# 问题1：大量死锁问题

## 问题表现

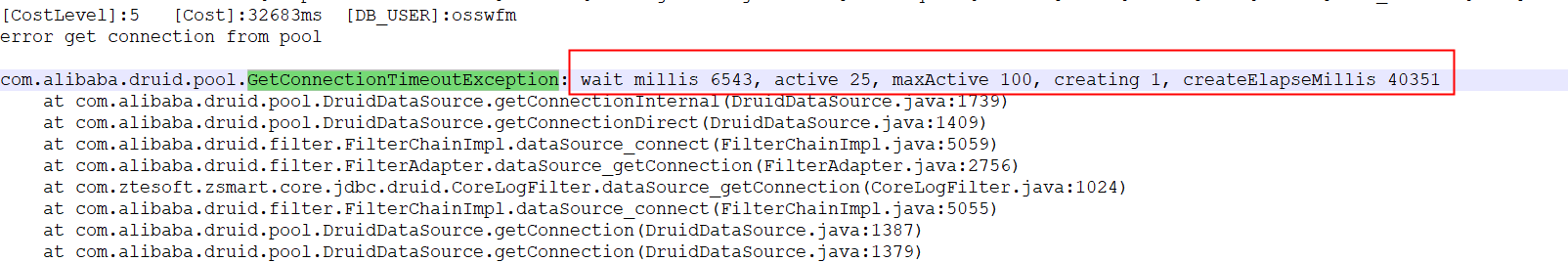
Wfm前端访问页面加载不出来

## 问题出现时间

2023-11-01 10:00 — 2023-11-01 22:00

