bureau 14
SOFTWARE R&D

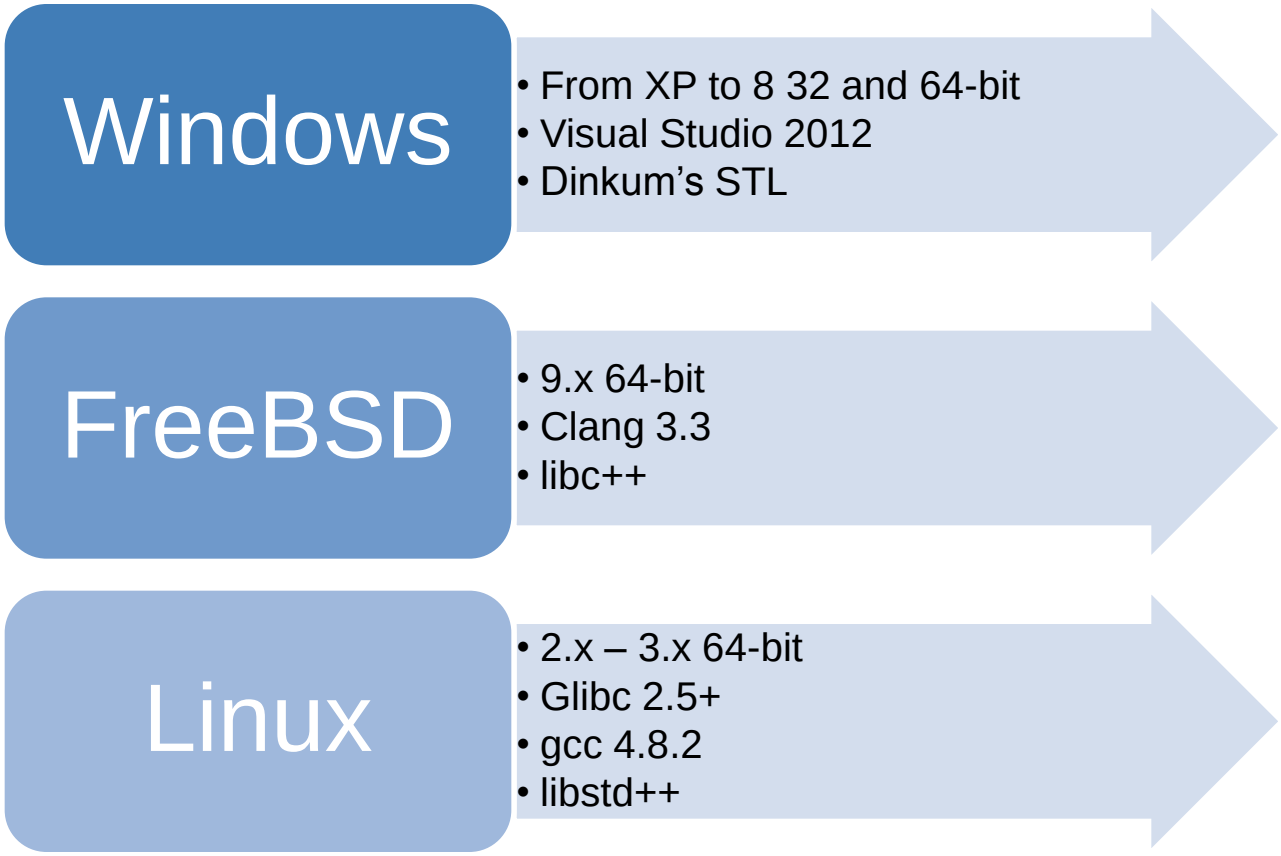
Multiplatform C++

Edouard Alligand
Chairman and CTO

Who we are

www.quasardb.net

It’s really multiplatform



But…

…isn't C++ multiplatform?

Does it blend?

```
#include <cstdlib>
#include <iostream>

int main(int argc, char ** argv)
{
	std::cout << “giggidy” << std::endl;

	return EXIT_SUCCESS;
}
```

Does it blend?

```
#include <cstdlib>
#include <iostream>

int main(int argc, char ** argv)
{
	std::cerr.sync_with_stdio(false);
	std::cout << “giggidy” << std::endl;
	return EXIT_SUCCESS;
}
```

Does it blend?

```
#include <cstdlib>
#include <iostream>

int main(int argc, char ** argv)
{
	const char * const char blah [] = “giggidy\n”;
	fwrite(blah, sizeof(blah), stdout);
	return EXIT_SUCCESS;
}
```

Does it blend?

```
int main(int argc, char ** argv)
{
#if BOOST_OS_WINDOWS
#ifndef _DEBUG
	_set_abort_behavior(0, _WRITE_ABORT_MSG);
	_set_abort_behavior(1, _CALL_REPORTFAULT);
	_set_error_mode(_OUT_TO_STDERR);
#endif
#endif
	const char * const char blah [] = “giggidy\n”;
	fwrite(blah, sizeof(blah), stdout);
	return EXIT_SUCCESS;
}
```

