Экосистема языка C++, в болезни и в здравии

Anastasia Kazakova
JetBrains
@anastasiak2512

Agenda

1. C++ ? I’ve see some C with classes 20 years ago.
2. C++ tops areas
3. The state of C++ now
4. New vs Old. Conservatism ?

What is C++ today?

```
template<class T, int ... X>
T pi(T(X...));

int main() {
    return pi<int, 42>;
}
```

What is C++ today?

```
#define X(a) myVal_##a,
enum myShinyEnum {
#include "xmacro.txt"
};
#undef X

void foo(myShinyEnum en) {
    switch (en) {
        case myVal_a:break;
        case myVal_b:break;
        case myVal_c:break;
        case myVal_d:break;
    }
}
```

```
//xmacro.txt

X(a)
X(b)
X(c)
X(d)
```

What is C++ today?

```
$class interface {
    constexpr {
        compiler.require($interface.variables().empty(),
                         "interfaces may not contain data");
        for... (auto f : $interface.functions()) {
            compiler.require(!f.is_copy() && !f.is_move(),
                "interfaces may not copy or move; consider a"
                " virtual clone() instead");
            if (!f.has_access()) f.make_public();
            compiler.require(f.is_public(),
                "interface functions must be public");
            f.make_pure_virtual();
        }
    }
    virtual ~interface() noexcept { }
};
```

```
interface Shape {
    int area() const;
    void scale_by(double factor);
};
```

```
struct Shape {
    virtual int area() const = 0;
    virtual void scale_by(double factor) = 0;
    virtual ~Shape() noexcept {
    }
}
```

C++ per areas

C++ per areas

- Finances / Banking / Trading
- Embedded
- Games

C++ in Banking and Trading

C++ in Banking and Trading

- Language choices:
  - **Java** for the big enterprise systems, back end trading platforms etc.
  - **C++** for the low latency / high performance stuff
  - **C#** for front-end / desktop apps
  - **Python** for various scripting
- C++ is a primary choice
- Especially low latency trading and quantitive analytics
- Performance

C++ in Banking and Trading

Performance:

- Low latency, not quick throughput
- And safety
- Requires understanding of the compiler output

Carl Cook “When a Microsecond Is an Eternity: High Performance Trading Systems in C++” (CppCon 2017)

C++ in Banking and Trading

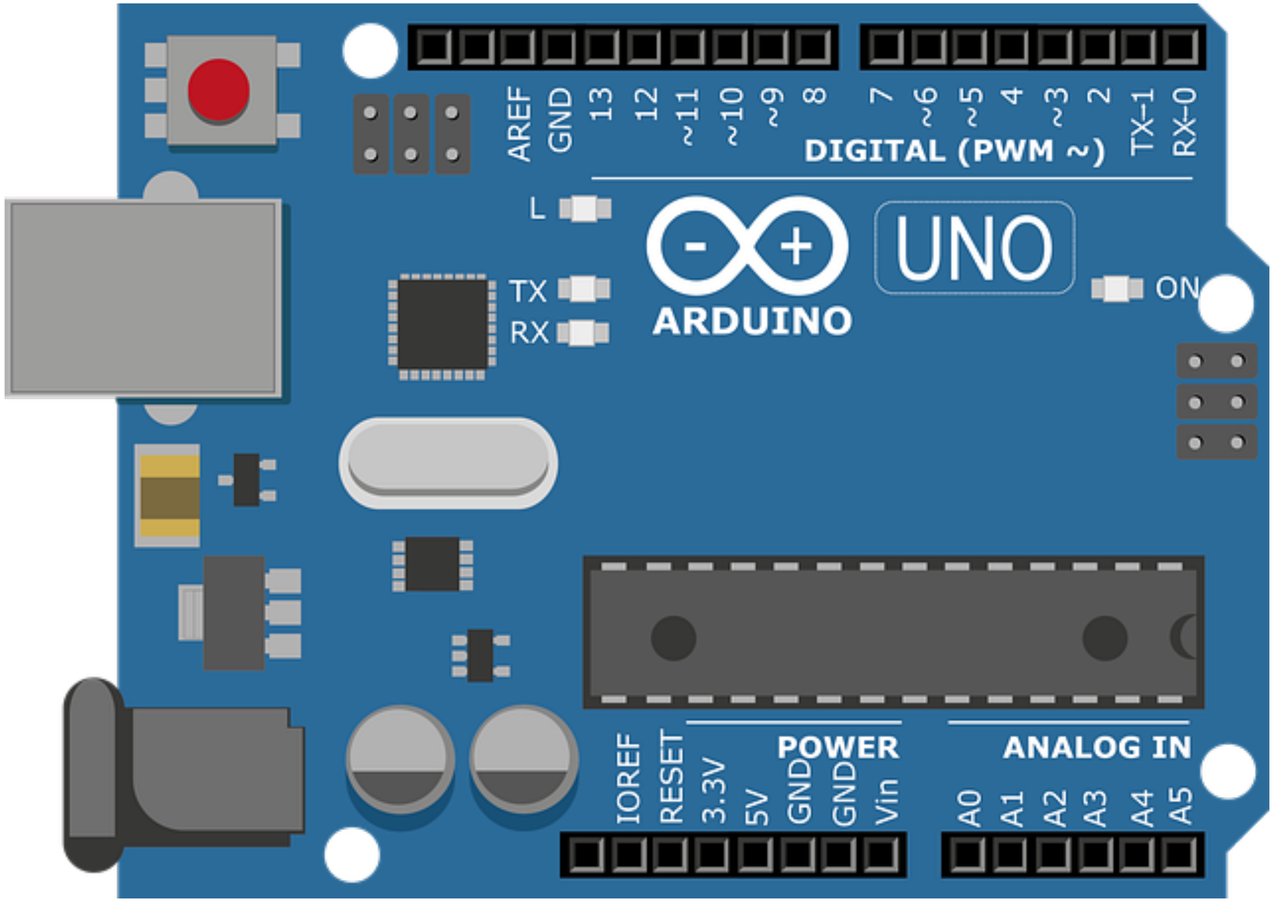
C++ usage:

- Allocations are important
- Exceptions are fine, if they don’t throw and not in the control flow
- Templates over virtual functions and branches
- Usage of low-level CPU instructions

Related ecosystem:

- Huge infrastructure, learning materials, wide expertise
- Lots of SDKs (CUDA, QuantLib)
- High cost of moving to the new technologies
- Affects clients

C++ in Embedded



C++ in Embedded

- Controlled by MCUs vendors
- Testing / Standards compliance / Certification tools
- Language choices:
  - C and C++, often more C than C++
  - Python, Lua, etc. for scripting, configurations, etc.
- Vendor’s compilers / debuggers / etc.

C++ in Embedded

C++ usage:

- Classes are C structs with function pointers
- Macros are everywhere
- Direct memory/registers access
- Data structures in memory are specifically packed

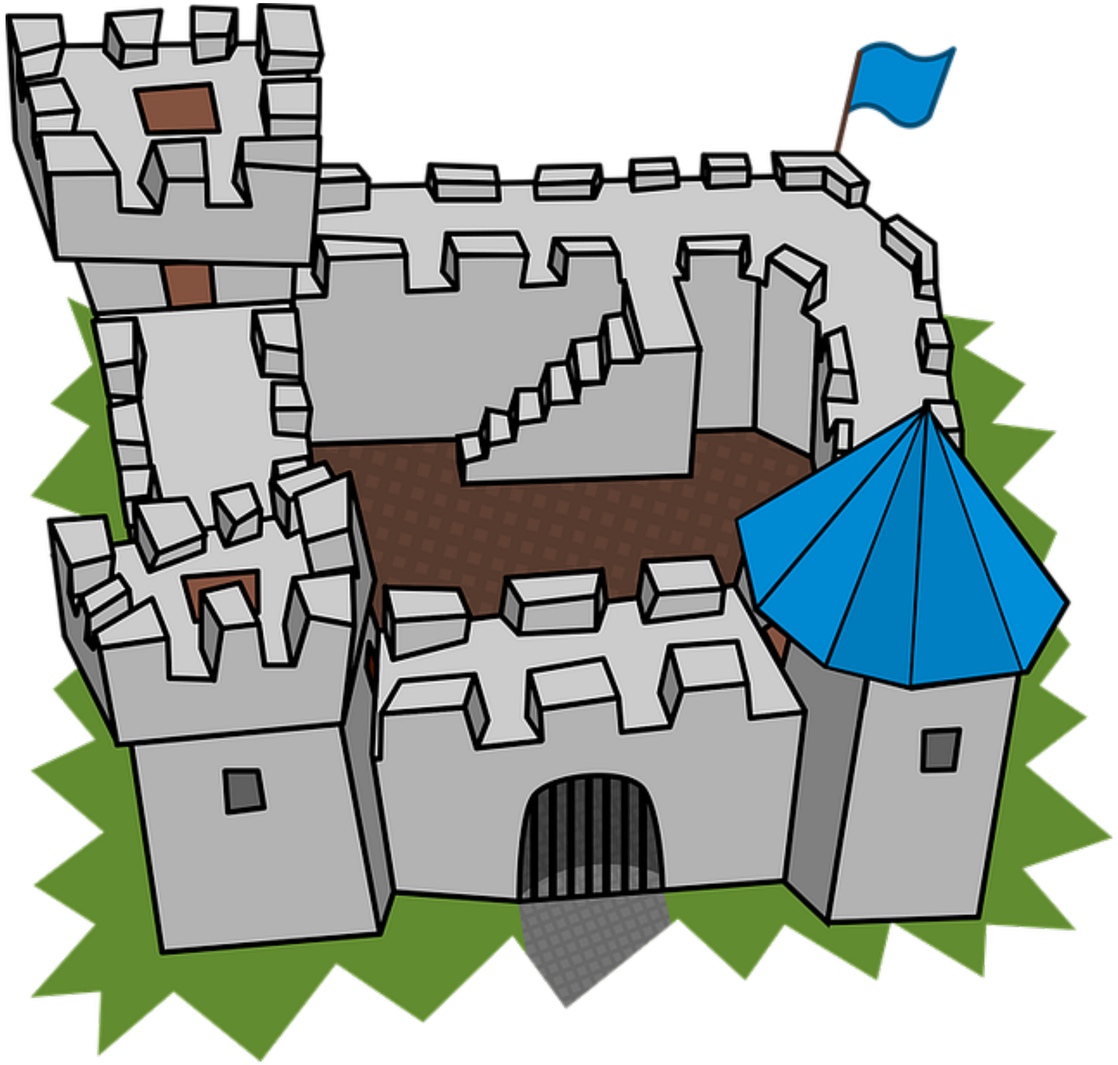
Macros sample

```
//foo.h
#ifdef MAGIC
template<int>
struct x {
    x(int i) { }
};
#else
int x = 100;
#endif
```

```
//foo.cpp
#include "foo.h"
void test(int y) {
    const int a = 100;

    auto k = x<a>(0);
}
```

