Doc. no.: P5000R0
Date: 2026-02-18
Programming Language C++
Audience: All WG21
Reply to: *D. Vandevoorde* (daveed@vandevoorde.com)
Author: Directions Group

DIRECTION FOR ISO C++29

J. Garland, P. McKenney, R. Orr, B. Stroustrup, D. Vandevoorde, M. Wong

Revision History:

- **R0: this paper (2026-02-23 published):**
  - Initial revision, suggesting priorities for C++29
  - added Recommended Action Guidelines for C++29

1. Abstract

This document proposes specific goals and priorities for C++29. See P2000 for longer-term direction suggestions.

While P2000 outlines the decades-long philosophical vision for C++, this document (P5000) focuses strictly on the immediate priorities for the C++29 cycle. To execute this cycle effectively, the Direction Group has published a series of actionable, standalone directives that give guidance to authors and chairs. These include strict mandates on safety **P3970R0**: Profiles and Safety: a call to action, , proposal convergence (**P4024R0**), and committee governance regarding AI (**P4023R0**). We view compliance with these standalone directives as a prerequisite for success in C++29."

2. Tier 0: Safety

For the past several years, memory safety and other kinds of safety have been dominating the discourse about C++ as a programming language. C++29 should endeavor to deliver significant progress. We encourage both Language and Library groups to prioritize proposals that close undefined behavior and make C++ safer via improved user APIs over other work.

2.1 Profiles

Currently, the “Profiles” feature is the primary proposal on the table to tackle these issues (See, e.g., P3970R0 “Profiles and Safety: a call to action” and P3589R2 “C++ Profiles: The Framework”). We recommend giving their development and review top-priority in the C++29 standardization cycle.

2.2 Library Facilities

Library facilities have an important role to play in shoring up the ability of users to write safe programs. This includes additional library hardening. In addition, providing safe APIs to replace those reliant on unsafe practices - such as pointer returns that can lead to use after free and other memory errors. Furthermore, assisting users to write type safe APIs not subject to legacy conversion rules.

3. Tier 1: Maintenance

C++26 is a major release: The inclusion of support for reflection, contracts, and `std::execution` ("senders/receivers") is evidence of that. No doubt, as these features are deployed, we are going to find "gaps" that must be filled.

P3962 also testifies that implementers are having a difficult time keeping up with the changing language. Some of it is due to the significant amount of work implied by the major features, but sometimes even "small-looking" features lead to a considerable implementation burden.

We therefore suggest that C++29 be considered a "maintenance release": Aim to reduce friction surrounding what is already in the language, and avoid forcing through proposals that are likely to cause significant delay in conformance.

4. Tier 2: Carry Over Priorities from C++26

Despite the size of C++26 we acknowledge two past priorities carrying over to the next cycle.

3.2. Pattern Matching

Support for pattern matching has remained in high demand and work was done during the C++26 cycle to progress this feature. It was considered a priority for C++26. Having missed that release, it becomes a priority for C++29.

3.3. Networking

The community has been waiting for well over a decade now to get standard support for networking. C++26 adds std::execution support — which should enable high-performance asynchronous work — but there is still no standard framework for run-of-the-mill networking tasks (such as dealing with internet-based protocols). Progress here would be very welcome.

5. Study Proposals

The above suggests priorities for standardization goals. However, we believe WG21's work could also benefit from topical studies to gather data that will hopefully bring better understanding of the reality of C++. Example include:

- Impact of optimization on predictability/security.
- Exception handling vs. error codes (cost/performance).
- Compile-time and run-time impact of Modules and Contracts.

6. Recommended Action Guidelines for C++29

Given the severe constraints on committee time and implementation resources for C++29, the DG strongly encourages adherence to the following procedural guidelines in addition to Section 5 Process of P2000 as guidance to the chairs and members:

- **Safety and Profiles Framework:** A suggested roadmap for this cycleis detailed in the standalone paper: **P3970r0**: **Profiles and Safety: a call to action**.
- **Encouraging Convergence:** Chairs are asked to act as matchmakers to help merge competing proposals See **P4024R0: Guidance on Building Consensus and Converging Proposals**.
- **AI Authorship Guidelines:** Human authors should always remain the 'intelligence of record.' See **P4023R0: Strategic Direction for AI in C++: Governance, and Ecosystem**.