Towards a Community-Driven Controlled Natural Languages Evolution

Martin LUTS[a,b], Monika SAARMANN[a,b], Daniel TIKKERBÄR[c], and Marius KUTATELADZE[d]

[a] *ELIKO Competence Centre in Electronics-, Information- and Communications Technologies, Estonia*

[b] *Department of Informatics, Tallinn University of Technology, Estonia*

[c] *M3D Ltd, Estonia*

[d] *FocusIT Ltd, Estonia*

Abstract. Controlled Natural Languages (CNLs) are usually engineered in a design-centric paradigm by a coherent project team. We can draw a parallel between such an approach and a commercial software development process, genetically modified organism engineering, cathedral design, auxiliary languages design. According to our knowledge, no planned international auxiliary language has turned out to be a viable solution, contrary to creoles and pidgins. The paper proposes a methodological vision towards a community-driven CNLs evolution – a paradigm where a group of volunteers develop CNLs based on the principles of bazaar-style open software development methodologies. We demonstrate the application of the approach on the basis of industrial use-cases elaborated for developing a Controlled Natural Estonian.

Keywords: Controlled Natural Language design, use-cases, community-driven

1. Introduction

Creole and pidgin languages are often considered simplified languages *unconsciously* born from a practical situation. Sociolinguistically, an environment *spontaneously* gives birth to a pidgin language [1]. A planned language, on the other hand, is created *consciously* by a person (Esperanto, Volapük, Latino sine flexione, et al.) or by a coherent project team (Interlingua de IALA, Lojban, et al.). Because there is a *group to create* the pidgin and creole language, a group of speakers is not lacking. In contrast, an individual creates a planned language; to stay alive, it has to *draw a group* of speakers to learn and use it [ibid].

After more than centuries of crafting planned international auxiliary languages, and despite marketing efforts, no viable language has emerged [2]. In contrast, "creolization" of some pidgins has given us several lingua francas.

Proceeding from the hypothesis that natural phenomena are more viable than artificial, we mix the natural phenomenon with open software development approach and present, thus, a methodological vision on how to develop community-driven CNLs (cf Sec.2). As regards the applicability of the approach in the business context, we have started with industrial stakeholders to elaborate the first motivating use cases (cf

Sec.3). These use cases will serve as a basis for the designers of a CNL, providing them with descriptions of sociolinguistic profiles, abilities and requirements of the users.

2 Community-driven CNLs evolution: methodological vision

Supporting the formation of natural diction with open software development techniques, the methodological approach towards a community-driven CNL development is as follows:

1. Attracting the attention of organizations and projects, where CNL could provide business value.
2. Elaborating motivating use cases. In order to map target groups and fields of application, motivating use cases are described. The use cases will broadly meet the requirements stated in the OpenUP process framework: name, brief description and purpose, actor(s), work flows and conditions of the use case will be described [3]. As a result, it is possible to detail the needs and requirements for specific user profiles.
3. For every use case, an initial human-authored corpus of text samples is assembled, a process called Corpus-Based Requirements Analysis [4].
4. Composing the sociolinguistic profiles, abilities and requirements of the user.
5. Adjusting the properties of the CNL according to the framework proposed by Adam Wyner et al. [5] within a number of generic, design, linguistic, and relational properties.
6. Selecting reusable components (software, linguistic assets, etc) from a CNL repository, based on a distance between the properties of the CNL under construction (see Step 5) and available components described by the same framework.
7. Customizing these components to the needs outlined in steps 2-5.

These process-steps will not be carried out in a rigorous order, but in a flexible, iterative manner. The process description itself will be open for improvements by the CNL community – we will make arrangements to publish the process description using a EPF Wiki engine [6], based on Software & Systems Process Engineering Meta-Model (SPEM 2.0) [7].

The aim of such an approach is to (i) foster the exploitation of CNLs by lowering the barrier of creating a CNL and (ii) to speed up the time to market.

3 Motivating use cases

The scenario, functional and non-functional requirements, context, user profiles, user needs and other aspects of our project are explained in a manner familiar to software developers – we use traditional use case technique invented by Ivar Hjalmar Jacobson, widely recommended by software development methodologies, including OpenUP [3].

3.1 UC1 – Information retrieval based on semantic similarity

Main idea: document abstracts and (wiki) article summaries are written in a Controlled Natural Language, enabling semantic search and article recommendation based on the semantic distance of CNL abstracts (for an overview of semantic relatedness, similarity and distance, see [8], for the use of CNLs in wikis, see [9], [10]).

Applicability: search engines, recommendation engines, wiki engines, Electronic Document and Records Management Systems (EDRMS).

Features requested by our clients are:

- A look-ahead editor that goes beyond predicting only by one word – a pattern-based sentence authoring is needed: an annotator selects an intention from a list and a sentence pattern in a CNL is provided by the system, enabling to fill in words from ontologies and to extend the sentence with sub-patterns.
- A possibility to create a mixed content, i.e. a text where a CNL-part is not spatially separated from the main, "uncontrolled" part of the document, rather distributed throughout a text and marked with microformats. This will allow domain experts to start with semi-correct CNL texts and other users more fluent in CNLs to iteratively improve the correctness of the CNL part, while preserving the semantics.
- An editor with real-time named entity recognition (NER) and disambiguation capabilities – a feature rated more useful than consistency checks, for example.

