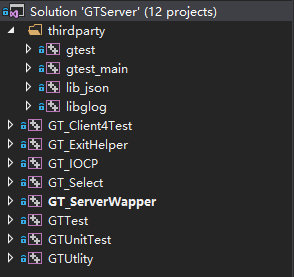
**GT\_Server Architect**

1. For what?

GT\_IOCP is a project for build a network server, and use windows IOCP network model.

1. Directory Tree



**gtest && gtest\_main**: is the project for unit test

**lib\_json**: is the lib for config

l**ib\_glog:** is the lib for GT LOG

**GT\_Client4Test**: is the project for test server

**GT\_ExitHelper**: is the project for help GT\_Server exit service,(GT Server ignore Ctrl+C and close button)

**GT\_IOCP:** core project for iocp.dll

**GT\_Select**: core project for select.dll

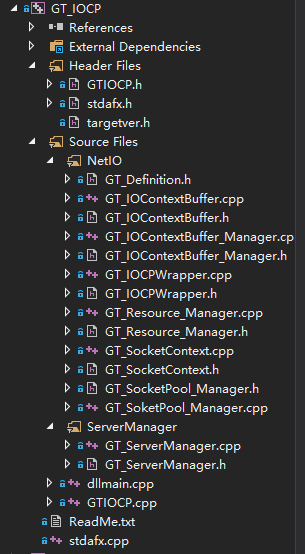
**GT\_ServerWrapper**: serve as a platform for IOCP and Select module

**GT\_Test**: contain some test

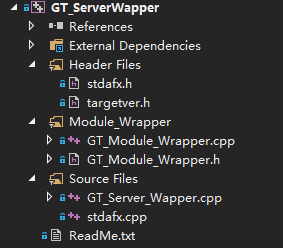
**GT\_UnitTest**: contain the whole unit test for the project

**GT\_Utility:** contain the utility interface and wrapper of third party

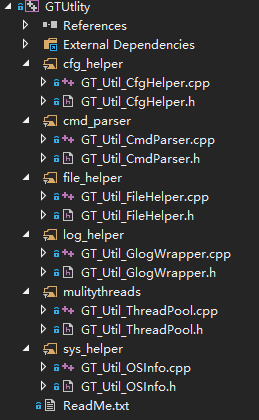
1. Main Project file means:
2. GT\_IOCP



1. GTIOCP.h/.cpp is the interface of the iocp.dll
2. GT\_IOContextBuffer.h/.cpp serve as the PER\_IO\_EVENT\_HANDLE of IOCP
3. GT\_IOContextBuffer\_Manager.h/.cpp serve as the resource manager of IOContextBuffer
4. GT\_IOCPWrapper.h/.cpp is the core logical file for IOCP
5. GT\_SocketContext.h/.cpp serve as the PER\_SOCK\_HANDLE of IOCP
6. GT\_SocketPool\_Manager.h/.cpp serve as the socket resource pool for SocketContext
7. GT\_Resource\_Manager.h/.cpp serve as the whole resource manager of the service, and it is the only interface for service use resource.
8. GT\_ServerManager.h/.cpp serve as the only interface for dll interface use IOCP service, and it is a wrapper of IOCP Wrapper and Resource Manager.
9. GT\_Definition.h define the error code and enum struct
10. GT\_ServerWrapper



1. GT\_Module\_Wrapper.h/.cpp serve as the wrapper of IOCP and Select
2. GT\_Server\_Wapper.cpp is the main function for the GT\_Server.exe
3. GTUtlity



1. cfg\_helper: is a wrapper of lib\_json
2. cmd\_parser: is a simple of the command line parser
3. file\_helper: is a function set of file operation
4. log\_helper: is the glog.lib wrapper
5. mulitythread: serve as a multi\_thread pool manager object use c++11 thread
6. sys\_helper: is a function set for supply some sys request, such as get num of cpu or get current folder.
7. Main Logical
8. GT\_IOCP

SOCKET\_CONTEXT🡪PER\_HANDLE\_DATA

IOBUFFER\_CONTEXT🡪PER\_IO\_DATA

There are two parts of the IOCP: Resource and Logical.

