TMTVision SDK

──────── 底层架构2.0Beta

目录

[一、 模块化多线程编程 2](#_Toc451758962)

[二、 使用方法 4](#_Toc451758963)

[三、 进度与改进 5](#_Toc451758964)

# 模块化多线程编程

为了方便搭建多线程程序，实现模块化多线程编程。采用生产者~消费者模式，定义了节点模块INode接口，和链路模块Link类。link类。

模块中定义了：

1. 节点模块接口INode，采用json格式字符串进行设置；
2. 消费者模块IConsumer<T>：消费数据，提供写数据函数Write（运算符<<），清空函数Clear（运算符--），以及外触发函数Click；
3. 生产者模块IProductor<T>：生产数据，提供读数据函数 Read（运算符<<），更新函数Update（运算符++），以及外触发函数Click；
4. 处理者模块IProcessor<T1,T2>：处理数据，提供处理数据函数Process（运算符()），以及外触发函数Click；
5. 数据链路模块Link<T1,T2>：驱动数据流动方向为：生产者->处理者（可选）->消费者，内置独立线程用于内触发，也可采用Click方法外触发；
6. 数据缓存节点DataBuffer<T>：同时继承生产者IConsumer、生产者IProductor，利用环形静态队列实现数据缓冲；
7. 网络节点SocketNode< T1,T2>：同时继承生产者IConsumer、生产者IProductor，用于网络设备读写；
8. 文件节点FileNode< T >：未完成；
9. 数据库节点DataBaseNode< T1,T2>：未完成。

上述内容调用了SK中的两个基类：

1. 线程管理类BThread：基于boost:: thread的线程类，内置多种中断管理、外触发、多线程保护、负载分析等功能，派生其Task函数实现自定义功能。
2. boost:: circular\_buffer：为固定内存和头尾指针实现的环形存储结构，可以实现队列Queue和堆栈Stack功能。
3. PointerPool：静态数组形式存储指针，实现了查找、添加删除等功能。

继承和调用关系如图所示：

图1 继承和调用关系

# 使用方法

由于BufferDevice等均派生自Device基类，具有相同的IO读写函数接口>>和<<，可以自由组合串联；IO接口内置多线程保护，支持并联模式。

数据类DeData包含重复比较和生命周期管理方法；线程指针MotoLink含工作频率分析；数据设备类BufferDevice含存储空间分析方法，便于分析负载。使用方法如图所示：

|  |  |
| --- | --- |
| a.异步读设备处理并写入缓存 | b. 异步并行读写和处理数据 |
| c.利用网络设备实现多进程数据交互 | |

图2使用方法的多样性

# 进度与改进

图3 进度

进一步改进：需要增加回调函数和消息接口，增加控制方法。



*2016/6/13*

[*smartsupervisor@163.com*](mailto:smartsupervisor@163.com)

[*towanglei@163.com*](mailto:towanglei@163.com)