

Doc No: SC22/WG21/N1864
J16/05-0124
Date: Aug 22, 2005
Project: JTC1.22.32
Reply to: Herb Sutter
Microsoft Corp.
1 Microsoft Way
Redmond WA USA 98052
Email: hsutter@microsoft.com

TG5 Liaison Report #10

No face-to-face meeting of Ecma TC39/TG5 (C++/CLI) has been held since March 2005. However, phone meetings have been used to review and approve numerous changes to the draft.

The following TG5 documents are attached to this liaison report:

- TC39-TG5/2005/016 *Working draft 1.11 of the C++/CLI Standard, Language Specification, April 2005
- TC39-TG5/2005/017 Project Editor's Report, April 2005
- TC39-TG5/2005/018 C++/CLI Specification Comments - revision 28 April 2005
- TC39-TG5/2005/019 *Working draft 1.12 of the C++/CLI Standard, Language Specification, May 2005
- TC39-TG5/2005/020 C++/CLI Specification Comments - revision 7 July 2005
- TC39-TG5/2005/021 *Working draft 1.13 of the C++/CLI Standard, Language Specification, July 2005
- TC39-TG5/2005/022 Agenda for the 11th meeting of TC39-TG5, Redmond, September 2005
- TC39-TG5/2005/023 C++/CLI Specification Comments - revision 22 August 2005
- TC39-TG5/2005/024 *Working draft 1.14 of the C++/CLI Standard, Language Specification, August 2005

*Documents TC39-TG5/2005/016, 019, and 021 were intermediate committee drafts of the specification and are not included here. They are superseded by document 024, which can be found at the following URLs:

<http://www.plumhall.com/ecma/index.html>
<http://go.microsoft.com/fwlink/?LinkId=50042>

This is a replacement/place-holder for Documents TC39-TG5/2005/016, 019, 021, and 024. Documents 016, 019, and 021 were intermediate committee drafts of the specification, and are not included here. They are superseded by document 024, which can be found at the following URLs:

<http://www.plumhall.com/ecma/index.html>

<http://go.microsoft.com/fwlink/?LinkId=50042>

2005-04 Project Editor's Report

Rex Jaeschke

Ecma TC39 TG5 project editor

rex@RexJaeschke.com

+1 703 860-0091

Working Draft 1.11 has been produced and distributed. The following work went into producing it:

1. I applied corrections resulting from the Hawaii March meeting.
2. I applied corrections arising from the reviews by Dinkumware, Jon Caves, and Andy Rich.
3. I incorporated some of Brandon's new postings.
4. I made a pass over the whole draft, making many copy edits and corrections to examples; only the non-trivial edits were tracked.
5. People continue to dislike the subtle distinction between "array" and "Array". As such, I have changed "Array" to "CLI array".
6. The contents of Annex E, "CLI naming guidelines" has been replaced with a URL.