C++ in Games



C++ in Games

- Language choices:
  - Unity/C# takes the biggest part of the market
  - AAA is mostly C++, Unreal Engine, Lumberyard, CryEngine and custom in-house engines
  - Rendering is mostly in C
- Console SDKs in binaries
- Performance (latency)

C++ in Games

C++ usage

- C++03 and C++11
- In-house reflection implementations
- No Boost or STL because of the allocations
- Minimal template usage
- No exceptions because of their cost

C++ in Games

Reflection

- For serialization
- For GC
- For network replication
- For various characteristics

C++ in Games

Reflection in Unreal Engine:

- Serves for interaction between C++/Blueprint
- Implemented with macros
- RPC methods

```cpp
#include "MyObject.generated.h"

UCLASS(Blueprintable)
class UMyObject : public UObject
{
    GENERATED_BODY()

public:
    MyUObject();

    UPROPERTY(BlueprintReadOnly, EditAnywhere)
    float ExampleProperty;

    UFUNCTION(BlueprintCallable)
    void ExampleFunction();
};
```

```cpp
    /** [server] remove all weapons from inventory and destroy them */
    void DestroyInventory();

    /** equip weapon */
    UFUNCTION(reliable, server, WithValidation)
    void ServerEquipWeapon(class AShooterWeapon* NewWeapon);
```

AShooterCharacter::ServerEquipWeapon_Implementation(AShooterWeapon* Weapon) -> void

AShooterCharacter::ServerEquipWeapon_Validate(AShooterWeapon* Weapon) -> bool

```cpp
    void ServerSetTargeting(bool bNewTargeting);

    /** update targeting state */
    UFUNCTION(reliable, server, WithValidation)
    void ServerSetRunning(bool bNewRunning, bool bToggle);
```

C++ in Games

Custom STL & Allocations

- No STL, custom structures, plain arrays
- Non-default memory alignment requirements
- Newly constructed or reset container allocates no memory
- Avoiding heap
- Temporal allocators with the life-time of the frame

Sample: InplaceArray<ubi32, 8>

Nicolas Fleury "C++ in Huge AAA Games” (CppCon 2014)
Scott Wardle "Memory and C++ debugging at Electronic Arts” (CppCon 2015)
EASTL – Electronic Arts Standard Template Library

"Among game developers the most fundamental weakness [of the STL] is the std allocator design, and it is this weakness that was the largest contributing factor to the creation of EASTL.”

The State of Developer Ecosystem

- Yearly: 2017, 2018, 2019
- ~15K respondents total
- 6 languages
- Enough data from all over the world
- Weighting

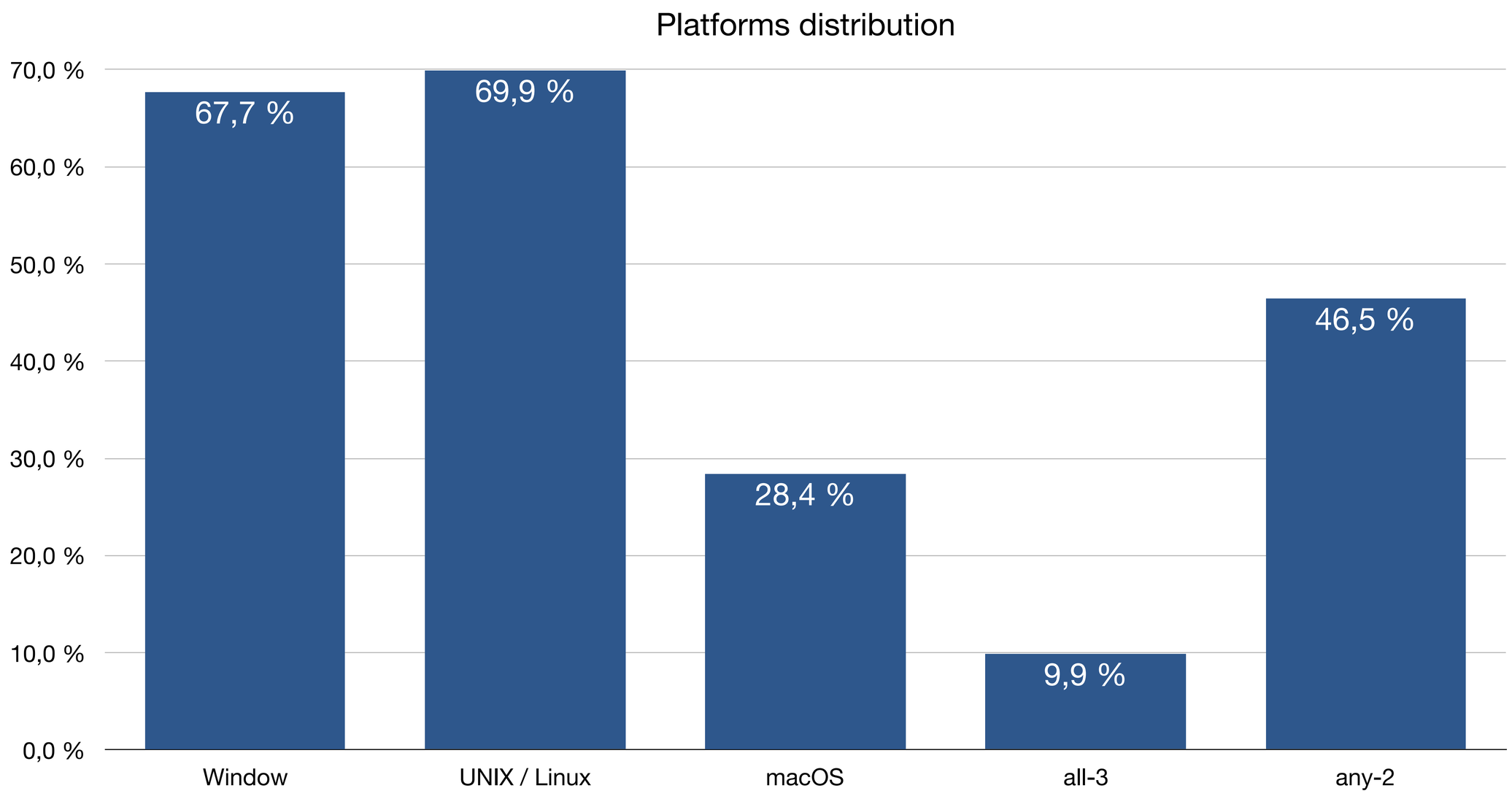
The State of Developer Ecosystem: C++

- C or C++ used in the last 12 months - **5427**
- C used in the last 12 months - **3410**
- C++ used in the last 12 months - **4148**
- Primary C++ - **1698**

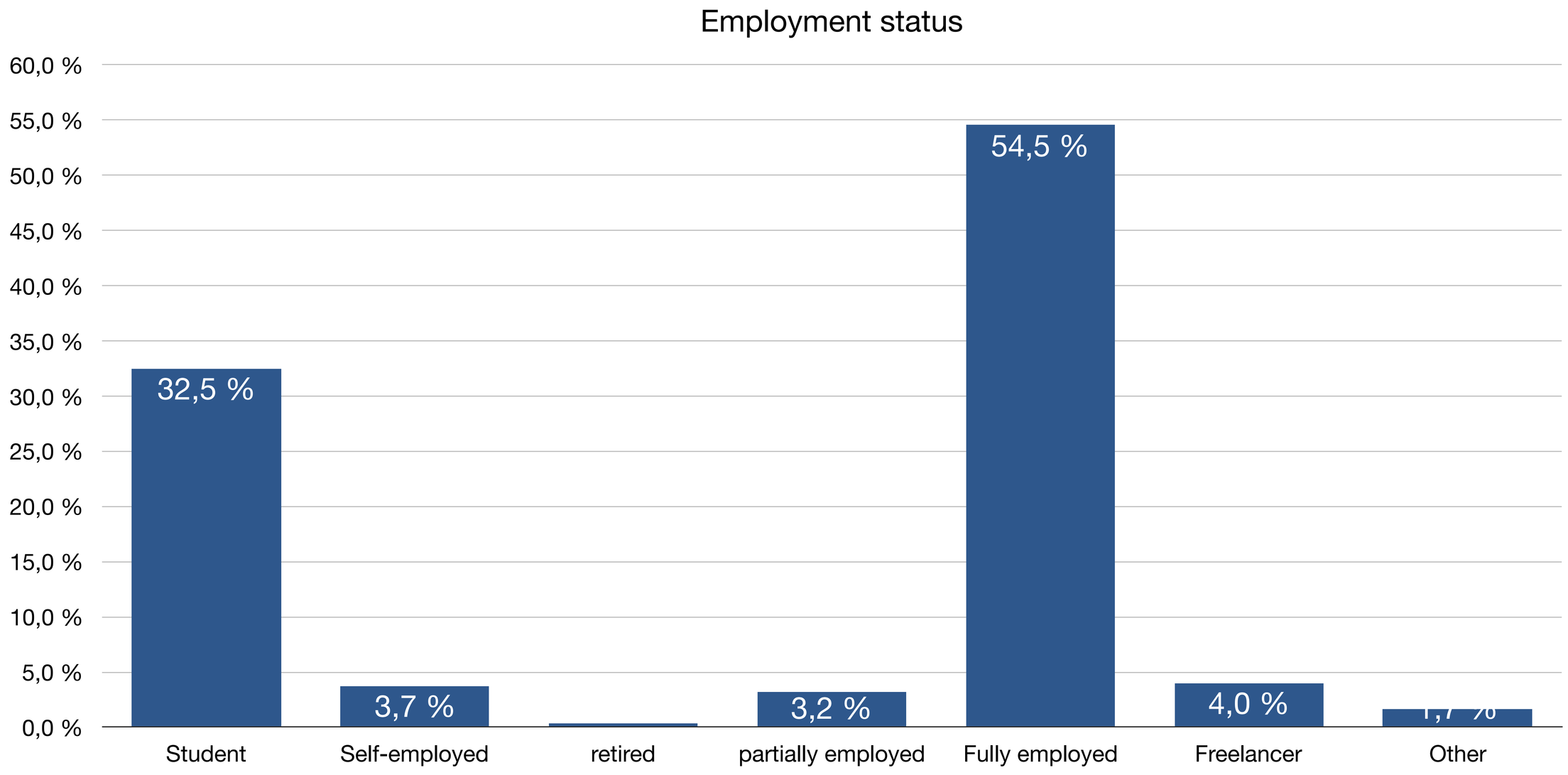
C++ Developer Survey by CPP Foundation

- 2018
- C++ used at work - 2884
- Hobby/personal - 2380
- >50% have >5 years in C++

