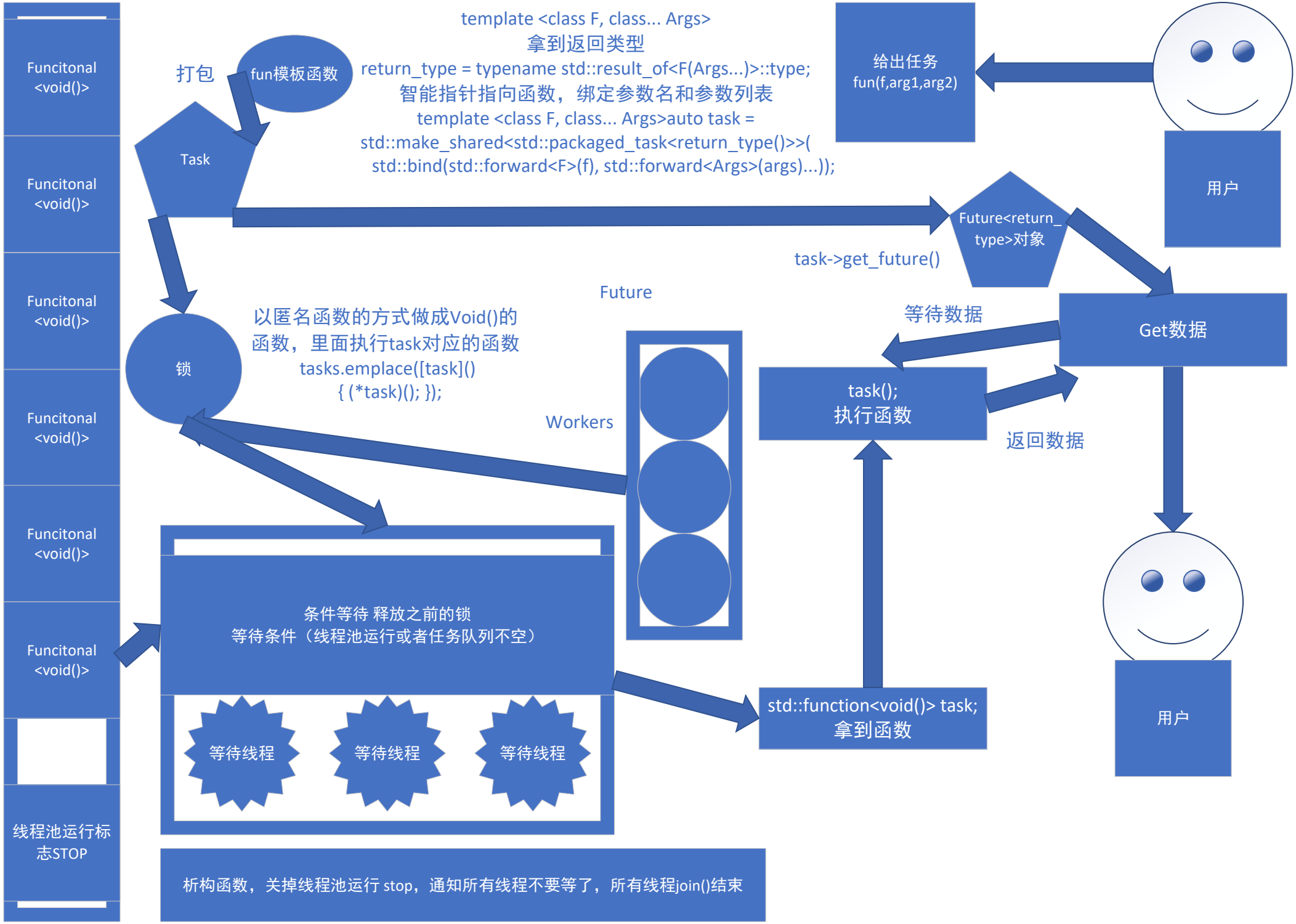
Funcitonal <void()>
Funcitonal <void()>
Funcitonal <void()>
Funcitonal <void()>
Funcitonal <void()>
Funcitonal <void()>
线程池运行标志STOP
打包
fun模板函数
Task
template <class F, class... Args>
拿到返回类型
return_type = typename std::result_of<F(Args...)>::type;
智能指针指向函数，绑定参数名和参数列表
template <class F, class... Args>auto task =
std::make_shared<std::packaged_task<return_type()>>(
std::bind(std::forward<F>(f), std::forward<Args>(args)...));
给出任务
fun(f,arg1,arg2)
用户
Future<return_type>对象
task->get_future()
Future
锁
以匿名函数的方式做成Void()的函数，里面执行task对应的函数
tasks.emplace([task]()
{ (*task)(); });
Workers
等待数据
Get数据
task();
执行函数
返回数据
条件等待 释放之前的锁
等待条件（线程池运行或者任务队列不空）
等待线程
等待线程
等待线程
std::function<void()> task;
拿到函数
用户
析构函数，关掉线程池运行 stop，通知所有线程不要等了，所有线程join()结束