Document Number: J16/04-0056 = WG21 N1616
Date: 2004-03-26
Reply To: William M. Miller
The MathWorks, Inc.
wmm@world.std.com

Extended `friend` Declarations (Rev. 1)

I. Introduction

The earlier version of this document (J16/03-0103 = WG21 N1520) described the need to extend the current language to support a wider range of `friend` declarations, in particular, `friend` declarations in which the befriended class is named by a template parameter. Currently, `friend` declarations that name a class must do so via an *elaborated-type-specifier*, and *elaborated-type-specifier*s cannot refer to a template parameter.

The earlier paper proposed specific wording for changing the Working Paper; however, because of a lack of consensus during preliminary discussions, it suggested three different alternatives and made no recommendation. The Evolution Working Group has now agreed on one of those alternatives. Furthermore, comments during the Core Working Group's consideration of the paper suggested that one intended feature, although implied by the normative wording, would benefit from an explanatory note.

The purpose of this paper is to present the proposed wording change in a coherent form for final review before incorporation into the Working Paper.

II. Proposed Wording

1. Change 9.2¶7 from

> The *member-declarator-list* can be omitted only after a *class-specifier*, an *enum-specifier*, or a *decl-specifier-seq* of the form `friend` *elaborated-type-specifier*.

to

> The *member-declarator-list* can be omitted only after a *class-specifier* or an *enum-specifier* or in a `friend` declaration (11.4).

2. Delete the following wording from 11.4¶2:

> An *elaborated-type-specifier* shall be used in a friend declaration for a class. *[Footnote:* The *class-key* of the *elaborated-type-specifier* is required.*]*

3. Add the following as a new paragraph following 11.4¶2:

A `friend` declaration that does not declare a function shall have one of the following forms:

> `friend` *elaborated-type-specifier* `;`
> `friend` *simple-type-specifier* `;`
> `friend` *typename-specifier* `;`

[*Note:* a `friend` declaration may be the *declaration* in a *template-declaration* (clause 14, 14.5.3).] A `friend` declaration whose type specifier designates a non-class type is ignored. [*Example:*

```
class C;
typedef C Ct;

class X1 {
   friend C;           // OK: class C is a friend
};

class X2 {
   friend Ct;          // OK: class C is a friend
   friend D;           // error: no type-name D in scope
   friend class D;     // OK: elaborated-type-specifier declares new class
};

template <typename T> class R {
   friend T;
};

R<C> rc;               // class C is a friend of R<C>
R<int> Ri;             // OK: “friend int;” is ignored
```

—*end example*]

[Drafting note: The note in the added text above is to make clear that the syntactic restrictions do not prevent a `friend` *declaration from occurring in a template-declaration, e.g.,*

```
class X {
   template <typename T> friend class Y;
};
```

— end drafting note]