The State of Developer Ecosystem: C++

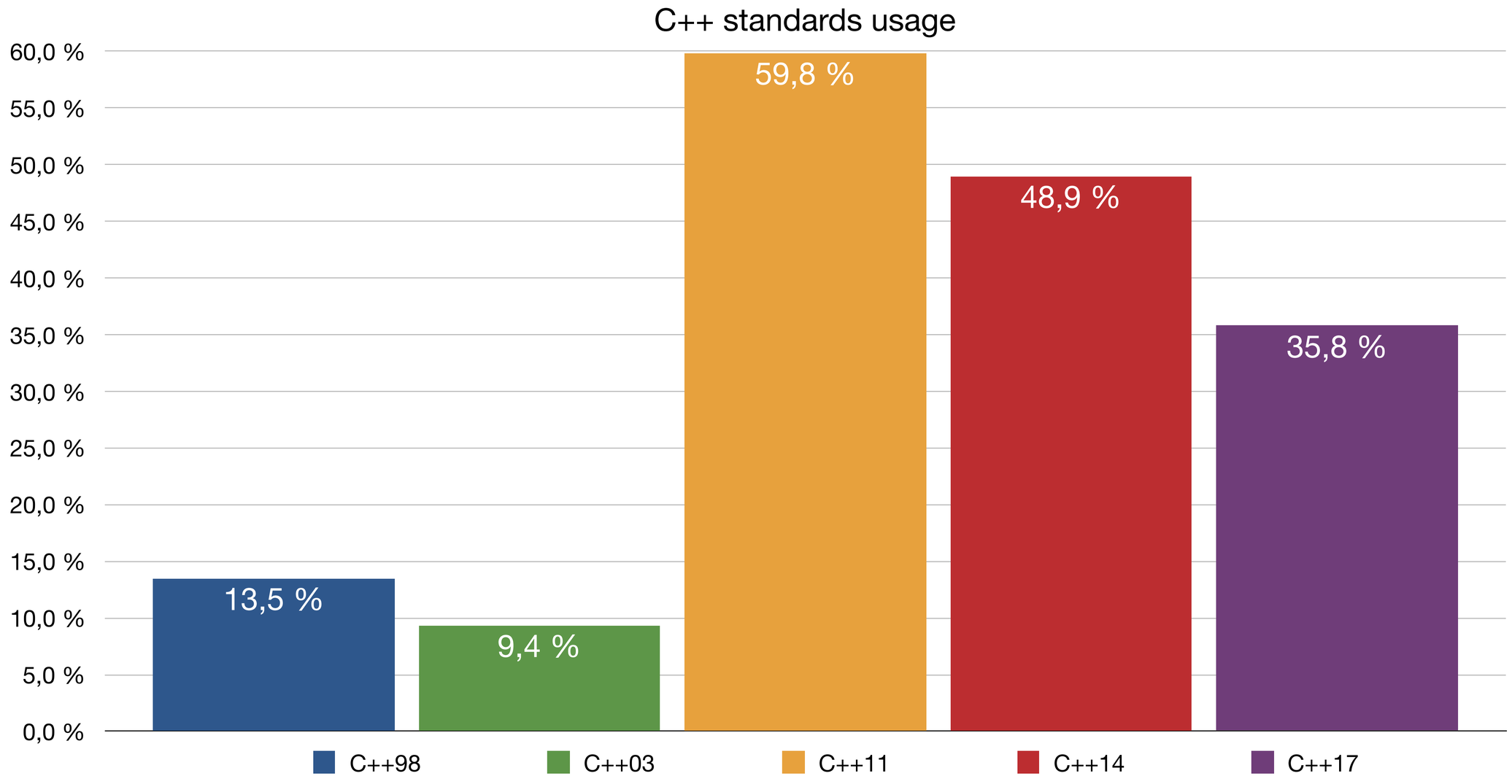


The State of Developer Ecosystem: C++

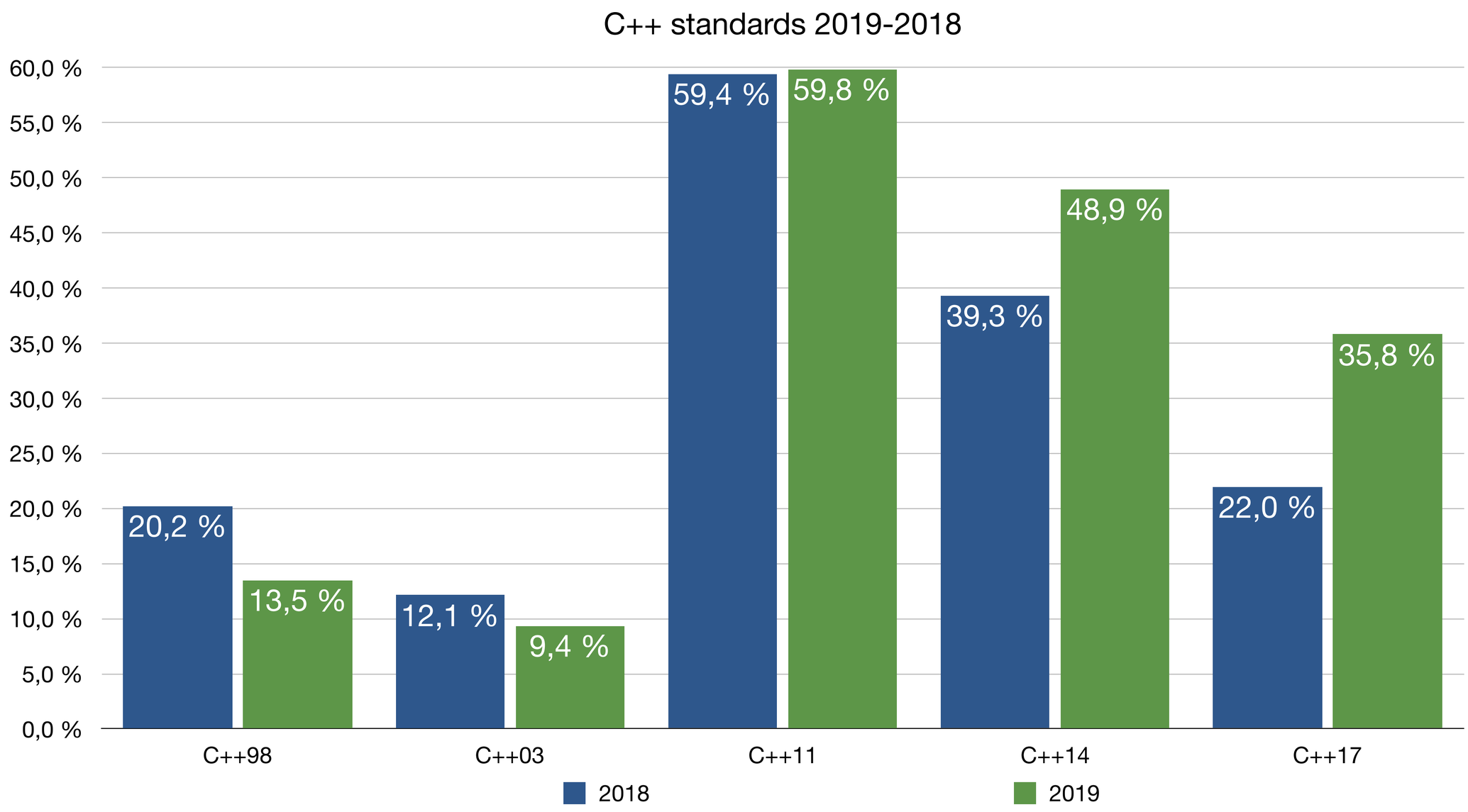


C++ standards

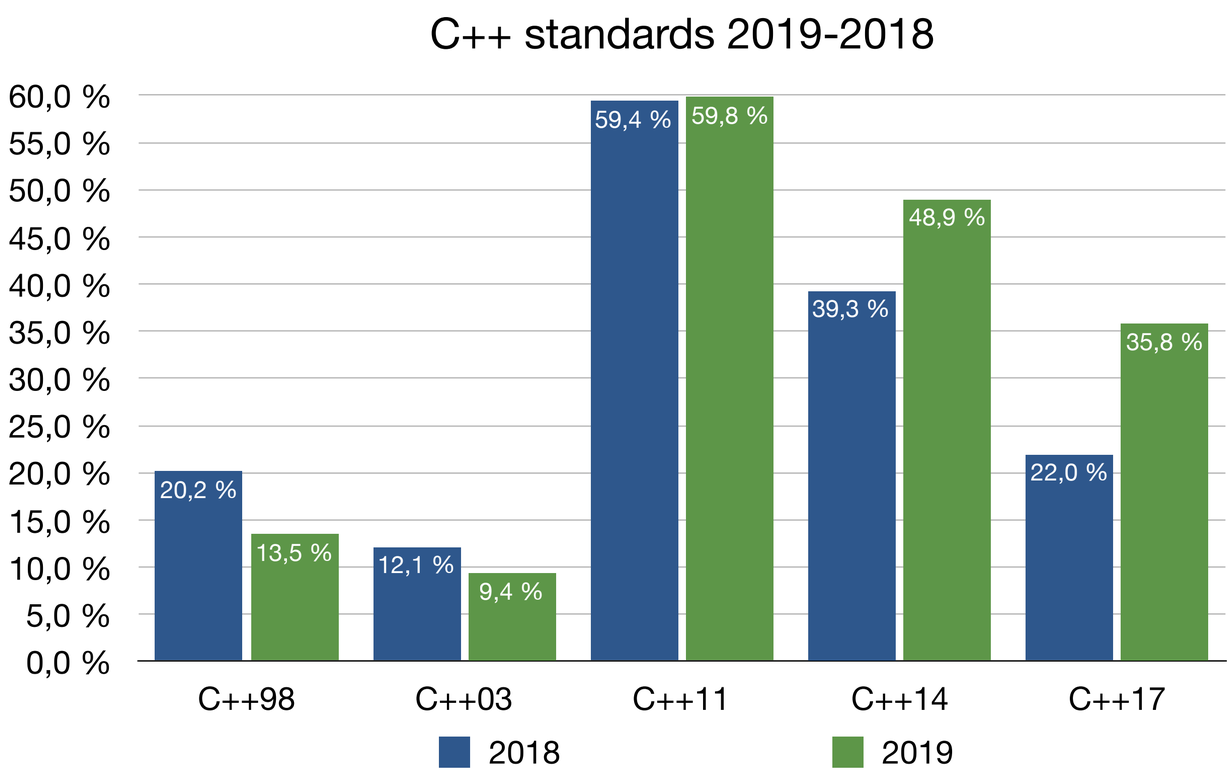
C++ standards

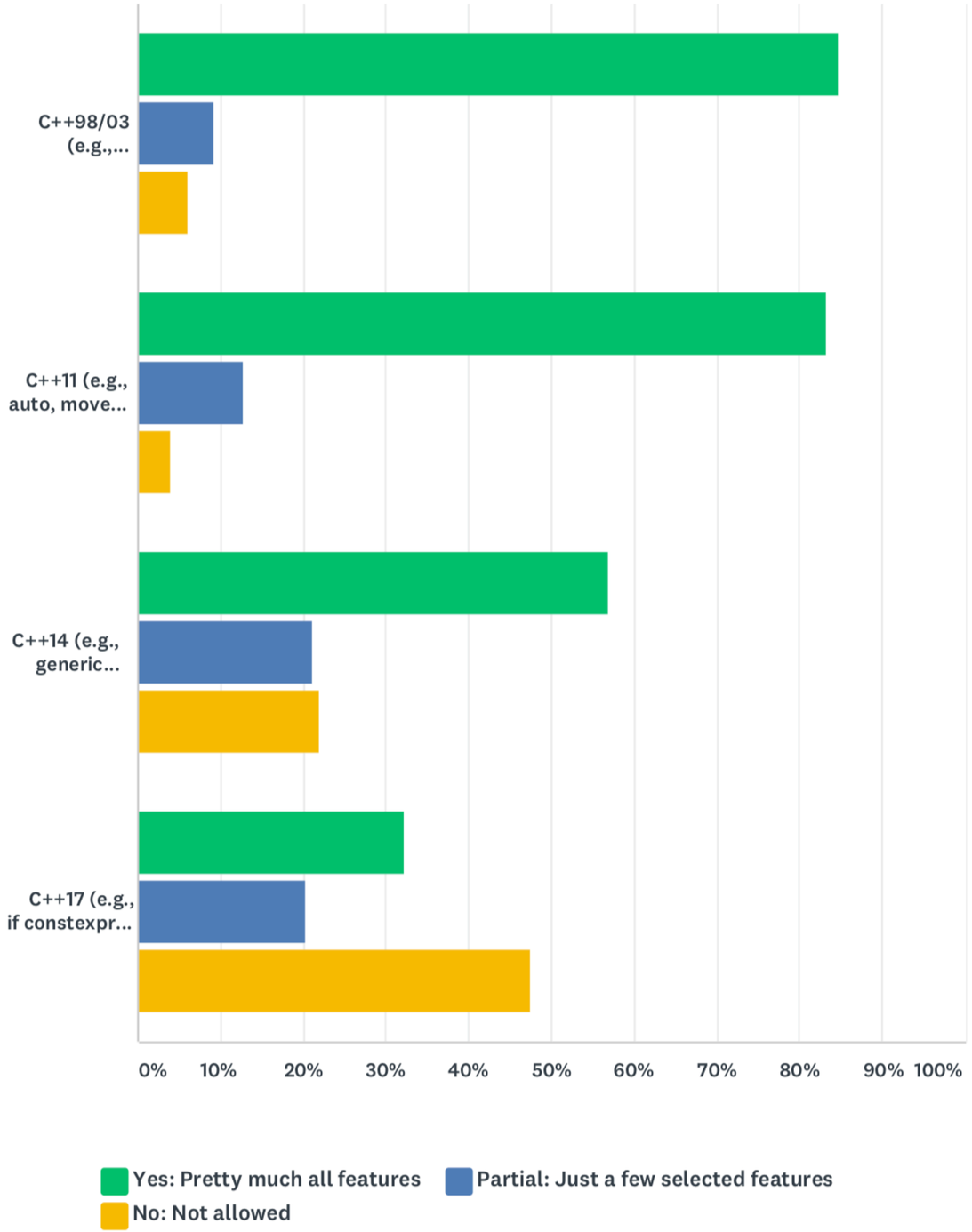

C++ standards usage
60,0 %
55,0 %
50,0 %
45,0 %
40,0 %
35,0 %
30,0 %
25,0 %
20,0 %
15,0 %
10,0 %
5,0 %
0,0 %
13,5 %
9,4 %
59,8 %
48,9 %
35,8 %
C++98
C++03
C++11
C++14
C++17

C++ standards