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
2	7-Oct-03	Rex Jaeschke		Technical		P.J. Plauger	The current CLI spec supports Unicode V3.0. What, if anything, should we do w.r.t V3.1/V4.0?	Brought up during the phone meeting of 10/7/2003. Meeting #4 (NJ): Take no action. Don't mention more that necessary.	Yes	
3	7-Oct-03	Tom Plum		Technical		Tom Plum	Diagnostics: How should we deal with warnings and such?	Meeting #3 (Melbourne): Tom will adapt text from the C# spec and present it. Meeting #4 (NJ): Withdrawn without action.	Yes	
4	10-Oct-03	Phone meeting		Editorial		Editor	Future directions: Should there be an informative annex listing future directions? Possible entries are: 1. Supporting static members in interfaces 2. Mixed types 3. gcnew of unmanaged types 4. new of managed types		Yes	
5	10-Oct-03	Tom Plum		Technical		Tom Plum	While discussing enums (25.1.3) and wchar_t's not being permitted as an underlying type, a discussion arose w.r.t CLI's requiring wchar_t to have the same representation as System::Char; that is, a 16-bit character. This needs further investigation. Possible need to look at/point to the PDTR currently out from WG11 (ISO C). This is part of a more general issue. Do we require exact mapping for types, or do we allow a certain amount of flexibility? See issue #93.	In email on 2003-10-12 Tom Plum wrote: Refining my comments re wchar_t, I see a short-term and a long-term ... Short-term, there's no need to change anything. The 16-bit unicode type is wchar_t in VC++ and in C++/CLI. Long-term, the decision is up to TG5, and depends upon who participates. My own guess is that TG5 in fact will be the first group that has to integrate Unicode 3.1 and 4.0 into its language definition. I suspect that before we're done we'll have four types of character (and literal and C++ string): char - has to be 8 bits to integrate with CLI 'x' "str" string = basic_string<char> wchar_t - implementation's legacy choice of widechar L'x' L"str" wstring = basic_string<wchar_t> char16_t - 16-bit character type, has to be UCS-2 or UTF-16 for CLI u'x' u"str" ustring (?) = basic_string<char16_t> (or string16?) char32_t - 32-bit character type, has to be UTF-32 for CLI U'x' U"str" Ustring (?) = basic_string<char32_t> (or string32?) wchar_t can be the same type as char16_t or	Yes	
6	10-Oct-03	Phone meeting		Technical		Brandon Bray	Issue of mapping system value types to the fundamental types, and interop with the standard library.	Merged in with issue #93	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
7	21-Oct-03	Rex Jaeschke	7	Technical		P.J. Plauger	What is the interaction between the standard I/O streams and System::Console?	Meeting #3 (Melbourne): It appears that there will not be any synchronization between the two. Meeting #8 (WA): Decided to say nothing about this.	Yes	
8	4-Dec-03	meeting #1 (TX)	12.1.1	Technical		Steve Adamczyk	64-bit integer mapping. Meeting #1 (TX): Steve to write a paper for Jan 04 meeting. Done.	Meeting #2 (HI): This paper will be presented at the March meeting of WG21. Let's see how it is received? Meeting #4 (NJ): Steve will suggest how to tighten existing wording w.r.t a 64-bit integer type in the current draft, as part of the cleanup for the public drop. As to how to document the library support has yet to be determined.	Yes	
9	4-Dec-03	meeting #1 (TX)		Technical		Brandon Bray	Write a paper on "It just works"		Yes	
10	4-Dec-03	meeting #1 (TX)	14	Technical	R	Brandon Bray	pull together all the conversion information into one place. Make sure all conversions are covered.		Yes	
11	4-Dec-03	meeting #1 (TX)	15.3.2	Technical		Steve Adamczyk	comma vs. semicolon as separator in indexed access expressions In indexed access expressions (§15.3.2), comma operators are currently disallowed inside [] unless they are enclosed in parentheses. This conflicts with usage in existing template libraries (e.g., Lambda), in which the comma operator occurs inside [] without enclosing it in parentheses.	Meeting #2 (HI): Can we treat commas in [] not having enclosing parenthesis, in any context, always be treated as punctuators? Yes. Steve will provide words to the editor for this. Meeting #3 (Mel): Steve produced a paper. He reported one outstanding issue: In 15.3.2, "Indexed Access", in the C++/CLI spec is rather vague. There, we have indexed-access: indexed-designator [expression-list] where indexed-access is defined as an additional alternative for postfix-expression: postfix-expression: indexed-access Unfortunately, there isn't any definition of indexed-designator, so I'm not quite sure whether all the multi-dimensional cases are supposed be handled by indexed-designator, leaving the traditional cases to be handled by the original (possibly modified) syntax. An alternative would be not to introduce indexed-access at all, and use the definition postfix-expression: postfix-expression [expression-list] to handle all the cases, for both traditional subscripting and the new C++/CLI indexer references. There was agreement to this, so Steve will update his p	yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
12	4-Dec-03	meeting #1 (TX)	9	Technical		Tom Plum	Issue of source code/Unicode mapping. What assumptions, if any, should we make about the form of input text? Handling of string literals, character constants, and comments.	Meeting #3 (Melbourne): Had a short discussion. Tom will produce a paper for the May meeting. Meeting #4 (NJ): Tom got more input at this meeting, and will produce a paper for the Jun meeting. DONE (see email "TG5 issue #12 - character sets" from 5/29 EDT) Meeting #5 (Redmond): Discussed Tom's paper in detail. He'll update and recirculate. Meeting #6 (Redmond): Closed out this issue with the string literal portion of this issue being transferred to	Yes	
13	4-Dec-03	meeting #1 (TX)	12	Technical	M	Brandon Bray	Add a diagram of the type tree		Yes	
14	5-Dec-03	meeting #1 (TX)	15.3.9	Technical		Editor	alternative syntax for typeid <type-id> The current syntax typeid <type-id> is too close to the Standard C++ forms.	Meeting #2 (Hawaii): Ownership of this issue transferred from John to Herb. Several alternatives were discussed, including a keyword CLI_typeid or CLI_typeof, and a static member .class ala Java. Also ::typeid. Herb addressed this in his keywords paper, which was adopted in Melbourne.	Yes	
15	5-Dec-03	meeting #1 (TX)	16.1.1	Technical		Tom Plum	Write a paper for Jan, 04, meeting on use of for-each with STL types. TG5 will not pursue this as it's part of the work being considered by WG21's evolution group.		Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
16	5-Dec-03	meeting #1 (TX)	16.1.1	Technical		P.J. Plauger	<p>The for each statement.</p> <p>Meeting #1 (Texas): Write a paper for Jan, 04, meeting on spelling "for each" simply as "for".</p>	<p>Meeting #2 (Hawaii): Tom presented his proposal from his email entitled {"for" in the style of "for each"} from January 28. A discussion ensued, during which the following alternatives (the colon versions of which were new) were discussed in detail:</p> <ol style="list-style-type: none"> 1. for each (type var in coll) 2. for (type var in coll) 3. for each (type var : coll) 4. for (type var : coll) <p>A straw poll indicated a preference for the alternatives 1 or 3, so these will be considered further.</p> <p>Subsequent discussion on the liaison reflector lead to a preference for</p> <p>A. for (type var : coll) or</p> <p>B. for (type var ; coll) // various TG5 members believe this is too error prone</p> <p>Meeting #4 (NJ): Bill will submit a proposal for the Jun meeting on the semantics of the for-each statement. Syntax remains as for each (type var in coll)</p> <p>Meeting #5 (Redmond): Bill reported that nothing need change in the TG5 spec in this regard. He's found library solutions for his STL .NET-related concerns.</p>	Yes	
17	5-Dec-03	meeting #1 (TX)	17	Technical		John Spicer	Check on the UK submission to WG21 re opening nested namespaces.	Meeting #2 (Hawaii): John doesn't see a problem with the basic mechanism. Let WG21 handle this.	Yes	
18	5-Dec-03	meeting #1 (TX)	18.3.6	Technical		Bjarne Stroustrup	How might parameter arrays fit into sequence constructors being considered in WG21?	We liaised. No action.	Yes	
19	5-Dec-03	meeting #1 (TX)		Technical	L	Brandon Bray	list of overlap between Standard C++ and features proposed by C++/CLI	Meeting #9 (NJ): Close without action.	Yes	
20	8-Dec-03	Herb Sutter	18.7.1	Technical		Herb Sutter	<p>Subject: RE: CLI binding: Delegating constructors and exceptions</p> <p>>>> "Herb Sutter" <hsutter@microsoft.com> 24 November 2003 18:33:42 >>></p> <p>> Actually, it's in there, thanks to BSI.</p> <p>> EDG suggested that we specify the answer in terms of object lifetime, so that other answers, > including the destructor calling question, can just fall out from rest of ISO C++ which specifies</p>	Herb responded. Resolved.	Yes	
21	24-Nov-03	Attila Feher		Editorial		Editor	When distilling PDF, add bookmarks. Look at other options too (such as hotlinks).		Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
22	24-Nov-03	Attila Feher	8.4	Technical			Base doc, pp. 17, line 43 (Automatic memory management). Object^ Pop() { if (first == nullptr) throw gcnew Exception("Can't Pop from an empty Stack."); Why do you gcnew the Exception? Is it necessary? There you throw a hat (handle), if I understand correctly. But why... Cannot even a value type just be thrown and make the catch box it, as it happens in C++?	Not an issue for TG5.	Yes	
23	16-Dec-03	Phone meeting	8.2.3	Editorial	R	Brandon Bray	Say more, especially w.r.t the template class <code>array<element-type></code> .		Yes	
24	16-Dec-03	Phone meeting	9	Technical	R	Brandon Bray	Review this clause.		Yes	
25	16-Dec-03	Phone meeting	10	Technical	H	Brandon Bray	Revise this clause by covering topics including application entry point, assembly boundaries, among others.		No	
26	16-Dec-03	Phone meeting	10.2.1	Technical		Brandon Bray	Clarify the ordering definition when multiple <u>accessibility keywords are used</u> .		Yes	
27	16-Dec-03	Phone meeting	12.13.6	Technical	H	Brandon Bray	Describe how <code>interior_ptr</code> , <code>pin_ptr</code> , <code>array</code> , and <code>safe_cast</code> are template-like with certain constraints.		Yes	
28	16-Dec-03	Phone meeting	12.3.6	Technical	M	Brandon Bray	Describe how the compiler will need to emit a modopt to distinguish <code>interior_ptr<T></code> from tracking reference to <code>T (T%)</code> in the metadata.		Yes	
29	16-Dec-03	Phone meeting	12.3.6.2	Technical	M	Brandon Bray	Spell out target type restrictions (for an <code>interior_ptr</code>)		Yes	
30	16-Dec-03	Phone meeting	12.3.6.3	Editorial		Brandon Bray	Describe the dangers of pointer arithmetic and <code>interior_ptr</code> s.	merged into issue #87.	Yes	
31	16-Dec-03	Phone meeting	12.3.7	Technical		Brandon Bray	Provide a grammar for <code>pinning_ptr</code>	merged into issue #27.	Yes	
32	16-Dec-03	Phone meeting	13	Technical		Tom Plum	What, if anything, goes in this clause?		Yes	
33	16-Dec-03	Phone meeting	14.1.1	Editorial	R	Brandon Bray	Review this subclause.		Yes	
34	16-Dec-03	Phone meeting	14.4	Editorial	R	Brandon Bray	Review this subclause.		Yes	
35	16-Dec-03	Phone meeting	15.1	Technical	H	Brandon Bray	The rewrite rules for <code>e[x]</code> (default indexed accesses) are different where there is only one index. This is because there is a potential ambiguity with the <code>C++</code> operator <code>[]</code> . Is this mentioned elsewhere?		Yes	
36	16-Dec-03	Phone meeting	15.3.8	Technical	M	Brandon Bray	<code>cv-qualification</code> needs to be considered for <code>dynamic_cast</code> .		No	
37	16-Dec-03	Phone meeting	15.3.9	Technical		Brandon Bray	Are <code>typeid<long></code> and <code>typeid<char></code> allowed (and if so, what do they mean).	They are allowed and are distinct.	Yes	
38	16-Dec-03	Phone meeting	15.3.9	Technical	L	Brandon Bray	Provide a spec for standard <code>typeid</code> (that returns <code>std::type_info</code>) in addition to the new <code>typeid</code> (that returns <code>System::Type</code>).	Meeting #9 (NJ): Close and list in Future Directions.	Yes	
39	16-Dec-03	Phone meeting	15.3.13	Editorial	H	Brandon Bray	Update this subclause		Yes	
40	16-Dec-03	Phone meeting	15.4.1.1	Editorial	R	Brandon Bray	Review this subclause.		Yes	
41	16-Dec-03	Phone meeting	15.4.1.4	Technical		All	Should a unary <code>^</code> operator exist?	Meeting #4 (NJ): No	Yes	
42	16-Dec-03	Phone meeting	15.4.6	Technical		Brandon Bray	Define the grammar for <code>gcnew</code> array, and describe array creation expression.		Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
43	16-Dec-03	Phone meeting	15.11.1	Technical		Mark Hall	Add support for handle equality comparison, and handle == != nullptr, and vice versa.	<p>Meeting #3 (Mel): Had a short discussion. Mark will produce a paper for the May meeting.</p> <p>Meeting #4 (NJ): No progress. To be discussed via email, and at the Jun meeting</p> <p>Meeting #5 (WA): Discussed briefly. Asked Mark to write this up and distribute to the reflector.</p> <p>Phone call Jun 29: This issue was resolved; just needs drafting of final words.</p> <p>Meeting 7 (WA): In the case of if(handle), which conversions are attempted before comparison against nullptr is used?</p> <p>We agreed that if an explicit conversion to bool exists, if(handle) uses that.</p> <p>There is no implicit unboxing.</p> <p>Steve and Mark worked on this and presented it to the full committee on the 2nd day.</p> <p>Based on committee feedback, Mark will write this up for future consideration.</p>	No	
44	16-Dec-03	Phone meeting	15.18	Technical	H	Brandon Bray	Add words to discuss assignment for properties and events from the point of view of the rewrite rules.		Yes	
45	16-Dec-03	Phone meeting	15.2	Technical		Brandon Bray	Investigate whether string literals include compile-time expressions, such as concatenation of strings with non-strings.	Meeting #4 (NJ): No action to be taken at this time.	Yes	Yes
46	16-Dec-03	Phone meeting	16.3	Technical		Jonathan Caves		<p>Meeting #3 (Melbourne): It was suggested that this issue be brought to WG21. It's a security issue in standard C++; it's not a CLI-specific issue. Jonathan will produce a paper for the May meeting.</p> <p>Meeting #4 (NJ): TG5 expressed opposition to expression-level checked/unchecked. Not to bring it to WG21.</p>	Yes	Yes
47	16-Dec-03	Phone meeting	17	Technical	M	Brandon Bray	Provide text for this clause (Namespaces)		No	
48	16-Dec-03	Phone meeting	18.3.1	Technical		Editor	Explain the difference between using 'override' and '= function-name'; one creates an .override directive in CIL, the other does not.		Yes	
49	16-Dec-03	Phone meeting	18.3.4	Technical		Brandon Bray	Describe in more detail the semantics of new, including its use on static member functions (currently new only applies to overriding, not to hiding).		Yes	
50	16-Dec-03	Phone meeting	18.4	Technical	M	Brandon Bray	Extend declarator-id's by adding a new production that allows default.		No	
51	16-Dec-03	Phone meeting	18.4	Technical		Brandon Bray	The grammar for indexer-parameter-declaration does not allow handles or pointers, but full declarators are not needed. The grammar should allow a simpler sequence of ptr-operator.		Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
52	16-Dec-03	Phone meeting	18.4.2	Technical	H	Brandon Bray	This subclause only covers how the accessor functions must be defined. The expressions clause needs to cover the rewrite rules that call accessor functions.		Yes	
53	16-Dec-03	Phone meeting	18.4.2	Technical		Brandon Bray	Property syntax: Describe the qualified name of a property. Meeting #2 (Hawaii): Agreed to keep the current syntax		Yes	
54	16-Dec-03	Phone meeting	18.5.2	Editorial	R	Brandon Bray	Review this subclause.		Yes	
55	16-Dec-03	Phone meeting	18.6	Editorial	R	Brandon Bray	Review this subclause.		Yes	
56	16-Dec-03	Phone meeting	18.7.4	Technical	M	Brandon Bray	Identify when (operator) synthesis would and would not occur.		Yes	
57	16-Dec-03	Phone meeting	18.6.5.1	Technical	L	Brandon Bray	Writeup of true and op false operators	DUPE OF #145	Yes	
58	16-Dec-03	Phone meeting	18.6.6.1	Technical		Mark Hall	Reword this subclause similarly to the way special member functions are described.	Meeting 7 (WA): ?? To be done in Tue morning work sessions.	No	
59	16-Dec-03	Phone meeting	18.6.6.1	Technical	H	Brandon Bray	Add another subclause to cover the compiler-generated conversion from handle to unspecified pool type.	Meeting 7 (WA): ?? To be done in Tue morning work sessions.	Yes	
60	16-Dec-03	Phone meeting	18.9	Technical		Brandon Bray	Add grammar for literal-constant-initializer = Standard C++ constant-initializer + float/double + String + nullptr.		Yes	
61	16-Dec-03	Phone meeting	18.9, 18.10	Technical		Brandon Bray	Justify why we need literal and initempty fields.	They are used in the BCL.	Yes	
62	16-Dec-03	Phone meeting	18.10.1	Technical	L	Brandon Bray	Add a description that for any value class we have to make the copy before calling member functions.	Meeting #9 (NJ): Needs to be done.	No	
63	16-Dec-03	Phone meeting	18.11	Technical	H	Brandon Bray	Say more about finalizers (including Dispose/~T and Finalize/!T) and add some examples.	Paper included in WD1.10.	Yes	
64	16-Dec-03	Phone meeting	19	Technical		Brandon Bray	Supply more text for this clause.		Yes	
65	16-Dec-03	Phone meeting	18.1	Technical		Editor	As a cross-language issue, come up with terminology to distinguish between destructors and finalizers. Perhaps "deterministic destructor" vs. "non-deterministic finalizer." Add some text in spec re this, esp. w.r.t C#'s use of destructor.	Feb 2005. Issue was dropped as the revised version of Brandon's "Destructors and Finalizers" paper makes this intent clear, and TG2 has now dropped the use of "destructor" in favor of "finalizer".	Yes	
66	16-Dec-03	Phone meeting	21	Editorial	M	Brandon Bray	Introduce value classes -- Discuss the following: value classes are optimized for small data structures. As such, value classes do not allow inheritance from anything but interface classes. Tie in fundamental classes.		No	
67	16-Dec-03	Phone meeting	21.4.1	Technical	H	Brandon Bray	Add words about instance constructors and static constructor. Value classes cannot have SMFs (specifically, default constructor, copy constructor, assignment operator, destructor, or finalizer. Need to add specification for this along with rationale.		No	
68	16-Dec-03	Phone meeting	22	Technical	L	Brandon Bray	Consider writing some text for this "place-holder" clause. Should this all go in the new annex "Future directions"?	Meeting #9 (NJ): Existing words adequate.	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
69	16-Dec-03	Phone meeting	23	Technical		Editor	The spec currently states "Throughout this Standard, the term "array" is used to mean an array in C++/CLI. A C++-style array is referred to as a native array whenever the distinction is needed." Tom was concerned that this was, perhaps, too subtle. He will try to come up with an alternative name for C++/CLI arrays. Meeting #2 (Hawaii): Use "Array" when we mean CLI array, and "array" means C-style array.		Yes	
70	16-Dec-03	Phone meeting	23	Technical		Sean Perry	Check if the term "array" is used in the library extensions plan of WG21.	Yes it is.	Yes	
71	16-Dec-03	Phone meeting	23	Editorial	R	Brandon Bray	Will review this whole clause.		Yes	
72	16-Dec-03	Phone meeting		Technical		Sean Perry	Look into possible performance issues re "for each" and delegates.	No information.	Yes	
73	16-Dec-03	Phone meeting	23.4	Technical		P.J. Plauger	Every array type inherits the members declared by the type System::Array. Currently, arrays do not have iterators compatible with Standard C++'s template library. Should they?	Meeting #5 (Redmond): Bill reported that nothing need change in the TG5 spec in this regard.	Yes	
74	16-Dec-03	Phone meeting	23.5	Technical	M	Brandon Bray	Write-up array covariance w.r.t arrays.		No	
75	16-Dec-03	Phone meeting	23.6	Technical	M	Brandon Bray	Write up array initialization.		No	
76	16-Dec-03	Phone meeting	24.4	Technical	H	Brandon Bray	Address what happens when a ref class does not implement an interface function (and what happens when a base class has a non-virtual function with the same name).	Resolved in H1; incorporated into 1.11	Yes	
77	16-Dec-03	Phone meeting	25	Technical		Herb Sutter	Coordinate with WG21's extended enum proposal.	see #102	Yes	
78	16-Dec-03	Phone meeting	26.1	Technical		Brandon Bray	Redo the grammar for delegate-definition, and find a place for it in the type tree. Replace all uses of "return-type" with appropriate production.		Yes	
79	16-Dec-03	Phone meeting	27	Technical	H	Brandon Bray	Cover unification of CLI and Standard C++ exception handling models, and anything else that might go in this clause. Are exceptions asynchronous now in some cases? Yes they are. (For example, NullReferenceException.)	Meeting #5 (WA): Kevin Free (Microsoft) gave a verbal presentation. catch(...) catches managed and native exceptions. catch(System::Object^) also catches both kinds, but won't invoke the destructor (so can leak). CLI exception handling supports more features than we expose. The issue remained with Brandon to write up, as before.	No	
80	16-Dec-03	Phone meeting	20.5.1	Technical		Brandon Bray	Check the name System::Reflection::DefaultMemberAttribute; it might have been renamed in the CLI standard.		Yes	
81	16-Dec-03	Phone meeting	20.5.2	Technical	R	Brandon Bray	Describe MethodImplOptions metadata generation.	The editor has added quite a bit of text re this attribute. See if that is sufficient.	Yes	
82	16-Dec-03	Phone meeting	29	Technical	M	Brandon Bray	Flesh out "Templates" clause.		No	
83	16-Dec-03	Phone meeting	30	Technical		Editor	Flesh out "Generics" clause.		Yes	
84	16-Dec-03	Phone meeting	31	Technical		P.J. Plauger	Suggest possible standard library interaction issues apart from I/O synchronization.	Meeting #8 (WA): Decided to say nothing about this.	Yes	
85	16-Dec-03	Phone meeting	32	Technical		Brandon Bray	Flesh out "CLI libraries" clause.		Yes	
86	16-Dec-03	dummy entry							yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
87	16-Dec-03	Phone meeting	A	Technical	L	Brandon Bray	Flesh out "Verifiable code" clause.	Meeting #9 (NJ): Close without action.	Yes	
88	16-Dec-03	Phone meeting	B	Technical	L	Editor	Flesh out "Documentation comments" clause.		Yes	
89	16-Dec-03	Phone meeting	C	Technical		Editor	Add any non-normative references		Yes	
90	16-Dec-03	Phone meeting	D	Technical		Editor	Add naming guidelines for generics		Yes	
91	29-Jan-04	meeting #2 (HI)	9.1.2	Technical		Editor	<p>Steve asked:</p> <p>Keywords:</p> <p>Are they keywords or identifiers?</p> <p>If keywords, are they always present or only in some modes?</p> <p>Are they recognized at the lexical level or at the syntactic level?</p> <p>If at the syntactic level, what are the rules? (disambiguation?)</p> <p>Should keywords like ref class have a space in the keyword or are they two words?</p>	<p>Meeting #2 (Hawaii): Herb will write a paper on keywords to cover the following:</p> <p>1) If it can be an identifier, it is.</p> <p>2) Use Mark's preprocessor option 1 (to not make the spaced words pp tokens, but rather, to assemble them early in translation phase 4).</p> <p>3) Add the fallback for namespace keywords.</p> <p>Address why "generic" shouldn't be spelled in some other way, perhaps as a spaced keyword, so that it need not be a regular keyword.</p> <p>Meeting #3 (Melbourne): Done, accepted, Editor to integrate. Steve will add more words (see issue #121).</p>	Yes	
92	29-Jan-04	meeting #2 (HI)		Technical	M	Brandon Bray	<p>"size size" name lookup issue (see email thread started by Herb Sutter on January 14 on the liaison reflector under the topic {Name lookup 1 (of 2): "Size Size" (CLI property naming idiom)}.)</p> <p>This is the common CLI idiom of naming a property (or potentially other members) with the same name as its type. In particular, here are two common examples:</p> <pre>value class Size { /*...*/ }; value class Color { /*...*/ }; ref class X { public: property Size Size; property Color Color; };</pre> <p>In other languages, it's easy to simply use the identifier "Size" without qualification and have the compiler Do the Right Thing™. But C++ name lookup is different. The status quo in Managed C++ syntax was that we made no change to C++ lookup rules, with the result that authors of classes that use this idiom are required to qualify most occurrences of "Size" which is ugly. The issue mostly appears only within the class itself (and in derived classes).</p> <p>Here's a brief description of the problem:</p> <pre>ref class X { public: property Size Size {</pre>	Meeting #8 (WA): Decided to not include this in V1.	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
93	29-Jan-04	meeting #2 (HI)	12.1	Technical		Tom Plum	<p>Do we require exact mapping for types, or do we allow a certain amount of flexibility?</p> <p>Should the size and representation of types long, long long, and long double (as well as wchar_t, see issue #5) be implementation-defined. Should all (or almost all) of the fundamental types being implementation-defined.</p> <p>The CLI types System::Single and System::Double require IEEE (IEC 559) representation. On many systems these naturally map to float and double, respectively. However, the IBM 390 does not use IEEE format for either of these types. A C++/CLI program running in that environment would want float/double to map to 390 types, so there would need to be a conversion to/from the CLI floating types.</p> <p>In order to encourage the writing of portable code, we'd need the largest core of fundamental type mapping as possible; for example, signed and unsigned 8-, 16-, and 32-bit integer mapping.</p>	<p>Meeting #3 (Mel): There was a lengthy discussion. No resolution.</p> <p>Meeting #4 (NJ): There was a lengthy discussion.</p> <p>Meeting #5 (WA): There was another lengthy discussion, which resulted in Plum's notes being incorporated into the meeting minutes.</p> <p>The edits from Plum's subsequent paper were incorporated into WD1.6 for Meeting #6 (WA).</p>	Yes	
94	29-Jan-04	meeting #2 (HI)		Technical		Mark Hall	<p>Relationship between primitive types and CLI types.</p> <p>The current spec allows the following: <code>int i = 10; String^ s = i.ToString();</code></p> <p>Standard C++ doesn't allow member selection on expressions of primitive type. Assuming <code>int</code> maps to <code>System::Int32</code>, just how much alike are these two types? Specifically, when do we treat the primitive as the underlying class.</p>	<p>Meeting 5 (WA): Asked Mark to write this up and distribute to the reflector. Please address the side-effect issue; that is, given <code>(i++).ToString()</code>, is the increment done?</p> <p>Meeting 7 (WA): ?? To be done in Tue morning work sessions.</p> <p>Re the side-effect, yes, it must be done.</p>	No	
95	29-Jan-04	meeting #2 (HI)	10	Technical	H	Brandon Bray	Provide words for #using.	The editor has added quite a bit of text re this topic.	Yes	
96	29-Jan-04	meeting #2 (HI)	9.1.1	Technical	M	Editor	The spec does not provide a way to use a keyword as an identifier. (VC++ uses the intrinsic <code>__identifier(name)</code> to achieve this; C# uses a leading <code>@.</code>) This is an issue for inter-operability; for example, being a consumer of a public type (written in something other than C++) that has a name (or contains a public member that has a name) that is a keyword in C++.	Meeting #8 (WA): It was proposed we support the intrinsic approach, accepting <code>__identifier(x)</code> , where <code>x</code> is a string literal or an identifier. String version is reserved for implementers.	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
97	29-Jan-04	meeting #2 (HI)		Technical		Editor	Overloading on arity. (This is a liaison issue with TG3.) The issue involves the overloading of a non-generic type with a one or more generic types of the same name in the same namespace. For example, the following is permitted by the CLS: ref class X { /*...*/ }; generic<typename T> /*...*/ ref class X { /*...*/ }; generic<typename T, typename U> /*...*/ ref class X { /*...*/ };	Meeting 3 (Mel): Herb presented this issue, which was then reassigned to Brandon. Meeting 5 (WA): In this version, we'll support a generic and non-generic version of a type in the same namespace, but not in different namespaces. There was a discussion about using something like "using generic x::y" to provide cross-namespace support as well. Rex to work with Brandon to get this into the draft. Meeting 7 (WA): Herb reported that the MS implementation can consume same-named generics that overload on arity in the same assembly, but it cannot create them.	Yes	
98	29-Jan-04	meeting #2 (HI)	30	Technical	R	Brandon Bray	Restrictions on generics re generic code generation. The current generics clause needs to be fleshed out, especially w.r.t how overload resolution works within the CLI.	Meeting #2 (HI): Brandon will write a paper on this. Meeting #4 (NJ): The fleshing out of Clause 30 is a significant contribution toward this. More work needed in declarations and function calls.	No	
99	29-Jan-04	meeting #2 (HI)		Technical		Daveed Vandevoorde	Write a paper proposing properties as specified by C++/CLI, for the March 2004 meeting of WG21.		Yes	
100	29-Jan-04	meeting #2 (HI)		Technical		Herb Sutter	nullptr: Write a paper proposing this to WG21.	Meeting #4 (NJ): WG21 expressed interest.	Yes	
101	29-Jan-04	meeting #2 (HI)		Technical		Herb Sutter	delegating constructors: Write a paper proposing this to WG21.	Meeting #4 (NJ): No implementation of this is expected anytime soon. TG5 agreed to not include this in this round. Editor will move 8.8.7.1 and 18.7.1 to Annex E, and remove any usage of delegating constructors from examples in other clauses.	Yes	Yes
102	29-Jan-04	meeting #2 (HI)		Technical		Herb Sutter	enhanced enums: Write a paper proposing this to WG21.	Meeting #4 (NJ): WG21 doesn't like enum class. WG21 doesn't know yet what it wants to do in this regard. However, if WG21 adopts a feature like this, but with different syntax, TG5 will revisit this when appropriate.	Yes	
103	29-Jan-04	meeting #2 (HI)		Technical		Brandon Bray	Explicit overriding: Propose to WG21	Meeting #4 (NJ): withdrawn	Yes	
104	29-Jan-04	meeting #2 (HI)		Technical		Steve Adamczyk	sealed, on classes and methods: Propose to WG21	Meeting #4 (NJ): withdrawn	Yes	
105	29-Jan-04	meeting #2 (HI)	14.5.1	Technical		Mark Hall	Constructors can't be used in casts in managed classes. Should they be allowed in explicit conversions? All managed type constructors being explicit by default. (Already yes, but reconfirm this.)	Meeting #4 (NJ): Steve will send the editor sufficient text to go into the public drop to indicate our intention re this topic. DONE. Meeting 5 (WA): Asked Mark to write this up and distribute to the reflector. Meeting 7 (WA): Steve and Mark worked on this and presented it to the full committee on the 2nd day. Mark will write this up for future consideration.	No	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
106	29-Jan-04	meeting #2 (HI)		Technical		Editor	Should >> handled as two tokens rather than one; e.g., List<List<int>>>.	<p>Meeting #3 (Mel): Had a short discussion. Tom will produce a paper for the May meeting.</p> <p>Meeting #4 (NJ): TG5 agreed that if a < for a template is seen, and >> that are not inside parentheses, that >> will always be considered to be the closing delimiter of two < symbols, and results in an error if there are not two such corresponding < symbols.</p> <p>Refer to Daveed's paper WG21/N1649 for more information.</p> <p>Meeting #7 (WA): This paper was updated (see N1699). It was discussed in TG5 and will be discussed at the up-coming WG21 meeting, at which TG5 members will participate.</p> <p>Meeting #8 (WA): Daveed presented this at the WG21 meeting this week. He proposed option 1, to which WG21 agreed. He was charged to write the final words.</p> <p>Meeting #9 (NJ): Daveed submitted a revised paper, which was accepted.</p> <p>Implemented in WD1.10.</p>	Yes	
107	29-Jan-04	meeting #2 (HI)		Technical		Editor	Look at the usage of the term "object" within the spec, and compare with the C++ std.		Yes	
108	19-Feb-04		12.3.6	Technical		Brandon Bray	Provide syntax for interior_ptrs		Yes	
109	19-Feb-04		12.3.6.3	Technical	L	Brandon Bray	Cover the dangers of pointer arithmetic and interior_ptrs	Meeting #9 (NJ): Close without action.	Yes	
110	19-Feb-04		12.3.7.1	Technical		Brandon Bray	Provide syntax for pinning_ptrs		Yes	
111	19-Feb-04		15.3.2	Technical	M	Brandon Bray	Need to consider how indexed access expressions are interpreted in templates.		No	
112	19-Feb-04		15.3.9	Technical		Brandon Bray	Check if long::typeid, char::typeid, etc. are allowed (and if so, what do they mean).	Meeting #4 (NJ): Allowed, but no modopts	Yes	
113	19-Feb-04		28.5.1.2	Technical		Brandon Bray	Provide text for MethodImplOptions attribute	duplicate	Yes	
114	19-Feb-04		15.4.6.2	Technical		Brandon Bray	Does new-initializer need to be changed?		Yes	
115	19-Feb-04		15.2	Technical		Brandon Bray	Do string literals include compile-time expressions, such as string concatenation?	duplicate	Yes	
116	19-Feb-04		18.4.2	Technical	H	Brandon Bray	Add some discussion of how accesses to properties are rewritten into accessor functions. This should be covered in rewrite rules in the expressions clause. Note that access checking for whether a property can be written to or read from is done after rewriting and overload resolutions.		Yes	
117	19-Feb-04		18.4.2	Technical	H	Brandon Bray	The qualified name of a property needs to be described somewhere. Once that happens, how an out-of-class definition is done will already be covered by existing rules.		No	
118	19-Feb-04		23.1.1	Technical		Editor	Is reference conversion the correct term?	No; it's a handle conversion	Yes	
119	19-Feb-04		28.5.1.1	Technical		Editor	Check this name (DefaultMember); this attribute might have been renamed in the CLI standard.	It has not been renamed, and appears in Beta 1 with that name.	Yes	
120	19-Mar-04	meeting #3 (Mel)		Technical		Tom Plum	Does typename allow us to pursue a containment policy re elaborated specifiers?	Meeting 7 (WA): Decided to drop this issue.	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