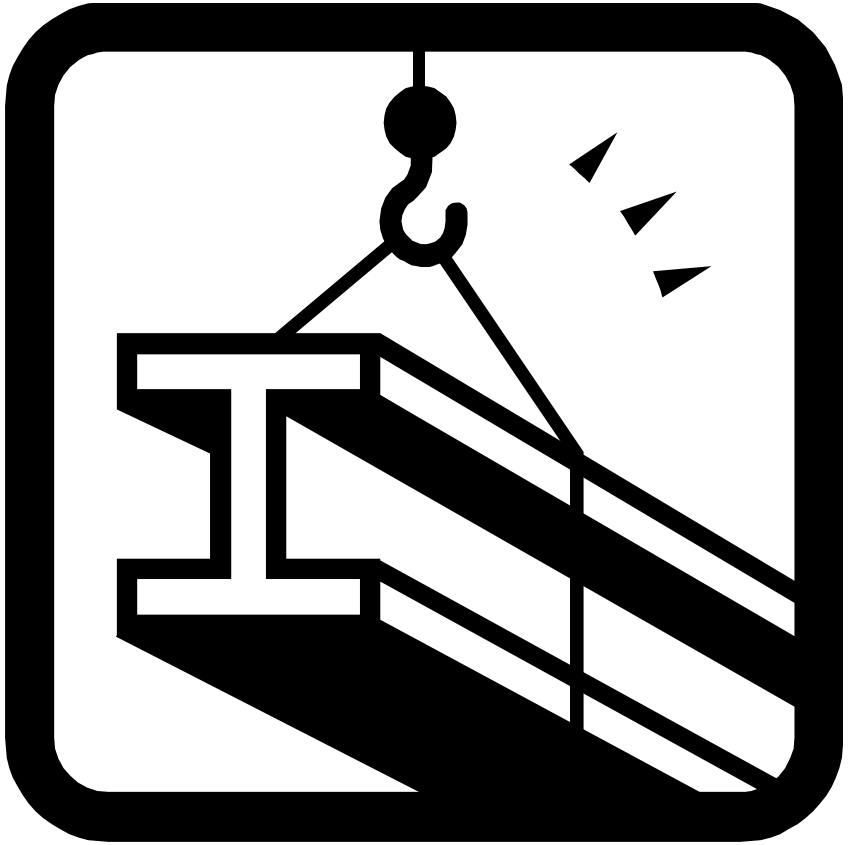
Does it blend?

```
int main(int argc, char ** argv)
{
#if BOOST_OS_WINDOWS
#ifndef _DEBUG
	_set_abort_behavior(0, _WRITE_ABORT_MSG);
	_set_abort_behavior(1, _CALL_REPORTFAULT);
	_set_error_mode(_OUT_TO_STDERR);
#endif
#endif
	std::setlocale(LC_ALL, "en_US.UTF-8");
	const char * const char blah [] = “Привет!\n”;
	fwrite(blah, sizeof(blah), stdout);
	return EXIT_SUCCESS;
}
```

