Document number: N3622
Date: 2013-05-03
Reply to: Kyle Kloepper (Kyle.Kloepper@riverbed.com)

Minutes

WG21 Meeting No. 55
15-20 April 2013 – Bristol, UK

1 Opening activities

Steve Clamage calls the meeting to order Monday 15 April 2013 at 9:19 a.m. BST.

1.1 Opening comments, welcome from host

Clamage introduces himself and mentions there will be over 100 participants at this meeting. He also relates that Herb Sutter is in transit and expected to arrive by the afternoon session.

Clamage invites the host to say a few words. Roger Orr welcomes everyone to sunny Bristol. He thanks the sponsors and to much applause he announces that lunch will be provided.

1.2 Introductions

Everyone in the room introduces themselves. Clamage asks for a show of hands for all first time attendees. About a third of the room raises their hands.

1.3 Meeting guidelines (Patent and Anti-Trust)

Clamage directs group to the following websites without further comment:

- http://www.incits.org/pat_slides.pdf
- http://www.incits.org/inatrust.htm

1.4 Membership, voting rights, and procedures for the meeting

At Clamage's request Clark Nelson explains membership and voting rights.

1.5 Agenda review and approval

```
Motion: to accept agenda in N3469.
Moved by: Barry Hedquist
Seconded by: Nevin Liber

    Unanimous consent.
```

1.6 Approval of the minutes of the previous meeting

```
Motion: to accept minutes from previous meeting with editorial changes
(N3454 and N3455).
Moved by: Barry Hedquist
Seconded by: Clark Nelson
```

```
    Unanimous consent.
```

1.7 Editor's report, approval of draft

Stefanus Du Toit summarizes the editor's report from post-Portland (N3486). The only changes were those voted in at Portland along with a large number of editorial changes.

```
Motion: to accept N3485 as the working draft for Bristol.
Moved by: Jonathan Caves
Seconded by: Stefanus Du Toit

    Unanimous consent.
```

1.8 Liaison reports (including WG21 study groups)

WG14 Report:

Tom Plum directs everyone to the liaison report in the minutes from the pre-Bristol teleconference (N3621).

1.9 WG and SG progress reports and work plans for the week

Clamage relays comments from Sutter. Intent of this meeting is to produce a CD. SG chairs are asked to prioritize C++14 issues. The previous ballot for a document number for C++14 had errors and will be redone.

Nelson shows agenda for evening meetings. A starting time of 7:30 is agreed upon.

Clamage ask sub-group and study group chairs to give status report:

Core (CWG):

Mike Miller reports CWG will be prioritizing C++14 issues. CWG will process a couple of paper referred from EWG, some additional papers on wiki, and Walter Brown has a few papers that deal with editorial changes. CWG will also process anything that passes EWG. Beyond that CWG will do normal issue resolution.

Library (LWG) and Library Evolution (LEWG):

Alisdair Meredith reports that, as ever, the pot is overflowing with work. LWG has split out LEWG to increase throughput. LWG is looking at wording as it goes into standard documents. New proposals and significant extensions go to LEWG. Both groups have 20 papers. Beman Dawes has stepped forward as chair for LEWG. His initial task is to find a replacement. As Dawes is unable to attend this meeting Jeffrey Yasskin and Alan Talbot will be co-chairing LEWG this week

Kloepper asks how many of the 20 papers are targeted for C++14. Meredith says LWG needs to decide what will be considered.

Mike Spertus expresses confusion about which group to submit papers to. Meredith says make your best guess, if small addition send to LWG and larger extension got to LEWG. Chairs of group will make decision to get it to correct place. Like idea of fuzzy border in meeting which allows group to be more productive in meeting. Want turnover in attendance between groups.

Josuttis says it is important to put schedule on wiki. Meredith will come up with initial draft for agenda.

Evolution (EWG):

Stroustrup reports that goal is to decide what goes into C++14. There is an initial triage of papers Voutilainen put up on wiki. Miller asks if there will be additional papers to go through other that what is on the wiki. Stroustrup says yes, but not sure what yet. He adds that in the copious spare time EWG will discuss digit separators.

Study group 1 – Concurrency (SG1):

Hans Boehm reports group will be busy all week.

Study group 2 – Modules (SG2):

Doug Gregor reports there are no new papers, but will be meeting for half a day later on in the week.

Study group 3 – File System (SG3):

Meredith reports on behalf of Beman Dawes that the wording will be brought forward for vote on Friday. Chandler Carruth says he agrees with Dawes wording and that the wording addresses his security concerns.

Study group 4 – Networking (SG4):

Kloepper reports there are three papers and two late papers. Goals for this week are to come to consensus on IP address papers and bring byte order paper to vote on Friday.

Study group 5 – Transactional Memory (SG5):

Michael Wong reports that a TM group has been meeting twice a month since 2008. There are three papers to be considered at this meeting.

Study group 6 – Numerics (SG6):

Lawrence Crowl reports there are a few papers in the works. Group will spend half a day or a day meeting this week. Primary papers are on different numeric representation types.

Study group 7 – Reflection (SG7):

Carruth reports there is not much to discuss and SG7 would like to have one evening meeting.

Study group 8 – Concepts (SG8):

Matt Austern reports that static if has been dropped. SG8 is not planning to meet independently, but as part of EWG.

Nicolai Josuttis asks if goal is for concepts lite to be part of C++14. Austern says that will be decided. Stroustrup adds that will be the last thing to be decided.

Study group 9 – Ranges (SG9):

Marshall Clow reports group is currently suffering from an excess of vision. There are no papers. SG9 is trying to constrain scope. Nothing expected to go into C++14.

Study group 10 – Feature Test (SG10):

Nelson reports that SG10 will have inaugural meeting. Background is to define macros to determine what new features are implemented and which are not. Not targeting anything for C++14.

