# 1 工作原理

定时调用

数据库

异常保存

故障监测主控程序

网页展现

提取数据

通用后台服务

# 2 程序说明

## 2.1程序包结构

程序代码位置com.af.faultcheck下，

根目录下为主控程序和检查器接口，

检查器实现类位置在com.af.faultcheck.checker包下

## 2.2 MonitorCheckJob 主控程序

定时调用配置的多个检查器接口，只负责调用，不负责具体业务处理

## 2.3 IChecker 检查器接口

检查器接口必须实现此接口,检查函数Check()

每个检查器接口需自己放入部署地点，异常名称，异常数据，时间，然后保存进数据库。

## 2.4 SqlChecker 检查器例子

sqlChecker,从自己的配置位置读取sql,执行后，如果有异常数据插入数据库

属性： filePath 文件路径

exceptionName 异常名称 例如 用户编号重复

# 3 部署配置

1. 故障监测采用单独的spring配置文件，配置文件名称

applicationContext\_faultcheck.xml

2) web.xml中添加 spring多配置文件

在<display-name>faultcheck</display-name>下添加

<context-param>

<param-name>contextConfigLocation</param-name>

<param-value>

/WEB-INF/applicationContext\*.xml

</param-value>

</context-param>

3)

4) 配置文件 fault.hbm.xml

Hibernate.cfg.xml中添加

<!--故障记录表-->

<mapping resource=*"系统/faults.hbm.xml"*/>

5)sql文件全部放在sqls文件夹下，

需手动拷贝到tomcat/root下

# 4 数据库表

t\_faults

id采用uuid,避免后期数据统一上传云服务器时候id冲突。待定

# 5 遗留问题

1. 地点名称没地方映射
2. Sqls目录下文件自动化部署
3. 需给检查器注入 hibernate sessionFactory,因包名问题（现在新编写程序采用com.af目录）与现有com.aote.rs不一致，因事务配置

<aop:pointcut id="allTestServiceMethod" expression="execution(\* com.aote.rs.\*.\*(..))"/>

无法开启事务，需改为

<aop:pointcut id="allTestServiceMethod" expression="execution(\* com.af.\*.\*.\*(..))"/>

是否对现有有影响，如果无影响，修改产品中现有配置