Doc No: SC22/WG21/N1761
J16/05-0021
Date: Feb 1, 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 #8

Meeting #9 of Ecma TC39/TG5 (C++/CLI) was held in Westfield, NY, USA, on January 20–21, 2005.

The following TG5 documents are attached to this liaison report:

- TC39-TG5/2005/001 Venue Information for the 9th meeting of TC39-TG5, Westfield, USA, January 2005
- TC39-TG5/2005/002 **Intentionally omitted (see below)**
- TC39-TG5/2005/003 C++/CLI Specification Comments - revision 2 January 2005
- TC39-TG5/2005/004 Project Editor's Report, January 2005
- TC39-TG5/2005/005 Agenda for the 9th meeting of TC39-TG5, Westfield, USA, January 2005
- TC39-TG5/2005/006 C++/CLI Specification Comments - revision 23 January 2005
- TC39-TG5/2005/007 Minutes of the 9th meeting of TC39-TG5, Westfield, NJ, January 2005

Document TC39-TG5/2005/002, “Working Draft 1.9 of the C++/CLI Standard, Language” is not included. This draft can be found at the following URL: www.plumhall.com/tc39-tg5-2005-002.html. Note that this is a preliminary draft in that it has not been approved by TG5.

Venue Information

for the:	***9th meeting of Ecma TC39-TG5***
to be held in:	***Westfield, NJ, USA***
on:	***20-21 January 2005***

TIME :

09:00 till 17:00 on Thu 20th January 2005
09:00 till 17:00 on Fri 21st January 2005
[8:30 AM Breakfast, Noon Lunch each day]

LOCATION :

Best Western Westfield Inn
435 North Avenue West
Westfield, NJ 07043 USA
Phone : 800-688-7474
(Directions: see below)

CONTACT :

J. Stephen Adamczyk

jsa@edg.com

The January meeting of TG5 will be held at the Best Western Westfield Inn in Westfield, NJ. To make a reservation, call the hotel directly at 800-688-7474 and ask for the TG5 block. You should get a rate of $109 per night. Here's the Expedia link for the hotel, which gives you general information, pictures, and directions: *http://www.expedia.com/pub/agent.dll/qscr=dspv/itty=new/from=m/htid=20180/nojs=1/rfrr=20906/eapi=23708*
(However, don't use Expedia to make your reservation; contact the hotel directly so you can be counted against our allocation.)

The nearest airport is Newark Liberty International, airport code EWR. ("Liberty" is a post-9/11 addition; it was formerly just "Newark International"). A taxi from the airport to the hotel is around $40 plus tip, but you may find that the cab driver can't find the exact address of the hotel without help from you. The hotel is very near the train station, if that helps. If you'd prefer to have a car meet you, a good car service is Airbrook Limousine, 800-800-1990. They'll charge you roughly $70 to meet you at the airport and drive you to the hotel, and they will handle the exact address just fine.

You could also rent a car. However, while in general a car is highly desirable in New Jersey, it's not necessary in Westfield. The town is highly walkable and there are many excellent restaurants within a short walk from the hotel. If you do bring a car, there is parking available at the hotel.

The hotel has Wi-Fi Internet access that covers both the guest rooms and the meeting rooms. We don't plan to provide wired access unless someone requests it.

We will provide breakfast and lunch both days, and dinner on Thursday night. The latter we expect will be in the restaurant that is adjacent to the Hotel. It's a first-rate French restaurant called "Chez Catherine".

This is a replacement/place-holder for Document TC39-TG5/2005/002, “Working Draft 1.9 of the C++/CLI Standard, Language”. This draft can be found at the following URL: www.plumhall.com/tc39-tg5-2005-002.html. Note that this is a preliminary draft in that it has not been approved by TG5.