121	19-Mar-04	meeting #3 (Mel)		Technical		Steve Adamczyk	In the context of Herb's keywords paper (2004-05), Steve will write up the notion "If it can be an identifier, it is."		Yes	
122	19-Mar-04	meeting #3 (Mel)		Technical		Steve Adamczyk	Write a WG21 paper on extended integer types, promotion rules, costs of conversion, and the like, for the May meeting.	Meeting #4 (NJ): Not yet done, but still planned.	Yes	
123	3-May-04	meeting #4 (NJ)		Technical		Tom Plum	The draft uses the term "constructed type". It was suggested that the corresponding Standard C++ term is "instantiation". Which should we use?	Meeting 7 (WA): Chose to use "constructed type". No change needed to the spec.	Yes	
124	10-Jun-04	Jonathan Caves		Technical		Jonathan Caves	<p>Indexed properties -- Consider the following:</p> <pre> interface class I1 { property int Value; }; interface class I2 { property int Value[String^] { int get(String^); void set(String^, int); }; }; ref class D : I1, I2 { // Implements the properties }; D^ d; d->Value["Foo"]; The question is what does the last line do? Which leads to a language design question - what should the compiler do when faced with a property followed by a '[' 1) Should it look for just parameterized properties and if there isn't one fail - I suspect not 2) Should it look for all properties and if the returned set contains a parameterized property it should prefer it - this sounds like magic to me. 3) Should it look for all properties perform overload resolution across the whole set and if the resulting call is ambiguous then issue an error.</pre>	<p>Meeting #5 (WA): Discussed this. Option #3 preferred.</p> <p>Meeting 7 (WA): Discussed this in detail.</p> <pre> property int Value[int] { void set(int, int); }; x->Value[1] = 4 is treated as x->set_Value(1,4); ----- property array<int>^ Value { array<int>^ get(); } x->Value[1] = 4 is treated as x->get_Value()[1] = 4 ----- property int% Value[int] { int% get(int); } x->Value[1] = 4 is treated as x->get_Value(1) = 4 This construct violates the principle of properties (that of setting/getting the value of some property), so is not to be encouraged; however, it is supported, but no need to consider it further here.</pre>	No	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
125	14-Jun-04	meeting #5 (WA)	8.15.3	Technical	M	Brandon Bray	Based on the rules for type deduction in templates, it seems surprising that you can match <code>array<ItemType> ^</code> with an argument of type <code>int</code> . Here is a standard C++ example intended to illustrate the issue: <pre>template <class ItemType> struct Stack {}; template <class ItemType> struct Array { Array(ItemType); }; template <class ItemType> void PushMultiple(Stack<ItemType>, Array<ItemType>); int main() { Stack<int> s; PushMultiple(s, 1); // deduction fails PushMultiple<int>(s, 1); }</pre> Are the rules for generic different in this area? [There seems to be information related to this in 30.3.2. See that subclause for further comments on this issue.]		No	
126	14-Jun-04	meeting #5 (WA)	12.1	Technical		Editor	The type <code>long long</code> will be defined by pointing to Add text to indicate the circumstances under which The compiler will need to emit a <code>modopt</code> to distinguish <code>interior_ptr<T></code> from tracking reference to <code>T</code> (Need to add text to indicate the circumstances under which the <code>modopt IsPinned</code> shall be emitted (i.e., Separate the list of conversions from the order of Add text to indicate the circumstances under which type modifiers shall be emitted, and point to each modifier's definition. Unboxing and boxing are described as preferred user-defined conversions; however, this is incorrect. In a static cast of a handle to a base type to a handle for a derived type, there is no checking. This can be unverifiable and might cause a gc hole. Add text to indicate the circumstances under which the <code>modreq IsUdtReturn</code> shall be emitted (i.e., ref class type retruned by value). Point to that <code>modreq</code> 's spec.	Meeting 7 (WA): Steve has produced a revised version, N1693. Editor to fold this in the spec. TG5 understands that WG21 has not yet accepted this paper, but is expected to at its Oct 2004 meeting.	Yes	
127	14-Jun-04	meeting #5 (WA)	12.3.3	Technical	L	Brandon Bray		Meeting #9 (NJ): MS-specific; Close without action.	Yes	
128	14-Jun-04	meeting #5 (WA)	12.3.6	Technical	L	Brandon Bray			Yes	
129	14-Jun-04	meeting #5 (WA)	12.3.7	Technical	L	Brandon Bray			Yes	
130	14-Jun-04	meeting #5 (WA)	14.1.1	Technical	L	Brandon Bray		Meeting #9 (NJ): Close without action.	Yes	
131	14-Jun-04	meeting #5 (WA)	15.3.3	Technical	M	Editor	Add text to indicate the circumstances under which type modifiers shall be emitted, and point to each modifier's definition. Unboxing and boxing are described as preferred user-defined conversions; however, this is incorrect. In a static cast of a handle to a base type to a handle for a derived type, there is no checking. This can be unverifiable and might cause a gc hole. Add text to indicate the circumstances under which the <code>modreq IsUdtReturn</code> shall be emitted (i.e., ref class type retruned by value). Point to that <code>modreq</code> 's spec.	Meeting #9 (NJ): Needs to be done. Done in WD1.10.	Yes	
132	14-Jun-04	meeting #5 (WA)	15.3.10	Technical	M	Brandon Bray			No	
133	14-Jun-04	meeting #5 (WA)	15.3.10	Technical	L	Brandon Bray		Meeting #9 (NJ): Close without action.	Yes	
134	14-Jun-04	meeting #5 (WA)	16.3.3	Technical	M	Editor		Meeting #9 (NJ): Needs to be done. Done in WD1.10.	Yes	
135	14-Jun-04	meeting #5 (WA)	18	Technical	R	Brandon Bray		This table and corresponding sections should include Special Member Functions (SMFs) like <code>destruct</code>	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
136	14-Jun-04	meeting #5 (WA)	18.2.1	Technical		Editor	Need to address the following: C++/CLI uses the System::Reflection::DefaultMemberAttribute attribute to specify that something other than the default name, "Item", should be used. Given that, the text describes what happens if no name is chosen; that is, Item is used by default. Once the name has been set with DefaultMember, it cannot be changed in a derived class. If two interfaces have different DefaultMember attributes, implementing both interfaces is ill-formed.	Meeting #9 (NJ): Editor to mention this in the default indexer clause. Incorporated in WD1.10.	Yes	
137	14-Jun-04	meeting #5 (WA)	18.3	Technical		Brandon Bray	Extend the grammar to accommodate attributes on functions.		Yes	
138	14-Jun-04	meeting #5 (WA)	18.4	Technical		Mark Hall	Need to write up the restrictions on trivial properties.		No	
139	14-Jun-04	meeting #5 (WA)	18.4	Technical		Editor	We probably should say something about the reserved names get_Item and set_Item, and their relationship with default indexed properties. Also, add a forward pointer to the corresponding attribute.	Meeting #9 (NJ): Needs to be done. Handled as part of the resolution of #136.	Yes	
140	14-Jun-04	meeting #5 (WA)	18.5	Technical		Brandon Bray	The production event-type has not yet been defined. The syntactic category of this element needs to be		Yes	
141	14-Jun-04	meeting #5 (WA)	18.5.2	Technical		Brandon Bray	It is a bit strange to define grammar productions for these functions. We probably should either make		Yes	
142	14-Jun-04	meeting #5 (WA)	18.5.3	Technical	L	Brandon Bray	An event with the new modifier introduces a new event that does not override an event from a base class. Make sure the complete specification is provided in the clause for the new modifier.	Meeting #9 (NJ): Already in draft.	Yes	
143	14-Jun-04	meeting #5 (WA)	19.7	Technical	L	Brandon Bray	The restriction below does not apply to non-static member operators – that need not have a parameter of the type of the class.	Meeting #9 (NJ): Needs to be done.	No	
144	14-Jun-04	meeting #5 (WA)	18.6.1	Technical	L	Brandon Bray	Provide an example for "Homogenizing the candidate overload set".		Yes	
145	14-Jun-04	meeting #5 (WA)	18.6.5.2	Technical		Editor	Provide C++ names for operator True and False	Meeting #8 (WA): Move to future directions and close out.	Yes	
146	14-Jun-04	meeting #5 (WA)	18.9	Technical		Brandon Bray	add literal to storage-class-specifier		Yes	
147	14-Jun-04	meeting #5 (WA)	18.1	Technical		Brandon Bray	add initonly to storage-class-specifier		Yes	
148	14-Jun-04	meeting #5 (WA)	20.2	Technical		Editor	Add text to indicate the circumstances under which type modifiers shall be emitted, and point to each modifier's definition.	Meeting #9 (NJ): Needs to be done. Done in WD1.10.	Yes	
149	14-Jun-04	meeting #5 (WA)	20.3	Technical	L	Editor	Add text to indicate the circumstances under which type modifiers shall be emitted, and point to each modifier's definition.	Meeting #9 (NJ): Needs to be done. Done in WD1.10.	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
150	14-Jun-04	meeting #5 (WA)	21.4.1	Technical		Brandon Bray	Add words about instance constructors and static constructor.		Yes	
151	14-Jun-04	meeting #5 (WA)	25.2	Technical	M	Brandon Bray	The note says "pickup the restrictions from page 333 (of Brandon's paperback copy of the C# spec)".		No	
152	14-Jun-04	meeting #5 (WA)	25.1.3	Technical		Brandon Bray	Complete the production enum-base. Also, since this production is used by both native and CLI enums, yet it's described in the native section, wording might need to be re-arranged to make it read better from both enums' perspectives.		Yes	
153	14-Jun-04	meeting #5 (WA)	30.1	Technical	M	Brandon Bray	The text indicates that a generic-declaration may appear in a class scope, but the syntax of member declaration has not been extended to permit a generic-declaration. [[#98]]		No	
154	14-Jun-04	meeting #5 (WA)	30.1	Technical	R	Brandon Bray	Doesn't the text "a generic name declared in namespace scope or in class scope shall be unique in that scope" make the first sentence of this paragraph redundant? Re the reference to 14.5.4: That is the section on partial specialization. Generics can't be partially specialized, can they? The spec. should probably answer that explicitly.		No	
155	14-Jun-04	meeting #5 (WA)	30.1	Technical	R	Brandon Bray	What is a non-generic type? Does it mean that the rules are the same as classes? As template classes? Something else?		No	
156	14-Jun-04	meeting #5 (WA)	30.1	Technical		Editor	Can generic types be nested in native classes?	Included in WD1.10.	Yes	
157	14-Jun-04	meeting #5 (WA)	30.1	Technical		Brandon Bray	Type Overloading – This involves overloading on arity, and is currently under investigation. Such a feature permits the following: ref class X {}; generic<typename T> ref class X {}; generic<typename T, typename U> ref class X {};	Duplicate of #97	Yes	
158	14-Jun-04	meeting #5 (WA)	30.1.1	Technical	R	Brandon Bray	The equivalent wording for template parameters in the working paper has been changed to "defines its identifier to be a typedef-name". The revised wording should probably be used here too (see core issue 283)		No	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
159	14-Jun-04	meeting #5 (WA)	30.1.2	Technical	R	Brandon Bray	30.1.2 says "Like templates in Standard C++, within the body of a generic type any usage of the unqualified unadorned name of that type is assumed to refer to the current instantiation." 30.1.3 then goes on to describe "The instance type". Those seem like to different ways of describing the same concept. Can they be unified in some way?		Yes	
160	14-Jun-04	meeting #5 (WA)	30.1.6	Technical	R	Brandon Bray	This subclause describes when a static constructor is invoked. In 18.8, it references the CLI Standard Partition II (10.5.3). Are the rules the same? (Yes) Should this subclause also just reference the CLI spec? There are two sets of behavior; we need to say which one we use.		No	
161	14-Jun-04	meeting #5 (WA)	30.1.7	Technical	M	Brandon Bray	What to say about explicit conversion functions (which can only occur in managed class types)?		No	
162	14-Jun-04	meeting #5 (WA)	30.2.2	Technical	R	Brandon Bray	This subclause lists the types that can and cannot be generic arguments. Fundamental types are not included in either set, neither are function types. The subclause does not say whether or not cv-qualified types are allowed.		No	
163	14-Jun-04	meeting #5 (WA)	30.2.4	Technical	R	Brandon Bray	"The non-inherited members of a constructed type are obtained by substituting, for each generic-parameter in the member declaration, the corresponding generic-argument of the constructed type. The substitution process is based on the semantic meaning of type declarations, and is not simply textual substitution." It would be helpful to explain this in more detail and/or give an example where this makes a difference.		Yes	
164	14-Jun-04	meeting #5 (WA)	30.3	Technical		Editor	Can a generic function be declared inside a native class? (Yes) Can generic functions (and member functions) be declared inside a native class?		Yes	
165	14-Jun-04	meeting #5 (WA)	30.3	Technical		Editor	Types not used as a parameter type to a generic function cannot be deduced. Are the nondeduced context rules the same as Standard C++ or not? The sentence before this is true, but not complete if the rules are the same as Standard C++.	Meeting #8 (WA): The intent for V1 is to use the same rules as for templates. Meeting #9 (NJ): Say the following: "Types that cannot be deduced for function templates cannot be deduced for generic functions."	Yes	
166	14-Jun-04	meeting #5 (WA)	30.3	Technical		Editor	What, if anything, does it mean for a generic function to be deduced?	Meeting #6 (WA): all have the usual meaning.	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
167	14-Jun-04	meeting #5 (WA)	30.3	Technical	L	Brandon Bray	"When the type of a parameter or variable is a type parameter, the declaration of that parameter or variable shall use that type parameter's name without any pointer, reference, or handle declarators." What about cv-qualifiers?	Meeting #9 (NJ): Needs to be done. CV-qualifiers are not permitted.	No	
168	14-Jun-04	meeting #5 (WA)	30.3	Technical	L	Brandon Bray	Can you take the address of a generic function ins	Meeting #6 (WA): Tentatively decided, NO. Meeting #8 (WA): Reconsidered, and now think YES. Consider the following example: delegate void D(int); generic <class T> void F(T t); D^ d = gcnew D(&F<int>);	Yes	
169	14-Jun-04	meeting #5 (WA)	30.3.2	Technical	L	Brandon Bray	The issue raised in 8.15.3 is somewhat answered here. 18.3.6 seems to deal with expanded forms of calls, not expanded forms of function declarations. I interpret the text above as saying that deduction is done as if the function were declared like this: generic <typename ItemType> void PushMultiple(Stack<ItemType>^, ItemType i1, ItemType i2,/* ... */); Is that correct? I think this requires a more detailed description.	Meeting #9 (NJ): Needs to be done. Add example(s).	No	
170	14-Jun-04	meeting #5 (WA)	30.3.2	Technical	L	Brandon Bray	Something needs to be said about instantiating a generic delegate using a generic function.	Meeting #9 (NJ): Needs to be done.	No	
171	14-Jun-04	meeting #5 (WA)	30.4.2	Technical	H	Brandon Bray	When are members considered hidden? Is it using the rules described later? Those are described as applying only when a type parameter has both a class constraint and one or more interface constraints though.	Meeting #9 (NJ): Needs to be done. Resolved in H1; incorporated into 1.11	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
172	14-Jun-04	meeting #5 (WA)	30.4.4	Technical	H	Brandon Bray	Miscellaneous generics issues: 1. I seem to recall discussions of other kinds of constraints (I believe one of them concerned whether you could do a "new T()"). 2. Doesn't there need to be some discussion of how overload resolution works when a function argument has a type parameter as its type? 3. Are the typename and template rules for syntactic disambiguation the same in generics as in templates? Presumably, the lack of specialization would eliminate the need for these. 4. If scope contains a set of overloaded generic functions, is partial ordering used to choose between them? 5. I assume since there is nothing that says otherwise, that generics can be friends of other classes and generics can make other classes, functions, (including generics) friends? 6. If friendship is supported, can a generic first be declared in a friend declaration (suggested answer: no). 7. Standard C++ has restrictions on type parameters such as prohibiting types with no linkage. Does this rule apply to generic arguments? 8. Are there generic conversion functions?	Meeting #8 (WA): 1. For V1, we can consume and enforce these special constraints, but we can't author them. However, we plan to do so in future, so add this to "Future directions". Resolved in H1; incorporated into 1.11	Yes	
173	14-Jun-04	meeting #5 (WA)	32.1.4	Technical	L	Brandon Bray	To ensure that signatures for the same Type produced by different implementations match, the ordering in such a set of modreqs and modopts is as follows: first modreqs in ascending order by name, then modopts in ascending order by name, with case being significant. [[We need some rule here; is this the one?]].	Meeting #9 (NJ): Add a description of our best guess at the correct solution, to Future Directions, then mark this Postponed. Point to this from the normative text somehow.	No	
174	14-Jun-04	meeting #5 (WA)	32.1.4	Technical	L	Brandon Bray	If IsBoxed is retained for the standard, we have an ordering issue to consider: Currently, the value type special modopt is emitted before the IsBoxed modreq. For example, class [mscorlib]System.ValueType modopt([mscorlib]System.Int32) modreq([a]n.IsBoxed). That puts a modopt before a modreq.	Meeting #9 (NJ): MS-specific; Close without action.	Yes	
175	14-Jun-04	meeting #5 (WA)	32.1.5.1	Technical	L	Brandon Bray	This modifier [IsBoxed] is a workaround for the MS implementation. Does it have any long-term value for the standard, even if only as an historical note?	Meeting #9 (NJ): MS-specific; Close without action.	Yes	
176	14-Jun-04	meeting #5 (WA)	E	Technical	R	Brandon Bray	Flesh out Future Directions		Yes	
177	14-Jun-04	meeting #5 (WA)	E.7	Technical		Brandon Bray	Add text to show the behavior in the CLI (including	Feature dropped. So no need to persue.	yes	
178	14-Jun-04	meeting #5 (WA)	F	Technical		Brandon Bray	Flesh out anything in incompatibilities with Standard C++	Duplicate so closed this one.	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
179	23-Jul-04	TG3 liaison		Technical		Mark Hall	Support for Hide-By-Signature on Methods in ref classes (This would also apply to setter/getter methods for properties.)	See email thread started by Rex J. on Jul 24. Meeting #6 (WA): Some possible ways to address this (and results of a straw poll) are: 1) Support hidebyname only and issue better error messages. [0 in favour] 2) Make all ref class methods be hidebysig; a. Only [0 in favour] b. Default, with an option to select hidebyname [6 in favour] 3) Add hidebysig keyword to allow explicit marking of methods. [0 in favour] with 3 people unsure. We could go two routes: A) Bring hidebysig in via "using" directive to hoist base class/interface names (this is an approximate solution only, as it doesn't allow hoist-by-signature, only hoist-by-name) [0 in favour] B) Do repeated lookup in all base classes (like C#) [8 in favour] Tom circulated the relevant pages from the CLI spec (Partition I, 7.10.4). We need to take into account the CLS rules when resolving this issue. Meeting #7 (WA): Had a brief discussion. No progress.	Yes	
180	14-Jun-04	meeting #5 (WA)	26	Technical		Editor	Committee agreed with Rex's proposal to require that delegates have the optional BeginInvoke and EndInvoke methods for async processing of delegates.	This was reported to TG3 at its Jun 04 meeting, but there were concerns about the Compact Profile's not being required to support these at runtime. Since this is still an open issue in TG3, this issue will remain open in TG5.	Yes	
181	27-Jun-04			Technical		Tom Plum	Here are Tom's assumptions: C++/CLI will not initially have a built-in type for decimal the way C# has. In C++/CLI, you have to use namespace System::Decimal. The C++/CLI draft doesn't specify anything about semantics of Decimal; the requirements are as given in CLI (TG3). So we benefit from all the work done in TG3 on allowing IEEE Decimal as an alternative to .NET Decimal. Re the methods of the type System::Decimal methods, are they adequate for the C++ programmer, or should the compiler know something	Phone call Jun 29: discussed Decimal; agreed C++/CLI can just use constructors.	yes	
182	26-Jul-04	phone meeting		Technical	H	Brandon Bray	Discussion of passing a string literal in the presence of overloads taking String^ and const char * (what about char *?)	Meeting #6 (WA): The compiler currently chooses the String^ over the const char*. Involves type deduction across templates and generics. Reassigned from Mark to Brandon. String literal portion of issue 12 was transferred to #182.	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
183	2-Aug-04	meeting #6 (WA)		Technical	M	Brandon Bray	Overload assignment operator for handles.	Post-meeting #7. MS design team discussed this and believes that we should drop this issue.	Yes	
184	2-Aug-04	meeting #6 (WA)		Technical		Herb Sutter	Describe problem with overloading on % vs. & Herb presented the following code: #include <iostream> using namespace std; void f(const int&) { cout << "f(const int&)" << endl; } void f(int&) { cout << "f(int&)" << endl; } void g(int%) { cout << "g(int%)" << endl; } void g(int&) { cout << "g(int&)" << endl; } int main() { const int ci = 0; int i = 0; int^ hi = gcnew int; f(ci); f(i); g(*hi); // g(i); // ambiguous: should g(int&) be preferred? } The following code was his attempt to write an agnostic swap: template<typename T> void swap(T% a, T% b) { #if defined NO_PIN_PTR // doesn't work T temp = a; a = b; b = temp; #elif defined PIN_PTR_BUG // doesn't compile T temp = *pin_ptr<T>(a); *pin_ptr<T>(&a) = *pin_ptr<T>(&b);	Meeting #8 (WA). Decided to drop it.	No	
185	2-Aug-04	meeting #6 (WA)		Technical		Herb Sutter	Collapsing reference to reference. (It's in the C++0x spec.)	Meeting #9 (NJ): Close without action.	Yes	
186	2-Aug-04	meeting #6 (WA)		Technical	M	Brandon Bray	Should we standardize traits?	Meeting 9 (NJ): Agreed to drop this.	Yes	
187	2-Aug-04	meeting #6 (WA)		Technical		Brandon Bray	user-defined assignment operator for handles	duplicate of #183	Yes	
188	2-Aug-04	meeting #6 (WA)		Technical	H	Brandon Bray	Look at using + to implement String concatenation.		Yes	
189	2-Aug-04	meeting #6 (WA)		Technical		Editor	Look at the changes to the grammar for C++0x and note where they affect the C++/CLI grammar.	Put note in clause 3 using Steve's note to me as an example.	Yes	
190	2-Aug-04	meeting #6 (WA)		Editorial		Editor	Add an annex identifying behavior that is implementation-defined, undefined, or unspecified.	Done in WD1.10.	Yes	
191	2-Aug-04	meeting #6 (WA)		Technical	R	Brandon Bray	Review the specification checking the usage of accessibility vs. visibility		Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
192	2-Aug-04	meeting #6 (WA)		Technical	L	Brandon Bray	Provide an annex containing the differences between the grammar of Standard C++ and C++/CLI	Meeting #9 (NJ): Close without action.	Yes	
193	2-Aug-04	meeting #6 (WA)		Technical		Sean Perry	Look at the issue of whether or not the mapping of bool should be implementation-defined.	Meeting 7 (WA): Sean wrote this up and presented it to the full committee on the 2nd day. Based on committee feedback, Sean will revise his paper for future consideration. This was integrated into WD1.9.	Yes	
194	2-Aug-04	Anthony Williams	15.3.2	Technical		Jonathan Caves	Re Anthony's post to the reflector re "default index	Meeting 7 (WA): Discussed the possibility of disallowing both the default indexed property and operator[].	No	
195	25-Aug-04	Rex Jaeschke	14.1.	Technical	L	Brandon Bray	Separate the list of conversions from the order of preference (such as how Standard C++ separates Standard Conversions from overload resolution).	duplicate of #130	Yes	
196	30-Sep-04	meeting #7 (WA)		Technical		Herb Sutter	In native types, % behaves like &.		No	
197	30-Sep-04	meeting #7 (WA)	19.1	Technical		Herb Sutter	Should generic member functions be allowed in native classes? This feature appeared in the draft as an "editorial" addition. Does MS really intend to implement this feature? Yes, MS did.		Yes	
198	30-Sep-04	meeting #7 (WA)	2	Technical		Herb Sutter	Propose wording to require that extensions over and above ISO C++ requirements, be diagnosed.	Meeting 9 (NJ): Re the new paragraph added to S2. "Conformance" in response to spreadsheet issue #198, the committee believed this text does not adequately address the issue. The editor was asked to remove it. Ownership was transferred from Tom to Herb.	No	
199	30-Sep-04	meeting #7 (WA)	16.2.1	Technical	R	Brandon Bray	Proof the text on Collection type and how a for each is executed.		Yes	
200		meeting #7 (WA)	19.1	Technical		Herb Sutter	Regarding "Member functions in a native class can be generic", support for this appears to have been added inadvertently. However, is there any user need for it?	Since the MS product was going to support this anyway, Steve A. agreed to have it in the std.	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
201	23-Oct-04	meeting #8 (WA)		Technical	H	Brandon Bray	<p>How to accomodate non-CLI calling conventions on other platforms.</p> <p>Meeting #8 (WA):</p> <p>delegate void D(int);</p> <pre>generic<class T> void F(T t) { System::Console::WriteLine(t->ToString()); }</pre> <pre>typedef void (* FP)(int); void G(FP fp) { D^ d = gcnew D(fp); d(1010); }</pre> <pre>int main() { D^ d = gcnew D(&F<int>); d(42); FP fp = &F<int>; fp(101); G(&F<int>);</pre> <p>In MS's implementation, need to use __clrcall to indicate the clr calling convention. This lead to a discussion of how to accomodate non-CLI calling conventions on other platforms. It was noted that the CLI draft spec, Partition II, 15.3, "Calling convention", states:</p> <p>"When dealing with methods implemented outside the CLI it is important to be able to specify the calling convention required. For this reason there</p>		No	Yes
202	23-Oct-04	meeting #8 (WA)		Technical	H	Brandon Bray	Name lookup in managed classes ignores interfaces.		Yes	
203	26-Oct-04	Rex Jaeschke	10.1.2	Technical	M	Brandon Bray	[Note: The compiler needs to add typedef members to the class so that template code can use the return type or the parameter types. [[Need more explanation.]] end note]		No	
204	26-Oct-04	Rex Jaeschke	12.2.2	Technical	M	Brandon Bray	Write intro text.		No	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
205	26-Oct-04	Rex Jaeschke	15.5	Technical	H	Brandon Bray	15.5 Explicit type conversion (cast notation) The rules in the C++ Standard (§5.4/5) have been extended for C++/CLI by including safe casts before static casts. <ul style="list-style-type: none"> • a const_cast • a safe_cast • a safe_cast followed by a const_cast • a static_cast • a static_cast followed by a const_cast • a reinterpret_cast • a reinterpret_cast followed by a const_cast [Note: Standard C++ programs remain unchanged by this, as safe casts are ill-formed when either the expression type or target type is a native class. end note] Provide background on the expected behavior and rationale. (Get this from the updated casting proposal.)		No	
206	26-Oct-04	Rex Jaeschke	21.4	Technical	M	Brandon Bray	Simple value classes: Flesh this out.		No	
207	26-Oct-04	Rex Jaeschke	24.2.5	Technical	H	Brandon Bray	Interface member access: Write up. Attribute specification: Write up net modules. Should safe_cast allow casting to void?		No	
208	26-Oct-04	Rex Jaeschke	27.2	Technical	L	Brandon Bray		Meeting #9 (NJ): Close without action. The standard will not mention net modules.	Yes	
209	24-Nov-04		15.3.13	Technical	L	Brandon Bray		Meeting #9 (NJ): This is allowed.	Yes	
210	4-Dec-04	Rex Jaeschke	29.5.1	Technical	M	Brandon Bray	There is confusion about DefaultMember attribute and IndexerNameAttribute. In the current implementation, it appears that the first one is exhibiting the behavior of the second one, and the second one is being emitted into metadata directly when it should be consumed by the compiler.		No	
211	4-Dec-04	Rex Jaeschke	17.1	Technical	L	Brandon Bray	The namespace cli is reserved. However, what if the compiler imports an assembly created by C#, for example, containing a user-defined namespace cli having a type T, or a user-defined type called cli defined at the global namespace level and having a type T. Both of these appear to C++/CLI as the same names, namely ::cli::T? (BTW, this works with the current implementation.)		Yes	
212	4-Dec-04	Rex Jaeschke		Technical	M	Brandon Bray	Since static constructors are emitted in metadata as protected members, TG5 required that they be defined as protected, rather than the previous treatment, which allowed the programmer to give them any accessibility, but that was ignored by the compiler. (The same situation occurs with a finalizer and a destructor for a ref class.) Now that an interface is allowed to have a static constructor, we have no way to explicitly declare that member to be protected; all members in an interface are implicitly public. What to do?	Meeting 9 (NJ): Leave as is; that is, require a diagnostic if the accessibility specified contradicts what is required. Make sure this applies to destructors and finalizers as well.	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
213	4-Dec-04	Rex Jaeschke		Technical	M	Brandon Bray	13.3.3.2/4 of the C++ Standard has rules for pointer conversions, that need to be adapted to handles. Review this subclause and determine the changes needed for the C++/CLI spec.		No	
214	4-Dec-04	Rex Jaeschke		Technical		Editor	Representation of false and nullptr. After changes made earlier this year by TC39/TG3, the definition of System::Boolean requires that an instance of that type be 8 bits, that false be all-bits-zero, and that true have any one or more bits set. However, some months ago, TG5 agreed to NOT require that C++/CLI's bool type map to System::Boolean. As such, the representation of true and false is now unspecified. Consider a value class that contains a bool member. Being a value class it can't have a default constructor; instead, instances are born with the guaranteed default value all-bits-zero. However, without having any guarantee about the representation of true and false, we are not guaranteed what, if anything, that default value means. I believe it would be most useful for C++/CLI to require that false be all-bits-zero, and that true have any one or more (unspecified) bits set. (Note that TG3 and TG2 have a similar issue with System::Decimal, which is a 128-bit value class. As it happens, while all-bits-zero represents value zero in both the MS and IEEE 754r decimal representations	Implemented in WD1.10.	Yes	
215	Feb-3-2005	Jeff Peil	13.1.1, 13.1.3	Technical	2	Brandon Bray	\$13.1.1 and 13.1.3 disagree, one describes gc-lvalue->lvalue as a conversion for native types, the other describes it as never having gc-lvalues for these (they are always l-values) They need to be made consistent	Resolved in H1; incorporated into 1.11	Yes	
216	Feb-3-2005	Jeff Peil	18.5	Technical		Editor	Shouldn't DllImport be allowed on static member functions in ref/value classes?	Done in WD1.10.	Yes	
217	Feb-3-2005	Sean Perry	29.1.1	Technical		Editor	How do attributes work with derived classes. If I declare class B and D, which derives from B, and apply attribute X to both of them. What happens for the various values of AllowMultiple & Inherited? (pg. 154, line 35) We need to talk about inheritance. What happens to the attributes of the base class when we provide attributes on the derived class?	Editor posted a response to the liaison reflector on Feb 15, and made several small improvements to WD1.10.	Yes	
218	Feb-10-2005	Rex Jaeschke	15.3	Technical	M	Brandon Bray	Are the productions postfix-expression . pseudo-finalizer-name postfix-expression -> pseudo-finalizer-name necessary, and, if so, should the "pseudo-" prefix be dropped?		No	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
219	3-Mar-05	Brandon Bray		Technical	M	Brandon Bray	Currently, the Visual C++ compiler allows a friend to first declare a generic type. Whether the language specification says this is allowed is up for discussion. Are there any issues we should consider before saying that it should be supported?		No	
220	7-Mar-05	Sean Perry		Technical	M	Brandon Bray	<p>Destroying members should happen after the base class stuff in the fault block. TG5 also brought up the destructor order (which Jeff brought up last week).</p> <p>From Herb: Constructor failures.</p> <p>We need to tweak the IL we generate for constructors to have smoother handling of constructor exceptions and deep virtual calls in constructors. Here's what we need to do:</p> <pre>.ctor { bool baseIsConstructed = false; try { construct all our own directly held members call our base class's constructor baseIsConstructed = true; run our own constructor body } fault { destroy all our own directly held members (if non-null) if(baseIsConstructed) call our base class's destructor (same as when chaining from Dispose(true)) } }</pre>		No	
221	28-Apr-05	Jonathan Caves	15.3.3	Technical	M	Brandon Bray	Add an example		No	
222	28-Apr-05	Jonathan Caves	15.4.5	Technical	M	Brandon Bray	I think that some text needs to be added specifiy that with a generic parameter <code>dynamic_cast</code> is used to convert <code>IDisposable</code> and if this conversion fails then no further action is taken. The test sort of says this but not explicitly enough.	Consider adding the new text in the generics clause.	No	
223	28-Apr-05	Jonathan Caves	15.21	Technical	M	Brandon Bray	Add an example which shows what happens if a property returns an array, or if it returns a reference and it is used in a 'set' context but it does not have <u>set method</u> .		No	
224	28-Apr-05	Jonathan Caves	19.7	Technical	M	Brandon Bray	Revise this to accommodate <code>-></code> as a static operator. Also revise 19.7.2.		No	