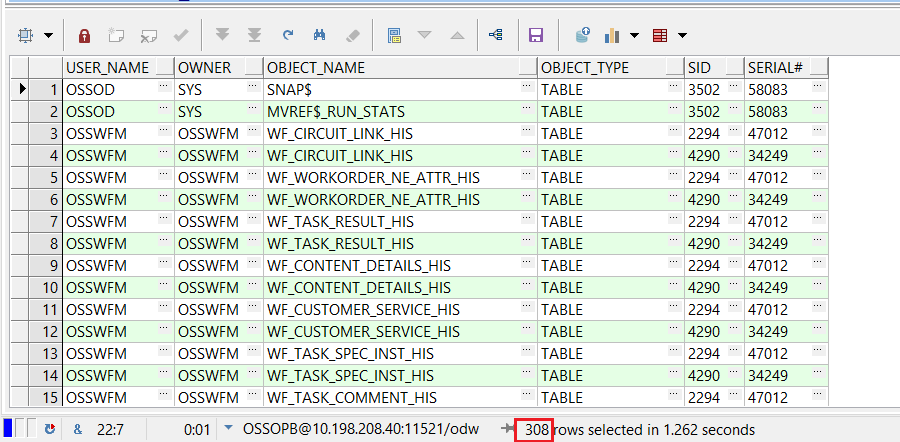
## 问题分析过程

#### 首先查看wfm业务日志，发现数据库连接超时问题

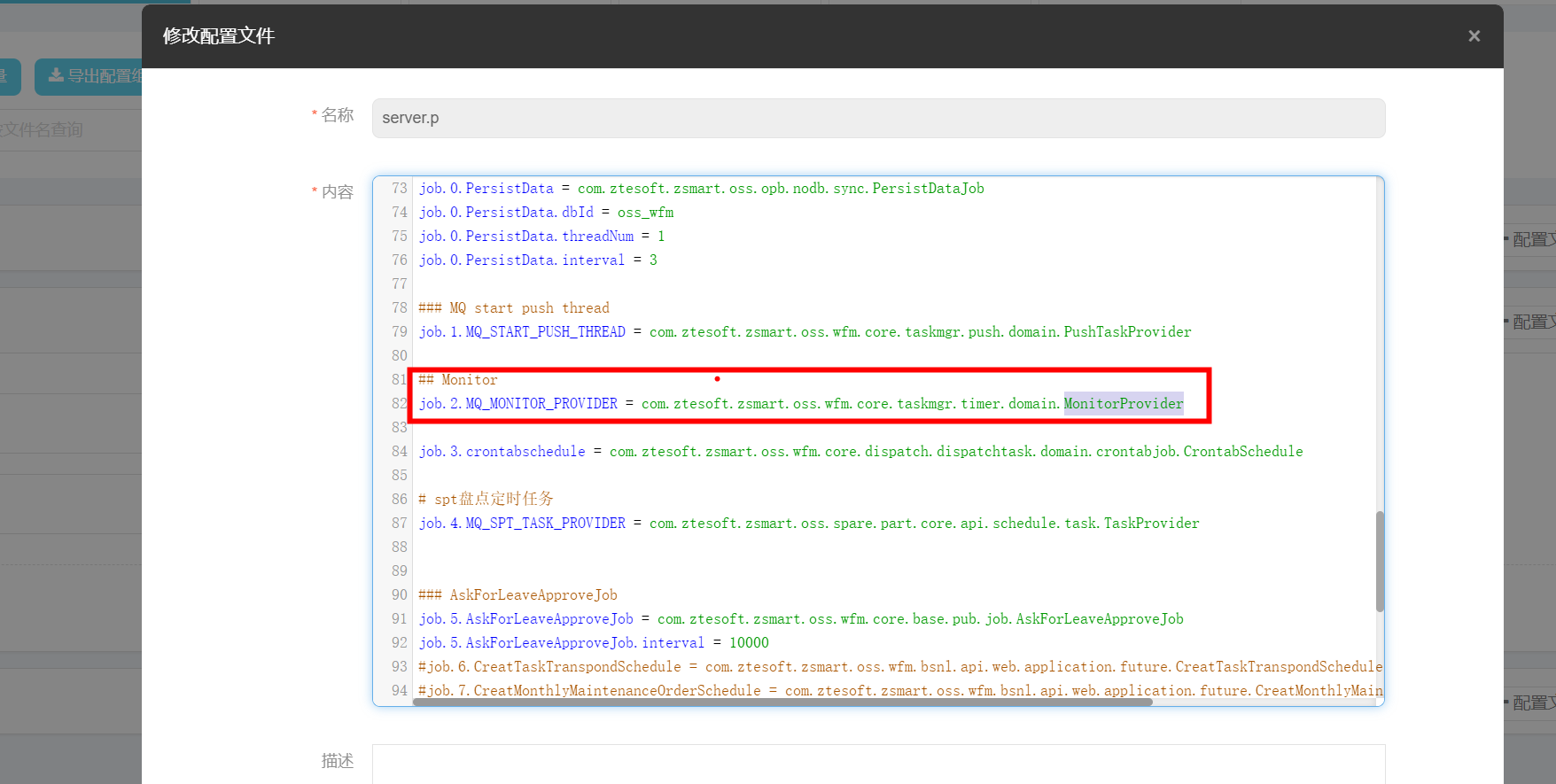


解决方式：猜测wfm最大连接数为30有点小，调整到了100

