Dependencies in C++

Why there is no SerialPort in C++?

oct-2014

I just want a SerialPort

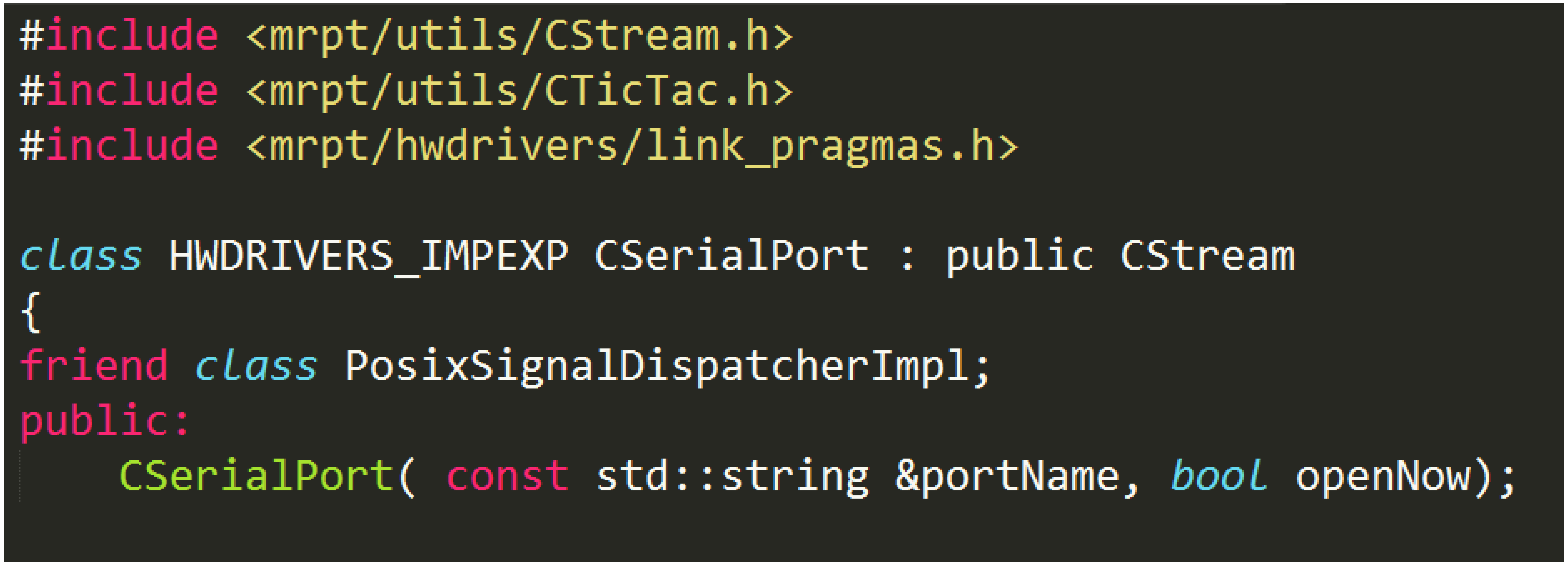
```
$ pip install pyserial
```

```
>>> import serial
>>> ser = serial.Serial(0)
>>> print ser.name
>>> ser.write("hello")
>>> ser.close()
```

Serial Port in MRPT

>5K files, 70 Mb
First class robotics SW.