	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): Withrawn 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.		No	
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 (possibily 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	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		No	
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	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 array<element-type>.		No	
24	16-Dec-03	Phone meeting	9	Technical	R	Brandon Bray	Review this clause.		No	
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 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_ptrs.	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.		No	
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.		No	
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).		No	
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 gcnew 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.	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.	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		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)		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.		No	
55	16-Dec-03	Phone meeting	18.6	Editorial	R	Brandon Bray	Review this subclause.		No	
56	16-Dec-03	Phone meeting	18.7.4	Technical	M	Brandon Bray	Identify when (operator) synthesis would and would not occur.		No	
57	16-Dec-03	Phone meeting	18.6.5.1	Technical	L	Brandon Bray	Writeup op_true and op_false operators		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	
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 initonly 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.		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.		No	
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 distingish 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.		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	
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"?		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?**
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.		No	
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).		No	
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 MethodImplOption metadata generation.	The editor has added quite a bit of text re this attribute. See if that is sufficient.	No	
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.		No	
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	

92	29-Jan-04	meeting #2 (HI)		Technical	M	Brandon Bray	"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)}.)
							
							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:
							
							value class Size { /*…*/ };
							
							value class Color { /*…*/ };
							
							ref class X {
							public:
							property Size Size;
							property Color Color;
							};
							
							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).
							
							Here's a brief description of the problem:
							
							ref class X {
							public:
							property Size Size {
								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	Do we require exact mapping for types, or do we allow a certain amount of flexibility? 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. 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. 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.	Meeting #3 (Mel): There was a lengthy discussion. No resolution. Meeting #4 (NJ): There was a lengthy discussion. Meeting #5 (WA): There was another lengthy discussion, which resulted in Plum's notes being incorporated into the meeting minutes. The edits from Plum's subsequent paper were incorporated into WD1.6 for Meeting #6 (WA).	Yes	
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	
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.	No	
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	
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>>.	Meeting #3 (Mel): 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.	No	
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		No	
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 MethodImplOption 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	
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	

	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?**
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	Indexed properties -- Consider the following:
							
							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 complier 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 it the resulting call is ambiguous then issue an error.
								Meeting #5 (WA): Discussed this. Option #3 preferred.
								
								Meeting 7 (WA): Discussed this in detail.
								
								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.
									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 array<ItemType>^ with an argument of type int. 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.]
									Yes	

