AeroCardio本地存储+网络转存方案：

1. 未调整前的逻辑（当前的版本）：App将数据缓存在内存当中（长度可调整），连接上Netty实时数据流存储服务器之后将缓存中数据直接上传。

2. 调整之后的逻辑：

1. App按照固定的时间长度T缓存心电。这个缓存可以是简单的内存缓存，或者考虑使用MMAP
2. 当Netty数据流存储服务器超过固定时间长度时，App直接将1中的数据转存到本地存储上，同时清空当前的缓存。
3. App重新开始新的时间长度为T的心电数据缓存。
4. 如果在任意一个时间长度T之内，App连接上了Netty服务器，则App将当前这一个时间长度内所有的心电数据全部上传至Netty数据流存储服务器。这个操作不影响App的本地数据存储逻辑，即App仍然按照固定时间长度定时向本地进行数据存储。
5. 本地存储设置一个最大的存储长度Max\_T，一旦超出这个长度则自动覆盖最早的数据记录，工作机制类似行车记录仪。
6. **小结：即在当前的Netty服务器交互机制上增加一个内存缓存+本地数据存储，这个逻辑并不影响之前的Netty服务器交互机制**

3. 本地存储数据处理逻辑：

1. 已经缓存在本地的数据，如果需要上传与同步，建议另外架设一个ChunkData上传服务器，不走当前的数据流服务器
2. 交互建议：
   1. **小改动方案**：上述所有逻辑全部在后台进行，只是在数据流服务器未连接上的时候进行友好的提示，告知用户当前App转入本地存储模式
   2. **中改动方案**：App默认为本地存储模式，当可以连接数据流服务器时（服务器在线），友好提示用户是否选择云存储模式，如果选择云存储模式，则另外进行服务收费
   3. **大改动方案**：用户可自主选择本地存储或者云存储，本地存储的话则所有分析功能App端需要实现，包括数据回放，数据分析，数据报告的导出，同时数据自动接入后端，ChunData服务器和数据流服务器两者自动完成数据的整合存储，数据的存储改为文件存储+MongoDb文件索引，文件存储按照固定的时间颗粒度进行存储。