Doc No: SC22/WG21/N3213
PL22.16/10-0203
Date: 2010-11-26
Project: JTC1.22.32
Reply to: Stefanus Du Toit
Intel Corporation
stefanus.du.toit@intel.com

Minutes of PL22.16 Meeting, November 8, 2010

1. Opening activities

Clamage called the meeting to order at 09:00 (UTC-6) on Monday, November 8, 2010.

1.1 Opening comments, welcome from host

The host welcomed the attendees and provided some organizational information.

1.2 Introductions

Clamage had the attendees introduce themselves.

1.3 Meeting guidelines (Anti-Trust)

Clamage reviewed the patent disclosure rules.

The following materials were displayed without any further interpretation or discussion:

http://www.incits.org/pat_slides.pdf

http://www.incits.org/inatrust.htm

1.4 Membership, voting rights, and procedures for the meeting

Nelson reviewed guidelines for filling in the attendance sheet. Clamage reviewed the rules for membership and voting rights.

Clamage asked whether the Concurrency Working Group would be meeting, which Boehm confirmed.

Clamage reviewed the procedures for voting during the meeting. Clamage noted that the project schedule called for a completed draft by the Madrid meeting.

Clamage noted that 8 WG21 National Body delegations were present:

Canada, France, Finland, Netherlands, Spain, Switzerland, UK, US

1.5 Agenda review and approval

Clamage presented the agenda (document PL22.16/10-0128 = WG21/N3138).

Motion to approve the agenda:

Moved by: Hedquist

Seconded by: Du Toit

PL22.16		WG21	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

1.6 Distribution of position papers, WG progress reports, WG work plans for the week, and other documents that were not distributed before the meeting.

Each of the Working Group chairs presented their progress and plans for the coming week.

Core Working Group (CWG)

Adamczyk reviewed the CWG progress. He reported that there was much progress on the NB comments, but some controversy that would affect the directions taken by the group. He expected to have a meeting on Tuesday morning to cover several issues, including implicit move, implicit generation of noexcept, and a discussion of implicit noexcept on destructors.

Meredith commented that some LWG work depended on these discussions, but added that by no means all LWG work did.

Spicer noted that some presentations were previously prepared to explain some of these issues, and was wondering whether there was interest in similar presentations during the meeting.

Abrahams volunteered to show his presentation on noexcept again. Adamczyk asked whether anyone else would be available to present on these topics. Stroustrup noted he would be willing to present on the issue of implicit move generation.

With no objections being raised, Adamczyk confirmed that these issues would be discussed on Tuesday morning.

Library Working Group (LWG)

Meredith reported that LWG was in good shape with regards to NB comment responses. He noted that there were no similarly controversial issues in LWG, but LWG work would depend on resolutions on the controversial issues from CWG.

Concurrency Working Group

Crowl reported that most issues had been processed at the previous meeting, and substantial progress had been made on other issues, but there was still a bit of work remaining.

WG21 Report

1.7 Approval of the minutes of the previous meeting

Du Toit noted that he was informed of a mistake in the meeting minutes:

> Item 12.1 lists the dates for the Madrid meeting as March 11-26, when they should be March 21-26.

Motion to approve the minutes (N3119/N310), as amended above

Moved by: Brown

Seconded by: Hedquist

PL22.16		WG21	
In favor:	23	In favor:	7
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

1.8 Liaison reports

WG14 Report

Benito reported on the progress of the WG14 group. He noted a ballot resolution meeting would be held prior to the WG21 Madrid meeting. He added that participation had declined. He noted that WG14 had no explicit meetings other than the London meeting planned for 2011.

Stroustrup asked what it meant for participation to have declined. Benito answered that there were only two national bodies, US and Canada.

Meredith asked what the status of concurrency issue alignment between WG14 and WG21 was. Benito noted that as far as he was aware, all issues had been brought up and the two groups were aligned.

Plum added that there was syntax available for atomics that could be used in either C or C++. He stated that if there were any qualifiers to be attached to an atomic declaration, they should be placed outside of the parentheses following the atomic keyword. He noted that the group was unsure as to what to do in the corner cases of having qualifiers such as const and volatile applied to atomic declarations.

Nelson reported that there was one more moderately significant issue, a request from WG21 to WG14 to disallow atomic floating point values. He said that WG14 was unsure as to the rationale for the request, so more information was needed before action could be taken.

1.9 Editor's report

Becker reported that post Rapperswil meeting there was a revised working draft, N3126, which incorporated most of the NB comment resolutions made in Rapperswil. He noted the editor’s report, N3127, explained the changes made in that draft.

Motion to approve document N3126 as current working draft

Moved by: Becker

Seconded by: Wong

PL22.16		**WG21**	
In favor:	23	In favor:	7
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

1.10 New business requiring actions by the committee

PL22.16 selection of US delegates to WG21 for 2011

Clamage noted that a PL22.16 selection of US delegates needed to be made, and Hedquist confirmed this could be done at the Saturday meeting. Hedquist stated he would place the list of delegates on the motions page.

2. Organize subgroups, establish working procedures.

Clamage announced that those present would be breaking up into working groups until Friday. He noted that the committee was in recess until then.

3. WG sessions (Core and Library, possibly Concurrency, Evolution).

The group broke up to meet in separate working group sessions.

Tuesday, Aug 3, 8:30am-5:30pm

4. WG sessions continue.

Wednesday, Aug 4, 8:30am-5:30pm

5. WG sessions continue.

Thursday, Aug 5, 8:30am-5:30pm

6. WG sessions continue.

Friday, Aug 6, 8:30am–12 noon

7. WG sessions continue.

Friday, Aug 6, 1:30pm–5:30pm

8. General session.

8.1 WG status and progress reports.

Hedquist explained the general motion confirming the PL22.16 delegation to JTC1 SC22/WG21.

