旧系统的不足：

1. selenium和firefox升级时版本对应问题

> 原系统是采用selenium驱动+Firefox来进行抓取数据，selenium和Firefox是有对应的版本，所以在升级selenium的时候一定要升级Firefox，不然就会出错；在selenium3.0之后，需要安装geckodriver.exe可执行文件，并在脚本中添加可执行文件的路径，方可启动浏览器，geckodriver.exe可执行文件和Firefox之间也是有对应的版本。
>
> - 某些旧的数据类型不再被支持
> - 需要额外下载driver

2. 部署繁琐

> 原系统的代码是打包成jar包，然后去每一台集群机器上部署启动，如果集群的数量一旦很多，那么在部署上面就很繁琐，需要挨个登录每一台机器；一旦升级版本，也是需要挨个登录每台机器。
>
> - 部署
> - 升级

3. 伪分布式

- 只是数量上的分布式

> 原系统中是采用单线程以及增加集群机器的数量来进行数据抓取，这样带来的问题就是一个脚本只能在一台主机上抓取数据，无法做到多台主机协作共同抓取一个网站。

- 任务分配不均匀

> 原系统中的任务分配策略是随机分配的，每台机器上最多分配3个任务，这就带来一个问题，如果有一些网站的数据量特别大，然后被分配在一台机器上，这就造成了，有的机器一直空闲着，有的机器一直在干活，达不到负载均衡，在抓取效率上也是不行的。

- 单线程，没有充分利用CPU资源

> 原系统是采用单线程的方式进行抓取数据，这就没有充分利用CPU资源，如果使用多线程+异步的方式，那么抓取效率会大大提高。

4. 可用性不高

> 原系统采用的是一台scheduler+多台从机集群的方式，这个带来的问题是如果scheduler宕机，那么集群的机器就会全部宕机，都需要重新启动
>
> - scheduler挂了
> - 从机挂了

5. 环境不兼容

> 原系统是部署在windows机器上，如果迁移到linux机器上部署的时候，就会有问题。
>
> - windows迁移到Linux有代价大。

    - 需要复杂配置
    - 目前采用的解决方案只适用于部分Linux版本

6. 扩展性差

> 原先系统是一台服务器+N台集群主机，是一对多的情况，无法扩展为多对多的情况，另外，每新增一个网站就要新写一个脚本。
>
> - 抓取页面和解析，都放在一起。其实抓取部分可以通过用xml文件来编排操作过程。

7. 断点重爬的实现不够优雅

> 原先系统存在断电虫爬的机制，但是非常复杂，不够优雅/通用.
>
> - 保留抓取的keyword
> - 保留URL