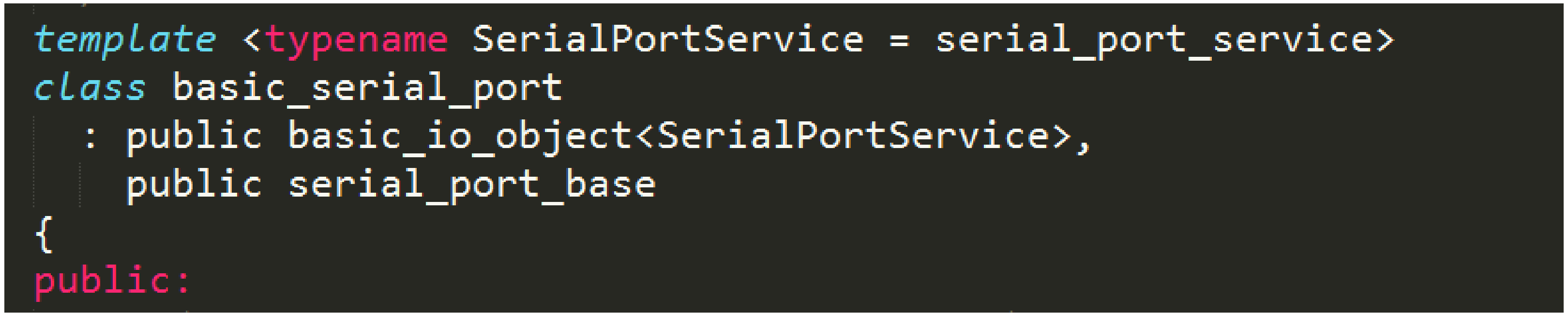
CSerializable, CObject…



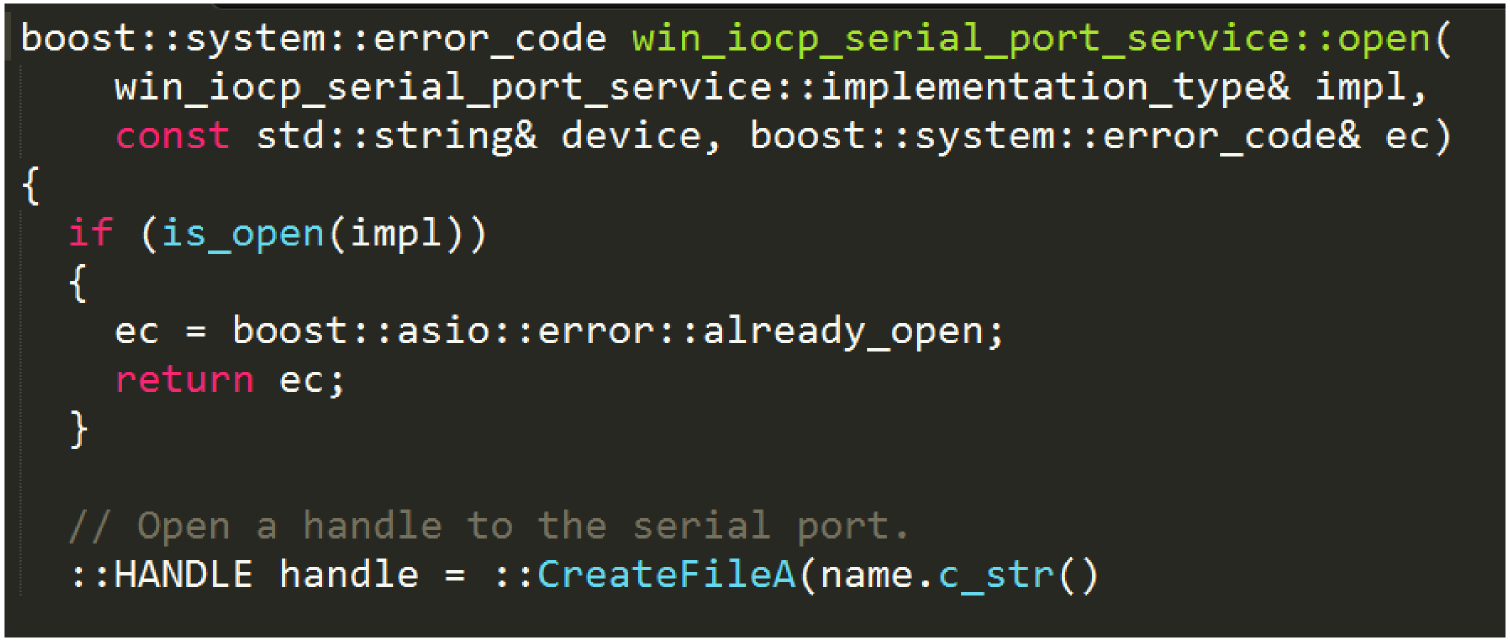
```
#include <mrpt/utils/CStream.h>
#include <mrpt/utils/CTicTac.h>
#include <mrpt/hwdrivers/link_pragmas.h>

class HWDRIVERS_IMPEXP CSerialPort : public CStream
{
friend class PosixSignalDispatcherImpl;
public:
    CSerialPort( const std::string &portName, bool openNow);
```