Library Working Group

Meredith reviewed the Library Working Group status. He announced a new email address, lwgchair@gmail.com, to which new issue reports should be sent. Meredith added that there were three new issue statuses, Immediate, Deferred, and Voting, for internal use in LWG during meetings.

Meredith explained that the issue numbers had been partitioned to make it clear which issues are related to responding to FCD comments, and which issues are not.

Meredith reviewed the new system for maintaining the issue database.

Meredith explained that all new papers had been addressed during the meeting. He explained that there had been little contention in regards to LWG issues and papers. Meredith went on to explain that a large number of issues related to clauses 29 and 30, and that these issues had been made the responsibility of the Concurrency Working Group.

Halpern asked what would happen to any of the issues in Immediate status if they were not voted in. Meredith answered that the issues would be placed in Open status.

Core Working Group

Adamczyk joked that proximity to some of the equipment to Fermilab has caused relativistic effects, because he had aged at least one month in the last week.

Adamczyk explained that he would be handling issues in roughly increasing order of controversy.

Adamczyk explained that there had been several NB comments that asked for features to be removed because they were not implemented yet. He noted that the group believed there to be only four such features left, with a great deal of implementation progress since the NB comments.

Nelson asked Wong what the likelihood of user-defined literals being implemented by IBM before the Madrid meeting was. Wong stated that this was his hope, but that he could not make any such promises.

8.2 Presentation and discussion of proposed responses to public comments. Straw votes taken.

The following motions had discussions prior to the taking of straw votes.

LWG Motion 3: *Move we apply the resolutions to the following Tentatively Ready issues from N3175 to the C++0X Working Paper:*
1333, 1334, 1335, 1337, 1338, 1339, 1340, 1517, 1518, 1519, 1520

Nelson asked whether there was any contention in the library group on LWG Motion 3. Meredith responded that there were none.

LWG Motion 5: *Move we apply N3110 "Problems with bitmask types in the library" to the C++0X Working Paper.*

Nelson asked whether this change depended on “constexpr”. Meredith explained that the changes fixed previous incorrect applications of “constexpr”.

LWG Motion 12: *Move we apply N3198, "Deprecating unary_function and binary_function", to the C++0X Working Paper.*

Nelson asked whether the new alternatives to unary_function and binary_function used variadic templates. Meredith responded that they did, although they were also present in TR1 without variadic templates.

Halpern commented that the two functors that were not deprecated should go away eventually, but would require a replacement facility to be provided first.

LWG Motion 14: *Move we apply N3189, "Observers for the three handler functions", to the C++0X Working Paper.*

Maurer asked whether the wording to add synchronization was reviewed by the Concurrency Working Group. Sommerlad confirmed that it had been.

LWG Motion 17: *Move we apply the edits proposed in the following two papers to the C++0X Working Paper.*
N3148 "throw() becomes noexcept (Version 2)"
N3150 "Removing non-empty dynamic exception specifications from the library"

Nelson asked whether anyone had actually implemented noexcept, including noexcept expressions. Meredith and Merrill responded that it had been implemented in GCC.

LWG Motion 18: *Move we apply N3195, "From Throws: Nothing to noexcept", to the C++0X Working Paper.*

Nelson asked whether this change was a simple global edit, with no exceptions. Meredith confirmed that it was.

LWG Motion 20: *Move we apply N3156 "More on noexcept for the diagnostics library", to the C++0X Working Paper.*

Wong asked how certain the group was that these changes would not cause any problems. Meredith stated that the group was very certain and gave an example of there being no dynamic allocation in any of these cases.

LWG Motion 22: *Move we apply N3180, "More on noexcept for the Strings Library", to the C++0X Working Paper.*

Svoboda asked whether this paper covered the case of strings being encoded in, say, UTF-8, and containing an illegal character. Austern responded that while some parts of the library do consider the encoding, it does not affect the parts in question.

Willcock asked whether a string implementation could internally use a variable length encoding. PJ Plauger responded that it could, but it would not matter because it would not affect any string operations.

LWG Motion 26: *Move we apply N3188, "Async Launch Policies", to the C++0X Working Paper.*

Hinnant asked whether an implementation could provide an implementation-defined "any" launch policy. Sommerlad confirmed that this was the case, and implementations were encouraged by the paper to provide their own launch policies. Meredith added that BSI would like to see such a feature added back but was happy with the paper and would file an issue on it.

Austern asked whether library vendors would have a potential conflict with a user macro. Hinnant noted that "any" is sufficiently reserved by the library to avoid such user macro conflicts. Austern stated that he was satisfied with this.

LWG Motion 27: *Move we apply N3191, "C++ Timeout Specification", to the C++0X Working Paper.*

Nelson asked whether this version has been implemented. Crowl responded that he believed the current implementations conform to the new proposed wording semantically. Nelson noted that an earlier version of the paper had implementation concerns on certain operating systems. Crowl and Vollman confirmed that this had been fixed.

LWG Motion 28: *Move we apply n3193, "Adjusting C++ Atomics for C Compatibility", to the C++0X working paper.*

Crowl asked if there was any objection to making a few fixes in editorial nature to better separate descriptive text and actual wording between the straw poll and the votes the next day. Boehm asked what the procedural rules would be in this case. Nelson stated that when the group switched to its present set of rules, there only had to be complete transparency about any such changes, and that therefore this was acceptable within those rules.

CWG Motion 6: *Move we apply N3190 "C and C++ Alignment Compatibility" to the C++0X Working Paper.*

Willcock asked whether the syntax still allowed the use of an expression, e.g. a number, in an alignas specifier. Adamczyk confirmed that it did.

