Document Number: N3529
Date: 2013-02-14
Authors: Michael Wong
Project: Programming Language C++, EWG, SG5 Transactional Memory
Reply to: Michael Wong <michaelw@ca.ibm.com>

SG5: Transactional Memory (TM) Meeting Minutes 2012/10/30-2013/02/04

Contents

Minutes for 2012/10/30 SG5 Conference Call

On Wed, 2012-10-24 at 23:17 -0700, Justin Gottschlich wrote:
> The current secretary rota list is:
>
Tatiana, Michael W, Victor, Justin, Mike S, Maged, Michael S, Paul M,
Mark, Torvald, Hans
>
> 1.1 Roll call of participants

Justin Gottschlich (JS), Mike Spear (MS), Mark Moir (MM), Michael Wong (MW), Torvald Riegel (TR), Hans Boehm (HB), Tatiana Shpeisman (TS), Dave Abrahams (DA), Victor Luchangco (VL)

> 1.2 Adopt agenda

JG / MW

> 1.3 Approve minutes from previous meeting

JG / MW

> 1.4 Review action items from previous meeting
> 1.4.1 Justin, Mark, and Maged to work on legal issues to "release" the
> specification to ISO.

MW forwarded a form that the Intel/IBM/Oracle lawyers could sign.
Carry-over action for Intel + Oracle.
>
> 2. Main issues
>
> 2.1 Discuss official SG5 report presented by Michael Wong at Portland
> C++ Standard Committee meeting.

MW: Very good meeting. Presented v1.1 spec. Many people now back the work in the sense that it is the right time to work on this. MW thinks sending a fully worded proposal in time for Bristol would be good. Want to avoid fighting for attention and review-time with other proposals that might be submitted soon.
Still some opponents to TM (e.g., in the Swiss delegation).

HB: Liked that Herb Sutter encouraged national delegates to support new projects. Thinks we have a reasonable chance at TM being voted in as at least a project that can be worked on.

AI: need clarification on how many national bodies need to support a project: 3 or 5?

> 2.2 Summary of technical discussions at C++ Standard Committee
> meeting.
> 2.2.1 Justin's spec overview talk

JG: Overview talk wasn't meant to be controversial. A question that developed for JG while preparing the talk was why we'd actually need relaxed transactions. Talk has an example why we want relaxed txns.

AI JG: Send out slides of the talk.

> 2.2.2 Mark's safe-by-default talk

MM: He sent a summary and slides out. [Summarizes the slides]
Proposal transformed from safe-by-default as a convenience mechanism to safe-by-default--by-default. Trade-off is convenience vs. potential code bloat, potential compile time overheads, and potential increase of size of binaries.
Got positive response, but no slam dunk yet because of overheads. But people concerned about code bloat can use transaction_unsafe to prevent the problem.
MM's conclusion: Try to flesh out a proposal. Fortunately, many of the distractions/obstacles we had in the past seem to fall away.

DA: Were there implementers in the room when MM discussed it?

TS: Anything that increases code size will get negative reactions.

MM: How much is really transactional? Is the code-size increase substantial?

MS: If several methods in a class, how do you do generate transactional versions selectively with a compiler flag?

MM: Compiler-flag approach would be broad brush. Doesn't replace transaction_[un]safe.

TS: People that don't care about TM must not have their binaries' sizes increase.

MM: Programmers would have to enable TM anyway.

TR/VL: If TM gets part of the standard, enabling it specially would be awkward.

DA: If in standard, it could be conditionally supported. Compiler with and without flag would be both conformant.

> 2.2.3 Mark and Dave's proposal for atomic transactions and
> exceptions

[Some discussion about whether it makes sense to discuss this now.]

MM: [Summarizes his email with title "Atomic transactions should be cancelable".] Says DA pointed out that exceptions do what transaction_cancel does, but in a non-TM-specific way. Could potentially avoid the additional may_cancel_outer attributes etc. because that can be done with exceptions. Would simplify the proposal.

JG: A lot of this discussion started by probing what people think about cancel. JG had reservations about cancel and the attributes. JG is now strongly in favor of Mark and Dave's approach.

MM: An important point is the difference between ability to cancel and the language feature to cancel.

MS: Any consequence for the equivalence of C and C++? Nonissue?

MM: Might not be a nonissue. Need to think about that.

DA: Might need a special language feature for C that allows us to cancel it.

> 2.3 Discussion of our declaration to submit a fully worded Technical
> Specification proposal for the April 15, 2013, C++ Standard Meeting in
> Bristol, UK.
>
> 2.4 - Legal issues discussion
>
> 2.5 - Time of teleconference, extending it, changing it

MM sent out a poll where people can note when they would be available.

AI everyone: Fill out the poll.

Minutes for 2012/11/12 SG5 conference call

Participants:

Justin Gottschlich
Torvald Riegel
Mark Moir
Victor Luchangco
Michael Scott
Jens Maurer
Mike Spear
Maged Michael

Secretary Rota for next meeting:

Tatiana, Michael W, Justin, Mike Spear, Maged, Michael Scott, Paul M, Mark, Hans, Torvald, Victor

Revised agenda adopted.

Previous minutes approved.

Review action items from previous meeting:

> Everyone to fill out Mark's schedule poll

DONE (and we already have a new meeting time this week)

> Justin to send out slides from his talk at C++ Standards Committee Meeting

DONE

> Michael Wong to get clarification on how many national bodies are needed for Technical Specification.

Carry-over

> Michael Wong to send ISO legal document to Justin

DONE

> Justin, Mark and Maged to work legal issues to "release" the specification.

