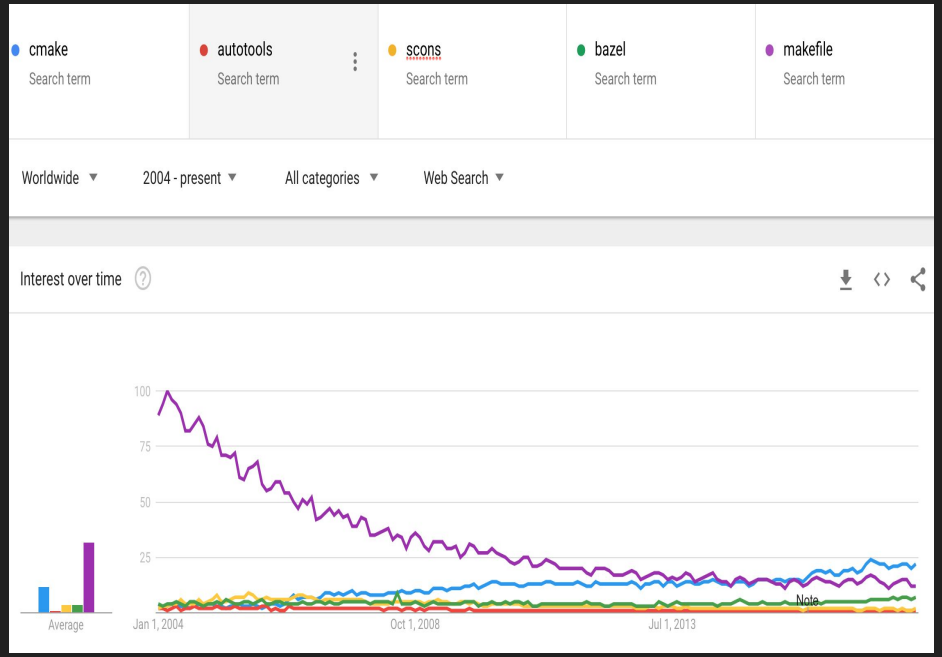
CMake Antipatterns

Geoffrey L Viola

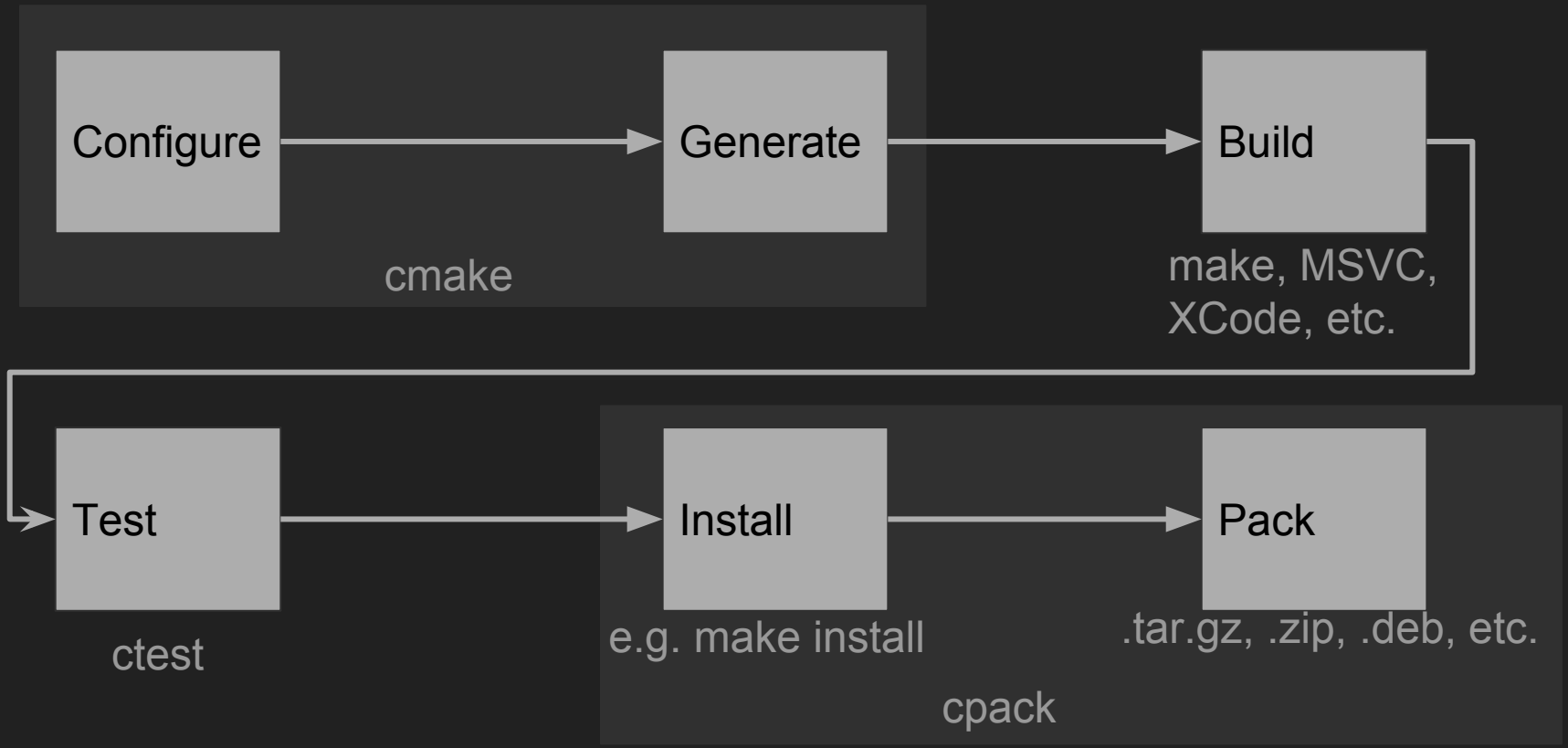
Motivation



Ament

Conan

CMake Pipeline



Antipattern Trends

Set global CMake variables -> call specific CMake function

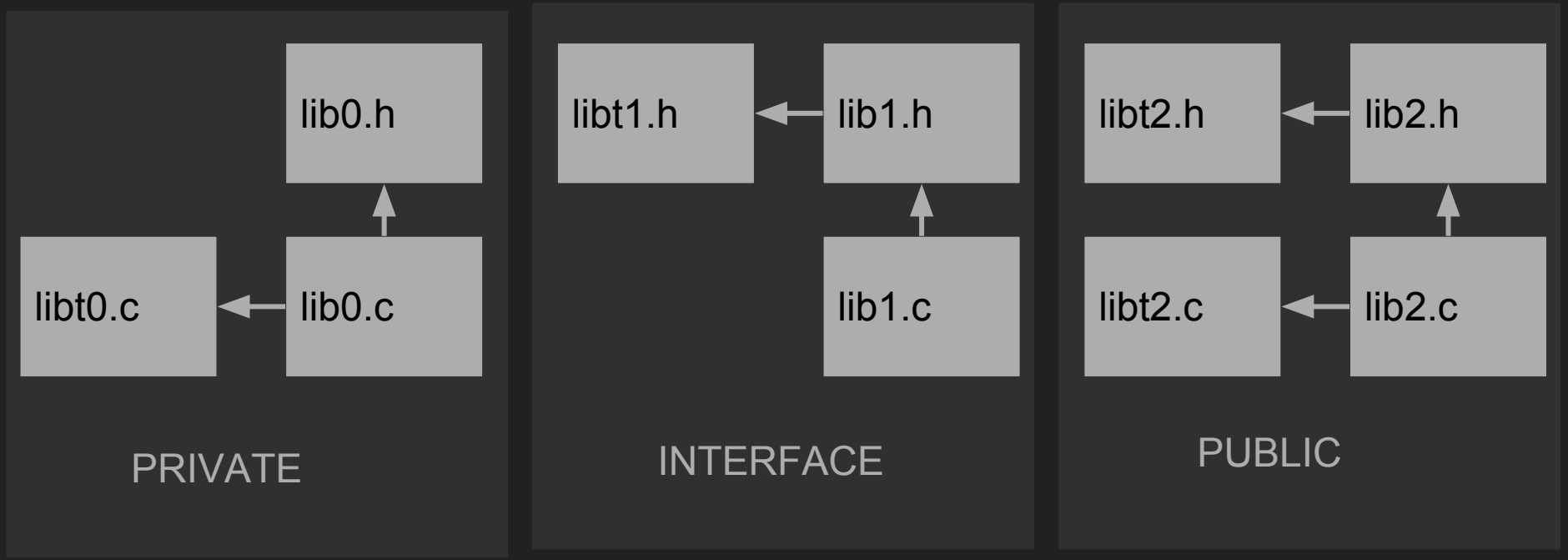
Compiler flags -> define generic requirements

Directory level property setting -> target level settings

Get all or no transitive dependencies -> get only the necessary ones

Running cmake every time to build -> the build system knows what to do

Transitive Libraries Are Public By Default



target_include_directories on an Executable

```
add_library(my_lib include/my_lib/my_lib.h src/my_lib.cc)

target_include_directories(my_lib PUBLIC include)

add_executable(my_executable src/my_executable.cc)

target_link_libraries(my_executable my_lib)

target_include_directories(my_executable include)
```

Linking All the 3rd Party Libraries (CMake 3.5)

```
find_package(Boost 1.58 REQUIRED)

add_executable(my_app my_app.cc)

target_link_libraries(my_app ${Boost_LIBRARIES})


find_package(Boost 1.58 REQUIRED COMPONENTS filesystem)

add_executable(my_app my_app.cc)

target_link_libraries(my_app Boost::filesystem)
```

Threads Via Flags

```
add_executable(threaded_app threaded_app.cc)