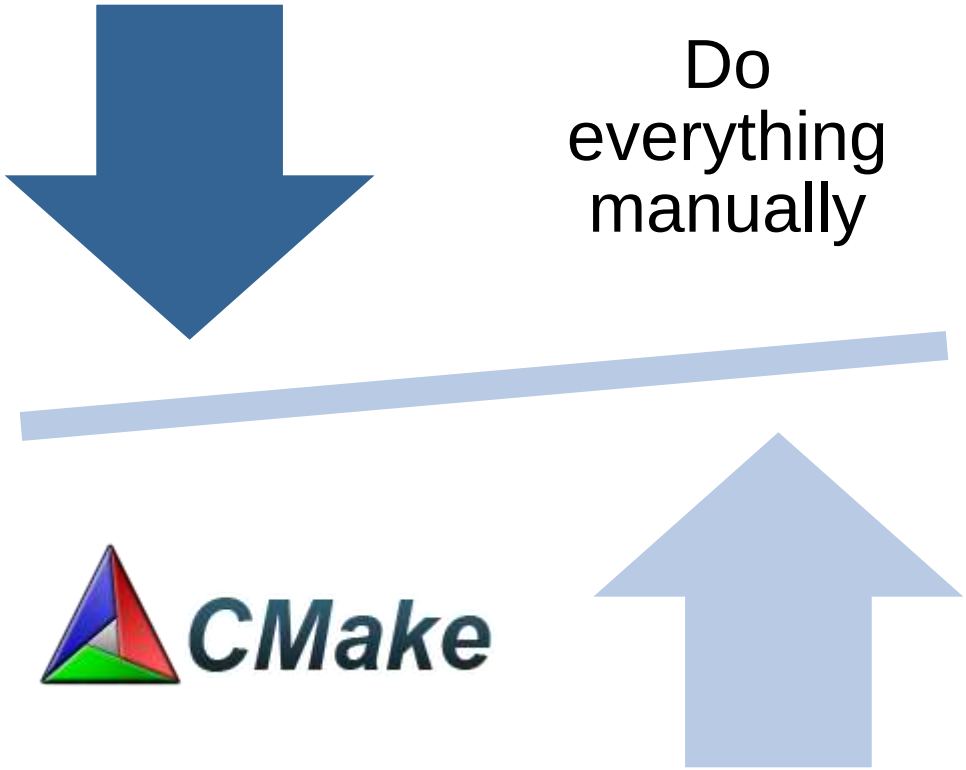
Conclusion

Topics not covered

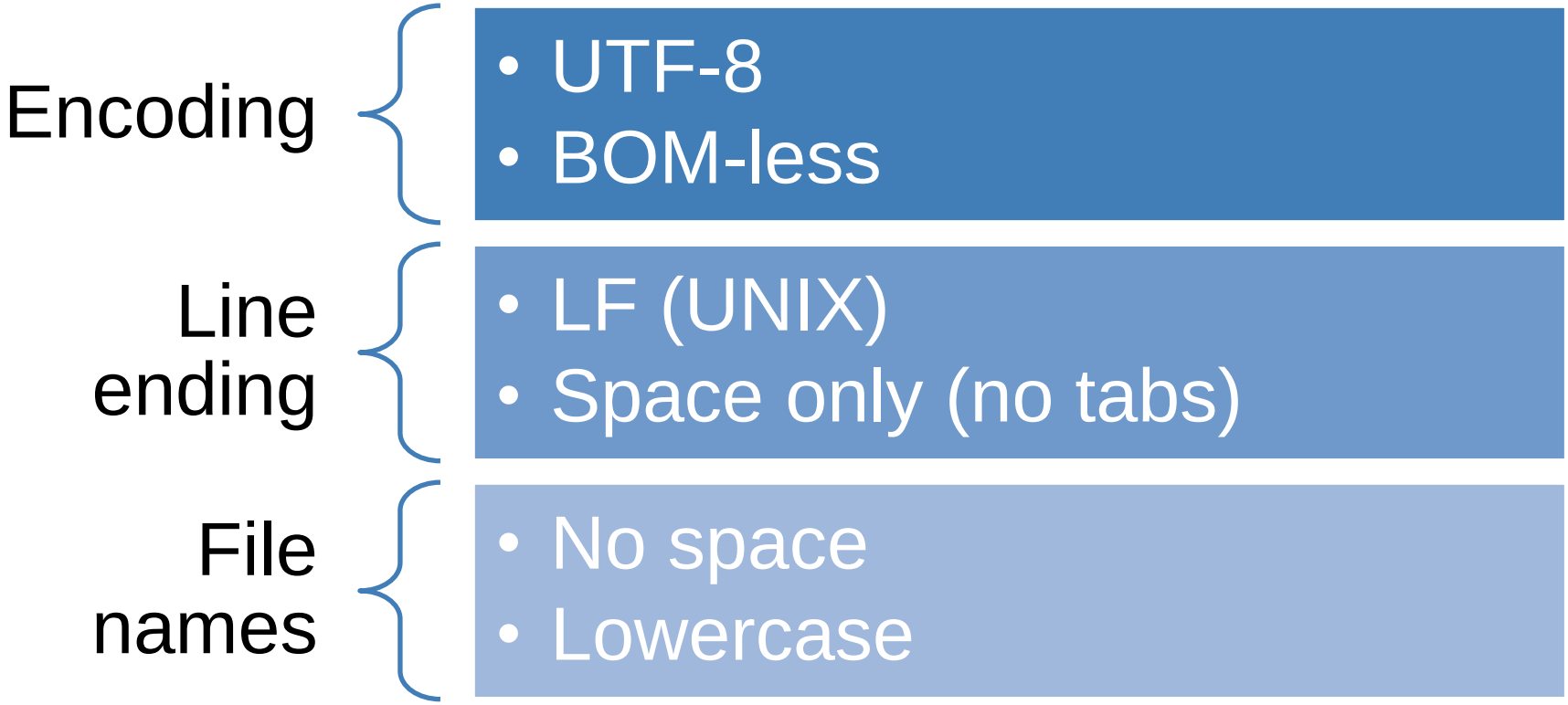
First things first



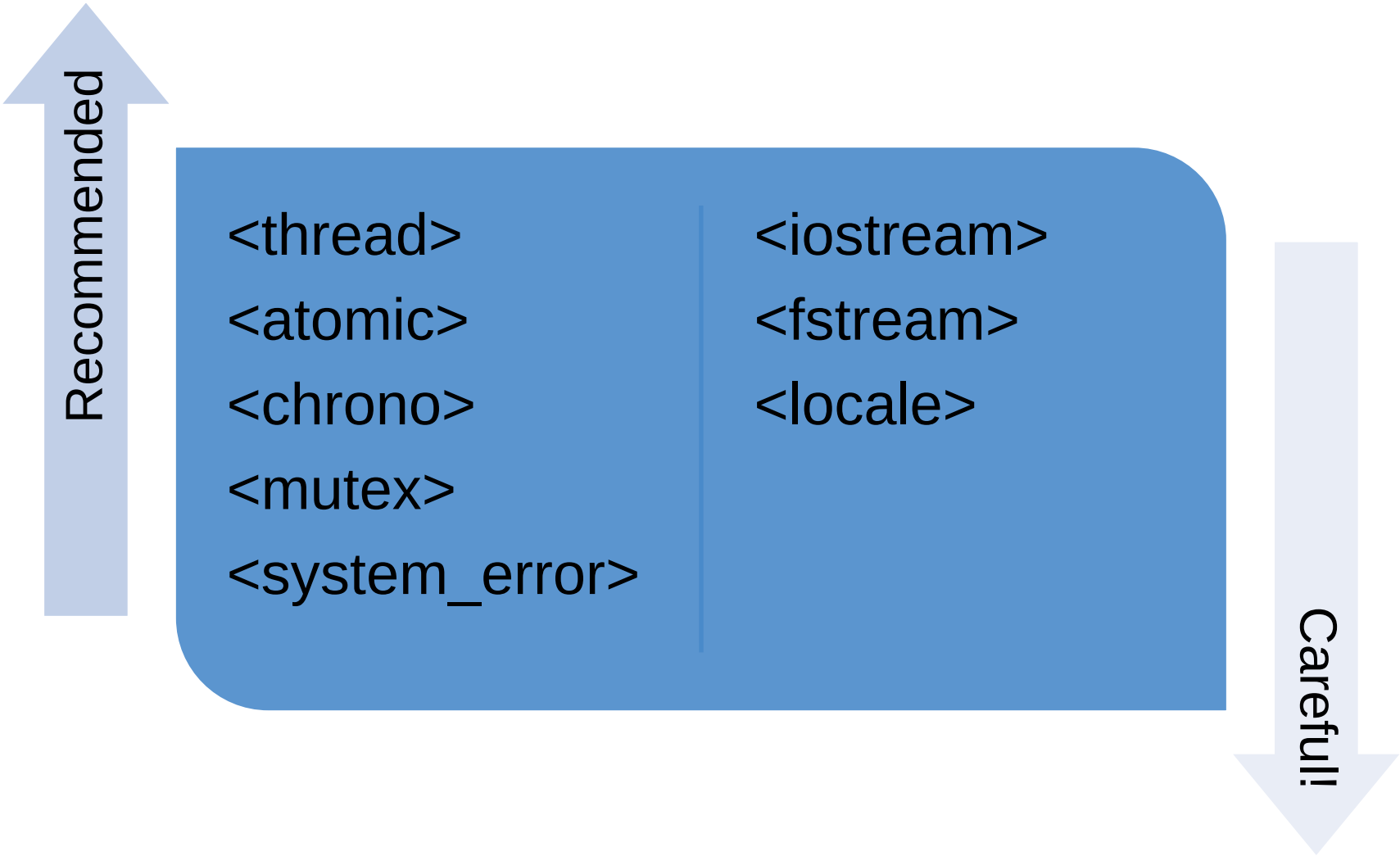
How to compile



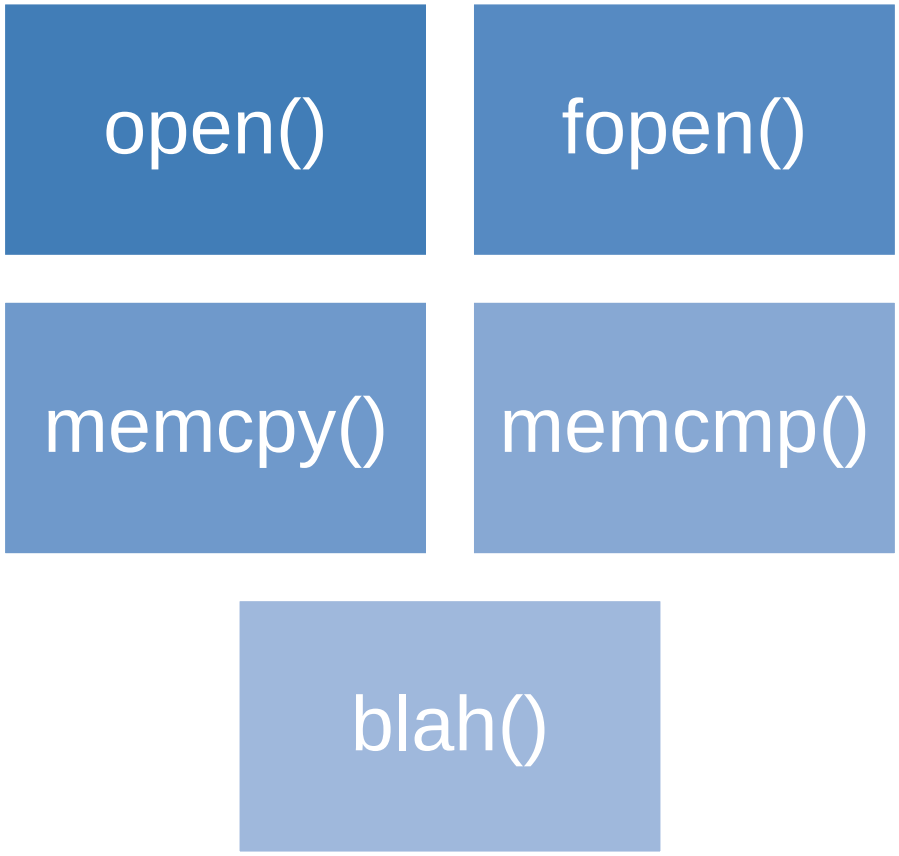
Important trivialities



STL


Recommended
<thread>
<atomic>
<chrono>
<mutex>
<system_error>
<iostream>
<fstream>
<locale>
Careful!

C to the rescue!



You didn’t expect it to be that easy, did you?

```
const unsigned char buf[1] = { 0 };
static_assert(sizeof(buf) == 1, “unexpected size”);

#if BOOST_OS_WINDOWS
  int fd = ::_open(“file”, _O_BINARY | _O_RDONLY, 0);