126	14-Jun-04	meeting #5 (WA)	12.1	Technical		Editor	The type long long will be defined by pointing to	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 the modreq IsBoxed shall be emitted (i.e., passing		No	
128	14-Jun-04	meeting #5 (WA)	12.3.6	Technical	L	Brandon Bray	The compiler will need to emit a modopt to distinguish interior_ptr<T> from tracking reference to T (		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 modopt IsPinned shall be emitted (i.e.,		Yes	
130	14-Jun-04	meeting #5 (WA)	14.1.1	Technical	L	Brandon Bray	Separate the list of conversions from the order of preference (such as how Standard C++ separates Sta		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?**
131	14-Jun-04	meeting #5 (WA)	15.3.3	Technical	L	Brandon Bray	Add text to indicate the circumstances under which the following type modifiers shall be emitted, and point to each modifier's definition: • IsBoxed i.e., passing a handle to a value type). • IsByValue (i.e., ref class type passed by value). • IsConst (i.e., pointer or reference to a const-qualified type). • IsExplicitlyDereferenced (i.e., interior_ptr as a parameter). • IsImplicitlyDereferenced (i.e., parameter is a reference). • IsLong (i.e., long/unsigned long/long double parameters). • IsExplicitlyDereferenced (i.e., pin_ptr as a parameter). • IsSignUnspecifiedByte (i.e., plain char's sigedness). • IsUdtReturn (i.e., ref class type returned by value). • IsVolatile (i.e., pointer or reference to a volatile-qualified type).		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	
133	14-Jun-04	meeting #5 (WA)	15.3.10	Technical	L	Brandon Bray	The null value is converted to the null value of the destination type. This can be unverifiable and migh		No	
134	14-Jun-04	meeting #5 (WA)	16.3.3	Technical	M	Brandon Bray	Need to add text to indicate the circumstances under which the modreq IsUdtReturn shall be emitted (		No	
135	14-Jun-04	meeting #5 (WA)	18	Technical	R	Brandon Bray	This table and corresponding sections should include Special Member Functions (SMFs) like destruct		No	
136	14-Jun-04	meeting #5 (WA)	18.2.1	Technical	L	Brandon Bray	Need to address the following: C++/CLI uses the System::Reflection::DefaultMemberAttribute attribu		No	
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	L	Brandon Bray	We probably should say something about the reserved names get_Item and set_Item, and their relation		No	
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 cla		No	
143	14-Jun-04	meeting #5 (WA)	18.6	Technical	L	Brandon Bray	The restriction below does not apply to non-static member operators – that need not have a parameter		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?**
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	L	Brandon Bray	Provide C++ names for operator True and False	Meeting #8 (WA): Move to future directions.	No	
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	L	Brandon Bray	Add text to indicate the circumstances under which the following type modifiers shall be emitted, and point to each modifier's definition: • IsConst (i.e., data member involves a cv type). • IsImplicitlyDereferenced (i.e., has a reference type). • IsLong (i.e., long/unsigned long/long double type). • IsSignUnspecifiedByte (i.e., plain char's sigedness). • IsVolatile (i.e., data member involves a cv type).		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?**
149	14-Jun-04	meeting #5 (WA)	20.3	Technical	L	Brandon Bray	Add text to indicate the circumstances under which the following type modifiers shall be emitted, and point to each modifier's definition: • IsBoxed i.e., passing a handle to a value type). • IsByValue (i.e., ref class type passed by value). • IsConst (i.e., pointer or reference to a const-qualified type). • IsExplicitlyDereferenced (i.e., interior_ptr as a parameter). • IsImplicitlyDereferenced (i.e., parameter is a reference). • IsLong (i.e., long/unsigned long/long double parameters). • IsExplicitlyDereferenced (i.e., pin_ptr as a parameter). • IsSignUnspecifiedByte (i.e., plain char's signedness). • IsUdtReturn (i.e., ref class type returned by value). • IsVolatile (i.e., pointer or reference to a volatile-qualified type).		No	
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		Brandon Bray	The text indicates that a generic-declaration may appear in a class scope, but the syntax of member-de		Yes	
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		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?		No	
156	14-Jun-04	meeting #5 (WA)	30.1	Technical		Editor	Can generic types be nested in native classes?		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?**
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		No	
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?		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 in		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.		No	
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 fu		Yes	
165	14-Jun-04	meeting #5 (WA)	30.3	Technical	L	Brandon Bray	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.	No	
166	14-Jun-04	meeting #5 (WA)	30.3	Technical		Editor	What, if anything, does it mean for a generic func	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?		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.		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.		No	
171	14-Jun-04	meeting #5 (WA)	30.4.2	Technical	L	Brandon Bray	When are members considered hidden? Is it using the rules described later? Those are described as a		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?**
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".	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?]].		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-typ		No	
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 valu		No	
176	14-Jun-04	meeting #5 (WA)	E	Technical	R	Brandon Bray	Flesh out Future Directions		No	
177	14-Jun-04	meeting #5 (WA)	E.7	Technical		Brandon Bray	Add text to show the behavior in the CLI (includi	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. Meeting #8 (WA). Decided to drop it.	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>(*pa) = *pin_ptr<T>(*pb);
									No	

185	2-Aug-04	meeting #6 (WA)		Technical		Herb Sutter	Collapsing reference to reference. (It's in the C++0x spec.)		No	
186	2-Aug-04	meeting #6 (WA)		Technical	M	Brandon Bray	Should we standardize traits?		No	
187	2-Aug-04	meeting #6 (WA)		Technical		Brandon Bray	user-defined assignment operator for handles	dupelicate 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		??	Look at the changes to the grammar for C++0x and note where they affect the C++/CLI grammar.		No	
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 accessibility vs. visibility		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?**
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		No	
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.	No	
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).		No	
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		Tom Plum	Propose wording to require that extensions over and above ISO C++ requirements, be diagnosed.		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.		No	
200	30-Sep-04	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?		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?**

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-G193CLI 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
									No	

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. • 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.		No	
208	26-Oct-04	Rex Jaeschke	27.2	Technical	L	Brandon Bray	Attribute specification: Write up modules.		No	
209	24-Nov-04		15.3.13	Technical	L	Brandon Bray	Should safe_cast allow casting to void?		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	
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?		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?**
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		No	

2005-01 Project Editor's Report

Rex Jaeschke

Ecma TC39-TG5 project editor

rex@RexJaeschke.com

+1 703 860-0091

Working Draft 1.9 has been produced and distributed. The following work went into producing it:

1. I applied corrections resulting from the Redmond Oct meeting.
2. To make it easier to distinguish source code from metadata when the two are interspersed, the metadata now has a 15% grey background.
3. I've added "Microsoft-specific text" delimiters for two reasons: 1) As well as being the standard for C++/CLI, this spec also serves as MS's product documentation. (When the final standard is produced, the MS-specific text will be removed, as is done with the TG3 spec.) 2) The spec currently contains some text that isn't needed by a conforming implementation, but provides useful information.

