C++ Club

27 July 2017

Presenting effectively in meetings

Dirk Haun, ACCU 2017

Figure 1: 80%

Roger Orr - Making Templates Easier (ACCU)

Meetup | Video

- Templates
- `enable_if`
- Concepts

CLion 2017.3 roadmap

Post

- Valgrind integration
- Remote development support
- Better C++ language support
- Multiple toolchains
- Custom pretty printers for debugger
- GDB 8, LLDB 5
- Gutter icons for unit tests
- Improvements and fixes for Google Test and Catch

Thoughts on Metaclasses - Herb Sutter - Keynote [ACCU 2017]

YouTube | Post

- Depends on:
  - C++17
  - concepts (C++20)
  - `constexpr`, `if constexpr`
  - compile-time meta-programming (P0589, P0633)
  - reflection (P0194, P0385, P0578, P0590, P0598)
- Start watching at 18m

Metaclasses (cont.)

Reflection

```
$T, $expr
```

Compile-time programming

```
constexpr {
    for (auto m: $T.variables())
        if (m.name() == "xyzzy")
            -> { int plugh; }
}
```

The Detection Idiom - a simpler way to SFINAE - Marshall Clow [ACCU 2017]

YouTube

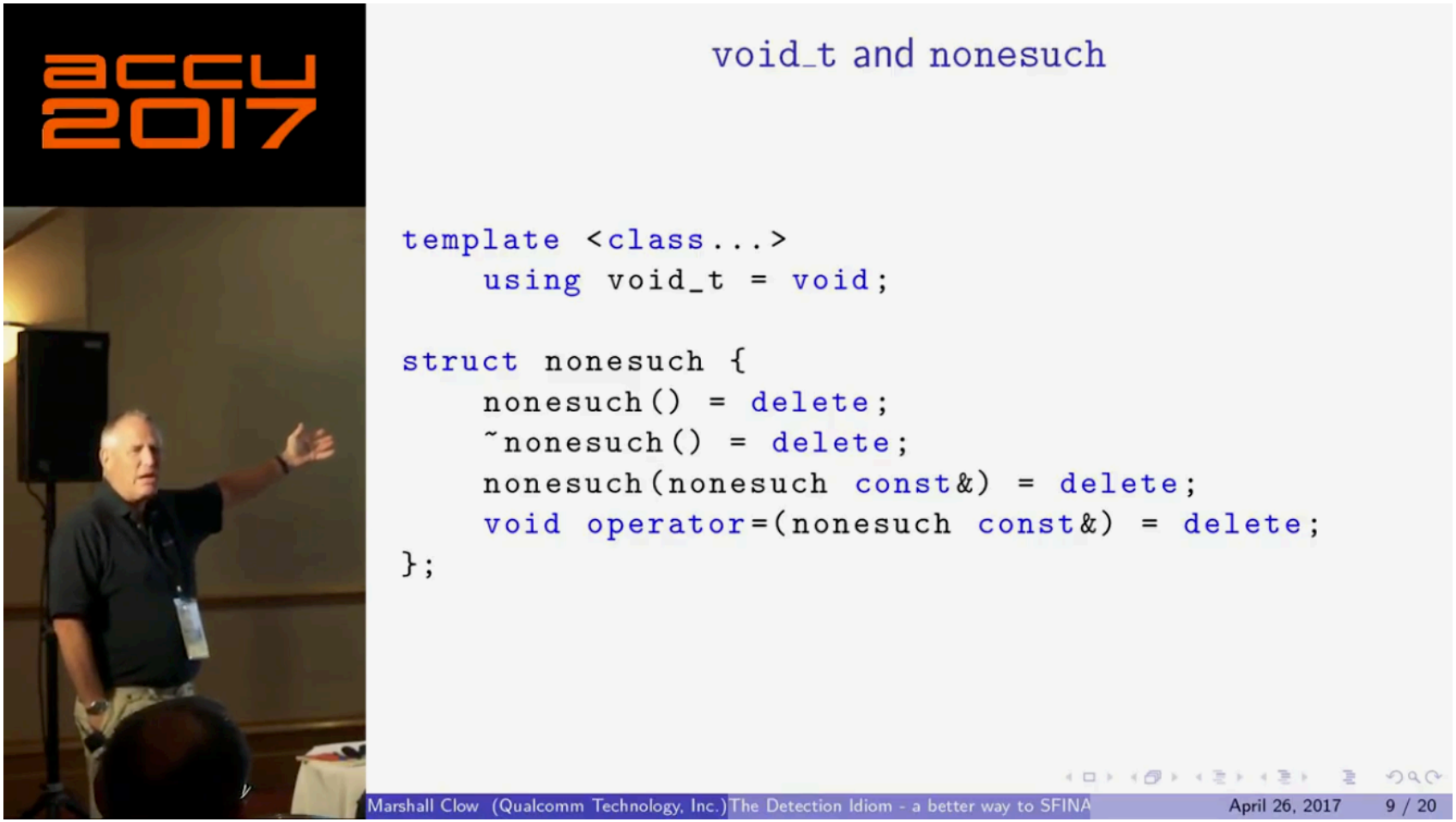

Figure 2: Inline 70%

The Detection Idiom - a simpler way to SFINAE - Marshall Clow [ACCU 2017]