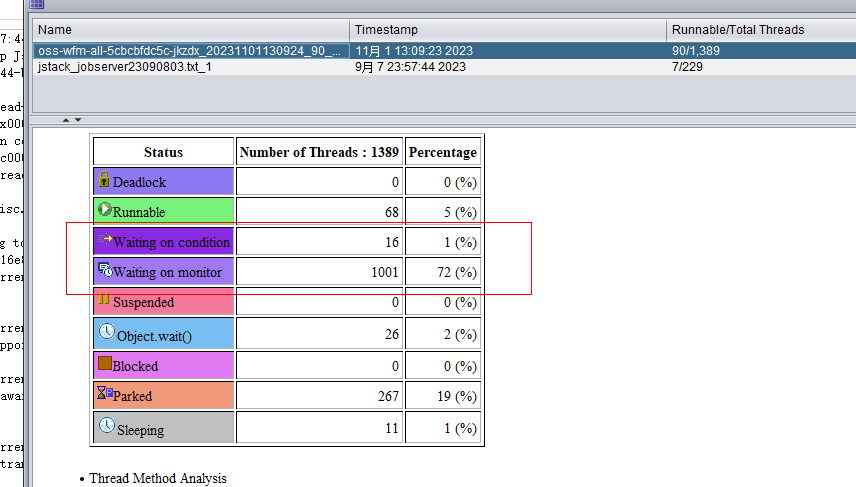
#### 2.数据库有大量锁表，同时控制台打印大量job异常

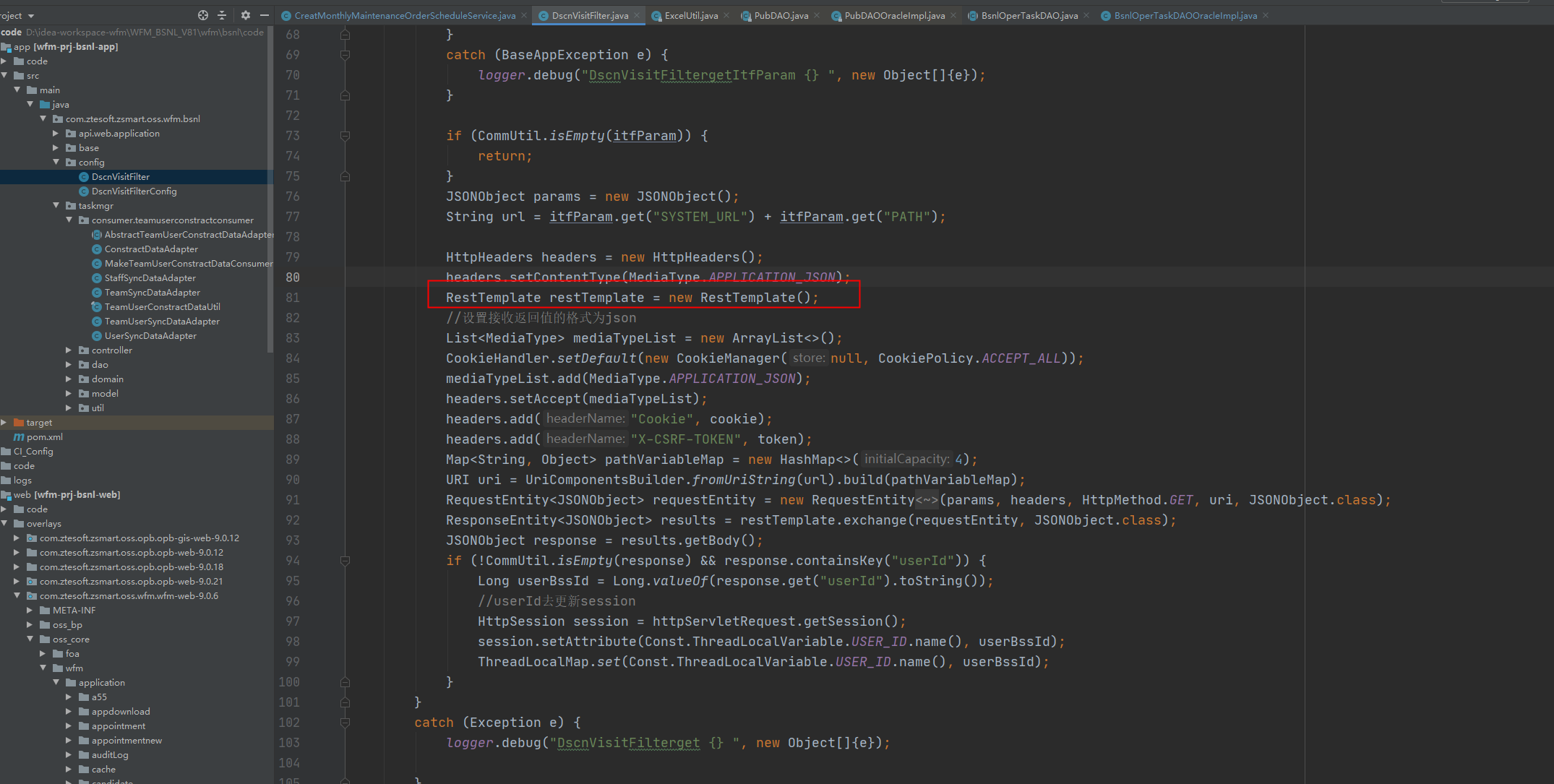


解决方式：因为锁的表都是his表，就先把定时归档的入口先注掉



