Initial draft proposal for core language UB white paper: Process and major work items

Doc #	P3656 R0
Authors	Herb Sutter, Gašper Ažman
Date	2025-03-10
Audience	EWG

Abstract: Background and scope

At Hagenberg 2025-02, EWG encouraged the following work:

Poll: Pursue a language safety white paper in the C++26 timeframe containing systematic treatment of core language Undefined Behavior in C++, covering Erroneous Behavior, Profiles, and Contracts. Appoint Herb and Gašper as editors.

SF	F	N	A	SA
32	31	6	4	4

Details about white papers are available at SD-4 > Technical specifications and white papers.

The first step is to get EWG input and work to consensus on process and method:

- **Process: How we intend to work and gain consensus.** This paper proposes EWG approval (possibly in telecon) for each item to be adopted into the white paper working draft. EWG might direct some specific discussion happen in an SG first, for example to see a proposed addition in SG23 (Safety and Security) before bringing it to EWG.
- **Major work items: What things will need to be done.** This paper proposes four main work items to enumerate and address core language UB.

1. Proposed process

This paper proposes that we:

1. start with an empty white paper;
2. iteratively get EWG approval (possibly in telecon) for each item to be adopted into the white paper working draft; and
3. finally as usual get EWG approval to forward the result to CWG for wording and then plenary for approval to send the white paper to SC 22 for their publication vote.

We suggest that to maintain the draft white paper we:

- Have a public GitHub repo.
- Use pull requests: Iterations occur on the PR. EWG can review a PR and approve merging.
- Use issues: This tracks rationale, can link to supporting papers, etc.

For each meeting:

- Tag a release.
- Publish the current draft white paper as a Pxxxx Rnext paper, with a list of changes from Rprev (export from GitHub) and issues.
- Request WG 21 member review comments, to be submitted as GitHub issues.

Notes:

- For a given proposed addition, EWG might direct the author to get feedback from an SG, for example to get security feedback in SG23 (Safety and Security) and bring the possibly-updated paper together with that feedback also to EWG.
- This section says "EWG" because the mandate for this work originated in EWG, and the mandate scope was a white paper on UB in the *core language* rather than in the standard library. However, if we find areas that touch library, of course we should also include LEWG for those.

2. Proposed major work items

This paper proposes four main work items.

2.1 List cases: Enumerate language UB

Goal: End up with a complete list of all language UB.

Today, just searching for "undefined" isn't enough to find all UB in the standard and systematically list it. We need to systematically list core language UB throughout the standard.

Thoughts on how:

- Crowdsource: Call for volunteers to sign up for a clause.
- Check: Against other (sometimes partial) sources, notably Shafik's draft paper, P3100 authors' list so far, compilers' `constexpr` evaluator checks.
- Review: Round of review with CWG experts (or by CWG?) – what did we miss?

At this stage we can do basic categorization, for example:

- Which ones are security-related? Have security experts tag which have security impact always, never, or sometimes.
- Which are efficiently locally diagnosable? Note the type of information needed to diagnose violations, for example whether a compiler can see it locally vs. there is escaped information that requires extra state and sanitizer-like tools.

Specific suggestion: Add a LaTeX `\tag{}` for UB right in the working draft sources (spelling TBD, ask `edit` reflector experts).

- Since adding LaTeX tags can be done without disturbing the generated standard, we can start adding them via PRs to the IS working draft itself.
- This allows them to be maintained together with the IS and minimize getting out of sync.
- This also allows easily generating a non-normative UB Annex in the standard (if we want).

2.2 List tools: Create "menu of tools"

Goal: Get agreement on the options we can choose from to deal with a given case of UB.

For each case of UB, the key question is "what do we do about it?" We want to generate a list of possible options we can pick from to decide how to handle each case of UB.

The Hagenberg 2025-02 EWG poll already suggested three possible basic tools:

- Erroneous Behavior (already in C++26)
- Profiles (not in C++26, white paper will need to add or reference, e.g., P3589R2)

- Contracts (already in C++26, except labels/grouping which white paper will need to add or reference, e.g., from P3400 (see 2.4))

So a candidate list might be:

- Make the UB well-defined
- Make the UB instead be EB in the language always (on by default, like uninitialized locals in C++26)
- Make the UB instead be EB sometimes via opt-in (to be grouped into actual profiles/labels later)
- Make the UB use a language `contract_assert` check (to be grouped into actual profiles/labels later)
    - Note: the previous two might be mergeable? opinions vary
- Additional ideas are welcome: Papers please!

Do we want to allow:

- nondeterministic heuristics?
- hardware support?

Finally, if possible we should determine guidelines for when to use each tool, particularly what are the considerations for each:

- performance considerations
- adoption hurdles like crashes
- etc.

2.3 Apply: Take a first pass at penciling in which tool to use for each UB case

Papers please! This requires design.

Could do it one clause at a time

- Crawl-walk-run: Get feet wet with one clause first.
- Start with some interesting clause (e.g., wherever array subscript out-of-bounds UB would be covered).

Each UB case could be a GitHub issue with its dedicated discussion thread

- this can link to your papers
- maintains a record of the discussion of this EB case to document rationale
- also separates the discussion of this EB case from the others (hopefully more manageable than a single email list that has a huge volume of hard-to-sort traffic)
- lets people vote on individual comments/answers

2.4 Group: Group UB cases (contract labels / profile names)

Identify cohesive groups of UB that want to be addressed together

- "Establish cohesion in solution spaces."
- Can overlap: A given case could be under more than one group.

Ideally crowdsource initial suggestions?

- Make a list of possible labels/profiles.
- Do another spreadsheet "voting table" for groupings.

What's next

Get EWG telecon discussion and direction on the above.