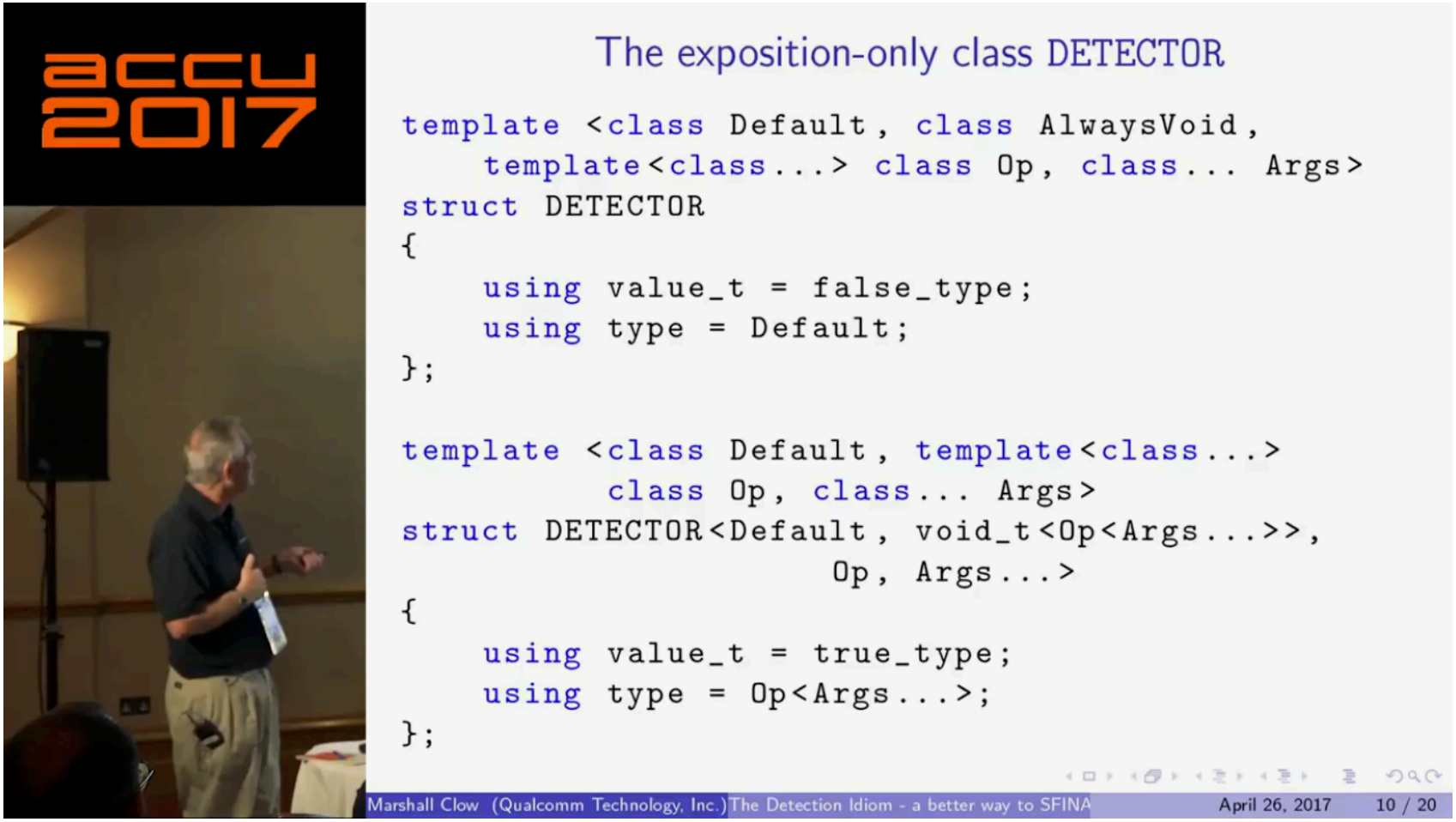

Figure 3: Inline 70%

The Detection Idiom - a simpler way to SFINAE - Marshall Clow [ACCU 2017]