  ::_lseeki64(fd, 0, SEEK_END);
  ::_write(fd, buf, sizeof(buf));
  ::_close(fd);
#else
  int fd = ::open(“file”, O_RDONLY, 0);
  ::lseek(fd, 0, SEEK_END);
  ::write(fd, buf, sizeof(buf));
  ::close(fd);
#endif
```

Boost.Predef

- Header only and in Boost
- Externalizes the problem
- Simple macros

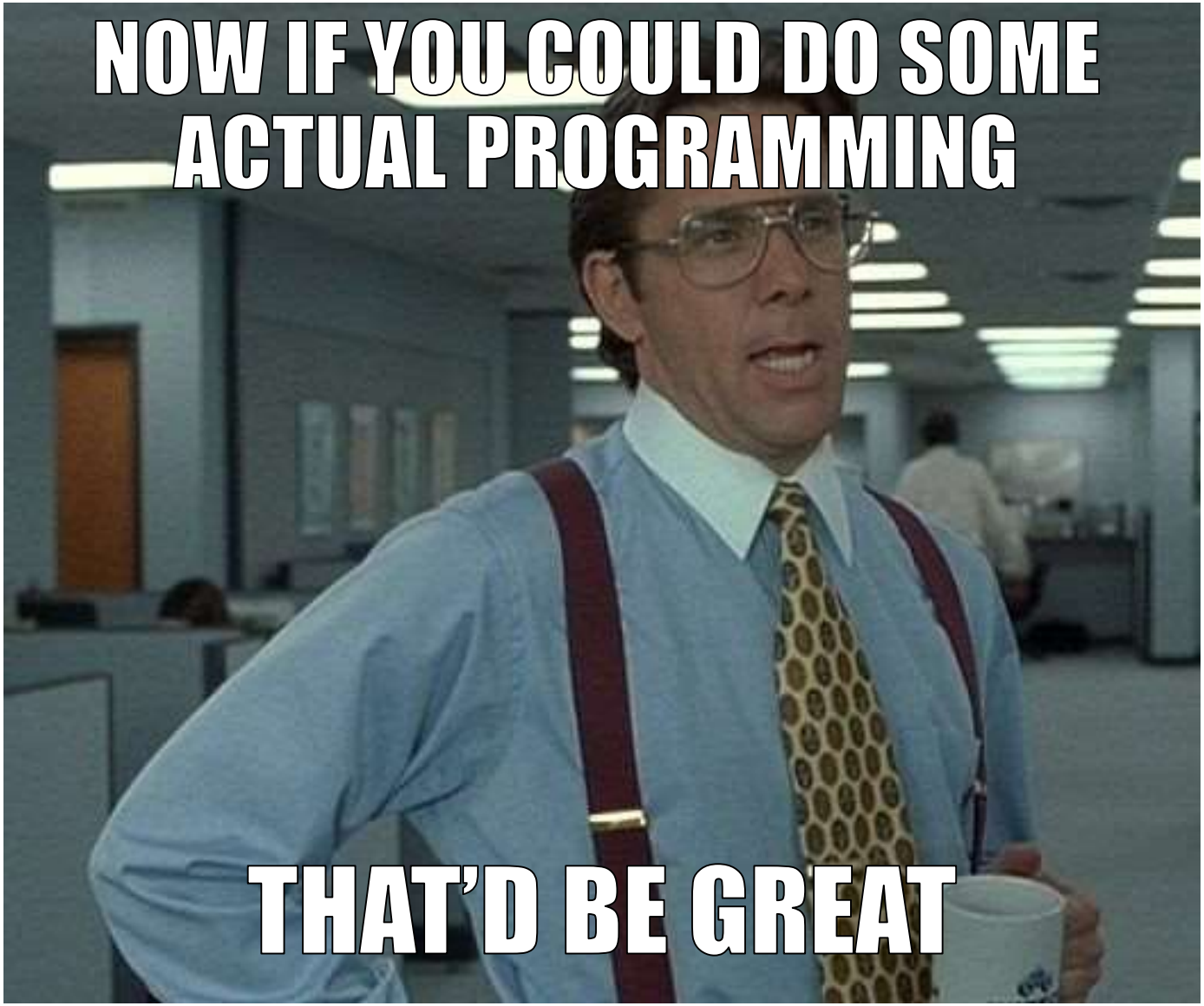
Boost.Predef example 1

```
void func(void)
{
#if BOOST_OS_WINDOWS
        // something Windows
#endif
#if BOOST_OS_BSD_FREE
        // something FreeBSD
#endif
#if BOOST_OS_LINUX
        // something Linux
#endif
}
```