Serial Port in Boost/Asio



```
template <typename SerialPortService = serial_port_service>
class basic_serial_port
  : public basic_io_object<SerialPortService>,
    public serial_port_base
{
public:
```



```
boost::system::error_code win_iocp_serial_port_service::open(
    win_iocp_serial_port_service::implementation_type& impl,
    const std::string& device, boost::system::error_code& ec)
{
  if (is_open(impl))
  {
    ec = boost::asio::error::already_open;
    return ec;
  }

  // Open a handle to the serial port.
  ::HANDLE handle = ::CreateFileA(name.c_str()
```

CONCLUSION

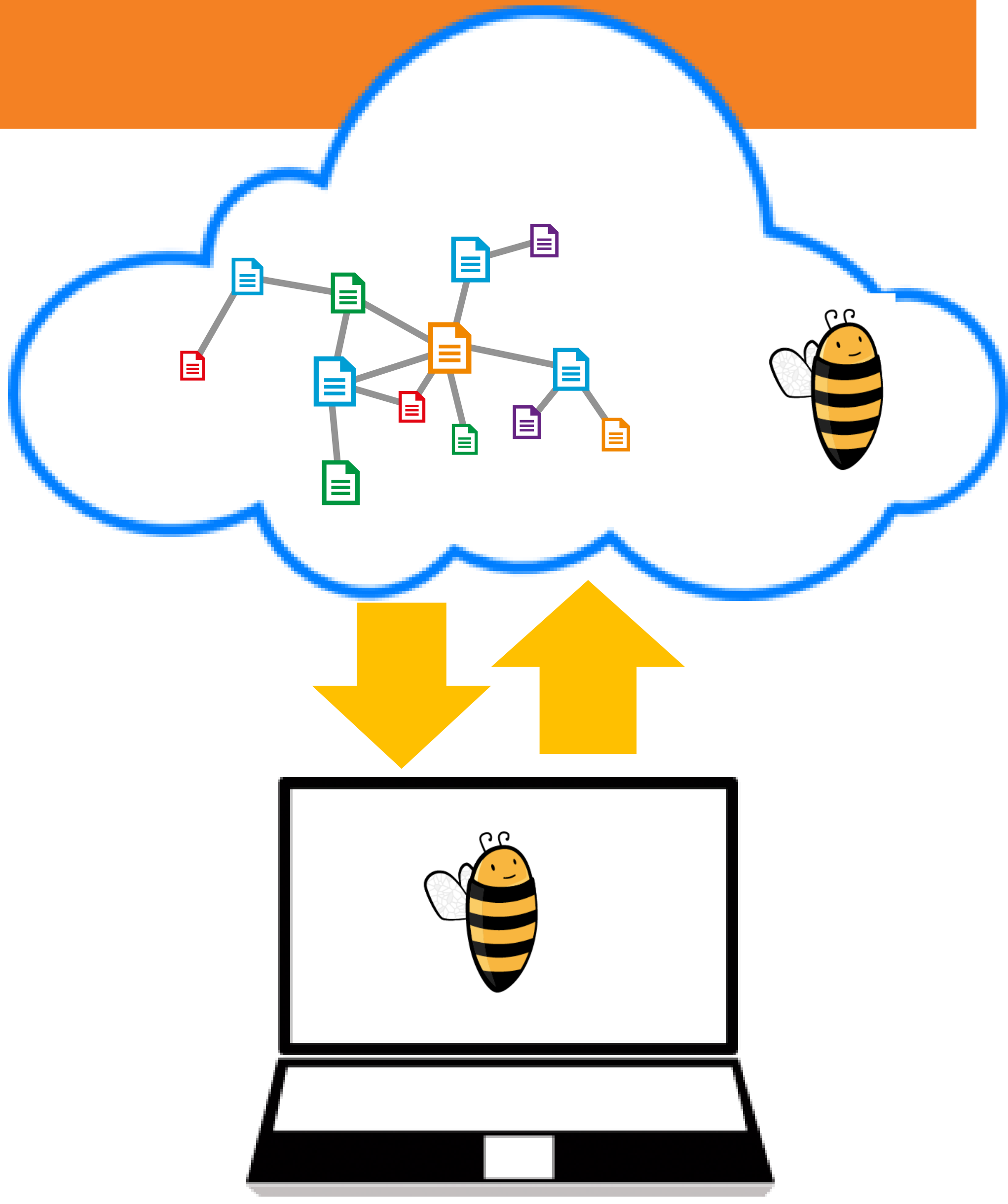
- There is no such “SerialPort” in C/C++