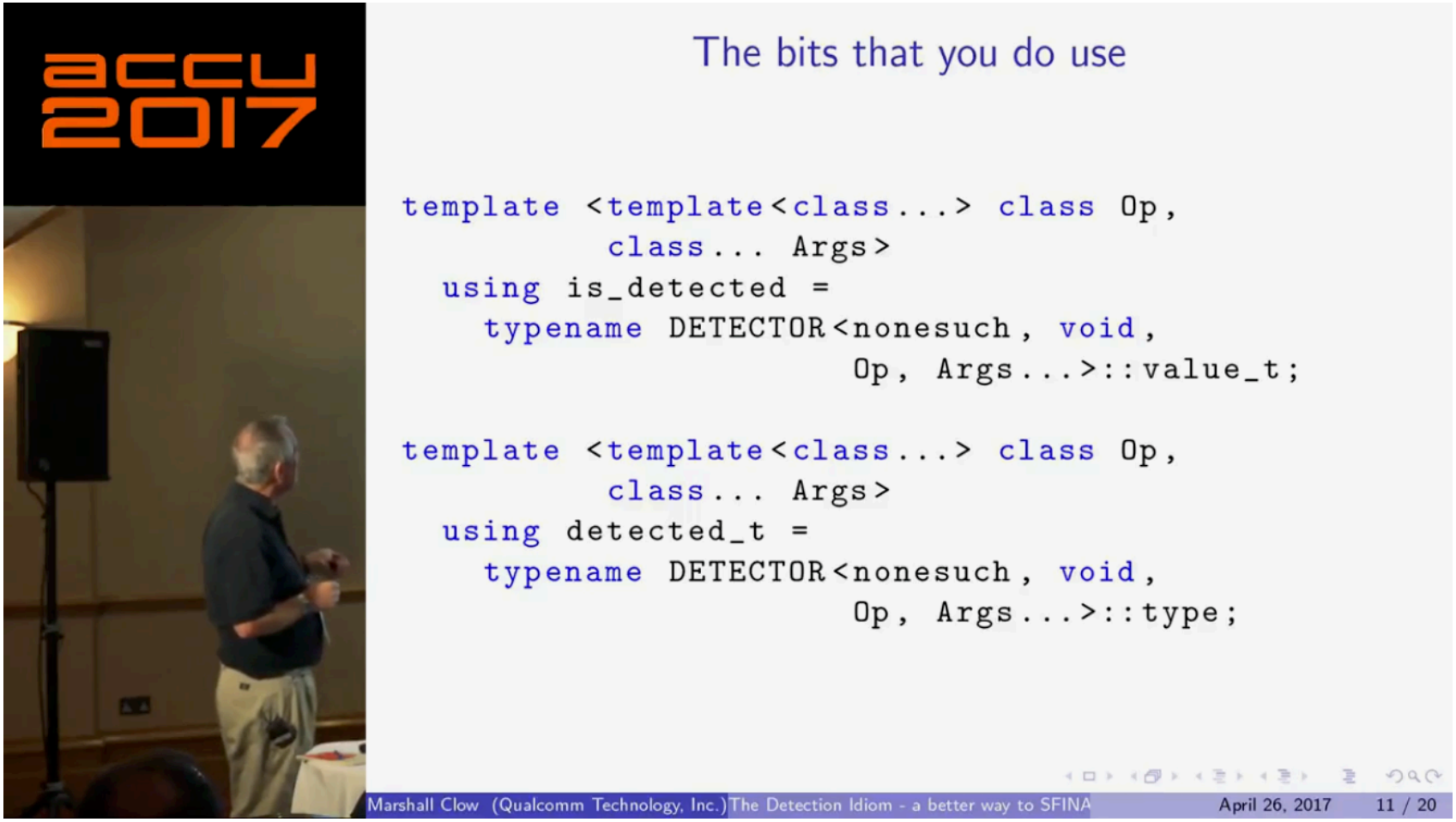

Figure 4: Inline 70%

The Detection Idiom - a simpler way to SFINAE - Marshall Clow [ACCU 2017]

