实习0.5个月复盘

2019-01-25

0.5个月，2个星期，共经手2个工单。

第1个工单

第1个工单内容较简单，需求为原先某字段为空时不允许记录重复写入，变更为允许。解决方式也简单，判断字段为空时不做重复校验，一律通过就行。刚开始主要还是熟悉开发流程。

1. 确认需求，OA出工单；
2. 新建分支，进行开发；
3. 自测；
4. 代码评审；
5. 提测；
6. 收到测试报告，申请上线；
7. 上线完成，代码合并基线。

事实上沟通有一定成本，其他较复杂的需求可能沟通成本会更高。

第2个工单

第2个工单需求为修复平台漏洞，1个SQL注入的漏洞，1个反射型XSS攻击的漏洞。解决思路也是常规。

代码评审后改正的第1个问题，也是带给我的第1个教训是，程序里if-else不可以嵌套过多。修复SQL注入漏洞需要对多个参数做校验，我一开始的做法是：

```
if (param1) {

} else if (param2) {

} else {

}
```

这样的做法太糟糕了！而且明显是规范里不推荐的写法。经导师提醒，将代码改为扩展性更好的这种写法：

```
Map<String, String> paramMap = new ConcurrentHashMap<>();
map.put(param1);
map.put(param2);
if (map) {}
```

乍一看没有问题，然而这段代码有bug潜伏，第2个给我带来教训的正是这里。
ConcurrentHashMap不允许传入null！和HashMap相比，有非常大的NPE风险，因为这里传入的参数来自页面（或者参数传入时就应该做非空校验？）。生产环境的代码不允许差错。

第3个教训关于配置文件。程序的配置文件，应该达到的效果是，没有这几行配置，程序就像没上线的版本一样，所更改添加的功能都依赖这个配置文件。当然同时也就要求，程序在没有添加配置时做出相应的默认处理。我自然是没能提前考虑到，配置不存在，直接NPE。

第4个教训是，画蛇添足。程序前一天晚上上线，第二天早上8点接到异常报告，上班之后马上回滚。产生异常的原因是，我改了需求之外部分的代码。上下两个一模一样的方法，看到上面的方法加上参数校验了，虽然需求里面没写，也索性给下面那个方法加上参数校验吧。然后NPE。因为在需求之外，自测没测出来，代码评审没看出来，提测没测出来，上线就出来了。

程序回滚的损失大概就是，线上系统一晚上的异常，涉及到的各部分人员再走一次工单流程，去

修复bug。就两行代码。

总结

1. 要对代码负责，从需求到上线。
2. 积极处理其他事情，反应要快，沟通要顺畅，注意力要集中。