Meredith asked whether comma separated lists were allowed in alignas. Adamczyk responded that they were not.

Meredith stated that he found it surprising for the syntax for pack expansion to not have the ellipses after the closing parenthesis. Adamczyk noted that the group had considered this, but rejected it due to potential parsing ambiguities.

Willcock asked, given this, why it was not allowed to use a comma-separated list of types in the expression. Adamczyk explained that there were some difficulties, such as being inconsistent with expressions, which caused the group to not choose to allow this at this time.

Sommerlad asked whether the C standard might allow such a comma-separated syntax. Adamczyk responded that it did not, and that he did not think it would do so in the future.

CWG Motion 9: *Move we apply N3204 "Deducing 'noexcept' for destructors" to the C++0X Working Paper.*

Willcock stated that he was concerned about the backwards compatibility of this change. He claimed that any kind of class whose destructor might throw, like an RAII class, would just crash in this case.

Witt asked how it was possible to write a useful RAII class with a throwing destructor.

Willcock stated that he was thinking of simple things like scoped_guards.

Plum noted that these cases were well-discussed in the Core Working Group and at the preceding meeting.

CWG Motion 10: *Move we apply N3205 "Delete operators default to noexcept" to the C++0X Working Paper.*

Meredith asked for confirmation that this function does not call a destructor, it just cleans up the memory after the destructor is done. Adamczyk confirmed this.

More discussion occurred along these lines.

CWG Motion 11: *Move we apply N3206 "Override control: Eliminating Attributes" to the C++0X Working Paper.*

Meredith asked what the interactions were with late specified return types. Adamczyk reported that they had been considered and issues around them had been resolved.

Austern asked for an explanation of an example shown on the projector. Adamczyk explained the example.

Stroustrup stated that many of those present knew he had been speaking against these kinds of proposals for many years. He went on to say that he had not had a brisk conversion to liking it, but that his reason for supporting this proposal is very simple. He explained that he sent out four examples that morning of how to write programs, and if attributes were used, he explained, one would see many braces, and this would lead to macros defined by users to hide the braces. He stated that he liked the suggested approach because it provided one solution for the problem that everyone would use.

Meredith asked whether there was a special rule for the copy assignment operator in regards to “hiding new”. Adamczyk responded that yes, all special member functions are exempt from that.

Miller explained that if one used “hiding new” and there wasn’t something being hidden from the base class, one would get a compile error. He stated that one needed to specify it when one needed it, and one needed to omit it in the other cases.

Halpern asked how close this was to Microsoft’s existing practice in terms of syntax and meaning. Adamczyk answered that it was very similar. He explained that Microsoft’s use of new meant a new vtable slot.

Hall stated that the answer to Sutter’s question was “closer than you think.”

Adamczyk stated that in most cases it would come to be very close. He explained that one would use “final” instead of “sealed”, but that the feature otherwise behaved the same there.

Willcock asked to clarify that using “new” here didn’t require creating a new vtable slot. Adamczyk explained that it did not force the creation of a new vtable slot, and more precisely, it did not have any effect on the existing rules of vtable slots.

Lakos asked, if the class was final and explicit, whether the use of final on members was an error, since it was redundant. Adamczyk stated that he understood the point, but that the code being shown was just an example.

Lakos said that with explicit, it seems it should create an error.

Witt said that he liked 90% of this. He explained that the one area where he thought the group had erred was “explicit”. In essence, he said, the proposal put a marker in the class that expressed a requirement for slightly different rules than C++ had so far. He elaborated that the way override and virtual worked was one area where slightly different rules might be preferable, and that there were other areas where slightly different rules might apply. In the future, he explained, the group would not be able to reuse explicit to cover these cases like making constructors explicit by default.

Adamczyk reported that the group had had long discussions on this, and many votes. He stated that this by far reached the best consensus, but nonetheless everyone was likely to have some problem with it. He explained that the alternative was to stay with the status quo, which was not broken, but he noted that some people did not like that, and felt this was better.

A representative from France stated that the group did not have enough time to review the proposal because it was not a pre-meeting paper.

Adamczyk stated he believed anyone could invoke the two-week rule.

Plum explained that as long as there was a paper in the mailing, the committee was free to make revisions. He said that the question was whether people would be surprised that the topic was discussed. He believes it was clear in this case that the topic would be discussed.

Benito stated that the two week rule only applied to PL22.16. He explained that for ISO, anyone could object, but should be able to talk about it.

Gregor pointed out that N3163 by Herb Sutter had almost exactly the same wording as what was being proposed.

Crowl asked whether, when the proposal said “eliminates all the attributes”, it meant only the attributes being replaced by the proposal.

Adamczyk answered that yes, the attribute facility was being kept. He explained that we would only have two attributes if these papers went in: “noreturn” and “carries-dependency”.

The representative from France agreed that procedurally there was a paper in the pre-meeting, but said that there had not only been one proposal this week, but three. He expressed that he felt this had not yet settled, and that France would vote no because it was too new.

Lakes asked whether the group should vote for this proposal now and try to tweak it later, or wait, adjust it, and adopt it later. Adamczyk responded that even in general, the answer would be to vote it in now and tweak it later. He added that this was especially prudent in this case because of the timing.

Lakos stated that he had some confusion about the meaning of “new”.

Wiegley stated that he liked the “new” keyword, as it appeared to say “this is a new thing – disregard what’s in the base class”.

Adamczyk opined that he expected the understanding of this keyword would improve over time.

Some more discussion ensued.

Hinnant thanked those involved for putting a complete example in the standard.