Boost.Predef example 2

```
void func(void)
{
#if BOOST_COMP_GNUC
  static_assert(BOOST_COMP_GNUC
                > BOOST_VERSION_NUMBER(4, 0, 0),
                "invalid gcc version");
#endif

#if BOOST_ARCH_X86_64
        // something AMD64
#endif
#if BOOST_ARCH_IA64
        // something IA64
#endif
}
```


NOW IF YOU COULD DO SOME ACTUAL PROGRAMMING
THAT'D BE GREAT

Windows vs UNIXes – Some major differences

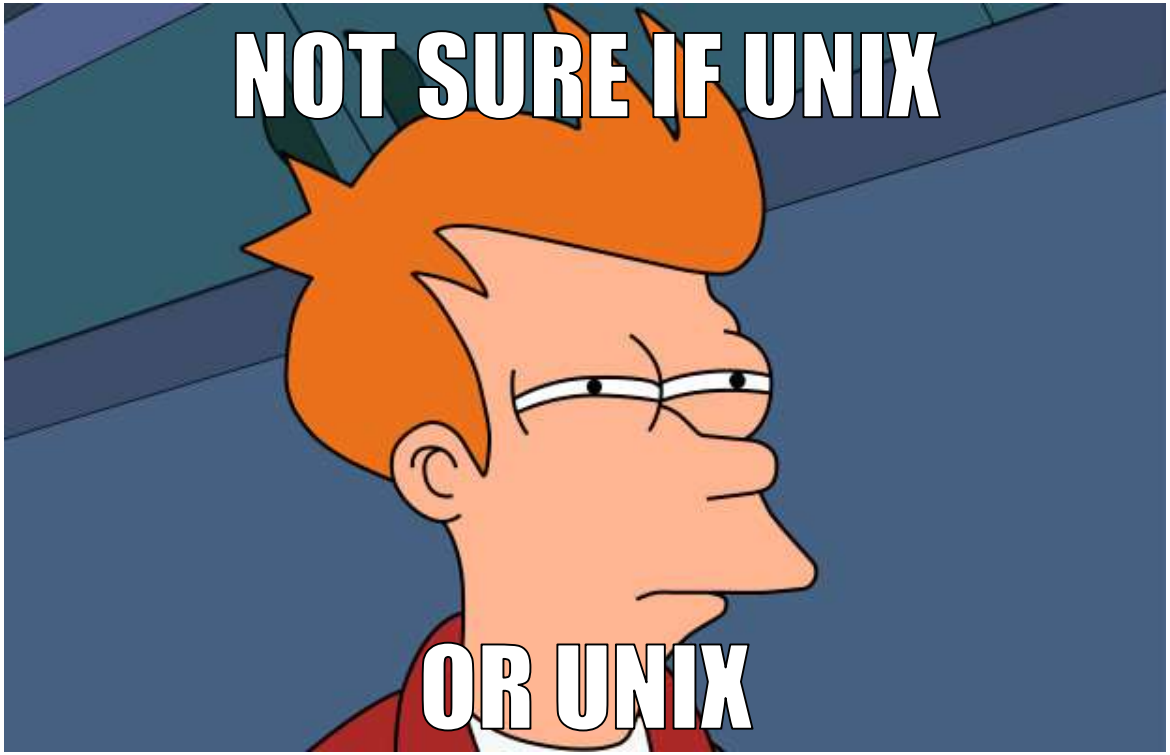
Windows

- UTF-16
- Drive letters, UNC
- GUI
- Local library 1st
- Locks files like there is no tomorrow

UNIX

- Depends
- Mount points
- Terminals
- System library 1st
- Rarely locks files

- Unavailable functions
  - backtrace(), fread_unlocked() (FreeBSD)
- Different configurations
- Different parameters
  - statfs() (FreeBSD vs Linux)
  - sockets (Old UNIXes)
- Different libraries
  - epoll() vs kqueue()
  - libc++ vs stdlibc++
  - glibc versions



Paths

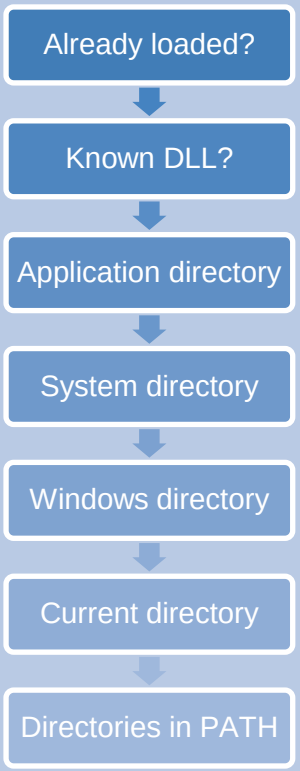
```
C:\Users\Edouard\AppData\Roaming\My Application\Settings
```

```
\\MyServer\Share\Music
```

