# **个 人 简 历**

**基本信息**

姓 名：张创 性 别：男

电 话：185-1090-4561 出生年月：1990.06

邮 箱：zhangchuang4561@126.com 住 址：北京昌平霍家营

**教育背景**

毕业院校 专业 学历

郑州轻工业学院 机电一体化技术 专科

**技能特长**

1.熟练Cobbler自动化安装系统

2.熟练Keepalived+Nginx七层负载均衡，对Keepalive+LVS也有一定了解

3.熟练Shell编程，熟练使用Shell以及文本处理工具grep，sed，awk等对

日志进行处理、分析

4.熟悉Web服务器Nginx，Tomcat的安全及性能优化

5.熟悉Mysql的SQL语句，对日常应用进行调优

6.熟悉Mysql的主从机制以及优化

7.熟练Mysql的备份与恢复，定期进行灾备演练

8.熟悉Sersync+Rsync的备份方案

9.熟练使用Zabbix、nagios、open-falcon对负载均衡，Web服务，数据库等服务器以及服务的监控

10.熟悉使用ELK对服务日志的收集s

11.熟练使用SlatStack、puppet对服务器的批量管理

12.熟练Jenkisn+Git+Maven和Jenkins+Git+gradle自动化部署上线等

13.熟练OpenVPN、ipsec、pptp、L2tp的搭建和日常使用

**工作经历**

2016.3-至今 北京钻联科技有限公司 Linux运维工程师

工作职责：

1.完成项目从SVN到Git的迁移

2.完成Maven第三方依赖包的管理

3.实现Jenkins自动化构建

4.使用Zabbix对服务器以及服务的监控

5.使用ELK实现对日志的收集

6.负责运维文档的编写，如环境配置Slat、puppet文档

7.KVM、xen虚拟化技术环境搭建，维护

2014.7--2016.2 郑州珑玲科技有限公司 Linux运维工程师

工作职责：

1.负责服务搭建及产品迭代

2.负责服务器及网络设备的管理维护

3.负责客户机，办公设备的日常运作

**项目经验**

**一、全网服务器监控解决方案实施**

需求：公司刚成立不久，只依赖于云平台大的监控系统，每次故障，无法报

警，故障对业务的影响很大，因此，依据现有的监控技术，研究监控方

案，撰写监控文档，以改善服务器报警不及时，最大限度保证网站故障

及时处理

职责：

1.根据需求对监控软件zabbix进行研究

2.对系统践行监控

3.根据不同服务器的具体业务定制监控模板进行监控，实现实时报警

4.对于重要的服务，手写脚本收集数据进行监控

