Gregor Engels, Regina Hebig, Matthias Tichy (Hrsg.): SE2023,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2023 35

Editing Support for Software Languages: Implementation Practices in Language Server Protocols (Summary)

Djonathan Barros[1], Sven Peldszus[2], Wesley K. G. Assunção[1,3], Thorsten Berger[2,4]

Abstract: We present our paper published at the 25th International Conference on Model Driven Engineering Languages and Systems (MODELS) [Ba22a]. Effectively using software languages requires effective editing support. Modern IDEs, modeling tools, and code editors typically provide sophisticated support to create, comprehend, or modify instancesof particular languages. Unfortunately, building such editing support is challenging. While the engineering of languages is well understood and supported by modern model-driven techniques, there is a lack of engineering principles and best practices for realizing their editing support. We study practices for implementing editing support in so-called language servers—implementations of the language server protocol (LSP). LSP is a recent de facto standard to realize editing support for languages, separated from the editing tools, enhancing the reusability and quality of the editing support. Witnessing the LSP's popularity, we take this opportunity to analyze the implementations of 30 language servers. We identify concerns that developers need to take into account when developing editing support, and we synthesize implementation practices to address them, based on a systematic analysis of the servers' source code. We hope that our results shed light on an important technology for software language engineering, that facilitates language-oriented programming and systems development, including model-driven engineering.

Keywords: Language engineering; code assistance; source code editor; implementation practices

1 Summary

Software languages are paramount—not only to software engineering, but also to many other engineering disciplines that need to create models and automate tasks. Effectively using software languages requires effective editing support. Modern IDEs and modeling tools often come with sophisticated editing support to create, comprehend, and modify programs or models expressed in a certain language. Typical editing support features are code completion, syntax highlighting, error marking, formatting, and refactoring, among others.

Unfortunately, creating proper editing support for languages is difficult. While for mainstream software languages, the vendors of software engineering or modeling tools typically invest the necessary resources to realize editing support, domain-specific languages (DSLs) are often created by smaller organizations or individual developers. Sometimes, their use is limited to specific purposes or only a few projects. As such, DSLs would especially benefit from better support to realize editing support—allowing their users focus on solving real problems, instead of wasting time with learning the exact use of individual DSLs.

[1] PPGComp, Western Paraná State University, Brazil
[2] Ruhr University Bochum, Germany
[3] Johannes Kepler University, Austria
[4] Chalmers | University of Gothenburg, Sweden

The language server protocol (LSP) [Mi] is a recent initiative from 2016 to modularize editing support into the so-called language servers. The LSP addresses the problem that language-specific editing support is currently deeply integrated into single IDEs or editors, preventing its reuse or extension for different tools. The LSP aims at enabling language engineers to make their languages, be it programming languages or DSLs, available to a wide range of editing tools while requiring minimal effort for adoption and reuse. The LSP describes a common API that can be implemented and reused by different clients [Mi].

We took this opportunity and investigated the design and realization of real language servers. Our goal was to identify implementation *concerns* related to the realization of language editing support. Our working hypothesis was that engineering principles can be identified within existing language servers, paving the way to our long-term goal of establishing patterns and pattern catalogs for realizing editing support. We followed a thematic synthesis method by coding, analyzing, grouping, and reporting how editing features are implemented.

We present the first set of engineering practices, systematically identified—quantitatively and qualitatively— from a sample of 30 LSP servers. We synthesized seven core concerns and present practices for realizing language editing support. We found that a variety of features are required to provide editing support, and the concrete aspects of languages play a rather minor role in the basic editing support. Still, for advanced editing support, characteristics of the target language become more important. We hope to provide researchers with concerns related to the realization of editing support and insights on implementation practices for addressing them, to spark a discussion on best practices and eventually developing a theory and novel techniques on realizing effective editing support for languages. We hope that practitioners can use our concerns and discussed solutions as guidelines when implementing new servers or extending existing ones. For designing LSP servers, we identified design and implementation practices that we hope to extend to a reference architecture in later works.

2 Data Availability

The summarized work is available open access in the conference proceedings [Ba22a] and we provide all raw and processed data in our replication package at Zenodo [Ba22b].

Bibliography

[Ba22a] Barros, D.; Peldszus, S.; Assunção, W. K. G.; Berger, T.: Editing Support for Software Languages: Implementation Practices in Language Server Protocols. In: International Conference on Model Driven Engineering Languages and Systems (MODELS). 2022. https://doi.org/10.1145/3550355.3552452.

[Ba22b] Supplementary Material – Editing Support for Software Languages: Implementation Practices in Language Server Protocols 59, https://doi.org/10.5281/zenodo.6974153.

[Mi] Microsoft: Language Server Protocol. https://microsoft.github.io/language-server-protocol/.