# 原始需求

Python语言创建共享内存，实现进程间的通信。主要方案如下：

1、watchdog读取配置文件，创建共享内存（为各被监控进程分配一块内存，比如监控10个进程，以配置文件的标识号分配大小为10字节的共享内存），初始化全为0，然后拉起各被监控进程。

2、被监控进程启动时打开共享内存。运行时，往本进程对应的共享内存置1喂狗，watchdog周期性去读该共享内存，如果内存块数据为1说明进程正常，此时置0。如果内存块数据为0，说明该位对应的进程异常，然后杀掉重启对应的进程。

# 协作图

Watch dog

Task--n

创建n个共享内次对象

Share Memory

每隔1秒循环检查每个变量，是否为0，如果为0， 则将该进程的技术加一；如果进程技术加大到大雨10，则kill进程，重启进程；否则设置内存对象=0

打开自己的共享内存对象

新建一个线程，每隔1秒，就设置共享内存对象=1

# 软件需求

## 监控进程（watch dog）

1. 启动进程
2. 读取配置文件，知道被监控的进程个数，以及重启进程的命令行参数
3. 创建一批共享内存对象（10个）
4. 新建一线程。循环检查每个共享内存对象是否为0
5. 如果“共享内存对象”==0， 设置该进程的计数+1，如果进程计数> 10，则kill该进程，将进程句柄，进程计数全部设置为0. 重启进程。
6. 如果“共享内存对象”==1， 设置该进程的计数=0
7. Sleep(xx)
8. 循环

问题：重启动被监控进程， 我建议放一个启动进程的命令行列表的配置文件。

## 被监控进程

1. 启动进程
2. 创建一个新的线程，打开共享的内存对象
3. 设置内存对象=1
4. Sleep（xxx）
5. 循环

问题：怎么知道该进程应该是哪个共享对象？我建议通过启动时的命令行参数

# Python创建共享内存

# Python打开共享内存

# Python创建进程

# Python创建线程

# 完