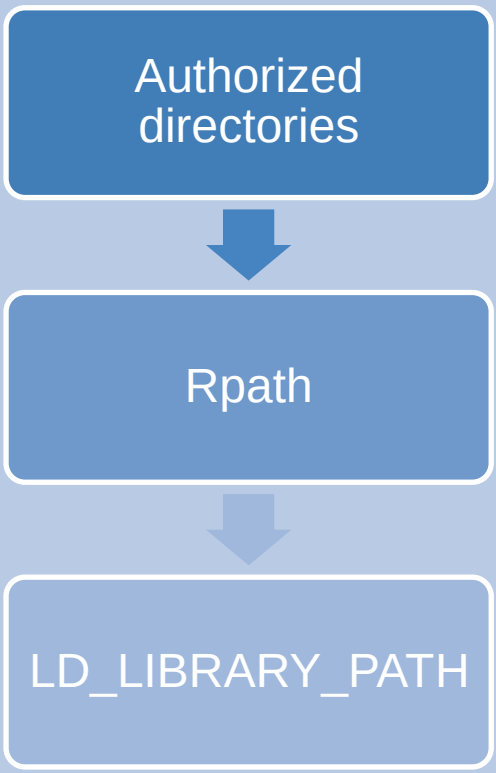
```
~edouard/.app
```

Library search order

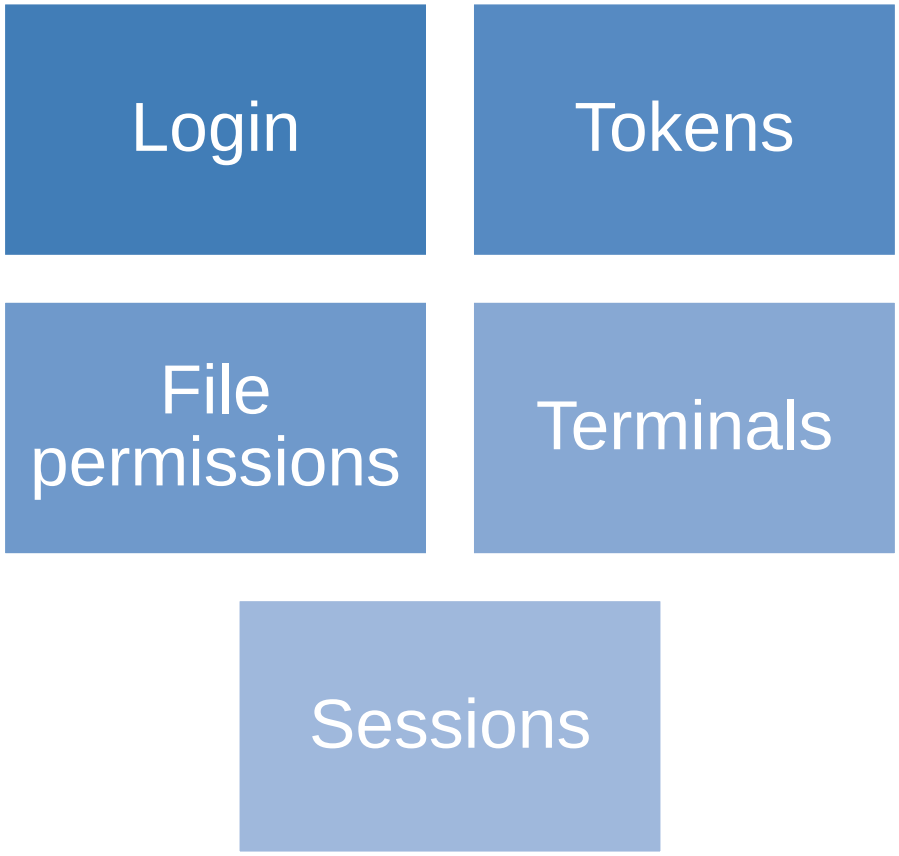
Windows



UNIX

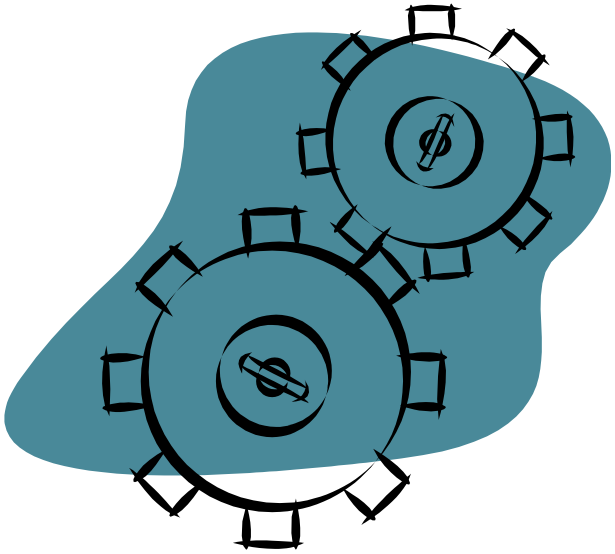


Credentials



Configuration

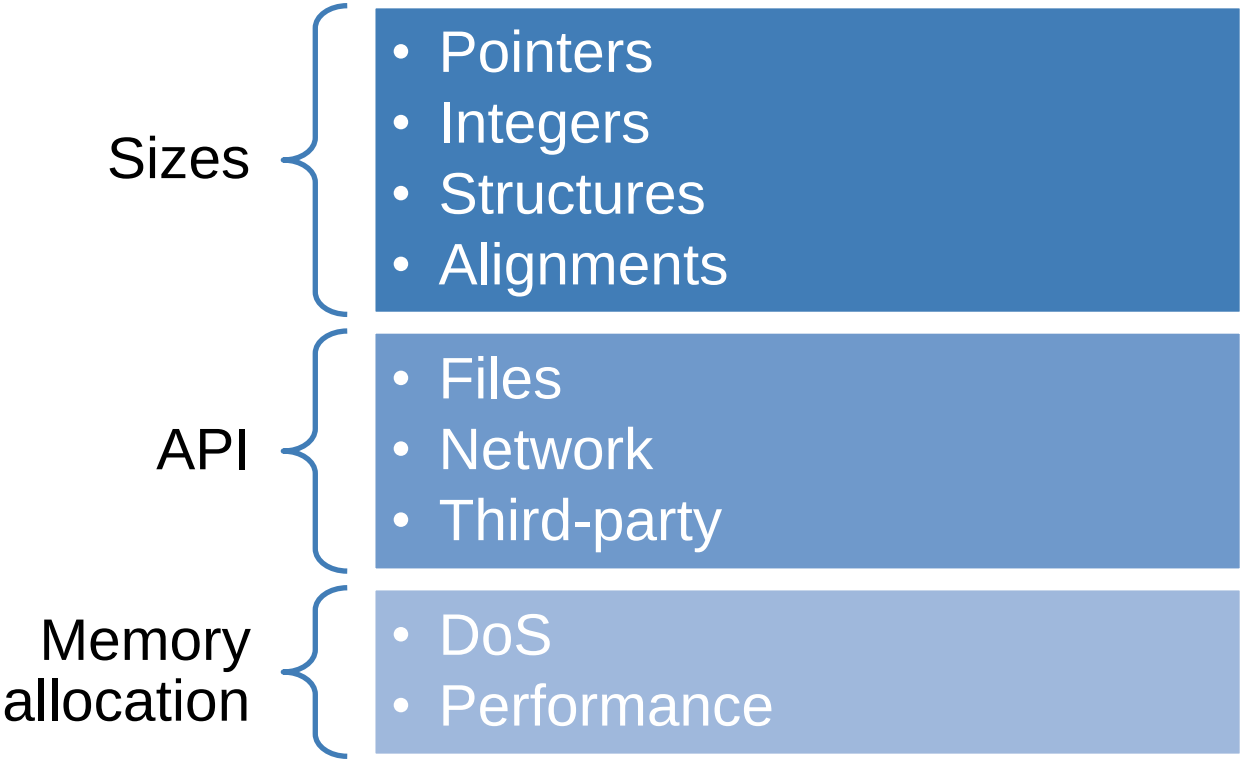
- /proc
- Windows registry
- sysctl
- Configuration files nightmare