   An example of the latter is the modopt IsLong (§33.1.5.6). As long int is not required to map to int (or long double to double), this modopt isn't part of the standard. However, for implementations where this mapping is used, it seems useful to give guidance as to the modopt name an extended implementation might use. If we agree, then this should be marked "Implementer guidance" rather than "Microsoft-specific".
4. I revised the static operator subclause, with the metadata-related text being moved to the metadata clause. I added a subclause on non-static operators.
5. I rewrote §14.8, "[Conversion] Naming conventions", and the metadata-related text was moved to the metadata clause.
6. I applied the changes to the long long specification per the WG21 meeting and Steve's Revision 2 paper, N1735.
7. I reorganized the "Future Directions" clause to match the major clauses of the standard.
8. I incorporated the 29 pages of edits from Microsoft, posted to the email reflector on Nov 23.
9. I added the annex on Documentation Comments.
10. I incorporated Steve's paper on extended integer types.

Issues needing resolution

1. The proposal to require that false and nullptr be represented as all-bits-zero. See email from Rex on 11/23 titled "Representation of bool" and on 12/8 titled "RE: Representation of bool (and nullptr)".
2. I replaced 15.3.1, "Subscripting" and 15.3.2, "Indexed access" with one subclause per Steve's email of 12/28. There is the question of indexed accessors in value and interface classes, and allowing [] on both classes and handles to classes.
3. I need precise text for the conversions needed to make System::Boolean bind closer to bool than any other C++ type.

Agenda for the: 9th meeting of Ecma TC39-TG5
to be held in: Westfield, NJ, USA
on: 20-21 January 2005

TIME: 09:00 till 17:00 on Thu 20th January 2005
09:00 till 17:00 on Fri 21st January 2005
[8:30AM Breakfast, Noon lunch each day]

LOCATION: Best Western Westfield Inn
435 North Avenue West
Westfield, NJ 07043 USA
Phone: 800-688-7474
(Directions: see TG5/2005/001)

CONTACT: J. Stephen Adamczyk, EDG
jsa@edg.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 2004/046)

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

7.1 March 7, 8, and 11 (9AM), Kona, HI; hosted by Plum Hall (Note schedule change to Mon Mar 7 and Tue Mar 8 approved by Joel Marcey email 11/17/2004).

NOTE

TC39 business meeting takes place March 11(PM)

	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): Withrawn 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 (possibily 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	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	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 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.		No	
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_ptrs.	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.		No	
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 gcnew 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.	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.	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.	No	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.	No	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 op_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 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 initonly 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.		No	
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 distingish 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.		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	
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).		No	
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 MethodImplOption 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	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	

92	29-Jan-04	meeting #2 (HI)		Technical	M	Brandon Bray	"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)}.)
							
							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:
							
							value class Size { /*…*/ };
							
							value class Color { /*…*/ };
							
							ref class X {
							public:
							property Size Size;
							property Color Color;
							};
							
							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).
							
							Here's a brief description of the problem:
							
							ref class X {
							public:
							property Size Size {
								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	Do we require exact mapping for types, or do we allow a certain amount of flexibility? 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. 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. 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.	Meeting #3 (Mel): There was a lengthy discussion. No resolution. Meeting #4 (NJ): There was a lengthy discussion. Meeting #5 (WA): There was another lengthy discussion, which resulted in Plum's notes being incorporated into the meeting minutes. The edits from Plum's subsequent paper were incorporated into WD1.6 for Meeting #6 (WA).	Yes	
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	
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.	No	
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	
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. Meeting #9 (NJ): Herb presented an update on the latest thinking within MS w.r.t destructors and finalizers. This involved the use of the patterns Dispose() and Dispose(bool).	Yes	
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.	No	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	

	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		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	
106	29-Jan-04	meeting #2 (HI)		Technical		Editor	Should >> handled as two tokens rather than one; e.g., List<List<int>>.	Meeting #3 (Mel): 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.	No	
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 MethodImplOption 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	

	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?**
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	
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	Indexed properties -- Consider the following:
							
							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 complier 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 it the resulting call is ambiguous then issue an error.
								Meeting #5 (WA): Discussed this. Option #3 preferred.
								