Resource:

**GT\_Resource\_Manager**: serve as the whole resource manager, and it control the PER\_HANDLE\_DATA and PER\_IO\_DATA resource and it has a cache for the PER\_HANDLE\_DATA, it contain **two long time worker thread** for resource collect, **one is for the unused socket handle collect**, and **one is for the connection checker**.

**Thread for handle collector:** check is the socket inuse pool got invalid socket handle, if got then push it to the temprate pool which will be merge to unused pool later.

**Connection checker thread:** PER\_HANDLE\_DATA have a time control member which will record the time that the handle is activity, when the handle do not active and have exceed the time limit, the thread will release it and push it to the PER\_HANDLE\_DATA cache.

Resouce Manager have a cache:

**std::*map*<*ULONG\_PTR*, SOCKETCONTEXT\_SHAREPTR> completion\_key\_ptr\_cache\_**

**use map and use the address of the PER\_HANDLE\_DATA as the key, and it will make it easer for find the PER\_HANDLE\_DATA shared\_ptr which serve as the memory protector.**

**GT\_SocketContext:** is a PER\_HANDLE\_DATA struct, and it contains: socket\_handle, socket\_type, socket\_addr, socket\_io\_buffer\_cache, time\_control.

1. Socket\_handle: record the who is serve for
2. Socket\_type: is use to distinguish the LISTEN\_SOCKET and ACCEPT socket, because when connection checker check the PER\_HANLDE\_DATA is active or not, we should the ignore the Listen socket which need wait for the new accept socket, but for the accepted socket we care its active time, if it exceed the time we should collect its resource. It is like a heart package.
3. Socket\_addr: record the client address
4. Socket\_io\_buffer\_cache: is use for record the PER\_IO\_DATA the accepted socket handle has send.
5. Time\_control: is use for record the time that the socket handle last event occurred.

GT\_SocketContext do not need allocate resource, because the resource have already allocate by the socket\_pool\_manager and iobuffercontext\_manager.

**GT\_IOBufferContext\_:** contain socket\_handle\_ptr, WSABUF, iobuffer\_, iobuffer\_size, io\_event\_type

1. Socket\_handle\_ptr: record who is the PER\_IO\_DATA serve for
2. WSABUF: this struct is use to recv and send the data
3. Io\_buffer\_: is the real buffer which WSABUF use
4. Io\_buffer\_size: buffer length
5. Io\_event\_type: there five types: IO\_EVENT\_NULL, IO\_EVENT\_ACCEPT,IO\_EVENT\_READ, IO\_EVENT\_WRITE, IO\_EVENT\_EXIT.

**GT\_IOBufferContext\_Manager:** IOBufferContext can be use by its manager, and when service initialize it will preallocate num of IOContex for later use, so that it will save the time for allocate at when need.

**GT\_Soket\_pool\_manager:** allocate socket for later use, and it contain three struct: unused\_socket\_pool, tobe\_use\_socket\_pool, inuse\_socket\_pool

1. Unuse\_socket pool is serve as the pool of the preallocate sockets, and if the service need a PER\_HANDLE\_DATA it will get the socket from this pool.
2. Inuse\_socket\_pool: when a PER\_HANDLE\_DATA allocated, the socket which used will push to the inuse\_socket\_pool.
3. Tobe\_use\_socket\_pool: when the socket allocated and is save in the inuse\_socket\_pool, the connection checker thread will check the inuse\_socket\_pool, of the socket which is setted to INVALID\_SOCEKT, we will collect it from the inuse\_socket\_pool and push it to the tobe\_use\_socket\_pool, when the unused\_socket\_pool size smaller than the num which control by the cfg, we will reallocate some sockets, but before allocate we will check the tobe\_use\_socket\_pool, if the pool is not empty, we will merge the socket to the unused\_socket\_pool for reuse.

**GT\_IOCP\_Wrapper:** this is the main logic of the IOCP, this will introducted by another paper.

1. GT\_Select