Overview of C++: Brief History

Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

Professor of Computer Science
Institute for Software Integrated Systems
Vanderbilt University
Nashville, Tennessee, USA
Overview of C++

Learning Objectives in this Part of the Lesson

- Recognize the key components of C++
- Know strategies for learning C++
- Understand C++ design goals
- Learn about conflicts of C++ design goals
- Be aware of the history of C++
Overview of C++ History
• C++ was designed at AT&T Bell Labs by Bjarne Stroustrup in the early 80’s

See [www.softwarepreservation.org/projects/c_plus_plus](http://www.softwarepreservation.org/projects/c_plus_plus)
C++ History in a Nutshell

- C++ was designed at AT&T Bell Labs by Bjarne Stroustrup in the early 80’s
- It originated from the confluence of object-oriented programming languages with systems programming languages
C++: From Translator to Native Compiler

• The original cfront translated C++ into C for portability

Cfront

From Wikipedia, the free encyclopedia

Cfront was the original compiler for C++ (then known as "C with Classes") from around 1983, which converted C++ to C; developed by Bjarne Stroustrup at AT&T Bell Labs. The preprocessor did not understand all of the language and much of the code was written via translations. Cfront had a complete parser, built symbol tables, and built a tree for each class, function, etc. Cfront was based on CPre (C compiler, which was started in 1979).

As Cfront was written in C++, it was a challenge to bootstrap on a machine without a C++ compiler/translator. Along with the Cfront C++ sources, a special "half-preprocessed" version of the C code resulting from compiling Cfront with itself was also provided. This C code was to be compiled with the native C compiler, and the resulting executable could then be used to compile the Cfront C++ sources.

Most of the porting effort in getting Cfront running on a new machine was related to standard I/O. Cfront's C++ streams were closely tied in with the C library's buffered I/O streams, but there was little interaction with the rest of the C environment. The compiler could be ported to most System V derivatives without many changes, but BSD-based systems usually had many more variations in their C libraries and associated stdio structures.

See en.wikipedia.org/wiki/Cfront
C++: From Translator to Native Compiler

• The original cfront translated C++ into C for portability

```cpp
class Base {
private:
    int private_data;
protected:
    int protected_data;
public:
    int public_data;

    Base (int arg = 1)
    : private_data (arg + 100),
      protected_data (arg + 10),
      public_data (arg) {}
};
```

```cpp
struct Base {
    int private_data__4Base;
    int protected_data__4Base;
    int public_data__4Base;
    struct __mptr *__vptr__4Base;
};
```
C++: From Translator to Native Compiler

• The original cfront translated C++ into C for portability

```c
static struct Base *
__ct__4BaseFi (struct Base *__0this, int __0arg) {
    if (__0this ||
        (__0this =
            (struct Base *) __nw__FUi (sizeof (struct Base))))
    {
        ((__0this->)__vptr__4Base =
            (struct __mptr *) __ptbl_vec_c___src_C_[0]),
        (__0this->)private_data__4Base =
            (__0arg + 1000)),
        (__0this->)protected_data__4Base =
            (__0arg + 100)),
        (__0this->)public_data__4Base =
            __0arg));
    return __0this;
}
```
C++: From Translator to Native Compiler

- The original cfront translated C++ into C for portability
- However, cfront output was hard to debug & was often inefficient for more advanced features
• The original cfront translated C++ into C for portability
  • However, cfront output was hard to debug & was often inefficient for more advanced features
• *Many* native host machine compilers now exist
  • e.g., GNU G++, LLVM Clang, IBM C++ Compiler, Microsoft Visual C++, Oracle Developer Studio, C++ Builder, etc.

See [www.stroustrup.com/compilers.html](http://www.stroustrup.com/compilers.html)
C++ Has Continued to Evolve

- C++ has continually evolved throughout its lifetime to address new programming paradigms, challenges, & hardware platforms.

See en.wikipedia.org/wiki/C++#History
End of Overview of C++: Brief History