THE PROBLEM: NO DEP MANAGER

- Deps manager:
  - Portable, C/C++, Win, Nix, Mac (& others)
  - Same workflow in all OS
  - Per project specification of deps (like maven pom.xml or python requirements.txt)
    - Different project can depend on different versions of same code, no changes to OS
  - Simple and fast (instant) to upload/share
  - Mainly source based, binaries deps only for caching/speed

EXISTING DEP MANAGERS

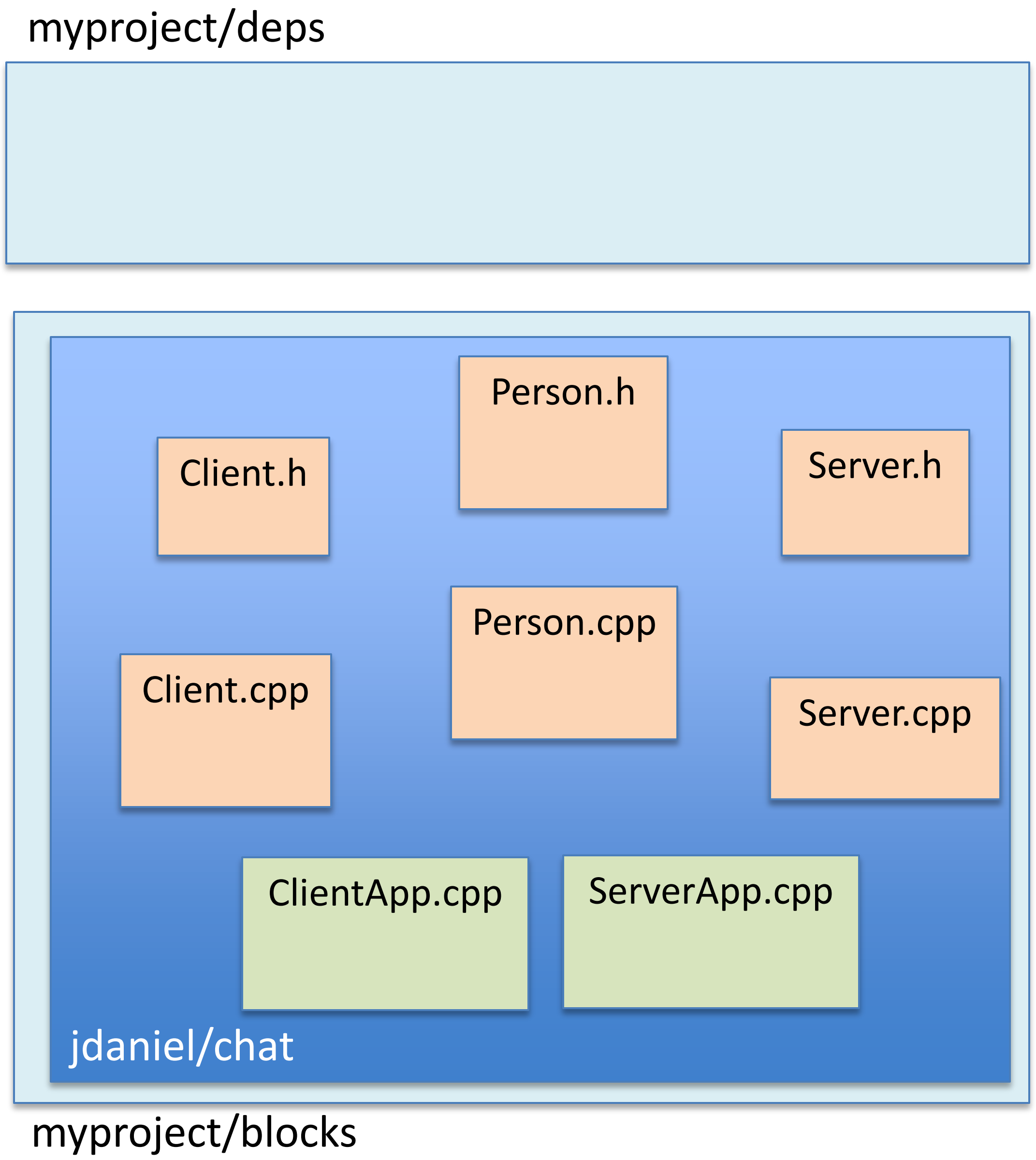
WHAT IS BIICODE?