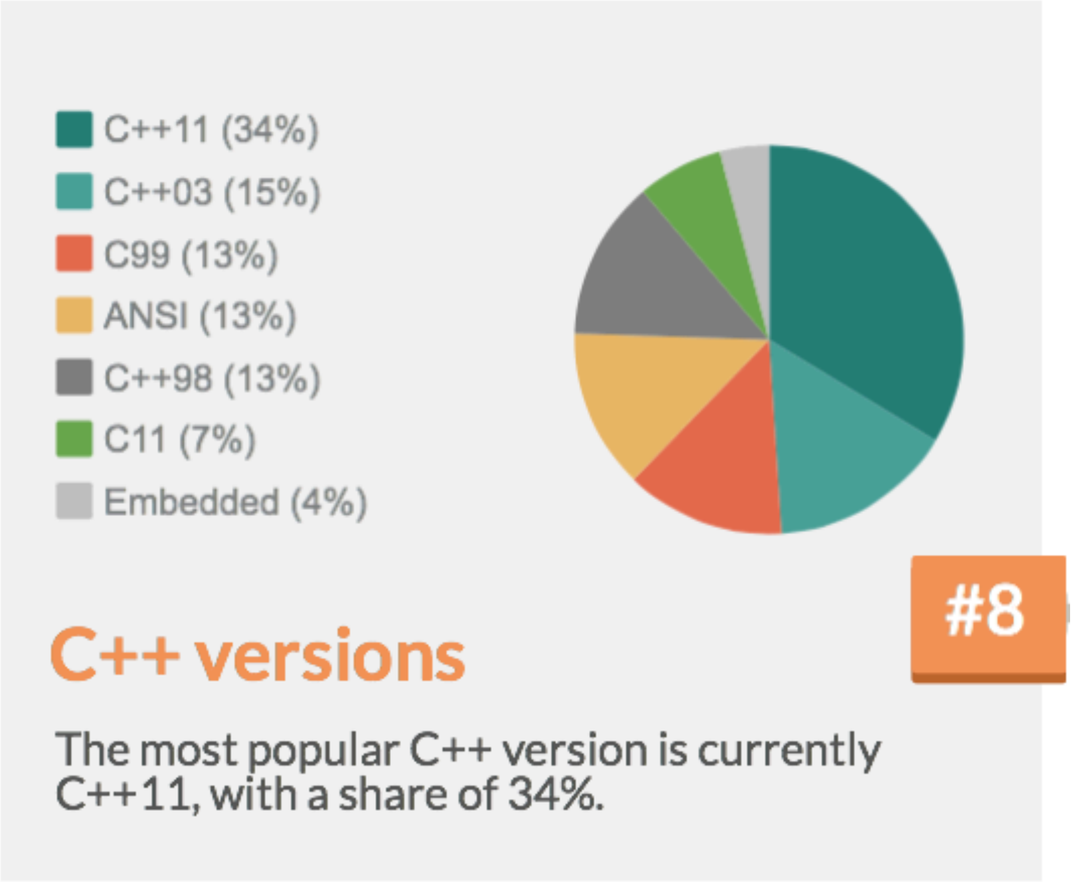

C++ standards 2019-2018
60,0 %
55,0 %
50,0 %
45,0 %
40,0 %
35,0 %
30,0 %
25,0 %
20,0 %
15,0 %
10,0 %
5,0 %
0,0 %
20,2 %
13,5 %
12,1 %
9,4 %
59,4 %
59,8 %
39,3 %
48,9 %
22,0 %
35,8 %
C++98
C++03
C++11
C++14
C++17
2018
2019

C++ standards


C++ standards 2019-2018
60,0 %
55,0 %
50,0 %
45,0 %
40,0 %
35,0 %
30,0 %
25,0 %
20,0 %
15,0 %
10,0 %
5,0 %
0,0 %
20,2 %
13,5 %
12,1 %
9,4 %
59,4 %
59,8 %
39,3 %
48,9 %
22,0 %
35,8 %
C++98
C++03
C++11
C++14
C++17
2018
2019