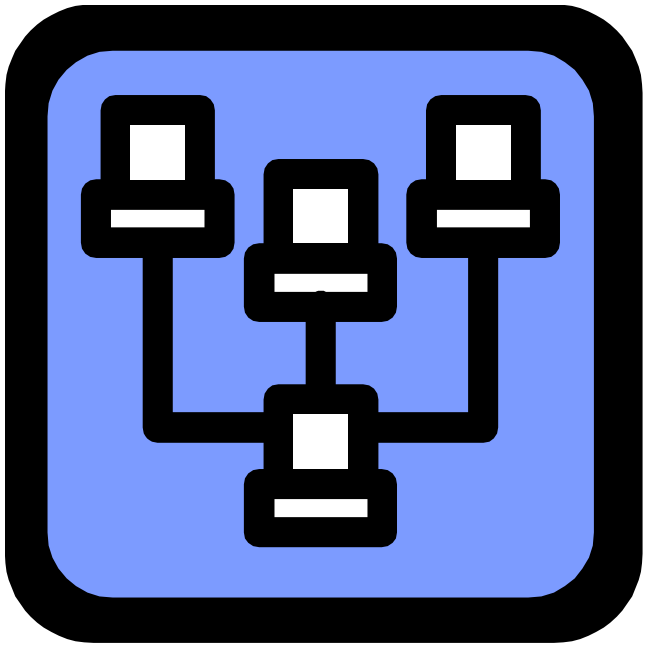
Serialization

- Endianness
- Floats
- Alignment
- Sizes

32-bit vs 64-bit



Networking



Create your own library…
…if you like pain and failure.

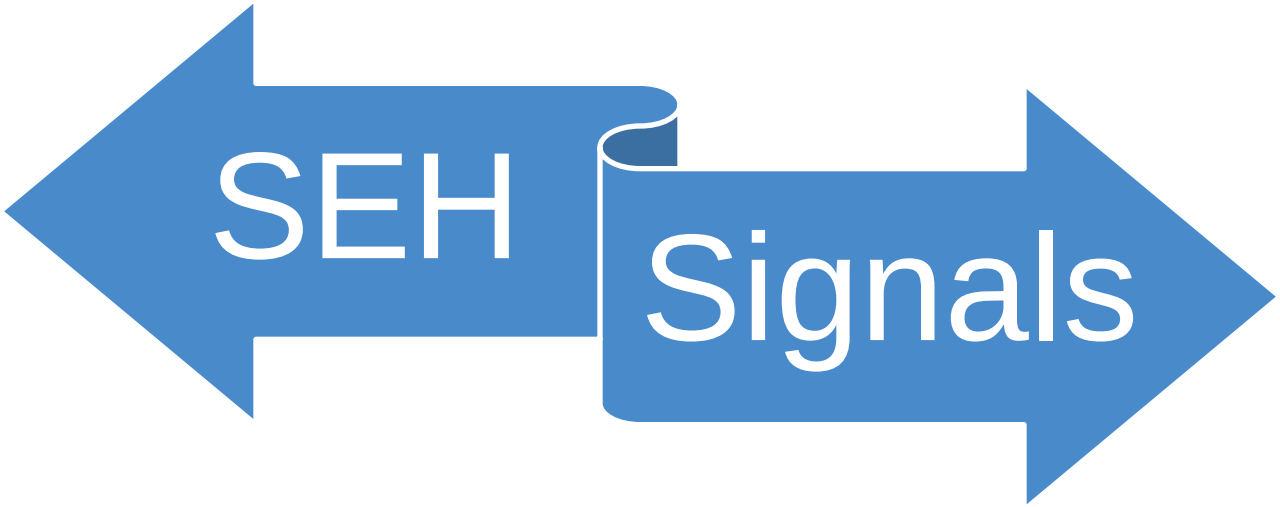
Boost.ASIO custom socket option example

```
#if BOOST_OS_WINDOWS
// on Windows we use the better and more secure
// SO_EXCLUSIVEADDRUSE option
int optval = 1;
auto native_socket = acceptor.native_handle();
if (::setsockopt(native_socket,
    SOL_SOCKET,
    SO_EXCLUSIVEADDRUSE,
    reinterpret_cast<const char *>(&optval), sizeof(optval)) != 0)
{ /* error management */ }
#else
acceptor.set_option(boost::asio::ip::tcp::acceptor::reuse_address(tru
e));
#endif
```