This is a replacement/place-holder for Documents TC39-TG5/2005/016, 019, 021, and 024. Documents 016, 019, and 021 were intermediate committee drafts of the specification, and are not included here. They are superseded by document 024, which can be found at the following URLs:

<http://www.plumhall.com/ecma/index.html>

<http://go.microsoft.com/fwlink/?LinkId=50042>

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
25	16-Dec-03	Phone meeting	10	Technical	H	Brandon Bray	Revise this clause by covering topics including application entry point, assembly boundaries, among others.		No	
43	16-Dec-03	Phone meeting	15.11.1	Technical		Mark Hall	Add support for handle equality comparison, and handle ==/!= nullptr, and vice versa.	<p>Meeting #3 (Mel): Had a short discussion. Mark will produce a paper for the May meeting.</p> <p>Meeting #4 (NJ): No progress. To be discussed via email, and at the Jun meeting</p> <p>Meeting #5 (WA): Discussed briefly. Asked Mark to write this up and distribute to the reflector.</p> <p>Phone call Jun 29: This issue was resolved; just needs drafting of final words.</p> <p>Meeting 7 (WA): In the case of if(handle), which conversions are attempted before comparison against nullptr is used?</p> <p>We agreed that if an explicit conversion to bool exists, if(handle) uses that.</p> <p>There is no implicit unboxing.</p> <p>Steve and Mark worked on this and presented it to the full committee on the 2nd day.</p> <p>Based on committee feedback, Mark will write this up for future consideration.</p>	No	
47	16-Dec-03	Phone meeting	17	Technical	M	Brandon Bray	Provide text for this clause (Namespaces)		No	
50	16-Dec-03	Phone meeting	18.4	Technical	H	Brandon Bray	Extend declarator-id's by adding a new production that allows default.		No	
58	16-Dec-03	Phone meeting	18.6.6.1	Technical		Mark Hall	Reword this subclause similarly to the way special member functions are described.	Meeting 7 (WA): ?? To be done in Tue morning work sessions.	No	
62	16-Dec-03	Phone meeting	18.10.1	Technical	L	Brandon Bray	Add a description that for any value class we have to make the copy before calling member functions.	Meeting #9 (NJ): Needs to be done.	No	
66	16-Dec-03	Phone meeting	21	Editorial	M	Brandon Bray	Introduce value classes -- Discuss the following: value classes are optimized for small data structures. As such, value classes do not allow inheritance from anything but interface classes. Tie in fundamental classes.		No	
67	16-Dec-03	Phone meeting	21.4.1	Technical	H	Brandon Bray	Add words about instance constructors and static constructor. Value classes cannot have SMFs (specifically, default constructor, copy constructor, assignment operator, destructor, or finalizer. Need to add specification for this along with rationale.		No	
74	16-Dec-03	Phone meeting	23.5	Technical	M	Brandon Bray	Write-up array covariance w.r.t arrays.		No	
75	16-Dec-03	Phone meeting	23.6	Technical	M	Brandon Bray	Write up array initialization.		No	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
79	16-Dec-03	Phone meeting	27	Technical	H	Brandon Bray	Cover unification of CLI and Standard C++ exception-handling models, and anything else that might go in this clause. Are exceptions asynchronous now in some cases? Yes they are. (For example, NullReferenceException.)	Meeting #5 (WA): Kevin Free (Microsoft) gave a verbal presentation. catch(...) catches managed and native exceptions. catch(System::Object^) also catches both kinds, but won't invoke the destructor (so can leak). CLI exception handling supports more features than we expose. The issue remained with Brandon to write up, as before.	No	
82	16-Dec-03	Phone meeting	29	Technical	M	Brandon Bray	Flesh out "Templates" clause.		No	
94	29-Jan-04	meeting #2 (HI)		Technical		Mark Hall	Relationship between primitive types and CLI types. The current spec allows the following: int i = 10; String^ s = i.ToString(); Standard C++ doesn't allow member selection on expressions of primitive type. Assuming int maps to System::Int32, just how much alike are these two types? Specifically, when do we treat the primitive as the underlying class.	Meeting 5 (WA): Asked Mark to write this up and distribute to the reflector. Please address the side-effect issue; that is, given (i++).ToString, is the increment done? Meeting 7 (WA): ?? To be done in Tue morning work sessions. Re the side-effect, yes, it must be done.	No	
98	29-Jan-04	meeting #2 (HI)	30	Technical	R	Brandon Bray	Restrictions on generics re generic code generation. The current generics clause needs to be fleshed out, especially w.r.t how overload resolution works within the CLI.	Meeting #2 (HI): Brandon will write a paper on this. Meeting #4 (NJ): The fleshing out of Clause 30 is a significant contribution toward this. More work needed in declarations and function calls.	No	
105	29-Jan-04	meeting #2 (HI)	14.5.1	Technical		Mark Hall	Constructors can't be used in casts in managed classes. Should they be allowed in explicit conversions? All managed type constructors being explicit by default. (Already yes, but reconfirm this.)	Meeting #4 (NJ): Steve will send the editor sufficient text to go into the public drop to indicate our intention re this topic. DONE. Meeting 5 (WA): Asked Mark to write this up and distribute to the reflector. Meeting 7 (WA): Steve and Mark worked on this and presented it to the full committee on the 2nd day. Mark will write this up for future consideration.	No	
111	19-Feb-04		15.3.2	Technical	M	Brandon Bray	Need to consider how indexed access expressions are interpreted in templates.		No	
117	19-Feb-04		18.4.2	Technical	H	Brandon Bray	The qualified name of a property needs to be described somewhere. Once that happens, how an out-of-class definition is done will already be covered by existing rules.		No	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
124	10-Jun-04	Jonathan Caves		Technical		Jonathan Caves	<p>Indexed properties -- Consider the following:</p> <pre> interface class I1 { property int Value; }; interface class I2 { property int Value[String^] { int get(String^); void set(String^, int); }; }; ref class D : I1, I2 { // Implements the properties }; D^ d; d->Value["Foo"]; The question is what does the last line do? Which leads to a language design question - what should the compiler do when faced with a property followed by a '[' 1) Should it look for just parameterized properties and if there isn't one fail - I suspect not 2) Should it look for all properties and if the returned set contains a parameterized property it should prefer it - this sounds like magic to me. 3) Should it look for all properties perform overload resolution across the whole set and if the resulting call is ambiguous then issue an error. Mark Hall says: Jonathan's looking into deferring the </pre>	<p>Meeting #5 (WA): Discussed this. Option #3 preferred.</p> <p>Meeting 7 (WA): Discussed this in detail.</p> <pre> property int Value[int] { void set(int, int); }; x->Value[1] = 4 is treated as x->set_Value(1,4); ----- property array<int>^ Value { array<int>^ get(); } x->Value[1] = 4 is treated as x->get_Value()[1] = 4 ----- property int% Value[int] { int% get(int); } x->Value[1] = 4 is treated as x->get_Value(1) = 4 This construct violates the principle of properties (that of setting/getting the value of some property), so is not to be encouraged; however, it is supported, but no need to consider it further here. ----- </pre>	No	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
125	14-Jun-04	meeting #5 (WA)	8.15.3	Technical	M	Brandon Bray	Based on the rules for type deduction in templates, it seems surprising that you can match <code>array<ItemType>^</code> with an argument of type <code>int</code> . Here is a standard C++ example intended to illustrate the issue: <pre>template <class ItemType> struct Stack {}; template <class ItemType> struct Array { Array(ItemType); }; template <class ItemType> void PushMultiple(Stack<ItemType>, Array<ItemType>); int main() { Stack<int> s; PushMultiple(s, 1); // deduction fails PushMultiple<int>(s, 1); }</pre> Are the rules for generic different in this area? [There seems to be information related to this in 30.3.2. See that subclause for further comments on this issue.]		No	
132	14-Jun-04	meeting #5 (WA)	15.3.10	Technical	M	Brandon Bray	Unboxing and boxing are described as preferred user-defined conversions; however, this is incorrect.		No	
138	14-Jun-04	meeting #5 (WA)	18.4	Technical		Mark Hall	Need to write up the restrictions on trivial properties.		No	
143	14-Jun-04	meeting #5 (WA)	19.7	Technical	L	Brandon Bray	The restriction below does not apply to non-static member operators – that need not have a parameter of the type of the class.	Meeting #9 (NJ): Needs to be done.	No	
151	14-Jun-04	meeting #5 (WA)	25.2	Technical	M	Brandon Bray	The note says "pickup the restrictions from page 333 (of Brandon's paperback copy of the C# spec)".		No	
153	14-Jun-04	meeting #5 (WA)	30.1	Technical	M	Brandon Bray	The text indicates that a generic-declaration may appear in a class scope, but the syntax of member declaration has not been extended to permit a generic-declaration. [[#98]]		No	
154	14-Jun-04	meeting #5 (WA)	30.1	Technical	R	Brandon Bray	Doesn't the text "a generic name declared in namespace scope or in class scope shall be unique in that scope" make the first sentence of this paragraph redundant? Re the reference to 14.5.4: That is the section on partial specialization. Generics can't be partially specialized, can they? The spec. should probably answer that explicitly.		No	
155	14-Jun-04	meeting #5 (WA)	30.1	Technical	R	Brandon Bray	What is a non-generic type? Does it mean that the rules are the same as classes? As template classes? Something else?		No	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
158	14-Jun-04	meeting #5 (WA)	30.1.1	Technical	R	Brandon Bray	The equivalent wording for template parameters in the working paper has been changed to "defines its identifier to be a typedef-name". The revised wording should probably be used here too (see core issue 283)		No	
160	14-Jun-04	meeting #5 (WA)	30.1.6	Technical	R	Brandon Bray	This subclause describes when a static constructor is invoked. In 18.8, it references the CLI Standard Partition II (10.5.3). Are the rules the same? (Yes) Should this subclause also just reference the CLI spec? There are two sets of behavior; we need to say which one we use.		No	
161	14-Jun-04	meeting #5 (WA)	30.1.7	Technical	M	Brandon Bray	What to say about explicit conversion functions (which can only occur in managed class types)?		No	
162	14-Jun-04	meeting #5 (WA)	30.2.2	Technical	R	Brandon Bray	This subclause lists the types that can and cannot be generic arguments. Fundamental types are not included in either set, neither are function types. The subclause does not say whether or not cv-qualified types are allowed.		No	
167	14-Jun-04	meeting #5 (WA)	30.3	Technical	L	Brandon Bray	"When the type of a parameter or variable is a type parameter, the declaration of that parameter or variable shall use that type parameter's name without any pointer, reference, or handle declarators." What about cv-qualifiers?	Meeting #9 (NJ): Needs to be done. CV-qualifiers are not permitted.	No	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
169	14-Jun-04	meeting #5 (WA)	30.3.2	Technical	L	Brandon Bray	The issue raised in 8.15.3 is somewhat answered here. 18.3.6 seems to deal with expanded forms of calls, not expanded forms of function declarations. I interpret the text above as saying that deduction is done as if the function were declared like this: generic <typename ItemType> void PushMultiple(Stack<ItemType>^, ItemType i1, ItemType i2,/* ... */); Is that correct? I think this requires a more detailed description.	Meeting #9 (NJ): Needs to be done. Add example(s).	No	
170	14-Jun-04	meeting #5 (WA)	30.3.2	Technical	L	Brandon Bray	Something needs to be said about instantiating a generic delegate using a generic function.	Meeting #9 (NJ): Needs to be done.	No	
173	14-Jun-04	meeting #5 (WA)	32.1.4	Technical	L	Brandon Bray	To ensure that signatures for the same Type produced by different implementations match, the ordering in such a set of modreqs and modopts is as follows: first modreqs in ascending order by name, then modopts in ascending order by name, with case being significant. [[We need some rule here; is this the one?]].	Meeting #9 (NJ): Add a description of our best guess at the correct solution, to Future Directions, then mark this Postponed. Point to this from the normative text somehow.	No	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
184	2-Aug-04	meeting #6 (WA)		Technical		Herb Sutter	<p>Describe problem with overloading on % vs. &</p> <p>Herb presented the following code:</p> <pre>#include <iostream> using namespace std; void f(const int&) { cout << "f(const int&)" << endl; } void f(int&) { cout << "f(int&)" << endl; } void g(int%) { cout << "g(int%)" << endl; } void g(int&) { cout << "g(int&)" << endl; } int main() { const int ci = 0; int i = 0; int^ hi = gcnew int; f(ci); f(i); g(*hi); // g(i); // ambiguous: should g(int&) be // preferred? }</pre> <p>The following code was his attempt to write an agnostic swap:</p> <pre>template<typename T> void swap(T% a, T% b) { #if defined NO_PIN_PTR // doesn't work T temp = a; a = b; b = temp; #elif defined PIN_PTR_BUG // doesn't compile T temp = *pin_ptr<T>(a); *pin_ptr<T>(*pa) = *pin_ptr<T>(*pb); *pin_ptr<T>(*pb) = temp;</pre>		No	
194	2-Aug-04	Anthony Williams	15.3.2	Technical		Jonathan Caves	Re Anthony's post to the reflector re "default inde	Meeting 7 (WA): Discussed the possibility of disallowing both the default indexed property and operator[.]	No	
196	30-Sep-04	meeting #7 (WA)		Technical		Herb Sutter	In native types, % behaves like &.		No	
198	30-Sep-04	meeting #7 (WA)	2	Technical		Herb Sutter	Propose wording to require that extensions over and above ISO C++ requirements, be diagnosed.	<p>Meeting 9 (NJ): Re the new paragraph added to §2. "Conformance" in response to spreadsheet issue #198, the committee believed this text does not adequately address the issue. The editor was asked to remove it.</p> <p>Ownership was transferred from Tom to Herb.</p>	No	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
203	26-Oct-04	Rex Jaeschke	10.1.2	Technical	M	Brandon Bray	[Note: The compiler needs to add typedef members to the class so that template code can use the return type or the parameter types. [[Need more explanation.]] end note]		No	
204	26-Oct-04	Rex Jaeschke	12.2.2	Technical	M	Brandon Bray	Write intro text.		No	
206	26-Oct-04	Rex Jaeschke	21.4	Technical	M	Brandon Bray	Simple value classes: Flesh this out.		No	
207	26-Oct-04	Rex Jaeschke	24.2.5	Technical	H	Brandon Bray	Interface member access: Write up.		No	
210	4-Dec-04	Rex Jaeschke	29.5.1	Technical	M	Brandon Bray	There is confusion about DefaultMember attribute and IndexerNameAttribute. In the current implementation, it appears that the first one is exhibiting the behavior of the second one, and the second one is being emitted into metadat directly when it should be consumed by the compiler.		No	
213	4-Dec-04	Rex Jaeschke		Technical	M	Brandon Bray	13.3.3.2/4 of the C++ Standard has rules for pointer conversions, that need to be adapted to handles. Review this subclause and determine the changes needed for the C++/CLI spec.		No	
218	Feb-10-2005	Rex Jaeschke	15.3	Technical	M	Brandon Bray	Are the productions postfix-expression . pseudo-finalizer-name postfix-expression -> pseudo-finalizer-name necessary, and, if so, should the "pseudo-" prefix be dropped?		No	
219	3-Mar-05	Brandon Bray		Technical	M	Brandon Bray	Currently, the Visual C++ compiler allows a friend to first declare a generic type. Whether the language specification says this is allowed is up for discussion. Are there any issues we should consider before saying that it should be supported?		No	
221	28-Apr-05	Jonathan Caves	15.3.3	Technical	M	Brandon Bray	Add an example		No	
222	28-Apr-05	Jonathan Caves	15.4.5	Technical	M	Brandon Bray	I think that some text needs to be added specifiy that with a generic parameter dynamic_cast is used to convert IDisposable and if this conversion fails then no further action in taken. The test sort of says this but not explicitly enough.	Consider adding the new text in the generics clause.	No	
223	28-Apr-05	Jonathan Caves	15.21	Technical	M	Brandon Bray	Add an example which shows what happens if a property returns an array, or if it returns a reference and it is used in a 'set' context but it does not have set method.		No	
224	28-Apr-05	Jonathan Caves	19.7	Technical	M	Brandon Bray	Revise this to accommodate -> as a static operator. Also revise 19.7.2.		No	