C++98/03 (e.g.,...
C++11 (e.g., auto, move...
C++14 (e.g., generic...
C++17 (e.g., if constexpr...
0%
10%
20%
30%
40%
50%
60%
70%
80%
90%
100%
Yes: Pretty much all features
Partial: Just a few selected features
No: Not allowed

C++ standards

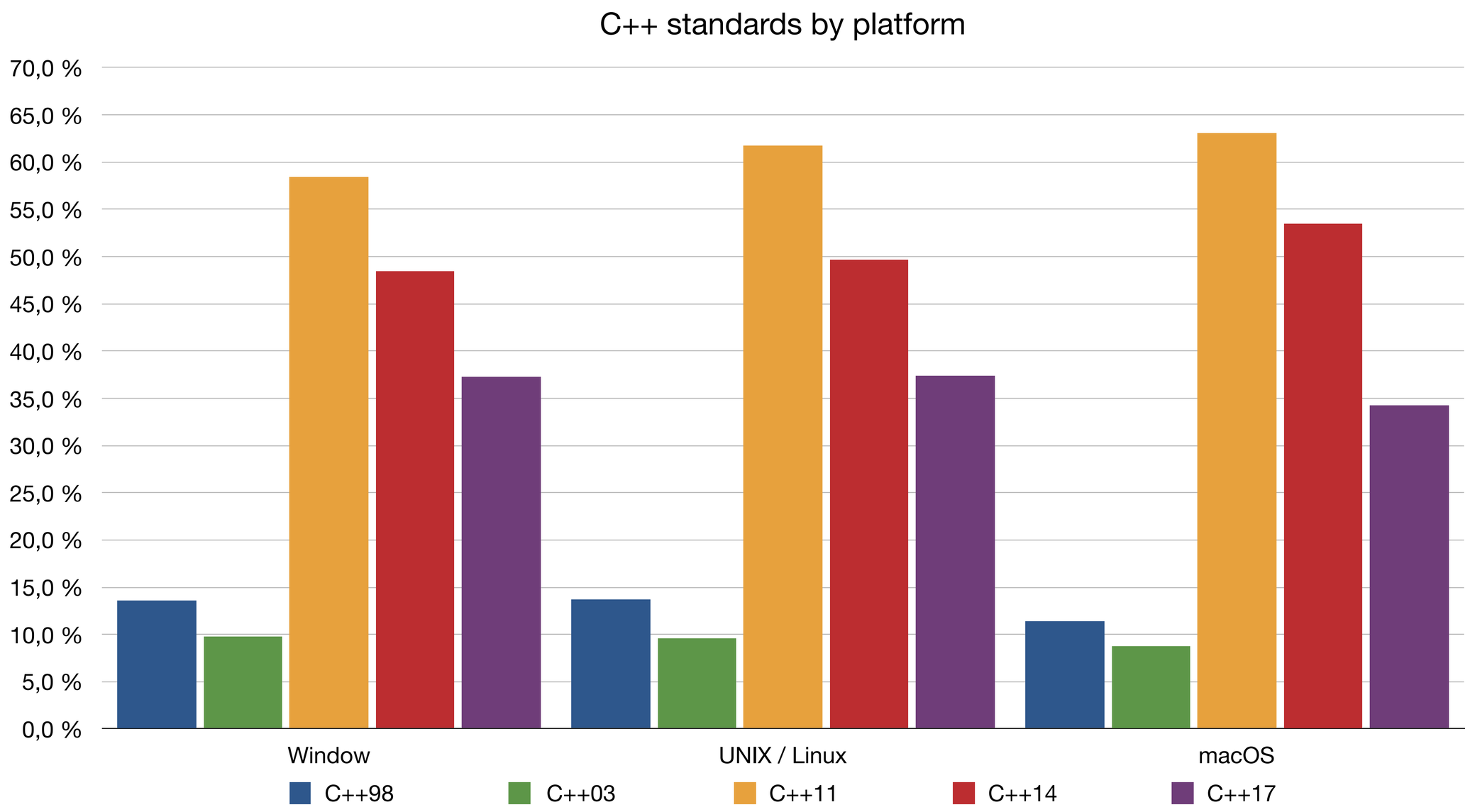

C++11 (34%)
C++03 (15%)
C99 (13%)
ANSI (13%)
C++98 (13%)
C11 (7%)
Embedded (4%)
#8
C++ versions
The most popular C++ version is currently C++11, with a share of 34%.

The State of Developer Ecosystem: C++

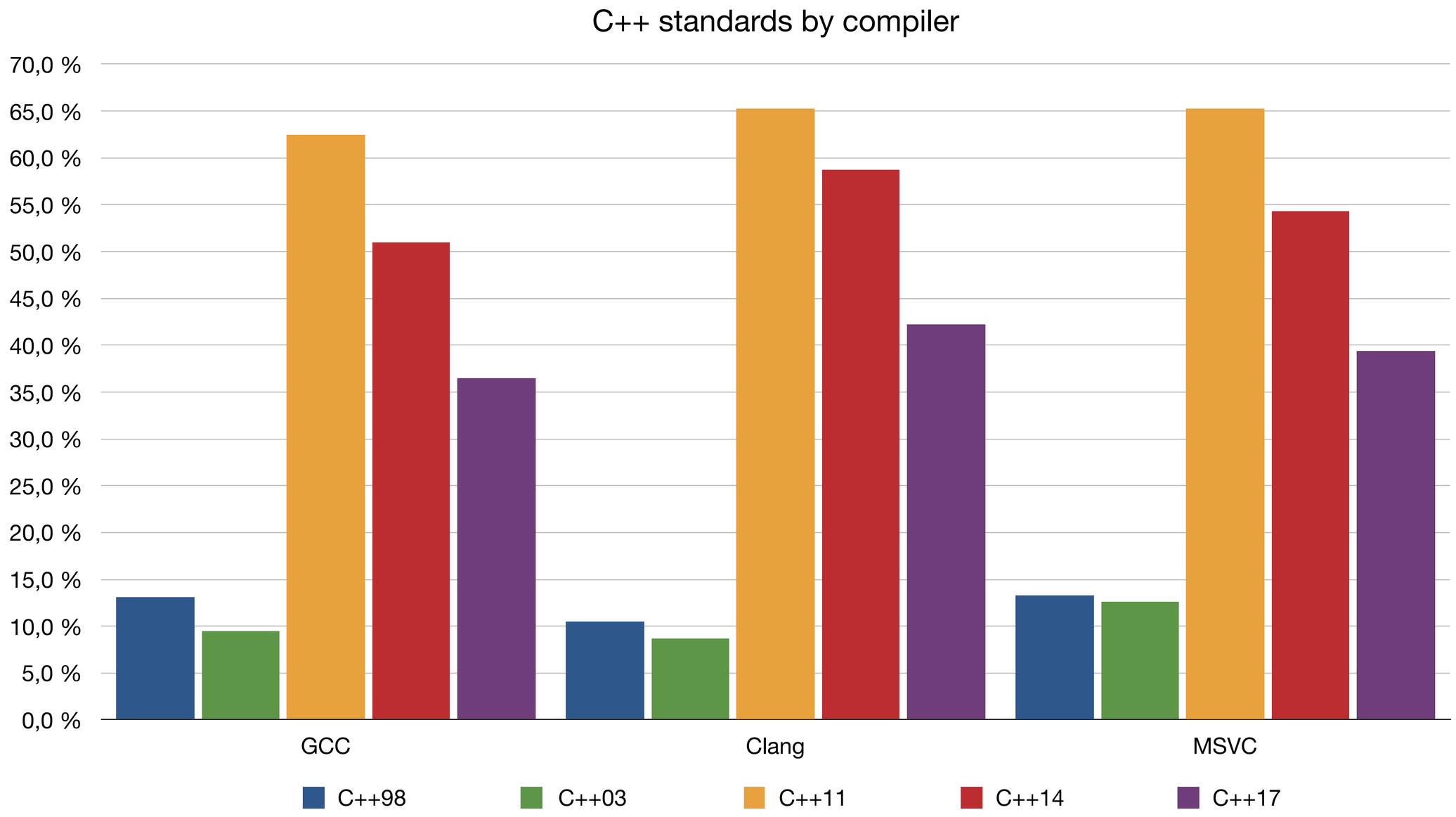
—

- Per platforms distribution
- Per compiler distribution
- Per area of development
- Per employment group