CWG Motion 12: *Move we apply N3203 "Tightening the conditions for generating implicit moves" to the C++0X Working Paper.*

CWG Motion 13: *Move we apply N3216 "Removing Implicit Move Constructors and Move Assignment Operators" to the C++0X Working Paper.*

Stroustrup claimed that N3202 was better than the status quo because it avoided problems with backwards compatibility, was a very small change to the FCD, and built on the consensus that has been built over the years. He explained that it was consistent with copy and move rules, which he noted was important because a lot of problems are caused by people getting copies and moves out of sync. He stated that he did not want to burden to users who want to use moves for simple types, in particular aggregates, and unless the standard guaranteed generating moves in the obvious cases, myths would build.

Wiegley stated that he was not convinced that N3202 did enough to guard against breaking cases, which he opined might be sufficient for people to avoid using C++0x in the first place.

Austern noted that N3216 made the standard library class “array” much less useful. Merrill explained that an aggregate with an explicitly defaulted move constructor remained an aggregate.

Lakos stated that he originally preferred N3216, until he heard Abrahams mention that if the group was going make a change in this area, it should something of value. He added that as a library writer, he would never default everything. He stated that he considered himself outside of the “lazy camp”, which would include people like application developers. He stated that in cases like taking an aggregate and putting it in a vector, N3202 would be better. So he concluded the choice did not affect him directly, and therefore he was prepared to take a bigger risk and get a bigger reward.

Gregor responded that one of the other things that Abrahams said is that he preferred the status quo over N3202, because if the group was going to take the risk, they should make it a bigger risk for a bigger win.

Du Toit noted that aggregates could be made to have move constructors implicitly.

Witt stated that it was difficult to talk about what would make it more difficult for users, and that no one knew exactly how much breakage or pain any of the options would cause. As an application developer, he added, he was also concerned about breakage. He reiterated that he would rather have his code not broken than make it easier to write in the first place.

Adamczyk asked if part of the characterization of N3202 was that it would reduce such breakage. Stroustrup answered affirmatively. He explained that it would still

cause breakage, and that that was a given no matter what course of action was taken. So, he concluded, the group desired to minimize along several constraints. He said that he saw a fair number of examples in the discussions on the web and in the paper, some but not all of which would be taken care of by N3202. He explained that the cases handled well were the ones where someone had a special invariant used in a special member function. He contrasted that the proposal did not handle, and could not possibly handle, cases where people have a simple collection of objects, and someone knows that there is some relation between data members but it's not expressed in any explicit functions. He stated that a lot of really simple data structures had a constructor, but nothing else.

Witt mentioned that there was an important distinction the group had not talked about and needed to be clear about. In his understanding, N3216 would not break any existing code, but N3202 had the potential of breaking existing code. He noted that N3202 did come with a benefit, which if he understood Stroustrup correctly was that someone started using move constructors it would probably be easier to get it right with N3202 than N3216.

Gregor explained that if someone wanted to write a move constructor, it is easy with both options to write it for the common simple case, because one could write "=default". N3202 would provide this for free for existing classes, but could also introduce breakage.

Austern stated that he was concerned about losing move constructors for simple aggregates. He noted that retaining just these was not an option on the table at the meeting, and may or may not be at next meeting.

Lakos stated that application developers would get a performance boost without writing it, not just for existing code but for future code.

Stroustrup responded to Austern's point. He mentioned that the group did think about the aggregate case, but that still left many classes that just add a constructor. He gave one reason for being concerned about the ease of use of move, which is presenting move constructors as a new idiom for how one returns big data structures out of functions. He stated that he wanted to allow returning things by value, without leading people into the memory management problems they get today.

Witt asked Austern to clarify that when he said "cannot move an aggregate", he really meant "cannot move it efficiently". Austern confirmed that was what he meant.

Svoboda mentioned a design tenet the group had, which was not to charge people for things they did not ask for. He opined that N3202 did not modify or break classes that knew anything about how they were to be copied or manipulated.

Brown stated that people who have chosen not to write copy constructors or copy assignments would get the defaults and additional functionality for free.

Svoboda added that they would also get possible breakage.

Brown stated that if someone had not written their own copy functions, the assumption was that the implicitly generated copy functions would be acceptable. He added that he did not see a big difference to the implicitly generated move functions.

More discussion ensued.

Woodcock stated that it was not just necessary for a class to have an invariant, but also for that invariant to be used in the destructor or left hand side of the assignment.

Wiegley stated that while that might be true, if a user were to run into that problem, they might not even know that the implicit move existed. He concluded that the user would be paying for something they never asked for.

Stroustrup stated that when the group first saw move constructors and rvalue references, it was reassuring that one could not get into trouble unless one called move(). He noted that the problem was that the library vendors had filled their libraries with it, so it was really hard for users to know that things would be moved from. He continued to explain his position.

Dennett stated that it was not just if one used it in a destructor or assignment operator. He explained that an object might be in a container, with some moved-from shells in such a container, and anything could happen on these objects.

Witt stated that the only time one could have one of these problems was if someone automatically introduced move constructors.

Meredith announced that Wiegley had said something that changed his position. He felt that many developers would put in the =default without thinking, and that taking away the need for people to put in affirmative statements was a good thing.

Stroustrup opined that if the group took the philosophy that explicit was better than implicit, one would see lots of cases with 6 defaulted things. Users would end up writing a macro for that.

Crowl explained that under N3202, if someone did not want a default move constructor used, they would need to explicitly mention the copy constructor, and not delete the move constructor.

Witt stated that he did not care so much how hard it would be to fix, and could even live with it being relatively hard to fix. In his view, he said, the problem was how to diagnose it, how to find the breakage. He explained that he was scared of having to review 100,000 lines of source to find a problem, because the only other chance would be runtime testing.