1.10 New business requiring actions by the committee
There is no new business.

2 Organize Working Groups and Study Groups, establish working procedures

3 WG and SG sessions

4 WG and SG sessions continue
Tuesday 16 April, 8:30am-5:30pm

5 WG and SG sessions continue
Wednesday 17 April, 8:30am-5:30pm

6 WG and SG sessions continue
Thursday 18 April, 8:30am-5:30pm

7 WG and SG sessions continue
Friday 19 April, 8:30am–noon

8 General session
Friday 19 April, 1:30pm–5:30pm

8.1 WG and SG status and progress reports
Reports take place along with straw votes in 8.2.

8.2 Presentation and discussion of proposals. Straw votes taken.
Clamage presents Miller's suggestion that discussion is limited to five minutes per issue with a vote to be taken at the end, or discussion differed to end of motions list. There is unanimous consent to this suggestion. Andy Sawyer volunteers to keep time.

Core motions
Miller presents CWG motions.

CWG motion 1 passes with unanimous consent.

CWG motion 2 passes with unanimous consent.

CWG motion 3 passes with unanimous consent.

CWG motion 4 passes with unanimous consent.

CWG motion 5 passes with unanimous consent.

CWG motion 6 passes with unanimous consent.

Vandevoorde asks if motion 7 just codifies existing practice. Miller says yes.

CWG motion 7 passes with unanimous consent.

In discussion for motion 8, Seymour asks what the difference with VLAs are and if we are injecting any C or C++ incompatibilities. Maurer says we do not support the full set of C VLA features. But we implement a compatible subset.

Meredith asks if common features use common syntax. Maurer says yes.

Stroustrup is on record as saying that VLAs are an abomination. This is VLAs as they should have been.

CWG motion 8 passes with unanimous consent.

CWG motion 9 passes with unanimous consent.

Nelson asks if motion 10 has been implemented. Merrill says something very similar in initial GCC wording. Nelson is ok that something like it has been implemented.

```
straw poll: CWG motion 10

          yes   no    abstain
PL22.16   18    0     2
WG21      unanimous consent
```

CWG motion 10 passes.

Meredith asks if motion 11 solves problem of copy constructing aggregates or is that separate issue. Miller says that is a separate issue.

CWG motion 11 passes with unanimous consent.

Nelson asks if motion 12 has been implemented. Merrill says yes.

CWG motion 12 passes with unanimous consent.

Miller discusses motion 13. Second part of paper has caused much discussion. Calling constexpr functions in. If a constexpr function can't mutate anything than it can just be marked const. The paper proposes removing that which can cause overload resolution issues where const is not explicitly written on constexpr functions. Another option that was considered was to reverse the normal default. So another keyword would be applied to constexpr functions to mark them as mutable. EWG did not prefer that, but core does not have an opinion.

Meredith asks how much of this has been implemented? Do we have ABI concerns? Richard Smith says there would be transition period. Meredith asks would this be the first C++14 proposal that would be ABI breakage? Spicer says ABI breakage is incorrect characterization—it requires a code change.

Vandevoorde says he would prefer to see it.

Carruth says it is uniquely nice code change that is backwards compatible and can automate and suggest.

Stroustrup says first half of this has been implemented and everyone agrees with it. Most discussion has to do with second part that everyone would have to say constexpr ... const. We should really get the first part.

Dos Reis says first part everyone wanted but second part leads to two options.

Nelson says that first part might be great, but putting it into C++14 would have the compiler do a bunch of loops. Want to see it in C++17.

Dos Reis says this came from users. They would like to see standard library functions in constexpr. Something C++ users really really want.

Stroustrup comments on pressure from users. I find myself saying go slowly.

Meredith says that if they are expected to make use of STL functions they are out of luck.

Smith says good reason to be coupled together is without the coupling there is inconsistency between user defined types.

Stroustrup asks if vote on both parts the only alternative we have?

Miller says we can move it as is. Move it as C++11 standard.

Time is up and discussion on motion 13 will resume after remaining motions have been discussed.

Miller discusses motion 14. CWG was not thrilled with this proposal. Straw poll was 9-3 against. Was selected by EWG and did not find technical issues with it. Deffering to EWG that this is proposal to bring forward.

Stroustrup summarizes EWG position. Crowl gave presentation of two dozen alternatives. This proposal was the least likely to cause problem. Issues are in corner cases. Stroustrup's personal recommendation was there was not going to be any better proposal.

Vandevoorde is very unhappy with proposal and was not in room when was discussed. This breaks not only hexadecimal but _3d.

Meredith asks what proposal is. Miller says separator is _ and suffix separator is .. when needed to disambiguate.

Doug says 1000_3d become 1,003.

Caves says he would like to have nothing instead of this. Spicer agrees that doing nothing is better than this.

Crowl says this is a persistent pain point so much so that ADA put it into the standard. Once we had UD literal we were on path to keep away from easy solution.

Time is up and discussion on motion 14 will resume after remaining motions have been discussed.

Boehm asks how motion 15 interacts with limited garbage collection of C++11? Crowl says all it would do is extend lifetime a little bit as first element is still live. Carruth says that this is always possible to do in source. Clarifying that optimizer can do it for you.

Halpern says since new is replaceable function thinks that replacement new will see fewer users. Miller says this is somewhat similar to copy elision.

Spertus says what he understands from discussions with compiler folks is that this defines existing practice.

Carruth says optimizers today regularly do that for C for malloc() and free(). This extends same behavior to C++ memory allocations. Some do a bit of this already but not as much as they could.

Stroustrup says this was nearly unanimous in EWG in Portland.

```
straw poll: CWG motion 15

         yes   no    abstain
PL22.16  17    0     5
WG21     unanimous consent
```

CWG motion 15 passes.

Nelson asks if motion 16 has been implemented. Dos Reis says yes. Miller says this is only for constexpr variable and was extended. Only constexpr case has been implemented.

```
straw poll: CWG motion 16

         yes   no    abstain
PL22.16  15    1     6
WG21     7     0     1
```

CWG motion 16 passes.

Miller says that CWG thinks motion 17 is solid at this point and wants to propose for C++14.

Garcia thinks that polymorphic lambdas are great idea and safest path is to be in TS. Specifically in concepts lite TS. May be some conflict with concepts lite and generic lambdas. Does not want generic lambdas to prevent concepts lite from being implemented.

Gregor says Sutton did not think there would be a problem.

Nelson asks if it has been fully implemented? Vali says yes. Gregor adds that code has been reviewed and looks good.

Stroustrup hopes there are no bad interactions, but will not oppose this feature.