Carry-over.

Main issues

Reordered agenda items to accommodate Maged's expected late arrival. We will need to discuss at least 2.1 again when Tatiana is present, as she has some concerns about it.

(My apologies for elisions in the transcript--I couldn't keep up and there were some sections I just didn't manage to capture at all.)

2.2. Discuss whether scoped locking with an elidable global lock is equivalent to relaxed transactions *in a world with no atomic transactions*

Mark:

Overview of issues described in email he sent.

Confusion about atomic and relaxed transactions. Some of the confusion is because we are using the same name for different things.

Question: What are relaxed transactions good for?
At C++ meeting, Lawrence Crowl said: They are for integrating transactional and nontransactional code. Thus, they are of no use without atomic transactions.

Initial agreement, but now backing off: There is some motivation for relaxed transactions alone, but these should be separated from atomic transactions. So it is useful to think of relaxed transactions in a world without atomic transactions.

Michael Scott:

Distinguish between uses of relaxed transactions
1. to mediate between transactional and nontransactional code
2. other uses of relaxed transactions.
By considering a world without atomic transactions, we can separate these concerns.

Mark:

Yes. We are not looking to get rid of atomic transactions, but to understand the role of relaxed transactions independent of atomic transactions.

After much pondering and false starts, made a proposal (sent on Nov 11), which turns out to be almost exactly what Michael Scott said previously.

Semantiically speaking, we are just talking about a global lock, but scoped aspect of it fits well

with transactions.

Tatiana has some reservations, sent in email. We should continue this discussion when she's present.

Maged joins the call, so we switch to:

2.1. Continue discussion on Mark and Dave's proposal for atomic transactions and exceptions

Mark (recapping this proposal):

I believe atomic transactions must be cancellable.
Sent out an example to make this point.
Separate issue from whether we support explicit programmer-specifiable cancelability.

Michael Scott:
Clarification: When we say cancel, we mean "aborting" or "rolling back", not necessarily allowing programmer to do so.

Mark:

Yes, but "abort" or "rollback" have problematic connotations we want to avoid, hence our use of "cancel". But we need to distinguish the ability for a TM implementation to cancel internally, and the ability for a programmer to explicit cause a transaction to be cancelled.

However, if we adopt this proposal, an uncaught exception escaping a transaction will cancel the transaction, so it provides programmers some mechanism to trigger cancellation.

Maged: Atomic transactions are not *worthless* without cancelability.

Victor: Can you give a concrete example of such a use?

Maged: Mark has said before that atomic transactions are useful without cancel.

Mark: To clarify: in that previous statement, I meant "explicit" cancel, not whether an implementation can do so under the covers.

One feature of this proposal is that we can get rid of a lot of new keywords and features--for example, transaction_cancel, cancel_outer, may_cancel_outer, etc.--because programmers can use exceptions instead of an explicit cancel mechanism.

Michael Scott:

We could have useful atomic transactions without cancelability if we can prove that they won't

need to be canceled, but this is likely to be harder than building in support for cancelling.

Jens:
With atomic transactions, there are limitations that allow us to implement them. Why not just have exceptions be such a thing?

Torvald:

If you already do all the analysis to prove that there are no problems within a transaction, then it's relatively easy to add support for actually cancelling.

Michael Scott:

We can't check that a transaction can't throw an exception.

Maged:

You can refactor code to catch the exceptions.
[Victor: I'm not sure I captured Maged's point correctly.]

Mike Spear:

What do we mean by exceptions? What about segmentation fault/divide-by-zero?

Jens:

Segmentation faults, etc. are undefined behavior.
We mean using the C++ exception mechanism.
What do we do when these exceptions cross transaction boundaries?

Justin:

Was on Maged's side for a long time, and understands Maged's points .
However, there are a number of advantages to Mark and Dave's proposal.
- reduction of features: transaction_cancel, cancel_outer, etc.

Maged:

I want to reduce the linguistic complexity as well, but concerns remain.
Why not leave the behavior undefined for "a short time".

Torvald:

[Victor: Sorry, I failed to capture this.]

Mark:

So programmers needs to implement their own exception handling mechanism to use within transactions, to avoid this problem?

Jens:

No, Maged is concerned about the situation when there is no multithreading.

Michael:

We're conflating two items:
If we guarantee that we can cancel, then programmers can use it even in sequential code.
[Victor: other item is using transactions to provide atomicity for multithreaded code.]

Maged:

Yes, the conflation of speculation with atomicity is the concern.
I would happy to explore whether these can be disentangled.

Jens:

I don't think they can be disentangled.
1. In the multithreading case, where you want the atomicity, you also have some speculation with it.
2. Rather probable that user-hand-written code will be faster than generic compiler undo of transaction.
(Mark: need to take into account the cost of instrumentation to allow undo.)
This was not my point. My point was that there are costs, both in terms of programming effort and complexity, and in terms of performance (duration of transaction, size of write set) for being *prepared* to undo a transaction using "user-hand-written code". These costs apply even if the circumstances under which the transaction needs to be undone *never* occurs.

Thus, narrow discussions about the performance of the case in which undo is necessary are not meaningful because they ignore the cost imposed on the case in which undo is not necessary, as well as the probability of each case occurring, not to mention that additional programmer effort required.

Cheers

Mark
3. Code in transaction is already limited.

Michael Scott:

Speculation may be... [Victor: sorry, I missed the rest of this]

Torvald:

Atomic transaction used for concurrency control can be done without speculation.