Debugging

- DEBUG=1
- _DEBUG=1
- _SECURE_SCL=1
- _HAS_ITERATOR_DEBUGGING=1
- _GLIBCXX_DEBUG=1

Error management

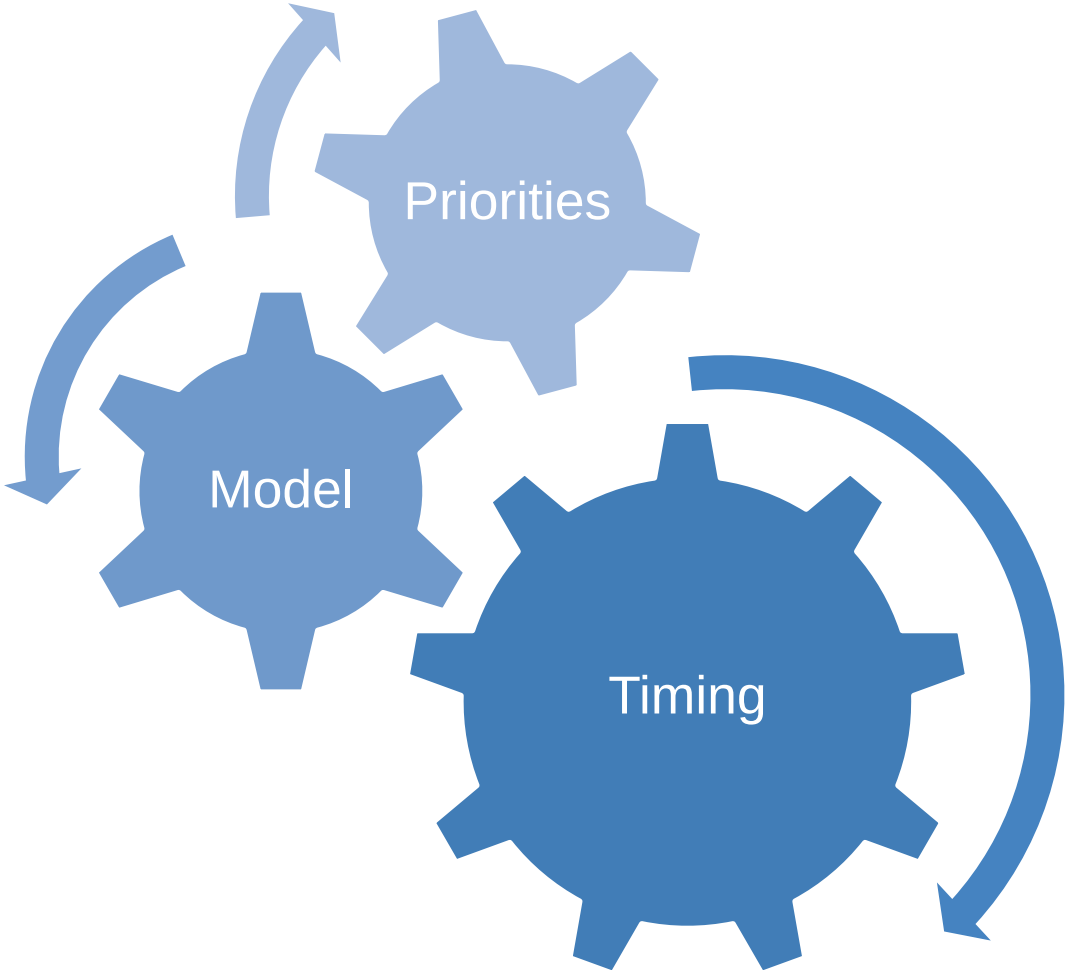


Important announcement

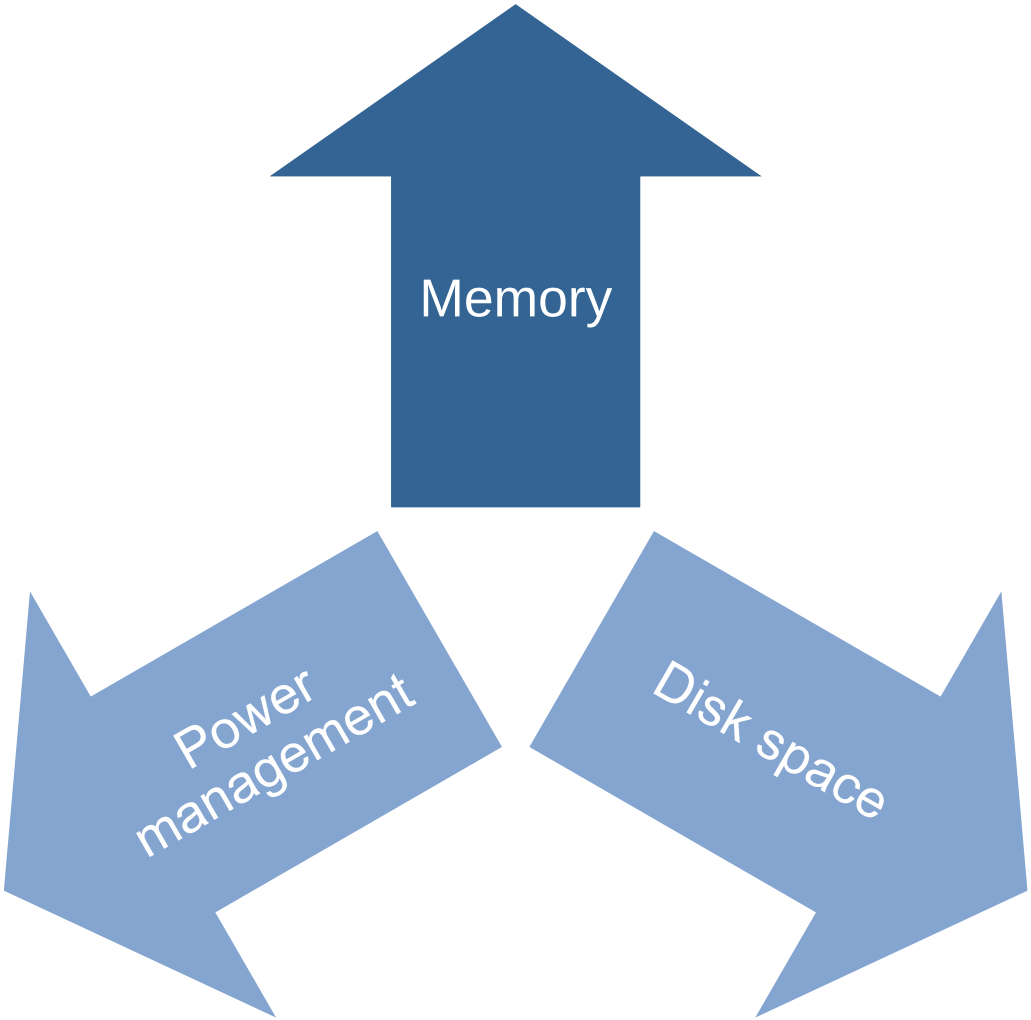
When things get real

Performance discrepancies

Multithreading “issues”



When it goes wrong



Tools of the trade

The Boost libraries
http://www.boost.org/

CMake
http://www.cmake.org/

Buildbot
http://buildbot.net/

Intel Threading Building blocks
http://threadingbuildingblocks.org/

Valgrind
http://valgrind.org/

Microsoft Application Verifier
http://www.microsoft.com/en-us/download/details.aspx?id=20028

Questions and answers

https://www.quasardb.net/
https://www.bureau14.fr/
@edouarda14
edouard@bureau14.fr