and corrected locally, before they propagate to higher levels of the system. Second, it ensures that the resulting language satisfies its design constraints incrementally, rather than requiring global revision after completion. As a consequence, the methodology supports both structural coherence and efficiency, aligning the evaluation process with the iterative nature of the construction.

More specifically, following [34], the assessment of a constructed language must be grounded in its intended purpose. In the present work, the goal is not to produce a naturalistic or auxiliary language, but to construct a linguistically coherent system under explicit structural constraints, while minimizing development time through a controlled interaction with a Large Language Model.

Under this perspective, traditional evaluation dimensions such as naturalness or expressivity are secondary, whereas internal coherence, constraint satisfaction, and learnability become central. Moreover, the proposed methodology introduces an additional dimension that is not explicitly considered in existing frameworks, namely the *processual validity* of the construction. This refers to the extent to which the language is obtained through a reproducible sequence of validated steps, rather than through ad hoc generation.

As a consequence, the evaluation of LABORON is structured along four main criteria: (i) internal coherence, (ii) constraint compliance, (iii) compositional transparency, and (iv) processual validity. These criteria are applied to the artifacts generated during construction, rather than only to the final system. Conceptually, the evaluation proposed is reported in Table 8. As applied to LABORON we have Table 9.

Criterion	Description
Internal coherence	Consistency of phonological, morphological, and syntactic rules across all generated artifacts.
Constraint compliance	Satisfaction of explicitly defined constraints (e.g., CV structure, syllable length, class-marking codas).
Compositional transparency	Predictability of structure and meaning from constituent elements.
Processual validity	Degree to which the language is constructed through validated and reproducible substeps.

Table 8

Evaluation criteria adapted from [34].

Criterion	Observations in LABORON
Internal coherence	No violations observed in the correspondence between word length, lexical category, and class-marking coda; syntactic patterns remain consistent across examples.
Constraint compliance	All lexical items conform to CV-based phonotactics; syllable counts and codas strictly follow predefined templates; no phonological collisions detected among functional words.
Compositional transparency	Grammatical roles are directly inferable from word structure; modifiers and arguments combine without additional marking; sentence interpretation is largely deterministic.
Processual validity	All components were generated through iterative substeps with explicit criteria and validation; artifacts were progressively fixed and reused, ensuring reproducibility of the construction process.

Table 9

Application of evaluation criteria to LABORON.

Additional evaluation dimensions. The evaluation of constructed languages has been addressed in the literature from multiple perspectives, depending on the intended purpose of the language. In interlinguistic contexts, particularly in the study of auxiliary languages, emphasis is placed on learnability, regularity, and ease of acquisition, often associated with reduced cognitive load and morphological transparency [35, 36]. In contrast, work on naturalistic constructed languages highlights typological plausibility and alignment with cross-linguistic patterns, drawing on insights from linguistic typology [37, 1].

Other approaches, especially in the context of engineered or logical languages, focus on formal consistency, unambiguous parsing, and the completeness of grammatical rules [38, 39]. From a more general linguistic perspective, additional criteria such as generativity and expressive adequacy are also relevant, referring respectively to the ability of a language to produce well-formed novel expressions and to support basic communicative functions [40, 41].

These dimensions are only partially aligned with the objectives of the present work. In particular, typological naturalness is not a primary goal, whereas structural learnability, compositional transparency, and formal consistency are directly relevant. The examples presented in Section 4 provide qualitative evidence that the constructed language supports productive combination of elements and predictable interpretation.

However, existing approaches generally evaluate the final linguistic artifact, rather than the process through which it is obtained. In this respect, the notion of *processual validity* introduced in this paper complements the aforementioned criteria by assessing the reproducibility, traceability, and internal validation of the construction process itself. This perspective extends recent proposals on conlang evaluation [34], by integrating evaluation within the methodology rather than treating it as a post-hoc analysis.

6. Discussion

The main result of this paper is methodological. More precisely, the paper does not show that a Large Language Model can autonomously design a constructed language of acceptable quality by means of unconstrained generation. Rather, it shows that a conlang can be developed through a structured human–LLM interaction protocol in which generation, evaluation, acceptance, rejection, and storage are explicitly separated and organized into validated substeps. Under this view, the LLM functions neither as an independent author nor as a mere text generator, but as a constrained assisting component embedded in a broader construction methodology.

This claim is supported in three ways. First, the paper formalizes the interaction protocol itself, specifying phase decomposition, prompt–criterion duality, artifact persistence, and human-controlled validation as the core principles of the method. Second, it provides a worked case study, in which the language LABORON is developed under these constraints and exhibits regular phonological structure, transparent lexical organization, and compositional sentence formation. Third, it evaluates the outcome by means of criteria adapted to the goals of the language, namely internal coherence, constraint compliance, compositional transparency, and processual validity. Taken together, these elements support the claim that the proposed methodology is capable of producing a linguistically structured artifact in a controlled and reproducible manner.

