

# Using the Eclipse C/C++ Development Tooling as a Robust, Fully Functional, Actively Maintained, Open Source C++ Parser

Danila Piatov, Andrea Janes, Alberto Sillitti, Giancarlo Succi

► **To cite this version:**

Danila Piatov, Andrea Janes, Alberto Sillitti, Giancarlo Succi. Using the Eclipse C/C++ Development Tooling as a Robust, Fully Functional, Actively Maintained, Open Source C++ Parser. Imed Hammouda; Björn Lundell; Tommi Mikkonen; Walt Scacchi. 8th International Conference on Open Source Systems (OSS), Sep 2012, Hammamet, Tunisia. Springer, IFIP Advances in Information and Communication Technology, AICT-378, pp.399-399, 2012, Open Source Systems: Long-Term Sustainability. <10.1007/978-3-642-33442-9\_45>. <hal-01519049>

**HAL Id: hal-01519049**

**<https://hal.inria.fr/hal-01519049>**

Submitted on 5 May 2017

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



# Using the Eclipse C/C++ Development Tooling as a Robust, Fully Functional, Actively Maintained, Open Source C++ Parser

Danila Piatov, Andrea Janes, Alberto Sillitti, and Giancarlo Succi

CASE, Free University of Bolzano, Piazza Domenicani 3, Italy  
{danila.piatov, ajanes, asillitti, gsucci}@unibz.it

**Abstract.** Open Source parsers that support contemporary C/C++, can recover from errors, include a preprocessor, and that are actively maintained, are rare. This work describes how to use the parser contained in the Eclipse C/C++ Development Tooling (CDT) as a Java library. Such parser provides not only the abstract syntax tree of the parsed file but also the semantics, i.e., type information and bindings. The authors used the same approach to obtain Java and JavaScript parsers.

Programming language parsers are used by industry and research to create compilers or interpreters, static code analysis tools, code metrics tools, source code editors with code completion, etc.

Parsing C/C++ is particularly tricky (e.g., a construct `a * b` can be a multiplication or a pointer definition depending on the type of `a`). Generic Open Source parser generators do not alleviate this task since the ambiguities cannot be resolved by a parser alone but require type information.

We searched for Open Source C++ parsers that include a preprocessor, perform semantic analysis (resolve type information and name bindings), are robust, and support contemporary C/C++ features. Of the found parsers, namely `cpp-ripper`, `Elsa`, `GCC` using the “`-fdump-translation-unit`” option, `GCC_XML`, `Clang`, and the Eclipse CDT parser only the latter 2 fulfilled the requirements. We decided to opt for the Eclipse parser since the approach could also (and did) provide us with parsers for other languages like Java and JavaScript.

The actual parser is located in the file “`org.eclipse.cdt.core_X.jar`”, in the Eclipse installation folder, where `X` stands for version of the file. The jar itself is an Eclipse plugin, however, it is possible to use it as a Java library, without initializing the Eclipse platform.

The instruction “`org.eclipse.cdt.core.dom.ast.gnu.cpp.GPPLanguage.getDefault().getASTTranslationUnit(FileContent, IScannerInfo, IncludeFileContentProvider, IIndex, int, IParserLogService)`” performs the actual parsing, returning an abstract syntax tree (AST).

This work deals with an (apparently) simple problem: to “find a working C++ parser”. Eclipse CDT contains such parser, but there is no official documentation about using it as a library outside of Eclipse. We hope that our poster can fill this gap and be of help.