Sommerlad asked if static analysis tools would be able to detect the problematic situation the group might get into with N3202.

Svoboda said he had a hard time imagining that because invariants are not in code anywhere.

Widman explained that it was very difficult for a static analysis tool to deduce the class invariants. He doubted the possibility of doing so in a significant number of cases.

Stroustrup added that, however, it would be trivial for an analysis tool to go through and look for things that had move operations that did not in C++03. This could provide users with the same information as what is provided by N3216.

Austern stated that he was not at all scared to read over things in his codebase to find places that might break, because the conditions under which moves would be generated were fairly small.

Lakos said he had heard that compiler vendors could provide a switch to disable this, and asked if that was true.

Adamczyk said he was suspicious of that claim. He explained that a switch that changes behavior based on what is and isn't a problem would have ABI issues.

Some more discussion ensued.

Wong explained that the biggest concern for IBM was the breaking of existing user code, and that this was probably true for other compilers too. He asked if there was anything anything that the compiler could do, perhaps with a runtime tool, or something like that?

Wiegley responded to Austern's earlier comment, stating that the group could not use itself as metrics as to how reasonable or unreasonable it is to find problems. He stated he had past experience when assisting new users of them making mistakes members of the group would never even think of making.

Brown responded to Wong, saying that there was a feature he had long wished compilers would offer related to this, a flag that would inform the user of what functions were generated implicitly. He added that this would be useful even today.

Widman noted that Gimpel has a feature similar to what Brown proposed.

Hinnant agreed with Brown that such a feature would be of great use.

Stroustrup stated that some of the problems seen today with the kind of users talked about by Wiegley include forgetting to define copy constructors or assignment operators. He said that this set of rules would help with both problems, and that people could increase their compiler's warning level.

Based on straw votes, Motion 13 was removed.

CWG Motion 14: *Move we apply N3207 "noexcept(auto)" to the C++0X Working Paper.*

Crowl asked whether the paper did or did not have implicit deduction in the absence of noexcept(auto). Adamczyk responded that it did not.

Austern asked whether there was any form of implicit deduction with defaulted special member functions. Merrill stated that the same mechanisms as there always were remained.

Wiegley asked whether anyone had implemented noexcept(auto). Merrill responded that no one had.

Gregor presented a short set of slides on noexcept(auto).

Meredith stated that this looked like a useful implementation technique for implementing, but from LWG experience this would not at all affect library specification.

Crowl said that the first time someone enabled optimization in their compiler settings they would get a different code path in this proposal.

Halpern stated that he did take issue with how poor the current specification was. But he added that he was concerned this feature would encourage over-use of noexcept, and that there was only a relatively small set of cases where one would really care about this. With this feature he said one could easily extend it to everything, which would increase compile times. He concluded that he was concerned that the feature needed to be better thought out.

Wiegley asked, with deduction happening for all function bodies, what would happen if one had a definition in another translation unit. Gregor explained that if one had a noexcept(auto) function declaration, one could not use it until it was defined.

Stroustrup stated he was worried about the complexity of the feature, and worried about it affecting implementation rather than specification, despite having brought this forward.

Du Toit objected to the paper on the grounds that it was clearly a new feature in his view, and could be added independently by compiler vendors.

Widman stated that while it was a new feature, it was also a regression fix for an added feature.

Liber said that he did not think noexcept with a conditional was really useful without noexcept(auto).

Nelson stated that it was his understanding that the NB comments about figuring out whether noexcept can be applicable were much more limited than what this is offering.

Gregor said he felt like the current state of noexcept was pretty close to unusable. He felt the group could handle it as experts in the standard, but users would not get it right. He felt that while the group could teach the pitfalls in the standard we can teach, noexcept had great complexity, and for that reason, despite his usual stance, said that he was in favor of noexcept(auto).

Sommerlad stated that IDEs/tools can generate noexcept specification in the same way that noexcept(auto) does. He mentioned that he was willing to have a student implement this in the next academic term.

Plum stated that if the group did not put in noexcept(auto), it would not be usable in a portable program. He said that the alternatives described by Ottosen were very serious looking. He saw this as a clear improvement over the original proposal to infer exception specifiers automatically.

Austern agreed with all of Gregor's points from his presentation: the group had painted itself into a corner, and noexcept could be very hard to use. He did not agree with the implicit assertion that noexcept(auto) would solve the usability problem.

Halpern said that despite continually hearing about the usability problem with noexcept, one did not need to decorate every function with whether or not it throws. He added that move constructors were the only case one really cared about. He summarized that the purpose of noexcept(auto) was to decorate 90% of code, when only 2% of code would need it.

Lakos observed that if decorated the majority of functions were decorated with noexcept, one would not be able to put in defensive checks and test them in that manner. He stated that if a defensive check were conditionally compiled, that would be one issue. But he stated that he also wished to check arguments passed into functions. He claimed that noexcept would negate the ability to do these kinds of negative checks. He concluded that he wanted to see noexcept used surgically, and that move constructors and move assignments were good places to do this because no assertions would need to be made on the argument.

Miller said that it was clear to him that the proposal had not had enough time. He stated that this was something that could be upward compatibly added through a TR or vendor extensions. He viewed the usability problems shown as real issues and was therefore strongly against the proposal.

Meredith reported that he had been playing with noexcept a little bit, and that the complexity did not scare him. He said that it was designed for move constructors to make things work in the library, and that that use was very simple. He said that unconditional usage was very simple, and that having noexcept(auto) would let users automatically get what's bad for them.