Michael:

The real issue before us is whether the ability of transaction to be cancelled is essential to the semantics of transactions that we want.
Is anyone proposing we could provide something we are willing to call atomic transactions but could not cancel and therefore cannot compose even sequential code?

Torvald:

I support the case of failure atomicity, but I'm concerned about changing the spec.

Michael:

Can you send mail that tries to make this case?

[Victor: There was a little bit more discussion that I failed to capture. One thing I remember is that Mark wanted to avoid trying again to precisely define speculation, as we've spent a lot of time in the past doing so without much to show for it.]

2.3. Discuss our declaration to submit a fully worded Technical Specification proposal at C++ Standard Meeting in Bristol (April 15, 2013).

Justin:

One of last things at Portland meeting: we will come to Bristol with fully worded proposal to be presented at the C++ meeting.

As a technical specification, we have more flexibility to move at our own pace, either faster or slower than ordinary process.

Sense from many C++ committee members: only one major feature will be accepted 2017. Our TM feature seems further along than many of the other proposed features, so if we push on this, that may be us. (Not sure if we want to do this, but we should think about it.)

Victor:

Is there a technical meaning of "fully worded technical specification proposal"?

Jens:

"Fully worded" means piece of paper that describes the changes to the C++ specification in detail. For example, "In Section 3.1, change paragraph 2 to say...".

No real precedent for this with core language: so far this has been done with libraries, which have less interaction with core language. Also, spec is changing, so it's a moving target.

Justin:

To have something for Bristol, we need consensus on current open proposals:
- safe-by-default
- Mark and Dave's proposal on exceptions and transactions
- renaming of relaxed transactions

We need to "close the door" on new features to add

Mark:

If things go the way I hope, the new spec will be shorter than the current one. But "fully worded" will be difficult, especially if memory model is changing.

Jens:

Memory model isn't changing, but other things might, more in sectioning, etc., so a "fully worded" proposal, mentioning sections, etc., may become invalid.

Mark:

We should focus on getting good proposal, rather than focus on "fully worded".

Michael:

Yes, I'm in favor of getting this incorporated more quickly if possible, but we should make sure we get it right.

Victor:

Not sure a fully worded proposal by Bristol meeting is reasonable goal. It requires being a lot more familiar with details of C++ spec, not just our proposal.

Jens:

Willing to help with "fully worded" proposal, but can't commit.

Victor:

[Recaps the issues to be resolved mentioned above]

Any others?

Justin:

Perhaps tm_waiver

Victor:

tm_waiver is quite different: we want it, but we may not want to specify it.

Justin:

We need to wrap up (5 mins over already). Next meeting is Nov 26.

2.4 Legal issues discussion

We didn't get to this.

---
3. Other business

4. Review

5. Closing process

We skipped the three above parts due to lack of time, except that we did manage to adjourn. :-)

The only new specific action item I recall is that Michael Scott asked Torvald if he could send mail that tries to make the case that we can have an implementation of atomic transactions without any speculation.

Next meeting is on Nov 26.

Minutes of 2012/11/26 SG5 Conference Call

Participants:

Justin Gottschlich
Michael Scott
Maged Michael
Mike Spear
Victor Luchangco
Hans Boehm
Torvald Riegel
Mark Moir
Tatiana Shpeisman

Secretary Rota for next meeting:

Tatiana, Michael W, Justin, Maged, Michael Scott, Paul M, Mark, Hans, Torvald, Victor, Mike Spear

Agenda adopted

Previous minutes approved

Review action items from previous meeting:

1.4.1: Torvald to send email regarding implementing atomics without speculation

It appears there was an error in the action item. Torvald *did* send an email, but it was not in regard to this topic. A more appropriate description is:

"Torvald to send an email that tries to make the case about supporting failure atomicity without changing the specification"

DONE

1.4.2: Mark to send ISO legal document to Justin

DONE

1.4.3: Justin, Mark, and Maged to work on legal issues to "release" the specification to ISO.

Justin -> At Intel, Justin and Tatiana are trying to find the right legal

contact, there is much still in flux here. Mark had suggested that the three legal teams meet, but this is still in progress.

Maged -> IBM is taking some time to study the issues. Also, Maged should be included in this process (Note: Mark sent relevant documents to Maged before phone conference was over)

Mark -> Oracle has some concerns, and Mark suspects that each of the companies will have concerns.

Carry-Over

1.4.4: Michael Wong to get clarification on how many national bodies are needed for Technical Specification

Carry-Over

Main Issues

2.1 Status of safe-by-default

Mark -> There hasn't been much progress on this topic. There isn't much controversy right now, and he expects that it will be straightforward to flesh out a proposal. Any proposal will decrease the size of the current spec, but it is probably best to wait.

Torvald -> Agrees with mark. However, it is probably best to wait until the scoped lock and exception issues are addressed first.

Mark -> While SBD changes to the spec are not likely to overlap with the changes induced by the scoped lock and exception discussions, it makes more sense to wait, rather than draft a set of changes to the spec only to have much of the spec change.

Michael Scott -> Agree

2.2 Relaxed transactions as scoped locks and their interaction with atomic transactions

Michael Scott -> Provided an overview. Michael had proposed in September. This proposal observed that relaxed transactions appear to be semantically equivalent to scoped locks using a global, elidable lock. From there, he proposed that the relaxed construct could be replaced with a special scoped, elidable lock that atomic transactions implicitly check is unheld. This should simplify semantics, since there is no need to specify relaxed transactions as

separate from existing language features. From there, Mark noted that concurrency among relaxed transactions is a good idea, and proposed a "workgroup" mechanism to allow code to prevent only "certain" atomic transactions from running. Thus "one kind" of transactions can be kept from running, or more generally a programmer can categorize types of atomics to achieve better scalability.