#### 3.分析dump日志发现有大量线程等待在filter的代码中





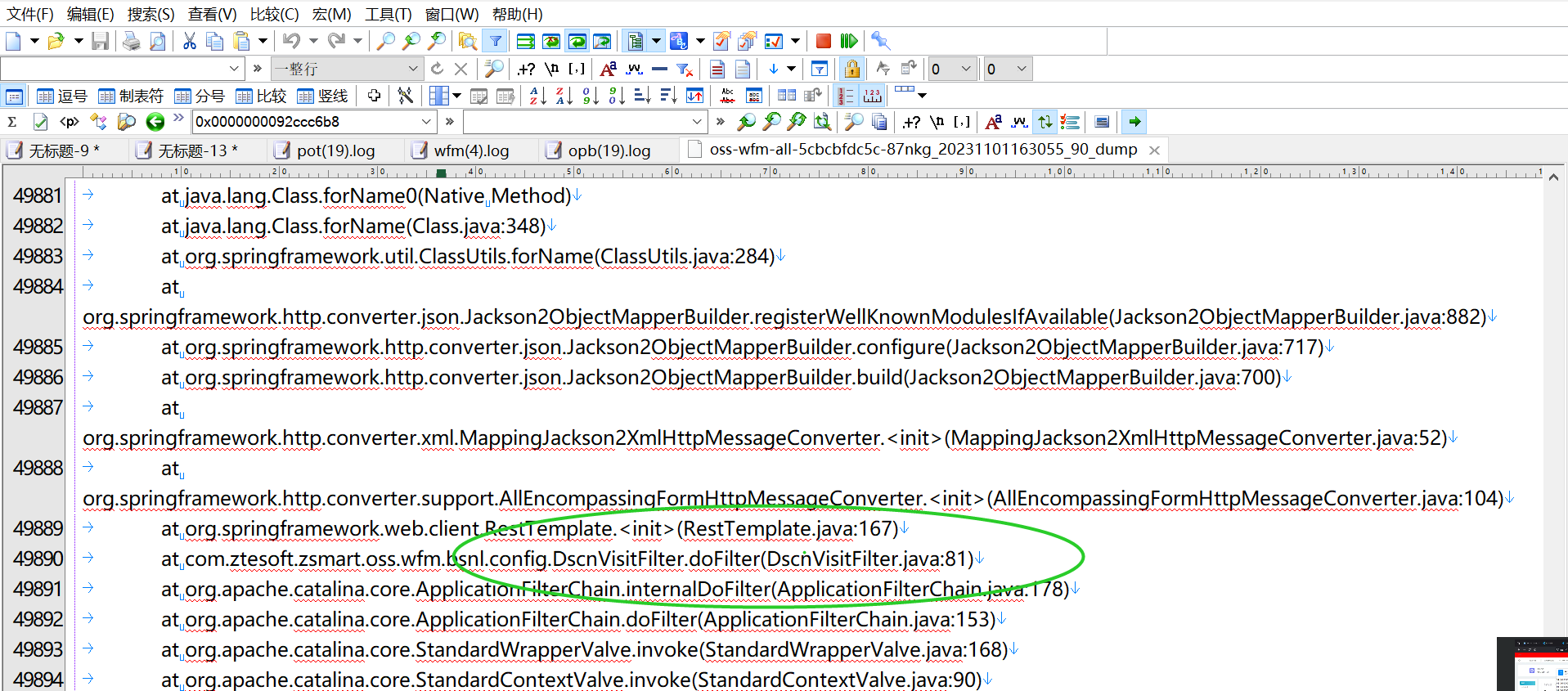
解决方式：

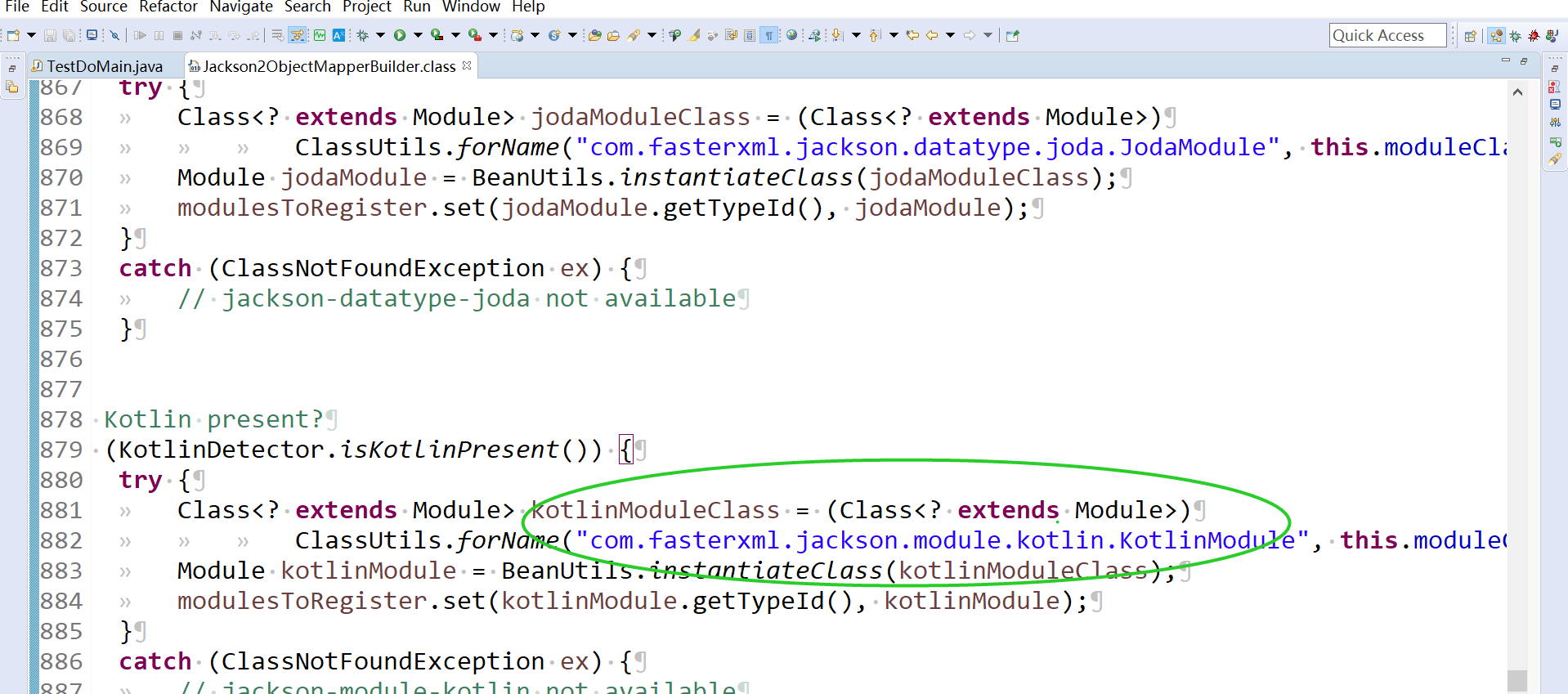
1. 了解到这个是之前做DSCM系统SSO功能的过滤器，现在没用，把过滤器代码删掉。但出现新问题，dscm访问wfm页面取不到session中的userId—不能解决问题

2. 判断由于B侧信息没有给，导致都锁死在这里调用，由于配置的获取session地址是外部网关地址，考虑到网络抖动这些不确定因素影响，换成内部网关地址—不能解决问题

