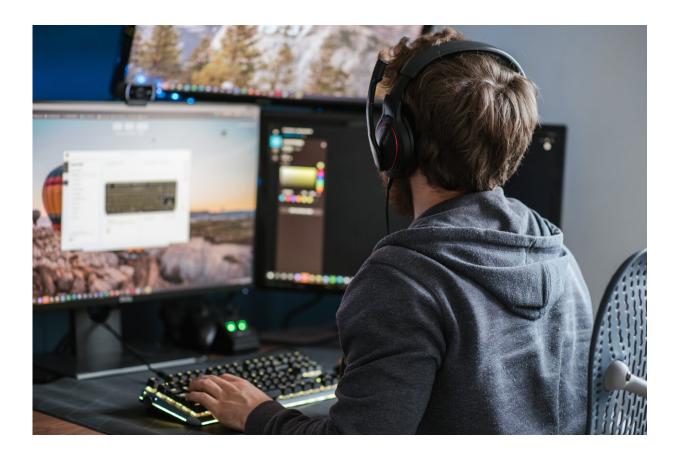# Creating tools to open up programming languages to multiple users

April 29 2022

Credit: Anete Lusina from Pexels

Programming languages are used by a variety of end-users, with different knowledge and experience. These languages usually offer a single user interface or syntax, while it is important to offer different

interfaces depending on the programming task and the background of the users. From the language engineering point of view, offering multiple user interfaces for the same language is expensive. Therefore, Ph.D. candidate Mauricio Verano Merino studied how to engineer different end-user interfaces for Domain Specific Languages (DSLs) in a practical way.

Computational notebooks, block-based environments, and projectional editors allow end-users to create programs, but these are generally not available for domain-specific languages (DSLs). Still, many DSLs would benefit from having these environments available for end-users. This is why Mauricio Verano Merino explored three different practical, yet successful, ways to make them available for DSLs.

Four technological spaces

In his thesis, he offers different mechanisms to engineer multiple programming environments for [end-users](#) of DSLs. These mechanisms rely heavily on reusing existing language components for existing languages or helping language engineers define these interfaces for new languages.

His studies focused on four technological spaces, namely, grammarware (text-based editors), computational notebooks, block-based environments, and projectional editors. A technological space is a shared context that contains a standard body of knowledge, concepts, and hosts different notations. This results in three different language-parametric interfaces for interacting with DSLs, namely computational notebooks, projectional editors, and block-based editors. These interfaces offer users different user experiences to match their needs in using the DSL of their choice.

Grammarware

Different notations are associated with different technological spaces. Consider for instance, grammarware, which is associated with text files, while block-based environments are associated with Blockly and JavaScript files. Therefore, Mauricio Verano Merino had to envision the technological space in question for which the end-user experience had to be built. That way he could offer the language engineers a choice of technological spaces to use. So they in turn could select the most appropriate technological spaces and interfaces for their target audiences.

In order to build these possibilities for language programmers, Mauricio Verano Merino defined grammarware as a common starting point. From this point on, he and his team envisioned different technological spaces (e.g., computational notebooks space, projectional editors space, or block-based space). Based on this idea, he developed three tools that allow language engineers to generate different interfaces for their DSLs, based on a grammar definition of the language.

Benefit users' experience

In his thesis, Mauricio Verano Merino shows that it is possible to generate different programming environments from a single language definition, such as a DSL. The tools he developed also decrease the effort required to create these. This might benefit users' experience towards programming, increase the adoption of domain-specific languages, and reduce engineering efforts.

However, additional research is required to improve the usability of the generated interfaces. This research should also look into ways to make the generation of these interfaces more flexible so that users' data can be

used as part of the generated interfaces.

More information: Engineering Language-Parametric End-User Programming Environments for DSLs. [research.tue.nl/en/publication …](research.tue.nl/en/publication) ramming-environments

Provided by Eindhoven University of Technology