Michael Scott: I would add that Mark's proposal also allows (true, semantic) concurrency among non-transactional blocks of code that all exclude transactions. This may be of equally important benefit to scalability.

Torvald -> These are still mutexes, but the two roles are separated: one role is to say "this is part of a group" and the other role is to say "don't run concurrent with another group".

Mark -> Sort of. This is more like group mutual exclusion, because it does not say who can run, only who cannot.

Michael Scott -> It is possible to get effect of a mutex with workgroups by inhibiting and associating with the same group.

Torvald -> Why the name "workgroup"

Mark -> Seems like a logical grouping for transactions, but nobody is invested in this name.

Torvald -> Had a question about example 1, and the reason why doing an associate and inhibit instead of an inhibit and scoped elidable lock would be preferred.

Mike Spear -> The use of a lock may be preferred because it introduces a lock elision opportunity that might be missed otherwise.

Michael Scott -> We shouldn't conflate semantics with implementation.

Torvald -> OK. Can we switch order of inhibit and associate?

Victor -> A clarification: Michael Scott's description of workgroups emphasized inhibiting transactions, but they are more general, and can inhibit any arbitrary code. Victor is not sure this is a good idea, though he is the one who pushed Mark toward this. There was email discussion about deadlocks, and Victor believes that while the general mechanism can deadlock, if only transactions are inhibited, then deadlocks cannot occur.

Tatiana -> Requested some background. It sounds like we have agreed to eliminate relaxed transactions, or that people do not want relaxed transactions

anymore.

Michael Scott -> The proposal is meant to capture that relaxed transactions serve to mediate transactional vs. nontransactional operations, but in a way that does not overload the word "transaction" to mean both atomic transactions and non-isolated operations.

Tatiana -> Expressed a concern that locks have a specific meaning, usage, and mental model that may be a problem for programmers.

Mark -> Claim that even if programmers see things as different, the semantics are equivalent.

Tatiana -> Urges against driving the choice of language features based on theoretical properties that don't make sense to programmers.

Michael Scott -> Believes we can "have cake and eat it" in this case, since the mental model is an /elidable/ lock.

Tatiana -> Expressed a concern that we shouldn't promote an optimization technique as being equivalent to a language feature.

Michael -> Asked if there is a concern that the elidable lock will be more difficult for the compiler to detect than the keyword "transaction_relaxed". Argued that the pairing of a workgroup.inhibit statement with a lock acquire should be easy for a compiler to detect.

Victor -> Clarify that "relaxed transaction" probably not exactly equivalent to "elidable lock" because we can implement relaxed transactions in many ways. We should not focus on the elision aspect, as it is a hint to the compiler, but has no semantic meaning.

Mark -> Claim that in previous discussion, many agreed not to use the term "elidable" since it is not the main focus. The focus should be on characterizing the language, understanding that compilers and implementations will vary.

Victor -> "Elidable" does not matter. "Scoped" matters. There exist three issues:

1) names: "relaxed transaction" is a name that is not universally favored, as it can be confusing.
2) role: there are two issues or functionalities provided by relaxed transactions. One is to prevent the concurrent execution of other relaxed transactions. A lock can do this. The other is to prevent the execution of concurrent atomic transactions. This requires a separate mechanism. Locks don't work for this role.

(these two points raise the question of how to get the functional equivalent (wrt semantics) of relaxed transactions. We would need a lock, plus something new. In essence, relaxed transactions are the sum of these 2 things.)

3) granularity: relaxed transactions are probably too coarse grained. We probably only want to forbid the concurrent execution of *some* relaxed transactions. It's natural to address this with locks. But we probably also only want to forbid the concurrent execution of *some* atomic transactions, which again would require a new mechanism. Workgroups can serve this role.

Tatiana -> Names matter, and there is a real difference between a language construct and a library feature. There are two questions: should there be a keyword for relaxed transactions, and if there should be a keyword, should it include the word "transaction" or not. It does not seem that everyone agrees on these questions.

Justin -> Generally impressed by the workgroups feature, sees a benefit, but has two concerns:

1) deadlock: it seems that workgroups can introduce deadlocks in ways that relaxed transactions cannot
2) complexity: this proposal is complex. There are more new instructions and named variables.

Mark -> Believes that the deadlock argument is due to drawing artificial lines. Any synchronization mechanism can lead to deadlock, and it is possible to get counterintuitive deadlocks with relaxed transactions.

Mark -> Regarding simplicity, believes that having "relaxed" and "atomic" transactions is, in itself, a source of complexity.

Mark -> Clarify that there is no desire to remove the functionality of relaxed transactions. Believes they are important in part because they allow us to keep atomic transactions "safe and pure" from a semantic perspective. Mark agrees with the use cases for relaxed transactions, wants us to focus on their role, not their name. Mark believes that there are more use cases than we've explored, and fears that relaxed transactions suggest something about performance that is not likely to happen in real systems. Believes that transactions are not always more scalable than locks, especially with relaxed transactions.

Torvald -> Does not see the use of the word "lock" as being helpful. Wonders if we can go forward with the "relaxed transaction" and "workgroup" proposals simultaneously, and see how things fall out later.

Michael Scott -> Is there general agreement that the scoped lock concept is equivalent to relaxed transactions? If so, we can discuss semantics.

Hans -> Yes, we agree.

Tatiana -> Semantic equivalence does not mean we can't have both in the language.