Liber says this is most requested feature of all users I know.

```
straw poll: CWG motion 17

         yes   no    abstain
PL22.16  unanimous consent
```

```
WG21      7     0     1
```

CWG motion 17 passes.

On a final note Miller does not want CWG to be jammed up with papers at end of CD. Please get papers done early so we are not jammed up.

Library motions

Meredith gives working group report.

LWG motion 1 passes with unanimous consent.

LWG motion 2 passes with unanimous consent.

LWG motion 3 passes with unanimous consent.

LWG motion 4 passes with unanimous consent.

Meredith says motion 5 helps with unique_ptr. Yasskin says it saves a line or two in places. Crowl says he has wanted something like this for a while.

```
straw poll: LWG motion 5

         yes   no    abstain
PL22.16  14    1     4
WG21     unanimous consent
```

LWG motion 5 passes.

LWG motion 6 passes with unanimous consent.

LWG motion 7 passes with unanimous consent.

LWG motion 8 passes with unanimous consent.

Nelson has mixed feelings about motion 9. Significant thing is that it was new at this meeting. First meeting it has ever been seen. It is a good thing, but as principle against rushing things I am going to abstain.

Lavavej notes that it has been implemented.

```
straw poll: LWG motion 9

         yes   no    abstain
PL22.16  13    1     8
WG21     unanimous consent
```

LWG motion 9 passes.

Crowl asks if he can use motion 10 to parse CSV files. That would be a major technical achievement. Halpern says input side of this does not have flexibility to match both single and double quote.

LWG motion 10 passes with unanimous consent.

Coffee break at 15:26. Discussion resumes at 15:42.

Discussion begins for motion 11.

Halpern says the urgency is not such that it needs to be voted out now.

Meredith reports that LWG felt this was an urgent problem. More of an issue fix than a paper. Boehm says this does not limit any future extension. Only applies to streams that synchronize to stdio.

Maclaren says with current implementations this might work but with future implementations it might not.

Carruth says that version presented to SG1 was much different than this. Asks if stdio provides the same level of guarantees as in C? Crowl says that reference documents do not speak to this.

Orr asks if a number can get fragmented. Crowl says yes. If cout and stdio go to the same place will they overlap. Crowl says yes.

```
straw poll: LWG motion 11

           yes    no     abstain
PL22.16    11     2      10
WG21       6      0      3
```

LWG motion 11 passes.

Talbot finds it disappointing that s is being used to mean string instead of string_view in motion 12.

Smith says previous operator “” space. Cannot have it be a suffix. If you remove it it does not have to lexically be a keword. So you can have an ‘if’ of you want it.

Du Toit was there when LWG discussed this most recently. Not happy that LEWG went one way and LWG simply decided to go another way and that was considered consensus of some kind.

Nelson says there is no way this should be rushed into C++14.

Stroustrup said that s as string was a motivating example for user defined literals. Being told that we should not do what we have been planning to do for several years because of new idea.

```
straw poll: LWG motion 12

           yes    no     abstain
PL22.16    7      6      8
WG21       6      2      0
```

LWG motion 12 passes.

```
straw poll: LWG motion 13

         yes   no    abstain
PL22.16  3     9     9
WG21     6     2     0
```

Stroustrup does not consider that consensus and motion 13 is removed.

```
straw poll: LWG motion 14

         yes   no    abstain
PL22.16  16    2     5
WG21     unanimous consent
```

LWG motion 14 passes.

Lakos would like more discussion for motion 15 to understand the allocator behavior.

Spertus says there is a precondition. Lakos asks if it is the same as splice.

Austern has concern that the wording came in late and believes this would make a non-node based paper invalid.

Clark asks if this has been implemented. Talbot says no, well partly.

Carruth is worried that this will limit what can be done in the future. Astern says example structure is arrays of arrays. Crowl says he almost never writes a chained hash table. When he needs to write a hash table it is an array one.

```
straw poll: LWG motion 15

         yes   no    abstain
PL22.16  1     15    8
WG21     1     2     5
```

Stroustrup says there is no consensus and LWG motion 15 fails.

Spicer suggests to vote motion 16 in later.

Austern says this solve the problem, but only if you explicitly opt in. Wakely says people do really want this now. Lavavej notes that C++11 fixed this for lower bound, but not map lower bound.

Lakos notes that comparisons will change the type. We are trying to get a benefit that we would like all maps to have. Alternative is to generate a different vocabulary type.

```
straw poll: LWG motion 16

         yes   no    abstain
PL22.16  11    12    0
```

```
WG21      7     0     1
```

LWG motion 16 passes.

LWG motion 17 passes with unanimous consent.

LWG motion 18 passes with unanimous consent.

LWG motion 19 (edited to motion 20) is not a motion. Nelson suggests taking a straw poll.

```
straw poll: Move we ask Beman as the filesystem TS editor to turn N3545
into the formal working draft, with a view to filing a PDTS ballot at
the next meeting.

   unanimous consent.
```

Evolution motions:

Stroustrup presents.

```
straw poll: EWG motion 1

         yes   no    abstain
PL22.16  16    0     3
WG21     unanimous consent
```

EWG motion 1 passes.

Library evolution motions:

Yasskin presents. He says that Sutter advises only moving a TS when there exists a tentative working draft. Networking in the only TS to have one so motions 2 and 3 will be removed.

Meredith raises concerns that work item proposed by motion 1 is too small.

Kloepper replies that SG4 has been clear that the intention from the outset of SG4 in Kona has been to provide annual incremental releases. This position was further affirmed in our recommendation to LEWG when suggesting this motion to be brought forward. The belief of SG4 is that a regular annual release schedule will help bring more papers forward and also take the pressure off of trying to get something in before it is ready. The only opposition heard is the personal opinion of Meredith.

LEWG motion 1 passes with unanimous consent.

Concurrency motions:

Hans Boehm presents.

SG1 motion 1 passes with unanimous consent.

In discussion of motion 2 Wakely says the join is done by time waiting. Don't understand how to implement the proposed solution. Said only the non-timed waiting functions should join.

Nelson says this is just a bug fix and does not know why it must be rushed. Boehm agrees and removes the motion.

SG1 motion 3 passes with unanimous consent.

Vollmann confirms that Hinnant was involved in the current wording for upgrade_mutex.

SG1 motion 4 passes with unanimous consent.

