Do not promise support for function syntax of operators

From Hyrum's Law [1]:

With a sufficient number of users of an API, it does not matter what you promise in the contract: all observable behaviors of your system will be depended on by somebody.

Proposal

In the Library Evolution Working Group (LEWG) room of the Kona 2019 ISO C++ meeting, it was mentioned that the only supported way to invoke operators on types in the standard library is operator syntax. Function syntax is not supported. This paper proposes updating SD-8 to convey this information to users of the standard library.

For clarity...

Unsupported: \texttt{x.operator+(y)}, \texttt{operator+(x, y)}

Supported: \texttt{x + y}

Wording option 1

Primarily, the standard reserves the right to:

[...]

* Make changes to existing interfaces in a fashion that will be backward compatible, if those interfaces are solely used to instantiate types and invoke functions. Implementation details (the primary name of a type, the implementation details for a function callable +, the way an operator is or is not overloaded ) may not be depended upon.

[...]

Wording option 2

Primarily, the standard reserves the right to:

[...]

+ * Change the way operators are defined on types in the standard library + (between not overloaded, member overloaded and non-member overloaded).

[...]
Wording option 3

Primarily, the standard reserves the right to:

[...]

+ * Assume operators are only invoked using operator syntax (not function syntax).

[...]