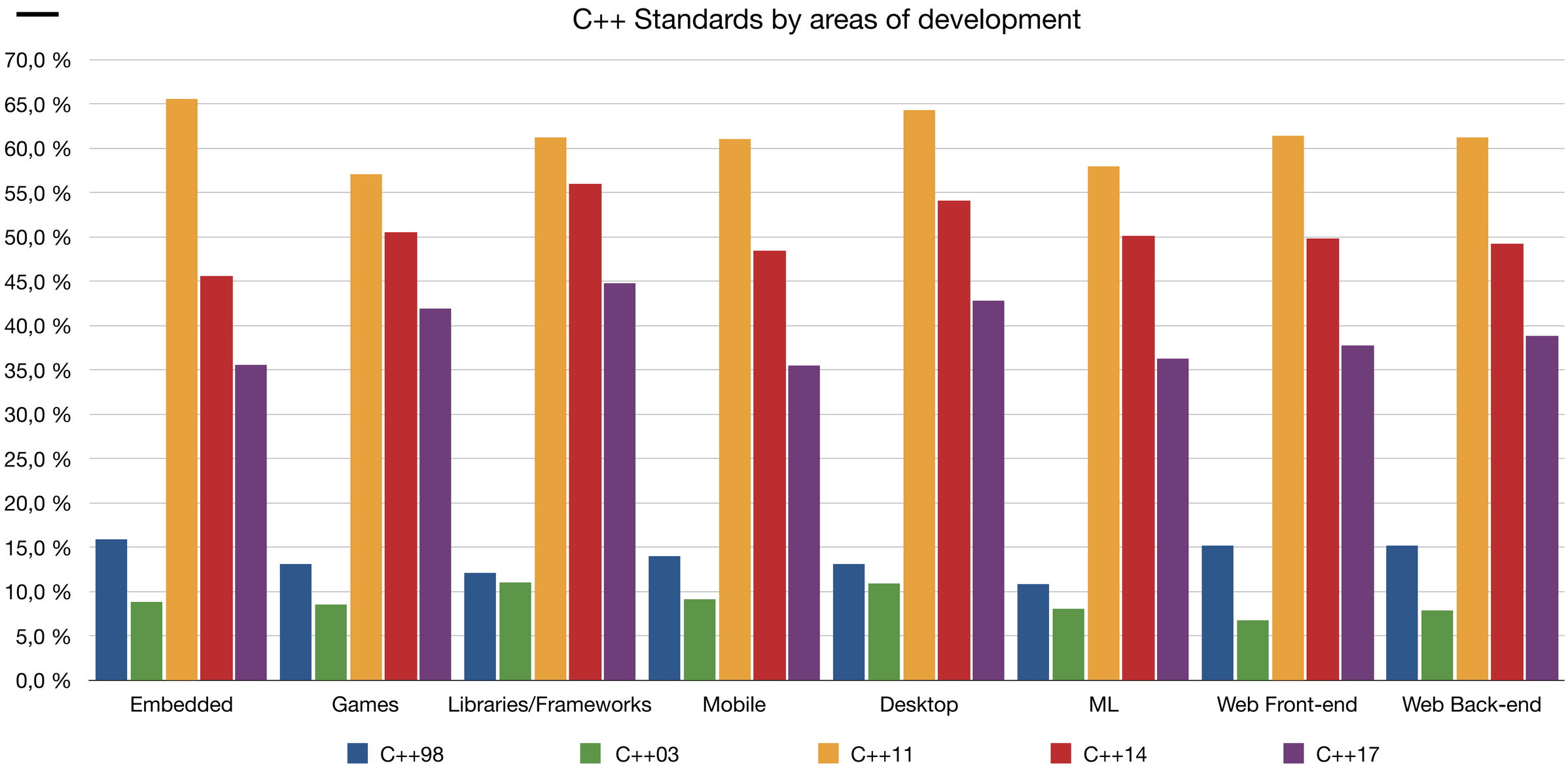
C++ standards


C++ standards by platform
70,0 %
65,0 %
60,0 %
55,0 %
50,0 %
45,0 %
40,0 %
35,0 %
30,0 %
25,0 %
20,0 %
15,0 %
10,0 %
5,0 %
0,0 %
Window
UNIX / Linux
macOS
C++98
C++03
C++11
C++14
C++17

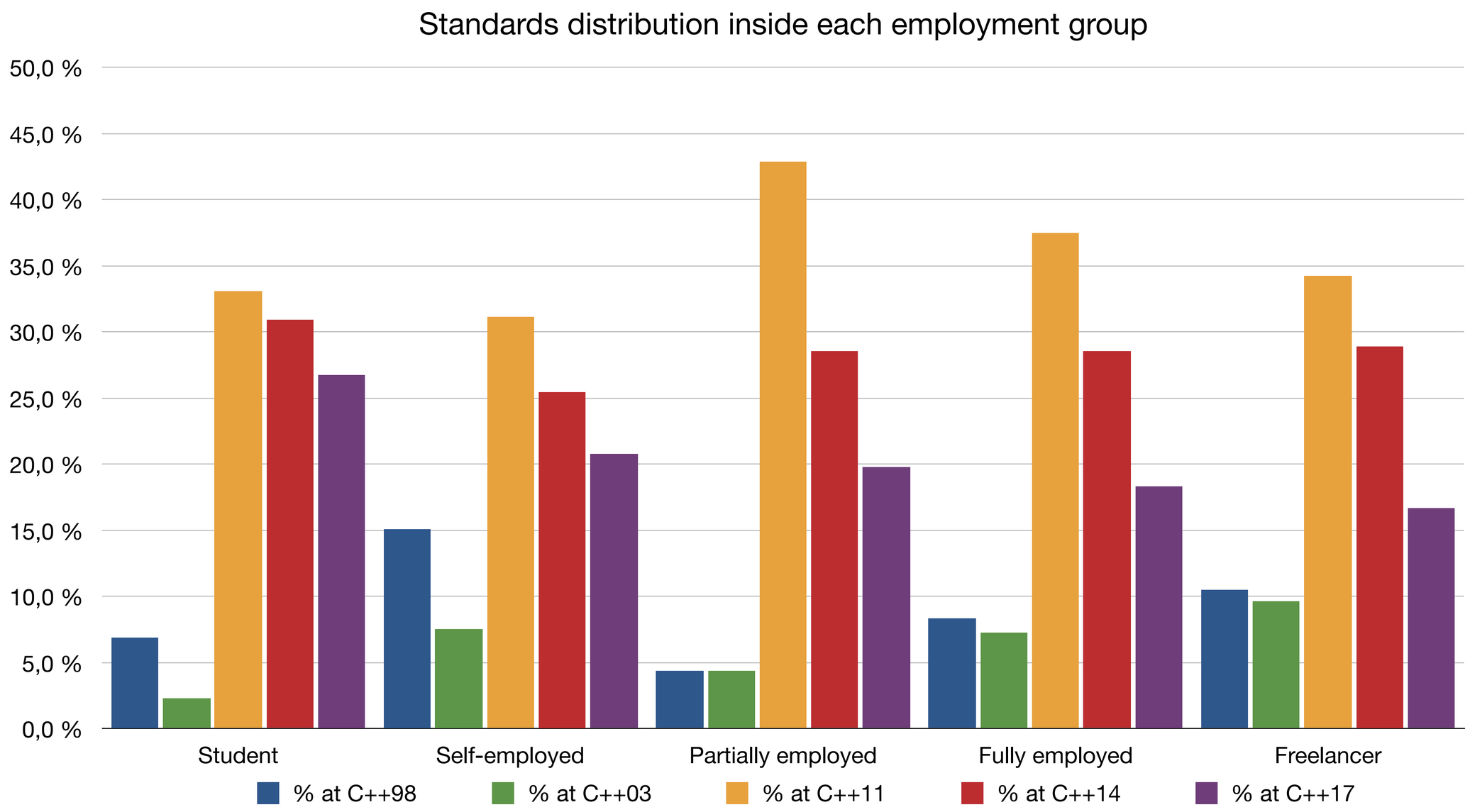
C++ standards


C++ standards by compiler
70,0 %
65,0 %
60,0 %
55,0 %
50,0 %
45,0 %
40,0 %
35,0 %
30,0 %
25,0 %
20,0 %
15,0 %
10,0 %
5,0 %
0,0 %
GCC
Clang
MSVC
C++98
C++03
C++11
C++14
C++17

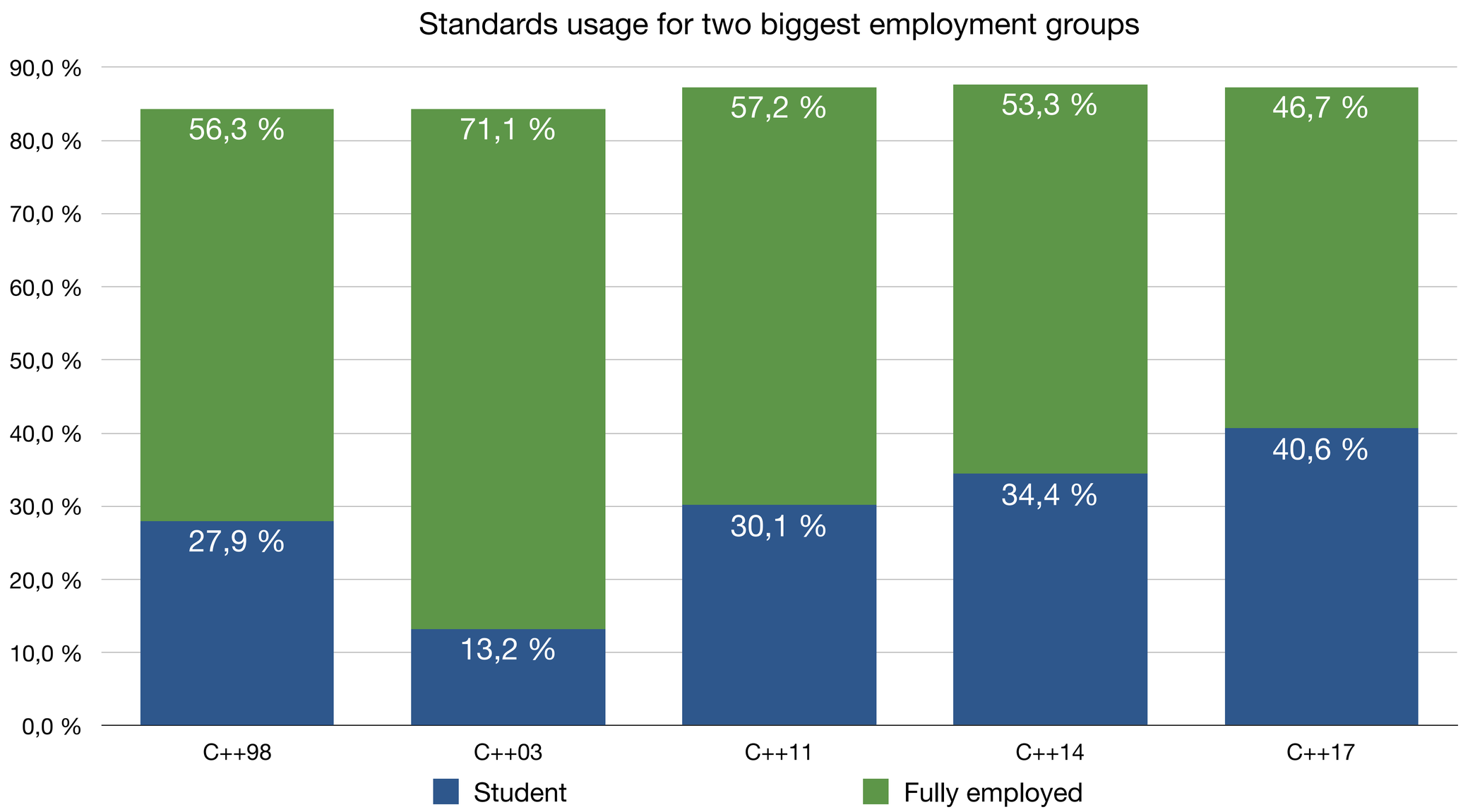
C++ standards


C++ Standards by areas of development
70,0 %
65,0 %
60,0 %
55,0 %
50,0 %
45,0 %
40,0 %
35,0 %
30,0 %
25,0 %
20,0 %
15,0 %
10,0 %
5,0 %
0,0 %
Embedded
Games
Libraries/Frameworks
Mobile
Desktop
ML
Web Front-end
Web Back-end
C++98
C++03
C++11
C++14
C++17

