Completing Boolean (revision 1)

Doc No: **X3J16/94-0045R1**
WG21/N0432R1
Date: **May 30, 1995**
Project: Programming Language C++
Reply-To: Neal M Gafter
neal@cs.rochester.edu

1 Introduction

(1) With the introduction of `bool` (and its constants `true` and `false`) into the C++ language, it seems we have actually lost some of the usual operations available on boolean types. We used to write code like this:

```
extern bool f(), g();

main()
{
        if (f() | g()) controlled_statement();
}
```

(2) The intent was clear: evaluate both `f()` and `g()` and execute the controlled statement if either one returned `true`. The same argument applies to the `&` and (perhaps more importantly) `^` operators.

(3) On the other hand, one might argue that it is not necessary to extend these bitwise operators to the `bool` type because their current definitions make the above code work just fine. That is, the values returned by `f()` and `g()` are converted to `int`, the bitwise operator is applied, and the result converted to `bool` for the `if` statement. I have the following objections to this logic:

- I would like compilers to warn about the automatic conversion of an `int` to a `bool`. However, I don't want a warning in this context. Similarly, warnings about automatic conversion from `bool` to `int` are reasonable except in this context.
- It would be nice to be able to "correctly" overload on an expression like this.

(4) Similarly, the assignment operators should be defined for `bool`. Moreover, it would be nice to have assignment versions of the logical operators.

2 Define `&`, `|`, `^`, `==`, and `!=` for `bool`

(1) Add the following to the beginning of the description of the bitwise and operator (5.11 [expr.bit.and]):

> If both operands are of type `bool`, the result is of type `bool` and is `true` only if both operands are `true`. Otherwise, ...

(2) Add the following to the beginning of the description of the bitwise exclusive or operator (5.12 [expr.xor]):

> If both operands are of type `bool`, the result is of type `bool` and is `true` only if the operands are unequal. Otherwise, ...

(3) Add the following to the beginning of the description of the bitwise or operator (5.11 [expr.or]):

> If both operands are of type `bool`, the result is of type `bool` and is `false` only if both operands are `false`. Otherwise, ...

(4) Similar text should be added to the relational operators so that the integral promotions are avoided.

3 Define `&=`, `|=`, and `^=` for `bool`

(1) The Working Paper description of the semantics of assignment operators (5.17 [expr.ass] paragraph 7) contains

> `E1` shall not have `bool` type.

(2) I propose to remove the above quoted phrase from the text of the Working Paper. The bitwise or-assignment operator is particularly useful, given that increment is deprecated for the type bool.