A second point established by the paper concerns efficiency. The evidence presented here does not amount to a fully controlled comparative benchmark against the entire history of conlanging, nor does it justify a universal claim that LLM-assisted construction is always faster than traditional practice. What it does show is that, under modest and discontinuous working conditions, the proposed protocol enabled the construction of multiple test languages and one experimental language within a limited period of time. In this sense, the paper provides preliminary but concrete evidence that a structured interaction with an LLM can reduce development time by lowering revision overhead, isolating decisions within local substeps, and progressively fixing validated artifacts before inconsistencies propagate across the system.

The case study also clarifies *how* this efficiency gain is obtained. The gain does not derive simply from the speed with which the model can generate candidate forms. Rather, it derives from the methodological organization of the process: the decomposition of tasks into manageable substeps, the use of explicit criteria at each stage, the possibility of iterative rejection and reformulation, and the cumulative reuse of accepted artifacts. In this respect, the paper suggests that the effectiveness of LLM assistance lies less in autonomous creativity than in disciplined interaction design.

The results further indicate that the proposed approach is especially suitable for experimental language design. Because the method makes constraints explicit and stores intermediate decisions, it allows the experimenter to treat language construction as a controlled environment for testing hypotheses about phonological, morphological, lexical, and syntactic organization. The acceleration obtained through the protocol is therefore not only a practical advantage, but also a methodological one: it makes repeated cycles of construction, testing, and revision more feasible than in longer and less formalized workflows.

At the same time, the scope of the present contribution remains limited. The evidence is still exploratory, the case study concerns a language constructed under a specific set of design goals, and several evaluation dimensions remain qualitative rather than quantitative. In particular, the paper does not yet offer a direct experimental comparison with alternative construction methods, nor does it measure learnability or communicative adequacy through user studies. For this reason, the present work should be understood as establishing the feasibility and promise of the methodology, rather than as providing a final or exhaustive assessment of its superiority.

Within these limits, however, the paper supports a broader conclusion. LLMs appear to be most useful in conlang construction not when they are treated as autonomous creators, but when they are inserted into a protocol that enforces explicit constraints, intermediate validation, and human control. This suggests a more general perspective in which LLMs may serve as effective assistants in the design of other formal or semi-formal systems, provided that the process is structured in a way that preserves

traceability, reproducibility, and critical supervision. The case study suggests that the main contribution of LLMs to conlang construction is not simply generative abundance, but interactive support under controlled conditions. In other words, the efficiency gain does not derive from asking the model to invent a language all at once, but from embedding the model in a disciplined methodological framework.

This has a broader significance. It indicates that LLMs may be fruitfully understood as agents within structured creative processes, especially in domains where consistency constraints are strong and where the human user is willing to operate as a supervisor, critic, and selector. Under this view, the methodological design of the interaction becomes more important than the raw generative capacity of the model.

The paper also raises the question of generalizability. If LLM-assisted conlang construction can be formalized in this way, similar methodologies may be applicable to other domains involving the design of formal or semi-formal systems, such as symbolic notations, controlled vocabularies, ontological taxonomies, or artificial communication schemes. The present work does not attempt to settle these broader questions, but it suggests that they deserve further investigation.

7. Conclusions and further work

This paper has proposed a structured methodology for the construction of a constructed language through controlled interaction with a Large Language Model. The central contribution is not the use of an LLM as an autonomous generator, but the formalization of an interaction protocol in which generation, evaluation, and validation are explicitly separated and organized into reproducible substeps.

The results obtained in the case study demonstrate that such a methodology can produce a linguistically coherent system under explicit structural constraints. The constructed language LABORON exhibits regular phonological organization, transparent lexical architecture, and compositional sentence formation, providing evidence that the protocol supports the systematic construction of a consistent linguistic artifact. LABORON is however an experimental case. To ascertain the validity of the approach we shall perform experiments on human cases on a larger base. The plan is to deliver groups of multiple developers and make comparisons on the results based on differentiated step sets assignments, and a control group developing a language only based on methodology, but without LLM assistance.

From an evaluative perspective, the methodology introduces a shift from post-hoc assessment to integrated validation. By embedding evaluation within each substep, the approach ensures that constraints are enforced locally and progressively, reducing the need for global revision and supporting convergence toward a coherent system. This processual dimension complements existing evaluation frameworks and constitutes a key aspect of the contribution.