								Meeting 7 (WA): Discussed this in detail.
								
								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.
									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 array<ItemType>^ with an argument of type int. 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.]
									Yes	

126	14-Jun-04	meeting #5 (WA)	12.1	Technical		Editor	The type long long will be defined by pointing to the paper WG21 N1565. How to do this normatively?	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 the modreq IsBoxed shall be emitted (i.e., passing a handle to a value type). Point to that modreq's spec.	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 modopt to distinguish interior_ptr<T> from tracking reference to T (T%) in the metatada.[[#28]] Need to add text to indicate the circumstances under which the modopt IsExplicitlyDereferenced shall be emitted (i.e., interior_ptr as a parameter). Point to that modopt's spec.		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 modopt IsPinned shall be emitted (i.e., pin_ptr as a parameter). Point to that modopt's spec.		Yes	
130	14-Jun-04	meeting #5 (WA)	14.1.1	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).	Meeting #9 (NJ): Close without action.	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?**
131	14-Jun-04	meeting #5 (WA)	15.3.3	Technical		Editor	Add text to indicate the circumstances under which the following type modifiers shall be emitted, and point to each modifier's definition: • IsBoxed i.e., passing a handle to a value type). • IsByValue (i.e., ref class type passed by value). • IsConst (i.e., pointer or reference to a const-qualified type). • IsExplicitlyDereferenced (i.e., interior_ptr as a parameter). • IsImplicitlyDereferenced (i.e., parameter is a reference). • IsLong (i.e., long/unsigned long/long double parameters). • IsExplicitlyDereferenced (i.e., pin_ptr as a parameter). • IsSignUnspecifiedByte (i.e., plain char's sigedness). • IsUdtReturn (i.e., ref class type returned by value). • IsVolatile (i.e., pointer or reference to a volatile-qualified type).		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	
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	Brandon Bray	Need to add text to indicate the circumstances under which the modreq IsUdtReturn shall be emitted (i.e., ref class type retruned by value). Point to that modreq's spec.		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?**
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, copy constructors, default constructors, assignment operators, conversion to special bool, handle equality. Many of these are not supported for value classes.		Yes	
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.	No	
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.	No	
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 reviewed.		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 these terms (and change the style accordingly) or just call them the add function, remove function, and raise function.		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	

	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?**
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.	No	
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 the following type modifiers shall be emitted, and point to each modifier's definition: • IsConst (i.e., data member involves a cv type). • IsImplicitlyDereferenced (i.e., has a reference type). • IsLong (i.e., long/unsigned long/long double type). • IsSignUnspecifiedByte (i.e., plain char's sigedness). • IsVolatile (i.e., data member involves a cv type).	Meeting #9 (NJ): Needs to be done.	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?**
149	14-Jun-04	meeting #5 (WA)	20.3	Technical		Editor	Add text to indicate the circumstances under which the following type modifiers shall be emitted, and point to each modifier's definition: • IsBoxed i.e., passing a handle to a value type). • IsByValue (i.e., ref class type passed by value). • IsConst (i.e., pointer or reference to a const-qualified type). • IsExplicitlyDereferenced (i.e., interior_ptr as a parameter). • IsImplicitlyDereferenced (i.e., parameter is a reference). • IsLong (i.e., long/unsigned long/long double parameters). • IsExplicitlyDereferenced (i.e., pin_ptr as a parameter). • IsSignUnspecifiedByte (i.e., plain char's signedness). • IsUdtReturn (i.e., ref class type returned by value). • IsVolatile (i.e., pointer or reference to a volatile-qualified type).		No	
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		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]]		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?**
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.		Yes	
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?		Yes	
156	14-Jun-04	meeting #5 (WA)	30.1	Technical		Editor	Can generic types be nested in native classes?		No	
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)		Yes	
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.		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?**
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.		Yes	
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 of generic classes, for that matter) have exception specifications? (No)		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."	No	
166	14-Jun-04	meeting #5 (WA)	30.3	Technical		Editor	What, if anything, does it mean for a generic function to be static/extern or inline?	Meeting #6 (WA): all have the usual meaning.	Yes	
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?**

