Computer-Aided Software Engineering (CASE) tools have been proposed for decades to assist a myriad of software development activities. CASE tools are meant to assist developers in automating processes that can be costly, time consuming and error prone when manually performed. Recent studies aimed at creating extensive catalogs of CASE tools as well as comparing their performance and/or usability. We are particularly interested in studies focus on tools designed to support the automated detection of different patterns on source code, e.g., code smells, object-oriented design patterns, and refactorings.

Specific studies focus on code smell detection techniques and tools. These studies investigate the state of the art in the field, so that researchers and practitioners could take advantage of existing tools to support their tasks. Dozens of tools have been proposed, targeting different programming languages and code smells. The tools rely on detection techniques that range from textual analysis to graph-based analysis. Unfortunately, only a few tools can actually be used on real software systems, which makes it hard to promote tool adoption in industry. More critically, existing code smell detection tools have been shown inaccurate. As suggested by previous work, existing tools have moderate to low precision and recall rates. Such accuracy may prevent some developers from adopting existing tools in industry settings.

One of the authors of this research paper informally asked a class of master's students on the existence of code smell detection tools. The class is quite diverse in terms of industry experience and constituted of international students. Only a few of them were aware of code smell detection tools, which may be partially due to the fact that only a few tools, e.g., SonarQube, gained popularity in industry. For this reason, this interview-based study targets on the developers' current practices when detecting code smells. We are particularly concerned about understanding to what extent a more recent generation of developers cares about code smells and, if it does, what code smell detection tools are used.

Previous work discusses that developers may be reluctant or too busy to use automated software development tools. This is particularly true in the case of tools designed for changing the internal structure of source code. For instance, the use of refactoring tools has been neglected by developers because developers are afraid that the tools will affect the software functionalities and introduce bugs. In our study, we are particularly interested in understanding why our target developers do not use code smell detection tools specifically.