Initialization

```
$ bii init myproject
$ bii new jdaniel/chat
```

Put files in folder



Processing

$ bii build

- CMake: 2 exe, 1 lib
- Error, missing include

$ bii deps

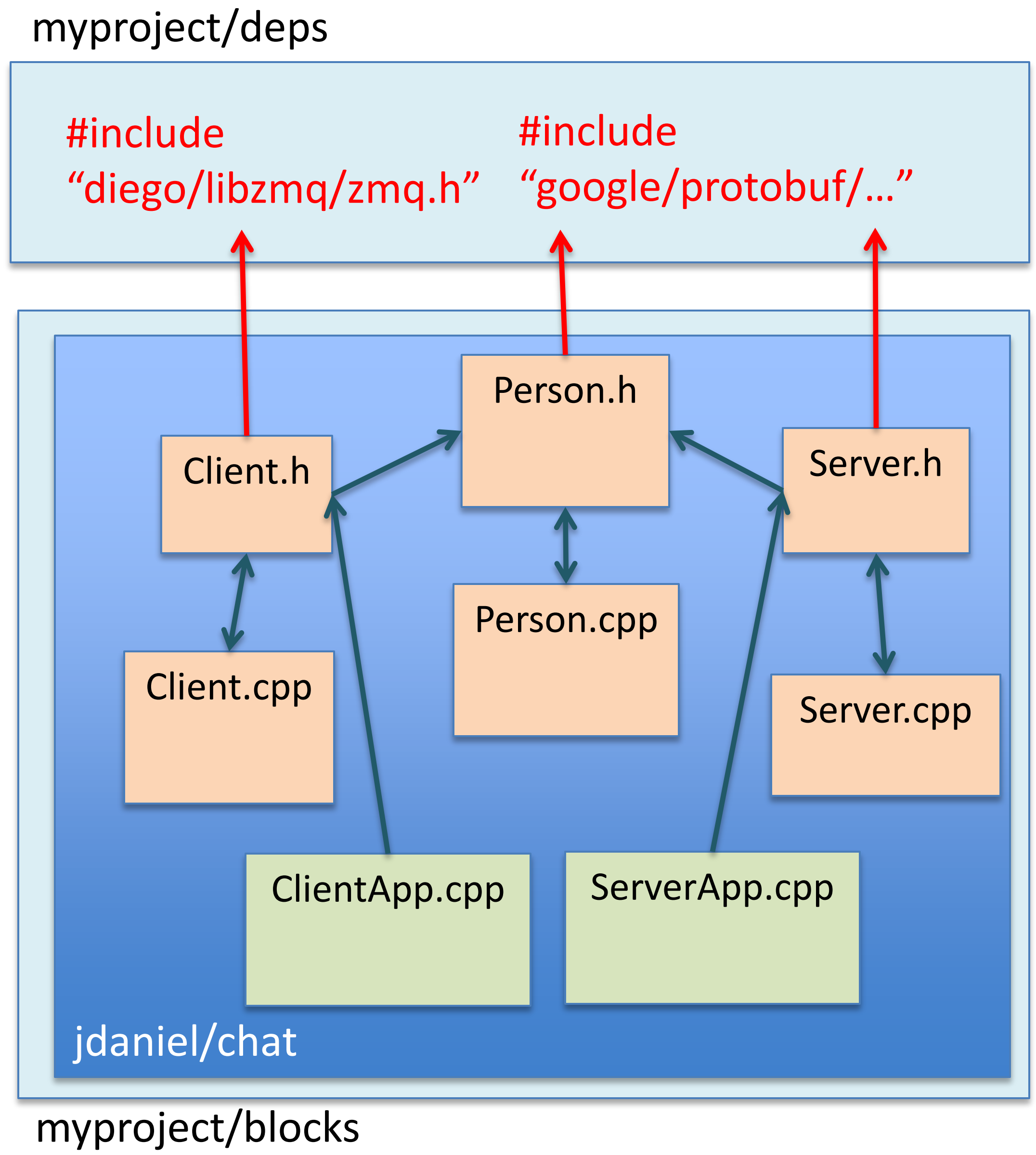
…

unresolved:

