Document No.: J16/98-0024 = WG21/N1167
Date: 26 October, 1998
Reply to: William M. Miller
`wmm1@flash.net`

Proposed Editorial Changes for Core Language Issues

3.3.6¶1 (core issue 42):

Current wording:

> A name `N` used in a class `S` shall refer to the same declaration in its context and when re-evaluated in the completed scope of `S`.

Proposed wording:

> A name `N` used in a class `S` shall refer to the same declaration when looked up in its lexical position (ignoring declarations that follow it) and in the completed scope of `S`.

3.4.1¶6 (core issue 41):

Current wording:

> A name used in the definition of a function *[footnote:* This refers to unqualified names following the function declarator; such a name may be used as a type or as a default argument name in the *parameter-declaration-clause*, or may be used in the function body. *end footnote]*...

Proposed wording:

> A name used in the definition of a function following the function's *declarator-id [footnote:* This refers to unqualified names used, for instance, in a type or default argument expression in the *parameter-declaration-clause* or used in the function body. *end footnote]*...

3.4.1¶8 (core issue 41):

Current wording:

> A name used in the definition of a function that is a member function (9.3) *[footnote:* That is, an unqualified name following the function declarator; such a name may be used as a type or as a default argument name in the *parameter-declaration-clause*, or may be used in the function body, or, if the function is a

constructor, may be used in the expression of a mem-`initializer`. *end footnote]* of class `X`...

Proposed wording:

> A name used in the definition of a member function (9.3) of class `X` following the function's *declarator-id [footnote:* That is, an unqualified name used, for instance, in a type or default argument expression in the *parameter-declaration-clause*, in the function body, or in an expression of a *mem-initializer* in a constructor definition. *end footnote]*...

3.4.2¶2 (core issue 33):

Add following the last bullet in the list of associated classes and namespaces for various argument types:

> In addition, if the argument is the name of a set of overloaded functions and/or template functions, its associated classes and namespaces are the union of those associated with each of the members of the set: the namespace in which the function or template function is defined and the classes and namespaces associated with its (non-dependent) parameter types and return type.