Roger WG14

JTC1/SC22/WG21/N0085
X3J16/92-0007

Position Paper
Extending C with the C++ Linkage Specification

AFNOR–C++ Experts Group

November 1991

The goal of this paper is to propose an extension to the C language with the C++ linkage directive `"extern "C"` in order to share C and C++ header files without preprocessor support.

Sharing C and C++ Header Files

Our starting point will be a simple C++ program including a standard header file:

```
# include <string.h>
```

Our concern is to determine what file is included, and more precisely, where the file `string.h` is localized. An immediate objection to this request is that this issue is not a standardization issue since part of the implementation. But closer examination leads to reconsider this objection.

First, let us assume that we can rely on a set of ISO-C header files, with the declaration of function prototypes. For example:

```
char *strcpy(char*, const char*);    /* C and C++ */
```

Localization of these header files may obey two alternatives:

- share the same ISO-C files between any C and C++ programs
- use separate copies of the header files

The only difference[1] between a C header file and a C++ header file is the C++ linkage directive to defeat C++ name mangling. Sharing files appears as the reasonable solution but requires the support of preprocessor conditional directives. Different schemes are commonly used, for example:

[1] We exclude there "local" differences, such as the null pointer.

```
#if defined(__cplusplus)
    extern "C" {
# endif

        // declarations

#if defined(__cplusplus)
    }
# endif
```

A previous recommandation of X3J16 was to discourage the use of preprocessor support and to reduce conditional compilation of source code. A solution suitable for C++ should be to *duplicate* the header files. Perfect world (at least for vendors of both C and C++ compilers) should be to *share* the header files, as such, without distinction. Our proposal aims at solving this dilemma.

Proposal

In order to share a common set of header files between C and C++, the AFNOR–C++ Experts Group asks SC22/WG14 to consider the introduction of the C++ linkage directive "`extern "C"`" as part of the C language.

The directive is already specified and documented in chapter 7 of document X3J16/91-0115 (SC22/WG21/N0048). Grammar extension is minimal since it introduces only two rules to the C++ grammar:

```
linkage-specification:
    extern string-literal { declaration-list }
    extern string-literal  declaration
```

The meaning of *string-literal* is implementation dependent. C++ requires that the linkage to C and C++ must be at least provided by every implementation. The AFNOR–C++ Group wishes that the linkage to C be at least provided by every implementation of C, but has not examined what might be the meaning of the linkage to other languages in the context of the C language.