Meredith is concerned that motion 5 is ABI breaking. Would preferred to wait until C++17 where that is the expectation. Halpern repeats that conclusion would be that result would be link errors and not compile errors which are almost as good. No silent failure. This is a serious problem and needs to be fixed.

Crowl points out that async is a template and the ABI is not the template but the implementation of the template. This is relatively weak as ABI breakages go.

Carruth emphasizes what Halpern said. At the moment you cannot use std::future to communicate between library APIs. Fixing this allows people to write libraries that use std::future. If we fail to fix this we lose our opportunity to fix this.

Nelson asks if it has been implemented. Carruth says yes. Gustafsson clarifies that waiting_future has not yet been implemented as it was invented two days ago.

```
straw poll: SG1 motion 5

           yes   no    abstain
PL22.16    15    0     7
WG21       7     0     1
```

SG1 motion 5 passes.

Vollmann says that he is strongly against rushing motion 6 at this meeting.

Lakos seem to recall that destructor can fail to join and throw in the destructor. We have noexcept except we can fail. The other suggestion is that we just terminate.

Halpern says that previous behavior is that if you destroy a thread that is not joined and not detached then you will terminate. This will terminate in strictly fewer cases.

```
straw poll: SG1 motion 6

           yes   no    abstain
PL22.16    14    2     6
WG21       5     1     2
```

Stroustrup declares consensus and SG1 motion 6 passes.

Study group reports:

Wong reports on SG5.

Crowl reports that SG6 work would best be handled by C and then incorporated by reference by C++.

SG7 Chandler reports did not have a chance to meet

SG8 was part of EWG and was reported on by Stroustrup.

Clow reports SG9.

Nelson reports on SG10.

Clamage asks for show of hands for who would like to participate in database study group. Seven committee members raise hands. Clamage will ask convener to create database study group.

Carruth requests break at 17:32. Clamage resumes meeting at 17:44.

Deferred motions:

Miller explains that motion 8 was passed assuming that more elaborate library facility dynarray would be passed.

Stroustrup asks if it is possible for LWG to revert decision to not include dynarray.

Meredith says this class may or may not allocate memory. It should never have to allocate memory, but it may have to. In the case it might have to we should provide an allocator.

Joly says does not need an allocator as it can allocate on a stack. This class by its self does not need an allocator. Both this and dynarray are neither of those.

Halpern has question of dynarray paper in current state. If it is missing something that we want to do with allocators as we can add it later. Is it in a consistent state right now?

Crowl says understanding is that with partial allocator injection it is in an inconsistent state. We should either go all the way forward or go back.

Meredith says this paper needs an allocator like a tuple needs an allocator.

Stroustrup says we have had problems with arrays for four years. We gave people an alternative to solve the most obvious problem with the std::array class. We give them the same thing as std::dynarray class. Would much rather have both.

Wakely says dynarray needs an allocator like an array needs an allocator.

Austern says when you say that all structs need an allocator is a certain programming style. Not everyone uses that style.

Carruth says the feature works very good without allocators. Concerned that entire plenary session is reworking a paper already gone over by LWG.

Meredith says not every struct needs an allocator, but every generic wrapper type needs an allocator.

Halpern has a procedural suggestion. Let's leave on formal motions. Vote on it before motion 8.

End of discussion timer sounds. Stroustrup says no we were supposed to get 10 minutes not 5. Room replies that the timer was 10 minutes.

Nelson suggests we slip this onto motions list and see how it turns out tomorrow.

Crowl says there is a paper that has allocators for elements on the LWG page and we will vote on that. N3662.

In discussing CWG motion 13 Stroustrup says we can get constexpr relaxations, or we can get nothing. We cannot redesign it here now. Vandevoorde they ask why it is mutable in lambdas.

Yasskin knew there was an ABI break. This might allow them to fix the ABI breakage between 98 and 11. Confirmed.

```
straw poll: CWG motion 13

         yes   no    abstain
PL22.16  11    2     5
WG21     5     1     2
```

Stroustrup declares consensus and CWG motion 13 passes.

Continuing discussion on CWG motion 14.

Crowl wrote substantial paper going through all detail and found the best solution we were going to get would be underscore and .. as disambiguation. Not ideal, but every proposal have had problems.

Vandevoorde says it is always an option to see where C goes. Always we have exceptions; it is painful to deal with.

Spicer thinks this will be a huge embarrassment for how you have to write user defined literals. Think we should put on backburner for a while.

Stroustrup is fully convinced that a large community for which this is important. Providing the feature is valuable. I don't think we will find anything better than this. If there was an obvious better solution we would have fund it by now.

Carruth never wants to explain to a user what 100_3d is, but with this I will.

Wong asks if there is a counter proposal that is better?

Spertus says the alternative is do nothing.

```
straw poll: CWG motion 14

         yes   no    abstain
PL22.16  5     13    1
WG21     6     1     1
```

Stroustrup declares consensus and CWG motion 14 passes.

There is concern about split vote. It is up to convener to determine consensus. The US is one national body.

LWG motion 12:

Loic asks is there a good candidate for string view? Meredith says sv is alternative string view and for string was str.

Stroustrup says we dropped this one for C++11 standard as we don't want to rush. Suddenly in the middle of the week.

Saariste if you want to use it for strings then it would be better in most cases to make it a string_view

Loic asks why you need a literal for string view? Meredith says because of type deduction.

Talbot finds Stroustrup's argument quite compelling.

Yasskin says LEWG says s suffix makes something a literal allocate memory and no longer be a literal.

Stroustrup says argument was known at the time s was chose.

Yasskin mentions user defined literals of s for string view was in early string_ref paper.

Orr asks apart from s are there any objections to paper?

Crowl just learned that there are two definitions for literal suffix s. Meredith says there is definition of why that works. Crowl says he is sure they do.

Nelson says it is a language feature in the library for no good reason.

Halpern thinks we are rushing, but str is a fine suffix for string.

```
straw poll: LWG motion 12

          yes   no    abstain
PL22.16   8     7     3
WG21      6     1     1
```

LWG motion 12 passes.

Stroustrup raises technical point if comments about speed are critical maybe we should have it as built-in.

Discussion for adding dynarray back as a motion:

Someone says that usage pattern is as members of structs. Way to get stack allocation inside of structs.