Michael Scott -> Agrees, cites "for" loops prior to ~1990, which were same as "while" loops.

Victor -> Can see that two different syntactic constructs for the same feature is OK, but wants to know if we want to have two mechanisms: one for being nonconcurrent with relaxed transactions, and another for being nonconcurrent with atomic transactions.

Justin -> Asked all attendees if they were in agreement on the semantic equivalence of the two mechanisms. All agree.

2.3 Mark and Dave's proposal on atomic transactions, cancel, and exceptions

We ran out of time, did not get to this.

2.4 Legal issues discussion

We ran out of time, did not get to this.

---

3. Any other business

4. Review

4.1 Review and approve resolutions and issues [e.g., changes to SG's working draft]

4.2 Review action items

Carry forward action items 1.4.3 and 1.4.4.

5. Closing process

5.1 Establish next agenda

5.2 Future meetings: December 10, teleconference

5.3 Adjourn

Minutes of 2012/12/10 SG5 Conference Call

Agenda:

1. Opening and introductions

1.1 Roll call of participants

Tatiana Shpeisman

Michael Wong

Maged Michael

Michael Scott

Mark Moir

Torvald Riegel

Victor Luchangco

Mike Spear

Jens Maurer

Secretary Rota for next meeting:

Michael W, Justin, Maged, Michael Scott, Paul M, Mark, Hans, Torvald, Victor, Mike Spear, Jens Maurer, Tatiana

1.2 Adopt agenda

Agenda has been adopted

1.3 Approve minutes from previous meeting

Minutes from the previous meeting have been approved

1.4. Review action items from previous meeting:

1.4.1. Justin, Mark, and Maged to work on legal issues to "release" the specification to ISO.

Mark reiterated the need for a meeting between lawyers from Oracle, Intel and IBM to discuss Oracle's concerns that might also be a concern for other companies. Everybody agreed that this is the right next step.

Maged will report request to appropriate people at IBM. Tatiana reported that Justin is working on locating an appropriate legal representative at Intel. Michael Scott questioned why this is an issue at all given that all participating companies have a long history of working with ISO. Mark clarified that the question is about the work that has been done before the group moved under the C++ standards committee umbrella.

Carry-over

1.4.2. Michael Wong to get clarification on how many national bodies are needed for Technical Specification.

Michael Wong -> It requires simple majority and 5 national bodies saying that they are willing to actively work on it.

There were many clarifying questions. In the interest of time, Michael W will send replies via e-mail.

There was a request to spend less time during the meetings discussing administrative matters.

---

2. Main Issues

2.1. Recap of 2012

- Michael W gave a quick summary of what we done in 2012. We brought the proposal to the Kona meeting, brought justifications and reasons to the meeting in Seattle and gave enough justifications to form SG5 study group. The group has been working very hard, much more actively than many other groups. Meeting in Portland showed that people are willing to give us a shot.

2.4. What is our goal for Bristol (April 2013)?

Michael Wong: We would like to put worded proposal before the committee. Suggested that we take time over Christmas and New Year to summarize five existing proposals under the discussion and write them up. Later, we could submit all or some of these proposals as SG5 papers. For controversial proposals, the paper could summarize both the proposal and the controversial points under the discussion so that the whole committee could participate if they are interested.

The five proposals are:

- Safe by default

- New exception design mechanism

- Scoped lock

- Work groups

- Catch outer idea that Mark started

Tatiana: Concerned that presenting only new proposals as papers will give the new ideas more weight compared to the existing specification. Suggested writing two proposals for controversial topics defending two points of view.

Michael W: Review the proposals within our group and then decide if we want to disseminate the proposals to the rest of the group.

Michael Scott: What is the border between exception issue and the catch outer idea?

Mark: Very happy that we have discussion about the controversial issues. Does not think we need to create competing proposals.

Michael Scott: Suggested that he and Tatiana create a short summary of the agreement and disagreement points for the scoped lock proposal.

Torvald: Encourages all the discussion to happen on the reflector rather than in private e-mail conversations

Victor: A counter proposal. Let's just describe the proposals as Michael Wong suggested.

Mark: Summarized where he thinks we are with the discussion of relaxed transactions/scoped lock proposal. Thinks that writing up a summary of the discussion points won't be useful and that we should continue e-mail discussions, instead, as we are making good progress on understanding each other positions.

Tatiana: Suggested that we attempt to reach decisions on the controversial issues before presenting the main proposal at Bristol.

Michael Wong: Encouraged people to write the proposals.

Mark: It is premature to write the details of the 'Safe by Default' proposal, as there will be significant changes depending on what we decide to do about exceptions.

Tatiana: Suggested that we determine next step for each proposal individually.

The group agreed that there are two main important topics under discussion – Exceptions and Scoped Lock.

Mark will try to write up exception proposal if he has time.

Tatiana will be happy to work with Michael Scott on summarizing Scoped Lock agreement and disagreement points but does not have time to write up the whole proposal.

Tatiana: Could Mark write a short summary of the exception proposal?

Mark: There is no proposal yet. The only way to catch up with the conversation is to read all the e-mails.

Next meeting will be on January 14th, going every other week after that.

Minutes of 2013/01/14 SG5 Conference Call

1. Opening and introductions

1.1 Roll call of participants

Michael W, Justin, Maged,Mark, Hans, Torvald, Mike Spear, Viktor, Tatiana

1.2 Adopt agenda
MW, JG

1.3 Approve minutes from previous meeting
MW, Maged M

1.4 Review action items from previous meeting

1.4.1 Everyone: Please review this priority section before Jan 14 meeting
http://wiki.edg.com/twiki/bin/view/Wg21bristol/SG5
Done

1.4.2 Maged has the IBM legal contact names, checking Oracle and Intel to set up a joint time to meet.
Intel trying to connect with a lawyer, Oracle and IBM has identified their legal rep. Intel has internal dependencies.

