Substantive issues:

Public comment 28 (5.3.4p16): If nothrow operator new is called and returns NULL, initialization should not be done (and the deallocation function should not be called). Applies to all signatures that contain a nothrow parameter, and not just those defined by the WP; also includes user replacements for such functions.

Core issue 774: Add 5.2.9p10 (like 5.2.10p7) making it clear that static_cast of pointer-to-object to void* and back again gives the original pointer.

No issue number: 5.2.11p2 should make it clear that const_cast to same type is okay only for types otherwise acceptable for const_cast, i.e., pointer, pointer-to-member, reference (and not, e.g., float --> float).

Core issue 778: 13.3.3 should indicate that the "conversion sequence" on the implicit object parameter for a static member function is no better, no worse than other conversion sequences (and therefore is never the deciding factor in selecting one function over another).

Paper 97-0012/N1050: Implicit use of conversion function to convert a class rvalue to an lvalue to which a reference is then bound directly should be okay.

Core issue 683: When an enumerator constant is used before the closing "}" of its enumeration it should have the type of the initializing expression (alternative (1) in the core issue).

Core issue 756, 734, 682: The behavior of the "?" operator prototype in 13.6 is not right.

(a) Many class cases are ambiguous because of 13.6p28 (this is issue 756).
(b) Cases like the following are ambiguous because of duplication between the signatures:

```cpp
struct A { operator int(); } a;
1 ? a : 1; // Ambiguous: operator?(bool, int, int) and operator?(bool, const int&, const int&)
```

We have no proposed resolution yet.

Editorial Issues:

Public comment 13: agree. "unsuccessful" after overload resolution in 13.3.1.2p9 means "no viable functions found" and does not include ambiguity.

Public comment 16: clarify that local classes have the same access to a containing class as does the containing function (i.e., they're not the same as nested classes).

Public comment 23 (5-23): WP is clear enough, but add example showing composite pointer type.

Public comment 28 (18.1p4): trivial editorial change ("null-pointer constant"
to "null pointer constant").

Public comment 36: Add suggested wording noting that main cannot be called recursively (3.6.1 says this already).

Core issue 773: Make clear in 4.2p2 that the deprecated string-to-char * standard conversion is done only when there is an explicit destination type of char *.

No issue number: 5.7p1 should make it clear that in pointer+-enum, the enum is treated as if it has the underlying integral type.

Items for which we recommend no action:

Public comment 29 (item 7): disagree. We think passing a non-POD to an ellipsis should remain undefined behavior. Lots of code does this.

Public comment 29 (item 8): disagree. Casting between pointer-to-signed and pointer-to-unsigned should remain a reinterpret_cast and not a static_cast.

Core issue 718: old-style cast between pointers to incomplete class types should remain unspecified with regard to the choice of static_cast/reinterpret_cast interpretation.

Core issue 775: static_cast from pointer to struct to pointer to first member should remain invalid. Use reinterpret_cast (should this be made well-defined under reinterpret_cast?).

(end of list)