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>);
									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.	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?**
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".	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	
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.	No	
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 CIL).	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. Meeting #8 (WA). Decided to drop it.	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>(*pa) = *pin_ptr<T>(*pb);
									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	L	Brandon Bray	Look at the changes to the grammar for C++0x and note where they affect the C++/CLI grammar.		No	
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 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.	No	
194	2-Aug-04	Anthony Williams	15.3.2	Technical		Jonathan Caves	Re Anthony's post to the reflector re "default indexed properties" and operator[], will post a response to the reflector and will provide some replacement words for 15.3.2, especially re the synthesizing of the operator.	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		Editor	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 §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.	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	30-Sep-04	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?		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?**

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
									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. • 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.		No	
208	26-Oct-04	Rex Jaeschke	27.2	Technical	L	Brandon Bray	Attribute specification: Write up net modules.	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	Should safe_cast allow casting to void?	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.		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		Editor	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.	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?**
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		No	
215	21-Jan-05	Herb Sutter		Technical		Herb Sutter	Should a const reference bind cause a temporary to be generated when unboxing?	Meeting 9 (NJ): Internal MS email thread was discussed. Herb will deal with this off-line with Steve.	No	

Minutes of the: *9th meeting of Ecma TC39-TG5*
held in: *Westfield, NJ, USA*
on: *20-21 January, 2005*

Rex Jaeschke
rex@RexJaeschke.com
2005-01-23

1 Opening

Convener Tom Plum welcomed everyone to the ninth meeting of TG5.

1.1 Appointment of Recording Secretary

Rex Jaeschke was appointed.

1.2 Introduction of participants

The participants introduced themselves. Those attending were: Steve Adamczyk (EDG), Brandon Bray (Microsoft), Rex Jaeschke (Microsoft), P.J. Plauger (Dinkumware), Tana Plauger (Dinkumware), Tom Plum (Plum Hall), John Spicer (EDG), Herb Sutter (Microsoft), and Christopher Walker (Dinkumware).

1.3 Host facilities/local information

Local information was provided.

2 Adoption of the agenda

Document 2005-05 was approved without objection.

3 Approval of Minutes of previous TG5 meeting

Document 2004-46 was approved without objection.

4 Matters arising from the minutes not covered elsewhere

None.

5 Project Editor's Report – Rex Jaeschke

Rex presented document 2005-04.

a. The proposal to require that false and nullptr be represented as all-bits-zero. See email from Rex on 11/23 titled "Representation of bool" and on 12/8 titled "RE: Representation of bool (and nullptr)".

   It was agreed that the representation of false and nullptr (the default value of a handle) shall be all-bits-zero.

   Action: Editor will apply these changes.

b. Editor replaced 15.3.1, “Subscripting” and 15.3.2, “Indexed access” with one subclause per Steve’s email of 12/28. There is the question of indexed accessors in value and interface classes, and allowing [] on both classes and handles to classes.

   It was agreed that indexed accessors are supported by all CLI class types.

   Action: Editor will apply these changes.

c. Editor needs precise text for the conversions needed to make System::Boolean bind closer to bool than any other C++ type. Sean provided this in email on 1/10.

   Action: Editor will apply these changes.

6 Approving tracked changes in latest draft

Given the large number and size of changes, it was agreed that while we’d deal with all issues raised w.r.t WD1.9, we would not formally review and approve all tracked changes. Rather, the review of the remaining changes would be folded into the review of the whole draft that will occur between the January and March meetings.

We reiterated our desire to vote out the standard at the March 2005 meeting. To that end, Tom proposed a review plan for achieving that goal.

Action: The clauses of the draft were assigned to the following groups for detailed review of both the existing text in WD1.9 and the new text to be distributed before the March meeting:

Reviewer(s)	Clauses
Dinkumware	20–28,32
EDG	14,15,30,31
Microsoft team #1 (Mark)	1-13,16-18
Microsoft team #2 (Brandon)	19,34
Plum Hall/IBM	29, 33, A–H

All technical proposals are to be posted to the liaison reflector (e-TC39-TG5-liaison@ecma-international.org) as soon as possible, and definitely before the March meeting. Each week, issues that need to be resolved can be addressed during the phone conferences (see §7.1 below). All editorial corrections should be sent directly to the editor.