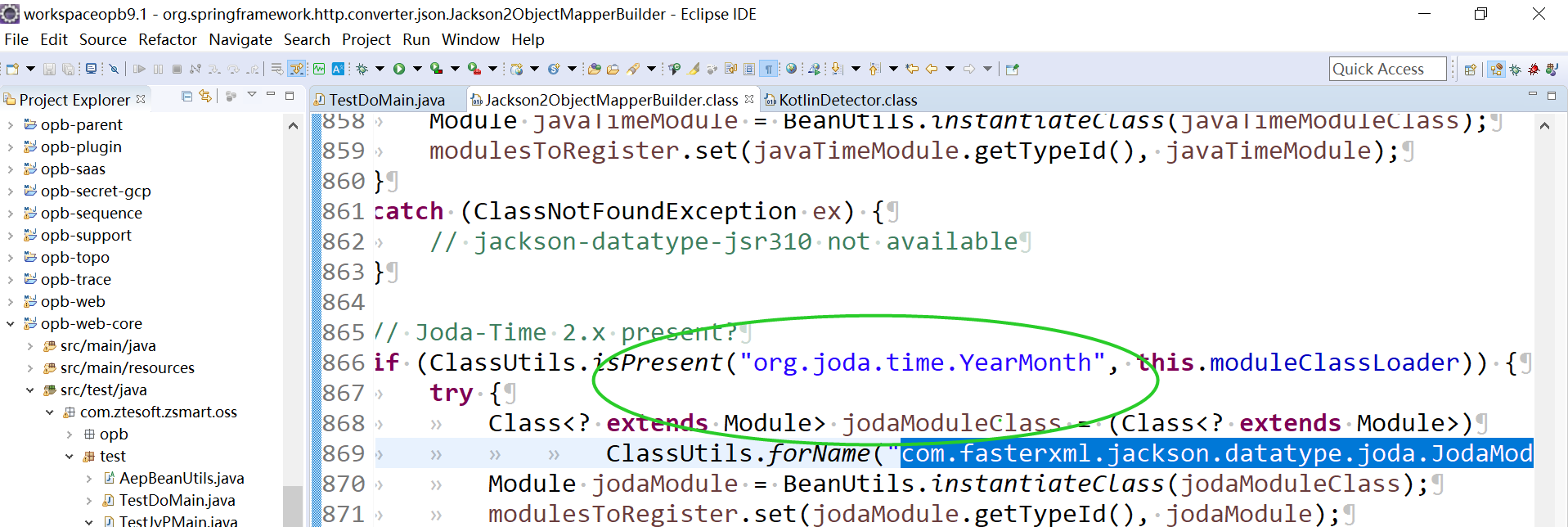
3. 猜测fastjson的版本有问题，将版本换到2.0.26版本 – 不能解决问题

