Project: ISO JTC1/SC22/WG21: Programming Language C++
Doc No: WG21 **P0895R0**
Date: 2018-01-31
Reply to: Nicolai Josuttis (nico@josuttis.de), Tony van Eerd (tvaneerd@gmail.com)
Audience: LEWG, LWG
Prev. Version:

Renaming cell<> to latest<>, Rev0

P0561R3 proposes a class for deferred reclamation so that multiple threads can easily access the "latest snapshot" of a state. Currently the class where all snapshots are managed is called a `cell` (with a `basic_cell`<> class template in header <`cell`>).

This paper proposes a different name for cell. The reason is that there are different meanings of cell and the most obvious one is counter-intuitive, because we talk about a container-like class, holding/managing multiple snapshots; while the intuitive meaning of "cell" is the smallest unit, which sounds more like being an *element* of a container.

Although the name works if you think of a electrochemical cell, holding and yielding a state; but even then it is confusing that the cell can handle multiple states at the same time.

We therefore propose to use a more intuitive name. After a lot of very constructive discussion, the name we proposed is:

`latest<>`

The authors of the original paper, Andrew Hunter and Geoffrey Romer are willing to support it.

The resulting API for some configuration would be as follows:

```
#include <snapshot>
…
std::latest<Config> config;                    // was:  std::cell<Config>
…
snapshot_ptr<const Config> currentConfig = config.get_snapshot();
```

The fact that you can read `std::`**`latest`**`<Config>` as "latest Config" is very intuitive.

As a consequence the concrete proposal is to adjust P0561 as follows:

- Rename the **header** to **`<snapshot>`**
    - As the snapshort is the most intuitive key concept here, this seems to be more intuitive than a header <`latest`>, which, however, is also possible.
- Rename the **basic class template** to **`basic_latest<>`**
- Rename the **alias template** to **`latest`**

with all corresponding fixes like changing the constructor names and so on.

Acknowledgements

Thanks to the authors of the proposed concept and all those who helped to find the name proposed here.

Feature Test Macro

No need for a feature macro because the goal of this paper is to change a name of a library feature that is not standardized yet. It should be used in a new/joined revision of P0561.