3.2 UC2 – Tagging of digital items

Main idea: Digital items are described by using Controlled Natural Language, enabling cross-lingual information access and delivery. Potentiality to facilitate and speed up the provision of online services centered around computer-based translation.

Applicability: Blog entries, user profiles in social networks, photos, other digital items (i.e. sources like Europeana[1]).

Storyboard. A person looking for digitized photos of old cars from historical archives. Mostly short content description is added in pertinent mother tongue. By writing those descriptions in a CNL, it is possible to make information widely accessible and it is much cheaper than using human-translations into various languages.

3.3 UC3 – Localizing (open) software

Purpose: User interfaces (UI) and documentation of software are available to end-users in their mother tongue, the translation process is smooth, quick, and cost-effective.

Actors: open software developers, localizers.

Main workflow: In the development process of a software product, its UI messages and documentation (i.e. comments, manuals) are written in a CNL (for example, Processable Afrikaans [11]). In the translation/localization process, a pivot

[1] http://www.europeana.eu

pattern/architecture is used, Interlingua de IALA as one possible candidate. In this way, no translation rules are needed for every language pair.

Remarks: These types of CNLs are not FOL-based.

3.4 UC4 – Communication with smart environment

Purpose: Enabling easier/flexible use of smart environment (e.g. home) automation.

Actor: Person controlling automated systems (e.g. inhabitant of a smart home).

Preconditions: A CNL supported automation control unit (ACU) with voice recognition capabilities.

Main Workflow:

1. Actor asks a question or gives an order in a natural speech (e.g. lower the house's internal temperature -2 degrees when alarm is engaged and restore default settings when alarm is disengaged).
2. ACU repeats the command in a CNL.
3. Actor confirms.
4. After the conditions are met (e.g. the alarm is engaged), ACU performs the required actions (e.g. a signal will be sent to a heating room control unit).

We hereby name additional use cases: management of (controlled) vocabularies; creation and management of BPMN[2] schemas; web-service annotation with CNL-based SA-WSDL[3] descriptions, web-service composition.

Conclusions and future work

The paper proposes and presents a methodological vision towards a community-driven CNLs evolution – a paradigm where a group of volunteers develop CNLs based on the principles of bazaar-style open software development methodologies. It is rather a vision than a clear methodology or descriptions of first results.

We call for:

1. Establishing a repository for collecting reusable CNL components: use-case descriptions, software, linguistic assets, etc. The metadata of these assets should take advantage of the framework proposed in [5].
2. Elaborating a "open-source" process description for creating/ customizing CNLs.

[2] Business Process Model and Notation

[3] Semantic Annotations for Web Services Description Language http://www.w3.org/TR/2007/REC-sawsdl-20070828

Acknowledgments

This research was partially supported by the target-financed theme No. EKKTT10-75 of the Estonian Ministry of Education and Research.

The authors would like to thank their industrial partners: Estonian Land Board, ELIKO Competence Centre in Electronics-, Information- and Communications Technologies, and four anonymous reviewers.

References

1. Haitao, L., Creoles, Pidgins, and Planned Languages. Schubert. K. (red.): Planned Languages: From Concept to Reality. Brussel: Hogeschool voor Wetenschap en Kunst, p. 121-177.
2. Olsen, N., Marketing an International Auxiliary Language: Challenges to a New Artificial Language. In Journal of Universal Language, 2003-4, p. 75-89.
3. OpenUP, http://epf.eclipse.org/wikis/openup.
4. Reiter, E., Dale, R., Building Natural Language Generation Systems. Cambridge University Press, 1999. ISBN 0-521-62036-8.
5. Wyner, A., et al. Controlled Natural Languages: Properties and Prospects. Fuchs, N.E. (ed.). In Proceedings of the Workshop on Controlled Natural Languages (CNL 2009), Marettimo Island, Italy, 8-10 June, 2009. LNCS/LNAI, vol. 5972, Springer, 2010.
6. Eclipse Process Framework Wiki, http://epf.eclipse.org/portal/about.
7. Software process engineering metamodel. Version 2.0. Final Adopted Specification formal/2008-04-01. Object management group (OMG), April 2008.
8. Budanitski, A., Lexical Semantic Relatedness and Its application in Natural Language Processing. Technical Report CSRG-390, Dept. of Computer System Science, University of Toronto, 1999.
9. Davis, B., Handschuh, S., Cunningham, H., Tablan, V., Further Use of Controlled Natural Languages for Semantic Annotation of Wikis. In Proceedings of the 1st Semantic Authoring and Annotation Workshop at ISWC2006, Athens, Georgia, USA, November 2006.
10. Bao, J., Smart, P.R., Shadbolt, N., Braines, D. and Jones, G., A Controlled Natural Language Interface for Semantic Media Wiki. In Proceedings of the 3rd Annual Conference of the International Technology Alliance (ACITA'09), 23rd-24th September 2009, Maryland, USA.
11. Pretorius, L., Schwitter, R., Towards Processable Afrikaans. Fuchs, N.E. (ed.). In Proceedings of the Workshop on Controlled Natural Languages (CNL 2009), Marettimo Island, Italy, 8-10 June, 2009. LNCS/LNAI, vol. 5972, Springer, 2010.