4. 看到还是有字符转换的错误，卡在字符串处理





好像一直在加载这个不存在的module



被识别为KotlinModulede请求进来，由于没有这个依赖，导致所有请求都在new这个类。响应下降，最后彻底卡死了

需要补一个koli的jackson的jar

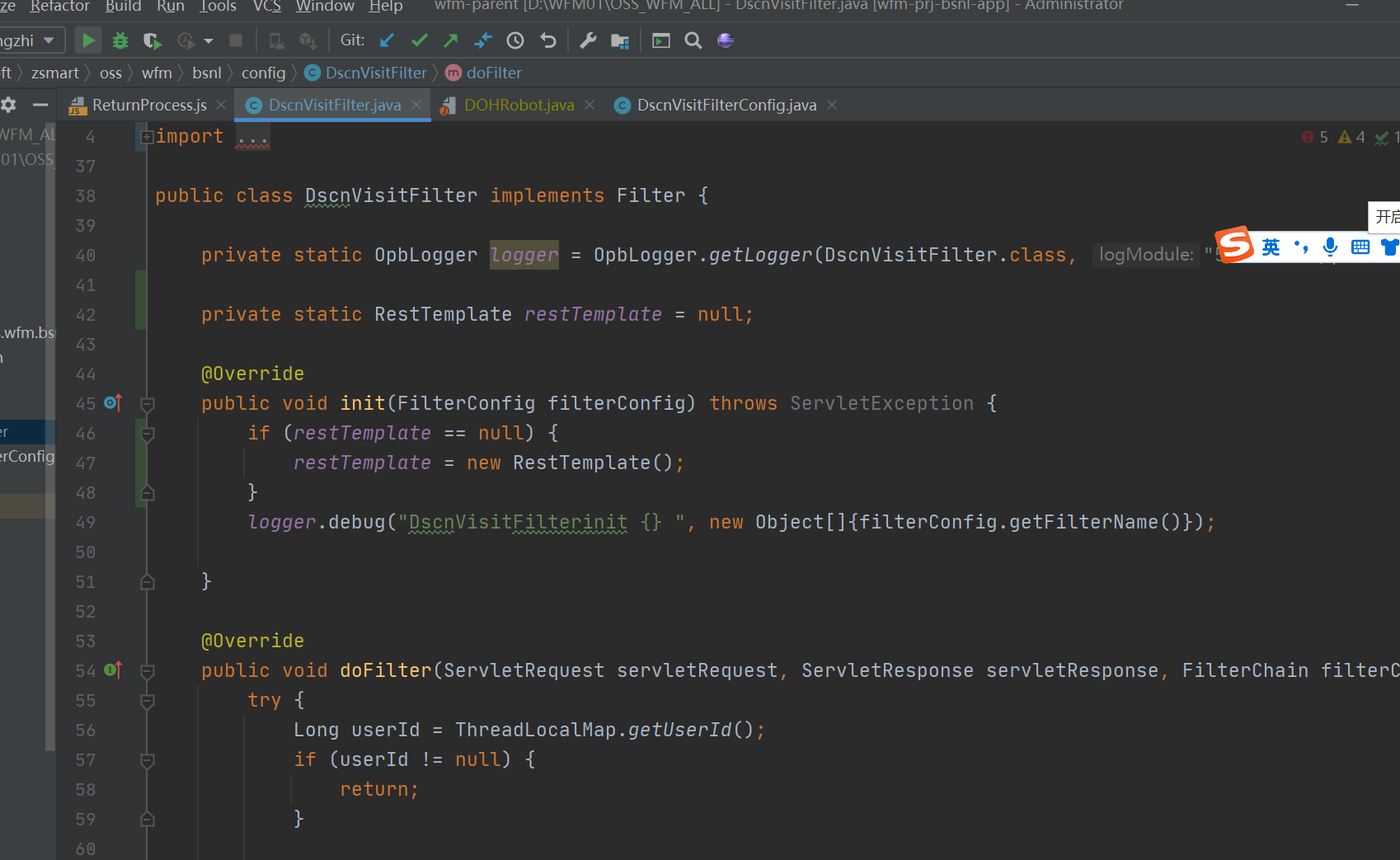
/root/ZSMART\_HOME/upload/upload/BSDK/install/OSS\_WFM  
-rw-r--r-- 1 zpaas zpaas 187460 Nov 1 15:46 fastjson-2.0.38.jar  
-rw-r--r-- 1 zpaas zpaas 1774097 Nov 1 15:46 fastjson2-2.0.38.jar  
-rw-r--r-- 1 zpaas zpaas 58264 Nov 1 15:46 fastjson2-extension-2.0.38.jar  
-rw-r--r-- 1 zpaas zpaas 154150 Nov 1 19:25 jackson-module-kotlin-2.15.2.jar  
-rw-r--r-- 1 zpaas zpaas 3058829 Nov 1 19:25 kotlin-reflect-1.6.21.jar

最终判断为

fastjson更新异常，导致wfm执行异常，堆积了非常大的调用，后续恢复后，业务调用大增，wfm的拦截器resttmplate是new的，会每次都去new jasckson的module类，导致性能降低。

处理方式：

改为getTeplame模式，init做下new，然后get时判下空



至此，问题1解决完毕，不再有死锁问题

## 问题解决方案

当前现场环境虽然通过代码避免了死锁，为了防止过滤器存在其他问题，Wfm不采用自己的过滤器，从opb的session信息中取DSCM过来的用户登录信息

## 后续避免措施

1. 过滤器这种代码得慎重，如果opb能获取到的数据，尽量从opb获取，减少新的公共代码产生的风险
2. 关注jar包升级信息，了解影响（如fastjson包）
3. 关注其他模块儿出现过的问题，避免踩一个坑（如fastjson包）
4. 代码编写规范问题，需要考虑到业务场景（如可以放在初始化的地方，不要反复去new）