结果：最大化保证系统的监控，实现了服务运行的监控，以及网络监控

**二、完成代码从SVN到Git的迁移**

需求：由于公司开发人员的增多，SVN的分支已经不适合团队协作，所以

研究当下最流行的版本管理工具Git进行代码管理，Git的分支管理

适合多人协作，因此需要深入研究Git

职责：

1.研究Git日常使用，保证代码管理的准确性

2.研究Git分支，确定分支结构，保证测试环境与正式环境代码的分离

3.保证团队协作代码互不影响

结果：给大家分享完Git使用，并在工作之余辅助大家完成从SVN到Git

迁移，现在公司的代码管理已经从SVN彻底迁移到了Git

**三、研究Maven实现Maven的命令打包**

需求：由于公司的Maven项目只能使用Eclipse打包，因此需要研究Maven

并使用Maven私服实现第三方依赖的Jar包发布，保证Maven的pom

文件的规范性，实现Maven命令打包，保证自动化构建的继续

职责：

1.研究Maven以及Nexus的使用，实现Maven的pom文件的规范

2.规范Maven，实现命令行打包项目

3.研究将第三方Jar包发布到私服，并使用第三方Jar包

结果：完成了Maven使用Nexus管理第三方Jar包，同时实现了Mave命令

打包，为自动化构建做了总要准备

**四、自动化部署上线方案的设计与实施**

需求：由于公司业务性质，开发发版很频繁，频繁的上测试环境不仅开发很

麻烦，对运维的压力也很大，因此需要一个自动化部署的方案，因此

提出了使用Jenkins自动化工具实现服务的自动化的部署，实现了

Jenkins+Git+Maven的自动化部署架构

职责：

1.设计公司上线流程，确定各部门职责，保证代码代码上线的正确性

2.根据上线流程，深入了解Jenkins权限管理，并设计了Jenkins的权

限职责，实现各个角色权限的明确性

3.由于已经从SVN迁移到了Git，而且Maven已经可以使用命令行打包

从了使用Jenkins实现了打线上包与测试包的分离，并使用Git的钩

子实现了自动打包，并且使用Shell脚本实现测试、正式包的分离

结果：实现了使用Jenkins自动化构建，简化了运维流程，减少人为失误

**六、服务日志的收集与展示**

需求：开发经常需要看日志，由运维找日志给开发又很占用时间，因此需要

做日志的收集、统计与展示的平台

职责：

1.根据ELK官网文档，确定ELK的架构，降低ELK各个模块的耦合性

2.根据各个模块的功能编写相应的配置文件

3.将开发需要的日志分门别类的进行展示

结果：开发、测试人员都通过ELK平台查看服务器日志，减少了运维工作量