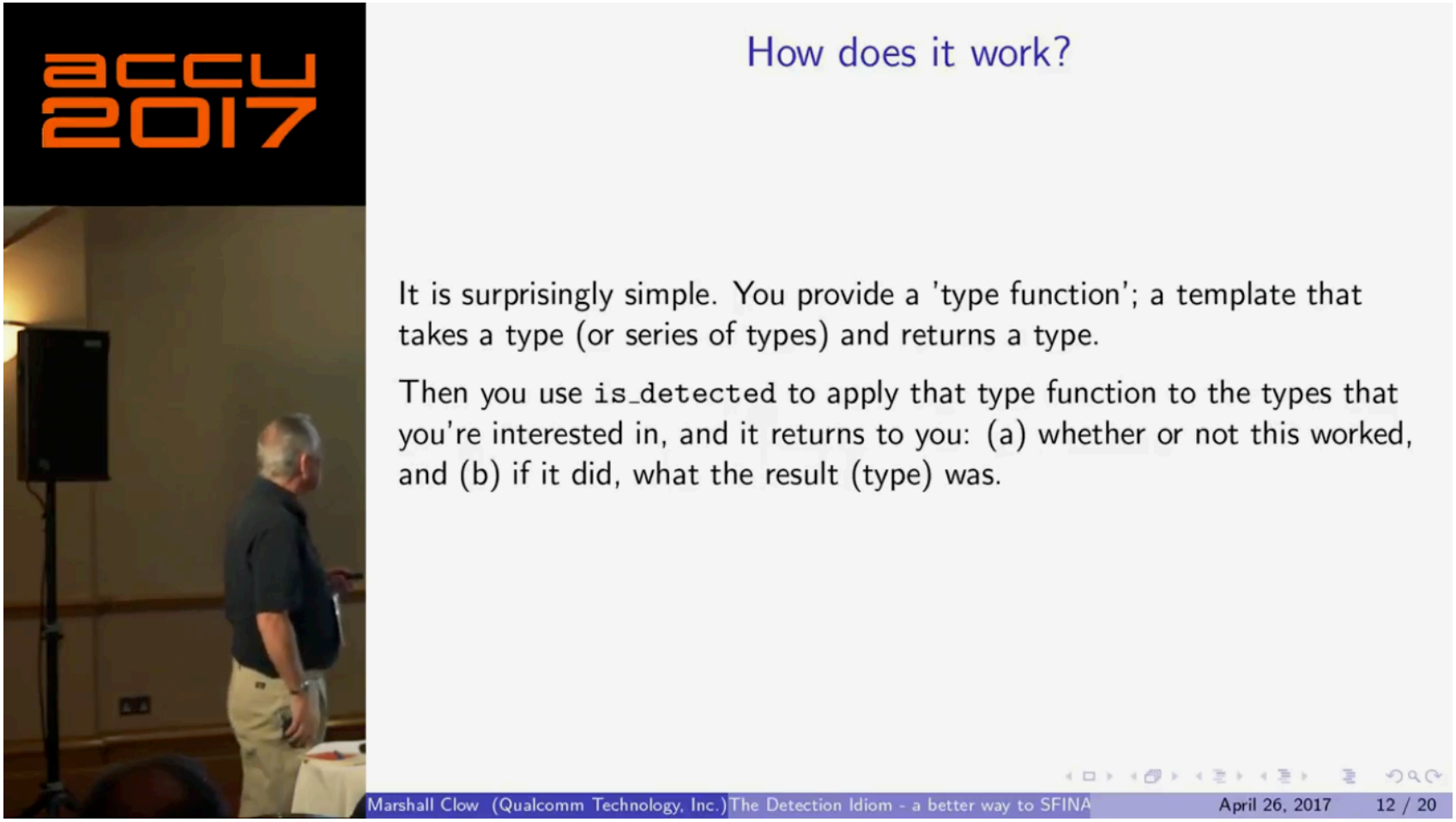

Figure 5: Inline 70%

The Detection Idiom - a simpler way to SFINAE - Marshall Clow [ACCU 2017]