C++ standards

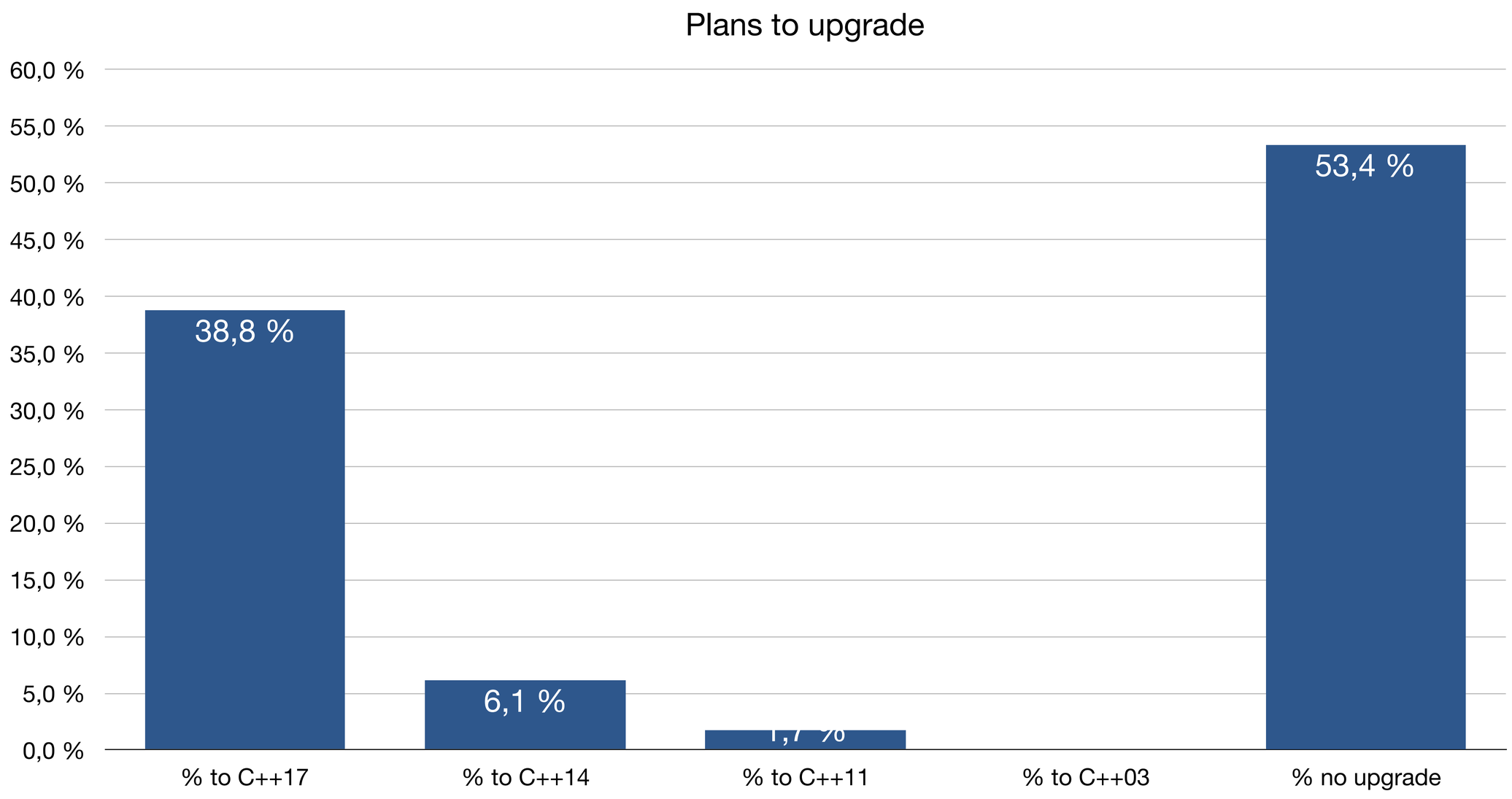

Standards distribution inside each employment group
50,0 %
45,0 %
40,0 %
35,0 %
30,0 %
25,0 %
20,0 %
15,0 %
10,0 %
5,0 %
0,0 %
Student
Self-employed
Partially employed
Fully employed
Freelancer
% at C++98
% at C++03
% at C++11
% at C++14
% at C++17

C++ standards

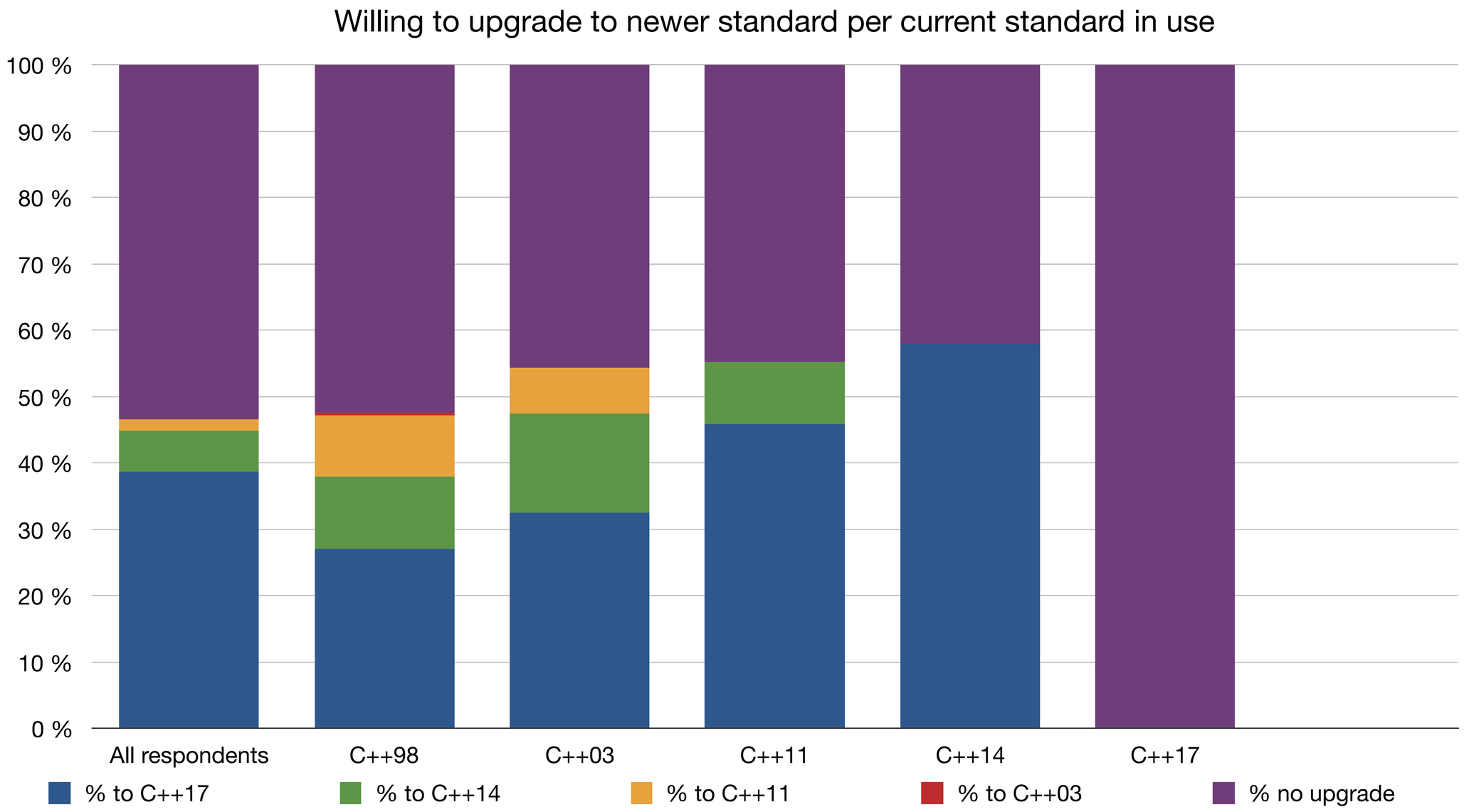

Standards usage for two biggest employment groups
90,0 %
80,0 %
70,0 %
60,0 %
50,0 %
40,0 %
30,0 %
20,0 %
10,0 %
0,0 %
56,3 %
27,9 %
71,1 %
13,2 %
57,2 %
30,1 %
53,3 %
34,4 %
46,7 %
40,6 %
C++98
C++03
C++11
C++14
C++17
Student
Fully employed