This is a replacement/place-holder for Documents TC39-TG5/2005/016, 019, 021, and 024. Documents 016, 019, and 021 were intermediate committee drafts of the specification, and are not included here. They are superseded by document 024, which can be found at the following URLs:

<http://www.plumhall.com/ecma/index.html>

<http://go.microsoft.com/fwlink/?LinkId=50042>

Agenda

for the:

11th meeting of Ecma TC39-TG5

to be held in:

Redmond, WA, USA

on:

September 2005

TIME: 09:00 till 17:00 on Mon 19th September 2005
09:00 till 17:00 on Tue 20th September 2005
09:00 till 10:00 on Fri 23rd September 2005
[8:30AM Breakfast, Noon lunch each day]

LOCATION: Microsoft Campus
Bldg 42, Room 1600
Redmond, WA 98052
USA
(Directions: see TG5/2004/021)

CONTACT: John Hawkins
johawk@microsoft.com

1 Opening

- 1.1 Appointment of Recording Secretary**
- 1.2 Introduction of participants**
- 1.3 Host facilities/local information**

2 Adoption of the agenda

3 Final approval of minutes of previous TG5 meeting (TG5/2005/015)

4 Matters arising from the minutes not covered elsewhere

5 Project Editor's Report

6 Approving tracked changes in latest draft

7 Date and place of next meetings

8 Reports from Liaisons

- 8.1 TC39 TG3 (CLI) – Rex Jaeschke**
- 8.2 SC22/WG21 (C++) – Tom Plum, P. J. Plauger, Tana Plauger, John Spicer, and Steve Adamczyk**
- 8.3 TC39 TG2 (C#) – Rex Jaeschke**

- 9 Action item spreadsheet review
- 10 Approval of TG5 spec to forward to TC39
- 11 Any other business, and appreciation of hosts
- 12 Adjournment