In addition, preliminary evidence suggests that the proposed approach can significantly reduce development time under realistic working conditions. While the present results are exploratory and do not yet constitute a controlled comparative benchmark, they indicate that the efficiency gain derives from the structured organization of the interaction rather than from unconstrained generative capacity alone.

Future work. Several directions for future research emerge from this work. A first line of investigation concerns the development of a systematic experimental framework for comparing LLM-assisted conlang construction with traditional methodologies. In particular, it would be valuable to design controlled experiments involving multiple experimenters working under different conditions, and to define quantitative metrics for evaluating development time, revision effort, and structural consistency. Such a framework would allow a more rigorous assessment of the efficiency claims advanced in this paper.

A second direction concerns the extension of the methodology to multi-agent settings. Rather than relying on a single human–LLM interaction, one could employ agentic AI systems in which multiple agents are assigned distinct linguistic roles (e.g., generator, validator, critic, or interpreter), interacting through structured protocols. In this context, it would be possible to study how linguistic artifacts and

utterances are negotiated, refined, and stabilized across agents, providing insights into the dynamics of distributed language construction and the management of consistency in collaborative environments.

More broadly, future work may explore the application of the proposed interaction protocol to other domains involving structured artifact design, such as formal languages, ontologies, or controlled vocabularies. These directions would contribute to a better understanding of how LLMs can be integrated into disciplined creative processes requiring both generative flexibility and strong structural constraints.

Declaration on Generative AI

During the preparation of this work, the authors used ChatGPT 5.3 to paraphrase and reword, perform grammar and spelling checks. After using these tools/services, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

This work also employed ChatGPT 5.3 (OpenAI), under an Education license, as a constrained component within the methodological framework described here. Its use strictly adhered to the guidelines and regulations established by the University of Verona regarding the responsible use of artificial intelligence in research.

References

- [1] D. J. Peterson, *The Art of Language Invention: From Horse-Lords to Dark Elves to Sand Worms, the Words Behind World-Building*, Penguin, New York, 2015.
- [2] S. Sowers-Wills, Phonology from the inside out: Constructed language as a pedagogical tool, *Canadian Journal of Linguistics/Revue canadienne de linguistique* 69 (2024) 307–332. doi:10.1017/cnj.2024.21.
- [3] A. Okrent, *In the Land of Invented Languages: Adventures in Linguistic Creativity, Madness, and Genius*, Spiegel & Grau, New York, 2010.
- [4] M. Alper, M. Yanuka, R. Giryas, G. Beguš, Conlangcrafter: Constructing languages with a multi-hop llm pipeline, arXiv preprint arXiv:2508.06094 (2025). doi:10.48550/arXiv.2508.06094. arXiv:2508.06094, accepted to ACL 2026.
- [5] Wikipedia, L. L. Zamenhof, 2026. URL: https://en.wikipedia.org/wiki/L._L._Zamenhof.
- [6] The Editors of Encyclopaedia Britannica, Esperanto, 2026. URL: <https://www.britannica.com/topic/Esperanto>.
- [7] Wikipedia, Volapük, 2026. URL: <https://en.wikipedia.org/wiki/Volap%C3%BCk>.
- [8] The Editors of Encyclopaedia Britannica, Volapük, 2026. URL: <https://www.britannica.com/topic/Volapuk>.
- [9] Wikipedia, Interlingua, 2026. URL: <https://en.wikipedia.org/wiki/Interlingua>.
- [10] The Editors of Encyclopaedia Britannica, Interlingua, 2026. URL: <https://www.britannica.com/topic/Interlingua>.
- [11] The Logical Language Group, What is lojban?, 2026. URL: <https://www.lojban.org/static/whatis.html>.
- [12] Wikipedia, Lojban, 2026. URL: <https://en.wikipedia.org/wiki/Lojban>.
- [13] sona pona, History, 2026. URL: <https://sona.pona.la/wiki/History>.
- [14] Wikipedia, Toki pona, 2026. URL: https://en.wikipedia.org/wiki/Toki_Pona.
- [15] Wikipedia, Láadan, 2026. URL: <https://en.wikipedia.org/wiki/L%C3%A1adan>.
- [16] S. H. Elgin, *A First Dictionary and Grammar of Láadan*, Society for the Furtherance and Study of Fantasy and Science Fiction, 1988.
- [17] The Tolkien Estate, Timeline (1892–1949), 2026. URL: <https://www.tolkienestate.com/life/timeline-1892-1949/>.
- [18] Wikipedia, Quenya, 2026-10-15. URL: <https://en.wikipedia.org/wiki/Quenya>.
- [19] Wikipedia, Klingon language, 2026. URL: https://en.wikipedia.org/wiki/Klingon_language.