-libzmq

-protobuf

-sqlite



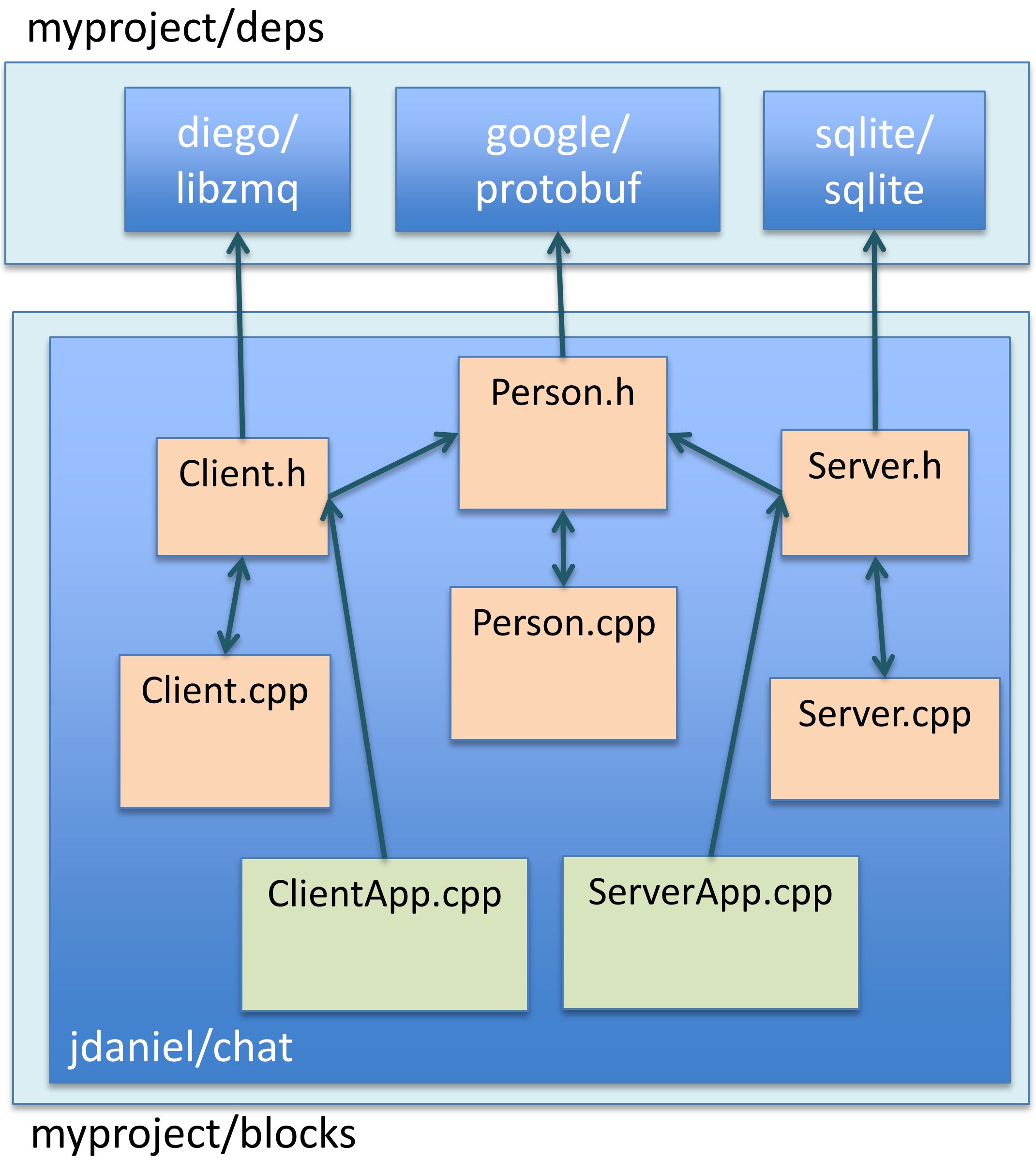
Find & retrieve deps

```
$ bii find
```