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
2	7-Oct-03	Rex Jaeschke		Technical		P.J. Plauger	The current CLI spec supports Unicode V3.0. What, if anything, should we do w.r.t V3.1/V4.0?	Brought up during the phone meeting of 10/7/2003. Meeting #4 (NJ): Take no action. Don't mention more that necessary.	Yes	
3	7-Oct-03	Tom Plum		Technical		Tom Plum	Diagnostics: How should we deal with warnings and such?	Meeting #3 (Melbourne): Tom will adapt text from the C# spec and present it. Meeting #4 (NJ): Withdrawn without action.	Yes	
4	10-Oct-03	Phone meeting		Editorial		Editor	Future directions: Should there be an informative annex listing future directions? Possible entries are: 1. Supporting static members in interfaces 2. Mixed types 3. gnew of unmanaged types 4. new of managed types		Yes	
5	10-Oct-03	Tom Plum		Technical		Tom Plum	While discussing enums (25.1.3) and wchar_t's not being permitted as an underlying type, a discussion arose w.r.t CLI's requiring wchar_t to have the same representation as System::Char; that is, a 16-bit character. This needs further investigation. Possible need to look at/point to the PDTR currently out from WG11 (ISO C). This is part of a more general issue. Do we require exact mapping for types, or do we allow a certain amount of flexibility? See issue #93.	In email on 2003-10-12 Tom Plum wrote: Refining my comments re wchar_t, I see a short-term and a long-term ... Short-term, there's no need to change anything. The 16-bit unicode type is wchar_t in VC++ and in C++/CLI. Long-term, the decision is up to TG5, and depends upon who participates. My own guess is that TG5 in fact will be the first group that has to integrate Unicode 3.1 and 4.0 into its language definition. I suspect that before we're done we'll have four types of character (and literal and C++ string): char - has to be 8 bits to integrate with CLI 'x' "str" string = basic_string<char> wchar_t - implementation's legacy choice of widechar L'x' L"str" wstring = basic_string<wchar_t> char16_t - 16-bit character type, has to be UCS-2 or UTF-16 for CLI u'x' u"str" ustring (?) = basic_string<char16_t> (or string16?) char32_t - 32-bit character type, has to be UTF-32 for CLI U'x' U"str" Ustring (?) = basic_string<char32_t> (or string32?) wchar_t can be the same type as char16_t or char32_t, but isn't required to be	Yes	
6	10-Oct-03	Phone meeting		Technical		Brandon Bray	Issue of mapping system value types to the fundamental types, and interop with the standard library.	Merged in with issue #93	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
7	21-Oct-03	Rex Jaeschke	7	Technical		P.J. Plauger	What is the interaction between the standard I/O streams and System::Console?	Meeting #3 (Melbourne): It appears that there will not be any synchronization between the two. Meeting #8 (WA): Decided to say nothing about this.	Yes	
8	4-Dec-03	meeting #1 (TX)	12.1.1	Technical		Steve Adamczyk	64-bit integer mapping. Meeting #1 (TX): Steve to write a paper for Jan 04 meeting. Done.	Meeting #2 (HI): This paper will be presented at the March meeting of WG21. Let's see how it is received? Meeting #4 (NJ): Steve will suggest how to tighten existing wording w.r.t a 64-bit integer type in the current draft, as part of the cleanup for the public drop. As to how to document the library support has yet to be determined.	Yes	
9	4-Dec-03	meeting #1 (TX)		Technical		Brandon Bray	Write a paper on "It just works"		Yes	
10	4-Dec-03	meeting #1 (TX)	14	Technical	R	Brandon Bray	pull together all the conversion information into one place. Make sure all conversions are covered.		Yes	
11	4-Dec-03	meeting #1 (TX)	15.3.2	Technical		Steve Adamczyk	comma vs. semicolon as separator in indexed access expressions In indexed access expressions (§15.3.2), comma operators are currently disallowed inside [] unless they are enclosed in parentheses. This conflicts with usage in existing template libraries (e.g., Lambda), in which the comma operator occurs inside [] without enclosing it in parentheses.	Meeting #2 (HI): Can we treat commas in [] not having enclosing parenthesis, in any context, always be treated as punctuators? Yes. Steve will provide words to the editor for this. Meeting #3 (Mel): Steve produced a paper. He reported one outstanding issue: In 15.3.2, "Indexed Access", in the C++/CLI spec is rather vague. There, we have indexed-access: indexed-designator [expression-list] where indexed-access is defined as an additional alternative for postfix-expression: postfix-expression: indexed-access Unfortunately, there isn't any definition of indexed-designator, so I'm not quite sure whether all the multi-dimensional cases are supposed be handled by indexed-designator, leaving the traditional cases to be handled by the original (possibly modified) syntax. An alternative would be not to introduce indexed-access at all, and use the definition postfix-expression: postfix-expression [expression-list] to handle all the cases, for both traditional subscripting and the new C++/CLI indexer references. There was agreement to this, so Steve will update his p	yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
12	4-Dec-03	meeting #1 (TX)	9	Technical		Tom Plum	Issue of source code/Unicode mapping. What assumptions, if any, should we make about the form of input text? Handling of string literals, character constants, and comments.	Meeting #3 (Melbourne): Had a short discussion. Tom will produce a paper for the May meeting. Meeting #4 (NJ): Tom got more input at this meeting, and will produce a paper for the Jun meeting. DONE (see email "TG5 issue #12 - character sets" from 5/29 EDT) Meeting #5 (Redmond): Discussed Tom's paper in detail. He'll update and recirculate. Meeting #6 (Redmond): Closed out this issue with the string literal portion of this issue being transferred to #182.	Yes	
13	4-Dec-03	meeting #1 (TX)	12	Technical	M	Brandon Bray	Add a diagram of the type tree		Yes	
14	5-Dec-03	meeting #1 (TX)	15.3.9	Technical		Editor	alternative syntax for typeid <type-id> The current syntax typeid <type-id> is too close to the Standard C++ forms.	Meeting #2 (Hawaii): Ownership of this issue transferred from John to Herb. Several alternatives were discussed, including a keyword CLI_typeid or CLI_typeof, and a static member .class ala Java. Also ::typeid. Herb addressed this in his keywords paper, which was adopted in Melbourne.	Yes	
15	5-Dec-03	meeting #1 (TX)	16.1.1	Technical		Tom Plum	Write a paper for Jan, 04, meeting on use of for-each with STL types. TG5 will not pursue this as it's part of the work being considered by WG21's evolution group.		Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
16	5-Dec-03	meeting #1 (TX)	16.1.1	Technical		P.J. Plauger	The for each statement. Meeting #1 (Texas): Write a paper for Jan, 04, meeting on spelling "for each" simply as "for".	Meeting #2 (Hawaii): Tom presented his proposal from his email entitled {"for" in the style of "for each"} from January 28. A discussion ensued, during which the following alternatives (the colon versions of which were new) were discussed in detail: 1. for each (type var in coll) 2. for (type var in coll) 3. for each (type var : coll) 4. for (type var : coll) A straw poll indicated a preference for the alternatives 1 or 3, so these will be considered further. Subsequent discussion on the liaison reflector lead to a preference for A. for (type var : coll) or B. for (type var ; coll) // various TG5 members believe this is too error prone Meeting #4 (NJ): Bill will submit a proposal for the Jun meeting on the semantics of the for-each statement. Syntax remains as for each (type var in coll) Meeting #5 (Redmond): Bill reported that nothing need change in the TG5 spec in this regard. He's found library solutions for his STL .NET-related concerns.	Yes	
17	5-Dec-03	meeting #1 (TX)	17	Technical		John Spicer	Check on the UK submission to WG21 re opening nested namespaces.	Meeting #2 (Hawaii): John doesn't see a problem with the basic mechanism. Let WG21 handle this.	Yes	
18	5-Dec-03	meeting #1 (TX)	18.3.6	Technical		Bjarne Stroustrup	How might parameter arrays fit into sequence constructors being considered in WG21?	We liaised. No action.	Yes	
19	5-Dec-03	meeting #1 (TX)		Technical	L	Brandon Bray	list of overlap between Standard C++ and features proposed by C++/CLI	Meeting #9 (NJ): Close without action.	Yes	
20	8-Dec-03	Herb Sutter	18.7.1	Technical		Herb Sutter	Subject: RE: CLI binding: Delegating constructors and exceptions >>> "Herb Sutter" <hsutter@microsoft.com> 24 November 2003 18:33:42 >>> > Actually, it's in there, thanks to BSI. > EDG suggested that we specify the answer in terms of object lifetime, so that other answers, > including the destructor calling question, can just fall out from rest of ISO C++ which specifies > most things in terms of object lifetimes. In the 11/21	Herb responded. Resolved.	Yes	
21	24-Nov-03	Attila Feher		Editorial		Editor	When distilling PDF, add bookmarks. Look at other options too (such as hotlinks).		Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
22	24-Nov-03	Attila Feher	8.4	Technical			Base doc, pp. 17, line 43 (Automatic memory management). Object^ Pop() { if (first == nullptr) throw gnew Exception("Can't Pop from an empty Stack."); Why do you gnew the Exception? Is it necessary? There you throw a hat (handle), if I understand correctly. But why... Cannot even a value type just be thrown and make the catch box it, as it happens in C++?	Not an issue for TG5.	Yes	
23	16-Dec-03	Phone meeting	8.2.3	Editorial	R	Brandon Bray	Say more, especially w.r.t the template class array<element-type>.		Yes	
24	16-Dec-03	Phone meeting	9	Technical	R	Brandon Bray	Review this clause.		Yes	
25	16-Dec-03	Phone meeting	10	Technical	H	Brandon Bray	Revise this clause by covering topics including application entry point, assembly boundaries, among others.	Resolved on July 21, 2005 conference call. Incorporated into 1.14a.	Yes	
26	16-Dec-03	Phone meeting	10.2.1	Technical		Brandon Bray	Clarify the ordering definition when multiple accessibility keywords are used.		Yes	
27	16-Dec-03	Phone meeting	12.13.6	Technical	H	Brandon Bray	Describe how interior_ptr, pin_ptr, array, and safe_cast are template-like with certain constraints.		Yes	
28	16-Dec-03	Phone meeting	12.3.6	Technical	M	Brandon Bray	Describe how the compiler will need to emit a modopt to distinguish interior_ptr<T> from tracking reference to T (T%) in the metatada.		Yes	
29	16-Dec-03	Phone meeting	12.3.6.2	Technical	M	Brandon Bray	Spell out target type restrictions (for an interior_ptr)		Yes	
30	16-Dec-03	Phone meeting	12.3.6.3	Editorial		Brandon Bray	Describe the dangers of pointer arithmetic and interior_ptr.	merged into issue #87.	Yes	
31	16-Dec-03	Phone meeting	12.3.7	Technical		Brandon Bray	Provide a grammar for pinning_ptr	merged into issue #27.	Yes	
32	16-Dec-03	Phone meeting	13	Technical		Tom Plum	What, if anything, goes in this clause?		Yes	
33	16-Dec-03	Phone meeting	14.1.1	Editorial	R	Brandon Bray	Review this subclause.		Yes	
34	16-Dec-03	Phone meeting	14.4	Editorial	R	Brandon Bray	Review this subclause.		Yes	
35	16-Dec-03	Phone meeting	15.1	Technical	H	Brandon Bray	The rewrite rules for e[x] (default indexed accesses) are different where there is only one index. This is because there is a potential ambiguity with the C++ operator[]. Is this mentioned elsewhere?		Yes	
36	16-Dec-03	Phone meeting	15.3.8	Technical	M	Brandon Bray	cv-qualification needs to be considered for dynamic_cast.	Resolved on July 7, 2005 conference call. This issue is overcome by events. Dynamic cast can no longer unbox values, and thus there is no need to consider cv qualification.	Yes	
37	16-Dec-03	Phone meeting	15.3.9	Technical		Brandon Bray	Are typeid<long> and typeid<char> allowed (and if so, what do they mean).	They are allowed and are distinct.	Yes	
38	16-Dec-03	Phone meeting	15.3.9	Technical	L	Brandon Bray	Provide a spec for standard typeid (that returns std::type_info) in addition to the new typeid (that returns System::Type).	Meeting #9 (NJ): Close and list in Future Directions.	Yes	
39	16-Dec-03	Phone meeting	15.3.13	Editorial	H	Brandon Bray	Update this subclause		Yes	
40	16-Dec-03	Phone meeting	15.4.1.1	Editorial	R	Brandon Bray	Review this subclause.		Yes	
41	16-Dec-03	Phone meeting	15.4.1.4	Technical		All	Should a unary ^ operator exist?	Meeting #4 (NJ): No	Yes	
42	16-Dec-03	Phone meeting	15.4.6	Technical		Brandon Bray	Define the grammar for gnew array, and describe array creation expression.		Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
43	16-Dec-03	Phone meeting	15.11.1	Technical	M	Brandon Bray	Add support for handle equality comparison, and handle ==/!= nullptr, and vice versa.	Meeting #3 (Mel): Had a short discussion. Mark will produce a paper for the May meeting. Meeting #4 (NJ): No progress. To be discussed via email, and at the Jun meeting Meeting #5 (WA): Discussed briefly. Asked Mark to write this up and distribute to the reflector. Phone call Jun 29: This issue was resolved; just needs drafting of final words. Meeting 7 (WA): In the case of if(handle), which conversions are attempted before comparison against nullptr is used? We agreed that if an explicit conversion to bool exists, if(handle) uses that. There is no implicit unboxing. Steve and Mark worked on this and presented it to the full committee on the 2nd day. Based on committee feedback, Mark will write this up for future consideration. Phone call Aug 18, 2005: Resolved. Added sections for handle equality operators and string equality.	Yes	
44	16-Dec-03	Phone meeting	15.18	Technical	H	Brandon Bray	Add words to discuss assignment for properties and events from the point of view of the rewrite rules.		Yes	
45	16-Dec-03	Phone meeting	15.2	Technical		Brandon Bray	Investigate whether string literals include compile-time expressions, such as concatenation of strings with non-strings.	Meeting #4 (NJ): No action to be taken at this time.	Yes	Yes
46	16-Dec-03	Phone meeting	16.3	Technical		Jonathan Caves		Meeting #3 (Melbourne): It was suggested that this issue be brought to WG21. It's a security issue in standard C++; it's not a CLI-specific issue. Jonathan will produce a paper for the May meeting. Meeting #4 (NJ): TG5 expressed opposition to expression-level checked/unchecked. Not to bring it to WG21.	Yes	Yes
47	16-Dec-03	Phone meeting	17	Technical	M	Brandon Bray	Provide text for this clause (Namespaces)	Resolved on July 21, 2005 conference call. Incorporated into 1.14a.	Yes	
48	16-Dec-03	Phone meeting	18.3.1	Technical		Editor	Explain the difference between using 'override' and '= function-name'; one creates an .override directive in CIL, the other does not.		Yes	
49	16-Dec-03	Phone meeting	18.3.4	Technical		Brandon Bray	Describe in more detail the semantics of new, including its use on static member functions (currently new only applies to overriding, not to hiding).		Yes	
50	16-Dec-03	Phone meeting	18.4	Technical	H	Brandon Bray	Extend declarator-id's by adding a new production that allows default.	Resolved on July 21, 2005 conference call. Incorporated into 1.14a.	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
51	16-Dec-03	Phone meeting	18.4	Technical		Brandon Bray	The grammar for indexer-parameter-declaration does not allow handles or pointers, but full declarators are not needed. The grammar should allow a simpler sequence of ptr-operator.		Yes	
52	16-Dec-03	Phone meeting	18.4.2	Technical	H	Brandon Bray	This subclause only covers how the accessor functions must be defined. The expressions clause needs to cover the rewrite rules that call accessor functions.		Yes	
53	16-Dec-03	Phone meeting	18.4.2	Technical		Brandon Bray	Property syntax: Describe the qualified name of a property. Meeting #2 (Hawaii): Agreed to keep the current syntax		Yes	
54	16-Dec-03	Phone meeting	18.5.2	Editorial	R	Brandon Bray	Review this subclause.		Yes	
55	16-Dec-03	Phone meeting	18.6	Editorial	R	Brandon Bray	Review this subclause.		Yes	
56	16-Dec-03	Phone meeting	18.7.4	Technical	M	Brandon Bray	Identify when (operator) synthesis would and would not occur.		Yes	
57	16-Dec-03	Phone meeting	18.6.5.1	Technical	L	Brandon Bray	Writeup op_true and op_false operators	DUPE OF #145	Yes	
58	16-Dec-03	Phone meeting	18.6.6.1	Technical	M	Brandon Bray	Reword this subclause similarly to the way special member functions are described.	Meeting 7 (WA): ?? To be done in Tue morning work sessions. Phone call Aug 18, 2005: Resolved. Text deleted and replaced with section on handle equality.	Yes	
59	16-Dec-03	Phone meeting	18.6.6.1	Technical	H	Brandon Bray	Add another subclause to cover the compiler-generated conversion from handle to unspecified bool type.	Meeting 7 (WA): ?? To be done in Tue morning work sessions.	Yes	
60	16-Dec-03	Phone meeting	18.9	Technical		Brandon Bray	Add grammar for literal-constant-initializer = Standard C++ constant-initializer + float/double + String + nullptr.		Yes	
61	16-Dec-03	Phone meeting	18.9, 18.10	Technical		Brandon Bray	Justify why we need literal and initempty fields.	They are used in the BCL.	Yes	
62	16-Dec-03	Phone meeting	19.12.1	Technical	L	Brandon Bray	Add a description that for any value class we have to make the copy before calling member functions.	Meeting #9 (NJ): Needs to be done.	No	
63	16-Dec-03	Phone meeting	18.11	Technical	H	Brandon Bray	Say more about finalizers (including Dispose/~T and Finalize/!T) and add some examples.	Paper included in WD1.10.	Yes	
64	16-Dec-03	Phone meeting	19	Technical		Brandon Bray	Supply more text for this clause.		Yes	
65	16-Dec-03	Phone meeting	18.1	Technical		Editor	As a cross-language issue, come up with terminology to distinguish between destructors and finalizers. Perhaps "deterministic destructor" vs. "non-deterministic finalizer." Add some text in spec re this, esp. w.r.t C#'s use of destructor	Feb 2005. Issue was dropped as the revised version of Brandon's "Destructors and Finalizers" paper makes this intent clear, and TG2 has now dropped the use of "destructor" in favor of "finalizer".	Yes	
66	16-Dec-03	Phone meeting	21	Editorial	M	Brandon Bray	Introduce value classes -- Discuss the following: value classes are optimized for small data structures. As such, value classes do not allow inheritance from anything but interface classes. Tie in fundamental classes.	Resolved in the 1.14b draft circulated for the August 4, 2005 conference call.	Yes	
67	16-Dec-03	Phone meeting	21.4.1	Technical	H	Brandon Bray	Add words about instance constructors and static constructor. Value classes cannot have SMFs (specifically, default constructor, copy constructor, assignment operator, destructor, or finalizer. Need to add specification for this along with rationale.	Resolved on July 21, 2005 conference call. Incorporated into 1.14a.	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
68	16-Dec-03	Phone meeting	22	Technical	L	Brandon Bray	Consider writing some text for this "place-holder" clause. Should this all go in the new annex "Future directions"?	Meeting #9 (NJ): Existing words adequate.	Yes	
69	16-Dec-03	Phone meeting	23	Technical		Editor	The spec currently states "Throughout this Standard, the term "array" is used to mean an array in C++/CLI. A C++-style array is referred to as a native array whenever the distinction is needed." Tom was concerned that this was, perhaps, too subtle. He will try to come up with an alternative name for C++/CLI arrays. Meeting #2 (Hawaii): Use "Array" when we mean CLI array, and "array" means C-style array.		Yes	
70	16-Dec-03	Phone meeting	23	Technical		Sean Perry	Check if the term "array" is used in the library extensions plan of WG21.	Yes it is.	Yes	
71	16-Dec-03	Phone meeting	23	Editorial	R	Brandon Bray	Will review this whole clause.		Yes	
72	16-Dec-03	Phone meeting		Technical		Sean Perry	Look into possible performance issues re "for each" and delegates.	No information.	Yes	
73	16-Dec-03	Phone meeting	23.4	Technical		P.J. Plauger	Every array type inherits the members declared by the type System::Array. Currently, arrays do not have iterators compatible with Standard C++'s template library. Should they?	Meeting #5 (Redmond): Bill reported that nothing need change in the TG5 spec in this regard.	Yes	
74	16-Dec-03	Phone meeting	23.5	Technical	M	Brandon Bray	Write-up array covariance w.r.t arrays.	Phone call Aug 18, 2005: Resolved. Text included in handle conversion section.	Yes	
75	16-Dec-03	Phone meeting	23.6	Technical	M	Brandon Bray	Write up array initialization.	Phone call Aug 18, 2005: Resolved. Section written.	Yes	
76	16-Dec-03	Phone meeting	24.4	Technical	H	Brandon Bray	Address what happens when a ref class does not implement an interface function (and what happens when a base class has a non-virtual function with the same name).	Resolved in HI; incorporated into 1.11	Yes	
77	16-Dec-03	Phone meeting	25	Technical		Herb Sutter	Coordinate with WG21's extended enum proposal.	see #102	Yes	
78	16-Dec-03	Phone meeting	26.1	Technical		Brandon Bray	Redo the grammar for delegate-definition, and find a place for it in the type tree. Replace all uses of "return-type" with appropriate production.		Yes	
79	16-Dec-03	Phone meeting	27	Technical	H	Brandon Bray	Cover unification of CLI and Standard C++ exception-handling models, and anything else that might go in this clause. Are exceptions asynchronous now in some cases? Yes they are. (For example, NullReferenceException.)	Meeting #5 (WA): Kevin Free (Microsoft) gave a verbal presentation. catch(...) catches managed and native exceptions. catch(System::Object^) also catches both kinds, but won't invoke the destructor (so can leak). CLI exception handling supports more features than we expose. The issue remained with Brandon to write up, as before. Phone call Aug 18, 2005: Resolved. Enough text exists in the document. If a specific issue is raised, it will be another item on this spreadsheet. As is, this item is too broad to leave open.	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
80	16-Dec-03	Phone meeting	20.5.1	Technical		Brandon Bray	Check the name System::Reflection::DefaultMemberAttribute; it might have been renamed in the CLI standard.		Yes	
81	16-Dec-03	Phone meeting	20.5.2	Technical	R	Brandon Bray	Describe MethodImplOptions metadata generation.	The editor has added quite a bit of text re this attribute. See if that is sufficient.	Yes	
82	16-Dec-03	Phone meeting	29	Technical	M	Brandon Bray	Flesh out "Templates" clause. Explicit and partial specializations of a class template shall have the same class kind as the primary template. For example, an explicit specialization of a ref class template cannot be a value class. -- this isn't true, but should be covered. Are there any issues with metadata name emission? Is it even necessary to standardize this since template specializations are really only useful inside an assembly. Non-type template parameters will not include %, ^, or nullptr.	Phone call Aug 18, 2005. Resolved. Template issues covered by the spec are sufficient for closing this work item. If other issues about templates need to be written, they will be submitted as a separate paper or subsequent specific work items.	Yes	
83	16-Dec-03	Phone meeting	30	Technical		Editor	Flesh out "Generics" clause.		Yes	
84	16-Dec-03	Phone meeting	31	Technical		P.J. Plauger	Suggest possible standard library interaction issues apart from I/O synchronization.	Meeting #8 (WA): Decided to say nothing about this.	Yes	
85	16-Dec-03	Phone meeting	32	Technical		Brandon Bray	Flesh out "CLI libraries" clause.		Yes	
86	16-Dec-03	dummy entry							yes	
87	16-Dec-03	Phone meeting	A	Technical	L	Brandon Bray	Flesh out "Verifiable code" clause.	Meeting #9 (NJ): Close without action.	Yes	
88	16-Dec-03	Phone meeting	B	Technical	L	Editor	Flesh out "Documentation comments" clause.		Yes	
89	16-Dec-03	Phone meeting	C	Technical		Editor	Add any non-normative references		Yes	
90	16-Dec-03	Phone meeting	D	Technical		Editor	Add naming guidelines for generics		Yes	
