Relaxing Packaging Rules for Exceptions Thrown by Parallel Algorithms - Proposed Wording (Revision 1)

1 Introduction

N4157 described the rationale for changing a future revision of N4105 to relax exception packaging rules. Specifically, the change permits an implementation to throw an exception that is not an exception_list if only one invocation of an element access function throws an exception. Unfortunately, the proposed wording in N4157 did not completely fix the problem. This document proposes new rewording.

2 Proposal

Edit Section 1.3.1, paragraph 3 as follows:

Parallel algorithms access objects indirectly accessible via their arguments by invoking the following functions:

- All operations of the categories of the iterators that the algorithm is instantiated with.
- Functions on those sequence elements that are required by its specification.
- User-provided function objects to be applied during the execution of the algorithm, if required by the specification.
- Operations on those function objects required by the specification. [Note: see clause 25.1 of C++ Standard Algorithms Library]

These functions are herein called element access functions.
Edit Section 3.1 paragraph 2, as follows:

If the execution policy object is of type `sequential_execution_policy` or `parallel_execution_policy`, the execution of the algorithm terminates with an `exception_list` exception. The exception shall be an `exception_list` containing all uncaught exceptions thrown during the invocations of element access functions, or optionally the uncaught exception if there was only one shall be contained in the `exception_list`.

[ Note: For example, the number of invocations of the user-provided function object in `for_each` is unspecified. When `for_each` is executed sequentially, if an invocation of the user-provided function object throws an exception, `for_each` can terminate with the uncaught exception, or throw an `exception_list` containing the original exception. Only one exception will be contained in the `exception_list` object. – end note ]