1.4.3 Michael Wong to get more clarification on what actively working on it means for a Technical Specification.
requires simple majority and 5 national bodies saying that they are willing to actively work on it.
A checkbox for the 5 NBs

1.4.4 Everyone: Bristol UK registration info
The following link will take you to the WG21 tab of the ACCU booking site:

http://www.cvent.com/events/accu-2013/custom-22-09ec03b22c4f4a0a832e28126a4585fc.aspx

As mentioned in N3397:
http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2012/n3397.pdf the event
is colocated with the ACCU conference - Apr 9-13)

2. Main issues

2.1 Goal for Bristol
Present a draft of TM and get through discussion on a few priority items?

MM: we have the page below, and got good understanding, do something along the lines of the **unified** suggestion, put it in similar qualify as 1.1 draft
JG: agreed unified approach is a good balance, slight lean to #2; not head down towards standarese
MM:good for Jens to start a first cut, how you specify can be affected by the machinery of how you specify

2.2 Prioritize TM discussion ahead of Bristol based on the following page
http://wiki.edg.com/twiki/bin/view/Wg21bristol/SG5
JG: Some of the work like keeping the names #7, #5 don't need work
MM: relaxed and scoped locks is still wide rift
TS: may be continue through wg proprosal, relaxed and scoped locks are not equivalent
MM: wg proposal aims to make relaxed tx more easily specified
relaxed tx are equivalent to a scoped lock of a single global lock
HB: implementation issue is serious, OK with the semantics of it
TS: should we keep relaxed tx keyword? this is the more contentious issue
MM:for now, for Bristol, keep the keyword (change the spelling), at least agree what it means
JG: OK with Mark's proposal
TS: this is a mental model?
MM: not just a mental model
MW: is the page a priority ordering?
TS/MM: no but can reprioritize
#3 nothing to do, #4 is dependent on #2
#5 nothing to do
#6/7 become non-items
just work on #1+#2
MW: good as we only have time for #1/2
JG/Maged: good to keep on the list
HB: can build an impl based on closed nesting, but Mark still questions that
MM: possible large performance impact and kills it, also could be large impl burden
HB:Jen's argument of doing everything using flat nesting before going to closed nesting could help out here

2.3 Writing Standarese for current TM draft
http://jmaurer.awardspace.info/wg21/tmspec.html
Defer as we agree to not put out standarese for Bristol

2.4 Schedule discussion for the next 6 weeks.
- Jan 14: Priority discussion today
- Jan 21: Discuss Priority 1 item
- Feb 04: Continue P1 item (Bristol registration ends about Feb 12)
- Feb 18: Discuss Priority 2 item; writing begins for Pre-Bristol deadline
- Mar 04: Continue P2 item (Pre-Bristol mailing deadline: 15 March 2013)
- Mar 18: Discuss Priroity 3 item (March break 2nd week: MW away to conference)
- Apr 01: Prep for Bristol (last call before Bristol;also Easter Monday)
- Apr 15: Bristol

1. meeting minutes
2. Flesh out safe-by-default proposal
3. Actively continue with the current exception discussion on how nesting, cancel, and exceptions interact. Make a decision on the exception behavior of the strong/atomic transactions once we have a detailed proposal for abort-on-exception behavior that addresses all the issues. Make the decision on [[outer]], [[may_cancel_outer]] and cancel-throw in the context of the same discussion.

Mark+Victor: should we update the spec? we feel that would be nice
TS: prefer not to update the spec, have license issues, and uncomfortable to update it at this point
Better to work on it as an independent proposal or as a change to the spec?
MM: probably work on it independently
VL: premature discussion
MW: perhaps just a status update on the delta
VL: agree that we should not push the spec if there are legal issues, this should be a secondary discussion
MW: OK sounds good, what is up for next week
MM: exception and nesting first, more items to discuss then SBD
All agree

3. Any other business

None

4. Review

4.1 Review and approve resolutions and issues [e.g., changes to SG's working draft]

4.2 Review action items

Maged has the IBM legal contact names, checking Oracle and Intel to set up a joint time to meet. Intel trying to connect with a lawyer, Oracle and IBM has identified their legal rep. Intel has internal dependencies.

Mark to put out Item 2 exceptions on nesting proposal

Everyone to read Mark's writing

Everyone: Bristol UK registration info: preferred dates of our meeting (Tuesday to Friday)

5. Closing process

5.1 Establish next agenda
Discuss Mark's proposal on Exceptions and nesting

5.2 Future meetings: January 21, teleconference

5.3 Adjourn
MW, JG

Minutes of 2013/01/21 SG5 Conference Call

1. Opening and introductions

1.1 Roll call of participants

Justin Gottschlich (JG), Michael Scott (MScott), Michael Wong (MW), Victor Luchangco (VL), Torvald Riegel (TR), Maged Michael (MMichael), Mike Spear (MSpear), Hans Boehm (HB), Mark Moir (MMoir)

1.2 Adopt agenda

Adopted, MW.

1.3 Approve minutes from previous meeting

Approved, MW.

1.4 Review action items from previous meeting

1.4.1 Maged has the IBM legal contact names, checking Oracle and Intel to set up a joint time to meet.
Intel trying to connect with a lawyer, Oracle and IBM has identified their legal rep. Intel has internal dependencies.