Stroustrup said that noexcept(auto) was such a small neat feature that it would encourages overuse, not just application for these "surgical" operations. He said that it would encourage people to think on a very detailed level when metaprogramming, and conditions on noexcept would percolate through the system.

Brown noted that he co-authored a paper in 2004 that proposed something like noexcept, called nothrow and nothrow_if. He stated that one of the real reasons it had been proposed was the need for performance in the High Energy Physics community and the fact that such a feature would be visible at each calling site for the purpose of improving the calling code, to avoid setup overhead for calling sites. He reported that it was pointed out to him in a brief but very enlightening conversation that using what was now called noexcept for such a purpose, i.e. to produce better code at the call site,

was largely wasted, because modern compilers were so good at using whole program analysis that they could detect that exception setup is not necessary in many cases.

Crowl responded that there was an optimization opportunity, but it did not go as far as one might hope.

Brown continued that he was persuaded at the moment that abuse of noexcept in any form, including noexcept(auto), was not going to give people the naïve performance improvement they might be looking for. He felt persuaded by the argument that it was a big feature, coming late in the process, and that it could be added on in this or another form later. He clarified that he did not dislike the feature and liked what it seemed to promise but that it had not had enough bake time. He concluded that he was voting against it for that reason, but would like to see it come back later.

Becker said he was intrigued by the comments that the primary purpose of noexcept was to make move constructors safe and that the group would have to worry about abuse of noexcept. He was concerned that the group just added noexcept to many places in the library.

Du Toit responded that almost all the places where the group added it were unconditional. He explained that conditional noexcept was the problematic case here.

Garcia concurred with Du Toit. He reported that the only cases where conditional noexcept was used pertained to move constructors and move assignment operators.

Halpern agreed, and noted that the other cases where the library was decorated were basically trivial.

Meredith wanted to address the notion as to whether this could be done as a bolt-on later, and mentioned that if the group added this later it might affect API compatibility.

Based on straw votes, Motion 14 was removed.

9. WG sessions continue

Saturday, Aug 7, 8:30am-12 noon

10. WG sessions continue

Saturday, Aug 7, 1:30pm–5:00pm

11. Review of the meeting

11.1 Motions

PL22.16 Motions

PL22.16 Motion 1

Move the following voting members of PL22.16 be designated as the US delegation to JTC1 SC22/WG21 for any and all WG21 meetings for 2011:

- Barry Hedquist, Perennial;
- Steve Clamage, Oracle;
- J. Stephen Adamczyk, Edison Design Group;
- Howard Hinnant, Apple Computer;
- Walter Brown, Fermi National Accelerator Laboratory;
- Tana Plauger, Dinkumware Ltd.;
- Thomas Plum, Plum Hall;
- Bjarne Stroustrup, Texas A&M University;
- Clark Nelson, Intel.

Moved by:	Hedquist
Seconded by:	Nelson

PL22.16	
In favor:	23
Opposed:	0
Abstain:	0

Motion carries.

Core Motions

Motion 1

Move we apply the resolutions of all issues in "Ready" and "Tentatively Ready" status from N3159 **except for issue 1027** to the C++0X Working Paper.

- **Ready:** 341 1012 1062 1070 1087 1103 1104 1107 1112 1113 1114 1121 1126 1128 1129 1130 1131 1138 1139 1144 1146 1152 1154 1155 1158 1159 1160 1161 1164 1168 1169 1171 1173

- **Tentatively Ready:** 694 964 1006 1009 1011 1016 1020 1025 1029 1034 1036 1037 1047 1051 1061 1064 1066 1069 1072 1075 1083 1086 1102 1106 1117 1119 1122 1134 1142 1148 1153 1156 1165

Moved by:	Adamczyk
Seconded by:	Hedquist

PL22.16		**WG21**	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 2

Move we apply N3146 "Recommendations for extended identifier characters for C and C++" to the C++0X Working Paper.

Moved by:	Adamczyk
Seconded by:	Nelson

PL22.16		**WG21**	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 3

Move we apply N3209 "Progress guarantees for C++0x" to the C++0X Working Paper.

Moved by:	Adamczyk
Seconded by:	Hedquist

PL22.16		**WG21**	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 4

Move we apply N3196 "Omnibus Memory Model and Atomics Paper" to the C++0X Working Paper.

Moved by:	Adamczyk
Seconded by:	Nelson

PL22.16		**WG21**	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 5

Move we apply N3214 "US 19: Ambiguous use of 'use'" to the C++0X Working Paper.

Moved by:	Adamczyk
Seconded by:	Hedquist

PL22.16		**WG21**	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 6

Move we apply N3190 "C and C++ Alignment Compatibility" to the C++0X Working Paper.

Moved by:	Adamczyk
Seconded by:	Crowl

PL22.16		**WG21**	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 7

Move we apply N3217 "Wording for brace-initializers as default arguments" to the C++0X Working Paper.

Moved by:	Adamczyk
Seconded by:	Stroustrup

PL22.16		**WG21**	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 8

Move we apply N3218 "Core Issue 1125: Unclear definition of 'potential constant expression'" to the C++0X Working Paper.

Moved by:	Adamczyk
Seconded by:	Merrill

PL22.16		WG21	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 9

Move we apply N3204 "Deducing 'noexcept' for destructors" to the C++0X Working Paper.

Moved by:	Adamczyk
Seconded by:	Hedquist

PL22.16		WG21	
In favor:	19	In favor:	6
Opposed:	1	Opposed:	0
Abstain:	3	Abstain:	2

Motion carries.

Motion 10

Move we apply N3205 "Delete operators default to noexcept" to the C++0X Working Paper.

Moved by:	Adamczyk
Seconded by:	Hedquist

PL22.16		WG21	
In favor:	18	In favor:	7
Opposed:	0	Opposed:	0
Abstain:	5	Abstain:	1

Motion carries.