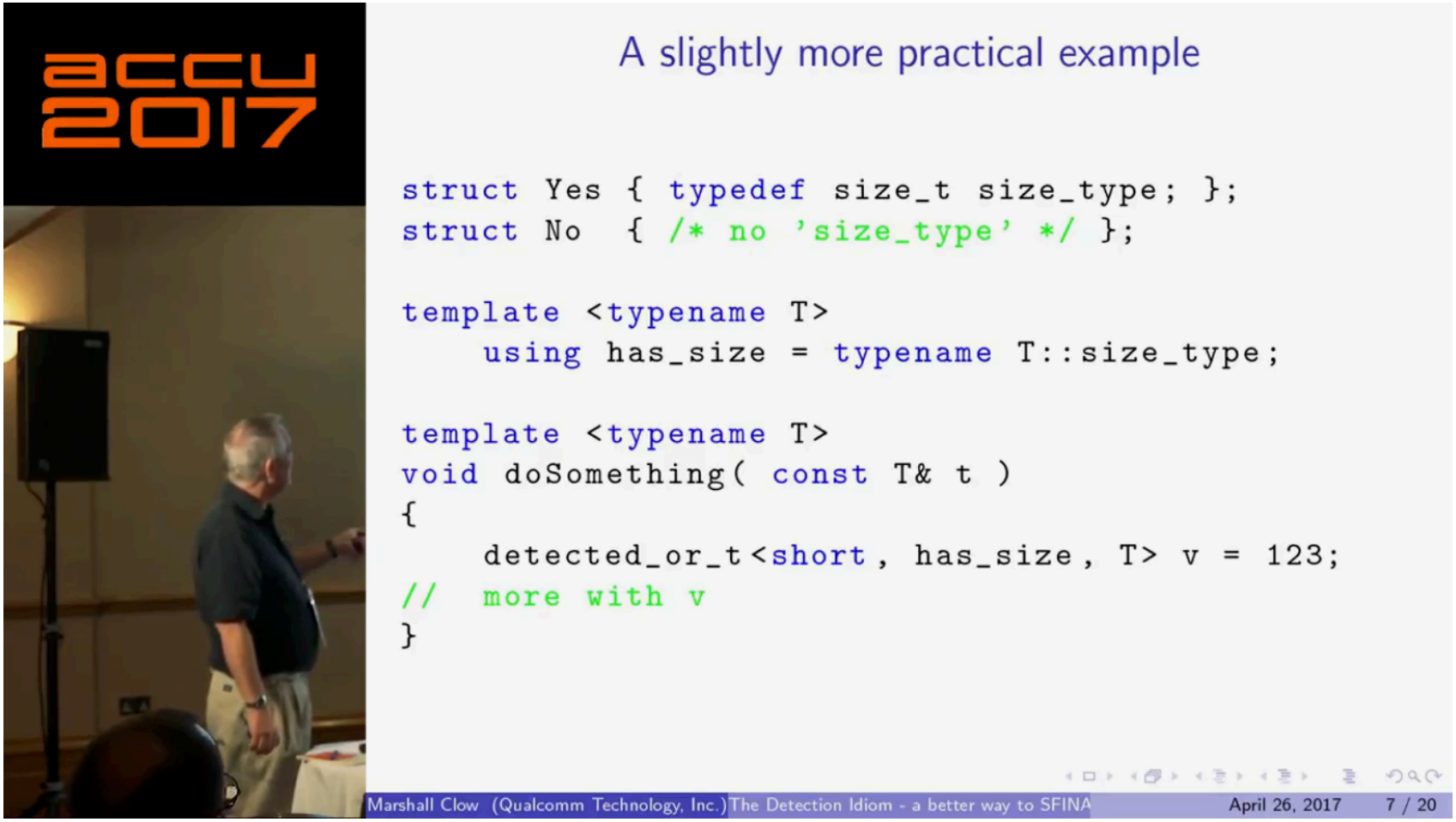

Figure 6: Inline 70%

The Detection Idiom - a simpler way to SFINAE - Marshall Clow [ACCU 2017]