Meredith thinks of it like a tuple.

Maclaren says it will need an extra location of size inside dynarray.

Crowl says this was never an aggregate type.

Sawyer asks did LWG decide that was ready? Meredith says yes.

Sawyer asks does Crowl think this is ready? Crowl says he is not an allocator expert, but the rest is ready.

Stroustrup says the non-allocator part of this has been basically stable forever.

Giroux says this is not a dynamic array. At minimum this is not even named right to go into IS. Lakos agrees.

Crowl says it is an array that is sized at construction. Is called dynarray because it is not static. This can be in the heap if you put it there.

Lakos says if this could exist only in the stack that would relieve it of the allocator requirement. If it can live beyond a function call than it needs allocators.

Spicer: VLA sound like they have variable length but they dont.

Orr asks if allocator support is added at some later point, is it overloaded on constructors, or would it break? Austern says version of paper being projected has allocator support.

Meredith says there are two kinds of allocators. This paper allocates space for elements, but not for container itself.

```
straw poll: Add N3662 to formal motions page.

         yes    no     abstain
PL22.16  9      2      6
WG21     6      0      2
```

Clamage declares discussion finished for the day at 18:43.

9 WG and SG sessions continue

10 WG and SG sessions continue

Saturday 20 April, 8:30am-noon

11 Review of the meeting

11.1 WG21 motions

The room is polled and 21 voting members of PL22.16 and 8 WG21 national bodies are present.

```
Summary of motion results:

Core     result
CWG 1    carried
CWG 2    carried
CWG 3    carried
CWG 4    carried
CWG 5    carried
CWG 6    carried
CWG 7    carried
CWG 8    carried
CWG 9    carried
```

CWG 10	carried
CWG 11	carried
CWG 12	carried
CWG 13	carried
CWG 14	lost
CWG 15	carried
CWG 16	carried
CWG 17	carried

<u>Library</u>

LWG 1	carried
LWG 2	carried
LWG 3	carried
LWG 4	carried
LWG 5	carried
LWG 6	carried
LWG 7	carried
LWG 8	carried
LWG 9	carried
LWG 10	carried
LWG 11	lost
LWG 12	carried
~~LWG 13~~	removed
LWG 14	carried
~~LWG 15~~	removed
LWG 16	carried
LWG 17	carried
LWG 18	carried
LWG 19	carried
~~LWG 20~~	removed

<u>Evolution</u>

EWG 1	carried

<u>Library Evolution</u>

LEWG 1	carried
~~LEWG 2~~	removed
~~LEWG 3~~	removed

<u>Concurrency</u>

SG1 1	carried
~~SG1 2~~	removed
SG1 3	carried
SG1 4	carried
SG1 5	lost
SG1 6	lost

The following motions are moved by the chair:

Core motions:

CWG motion 1: Move we accept as Defect Reports all issues in "ready" status from N3539 except for 1464 and apply their proposed resolutions to the C++ working paper:
129 223 240 312 496 616 1013 1310 1318 1320 1328 1330 1374 1405 1411 1412 1413 1425 1435 1437 1442 1456 1472 1475 1476 1479 1481 1489 1495 1502 1503 1504 1506 1510 1511 1515 1516 1522 1527 1528 1532 1533 1535 1537 1538 1539 1541 1543 1544 1550 1553 1556 1557 1559 1560

	yes	no	abstain
PL 22.16	21	0	0
WG21	8	0	0

CWG motion 2: Move we apply the proposed resolution of issue 1464 from N3539 to the C++ working paper.

	yes	no	abstain
PL 22.16	21	0	0
WG21	8	0	0

CWG motion 3: Move we accept as Defect Reports all issues in "tentatively ready" status from N3539 except for 1484 and 1514 and apply their proposed resolutions to the C++ working paper:
977 1356 1462 1465 1471 1473 1477 1482 1487 1492 1494 1507 1563 1605

	yes	no	abstain
PL 22.16	21	0	0
WG21	8	0	0

CWG motion 4: Move we accept as Defect Reports the following issues from N3539 and apply their proposed resolutions to the C++ working paper:
903 1213 1358 1531

	yes	no	abstain
PL 22.16	21	0	0
WG21	8	0	0

CWG motion 5: Move we accept as a Defect Report issue 974 from N3539 and apply its proposed resolution to the C++ working paper.

	yes	no	abstain
PL 22.16	21	0	0
WG21	8	0	0

CWG motion 6: Move we apply N3472, "Binary Literals in the C++ Core Language," to the C++ working paper.

	yes	no	abstain
PL 22.16	21	0	0
WG21	8	0	0

CWG motion 7: Move we apply N3624, "Core Issue 1512: Pointer comparison vs qualification conversions (revision 3)," to the C++ working paper.

	yes	no	abstain
PL 22.16	21	0	0
WG21	8	0	0

CWG motion 8: Move we apply N3639, "Runtime-sized arrays with automatic storage duration (revision 5)," to the C++ working paper.

	yes	no	abstain
PL 22.16	16	0	5
WG21	8	0	0

CWG motion 9: Move we apply N3638, "Return type deduction for normal functions," to the C++ working paper.

	yes	no	abstain
PL 22.16	21	0	0
WG21	8	0	0

CWG motion 10: Move we apply N3648, "Wording Changes for Generalized Lambda-capture," to the C++ working paper.

	yes	no	abstain
PL 22.16	17	0	4
WG21	8	0	0

CWG motion 11: Move we apply N3653, "Member initializers and aggregates," to the C++ working paper.

	yes	no	abstain
PL 22.16	21	0	0
WG21	8	0	0

CWG motion 12: Move we apply N3667, "Drafting for Core 1402," to the C++ working paper.

	yes	no	abstain
PL 22.16	21	0	0
WG21	8	0	0

CWG motion 13: Move we apply N3652, "Relaxing constraints on constexpr functions" and "constexpr member functions and implicit const," to the C++ working paper.

	yes	no	abstain
PL 22.16	14	1	5
WG21	8	0	0

CWG motion 14: Move we apply N3661, "Digit Separators," to the C++ working paper.

	yes	no	abstain
PL 22.16	1	17	2
WG21	2	3	3

CWG motion 15: Move we apply N3664, "Clarifying Memory Allocation," to the C++ working paper.

