生成C++语法树通常使用的是一种名为抽象语法树(Abstract Syntax Tree, AST)的数据结构。生成AST的过程通常需要使用词法分析器和语法分析器。

词法分析器将源码转换成一个个的标记，然后语法分析器据此构建出AST。在C++中，你可以使用如ANTLR、Bison等工具来生成词法分析器和语法分析器。

下面是一个简单的例子，使用ANTLR v4来生成C++的AST。

1. 首先，需要定义语法规则。ANTLR v4为C++提供了一套预定义的语法规则，可以在[这里](https://github.com/antlr/grammars-v4/blob/master/cpp/CPP14.g4)找到。将这个文件保存为CPP14.g4。
2. 然后，可以使用以下命令来生成词法分析器和语法分析器的代码：

antlr4 -Dlanguage=Cpp CPP14.g4

1. 在生成的代码中，CPP14Parser类就是语法分析器。下面是一个简单的例子，展示如何使用它来生成AST：

#include "CPP14Lexer.h"

#include "CPP14Parser.h"

#include <antlr4-runtime.h>

int main() {

std::ifstream stream("test.cpp");

antlr4::ANTLRInputStream input(stream);

CPP14Lexer lexer(&input);

antlr4::CommonTokenStream tokens(&lexer);

CPP14Parser parser(&tokens);

antlr4::tree::ParseTree\* tree = parser.translationunit();

*// ...*

return 0;

}

1. 在这个例子中，test.cpp是你想要解析的C源码文件，translationunit是C语法中的一个规则，代表了一个完整的C++程序。

这个过程只是生成AST的第一步，你还需要遍历AST并根据需要进行操作。ANTLR v4提供了ParseTreeWalker类来遍历AST，你也可以实现自己的访问者类。