Project: WG21/X3J16
Document Number: N1166/98-0023
Written by: Nathan Myers
ncm@cantrip.org
Date: 1998 October 6

Extension Namespace Recommendations

It is routine for other standards bodies to define extensions declared in standard headers. Vendors are obliged to implement these extensions regardless of their effect on C++ Standard conformance. Typically the extensions are defined for C only, and implementors must extrapolate those definitions to their C++ products. In any case, these extensions create difficulties for standards (plural) conformance, particularly when different standards bodies (or the same bodies at different times) define different semantics for the same names in the same header.

For example, the C9x group is considering a global name "clog" even though C++ already uses that name scoped as "std::clog". Since the namespace directives can easily throw these names into conflict, implementors will be confronted with difficult choices. The POSIX and X/Open groups define a variety of names specified to appear in standard headers, which for conformance must currently be guarded by #if blocks which rarely provide much actual help to users. In <time.h>, the semantics different groups specify for the same names are incompatible. Standards bodies sometimes change the interfaces or semantics of their own previous extensions.

While WG21 (obviously?) cannot specify non-standard interfaces, we can offer guidance to other standards bodies and (perhaps more importantly) to implementors translating standards to the C++ environment, to help avoid conflicts now and in the future. The immediate moment offers a unique opportunity to prevent conflict, rather than try to ameliorate it later, by recommending a pattern for namespaces to encapsulate extensions. This would be especially helpful for vendors adopting C extensions, because otherwise they must choose whether to make the names global, as in C, or put them in namespace std like the other names, or choose another namespace unlikely to match other vendors' choices.

The essence of this proposal is to resolve to issue a short technical report, as soon as possible, recommending a pattern for namespace encapsulation of library extensions, and to outline what the report might include.

For established standards we can recommend specific names; for others we can recommend a pattern that vendors can use when adopting (or defining their own) extensions. Given adherence to a well-chosen pattern, the actual name a vendor or committee chooses for names we do not specify is less important than might be assumed, because namespace aliasing allows many conflicts to be resolved.

For a hypothetical example, consider the C89 standard. The C++ standard adopts most of its names into the std namespace, but there are some differences. The C++ standard does not specify whether they are extern "C" or "C++", and it defines replacements for some, such as strchr. Suppose a vendor had a need to provide, in addition to the C++ definitions, access to the actual C declarations -- declared extern "C" with C calling conventions. These could certainly be wrapped in a namespace, but what would it be called?

The pattern we suggest for the namespace name could be simple: the standards body name with a date stamp, followed by an alias lacking the date stamp, such as:

```
  namespace _Iso_c_1989 {
    extern "C" char* strchr(char const* s, char const* p);
```

```
  }
  namespace _Iso_c = _Iso_c_1989;
```

The alias allows later revisions of the standard to be adopted as the default, as is likely for the case of ISO C. Names from previous standards can be adopted into each subsequent version, or shadowed. We can further recommend that preprocessor macros in C standards that can usefully be defined as properly-scoped C++ constants or inline functions should be.

Of course, the above is just an example; details of the recommended pattern will be worked out more fully in the proposal for the technical report text. This proposal is simply for the committee to resolve to publish a technical report as outlined above, and to request a detailed proposal of its contents.