During our discussion of WD1.9 (2005-02), the issues raised included the following:

a. Fundamental types are Standard C++ types; they are not value class types. However, they do each have a corresponding value class type. For example, in the expression i.ToString() (where i is an int), in this context, i is treated as its corresponding value class type.

   Action: Editor will review all places in which the spec says or suggests that a fundamental type is a value class type.

b. The term “byref” is used in several places, but this term is not defined in either the C++/CLI or CLI specifications. All occurrences of “byref” should be changed to “managed pointer”.

c. 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 text will be removed.

   Action: Herb will take over ownership of this issue from Tom.

d. §4, “Definitions”: Is the term “Object” necessary? After some discussion, it was decided that we should drop this term:

   I. If we mean “System::Object”, spell it that way.

   II. If we mean “an object on the CLI heap”, we should say that.

e. §4, “Definitions”, type, class, interface; ref and value: Change “binds to” to “is”.

f. §4, “Definitions”, type, class, any: … native [class] type.

g. §4, “Definitions”, type, value, simple: Change “CLI” to “native”.

h. §6, “Acronyms and abbreviations”: Move final three entries to the top of the list, and in front of the rest, add text that points to the ISO/IEC CLI standard.

i. §8.2.5, “Pointers, handles, and null”: re gc-lvalue, say that “it might be on the CLI heap” rather than “is on the CLI heap”.

j. §8.7, “Delegates”: Non-member functions are treated by the CLI as really being members of a class called “<Module>”. As such, a delegate can encapsulate a non-member function; however, that parent class name is not used. Instead, &function-name is all that is needed when initializing a delegate. Also make appropriate changes in the normative clause for delegates. Improve the wording w.r.t “is the address”.

k. §10.1: Add a 3rd bullet to the list w.r.t signatures with unknown modopts. See Brandon for details.

l. §12.3.3, final paragraph. What’s the purpose of this? After some discussion, it was agreed to remove this.

m. §12.3.5, Penultimate paragraph: Correct usage of V by introducing v as an instance of V.

n. §12.3.7.1: Remove new sentence.

o. §13.1.2, final bullet of three bullets: Remove this and merge change into an existing bullet in the C++ Standard. In 5.3.1/1, include “handle to T” text as well.

Response to Sean’s email dated 1/20 (page/line references pertain to the PDF version of the draft):

p. §9.1.4: Is "List<item>>5" allowed? The current wording doesn't seem to address this. The words say that the >> seen after a < for a template are treated as two closing > for templates. It might be better to say the >> is treated as two separate > tokens (first one is always a closing >).

   See Daveed's proposal re this (emailed 1/17/2005). If that doesn't address your concerns, please say so.

q. §10.1: the section on #using is thin. I need to read it more to comment on what is there. It should include information on how to map constructs into C++.

   We agree that this subclause needs more work. Please give us specific suggestions for what you think is needed there.

r. §10.1: Is #using a preprocessing directive? I don't think so. We should state one way or another.

   Correct, it is not. It is treated like #pragma.

s. §10.1 pg 55, line 5: Uses names that can't be written in C++ as an example of when to use another language. However __identifier() should nullify that reason.

   The issue in line 5 is not that they have names that can't be spelled in C++/CLI, but that these functions are not permitted to be written in C++/CLI, period. So, __identifier is not a work around.

t. §10.4, pg 43, line 35: This paragraph states accessibility is considered for overload resolution. This seems new. I can see visibility being considered, but not accessibility.

   Accessibility is *not* considered for overload resolution, but it is considered for name lookup. This is new, and it's intentional.

u. §12.0: The table for types at the beginning of section 12 in PDF is difficult to understand. It needs some borders to define the relationships.

   Agreed.

v. §14.2.1, line 35: Should the words "or handle" be removed since that is handled later on? I thought handles can't be converted to bool and that we extended the condition expression locations to accept a handle.

Agreed; in fact, neither of the two changes in this extract from the C++ Standard is correct. As such, this extract and related text will be removed.