Ongoing.

1.4.2 Mark to put out Item 2 exceptions on nesting proposal

Done.

1.4.3. Everyone to read Mark's writing (as we always do anyway)

Done.

1.4.4 Everyone: Bristol UK registration info: preferred dates of our meeting (Tuesday to Thursday)
The following link will take you to the WG21 tab of the ACCU booking site:

http://www.cvent.com/events/accu-2013/custom-22-09ec03b22c4f4a0a832e28126a4585fc.aspx

As mentioned in N3397: http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2012/n3397.pdf the event is colocated with the ACCU conference - Apr 9-13)

MW: Suggesting everyone attend from Tuesday – Thursday (April 16-18).
For most people in SG5, register only for WG21.

Action item: MW to provide a clearer explanation of registration process.

1.4.5. Everyone: Review important links:
a. Prioritize TM discussion ahead of Bristol based on the following page
http://wiki.edg.com/twiki/bin/view/Wg21bristol/SG5

b. Writing Standarese for current TM draft
http://jmaurer.awardspace.info/wg21/tmspec.html

2. Main issues

2.1 Discuss P2 item of Suggestion 3 Unified of
http://wiki.edg.com/twiki/bin/view/Wg21bristol/SG5

Actively continue with the current exception discussion on how nesting, cancel, and exceptions interact. Make a decision on the exception behavior of the strong/atomic transactions once we have a detailed proposal for abort-on-exception behavior that addresses all the issues. Make the decision on [[outer]], [[may_cancel_outer]] and cancel-throw in the context of the same discussion.

MMoir: (Providing some history about cancel) We've had lots of discussion about cancel: cancel, cancel outer. Believes we must have the ability to cancel the entire transaction. [Seems to be consensus in group.] Believes we should support at least the minimum, such as

cancel outer. However, Mark has received some concerns about this approach not supporting enough cases, such as closed nesting.

Moving forward, Mark aimed to put together a proposal to support both flat and closed nesting. The proposal is really meant to be taken a single point in space; not married to it, but thinks it's important to get us started.

MScott: Did you initially go into this process to avoid the need for true closed nesting?

MMoir: Sorry. To clarify, the reason I wanted to have proposal for closed nesting was it seemed others were arguing for it. If there is sufficient support for it, we should put it into the next spec.

TR: Can you talk about when a transaction is actually cancelled? Is it before the transaction enters its associated catch block or after it?

MMoir: It's afterward the catch block.

MScott: Care will be needed to understand what is accessible in that catch block.

HB: This is one of the points of the proposal that concerns me. Because you're in this somewhat ambiguous intermediate state, it may be unsafe to call arbitrary library code. Hans continued to point out that he believes that if we support cancellation at all, then we must also support closed nesting.

MMoir: If you can't call any library code in the transaction's catch

block, that's fairly restrictive. You should be able to at least call std::transaction_nesting_level(), which may be needed for to decide how to rethrow (cancel just the parent or all nested transactions).

Bottom line though, you should not be able to call arbitrary code.

HB: I still remain concerned. Essentially all operations are library calls. Must identify (perhaps manually?) which operations are safe to perform in the middle of these cancelled transactions.

MScott: The transaction's catch block must decide to commit or cancel, if the exception it receives is not a transaction cancellation exception.

TR: To clarify my question. When do something with a cancel transaction exception, you do it in the catch block after the atomic. These execute at the point where the transaction exception is thrown. If we took the contents of the catch block and rather supplied that to the transaction as a lambda, would we be able to avoid the catch block syntax. Is this reasonable?

MMoir: You're not proposing a change to the behavior, right? I sort of think that the behavior of the transaction / catch blocks are more similar rather than less similar to the try / catch blocks.

TR: Right, just cleaning up the syntax. The lambda approach might make it a bit easier for the programmer to reason about.

MMoir: Let's step back and understand why we want to execute the code in the catch block. First, if you catch a cancellation exception, then you must throw a cancellation exception.

VL: You can throw an exception that can be used in the transaction's catch block, but it cannot be thrown outside of the transaction because it includes part of the transaction's state. This is why you must throw transaction cancellation exceptions escaping a transaction's catch block, which is guaranteed to not include any transaction state.

HB: As a standard convention for C++, when an exception is thrown it is usually not handlable by the code that throws the exception.

HB is concerned about library code that is defined as noexcept, but then throws a transaction cancellation exception from user-defined code it invokes that must then be propagated outside of the library code because it cannot be caught by the catch mechanism of the library's noexcept catch handler.

VL: You're concerned about library code that might not be allowed to throw an exception, but then calls user-defined code that throws a transaction cancel exception.

MScott: In summary, the noexcept attribute means this method will not throw anything. But cancellation exceptions cannot be masked, so they violate the noexcept rule.

[My apologies, for the shortness of our notes here; I was having trouble hearing everyone and was also trying to be involved in the discussion, so I didn't catch all details of it. Please feel free to add comments about things you said here that I've missed.]

Continued significant discussion about transaction cancel exceptions thrown within the context of a noexcept library method that invokes user-defined (called-back) code.

Action item: Hans to come up with simple code example to concretely explain problem.

3. Any other business

4. Review

4.1 Review and approve resolutions and issues [e.g., changes to SG's working draft]

Action item: Hans to come up with simple code example to concretely explain problem about transaction cancel exceptions thrown from within the context of a noexcept library function.

Action item: Everyone to review Hans' example and begin discussion.

Action item: Michael Wong to clarify Bristol registration process.