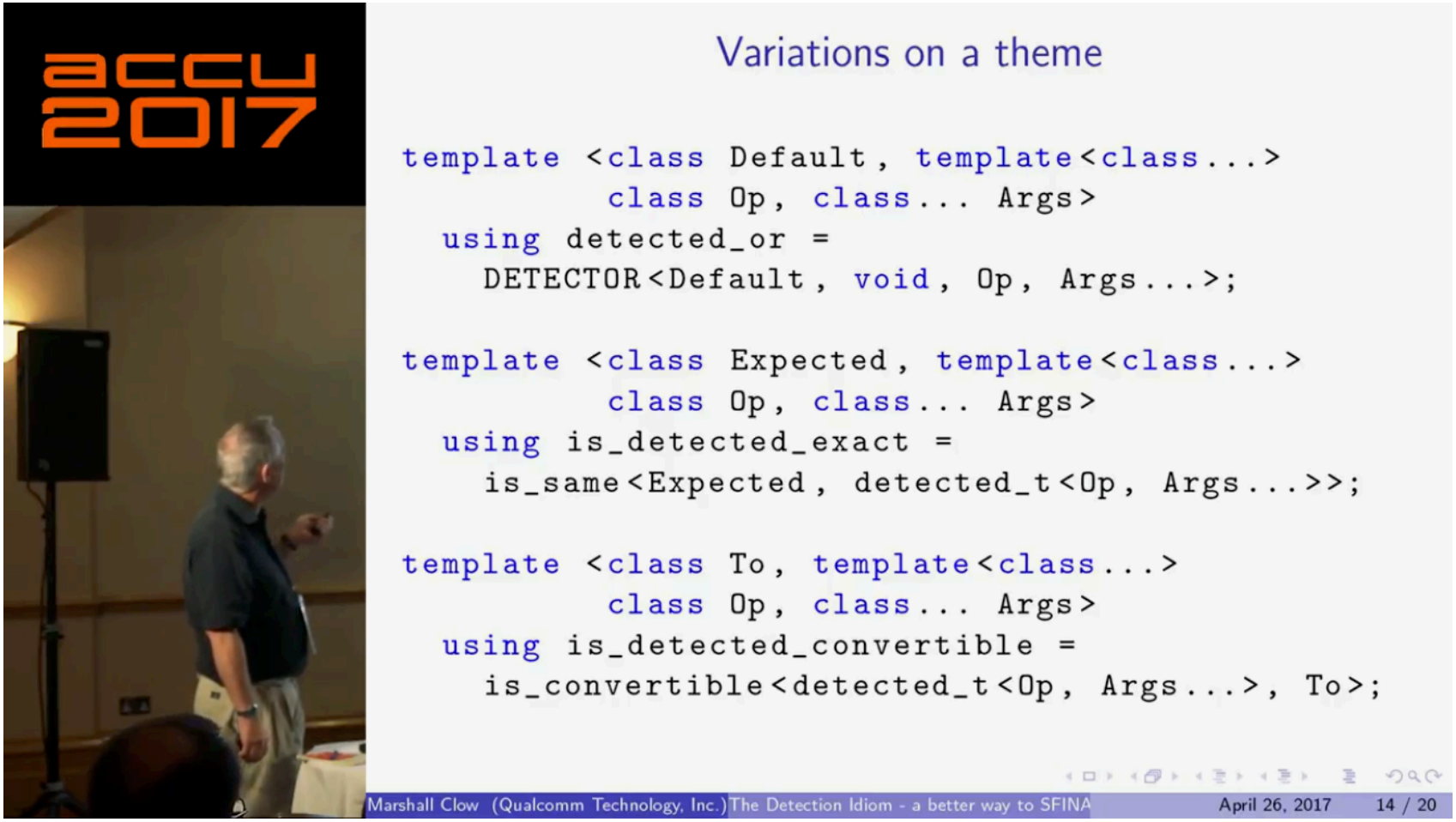

Figure 7: Inline 70%

The Detection Idiom - a simpler way to SFINAE - Marshall Clow [ACCU 2017]

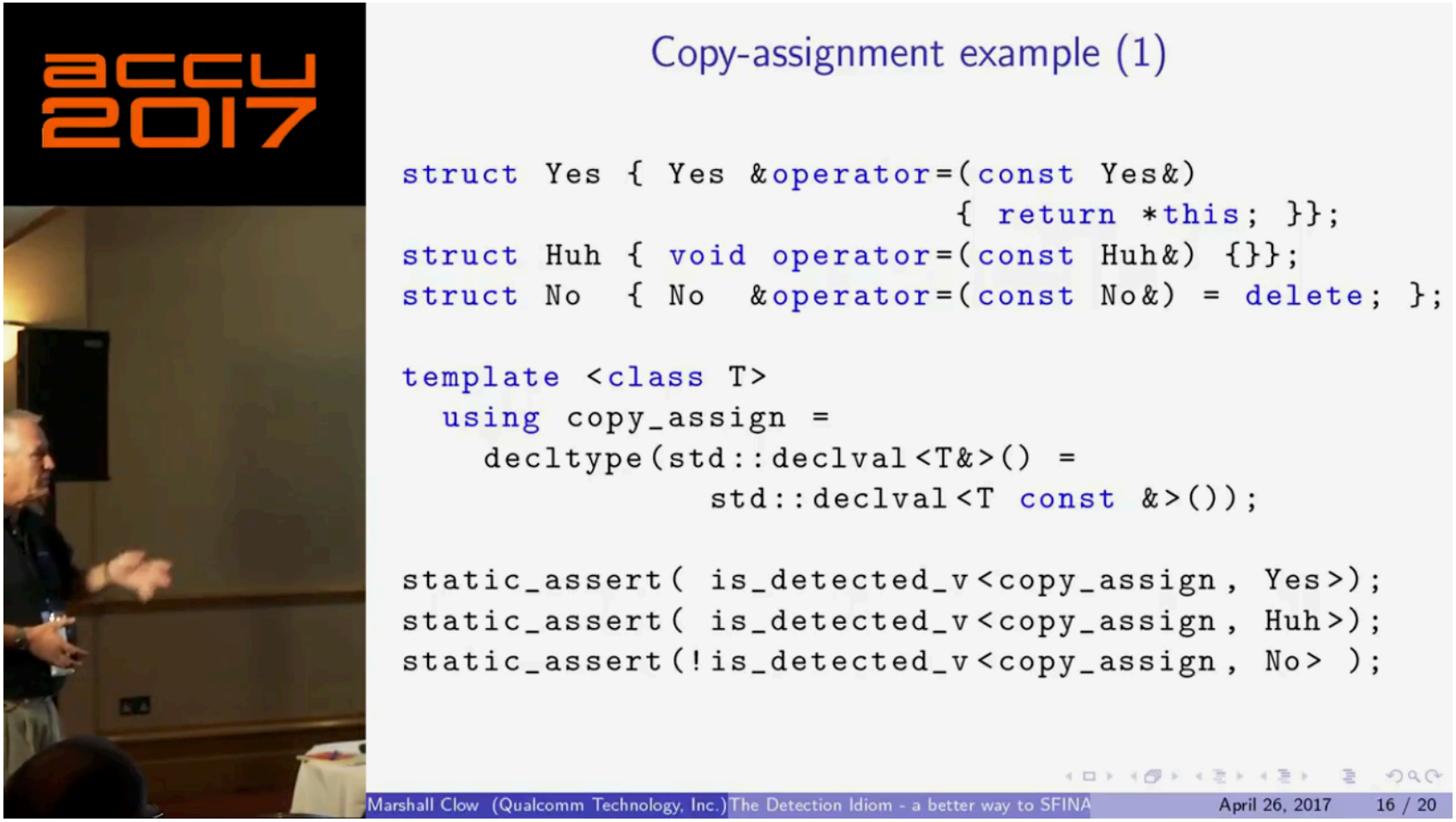

Figure 8: Inline 70%

The Detection Idiom - a simpler way to SFINAE - Marshall Clow [ACCU 2017]

