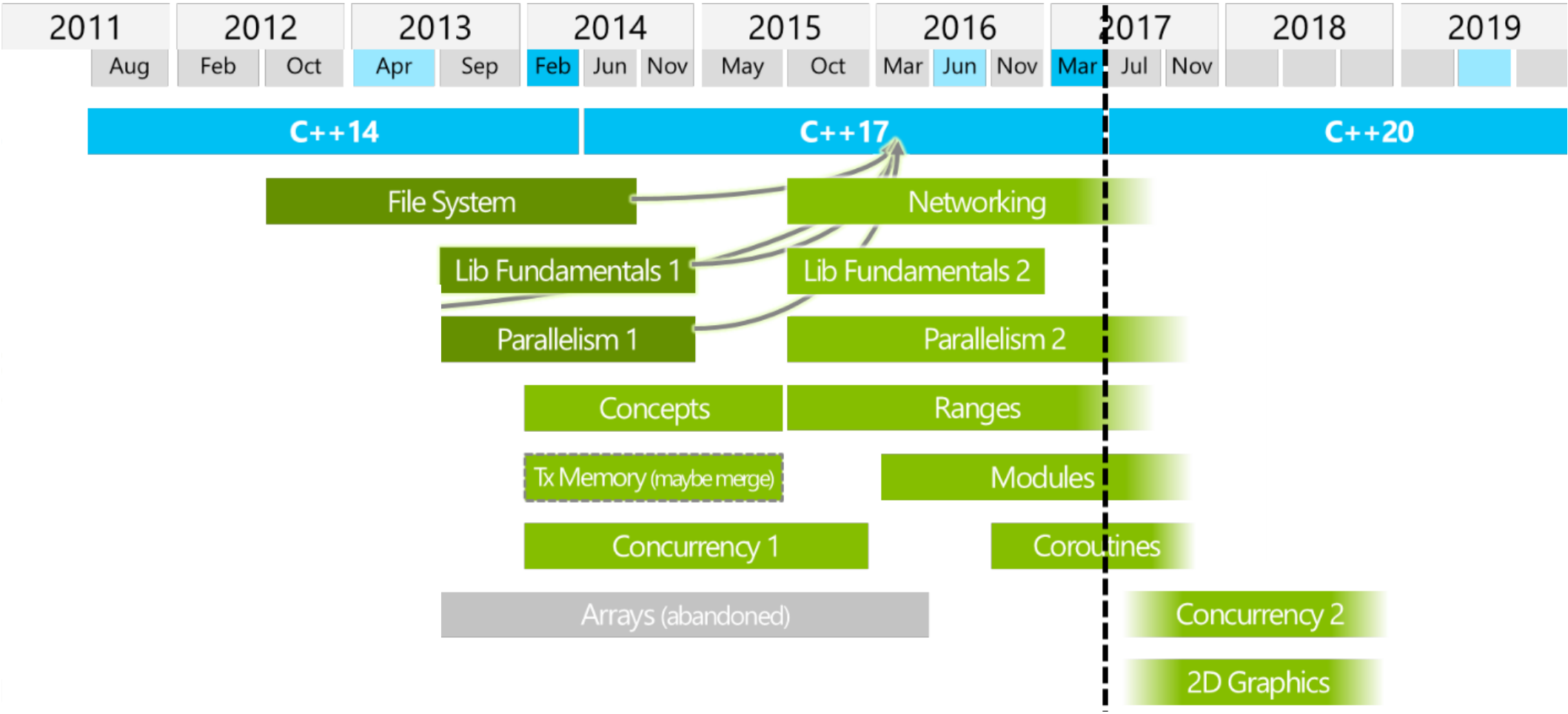
The C++ Roadmap

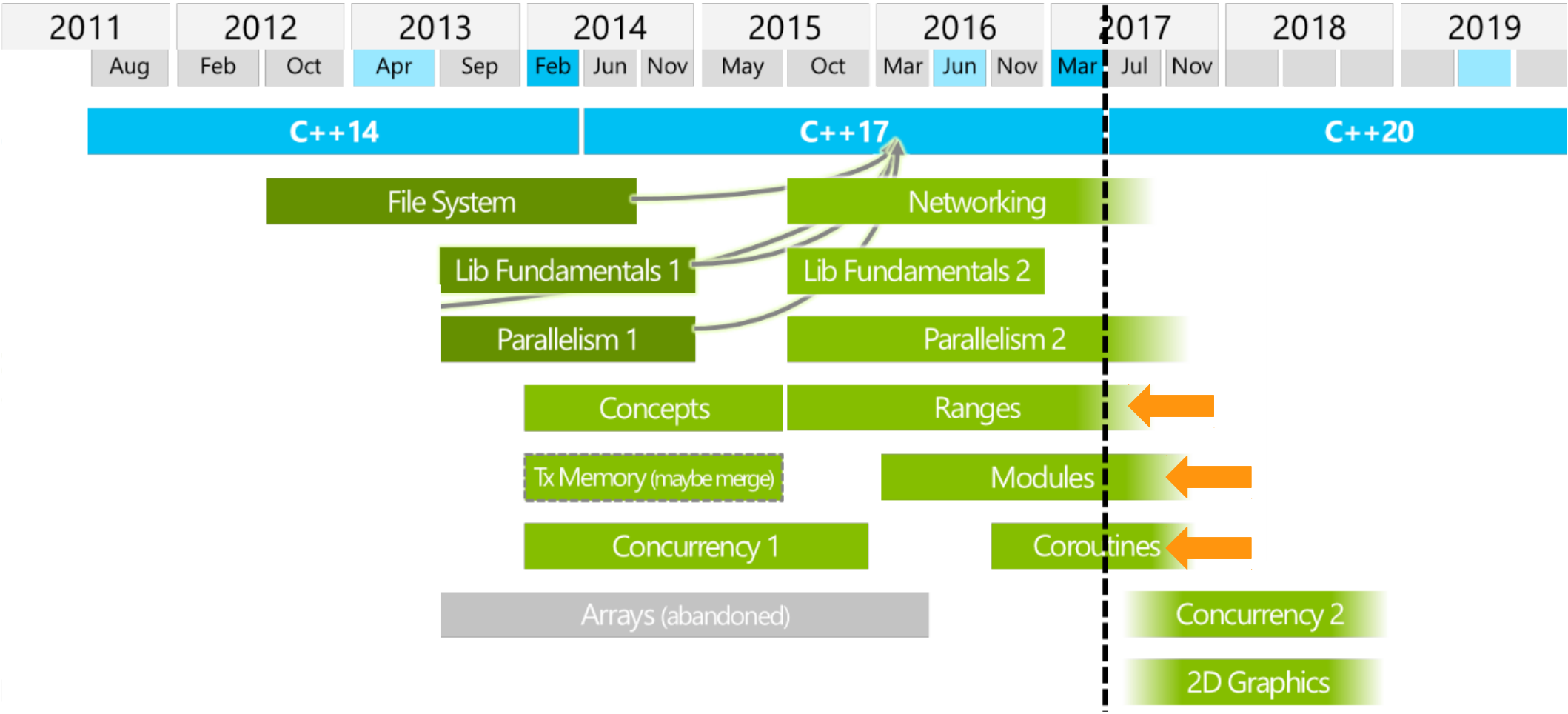
The 20s and beyond



Source: isocpp.org

The C++ Roadmap

The 20s and beyond



Source: isocpp.org

Ranges - N4128

Author: Eric Niebler

Documentation at https://ericniebler.github.io/range-v3/

```
#include <iostream>
#include <range/v3/all.hpp>

int main()
{
    using namespace ranges;
    int sum = accumulate(
        view::ints(0)
      | view::remove_if([](int i) { return i % 2 == 0; })
      | view::transform([](int i) { return i * i; })
      | view::take(10), 0);

    std::cout << "1² + 3² + … + 19² = " << sum << '\n'; // 1330
}
```

Source: isocpp.org

Modules - N4214

Author: Gabriel Dos Reis

Source: cppcast.com

```
#include <stdio>            // does it affect date.h?
#include “calendar/date.h” // what is in there?

int main() {
    using namespace chrono;
    Date date{ 10, month::may, 2017 };
    std::cout << “Today is “ << date << '\n';
    return 0;
}
```

Modules - N4214

Author: Gabriel Dos Reis

Source: cppcast.com

```
import std.io;
import calendar.date;

int main() {
    using namespace chrono;
    Date date{ 10, month::may, 2017 };
    std::cout << “Today is “ << date << '\n';
    return 0;
}
```

Modules - N4214

```
import std.io;
import std.string;

module calendar.date;

namespace chrono {
    export {
        struct Date {
            // conventional members
        };

        // other exported definitions
    }

    export std::ostream& operator<<(std::ostream&, const Date&);
    export std::string to_string(const Date&);
}
```

Coroutines - N3985

Author: Gor Nishanov “Gor-Routines”

Source: events.yandex.ru

```
#include <iostream>
#include <coroutines>

auto fib() {
    int a = 0, b = 1;
    while (true) {
        co_yield b;
        a = b;
        b = a + b;
    }
}

int main() {
    for (auto n : fib()) {
        std::cout << n << '\n';
    }
    return 0;
}
```

Coroutines - N3985

Author: Gor Nishanov “Gor-Routines”

Source: events.yandex.ru

```
#include <iostream>
#include <coroutines>

struct Node {
    public:
        int data;
        unique_ptr<Node> left;
        unique_ptr<Node> right;
}

auto flatten(Node* node) -> generator<int> {
    if (node == nullptr) return;
    co_yield flatten(node->left);
    co_yield data;
    co_yield flatten(node->right);
}
```

Info - Selbergroß