脑机接口客户端设计文档

1 本文档由黎永基编写。黎永基为系统主要开发者，从2017年11月8日至2018年1月31日兼职开发。

项目地址：https://github.com/gzweehai/BrainInterface

2 术语约定：硬件工程师将数据采集器称为放大器，对于客户端来说就是服务器。而硬件工程师一般将客户端称为上位机，放大器也称为下位机

3 系统概述

放大器收到开始采集命令后开始采样，数据源源不断按照一定间隔发送到客户端，数据是通过Socket/TcpIp协议发送的，客户端根据配置的滤波参数调用相应算法过滤并显示。

在开始采样前，客户端应该首先获取放大器设备参数，因为解码电压值以及滤波算法跟采样频率、放大倍数等参数相关。

目前已完成连接放大器，采样数据收集显示，阻抗显示，简单滤波功能（中值滤波，低通、高通、带通、带阻滤波，小波分析），滤波功能配置参数的界面未完善，只能通过修改配置文件控制滤波参数。

波形显示使用了商用的SchiChart类库，需要购买license。

后续需开发采样数据回放系统，对采样数据进行标记，对接病人信息系统，上传数据到服务器等功能。

4 整体设计思路

网络层(BrainNetwork)，支持函数(BrainCommon)，本地数据存储(DataAccess)，UI显示(32EEGChannel)分别作为子系统放在不同工程中。所有工程均可以依赖BrainCommon工程。

网络层负责连接放大器，编码命令，解码放大器回应数据包的工作。

解码电压值和阻抗值放在BrainCommon工程，因多个子系统均需要调用这个功能

DataAccess项目负责保存采样数据到文件和回放采样文件。

BrainCommon工程还包含配置文件的读写

BrainSimulator是一个放大器模拟器，用于测试。

BrainProtocolTester工程用于测试各个子系统，也有各个主要函数使用的例子。

连接各个子系统主要通过Rx的数据流完成。

例如网络层输出原始采样数据流，UI层解码电压值后转换为电压值/时间点的数据流，UI层主窗口，通道窗口等UI界面则订阅这些数据流来显示。

DataAccess层通过订阅原始数据流保存文件，回放文件的时候输出数据流给需要的子系统使用。

5 技术考虑

尽量使用Rx完成异步数据流操作，保证模块化

完成功能后才考虑如何优化

算法库使用开源的math-filtering和wavelet studio，算法如果效率不够高，应考虑使用SIMD指令/CUDA/OpenCL对矩阵向量计算进行优化

使用缓冲池管理大量数组分配/释放

尽量避免使用lock进行多线程同步，改用CAS指令完成（Interlocked类）

与显示不直接相关的功能不要放在UI线程执行，UI线程只执行对UI对象的更新（这个是强制的否则报错）

尽量避免类型转换，既低效又增加出错的机会

配置文件用json格式，灵活而不需要考虑效率问题（使用json.net）

6 小技巧

通过double dispatching可以减少类型转换

通过聚合实现多个滤波器的串行组合

CAS指令能够有效提高并发和效率，实现无锁算法

虚拟机安装SchiChart，在试用期结束后可以还原虚拟机状态重装SchiChart，开发的时候就可以不购买license