or edit “biicode.conf”

```
$ bii build
$ server
$ client
```

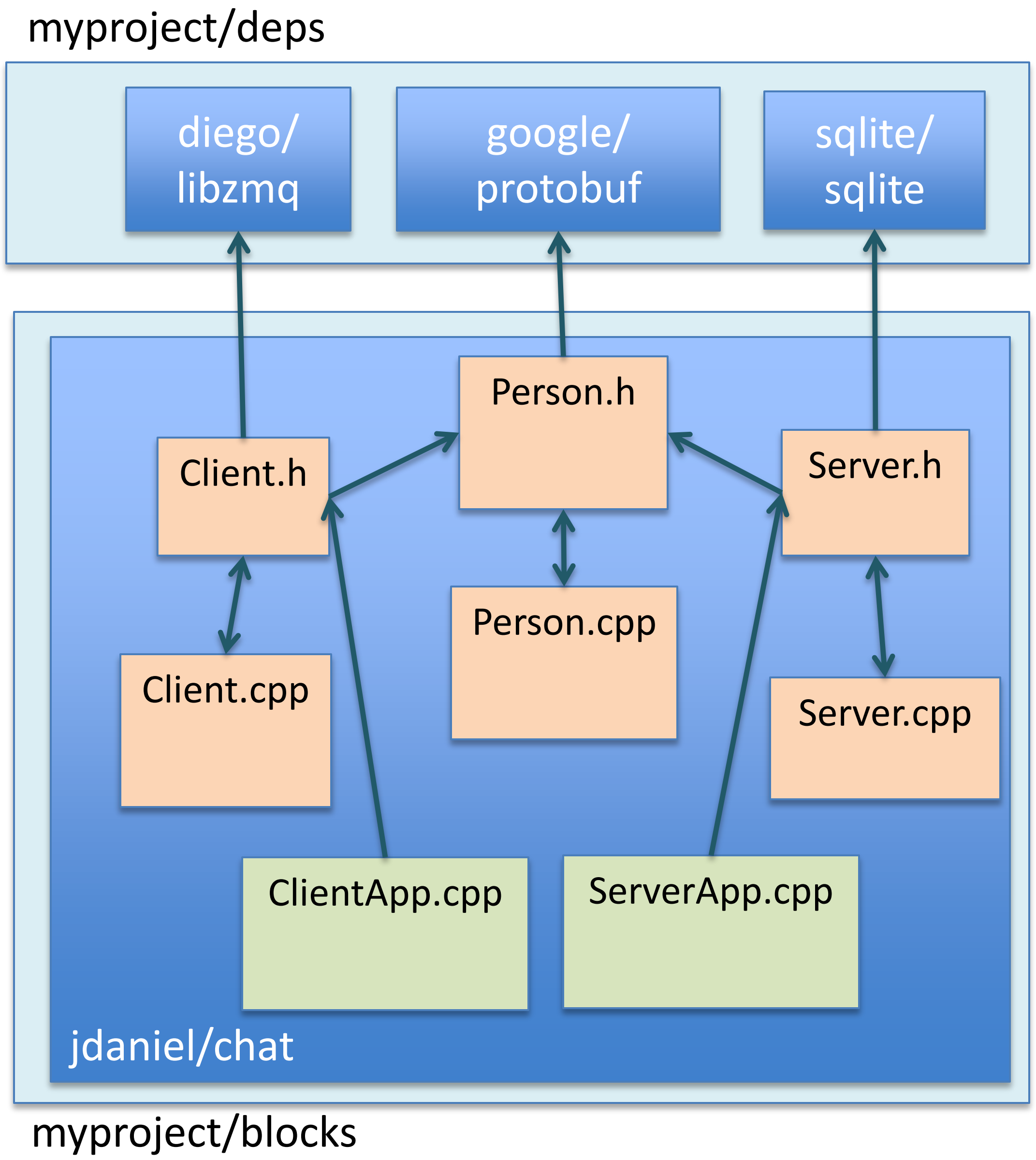
OK!



Publish

$ bii publish

- Uploads sources & graph of jdaniel/chat
- No packaging
- Available in web
- DEV (overwritable)
- ALPHA, BETA, STABLE



Reuse (same or other computer)

```
$ bii init project2
$ bii new other/client
```

