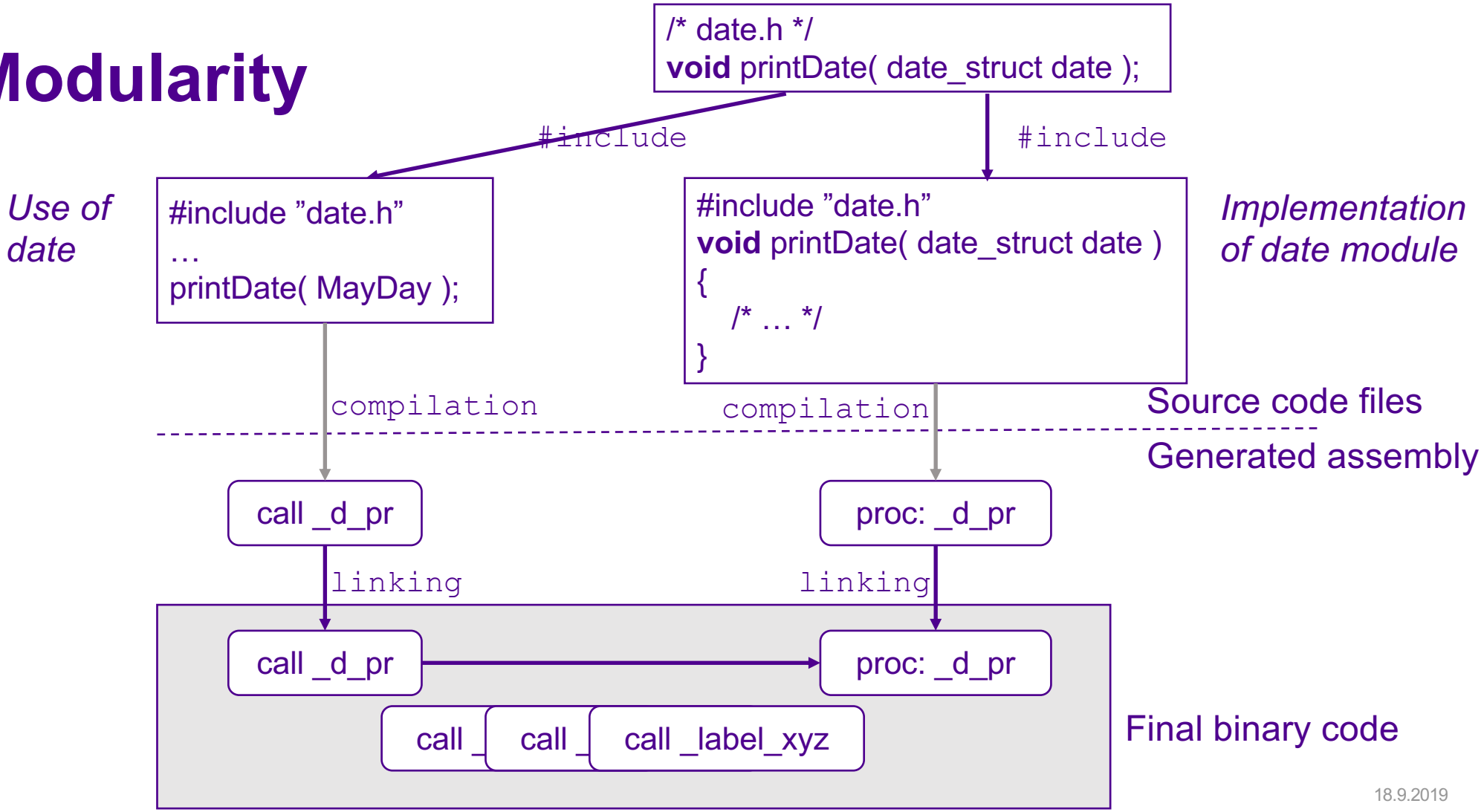
Tampereen yliopisto
Tampere University

Modularity
Namespaces

18.9.2019

Tampereen yliopisto
Tampere University

Modularity



Name conflicts

- Problem:
  - Same name used in different purposes ⇒ namespace pollution and name conflicts
- Solution: namespace

```
Date::print();
LibraryBook::print ();
```

Interface with namespace

```
#ifndef DATE_HH
#define DATE_HH

namespace Date {
    struct dateStruct { int d_, m_, y_; };
    dateStruct create( int day, int month, int year );
    void print( dateStruct d );
    ….
}
#endif
```

Advantages of namespaces

- Name conflicts decrease due to hierarchy
- Hierarchy visible in language syntax (operator ::)
- Short names can still be used inside a module (namespace)

Namespace std

- Standard defines one namespace: `std`, which includes the ISO C++ names

```
std::vector
std::cin
…
```

Short names: `using`

- `using` clause allows the names inside a certain namespace to be used in the current namespace

using clause

```
#include “date.hh”
#include <iostream>
void showDate( Date::dateStruct d )  {
    using std::cout;
    using std::endl;
    using namespace Date;
    cout << “Today is: ";
    print( d ); // Calls Date::print
    cout << endl;
}
```

Unnamed namespace

- Scope of a compilation unit

```
namespace {
        unsigned long int referenceCounter;
        void increaseReferenceCounter() {
        ++referenceCounter;
        …
        }
}
```