	yes	no	abstain
PL 22.16	21	0	0
WG21	8	0	0

CWG motion 16: Move we apply N3651, "Variable Templates (Revision 1)," to the C++ working paper.

	yes	no	abstain
PL 22.16	14	1	5
WG21	8	0	0

CWG motion 17: Move we apply N3649, "Generic (Polymorphic) Lambda Expressions (Revision 3) ," to the C++ working paper.

	yes	no	abstain
PL 22.16	21	0	0
WG21	7	0	1

Library motions:

LWG motion 1: Move we apply the resolutions of all issues in "Ready" and "Tentatively Ready" status from N3438 to the C++ Working Paper:

2091 Misplaced effect in m.try_lock_for()
2092 Vague Wording for condition_variable_any
2093 Throws clause of condition_variable::wait with predicate
2145 error_category default constructor
2147 Unclear hint type in Allocator's allocate function
2163 nth_element requires inconsistent post-conditions
2169 Missing reset() requirements in unique_ptr specialization
2172 Does atomic_compare_exchange_* accept v == nullptr arguments?
2080 Specify when once_flag becomes invalid
2109 Incorrect requirements for hash specializations
2144 Missing noexcept specification in type_index
2174 wstring_convert::converted() should be noexcept
2175 string_convert and wbuffer_convert validity
2177 Requirements on Copy/MoveInsertable
2187 vector<bool> is missing emplace and emplace_back member functions
2197 Specification of is_[un]signed unclear for non-arithmetic types
2200 Data race avoidance for all containers, not only for sequences

2209 assign() overspecified for sequence containers
2211 Replace ambiguous use of "Allocator" in container requirements
2222 Inconsistency in description of forward_list::splice_after single element overload
2225 Unrealistic header inclusion checks required
2231 DR 704 removes complexity guarantee for clear()

	yes	no	abstain
PL 22.16	21	0	0
WG21	8	0	0

LWG motion 2: Move we apply the resolutions of all issues in paper N3673, moved from "Review" to "Ready" during this meeting.

2094 duration conversion overflow shouldn't participate in overload resolution
2122 merge() stability for lists versus forward lists
2128 Absence of global functions cbegin/cend
2148 Hashing enums should be supported directly by std::hash
2149 Concerns about 20.8/5
2162 allocator_traits::max_size missing noexcept
2176 Special members for wstring_convert and wbuffer_convert
2196 Specification of is_*[copy/move]_[constructible/assignable] unclear for non-referencable types
2203 scoped_allocator_adaptor uses wrong argument types for piecewise construction
2207 basic_string::at should not have a Requires clause
2210 Missing allocator-extended constructor for allocator-aware containers
2229 Standard code conversion facets underspecified
2235 Undefined behavior without proper requirements on basic_string constructors

	yes	no	abstain
PL 22.16	21	0	0
WG21	8	0	0

LWG motion 3: Move we apply N3545, An Incremental Improvement to integral_constant, to the C++ Working Paper.

	yes	no	abstain
PL 22.16	21	0	0
WG21	8	0	0

LWG motion 4: Move we apply N3644, Null Forward Iterators, to the C++ Working Paper.

	yes	no	abstain
PL 22.16	21	0	0
WG21	8	0	0

LWG motion 5: Move we apply N3668, std::exchange(), to the C++ Working Paper.

	yes	no	abstain
PL 22.16	12	1	6
WG21	8	0	0

LWG motion 6: Move we apply N3658, Compile-time integer sequences, to the C++ Working Paper.

	yes	no	abstain
PL 22.16	21	0	0
WG21	8	0	0

LWG motion 7: Move we apply N3670, Addressing Tuples by Type, to the C++ Working Paper.

	yes	no	abstain
PL 22.16	21	0	0
WG21	8	0	0

LWG motion 8: Move we apply N3671, Making non-modifying sequence operations more robust, to the C++ Working Paper.

	yes	no	abstain
PL 22.16	21	0	0
WG21	8	0	0

LWG motion 9: Move we apply N3656, make_unique, to the C++ Working Paper.

	yes	no	abstain
PL 22.16	16	0	5
WG21	8	0	0

LWG motion 10: Move we apply N3654, Quoted Strings, to the C++ Working Paper.

	yes	no	abstain
PL 22.16	21	0	0
WG21	8	0	0

LWG motion 11: Move we apply N3665, Uninterleaved String Output Streaming, to the C++ Working Paper.

	yes	no	abstain
PL 22.16	3	11	7
WG21	3	2	3

LWG motion 12: Move we apply N3642, User-defined Literals, to the C++ Working Paper.

	yes	no	abstain
PL 22.16	8	5	8
WG21	5	1	1

~~LWG motion 13:~~ Move we apply N3660, User-defined complex Literals, to the C++ Working Paper.

[motion removed]

LWG motion 14: Move we apply N3655 (excluding part 4), Transformation Traits Redux, to the C++ Working Paper.

	yes	no	abstain
PL 22.16	14	1	5
WG21	8	0	0

~~LWG motion 15:~~ Move we apply N3645, Splicing Maps and Sets, to the C++ Working Paper.

[motion removed]

LWG motion 16: Move we apply N3657, Adding heterogeneous comparison lookup to associative containers, to the C++ Working Paper.

	yes	no	abstain
PL 22.16	21	0	0
WG21	8	0	0

LWG motion 17: Move we apply N3672, A proposal to add a utility class to represent optional objects, to the C++ Working Paper.

	yes	no	abstain
PL 22.16	21	0	0
WG21	8	0	0

LWG motion 18: Move we apply N3669, Fixing constexpr member functions without const, to the C++ Working Paper.

	yes	no	abstain
PL 22.16	21	0	0
WG21	8	0	0

LWG motion 19: Move we apply N3662, C++ Dynamic Arrays (dynarray), to the C++ Working Paper.

	yes	no	abstain
PL 22.16	14	0	7

WG21	6	1	1

~~LWG motion 20:~~ Move we ask Beman as the filesystem TS editor to turn N3545 into the formal working draft, with a view to filing a PDTS ballot at the next meeting.