**五、进行对MySQL数据库的优化提出方案并实施**

需求：在工作中频繁出现数据库CPU负载高，主从复制有延迟，查询过慢，

而且测试环境主从不同步时有发生

职责：

1.硬件层面更新了服务器的配置，对CPU核数及内存进行调整

2.系统层面删除服务器上未使用的安装包和守护进程

3.数据库层面主要针对innodb引擎进行优化调整my.cnf参数

4.将slow.log进行分析，之后请求开发协助优化SQL

结果：除了做活动的高峰期，数据库服务器的负载很平稳，主从同步延迟非

常小，库表的设计进行规范，如字符集、表大小等，及sql语句的优化，

设立慢查询日志分析，定期分析慢查询等

**七、全网服务器数据批量分发与批量管理**

需求：公司服务器逐渐增多，已经不适合手动管理，于是着手于批量管理，

对Ansible和SlatStack进行对比，确定管理工具，书写管理文件对全

网服务器数据分发与管理

职责：

1.确定服务器批量管理文件

2.根据现有业务，确定sls配置文件的结构，编写sls配置文件，

并要求降低sls文件的耦合性，实现管理清晰化，适用性高

3.对于分发机进行安全配置，例如：去掉外部IP，开启防火墙等

4.并在独一的低配置的服务器搭建OpenVPN，提升远程控制连接性

结果：现在实现了SlatStack初始化系统，保证系统的一致性，而相同角色、

相同功能的服务器，采用salt管理，保证了从用户到安装包版本，从

服务版本，到服务路径，再到服务启动的一致性

**八、全网服务器数据备份**

需求：每次服务器的上线并不能完全保证服务的运行的正确性，因此每次上

线都需要对原有的服务进行备份，而且公司数据必须保证数据的安全性，

一些图片也需要定期备份，并确定人工修改数据库的流程，书写文档，

明确职责，尽量避免人为误操作

职责：

1.研究rsync的C/S模式。确定备份架构的清晰性

2.研究sersync监控文件的服务，保证文件添加修改的实时备份

3.书写数据库全量备份脚本，监控binlog文件，保证binlog的实时备份

4.书写数据库事故处理方案，以便出现故障可以按完整的步骤解决问题

防止遗漏步骤

5.制定人工更新数据的流程及制度。明确权限

结果：各个服务器的数据进行了定期备份，保证了数据安全，人工更新数据

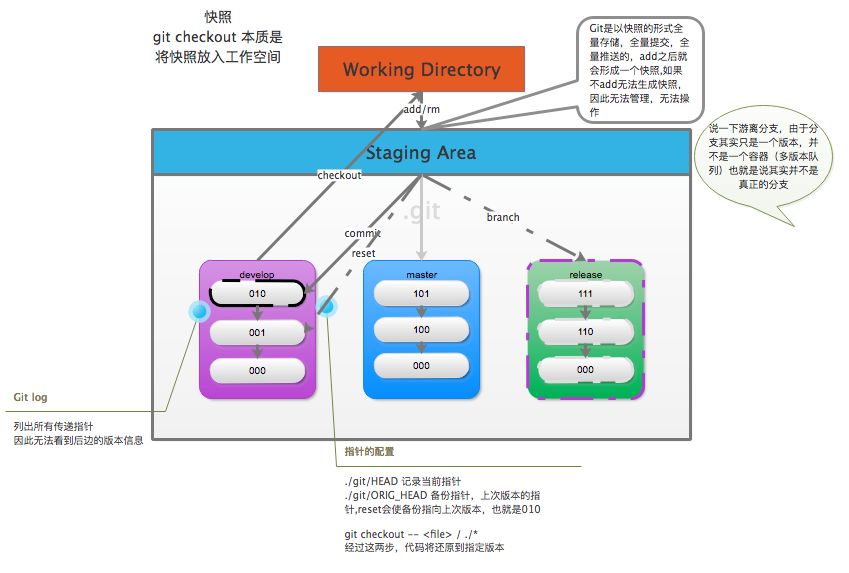
虽然流程麻烦了，但是减少了数据修改的误操作

**自我评价**

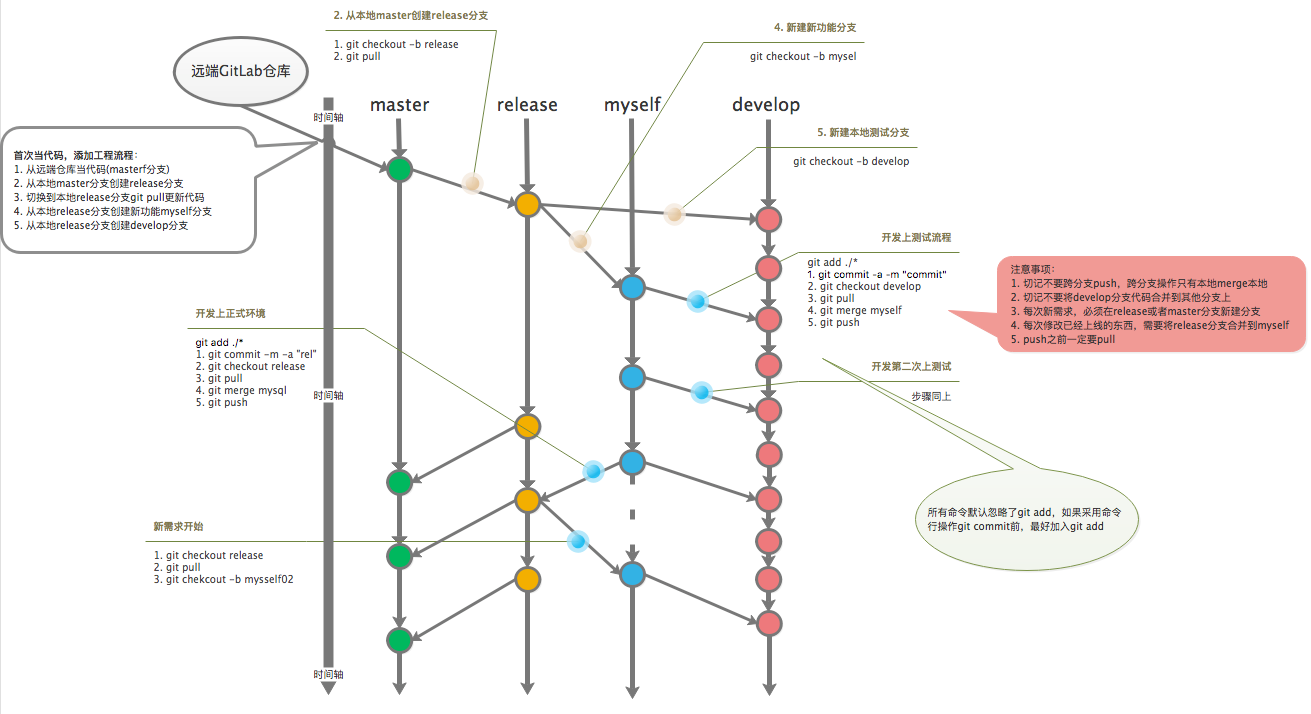
3年多的工作经验，即做过开发，也做过运维，让我更明白网站工作的流程，富有创新和开拓意识，善于合作、有较强的协调能力和团队精神、学习能力、自我管理能力强