Motion 11

Move we apply N3206 "Override control: Eliminating Attributes" to the C++0X Working Paper.

Moved by:	Adamczyk
Seconded by:	Hall

PL22.16		**WG21**	
In favor:	19	In favor:	6
Opposed:	1	Opposed:	1
Abstain:	3	Abstain:	1

Motion carries.

Motion 12

Move we apply N3203 "Tightening the conditions for generating implicit moves" to the C++0X Working Paper.

Moved by:	Adamczyk
Seconded by:	Hedquist

PL22.16		**WG21**	
In favor:	20	In favor:	8
Opposed:	1	Opposed:	0
Abstain:	2	Abstain:	0

Motion carries.

Library Motions

Motion 1

Move we apply the resolutions to the following Ready issues from N3175 to the C++0X Working Paper: 868, 951,

Moved by:	Meredith
Seconded by:	Hinnant

PL22.16		WG21	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 2

Move we apply the resolutions to the following Tentatively Ready issues from N3175 to the C++0X Working Paper:
956, 1118, 1171, 1181, 1183, 1191, 1198, 1207, 1234, 1240, 1249, 1292, 1295, 1316, 1319, 1323, 1325, 1404, 1414, 1432, 1449, 1516

Moved by:	Meredith
Seconded by:	Hinnant

PL22.16		WG21	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 3

Move we apply the resolutions to the following Tentatively Ready issues from N3175 to the C++0X Working Paper:

1333, 1334, 1335, 1337, 1338, 1339, 1340, 1517, 1518, 1519, 1520

Moved by:	Meredith
Seconded by:	Hinnant

PL22.16		WG21	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 4

Move we apply the resolutions to the following Immediate issues from N3208 to the C++0X Working Paper:

1294 1354 1362 1368 1370 1435 1436 1437 1439 1440 1522

Moved by:	Meredith
Seconded by:	Hinnant

PL22.16		**WG21**	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 5

Move we apply N3110 "Problems with bitmask types in the library" to the C++0X Working Paper.

Moved by:	Meredith
Seconded by:	Hinnant

PL22.16		**WG21**	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 6

Move we apply N3142, "Adjustments to constructor and assignment traits", to the C++0X Working Paper.

Moved by:	Meredith

Seconded by:	Hinnant

PL22.16		WG21	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 7

Move we apply N3215, "Fixing LWG 1322, Explicit CopyConstructible? requirements are insufficient", to the C++0X Working Paper.

Moved by:	Meredith
Seconded by:	Hinnant

PL22.16		WG21	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 8

Move we apply N3173, "Terminology for constructing container elements", to the C++0X Working Paper.

Moved by:	Meredith
Seconded by:	Hinnant

PL22.16		WG21	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 9

Move we apply N3143, "Proposed wording for US 90", to the C++0X Working Paper.

Moved by:	Meredith
Seconded by:	Hinnant

PL22.16		**WG21**	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 10

Move we apply N3123, "Bringing result_of near to INVOKE", to the C++0X Working Paper.

Moved by:	Meredith
Seconded by:	Hinnant

PL22.16		**WG21**	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 11

Move we apply N3140 , "Cleanup of pair and tuple", to the C++0X Working Paper.

Moved by:	Meredith
Seconded by:	Hinnant

PL22.16		WG21	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 12

Move we apply N3198, "Deprecating unary_function and binary_function", to the C++0X Working Paper.

Moved by:	Meredith
Seconded by:	Hinnant

PL22.16		WG21	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 13

Move we apply N3210, "New wording for arithmetic on ratios", to the C++0X Working Paper.

Moved by:	Meredith
Seconded by:	Du Toit

PL22.16		WG21	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 14

Move we apply N3189, "Observers for the three handler functions)", to the C++0X Working Paper.

Moved by:	Meredith
Seconded by:	Hinnant

PL22.16		**WG21**	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 15

Move we apply N3158 , "Missing preconditions for default-constructed match_result objects", to the C++0X Working Paper.

Moved by:	Meredith
Seconded by:	Hinnant

PL22.16		**WG21**	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 16

Move we apply N3168, "Problems with Iostreams Member Functions", to the C++0X Working Paper.

Moved by:	Meredith
Seconded by:	Hinnant

PL22.16		**WG21**	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 17

Move we apply the edits proposed in the following two papers to the C++0X Working Paper.

N3148 "throw() becomes noexcept (Version 2)"

N3150 "Removing non-empty dynamic exception specifications from the library"

Moved by:	Meredith
Seconded by:	Hinnant

PL22.16		**WG21**	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 18

Move we apply N3195, "From Throws: Nothing to noexcept", to the C++0X Working Paper.

Moved by:	Meredith
Seconded by:	Hinnant

PL22.16		**WG21**	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 19

Move we apply N3155 , "More on noexcept for the language support library", to the C++0X Working Paper.

Moved by:	Meredith
Seconded by:	Hinnant

PL22.16		**WG21**	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 20

Move we apply N3156 "More on noexcept for the diagnostics library", to the C++0X Working Paper.

Moved by:	Meredith
Seconded by:	Hinnant

PL22.16		**WG21**	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 21

Move we apply N3199, "More on noexcept for the General Utilities Library", to the C++0X Working Paper.

Moved by:	Meredith
Seconded by:	Hinnant

PL22.16		**WG21**	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 22

Move we apply N3180, "More on noexcept for the Strings Library", to the C++0X Working Paper.

Moved by:	Meredith
Seconded by:	Hinnant

PL22.16		**WG21**	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 23

Move we apply N3197, "Lockable requirements for C++0x", to the C++0X Working Paper.