target_compile_options(threaded_app “-lpthread”)


find_package(threads)

add_executable(threaded_app threaded_app.cc)

target_link_libraries(threaded_app Threads::Threads)
```

Not Using Install

```
set(RUNTIME_OUTPUT_DIRECTORY ${CMAKE_BINARY_DIR}/bin)


add_executable(my_executable my_executable.cc)

install(TARGETS my_executable RUNTIME DESTINATION bin)


cmake -DCMAKE_INSTALL_PREFIX:PATH=local_install ..
```

File(GLOB...

```
file(GLOB CC_FILES *.cc)

add_executable(my_executable ${CC_FILES})
```

```
add_executable(my_executable 0.cc 1.cc)
```

Inform the build system of new files

E.g. the make command wouldn’t know to rerun CMake after adding files

File(COPY…

```
file(COPY my_file.txt DESTINATION ${CMAKE_BINARY_DIR})
```

```
configure_file(my_file.txt ${CMAKE_BINARY_DIR} COPYONLY)
```

CMake can check the timestamp of the file and know when to copy it

e.g. the make command wouldn’t know to copy the file when it is updated

References

“C++Now 2017: Daniel Pfeifer “Effective CMake”
https://www.youtube.com/watch?v=bsXLMQ6WgIk

“Embracing Modern CMake”
https://www.youtube.com/watch?time_continue=3524&v=JsjI5xr1jxM

CppCon 2017: Mathieu Ropert “Using Modern CMake Patterns to Enforce a Good Modular Design” https://www.youtube.com/watch?v=eC9-iRN2b04

http://www.aosabook.org/en/cmake.html