[removed as this does not require a motion]

Evolution motions:

EWG motion 1: Move to direct the Convener to request a New Work Item for a Technical Specification on Concepts Lite.

	yes	no	abstain
PL 22.16	21	0	0
WG21	8	0	0

Library evolution motions:

LEWG motion 1: Move to direct the Convener to request a New Work Item for a Technical Specification on Networking.

	yes	no	abstain
PL 22.16	21	0	0
WG21	8	0	0

~~LEWG motion 2~~: Move to direct the Convener to request a New Work Item for a Technical Specification on Concurrency and Parallelism.

[motion removed]

~~LEWG motion 3:~~ Move to direct the Convener to request a New Work Item for a Technical Specification on Library Extensions 2.

[motion removed]

Concurrency motions:

SG1 motion 1: Move we apply the resolutions of the following issues in "Review" status from N3522 to the C++ Working Paper:

LWG2098 (promise throws clauses)
LWG2130 (missing ordering constraints for fences)
LWG2138 (atomic_flag::clear ordering constraints)
LWG2140 (notify_all_at_thread_exit synchronization)
LWG2190 (ordering of condition variable operations, reflects Posix discussion)

	yes	no	abstain
PL 22.16	21	0	0
WG21	8	0	0

~~SG1 motion 2:~~ Move we apply the resolution of issue LWG2100 (adjust async wording for timeouts) in "Review" status from N3522 to the C++ Working Paper, after deleting "non-timed" from the resolution.

[motion removed]

SG1 motion 3: Move we apply the resolution of issue LWG2185 (missing throws clause for future timed wait) in "Review" status from N3522 to the C++ Working Paper. If N3637 (Motion 5) is incorporated into the working paper, make the same changes to waiting_future.

	yes	no	abstain
PL 22.16	21	0	0
WG21	8	0	0

SG1 motion 4: Move that we apply N3659, Shared Locking in C++, to the C++ Working Paper.

	yes	no	abstain
PL 22.16	21	0	0
WG21	8	0	0

SG1 motion 5: Move we apply N3637, async and ~future, to the C++ Working Paper.

	yes	no	abstain
PL 22.16	8	3	10
WG21	4	2	2

SG1 motion 6: Move we apply N3636, ~thread Should Join, to the C++ Working Paper.

	yes	no	abstain
PL 22.16	11	2	8
WG21	5	2	1

Other motions:

Other motion 1: Move to appoint an editing committee composed of Richard Smith, Mike Miller, Daniel Krugler, and Alisdair Meredith to approve the correctness of the working draft as modified by the motions approved at the Bristol meeting, and to direct the Convener to forward the approved updated working draft to SC22 for CD ballot.

	yes	no	abstain
PL 22.16	21	0	0
WG21	8	0	0

11.2 PL22.16 motions

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

11.4 Issues delayed until today

Clamage asks if there are any issues delayed until today. No response.

12 Plans for the future

12.1 Next and following meetings

Nevin Liber talks about upcoming Chicago meeting.

Clamage takes a poll for who would be interested in attending an in person meeting between Chicago and Rapperswil. 19 say they would be.

Nelson moves to thank the host Roger Orr. Applause ensues.

Clow relays thanks for the live radio feed from the meetings.

Carruth thanks everyone who took notes this week.

12.2 Mailings

Nelson says the post-Bristol mailing deadline is 3 May and pre-Chicago deadline is 30 August. Nelson also directs everyone to look at SD-1 in the mailing as it is useful.

13 Adjournment

```
Motion: to adjourn
Moved by: Clark Nelson
Seconded by: Chandler Carruth

Unanimous consent.
```

Adjourned Saturday 20 April at 2:47 p.m. BST.

14 Attendance

14.1 PL22.16 members

All listed organizations have active voting status. Principal representatives are designated with *.

Organization	Representative	M	T	W	R	F	S
Apple	Doug Gregor	✓	✓	✓	✓	✓	✓
Bloomberg	John Lakos*	✓	✓	✓	✓	✓	✓
Bloomberg	Alisdair Meredith	✓	✓	✓	✓	✓	✓
Bloomberg	Dietmar Kühl	✓	✓	✓	✓	✓	✓
Cisco Systems	Martin Sebor*	✓	✓	✓	✓	✓	
Cisco Systems	Ismail Pazarbasi	✓	✓	✓	✓	✓	
Dinkumware	P.J. Plauger*	✓	✓	✓	✓	✓	
Dinkumware	Tana Plauger	✓	✓			✓	
Dinkumware	Margaret Trimble	✓					
DRW Holdings	Nevin Liber*	✓	✓	✓	✓	✓	✓
Edison Design Group	Daveed Vandevoorde	✓	✓	✓	✓	✓	✓
Edison Design Group	Jens Maurer	✓	✓	✓	✓	✓	✓
Edison Design Group	John H. Spicer	✓	✓	✓	✓	✓	✓
Edison Design Group	William M. Miller	✓	✓	✓	✓	✓	✓
Embarcadero Technologies	Dawn Perchik*	✓	✓	✓	✓	✓	✓
Gimpel Software	James Widman	✓	✓	✓	✓	✓	
Google	Lawrence Crowl*	✓	✓	✓	✓	✓	✓
Google	James Dennett	✓	✓	✓	✓	✓	
Google	JC van Winkel	✓	✓	✓	✓	✓	✓
Google	Matthew Austern	✓	✓	✓	✓	✓	
Google	Chandler Carruth	✓	✓	✓	✓	✓	✓
Google	Jeffery Yasskin	✓	✓	✓	✓	✓	✓
Google	Richard Smith	✓	✓	✓	✓	✓	✓
Hewlett-Packard Development	Hans Boehm	✓	✓	✓	✓	✓	✓
IBM	Michael Wong*	✓	✓	✓	✓	✓	✓
IBM	Maged Michael		✓	✓	✓		
Intel	Clark Nelson*	✓	✓	✓	✓	✓	✓
Intel	Pablo Halpern	✓	✓	✓	✓	✓	
Intel	Robert Geva	✓	✓	✓	✓	✓	
Intel	Stefanus Du Toit	✓	✓	✓	✓	✓	✓
Microsoft	Jonathan Caves*	✓	✓	✓	✓	✓	✓
Microsoft	Herb Sutter	✓	✓	✓	✓		
Microsoft	Niklas Gustafsson	✓	✓	✓	✓	✓	
Microsoft	Stephan Lavavej	✓	✓	✓	✓	✓	