Upgrading

C++ standards: upgrade

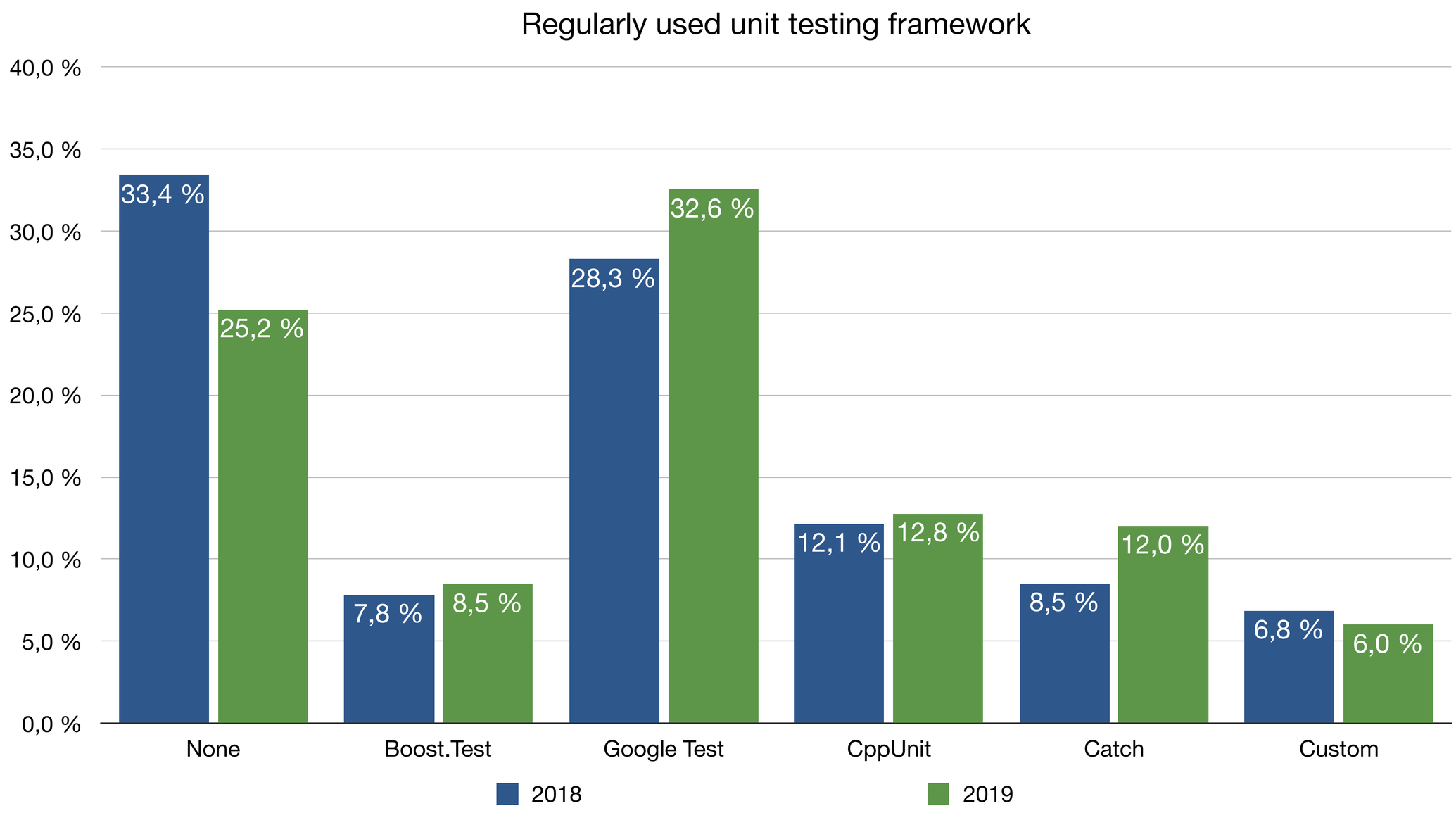


C++ standards: upgrade



Unit testing

Unit testing



Unit testing

- ~70 in the list: https://en.wikipedia.org/wiki/List_of_unit_testing_frameworks#C++
- Reddit discussions:
  - Most Popular C++ Unit Testing Frameworks
    https://www.reddit.com/r/cpp/comments/4e9afx/most_popular_c_unit_testing_frameworks/
  - Best way to do unit testing in c++?
    https://www.reddit.com/r/cpp/comments/36pru0/best_way_to_do_unit_testing_in_c/
  - Is there a de-facto standard "framework" for unit testing in C++?
    https://www.reddit.com/r/cpp/comments/1zh0p1/is_there_a_defacto_standard_framework_for_unit/
- Recommendations: Google Test (with Google Mock), Catch

Unit testing

Criteria	Framework
Feature rich	Google Test, Boost.Test
Easy-to-start	Catch
Integrations	Google Test

Unit testing

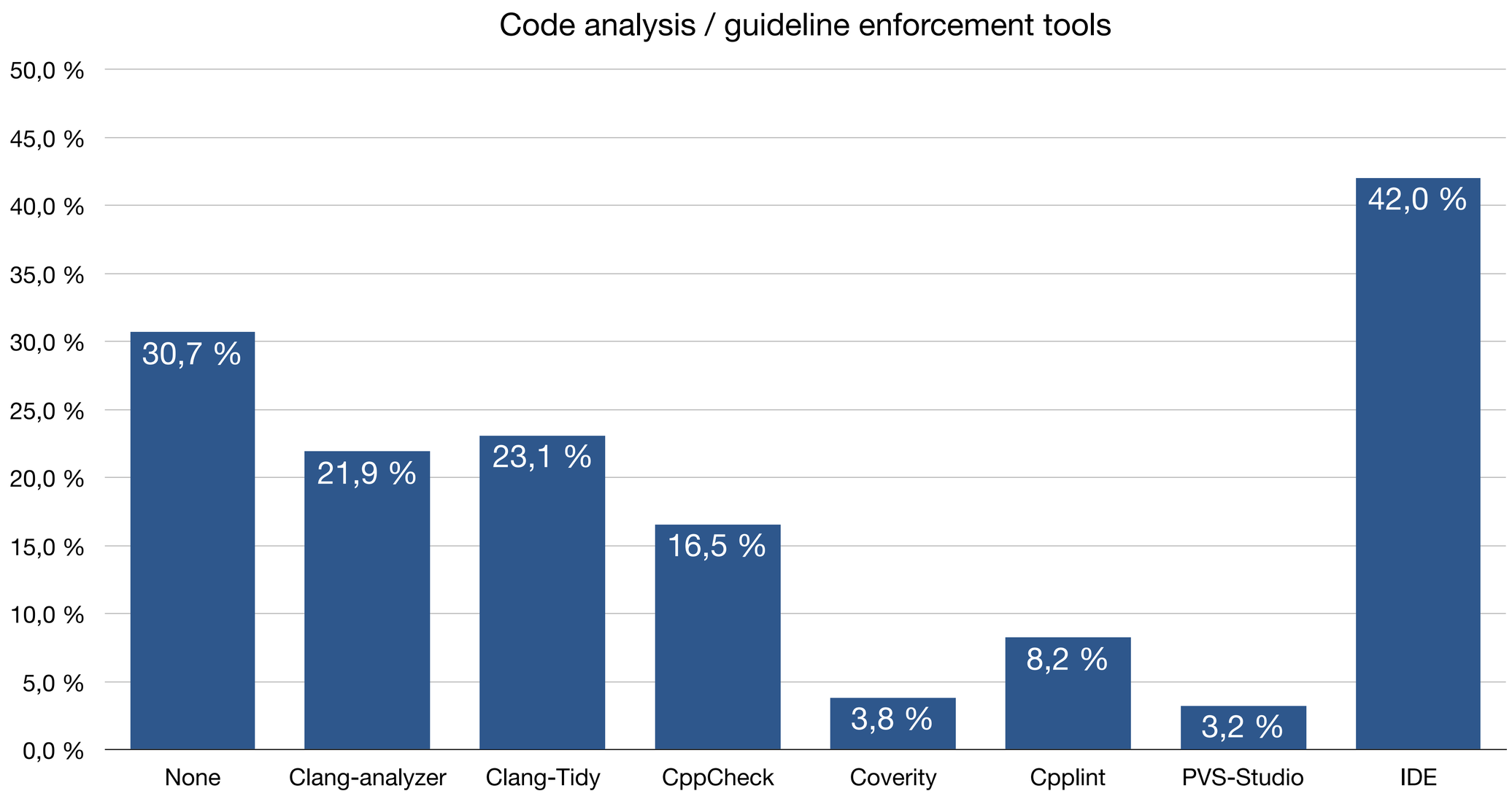
Embedded market:

- tests running on hardware
- tests are required for certifications according to the standards
- no home-made products because of the certification
- no integration into IDEs (Eclipse)
- pricy

	External channels N: 227			Internal channels N: 276		
values	**shares**	**lower CI**	**upper CI**	**shares**	**lower CI**	**upper CI**
No, I don't use any	89%	84%	92%	89%	84%	92%
Other - Write In	7%	4%	11%	8%	5%	12%
VectorCAST	1%	0%	4%	1%	1%	4%
TestPlant	1%	0%	3%	0%	0%	3%
Parasoft DTP	1%	0%	3%	-	-	-
RogueWave KlockWork	1%	0%	3%	2%	1%	4%
QA Systems CANTATA	1%	0%	4%	0%	0%	3%
Elvior TTCN-3	0%	0%	3%	-	-	-
hitex TESSY	0%	0%	3%	0%	0%	3%

Code analysis / guidelines enforcement

Code analysis



References

- C++ Foundation Developer Survey
  - [2018-2] https://isocpp.org/files/papers/CppDevSurvey-2018-02-summary.pdf
- The State of Developer Ecosystem Survey
  - [2017] https://www.jetbrains.com/research/devecosystem-2017/cpp/
  - [2018] https://www.jetbrains.com/research/devecosystem-2018/cpp/
  - [2019] https://www.jetbrains.com/research/devecosystem-2019 – results are not yet available!
- C/C++ Infographics
  - [collected 2013] https://blog.jetbrains.com/clion/2015/07/infographics-cpp-facts-before-clion/
- Nicolas Fleury "C++ in Huge AAA Games"
  - [CppCon 2014] https://www.youtube.com/watch?v=qYN6eduU06s
- Scott Wardle “Memory and C++ debugging at Electronic Arts”
  - [CppCon 2015] https://www.youtube.com/watch?v=8KIvWJUYbDA
- EASTL - Electronic Arts Standard Template Library
  - [2007] http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2007/n2271.html
  - [GitHub] https://github.com/electronicarts/EASTL
- Carl Cook “When a Microsecond Is an Eternity: High Performance Trading Systems in C++”
  - [CppCon 2017] https://www.youtube.com/watch?v=NH1Tta7purM

Thank you
for your attention
—

Приходите на стенд сообщества C++ задавать вопросы!
(а еще решать quiz и выигрывать призы!)