# 问题2：容器内存溢出问题

## 问题表现

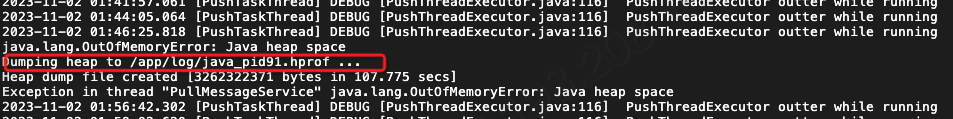
Wfm前端访问页面加载超时

## 问题出现时间

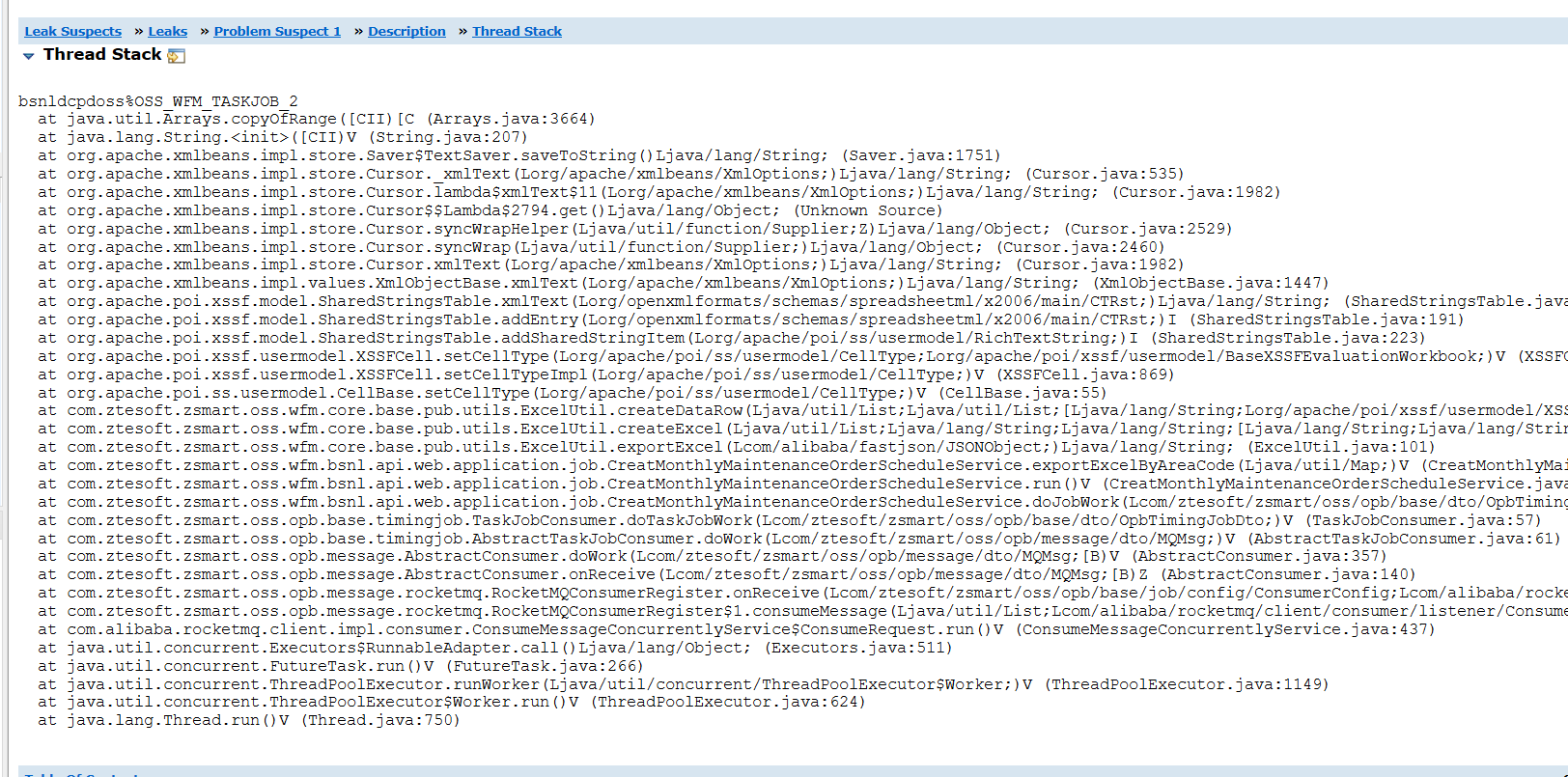
2023-11-02 08:30 — 2023-11-02 11:30

## 问题分析过程

#### 查看控制台日志，发现内存溢出



下载内存dump日志



定位到生成excel文件出现的问题

查询数据库，发现有近10万条数据，通过poi工具写入excel，就会报超时

## 问题解决方案

采取分页，每两万条数据生成一个excel

## 后续避免措施

1. 做大数据量的导入导出时需要考虑性能问题
2. 需要了解常用工具的性能问题，选择合适的工具很重要

# 总结

加强代码编写规范

掌握对dump文件的分析，学习使用性能分析工具