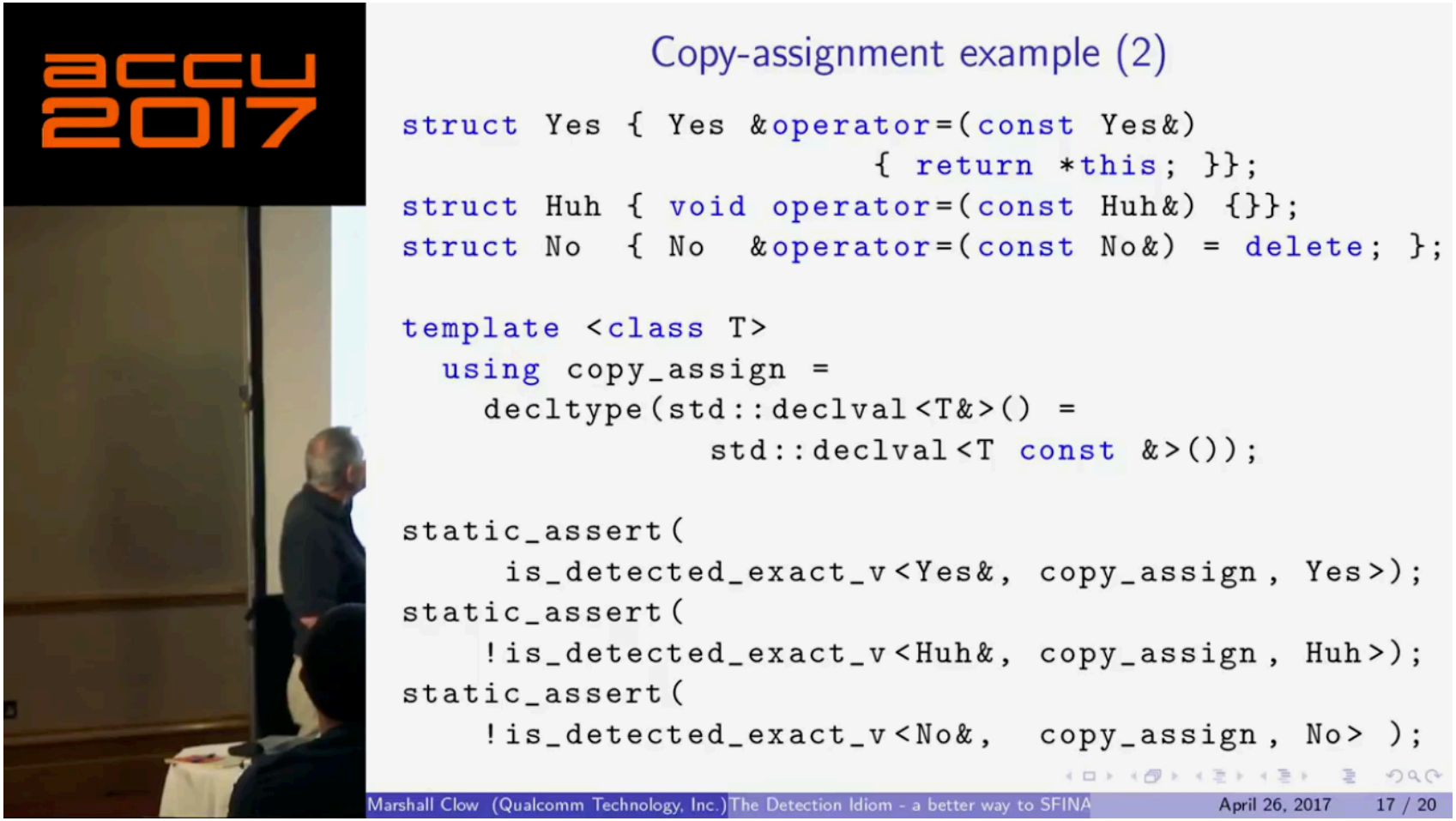

Figure 9: Inline 70%

GitHub

Pystring is a collection of C++ functions which match the interface and behavior of python's string class methods using `std::string`. Implemented in C++, it does not require or make use of a python interpreter. It provides convenience and familiarity for common string operations not included in the standard C++ library. It's also useful in environments where both C++ and python are used.

Overlapping functionality (such as `index` and `slice`/`substr`) of `std::string` is included to match python interfaces.

Video

- This is an “update” to the book “Modern C++ Design” by Andrei Alexandrescu
- Discusses updated template metaprogramming techniques:
  - policy-based class design;
  - replacing multiple templates with variadic templates.

Modern C++ Design reloaded (cont.)

“Concept”-based polymorphism (1)

```
class drawable_concept{
  public:
    drawable_concept() = default;
    virtual ~drawable_concept() = default;
    virtual void draw() = 0;
};

template<class T>
class drawable_model : public drawable_concept{
    T model_;
    public:
    drawable_model(T const& model) : model_(model){}
    void draw(){
        model_.draw();
    }
    ~drawable_model() = default;
};
```

Figure 10: 150%

Modern C++ Design reloaded (cont.)

