No, inplace_vector shouldn't have an Allocator

Document number: P3581R0
Date: 2025-01-13
Project: Programming Language C++, Library Evolution Working Group
Reply-to: Nevin "☺" Liber, nliber@anl.gov

This is in response to [P3160R2] An Allocator-Aware inplace_vector.

While [P3160R2] shows that there are folks "who would be OK" [P3160R2 section 2 poll] with adding another template parameter to inplace_vector, and there was technical discussion to get address their particular concerns (see [P3455R0] for the SG14 meeting minutes on [P3160R1]), no new technical information is presented to re-litigate the discussion that happened in Tokyo for [P3160R0].

In the Tokyo discussion (my summary of the arguments against):

We care about this use case.

It should be a different vocabulary type, such as basic_inplace_vector<T, N, A>.

Doing this in inplace_vector makes it an overly complicated type for the vast majority of users.

It is not worth the compilation costs, as the vast majority of users will not use this feature (backed up by statistics in [P3062R0]).

After that discussion, we polled:

POLL: We should promise more committee time to pursuing "An Allocator-aware inplace_vector," knowing that our time is scarce and this will leave less time for other work.

SF	F	N	A	SA
6	6	4	5	6

Attendance: 25 (in person) + 9 (remote)
# of Authors: 1
Authors' position: SF
Outcome: No consensus.
The room was not supportive of applying this paper to inplace_vector data structure.

During the Tokyo plenary, the author of [P3160R0] objected to unanimous consent for [P0843R11]. Because there was also a technical objection that most of the [P0843R11] authors agreed with, the paper was not polled in Tokyo. The author of [P3160R0] stated that SG14 would be discussed at the next telecon, before the St. Louis meeting. See [n4980] LWG motion 15 for details.

By St. Louis the technical objection was resolved and inplace_vector was voted on and approved for C++26, with strong consensus (only one dissenting vote). To quote [N4985] LEWG Motion 7:

7. Apply the changes in P0843R14 (inplace_vector) to the C++ working paper.

Herb Sutter : This paper was delayed in Tokyo. Have all the concerns now been addressed ?
Jonathan Wakely : Yes. The paper was changed to limit the types that are constexpr which makes it
possible to implement.

Inbal Levi : And there is also implementation experience now.

Objections in the room.
In favour : 70 (54 in person + 16 online)
Against : 1 (1 in person + 0 online)
Abstain : 29 (13 in person + 16 online)

Motion passes.

While "most of the [SG14 June 2024] attendees did not object to adding allocator support to inplace_vector" [P3160R2 section 2], this isn't an argument that adding an allocator template parameter to inplace_vector is the way forward. As already litigated, doing so is not the correct design to address this need. Make it a separate type, such as basic_inplace_vector<T,N,A>.

References

N4980 – WG21 March 2024 Hybrid meeting Minutes of Meeting
N4985 – WG21 June 2024 Hybrid meeting Minutes of Meeting
P0843R11 – inplace_vector
P0843R14 – inplace_vector
P3062R0 – C++ Should Be C++
P3160R0 – An Allocator-aware inplace_vector
P3160R1 – An Allocator-aware inplace_vector
P3160R2 – An Allocator-aware inplace_vector
P3455R0 – SG14: Low Latency/Games/Embedded/Financial Trading virtual Meeting Minutes 2024/6/12-2024/10/9