Handling `#ifdef` Expressions in cppstats

Claus Hunsen
hunsen@fim.uni-passau.de

September 2014

CPPSTATS was initially developed by Jörg Liebig at University of Passau for a set of studies [LAL+10, LKA11]. In 2013, Claus Hunsen from the same university has taken over development. The most current version of CPPSTATS is available at `https://github.com/clhunsen/cppstats/`. Further information can be found at `http://fosd.net/cppstats`.

1 Introduction

CPPSTATS is a tool for analyzing software systems regarding their variability. Therefore, we focus on software systems written in C using the capabilities of the CPP (the C preprocessor) to express variability. CPPSTATS handles the expressions of the CPP inclusion-guards (`#if`, `#elif`, `#endif`, etc.) in a special way, which is why we provide the used procedure in this document.

The handling of `#ifdef` expressions within CPPSTATS is divided in three parts: *1*) light adaption of the expressions during *preparation* part of CPPSTATS; *2*) collecting the expressions from the SRCML files and rewriting them by making implicit tangling explicit; and *3*) building a global expression pool for the analyzed software-project.

2 Example

As `#ifdef`s are explained best by means of an example, the three common pattern of CPP usage are shown in Figure 1. Part (a) shows nesting of `#ifdef`s, while Part (b) and (c) show the use of `#else` and `#elif` branches in correspondence to a single `#ifdef`.

During this document, the reader is referenced to these small examples to illustrate all matters.

```
1 #ifdef A
2   #ifdef B
3   #endif
4 #endif
```

(a) A nested `#ifdef` in file `X`.

```
5 #ifdef A
6 #else
7 #endif
```

(b) An `#else` branch in file `Y`.

```
8 #ifdef B && C
9 #elif D
10 #endif
```

(c) An `#elif` branch in file `Z`.

Figure 1: Short examples of the patterns that occur while using CPP and that are treated by CPPSTATS. Each example and their rewriting rules are explained in this very document.

3 Source-Code Preparation to srcML files

4 Collection of Expressions During File Analysis

5 Global Expression Pool

References

[LAL+10] Jörg Liebig, Sven Apel, Christian Lengauer, Christian Kästner, and Michael Schulze. An Analysis of the Variability in Forty Preprocessor-Based Software Product Lines. In *Proc. Int. Conf. Software Engineering (ICSE)*, pages 105–114. ACM, 2010.

[LKA11] Jörg Liebig, Christian Kästner, and Sven Apel. Analyzing the Discipline of Preprocessor Annotations in 30 Million Lines of C Code. In *Proc. Int. Conf. Aspect-Oriented Software Development (AOSD)*, pages 191–202. ACM, 2011.