91	29-Jan-04	meeting #2 (HI)	9.1.2	Technical		Editor	Steve asked: Keywords: Are they keywords or identifiers? If keywords, are they always present or only in some modes? Are they recognized at the lexical level or at the syntactic level? If at the syntactic level, what are the rules? (disambiguation?) Should keywords like ref class have a space in the keyword or are they two words?	Meeting #2 (Hawaii): Herb will write a paper on keywords to cover the following: 1) If it can be an identifier, it is. 2) Use Mark's preprocessor option 1 (to not make the spaced words pp tokens, but rather, to assemble them early in translation phase 4). 3) Add the fallback for namespace keywords. Address why "generic" shouldn't be spelled in some other way, perhaps as a spaced keyword, so that it need not be a regular keyword. Meeting #3 (Melbourne): Done, accepted, Editor to integrate. Steve will add more words (see issue #121).	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raisher?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
92	29-Jan-04	meeting #2 (HI)		Technical	M	Brandon Bray	<p>"size size" name lookup issue (see email thread started by Herb Sutter on January 14 on the liaison reflector under the topic {Name lookup 1 (of 2): "Size Size" (CLI property naming idiom)}.)</p> <p>This is the common CLI idiom of naming a property (or potentially other members) with the same name as its type. In particular, here are two common examples:</p> <pre>value class Size { /*...*/ }; value class Color { /*...*/ }; ref class X { public: property Size Size; property Color Color; };</pre> <p>In other languages, it's easy to simply use the identifier "Size" without qualification and have the compiler Do the Right Thing™. But C++ name lookup is different. The status quo in Managed C++ syntax was that we made no change to C++ lookup rules, with the result that authors of classes that use this idiom are required to qualify most occurrences of "Size" which is ugly. The issue mostly appears only within the class itself (and in derived classes).</p> <p>Here's a brief description of the problem:</p> <pre>ref class X { public: property Size Size { Size get() { return s_; } void set(Size s) { s_ = s; } // A</pre>	Meeting #8 (WA): Decided to not include this in V1.	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
93	29-Jan-04	meeting #2 (HI)	12.1	Technical		Tom Plum	<p>Do we require exact mapping for types, or do we allow a certain amount of flexibility?</p> <p>Should the size and representation of types long, long long, and long double (as well as wchar_t, see issue #5) be implementation-defined. Should all (or almost all) of the fundamental types being implementation-defined.</p> <p>The CLI types System::Single and System::Double require IEEE (IEC 559) representation. On many systems these naturally map to float and double, respectively. However, the IBM 390 does not used IEEE format for either of these types. A C++/CLI program running in that environment would want float/double to map to 390 types, so there would need to be a conversion to/from the CLI floating types.</p> <p>In order to encourage the writing of portable code, we'd need the largest core of fundamental type mapping as possible; for example, signed and unsigned 8-, 16-, and 32-bit integer mapping.</p>	<p>Meeting #3 (Mel): There was a lengthy discussion. No resolution.</p> <p>Meeting #4 (NJ): There was a lengthy discussion.</p> <p>Meeting #5 (WA): There was another lengthy discussion, which resulted in Plum's notes being incorporated into the meeting minutes.</p> <p>The edits from Plum's subsequent paper were incorporated into WD1.6 for Meeting #6 (WA).</p>	Yes	
94	29-Jan-04	meeting #2 (HI)		Technical	M	Brandon Bray	<p>Relationship between primitive types and CLI types.</p> <p>The current spec allows the following: int i = 10; String^ s = i.ToString(); Standard C++ doesn't allow member selection on expressions of primitive type. Assuming int maps to System::Int32, just how much alike are these two types? Specifically, when do we treat the primitive as the underlying class.</p>	<p>Meeting 5 (WA): Asked Mark to write this up and distribute to the reflector. Please address the side-effect issue; that is, given (i++).ToString, is the increment done?</p> <p>Meeting 7 (WA): ?? To be done in Tue morning work sessions.</p> <p>Re the side-effect, yes, it must be done.</p> <p>Phone call Aug 18, 2005: Resolved. Added text to 12.1.1 is fundamental types</p>	Yes	
95	29-Jan-04	meeting #2 (HI)	10	Technical	H	Brandon Bray	Provide words for #using.	The editor has added quite a bit of text re this topic.	Yes	
96	29-Jan-04	meeting #2 (HI)	9.1.1	Technical	M	Editor	The spec does not provide a way to use a keyword as an identifier. (VC++ uses the intrinsic __identifier(name) to achieve this; C# uses a leading @.) This is an issue for inter-operability; for example, being a consumer of a public type (written in something other than C++) that has a name (or contains a public member that has a name) that is a keyword in C++.	Meeting #8 (WA): It was proposed we support the intrinsic approach, accepting __identifier(x), where x is a string literal or an identifier. String version is reserved for implementers.	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
97	29-Jan-04	meeting #2 (HI)		Technical		Editor	Overloading on arity. (This is a liaison issue with TG3.) The issue involves the overloading of a non-generic type with a one or more generic types of the same name in the same namespace. For example, the following is permitted by the CLS: ref class X { /*...*/ }; generic<typename T> /*...*/ ref class X { /*...*/ }; generic<typename T, typename U> /*...*/ ref class X { /*...*/ };	Meeting 3 (Mel): Herb presented this issue, which was then reassigned to Brandon. Meeting 5 (WA): In this version, we'll support a generic and non-generic version of a type in the same namespace, but not in different namespaces. There was a discussion about using something like "using generic x::y" to provide cross-namespace support as well. Rex to work with Brandon to get this into the draft. Meeting 7 (WA): Herb reported that the MS implementation can consume same-named generics that overload on arity in the same assembly, but it cannot create them.	Yes	
	29-Jan-04	meeting #2 (HI)	30	Technical	R	Brandon Bray	Restrictions on generics re generic code generation. The current generics clause needs to be fleshed out, especially w.r.t how overload resolution works within the CLI. Some issues to consider are: (1) using templates inside of generics, (2) overloading rules, and (3) dynamic cast to type parameters. The high level goal with generics (as with other parts of C++/CLI) is to provide a close mapping of the underlying capabilities of the CLI, which means that C++ can potentially create generics that other languages might not be able to consume. Not all languages support all capabilities, but C++/CLI supports more than most. (However, C++/CLI does not support array co- or contra-variance.)	Meeting #2 (HI): Brandon will write a paper on this. Meeting #4 (NJ): The fleshing out of Clause 30 is a significant contribution toward this. More work needed in declarations and function calls.	No	
99	29-Jan-04	meeting #2 (HI)		Technical		Daveed Vandevoorde	Write a paper proposing properties as specified by C++/CLI, for the March 2004 meeting of WG21.		Yes	
100	29-Jan-04	meeting #2 (HI)		Technical		Herb Sutter	nullptr: Write a paper proposing this to WG21.	Meeting #4 (NJ): WG21 expressed interest.	Yes	
101	29-Jan-04	meeting #2 (HI)		Technical		Herb Sutter	delegating constructors: Write a paper proposing this to WG21.	Meeting #4 (NJ): No implementation of this is expected anytime soon. TG5 agreed to not include this in this round. Editor will move 8.8.7.1 and 18.7.1 to Annex E, and remove any usage of delegating constructors from examples in other clauses.	Yes	Yes
	29-Jan-04	meeting #2 (HI)		Technical		Herb Sutter	enhanced enums: Write a paper proposing this to WG21.	Meeting #4 (NJ): WG21 doesn't like enum class. WG21 doesn't know yet what it wants to do in this regard. However, if WG21 adopts a feature like this, but with different syntax, TG5 will revisit this when appropriate.	Yes	
103	29-Jan-04	meeting #2 (HI)		Technical		Brandon Bray	Explicit overriding: Propose to WG21	Meeting #4 (NJ): withdrawn	Yes	
104	29-Jan-04	meeting #2 (HI)		Technical		Steve Adamczyk	sealed, on classes and methods: Propose to WG21	Meeting #4 (NJ): withdrawn	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
105	29-Jan-04	meeting #2 (HI)	14.5.1	Technical	M	Brandon Bray	Constructors can't be used in casts in managed classes. Should they be allowed in explicit conversions? All managed type constructors being explicit by default. (Already yes, but reconfirm this.)	Meeting #4 (NJ): Steve will send the editor sufficient text to go into the public drop to indicate our intention re this topic. DONE. Meeting 5 (WA): Asked Mark to write this up and distribute to the reflector. Meeting 7 (WA): Steve and Mark worked on this and presented it to the full committee on the 2nd day. Mark will write this up for future consideration. Phone call Aug 18, 2005: Resolved. Deleted contradictory text in 14.5.1. Added 13.3 to cover the three different cases for direct initialization.	Yes	
106	29-Jan-04	meeting #2 (HI)		Technical		Editor	Should >> handled as two tokens rather than one; e.g., List<List<int>>.	Meeting #3 (MeI): Had a short discussion. Tom will produce a paper for the May meeting. Meeting #4 (NJ): TG5 agreed that if a < for a template is seen, and >> that are not inside parentheses, that >> will always be considered to be the closing delimiter of two < symbols, and results in an error if there are not two such corresponding < symbols. Refer to Daveed's paper WG21/N1649 for more information. Meeting #7 (WA): This paper was updated (see N1699). It was discussed in TG5 and will be discussed at the up-coming WG21 meeting, at which TG5 members will participate. Meeting #8 (WA): Daveed presented this at the WG21 meeting this week. He proposed option 1, to which WG21 agreed. He was charged to write the final words. Meeting #9 (NJ): Daveed submitted a revised paper, which was accepted. Implemented in WD1.10.	Yes	
107	29-Jan-04	meeting #2 (HI)		Technical		Editor	Look at the usage of the term "object" within the spec, and compare with the C++ std.		Yes	
108	19-Feb-04		12.3.6	Technical		Brandon Bray	Provide syntax for interior_ptr		Yes	
109	19-Feb-04		12.3.6.3	Technical	L	Brandon Bray	Cover the dangers of pointer arithmetic and interior_ptr	Meeting #9 (NJ): Close without action.	Yes	
110	19-Feb-04		12.3.7.1	Technical		Brandon Bray	Provide syntax for pinning_ptr		Yes	
111	19-Feb-04		15.3.2	Technical	M	Brandon Bray	Need to consider how indexed access expressions are interpreted in templates.	Resolved on July 21, 2005 conference call. Incorporated into 1.14a.	Yes	
112	19-Feb-04		15.3.9	Technical		Brandon Bray	Check if long::typeid, char::typeid, etc. are allowed (and if so, what do they mean).	Meeting #4 (NJ): Allowed, but no modopts	Yes	
113	19-Feb-04		28.5.1.2	Technical		Brandon Bray	Provide text for MethodImplOptions attribute	duplicate	Yes	
114	19-Feb-04		15.4.6.2	Technical		Brandon Bray	Does new-initializer need to be changed?		Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
115	19-Feb-04		15.2	Technical		Brandon Bray	Do string literals include compile-time expressions, such as string concatenation?	duplicate	Yes	
116	19-Feb-04		18.4.2	Technical	H	Brandon Bray	Add some discussion of how accesses to properties are rewritten into accessor functions. This should be covered in rewrite rules in the expressions clause. Note that access checking for whether a property can be written to or read from is done after rewriting and overload resolutions.		Yes	
117	19-Feb-04		18.4.2	Technical	H	Brandon Bray	The qualified name of a property needs to be described somewhere. Once that happens, how an out-of-class definition is done will already be covered by existing rules.	Resolved on July 21, 2005 conference call. Incorporated into 1.14a.	Yes	
118	19-Feb-04		23.1.1	Technical		Editor	Is reference conversion the correct term?	No; it's a handle conversion	Yes	
119	19-Feb-04		28.5.1.1	Technical		Editor	Check this name (DefaultMember); this attribute might have been renamed in the CLI standard.	It has not been renamed, and appears in Beta 1 with that name.	Yes	
120	19-Mar-04	meeting #3 (Mel)		Technical		Tom Plum	Does typename allow us to pursue a containment policy re elaborated specifiers?	Meeting 7 (WA): Decided to drop this issue.	Yes	
121	19-Mar-04	meeting #3 (Mel)		Technical		Steve Adamczyk	In the context of Herb's keywords paper (2004-05), Steve will write up the notion "If it can be an identifier, it is."		Yes	
122	19-Mar-04	meeting #3 (Mel)		Technical		Steve Adamczyk	Write a WG21 paper on extended integer types, promotion rules, costs of conversion, and the like, for the May meeting.	Meeting #4 (NJ): Not yet done, but still planned.	Yes	
123	3-May-04	meeting #4 (NJ)		Technical		Tom Plum	The draft uses the term "constructed type". It was suggested that the corresponding Standard C++ term is"instantiation". Which should we use?	Meeting 7 (WA): Chose to use "constructed type". No change needed to the spec.	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
124	10-Jun-04	Jonathan Caves		Technical	M	Brandon Bray	<p>Indexed properties -- Consider the following:</p> <pre>interface class I1 { property int Value; }; interface class I2 { property int Value[String^] { int get(String^); void set(String^, int); }; }; ref class D : I1, I2 { // Implements the properties }; D^ d; d->Value["Foo"];</pre> <p>The question is what does the last line do?</p> <p>Which leads to a language design question - what should the compiler do when faced with a property followed by a '['</p> <p>1) Should it look for just parameterized properties and if there isn't one fail - I suspect not</p> <p>2) Should it look for all properties and if the returned set contains a parameterized property it should prefer it - this sounds like magic to me.</p> <p>3) Should it look for all properties perform overload resolution across the whole set and if the resulting call is ambiguous then issue an error.</p> <p>Mark Hall says: Jonathan's looking into deferring the</p>	<p>Meeting #5 (WA): Discussed this. Option #3 preferred.</p> <p>Meeting 7 (WA): Discussed this in detail.</p> <pre>property int Value[int] { void set(int, int); }; x->Value[1] = 4 is treated as x->set_Value(1,4); ----- property array<int>^ Value { array<int>^ get(); } x->Value[1] = 4 is treated as x->get_Value()[1] = 4 ----- property int% Value[int] { int% get(int); } x->Value[1] = 4 is treated as x->get_Value(1) = 4</pre> <p>This construct violates the principle of properties (that of setting/getting the value of some property), so is not to be encouraged; however, it is supported, but no need to consider it further here.</p>	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
125	14-Jun-04	meeting #5 (WA)	8.15.3	Technical	M	Brandon Bray	Based on the rules for type deduction in templates, it seems surprising that you can match <code>array<ItemType>^</code> with an argument of type <code>int</code> . Here is a standard C++ example intended to illustrate the issue: template <class ItemType> struct Stack {}; template <class ItemType> struct Array { Array(ItemType); }; template <class ItemType> void PushMultiple(Stack<ItemType>, Array<ItemType>); int main() { Stack<int> s; PushMultiple(s, 1); // deduction fails PushMultiple<int>(s, 1); } Are the rules for generic different in this area? [There seems to be information related to this in 30.3.2. See that subclause for further comments on this issue.]	Phone call Aug 18, 2005: Resolved. The example suggested the wrong behavior, so the example was changed.	Yes	
126	14-Jun-04	meeting #5 (WA)	12.1	Technical		Editor	The type <code>long long</code> will be defined by pointing to the <code>long</code> type.	Meeting 7 (WA): Steve has produced a revised version, N1693. Editor to fold this in the spec. TG5 understands that WG21 has not yet accepted this paper, but is expected to at its Oct 2004 meeting.	Yes	
127	14-Jun-04	meeting #5 (WA)	12.3.3	Technical	L	Brandon Bray		Add text to indicate the circumstances under which	Meeting #9 (NJ): MS-specific; Close without action.	Yes
128	14-Jun-04	meeting #5 (WA)	12.3.6	Technical	L	Brandon Bray	The compiler will need to emit a <code>modopt</code> to distinguish <code>interior_ptr<T></code> from tracking reference to <code>T</code> (<code>interior_ptr<T></code>).		Yes	
129	14-Jun-04	meeting #5 (WA)	12.3.7	Technical	L	Brandon Bray	Need to add text to indicate the circumstances under which the <code>modopt IsPinned</code> shall be emitted (i.e., <code>IsPinned</code>).		Yes	
130	14-Jun-04	meeting #5 (WA)	14.1.1	Technical	L	Brandon Bray	Separate the list of conversions from the order of promotion.	Meeting #9 (NJ): Close without action.	Yes	
131	14-Jun-04	meeting #5 (WA)	15.3.3	Technical	M	Editor	Add text to indicate the circumstances under which type modifiers shall be emitted, and point to each modifier's definition.	Meeting #9 (NJ): Needs to be done. Done in WD1.10.	Yes	
132	14-Jun-04	meeting #5 (WA)	15.3.10	Technical	M	Brandon Bray	Unboxing and boxing are described as preferred uses.	Resolved on July 21, 2005 conference call. Incorporated into 1.14a.	Yes	
133	14-Jun-04	meeting #5 (WA)	15.3.10	Technical	L	Brandon Bray	In a static cast of a handle to a base type to a handle for a derived type, there is no checking. This can be unverifiable and might cause a gc hole.	Meeting #9 (NJ): Close without action.	Yes	
134	14-Jun-04	meeting #5 (WA)	16.3.3	Technical	M	Editor	Add text to indicate the circumstances under which the <code>modreq IsUdtReturn</code> shall be emitted (i.e., ref class type returned by value). Point to that <code>modreq</code> 's spec.	Meeting #9 (NJ): Needs to be done. Done in WD1.10.	Yes	
135	14-Jun-04	meeting #5 (WA)	18	Technical	R	Brandon Bray	This table and corresponding sections should include Special Member Functions (SMFs) like destructors.		Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
136	14-Jun-04	meeting #5 (WA)	18.2.1	Technical		Editor	Need to address the following: C++/CLI uses the System::Reflection::DefaultMemberAttribute attribute to specify that something other than the default name, “Item”, should be used. Given that, the text describes what happens if no name is chosen; that is, Item is used by default. Once the name has been set with DefaultMember, it cannot be changed in a derived class. If two interfaces have different DefaultMember attributes, implementing both interfaces is ill-formed.	Meeting #9 (NJ): Editor to mention this in the default indexer clause. Incorporated in WD1.10.	Yes	
137	14-Jun-04	meeting #5 (WA)	18.3	Technical		Brandon Bray	Extend the grammar to accommodate attributes on functions.		Yes	
138	14-Jun-04	meeting #5 (WA)	18.4	Technical		Mark Hall	Need to write up the restrictions on trivial properties.	There is no record of what this was referred to. Closed on Aug 4, 2005 conference call.	Yes	
139	14-Jun-04	meeting #5 (WA)	18.4	Technical		Editor	We probably should say something about the reserved names get_Item and set_Item, and their relationship with default indexed properties. Also, add a forward pointer to the corresponding attribute.	Meeting #9 (NJ): Needs to be done. Handled as part of the resolution of #136.	Yes	
140	14-Jun-04	meeting #5 (WA)	18.5	Technical		Brandon Bray	The production event-type has not yet been defined.	The syntactic category of this element needs to be	Yes	
141	14-Jun-04	meeting #5 (WA)	18.5.2	Technical		Brandon Bray	It is a bit strange to define grammar productions for these functions. We probably should either make t		Yes	
142	14-Jun-04	meeting #5 (WA)	18.5.3	Technical	L	Brandon Bray	An event with the new modifier introduces a new event that does not override an event from a base class. Make sure the complete specification is provided in the clause for the new modifier.	Meeting #9 (NJ): Already in draft.	Yes	
143	14-Jun-04	meeting #5 (WA)	19.7	Technical	L	Brandon Bray	The first citation of the C++ Standard does not apply to non-static member operators – that need not have a parameter of the type of the class.	Meeting #9 (NJ): Needs to be done.	No	
144	14-Jun-04	meeting #5 (WA)	18.6.1	Technical	L	Brandon Bray	Provide an example for "Homogenizing the candidate overload set".		Yes	
145	14-Jun-04	meeting #5 (WA)	18.6.5.2	Technical		Editor	Provide C++ names for operator True and False	Meeting #8 (WA): Move to future directions and close out.	Yes	
146	14-Jun-04	meeting #5 (WA)	18.9	Technical		Brandon Bray	add literal to storage-class-specifier		Yes	
147	14-Jun-04	meeting #5 (WA)	18.1	Technical		Brandon Bray	add initonly to storage-class-specifier		Yes	
148	14-Jun-04	meeting #5 (WA)	20.2	Technical		Editor	Add text to indicate the circumstances under which type modifiers shall be emitted, and point to each modifier's definition.	Meeting #9 (NJ): Needs to be done. Done in WD1.10.	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
149	14-Jun-04	meeting #5 (WA)	20.3	Technical	L	Editor	Add text to indicate the circumstances under which type modifiers shall be emitted, and point to each modifier's definition.	Meeting #9 (NJ): Needs to be done. Done in WD1.10.	Yes	
150	14-Jun-04	meeting #5 (WA)	21.4.1	Technical		Brandon Bray	Add words about instance constructors and static constructor.		Yes	
151	14-Jun-04	meeting #5 (WA)	25.2	Technical	M	Brandon Bray	The note says "pickup the restrictions from page 33	Phone call Aug 18, 2005: Resolved. This referred to elements of the array chapter, which is now complete.	Yes	
152	14-Jun-04	meeting #5 (WA)	25.1.3	Technical		Brandon Bray	Complete the production enum-base. Also, since this production is used by both native and CLI enums, yet it's described in the native section, wording might need to be re-arranged to make it read better from both enums' perspectives.		Yes	
153	14-Jun-04	meeting #5 (WA)	30.1	Technical	M	Brandon Bray	The text indicates that a generic-declaration may appear in a class scope, but the syntax of member-declaration has not been extended to permit a generic-declaration. [[#98]]	Resolved on July 21, 2005 conference call. Incorporated into 1.14a.	Yes	
154	14-Jun-04	meeting #5 (WA)	31.1	Technical	R	Brandon Bray	Doesn't the text "a generic name declared in namespace scope or in class scope shall be unique in that scope" make the first sentence of this paragraph redundant? Re the reference to 14.5.4: That is the section on partial specialization. Generics can't be partially specialized, can they? The spec. should probably answer that explicitly.		No	
155	14-Jun-04	meeting #5 (WA)	30.1	Technical	R	Brandon Bray	What is a non-generic type? Does it mean that the rules are the same as classes? As template classes? Something else?		No	
156	14-Jun-04	meeting #5 (WA)	30.1	Technical		Editor	Can generic types be nested in native classes?	Included in WD1.10.	Yes	
157	14-Jun-04	meeting #5 (WA)	30.1	Technical		Brandon Bray	Type Overloading – This involves overloading on arity, and is currently under investigation. Such a feature permits the following: ref class X {}; generic<typename T> ref class X {}; generic<typename T, typename U> ref class X {};	Duplicate of #97	Yes	
158	14-Jun-04	meeting #5 (WA)	30.1.1	Technical	R	Brandon Bray	Regarding the phrase, "A generic-parameter defines its identifier to be a type-name", the equivalent wording for template parameters in the working paper has been changed to "defines its identifier to be a typedef-name". The revised wording should probably be used here too (see core issue 283)		No	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
159	14-Jun-04	meeting #5 (WA)	30.1.2	Technical	R	Brandon Bray	30.1.2 says "Like templates in Standard C++, within the body of a generic type any usage of the unqualified unadorned name of that type is assumed to refer to the current instantiation." 30.1.3 then goes on to describe "The instance type". Those seem like to different ways of describing the same concept. Can they be unified in some way?		Yes	
160	14-Jun-04	meeting #5 (WA)	31.1.6	Technical	R	Brandon Bray	This subclause describes when a static constructor is invoked. In 18.8, it references the CLI Standard Partition II (10.5.3). Are the rules the same? (Yes) Should this subclause also just reference the CLI spec? There are two sets of behavior; we need to say which one we use.		No	