“Concept”-based polymorphism (2)

```
class drawable{
    std::unique_ptr<drawable_concept> object_;
    public:
    template <typename T>
    drawable(const T& x) : object_(
        new drawable_model<T>(x))     { }
    void draw(){
        object_->draw();
    };
};

struct my_widget{
    void draw(){};
};
int main(){
    drawable d{my_widget{}};
}
```

Bjarne Stroustrup on *Mapping the Journey* podcast

Transcript

Post

- Microsoft GSL
- GSL-Lite - works with C++98 and C++03
- Quick overview of the classes available

The Ultimate Question of Programming, Refactoring, and Everything

by Andrey Karpov, PVS Studio

Post

- Tips and tricks on C++ programming
- Examples of bugs and bad practices
- Lots of code snippets
- Based on PVS-Studio diagnostics

&& == and, & == bitand

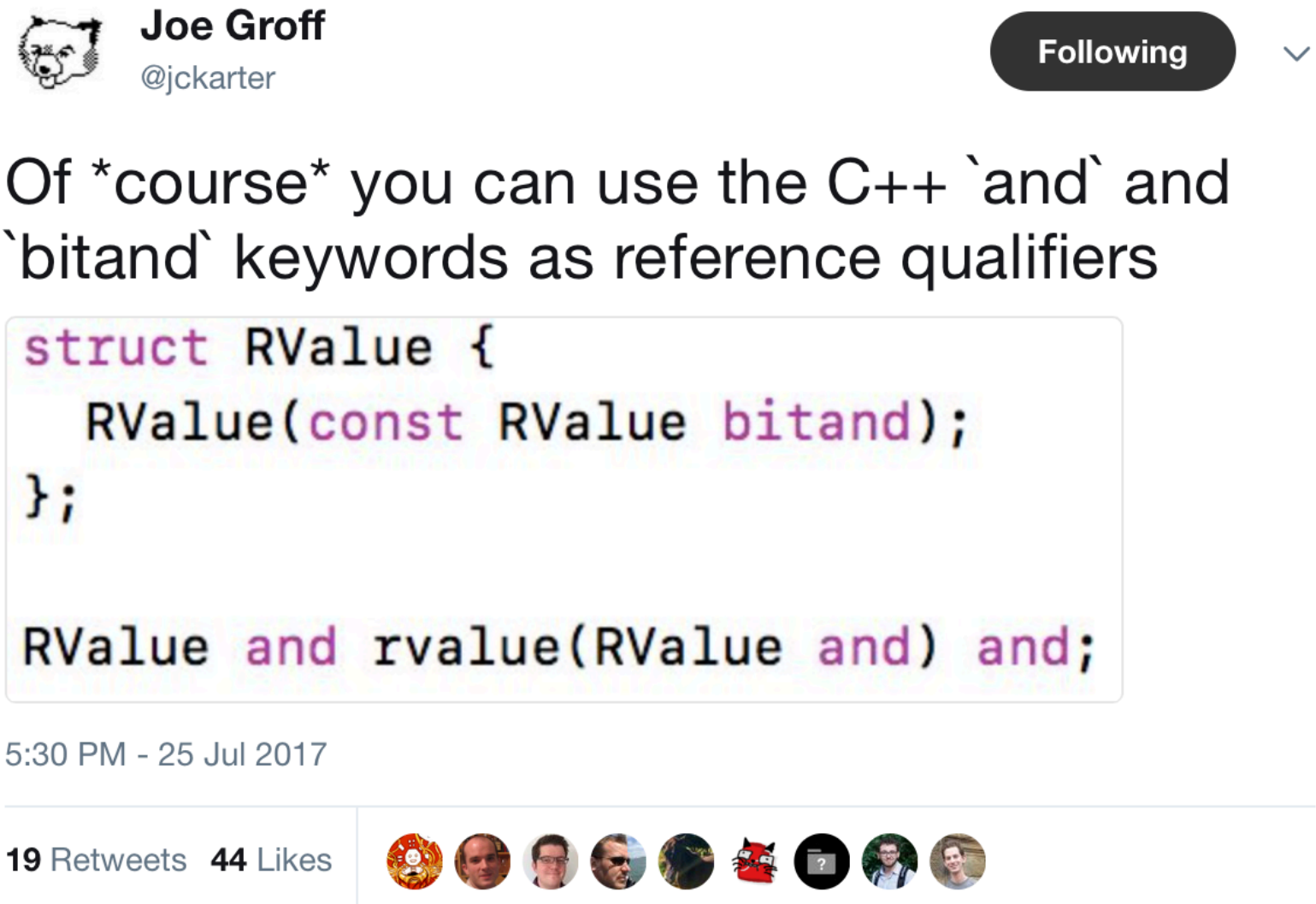

Figure 12: 70%

STL on friends