7 Date and place of next meetings

7.1 Next Meeting

The editor will close the next draft, WD1.10, at 8 PM EST on Friday, February 25.

March 7-8, 2005: Big Island, Hawaii; hosted by Plum Hall
(and, if needed, 1 hour at 9 am on May 11 to review work done post-main meeting)

Vote the spec out of TG5 then forward to the GA via the TC39 business meeting, to be held the afternoon of March 11.

The following phone conferences were scheduled to occur at 10 AM PST, for 2 hours max: February 3, 10, 17, and 24.

Action: Microsoft will host the teleconference facility; Herb will distribute the dial-in information.

8 Reports from Liaisons

8.1 TC39 TG3 (CLI) – Rex Jaeschke

Some time ago, it was proposed that CLS rule 25, which requires the accessors of a property to have the same accessibility, be relaxed. Although this was close to acceptance, the issue has been re-opened.

8.2 SC22/WG21 (C++) – Tom Plum, P.J. Plauger, Tana Plauger, John Spicer, and Steve Adamczyk.

No meeting has been held since the last TG5 meeting.

Herb reported that the idea of an enum class was well received by various members.

8.3 TC39 TG2 (C#) – Rex Jaeschke

None.

8.4 ISO/IEC JTC 1/SC 22 – Rex Jaeschke

None.

9 Action item and comment spreadsheet review

The issues owned by Brandon and marked Priority R (review) were closed, as all clauses will be reviewed by one or more review groups before the March meeting. These issues are: #10, 23, 24, 33, 34, 40, 54, 55, 71, 81, 98, 135, 154, 155, 158, 159, 160, 162, 163, 176, 191, and 199.

The issues owned by Brandon and marked Priority L (low) were reviewed, with the following decisions being made

#19: Closed without action.

#38: Reassigned to editor to list in Future Directions.

#57: Duplicate of #145; closed.

#62: Needs to be done.

#68: The existing words were deemed adequate; closed.

#87: Closed without action.

#109: Closed without action.

#127: Microsoft-specific; closed without action.

#130: Closed without action.

#131: Reassigned to editor.

#133: Closed without action.

#136: Reassigned to editor to mention this in the default indexer clause.

#139: Reassigned to editor.

#142: Already is in the draft; closed.

#143: Needs to be done.

#145: Reassigned to editor to move to future directions.

#148: Reassigned to editor.

#149: Reassigned to editor.

#165: Reassigned to editor. Say the following: "Types that cannot be deduced for function templates cannot be deduced for generic functions."

#167: Needs to be done. CV-qualifiers are not permitted.

#169: Needs to be done. Add example(s).

#170: Needs to be done.

#171: Needs to be done.

#173: Needs to be done. Add a description of our best guess at the correct solution, to Future Directions, then mark this Postponed.

#174: Microsoft-specific; closed without action.

#175: Microsoft-specific; closed without action.

#192: Closed without action.

#195: Duplicate of #130; closed.

#208: Closed without action. The standard will not mention modules.

#209: This is allowed. Closed without action.

A walk-through of the remaining issues took place:

#63: Herb presented an update on the latest thinking within MS w.r.t destructors and finalizers. This involved the use of the patterns Dispose() and Dispose(bool).

#186, Traits: Agreed to drop this. Closed without action.

#201, How to accommodate non-CLI calling conventions on other platforms: Agreed to postpone.

#212, Declared accessibility contradicting that required: 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.

#215: Added this as a new issue.

10 Any other business

10.1 Distribution of docs to WG21:

Action: Editor will distribute to the TG5 reflector, WD1.9, so members can make it available on their websites for access by WG21 members. Editor will also announce this availability to the liaison email reflector.

Action: Editor will concatenate the PDFs of all docs (except WD1.9) to WG21, and forward to Herb for distribution. (This package will include these draft minutes after TG5 has had a change to review and correct them via email.) This packet will include a document containing URLs from which the latest draft can be obtained.

10.2 Thank meeting host:

Everyone thanked meeting hosts EDG and Dinkumware for meeting and catering arrangements, and to Chris Walker of Dinkumware, Ltd., for network and audiovisual support.

11 Adjournment

The meeting was adjourned at 2:20 PM.