**附录：**

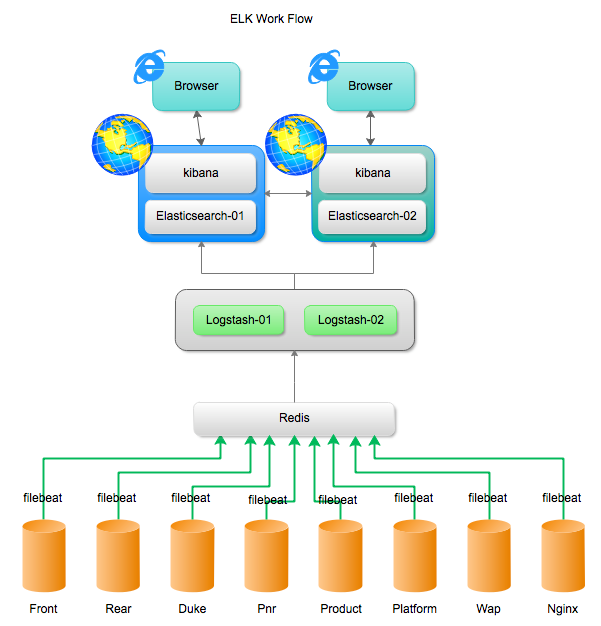
**附一：Git原理图**



**附二：Git分支架构图**



**附三：ELK架构图**



附四：脚本展示

# wap配置文件修改

PROJECT\_WAP\_CONIF\_SRC\_TEST="/data/jenkins/resources/zzr-wap/test"

PROJECT\_WAP\_CONIF\_DES\_TEST="/data/jenkins/workspace/zzr\_test-wap"

fun\_zzr\_test\_wap\_config() {

# 查找所有需要替换的文件

arrConfFileList=$(find ${PROJECT\_WAP\_CONIF\_SRC\_TEST} -type f)

for confFile in ${arrConfFileList[@]}; do

# 获取查找文件的文件路径，文件名

confFileName=$(basename ${confFile})

confFileDir=$(dirname ${confFile})

# 确定管理文件的存在性

retval=$(find ${PROJECT\_WAP\_CONIF\_DES\_TEST}/src -type f -name ${confFileName} |wc -l)

if [ 1 -ne ${retval} ]; then

echo "${confFile} is not only one, please manult replace"

errorFileName=$(${errorFileNum} + 1)

continue

fi

# 替换查找到的文件

project\_src\_file=$(find ${PROJECT\_WAP\_CONIF\_DES\_TEST}/src -type f -name ${confFileName})

if [ "$(basename ${confFileDir})" == $(basename $(dirname ${project\_src\_file})) ]; then

/usr/bin/cp ${confFile} ${project\_src\_file}

else

echo "${confFileName} in $(basename $(dirname ${project\_src\_file})) is not exited"

errorFileNum=$(${errorFileNum} + 1)

continue

fi

if [ 0 -ne ${errorFileNum} ]; then

exit 2

fi

done

}