161	14-Jun-04	meeting #5 (WA)	30.1.7	Technical	M	Brandon Bray	What to say about explicit conversion functions (w	Phone call Aug 18, 2005: Resolved. It wasn't really clear anything needed to be said, but an explicit mention of conversion functions was mentioned.	Yes	
162	14-Jun-04	meeting #5 (WA)	31.2.2	Technical	R	Brandon Bray			No	
163	14-Jun-04	meeting #5 (WA)	30.2.4	Technical	R	Brandon Bray	"The non-inherited members of a constructed type are obtained by substituting, for each generic-parameter in the member declaration, the corresponding generic-argument of the constructed type. The substitution process is based on the semantic meaning of type declarations, and is not simply textual substitution." It would be helpful to explain this in more detail and/or give an example where this makes a difference.		Yes	
164	14-Jun-04	meeting #5 (WA)	30.3	Technical		Editor	Can a generic function be declared inside a native class? (Yes) Can generic functions (and member fun		Yes	
165	14-Jun-04	meeting #5 (WA)	30.3	Technical		Editor	Types not used as a parameter type to a generic function cannot be deduced. Are the nondeduced context rules the same as Standard C++ or not? The sentence before this is true, but not complete if the rules are the same as Standard C++.	Meeting #8 (WA): The intent for V1 is to use the same rules as for templates. Meeting #9 (NJ): Say the following: "Types that cannot be deduced for function templates cannot be deduced for generic functions."	Yes	
166	14-Jun-04	meeting #5 (WA)	30.3	Technical		Editor	What, if anything, does it mean for a generic functi	Meeting #6 (WA): all have the usual meaning.	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
167	14-Jun-04	meeting #5 (WA)	31.3	Technical	L	Brandon Bray	"When the type of a parameter or variable is a type parameter, the declaration of that parameter or variable shall use that type parameter's name without any pointer, reference, or handle declarators." What about cv-qualifiers?	Meeting #9 (NJ): Needs to be done. CV-qualifiers are not permitted.	No	
168	14-Jun-04	meeting #5 (WA)	30.3	Technical	L	Brandon Bray	Can you take the address of a generic function instance?	Meeting #6 (WA): Tentatively decided, NO. Meeting #8 (WA): Reconsidered, and now think YES. Consider the following example: delegate void D(int); generic <class T> void F(T t); D^ d = gcnew D(&F<int>); We agreed that this was a useful idea.	Yes	
169	14-Jun-04	meeting #5 (WA)	30.3.2	Technical	L	Brandon Bray	The issue raised in 8.15.3 (#125 in this list) is somewhat answered here. 18.3.6 seems to deal with expanded forms of calls, not expanded forms of function declarations. I interpret the text above as saying that deduction is done as if the function were declared like this: generic <typename ItemType> void PushMultiple(Stack<ItemType>^, ItemType i1, ItemType i2,/* ... */); Is that correct? I think this requires a more detailed description.	Meeting #9 (NJ): Needs to be done. Add example(s).	No	
170	14-Jun-04	meeting #5 (WA)	31.3.2	Technical	L	Brandon Bray	Something needs to be said about instantiating a generic delegate using a generic function.	Meeting #9 (NJ): Needs to be done.	No	
171	14-Jun-04	meeting #5 (WA)	30.4.2	Technical	H	Brandon Bray	When are members considered hidden? Is it using the rules described later? Those are described as applying only when a type parameter has both a class constraint and one or more interface constraints though.	Meeting #9 (NJ): Needs to be done. Resolved in HI; incorporated into 1.11	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
172	14-Jun-04	meeting #5 (WA)	30.4.4	Technical	H	Brandon Bray	Miscellaneous generics issues: 1. I seem to recall discussions of other kinds of constraints (I believe one of them concerned whether you could do a "new T()"). 2. Doesn't there need to be some discussion of how overload resolution works when a function argument has a type parameter as its type? 3. Are the typename and template rules for syntactic disambiguation the same in generics as in templates? Presumably, the lack of specialization would eliminate the need for these. 4. If scope contains a set of overloaded generic functions, is partial ordering used to choose between them? 5. I assume since there is nothing that says otherwise, that generics can be friends of other classes and generics can make other classes, functions, (including generics) friends? 6. If friendship is supported, can a generic first be declared in a friend declaration (suggested answer: no). 7. Standard C++ has restrictions on type parameters such as prohibiting types with no linkage. Does this rule apply to generic arguments? 8. Are there generic conversion functions?	Meeting #8 (WA): 1. For V1, we can consume and enforce these special constraints, but we can't author them. However, we plan to do so in future, so add this to "Future directions". Resolved in HI; incorporated into 1.11	Yes	
173	14-Jun-04	meeting #5 (WA)	32.1.4	Technical	L	Brandon Bray	To ensure that signatures for the same Type produced by different implementations match, the ordering in such a set of modreqs and modopts is as follows: first modreqs in ascending order by name, then modopts in ascending order by name, with case being significant. [[We need some rule here; is this the one?]]. The above isn't the correct wording, so I removed it from the draft. The ordering probably should be specified.	Meeting #9 (NJ): Add a description of our best guess at the correct solution, to Future Directions, then mark this Postponed. Point to this from the normative text somehow.	No	
174	14-Jun-04	meeting #5 (WA)	32.1.4	Technical	L	Brandon Bray	If IsBoxed is retained for the standard, we have an ordering issue to consider: Currently, the value-type special modopt is emitted before the IsBoxed modreq. For example, class [mscorlib]System.ValueType modopt([mscorlib]System.Int32) modreq([a]n.IsBoxed). That puts a modopt before a modreq.	Meeting #9 (NJ): MS-specific; Close without action.	Yes	
175	14-Jun-04	meeting #5 (WA)	32.1.5.1	Technical	L	Brandon Bray	This modifier [IsBoxed] is a workaround for the MS implementation. Does it have any long-term value for the standard, even if only as an historical note?	Meeting #9 (NJ): MS-specific; Close without action.	Yes	
176	14-Jun-04	meeting #5 (WA)	E	Technical	R	Brandon Bray	Flesh out Future Directions		Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
177	14-Jun-04	meeting #5 (WA)	E.7	Technical		Brandon Bray	Add text to show the behavior in the CLI (including	Feature dropped. So no need to persue.	yes	
178	14-Jun-04	meeting #5 (WA)	F	Technical		Brandon Bray	Flesh out anything in incompatibilities with Standard C++	Duplicate so closed this one.	Yes	
179	23-Jul-04	TG3 liaison		Technical		Mark Hall	Support for Hide-By-Signature on Methods in ref classes (This would also apply to setter/getter methods for properties.)	See email thread started by Rex J. on Jul 24. Meeting #6 (WA): Some possible ways to address this (and results of a straw poll) are: 1) Support hidebyname only and issue better error messages. [0 in favour] 2) Make all ref class methods be hidebysig; a. Only [0 in favour] b. Default, with an option to select hidebyname [6 in favour] 3) Add hidebysig keyword to allow explicit marking of methods. [0 in favour] with 3 people unsure. We could go two routes: A) Bring hidebysig in via "using" directive to hoist base class/interface names (this is an approximate solution only, as it doesn't allow hoist-by-signature, only hoist-by-name) [0 in favour] B) Do repeated lookup in all base classes (like C#) [8 in favour] Tom circulated the relevant pages from the CLI spec (Partition I, 7.10.4). We need to take into account the CLS rules when resolving this issue. Meeting #7 (WA): Had a brief discussion. No progress.	Yes	
180	14-Jun-04	meeting #5 (WA)	26	Technical		Editor	Committee agreed with Rex's proposal to require that delegates have the optional BeginInvoke and EndInvoke methods for async processing of delegates.	This was reported to TG3 at its Jun 04 meeting, but there were concerns about the Compact Profile's not being required to support these at runtime. Since this is still an open issue in TG3, this issue will remain open in TG5.	Yes	
181	27-Jun-04			Technical		Tom Plum	Here are Tom's assumptions: C++/CLI will not initially have a built-in type for decimal the way C# has. In C++/CLI, you have to use namespace System::Decimal. The C++/CLI draft doesn't specify anything about semantics of Decimal; the requirements are as given in CLI (TG3). So we benefit from all the work done in TG3 on allowing IEEE Decimal as an alternative to .NET Decimal. Re the methods of the type System::Decimal methods, are they adequate for the C++ programmer, or should the compiler know something special about Decimal?	Phone call Jun 29: discussed Decimal; agreed C++/CLI can just use constructors.	yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
182	26-Jul-04	phone meeting		Technical	H	Brandon Bray	Discussion of passing a string literal in the presence of overloads taking String^ and const char * (what about char *?)	Meeting #6 (WA): The compiler currently chooses the String^ over the const char*. Involves type deduction across templates and generics. Reassigned from Mark to Brandon. String literal portion of issue 12 was transferred to #182.	Yes	
183	2-Aug-04	meeting #6 (WA)		Technical	M	Brandon Bray	Overload assignment operator for handles.	Post-meeting #7. MS design team discussed this and believes that we should drop this issue. Meeting #8 (WA). Decided to drop it.	Yes	
184	2-Aug-04	meeting #6 (WA)		Technical	L	Herb Sutter	Describe problem with overloading on % vs. & Herb presented the following code: #include <iostream> using namespace std; void f(const int&) { cout << "f(const int&)" << endl; } void f(int&) { cout << "f(int&)" << endl; } void g(int%) { cout << "g(int%)" << endl; } void g(int&) { cout << "g(int&)" << endl; } int main() { const int ci = 0; int i = 0; int^ hi = gcnew int; f(ci); f(i); g(*hi); // g(i); // ambiguous: should g(int&) be preferred? } The following code was his attempt to write an agnostic swap: template<typename T> void swap(T% a, T% b) { #if defined NO_PIN_PTR // doesn't work T temp = a; a = b; b = temp; #elif defined PIN_PTR_BUG // doesn't compile T temp = *pin_ptr<T>(a); *pin_ptr<T>(*pa) = *pin_ptr<T>(*pb); *pin_ptr<T>(*pb) = temp; }		No	
185	2-Aug-04	meeting #6 (WA)		Technical		Herb Sutter	Collapsing reference to reference. (It's in the C++0x spec.)	Meeting #9 (NJ): Close without action.	Yes	
186	2-Aug-04	meeting #6 (WA)		Technical	M	Brandon Bray	Should we standardize traits?	Meeting 9 (NJ): Agreed to drop this.	Yes	
187	2-Aug-04	meeting #6 (WA)		Technical		Brandon Bray	user-defined assignment operator for handles	dupelicate of #183	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
188	2-Aug-04	meeting #6 (WA)		Technical	H	Brandon Bray	Look at using + to implement String concatenation.		Yes	
189	2-Aug-04	meeting #6 (WA)		Technical		Editor	Look at the changes to the grammar for C++0x and note where they affect the C++/CLI grammar.	Put note in clause 3 using Steve's note to me as an example. Done in WD1.10.	Yes	
190	2-Aug-04	meeting #6 (WA)		Editorial		Editor	Add an annex identifying behavior that is implementation-defined, undefined, or unspecified.		Yes	
191	2-Aug-04	meeting #6 (WA)		Technical	R	Brandon Bray	Review the specification checking the usage of <u>accessibility vs. visibility</u>		Yes	
192	2-Aug-04	meeting #6 (WA)		Technical	L	Brandon Bray	Provide an annex containing the differences between the grammar of Standard C++ and C++/CLI	Meeting #9 (NJ): Close without action.	Yes	
193	2-Aug-04	meeting #6 (WA)		Technical		Sean Perry	Look at the issue of whether or not the mapping of bool should be implementation-defined.	Meeting 7 (WA): Sean wrote this up and presented it to the full committee on the 2nd day. Based on committee feedback, Sean will revise his paper for future consideration. This was integrated into WD1.9.	Yes	
194	2-Aug-04	Anthony Williams	15.3.2	Technical	M	Brandon Bray	Re Anthony's post to the reflector re "default indexe	Meeting 7 (WA): Discussed the possibility of disallowing both the default indexed property and operator[]. Phone call Aug 18, 2005: Resolved. The text of 15.3.1 disallows this in accordance with meeting 7.	Yes	
195	25-Aug-04	Rex Jaeschke	14.1.	Technical	L	Brandon Bray	Separate the list of conversions from the order of preference (such as how Standard C++ separates Standard Conversions from overload resolution).	duplicate of #130	Yes	
196	30-Sep-04	meeting #7 (WA)		Technical	M	Brandon Bray	In native types, % behaves like &.	This is correctly specified in 13.1.3; closed on August 4, 2005 conference call.	Yes	
197	30-Sep-04	meeting #7 (WA)	19.1	Technical		Herb Sutter	Should generic member functions be allowed in native classes? This feature appeared in the draft as an "editorial" addition. Does MS really intend to implement this feature? Yes, MS did		Yes	
198	30-Sep-04	meeting #7 (WA)	2	Technical		Herb Sutter	Propose wording to require that extensions over and above ISO C++ requirements, be diagnosed.	Meeting 9 (NJ): Re the new paragraph added to §2. "Conformance" in response to spreadsheet issue #198, the committee believed this text does not adequately address the issue. The editor was asked to remove it. Ownership was transferred from Tom to Herb. Closed on Aug 4, 2005 conference call	Yes	
199	30-Sep-04	meeting #7 (WA)	16.2.1	Technical	R	Brandon Bray	Proof the text on Collection type and how a for each is executed.		Yes	
200		meeting #7 (WA)	19.1	Technical		Herb Sutter	Regarding "Member functions in a native class can be generic", support for this appears to have been added inadvertently. However, is there any user need for it?	Since the MS product was going to support this anyway, Steve A. agreed to have it in the std.	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
201	23-Oct-04	meeting #8 (WA)		Technical	H	Brandon Bray	How to accomodate non-CLI calling conventions on other platforms. Meeting #8 (WA): delegate void D(int); generic<class T> void F(T t) { System::Console::WriteLine(t->ToString()); } typedef void (* FP)(int); void G(FP fp) { D^ d = gcnew D(fp); d(1010); } int main() { D^ d = gcnew D(&F<int>); d(42); FP fp = &F<int>; fp(101); G(&F<int>); In MS's implementation, need to use __clrcall to indicate the clr calling convention. This lead to a discussion of how to accomodate non-CLI calling conventions on other platforms. It was noted that the CLI draft spec, Partition II, 15.3, "Calling convention", states: "When dealing with methods implemented outside the CLI it is important to be able to specify the calling convention required. For this reason there are 16 possible overloads of the calling kind. Two are used		No	Yes
202	23-Oct-04	meeting #8 (WA)		Technical	H	Brandon Bray	Name lookup in managed classes ignores interfaces.		Yes	
203	26-Oct-04	Rex Jaeschke	10.1.2	Technical	M	Brandon Bray	[Note: The compiler needs to add typedef members to the class so that template code can use the return type or the parameter types. [[Need more explanation.]] end note]	Phone call Aug 18, 2005: Resolved. The compiler does not do this, so the text that suggested it happens was removed.	Yes	
204	26-Oct-04	Rex Jaeschke	12.2.2	Technical	M	Brandon Bray	Write intro text.	Phone call Aug 18, 2005: Resolved. Text written.	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
205	26-Oct-04	Rex Jaeschke	15.5	Technical	H	Brandon Bray	15.5 Explicit type conversion (cast notation) The rules in the C++ Standard (§5.4/5) have been extended for C++/CLI by including safe casts before static casts. <ul style="list-style-type: none">• a const_cast• a safe_cast• a safe_cast followed by a const_cast• a static_cast• a static_cast followed by a const_cast• a reinterpret_cast• a reinterpret_cast followed by a const_cast [Note: Standard C++ programs remain unchanged by this, as safe casts are ill-formed when either the expression type or target type is a native class. end note] Provide background on the expected behavior and rationale. (Get this from the updated casting proposal.)	Resolved on July 7, 2005 conference call. Incorporated into 1.13.	Yes	
206	26-Oct-04	Rex Jaeschke	21.4	Technical	M	Brandon Bray	Simple value classes: Flesh this out.	Resolved in the 1.14b draft circulated for the August 4, 2005 conference call.	Yes	
207	26-Oct-04	Rex Jaeschke	24.2.5	Technical	H	Brandon Bray	Interface member access: Write up. Attribute specification: Write up net modules. Should safe_cast allow casting to void?	Resolved on July 21, 2005 conference call. Incorporated into 1.14a.	Yes	
208	26-Oct-04	Rex Jaeschke	27.2	Technical	L	Brandon Bray		Meeting #9 (NJ): Close without action. The standard will not mention net modules.	Yes	
209	24-Nov-04		15.3.13	Technical	L	Brandon Bray		Meeting #9 (NJ): This is allowed.	Yes	
210	4-Dec-04	Rex Jaeschke	29.5.1	Technical	M	Brandon Bray	There is confusion about DefaultMember attribute and IndexerNameAttribute. In the current implementation, it appears that the first one is exhibiting the behavior of the second one, and the second one is being emitted into metadat directly when it should be consumed by the compiler.	Phone call Aug 18, 2005: Resolved. Added the ability to use IndexerName.	Yes	
211	4-Dec-04	Rex Jaeschke	17.1	Technical	L	Brandon Bray	The namespace cli is reserved. However, what if the compiler imports an assembly created by C#, for example, containing a user-defined namespace cli having a type T, or a user-defined type called cli defined at the global namespace level and having a type T. Both of these appear to C++/CLI as the same names, namely ::cli::T? (BTW, this works with the current implementation.)		Yes	
212	4-Dec-04	Rex Jaeschke		Technical	M	Brandon Bray	Since static constructors are emitted in metadata as protected members, TG5 required that they be defined as protected, rather than the previous treatment, which allowed the programmer to give them any accessibility, but that was ignored by the compiler. (The same situation occurs with a finalizer and a destructor for a ref class.) Now that an interface is allowed to have a static constructor, we have no way to explicitly declare that member to be protected; all members in an interface are implicitly public. What to do?	Meeting 9 (NJ): Leave as is; that is, require a diagnostic if the accessibility specified contradicts what is required. Make sure this applies to destructors and finalizers as well.	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
213	4-Dec-04	Rex Jaeschke		Technical	M	Brandon Bray	13.3.3.2/4 of the C++ Standard has rules for pointer conversions, that need to be adapted to handles. Review this subclause and determine the changes needed for the C++/CLI spec.	Phone call Aug 18, 2005: Resolved. Added 14.2.1.1 to rank handle conversions.	Yes	
214	4-Dec-04	Rex Jaeschke		Technical		Editor	Representation of false and nullptr. After changes made earlier this year by TC39/TG3, the definition of System::Boolean requires that an instance of that type be 8 bits, that false be all-bits-zero, and that true have any one or more bits set. However, some months ago, TG5 agreed to NOT require that C++/CLI's bool type map to System::Boolean. As such, the representation of true and false is now unspecified. Consider a value class that contains a bool member. Being a value class it can't have a default constructor; instead, instances are born with the guaranteed default value all-bits-zero. However, without having any guarantee about the representation of true and false, we are not guaranteed what, if anything, that default value means. I believe it would be most useful for C++/CLI to require that false be all-bits-zero, and that true have any one or more (unspecified) bits set. (Note that TG3 and TG2 have a similar issue with System::Decimal, which is a 128-bit value class. As it happens, while all-bits-zero represents value zero in both the MS and IEEE 754r decimal representations, the	Implemented in WD1.10.	Yes	
215	Feb-3-2005	Jeff Peil	13.1.1, 13.1.3	Technical	2	Brandon Bray	§13.1.1 and 13.1.3 disagree, one describes gc-lvalue->lvalue as a conversion for native types, the other describes it as never having gc-lvalues for these (they are always l-values) They need to be made consistent.	Resolved in HI; incorporated into 1.11	Yes	
216	Feb-3-2005	Jeff Peil	18.5	Technical		Editor	Shouldn't DllImport be allowed on static member functions in ref/value classes?	Done in WD1.10.	Yes	
217	Feb-3-2005	Sean Perry	29.1.1	Technical		Editor	How do attributes work with derived classes. If I declare class B and D, which derives from B, and apply attribute X to both of them. What happens for the various values of AllowMultiple & Inherited? (pg. 154, line 35) We need to talk about inheritance. What happens to the attributes of the base class when we provide attributes on the derived class?	Editor posted a response to the liaison reflector on Feb 15, and made several small improvements to WD1.10.	Yes	
218	Feb-10-2005	Rex Jaeschke	15.3	Technical	M	Brandon Bray	Are the productions postfix-expression . pseudo-finalizer-name postfix-expression -> pseudo-finalizer-name necessary, and, if so, should the "pseudo-" prefix be dropped?	Resolved on July 21, 2005 conference call. Incorporated into 1.14a. Pseudo-finalizer is not referenced in the document any more.	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
219	3-Mar-05	Brandon Bray		Technical	M	Brandon Bray	Currently, the Visual C++ compiler allows a friend to first declare a generic type. Whether the language specification says this is allowed is up for discussion. Are there any issues we should consider before saying that it should be supported?	Phone call Aug 18, 2005: Resolved. Added 20.5 to cover friends for native classes.	Yes	
220	7-Mar-05	Sean Perry		Technical	M	Brandon Bray	Destroying members should happen after the base class stuff in the fault block. TG5 also brought up the destructor order (which Jeff brought up last week). From Herb: Constructor failures. We need to tweak the IL we generate for constructors to have smoother handling of constructor exceptions and deep virtual calls in constructors. Here's what we need to do: .ctor { bool baseIsConstructed = false; try { construct all our own directly held members call our base class's constructor baseIsConstructed = true; run our own constructor body } fault { destroy all our own directly held members (if non-null) if(baseIsConstructed) call our base class's destructor (same as when chaining from Dispose(true)) } }	Revised on July 7, 2005 conference call. Incorporated into 1.13.	Yes	
221	28-Apr-05	Jonathan Caves	15.3.3	Technical	M	Brandon Bray	Add an example	Resolved in the 1.14b draft circulated for the August 4, 2005 conference call.	Yes	
222	28-Apr-05	Jonathan Caves	15.4.5	Technical	M	Brandon Bray	I think that some text needs to be added specify that with a generic parameter dynamic_cast is used to convert IDisposable and if this conversion fails then no further action in taken. The test sort of says this but not explicitly enough.	Consider adding the new text in the generics clause. Phone call Aug 18, 2005: Resolved. Added the phrase "dynamic cast" to 15.4.5 which covers the delete expression for generic type parameters.	Yes	
223	28-Apr-05	Jonathan Caves	15.21	Technical	M	Brandon Bray	Add an example which shows what happens if a property returns an array, or if it returns a reference and it is used in a 'set' context but it does not have set method.	Resolved in the 1.14b draft circulated for the August 4, 2005 conference call.	Yes	

	A	B	C	D	E	F	G	H	I	J
1	Date Raised?	Issue Raiser?	Reference	Issue Type	Priority	Owner	Comment	Other Remarks	Resolved?	Postponed?
224	28-Apr-05	Jonathan Caves	19.7	Technical	M	Brandon Bray	Revise this to accommodate -> as a static operator. Also revise 19.7.2.	Resolved in the 1.14b draft circulated for the August 4, 2005 conference call.	Yes	

This is a replacement/place-holder for Documents TC39-TG5/2005/016, 019, 021, and 024. Documents 016, 019, and 021 were intermediate committee drafts of the specification, and are not included here. They are superseded by document 024, which can be found at the following URLs:

<http://www.plumhall.com/ecma/index.html>

<http://go.microsoft.com/fwlink/?LinkId=50042>