P3173R0 2024-02-15 Reply-To: gdr@microsoft.com
Audience: EWG

P2900R6 May Be Minimal, but It Is Not Viable

Gabriel Dos Reis
Microsoft

1 Abstract

Microsoft does not consider the Contracts facility, currently proposed in P2900R6, as viable. These concerns are entirely independent of any suggestions that may have been discussed by SG21 in the past to address some alarming aspects of the Contracts facility in its current incarnation. The paper P2900R6 notes that a "contract is a set of conditions that expresses expectations of how the component interoperates with other components in a correct program." A contract system in which evaluation of contract specifiers is permitted to exhibit undefined behavior undermines the principle that SG21 agreed to, namely that safety should be prioritized. Microsoft considers that a viable Contracts system for C++ needs to consider safety by default in its design. Furthermore, the proposal would instill more confidence towards viability if it shows applications to the Standard Library and adequately addresses uses in combination with other foundational C++ facilities such as dynamic dispatch, function pointers, coroutines, etc.

2 Prioritizing Safety

SG21 took a poll on October 6th, 2020 to determine whether it agrees "to progress contract checking to enforce software safety first, and enable assumptions of injected facts at a later time". The result was

SF	F	N	A	SA
7	4	2	0	1

which represents a strong consensus to prioritize safety first for contracts. The "injected facts" aspect was separately progressed into C++23 via the `[[assume]]` attribute proposal (Doumler, 2022). So, that left the safety aspect of contracts for SG21 to focus on. The current proposal fails to deliver on safety, which remains a contemporary challenge to C++. A viable Contracts facility needs to (re)commit to prioritizing safety as that continues to be a major contemporary challenge for C++ regarding continued usability, relevance, and viability in key areas of application.

3 Undefined Behavior

The Contracts facility as currently specified in P2900R6 fails to address consequences of undefined behavior invoked during evaluation of contract specifiers. It is unacceptable that a contract check – typically uses to guard against bugs or unintended behavior -- can be elided (or worse exploited) because of undefined behavior. It is not enough that the specification claims that it "has deliberately

not introduced any new explicitly undefined behavior in the C++ language". See (Dos Reis, 2022) for further elaboration and examples. Without adequate limitation on undefined behavior in the evaluation of contract assertions, Microsoft considers the Contracts facility not viable in the contemporary environments where C++ is used.

4 Dynamic Dispatch and Indirect Calls

Dynamic dispatch and indirect function calls (e.g. through pointer to functions) remain a staple of software interface design and implementation in C++ ("modern" or otherwise) and of pervasive use. Consequently, a Contracts facility that fails to adequately support efficient use with virtual functions and pointer to functions is woefully inadequate and unready for prime use, and inclusion in C++26 in particular. Microsoft recommends against the inclusion of P2900R6 in C++26.

5 Standard Library

P2900R6 explicitly states that it does not propose any changes to the specification of the existing Standard Library. Failure to show how this facility can be used and depended upon in a component as fundamental as the Standard Library is implicit admission of immaturity and inadequacy of the Contracts facility at this point in time. As shown by numerous proposals in the past, application of a new language features to the Standard Library has often surfaced usability and viability problems. Microsoft recommends against P2900R6 without applications in the Standard Library, in particular in the algorithm section, and field experience.

6 Conclusion

Microsoft sees potential actionable value in a Contracts facility for C++ in its areas of application. However, Microsoft recommends against the current specification in P2900R6 to progress in C++26 (or future versions) unless it adequately addresses the concerns around safety, undefined behavior in evaluation on contract specifiers, dynamic dispatch and indirect calls, application to the Standard Library.

7 References

Dos Reis, G. (2022, December 15). Contracts for C++: Prioritizing Safety. Retrieved from https://open-std.org/JTC1/SC22/WG21/docs/papers/2023/p2680r1.pdf

Doumler, T. (2022, June 14). *Portable assumptions*. Retrieved from https://www.open-std.org/jtc1/sc22/wg21/docs/papers/2022/p1774r8.pdf