4.2 Review action items

5. Closing process

5.1 Establish next agenda

5.2 Future meetings: Feb 4, teleconference

Jan 14: Priority discussion (DONE)
Jan 21: Discuss P2 item
Feb 04: Continue P2 item (Bristol registration ends about Feb 12)
Feb 18: Discuss P1 item; writing begins for Pre-Bristol deadline

Mar 04: Continue P1 item (Pre-Bristol mailing deadline: 15 March 2013)
Mar 18: Discuss other item (March break 2nd week: MW away to conference)
Apr 01: Prep for Bristol (last call before Bristol;also Easter Monday)
Apr 15: Bristol

5.3 Adjourn

Minutes of 2013/02/04 SG5 Conference Call

1. Opening and introductions

1.1 Roll call of participants

Maged, Michael Scott, Paul M, Mark, Torvald, Victor, Mike Spear, Jens Maurer, Michael W, Justin, Hans

1.2 Adopt agenda

Michael W

1.3 Approve minutes from previous meeting

Michael W

1.4 Review action items from previous meeting

1.4.1 Maged has the IBM legal contact names, checking Oracle and Intel to set up a joint time to meet.
Intel trying to connect with a lawyer, Oracle and IBM has identified their legal rep. Intel has internal dependencies.

Identified representatives in each company and contacted the legal representatives with the info

1.4.2 Hans to put out example demonstrating his concerns

Done

1.4.3. Everyone to read Hans writing and feedback

Done

1.4.4 Everyone: Bristol UK registration deadline Feb 12: preferred dates of our meeting (Tuesday April 16 to Thursday April 18)
We should have a discussion as to who intends to go other then Hans and me.

The following link will take you to the WG21 tab of the ACCU booking site:

http://www.cvent.com/events/accu-2013/custom-22-09ec03b22c4f4a0a832e28126a4585fc.aspx
Select wg21 registration only
select full wg21 event
select Marriot if possible
select April 16-18 hotel as a minimum, but you are welcome to increase this time.

The longer invite N3397:
http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2012/n3397.pdf

1.4.5. Everyone: Review important links:
a. Prioritize TM discussion ahead of Bristol based on the following page
http://wiki.edg.com/twiki/bin/view/Wg21bristol/SG5

b. Writing Standarese for current TM draft
http://jmaurer.awardspace.info/wg21/tmspec.html

2. Main issues

2.1 Continue to Discuss P2 item of Suggestion 3 Unified of
http://wiki.edg.com/twiki/bin/view/Wg21bristol/SG5

Actively continue with the current exception discussion on how nesting, cancel, and exceptions interact. Make a decision on the exception behavior of the strong/atomic transactions once we have a detailed proposal for abort-on-exception behavior that addresses all the issues. Make the decision on [[outer]], [[may_cancel_outer]] and cancel-throw in the context of the same discussion.

(Summarizing, paraphrasing, misrepresenting, misunderstanding, etc... :)
Michael W: Is Hans' example of practical concern?
MLS: Interesting example but not rising to the level of countering Mark's proposal.
Hans: Concerned that the proposal requires more specification of what libraries can expect.
Mark: A check of whether the callback is in a transaction or not should prevent the problem
Hans: The callback arguments are hidden in the library
Hans: Maybe we should postpone discussion after reading Torvald's proposal
Victor: Time is tight before Bristol
Jens: In agreement with Hans
MLS: Transaction cancel has no effect on the world
Michael W: Let Torvald explain his proposal
Torvald: Exception and cancellation are different. Exceptions have strict semantics. Cancellation can be composable.
Torvald: Cancellation by calling transaction_fail()
Torvald: The handlers are set before starting the transaction and not bound to any state inside the transaction.
[Added by Torvald post meeting] The fail handler lambda could bind to state in the transaction. The progress handler does not. I suppose I need to clarify that in a
subsequent email.
Cancel calls whatever handlers are in effect at the time. Part of the handler is executed before cancel and another part after cancel. The fail handler may have to access the TM implementation ABI but the programmer need no know about it.
[Added by Torvald post meeting]And also clarify / expand on this.

3. Any other business

4. Review

4.1 Review and approve resolutions and issues [e.g., changes to SG's working draft]

4.2 Review action items

5. Closing process

5.1 Establish next agenda

Michael W: Should continue discussing exceptions, Torvald's proposal, or safe by default (SBD)
Mark: Probably not possible to pursue Torvald's proposal before Bristol
Michael W: Let's discuss SBD next and in the background Hans elaborate on his concerns
Mark: Don't want to stop making progress on exceptions.
Michael W: Agree
Hans: Best way to make progress on exception is to construct more examples offline
Victor: What are we going to say for Bristol
Hans: This is not a critical issue. We agree on exception (with statically nested cancellation). We can say that we are working on details.
Michael W: Keep discussing exceptions for one more meeting.
Jens: We should start discussing SBD
MLS: Someone should write a nice summary of SBD
Mark: Exceptions are higher priority. Not volunteering to write summary.
Michael W: Let's continue exceptions for one meeting

5.2 Future meetings: Feb 18, teleconference

Jan 14: Priority discussion (DONE)
Jan 21: Discuss P2 item (DONE)
Feb 04: Continue P2 item (Bristol registration ends about Feb 12)
Feb 18: Discuss P1 item; writing begins for Pre-Bristol deadline
Mar 04: Continue P1 item (Pre-Bristol mailing deadline: 15 March 2013)
Mar 18: Discuss other item (March break 2nd week: MW away to conference)
Apr 01: Prep for Bristol (last call before Bristol;also Easter Monday)
Apr 15: Bristol

5.3 Adjourn