- [20] I. Chen, Much ado about klingon, 2026. URL: <https://www.wired.com/1997/07/much-ado-about-klingon/>.
- [21] Wikipedia , Na'vi language, 2026. URL: https://en.wikipedia.org/wiki/Na%CA%BCvi_language.
- [22] P. Frommer, About na'viteri, 2026. URL: <https://naviteri.org/about/>.
- [23] Wikipedia , David j. peterson, 2026. URL: https://en.wikipedia.org/wiki/David_J._Peterson.
- [24] A. Watercutter, Interview: Creating language for hbo's game of thrones, 2026. URL: <https://www.wired.com/2010/08/a-game-of-thrones-david-j-peterson/>.
- [25] D. J. Peterson, About dothraki, 2026. URL: <https://dothraki.com/about-dothraki/>.
- [26] P. Sahoo, A. K. Singh, S. Saha, V. Jain, S. Mondal, A. Chadha, A systematic survey of prompt engineering in large language models: Techniques and applications, arXiv preprint arXiv:2402.07927 (2024). doi:10.48550/arXiv.2402.07927. arXiv:2402.07927.
- [27] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. V. Le, D. Zhou, Chain-of-thought prompting elicits reasoning in large language models, in: Advances in Neural Information Processing Systems, volume 35, 2022, pp. 24824–24837.
- [28] Q. Ye, M. Ahmed, R. Pryzant, F. Khani, Prompt engineering a prompt engineer, in: Findings of the Association for Computational Linguistics: ACL 2024, Association for Computational Linguistics, Bangkok, Thailand, 2024, pp. 355–385. doi:10.18653/v1/2024.findings-acl.21.
- [29] J. Li, Y. Yang, Y. Bai, X. Zhou, Y. Li, H. Sun, Y. Liu, X. Si, Y. Ye, Y. Wu, Y. Lin, B. Xu, B. Ren, C. Feng, Y. Gao, H. Huang, Fundamental capabilities of large language models and their applications in domain scenarios: A survey, in: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Bangkok, Thailand, 2024, pp. 11116–11141. doi:10.18653/v1/2024.acl-long.599.
- [30] W. Zhu, H. Liu, Q. Dong, J. Xu, S. Huang, L. Kong, J. Chen, L. Li, Multilingual machine translation with large language models: Empirical results and analysis, in: Findings of the Association for Computational Linguistics: NAACL 2024, Association for Computational Linguistics, Mexico City, Mexico, 2024, pp. 2765–2781. doi:10.18653/v1/2024.findings-naacl.176.
- [31] P. Guo, Y. Ren, Y. Hu, Y. Li, J. Zhang, X. Zhang, H. Huang, Teaching large language models to translate on low-resource languages with textbook prompting, in: Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024), ELRA and ICCL, Torino, Italy, 2024, pp. 15685–15697. URL: <https://aclanthology.org/2024.lrec-main.1362/>.
- [32] T. He, K. Choi, L. Tjautja, N. R. Robinson, J. Shi, S. Watanabe, G. Neubig, D. R. Mortensen, L. Levin, Wav2gloss: Generating interlinear glossed text from speech, in: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Bangkok, Thailand, 2024, pp. 568–582. doi:10.18653/v1/2024.acl-long.34.
- [33] A. Cai, C. Martens, Towards procedural generation of constructed languages for games, in: AIIDE Workshop on Experimental Artificial Intelligence in Games, volume 3626, CEUR Workshop Proceedings, 2023, pp. 1–10. URL: <https://ceur-ws.org/Vol-3626/paper3.pdf>.
- [34] D. Peterson, J. Peterson, A methodological proposal for conlang evaluation, Annual Review of Applied Linguistics 44 (2024) 126–132. doi:10.1017/S0267190524000084.
- [35] A. Gode, I. A. L. Association, Interlingua–English: A Dictionary of the International Language, Storm Publishers, 1951.
- [36] D. Blanke, Internationale Plansprachen: Eine Einführung, Akademie-Verlag, Berlin, 1985.
- [37] B. Comrie, Language Universals and Linguistic Typology: Syntax and Morphology, 2 ed., University of Chicago Press, 1989.
- [38] J. C. Brown, Loglan 1: A Logical Language, revised fourth ed., The Loglan Institute, 1999. URL: <https://www.loglan.org/Loglan1/>, 2026-10-15.
- [39] The Logical Language Group, The Complete Lojban Language, 1997. Reference grammar.
- [40] N. Chomsky, Aspects of the Theory of Syntax, MIT Press, 1965.
- [41] D. R. Dowty, R. E. Wall, S. Peters, Introduction to Montague Semantics, D. Reidel Publishing Company, Dordrecht, 1981. doi:10.1007/978-94-009-9065-4.