Moved by:	Boehm
Seconded by:	Crowl

PL22.16		**WG21**	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 24

Move we apply N3192, "Managing C++ Associated Asynchronous State", to the C++0X Working Paper.

Moved by:	Boehm
Seconded by:	Crowl

PL22.16		WG21	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 25

Move we apply N3194, "Clarifying C++ Futures", to the C++0X Working Paper.

Moved by:	Boehm
Seconded by:	Crowl

PL22.16		WG21	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 26

Move we apply N3188, "Async Launch Policies", to the C++0X Working Paper.

Moved by:	Boehm
Seconded by:	Crowl

PL22.16		WG21	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 27

Move we apply N3191, "C++ Timeout Specification", to the C++0X Working Paper.

Moved by:	Boehm
Seconded by:	Crowl

PL22.16		WG21	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Motion 28

Move we apply n3193, "Adjusting C++ Atomics for C Compatibility", to the C++0X working paper.

Moved by:	Boehm
Seconded by:	Crowl

PL22.16		WG21	
In favor:	23	In favor:	8
Opposed:	0	Opposed:	0
Abstain:	0	Abstain:	0

Motion carries.

Additional Motions

Nelson moved to thank the host. Applause ensued.

Brown moved to thank the officers, the scribes, the editor and everyone else who contributed to making the meeting the success it was. Applause ensued.

11.2 Review of action items, decisions made, and documents adopted by the committee

Clamage noted that there were no items for discussion.

11.3 Issues delayed until today.

Clamage reported that there were no issues delayed until today.

12. Plans for the future

12.1 Next and following meetings

Clamage presented the meeting schedule for upcoming meetings:

- Mar 21-26, 2011: Madrid, Spain
- Aug 15-19, 2011: Bloomington, Indiana, USA

Clamage noted that the Madrid meeting schedule would be one hour delayed beyond the usual schedule.

Becker stated that Indiana University in Bloomington would host the Summer 2011 meeting.

12.2 Mailings

Nelson reviewed the following mailing deadlines:

- November 26, 2010: Post-Batavia mailing
- February 25, 2011: Pre-Madrid mailing

13. Adjournment

Clamage asked whether there was any other business. There was no other business.

Hedquist moved to adjourn. Nelson seconded.

The meeting was adjourned at 14:09 (UTC-6) on Saturday, November 13, 2010.

Attendance

Company/Organization	NB	Representative	Mon	Tue	Wed	Thu	Fri	Sat
Apple Computer		Howard E. Hinnant	V	V	V	V	V	V
Apple Computer		Doug Gregor	A	A	A	A	A	A
Bloomberg		John Lakos	V	V	V	V	V	V
Bloomberg		Alisdair Meredith	A	A	A	A	A	A
Bloomberg		Dietmar Kuehl	A	A	A	A	A	A
BoostPro Computing		David Abrahams	V	V	V	V		
		John Wiegley	A	A	A	A	A	A
Carnegie Mellon University		David Svoboda	V	V	V	V	V	
Dinkumware		P. J. Plauger	V	V	V	V	V	V
Dinkumware		Tana Plauger	A	A	A	A	A	A
Edison Design Group		J. Stephen Adamczyk	V	V	V	V	V	V
Edison Design Group		Jens Maurer	A	A	A	A	A	A
Edison Design Group		William M. Miller	A	A	A	A	A	A
Edison Design Group		John H. Spicer	A	A	A	A	A	
Gimpel Software		James Widman	V	V	V	V	V	V
Google		Matthew Austern	V	V	V	V	V	V
		Lawrence Crowl	A	A	A	A	A	A
		James Dennett	A	A	A	A	A	A
Hewlett-Packard		Hans Boehm	V	V	V	V	V	V
IBM	CA	Michael Wong	V	V	V	V	V	V
IBM		Paul E. McKenney	A	A	A	A		
Indiana University		Jeremiah Willcock		A	V	A	A	V
		Larisse Voufo	V	V	A	V	V	A
Intel		Clark Nelson	V	V	V	V	V	V
Intel		Pablo Halpern	A	A	A	A	A	
Intel	CA	Stefanus Du Toit	A	A	A	A	A	A
Microsoft		Mark Hall	V	V	V	V	V	V
Oracle		Stephen D.	V	V	V	V	V	V

Company/Organization	NB	Representative	Mon	Tue	Wed	Thu	Fri	Sat
		Clamage						
Perennial	US	Barry Hedquist	V	V	V	V	V	V
Plum Hall		Thomas Plum	V	V	V	V	V	V
Red Hat		Jason Merrill	V	V	V	V	V	V
Red Hat		Benjamin Kosnik	A	A	A	A	A	
Riverbed Technology		Kyle Kloepper	A	A	A	A	A	
Roundhouse Consulting		Pete Becker	V	V	V	V	V	V
Symantec		Mike Spertus	V	V	V		V	V
Texas A&M		Bjarne Stroustrup	V	V	V	V	V	V
Zephyr Associates		Thomas Witt	V	V	V		V	V
	PL22.16 Non-members							
Blue Pilot		John Benito	N	N	N	N	N	
DRW Holdings		Robert Douglas	N	N		N	N	
		Nevin Liber	N	N	N	N	N	
HSR	CH	Peter Sommerlad	N		N	N	N	N
Symbio	FI	Ville Voutilainen	N	N	N	N	N	N
University Carlos III	ES	J. Daniel Garcia	N	N	N	N	N	
Vollmann Engineering	CH	Detlef Vollmann	N	N	N	N	N	N
[No Affiliation]		Faisal Vali	N			N	N	
		Alan Talbot	N	N	N	N	N	