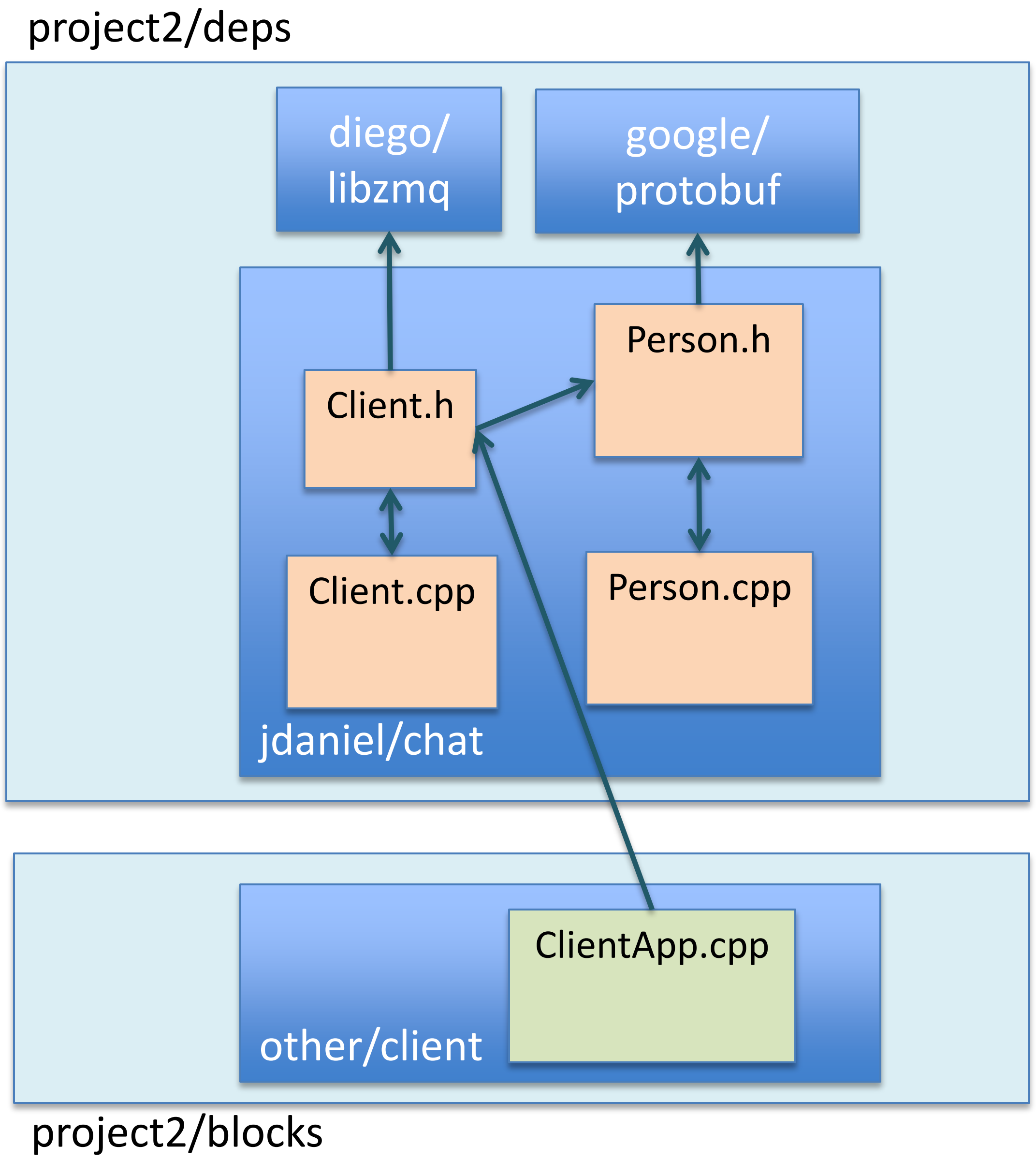
Copy ClientApp.cpp

```
$ bii find
$ bii build
$ client… OK!
```

NO SERVER

NO SQLITE

NO CONFIG



SUMMARY

- Pros:
  - Portable, C/C++, Win, Nix, Mac (& others)
  - Same workflow in all OS
  - Per project specification of deps (like maven pom.xml or python requirements.txt)
    - Different project can depend on different versions of same code, no changes to OS
  - Simple and fast (instant) to upload/share
  - Growing community, hundreds of users
  - SDL, ZMQ, Gtest, Parsers, SFML, Protobufs…
- Cons:
  - Require (minor) modifications to project for new model
  - Large libraries not handled now (Qt, WxWidgets, Boost), impossible to publish. Might be available via hooks for automated external download & setup
  - All sources, might be slow for large repos (proof of concept with binaries)
  - CMake based. Can work in the future with other build systems, as it is independent from it

OPEN SOURCE