NVidia	Olivier Giroux	✓	✓	✓	✓	✓	✓
NVidia	Jaydeep Marathe		✓				
NVidia	Jared Hoberock		✓	✓	✓		
NVidia	Michael Garland		✓	✓	✓		
Oracle	Paolo Carlini*	✓	✓	✓	✓	✓	✓
Oracle	Stephen D. Clamage	✓	✓	✓	✓	✓	✓
Oracle	Darryl Gove	✓	✓	✓	✓		
Oracle	Victor Luchangco	✓	✓	✓			
Perennial	Barry Hedquist*	✓	✓	✓	✓	✓	✓
Plum Hall	Thomas Plum*	✓	✓	✓	✓	✓	
Programming Research Group	Richard Corden*	✓	✓	✓	✓	✓	
Programming Research Group	Christof Meerwald	✓	✓	✓	✓	✓	
Qualcomm	Marshall Clow*	✓	✓	✓	✓	✓	✓
Red Hat	Jason Merrill*	✓	✓	✓	✓	✓	
Red Hat	Benjamin Kosnik	✓	✓	✓	✓	✓	✓
Red Hat	Torvald Riegel	✓	✓	✓	✓	✓	
Red Hat	Matt Newsome	✓			✓		
Riverbed	Kyle Kloepper*	✓	✓	✓	✓	✓	✓
Riverbed	Neal Meyer	✓	✓	✓	✓	✓	
Seymour	Bill Seymour*	✓	✓	✓	✓	✓	✓
Symantec	Mike Spertus*	✓	✓	✓	✓	✓	✓
Symantec	Dinka Ranns	✓	✓	✓	✓	✓	
Texas A&M University	Bjarne Stroustrup*	✓	✓		✓	✓	
Texas A&M University	Andrew Sutton	✓	✓	✓	✓	✓	

14.2 PL22.16 non-members

Organization	Representative	M	T	W	R	F	S
BLDL, University of Bergen	Magne Haveraaen	✓	✓	✓	✓	✓	
Boost	Paul Bristow			✓			
Bruker Daltonics	Daniel Krügler	✓	✓	✓	✓	✓	
BSI	Alan Lenton	✓	✓	✓	✓		
CERN	Axel Naumann	✓	✓	✓	✓	✓	
Frankfurt Inst. for Adv. Studies	Matthias Kilpeläinen	✓	✓	✓	✓	✓	
GETCO	Jonathan Wakely	✓	✓	✓	✓	✓	
HSR	Peter Sommerlad	✓					
Inform GmbH	Frank Birbacher	✓	✓	✓	✓	✓	✓
Integrable Solutions	Gabriel Dos Reis	✓	✓	✓	✓	✓	
Jeppesen Systems	Ove Svensson	✓					

LTK Engineering	Alan Talbot	✓	✓	✓	✓	✓	✓
Metascale	Mathias Gaunard		✓				
Oxyware	Hubert Matthews	✓					
Poco	Alex Fabijanic	✓	✓	✓			
QCS	Sam Saariste	✓	✓	✓	✓	✓	✓
Research in Motion	Tony Van Eerd		✓	✓	✓		
SDL	Jamie Lewis	✓	✓				
SDL	Harvey Shields	✓	✓				
Siemens	Tobias Schuele	✓	✓	✓	✓		
SMSS	Francis W. Glassborow	✓	✓				
Technische Universität München	Thomas Neumann		✓	✓			
think-cell Software	Arno Schödl	✓	✓	✓			
think-cell Software	Fabio Fracassi	✓	✓	✓	✓	✓	✓
think-cell Software	Valentin Ziegler	✓	✓	✓	✓		
University Carlos III	J. Daniel Garcia	✓	✓	✓	✓	✓	✓
University of Cambridge	Nick Maclaren	✓	✓	✓	✓	✓	
University of Nice	Jean-Paul Rigault	✓	✓	✓	✓	✓	
University of York	Pattabi Raman	✓	✓	✓	✓	✓	
Vollmann Engineering	Detlef Vollmann	✓	✓	✓	✓	✓	✓
	Nicolai Josuttis	✓		✓	✓		
	Ville Voutilainen	✓	✓	✓	✓	✓	✓
	Andy Sawyer	✓	✓	✓	✓	✓	✓
	Roger Orr	✓	✓	✓	✓	✓	✓
	Cassio Neri	✓	✓	✓	✓	✓	✓
	Faisal Vali	✓	✓	✓	✓	✓	
CAST	Loïc Joly	✓	✓	✓	✓	✓	
	Marco Poletti	✓	✓		✓	✓	
	Mark Boyall	✓	✓	✓	✓	✓	
	Niels Dekker	✓	✓	✓	✓	✓	
SimuNova & Tu Dresden	Peter Gottschling	✓	✓	✓	✓	✓	✓
	Daniel Phifer	✓	✓	✓	✓	✓	
	Attila Nagy	✓					
Strusoft	Pal Balog	✓	✓				
	Vincent Reverdy	✓	✓	✓	✓	✓	
BSI	Jamie Allsop	✓	✓	✓	✓	✓	✓
Just Software Solutions LTD	Anthony Williams	✓	✓				
	Lois Goldthwaite		✓	✓			

14.3 JTC1/SC22/WG21 technical experts

Heads of delegation designated with *.

Affiliation	Representative
Netherlands	JC van Winkel*
Canada	Michael Wong*
Canada	Stefanus Du Toit
United States	Barry Hedquist*
Germany	Daniel Krügler
Germany	Nicolai Josuttis
Germany	Peter Gottschling*
Germany	Daniel Phifer
Switzerland	Axel Naumann
Switzerland	Peter Sommerlad
Switzerland	Detlef Vollmann*
Spain	J. Daniel Garcia*
United Kingdom	Nick Maclaren
United Kingdom	Andy Sawyer
United Kingdom	Roger Orr*
United Kingdom	Jamie Allsop
United Kingdom	Anthony Williams
United Kingdom	Lois Goldthwaite
Finland	Ville Voutilainen*
Hungary	Attila Nagy
Hungary	Pal Balog