Accredited Standards Committee*
X3, INFORMATION PROCESSING SYSTEMS

Doc No: X3J16/95–0031
WG21/N0631
Date: 31 January 1995
Project: Programming Language C++
Reply to: Andrew Koenig
AT&T Bell Laboratories
PO Box 636
600 Mountain Avenue
Room 2C–306
Murray Hill, NJ 07974 USA
ark@research.att.com

Relaxing restrictions on overloading

WP Clause 13.1, paragraph 2 presently contains the following text:

— Since, for any type "`T`," a parameter of type "`T`" and a parameter of type "reference to `T`" accept the same set of initializer values, function declarations with parameter types differeng only in this respect cannot be overloaded.

There is a box after that paragraph saying: ''This restriction is hard to check across translation units. Moreover, ambiguities can be detected just fine at call time. Perhaps we should remove it.''

I propose to remove the restriction by deleting from paragraph 2 the text beginning ''— Since, for any type "`T`," a parameter'' through the text ending ''declarations that differ only in this respect can be overloaded.'' My rationale:

— The statement was true when written, because it dates back to when it was possible to bind a reference to an rvalue. However, it is not true any more. For example, a parameter of type `int` can accept an argument `3` and a parameter of type `int&` cannot.

— It is always possible to disambiguate between variants of an overloaded function: use the function to initialize a variable with the appropriate pointer-to-function type.

— The restriction does not remove the potential for ambiguity. For example, `f(short)` and `f(long)` will yield an ambiguity when called as `f(3)`.

— I do not know how to revise the rule so as to make it do ''the right thing,'' especially in the presence of templates or namespaces. In the absence of a concrete proposal for how to correct it, I would rather delete it altogether.

Of course, I am not proposing to abandon ambiguity checking. I just don't see much merit in checking a subset of possible ambiguities at the point of declaration and then checking again for them all at the point of use. People who try to do ambiguous things will still be stopped. I think it's important, though, for people to be able to combine contexts, even if they contain potentially ambiguous overloaded functions.

---

* *Operating under the procedures of the American National Standards Institute (ANSI)*
Standards Secretariat: CBEMA, 1250 Eye Street NW, Suite 200, Washington DC 20005