详细设计报告

网管子系统

版本：v0.6

生效日期：

编写：

批准：

# 文档介绍

## 文档目的与背景

此文档为网管子系统详细设计报告，主要为网管子系统的开发、测试人员提供开发及测试的依据，用于规范和指导系统的开发、测试工作。

## 文档范围

本文档阐述了网管子系统的详细设计的流程算法和处理步骤等等，对系统外部、内部接口，数据库逻辑及各个子模块结构功能作了具体的描述。同时说明了开发、运行、测试环境。

本文档面向的主要对象是项目主管，开发人员和测试人员。

## 参考文献

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 编号 | 资料名称 | 作者 | 日期 | 出版单位 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

## 术语与缩写解释

|  |  |
| --- | --- |
| 缩写、术语 | 解 释 |
| SNMP | Simple Net Management Protocol 简单网络管理协议 |
|  |  |

# 程序系统的结构

### 可独立运行程序部署

本系统有两套可独立运行的系统，一套是管理界面系统（UI），使用tomcat容器运行，一套是后台网管站系统（NMS），使用java虚拟机直接运行。

管理界面系统发布的是war包，将其部署与tomcat的webapps目录下；

网管站系统是一个tar包，直接解压即可生产程序的目录 nms。

### 源代码部署

#### 管理界面系统源代码部署

|  |  |
| --- | --- |
| 目录 | 说明 |
|  |  |
|  |  |
|  |  |

#### 网管站系统源代码部署

|  |  |
| --- | --- |
| 目录 | 说明 |
| .mals | 程序最顶层包 |
| .mals.db | 连接数据库所需要的程序包 |
| .mals.sql | sql语句包，包含所有ibatis的xml文件 |
| .mals.bus | 系统业务代码包 |
| .mals.bus.trap | 系统告警代码 |
| .mals.bus.cap | 系统数据采集代码 |

## 子系统进程/线程结构

无

## 子系统数据流图

无

## 子系统通讯接口

1. 远程telnet,执行远程主机的shell脚本。

# 用户权限配置管理模块（UI）设计说明

## 功能

此模块是系统权限的核心控制模块，通过为用户分配角色的方式为其赋予权限，用户只能拥有其权限控制范围内的操作，只能看到其权限分配的功能选项。系统默认一个超级管理员，具有权限分配，角色管理及用户管理的权限。其他用户统一由此管理员进行管理。

权限信息统一存放在数据库中的mals\_privilege\_t表中，网管用户不具备对权限信息的管理功能，网管开发人员具有对权限信息的管理功能。

权限管理由开发人员用特殊帐户和页面进入系统，对权限信息进行管理，此功能在网元变动是使用。

权限增加：增加新的权限信息；

权限更改：更改已存在的权限信息；

权限删除：删除已有的权限信息；

权限查询：查询所有权限信息；

角色管理由超级管理员控制，对角色信息进行管理，可以为角色分配多个权限。

角色增加：增加新的角色；

角色更改：更改已存在的角色信息；

角色删除：删除已有的角色；

角色查询：查询所有角色信息；

用户管理由超级管理管理员控制，对用户进行管理，可以为用户分配多个角色。

角色增加：增加新的角色；

角色更改：更改已存在的角色信息；

角色删除：删除已有的角色；

角色查询：查询所有角色信息；

## 性能

确保稳定性，正确性和操作速度。

## 输入项

## 输出项

## 算法

无

## 流程逻辑



图 3-1：业务时序图

## 模块的伪代码

## 接口

## 存储分配

无

## 注释设计

## 限制条件

用户操作限制在其所拥有的权限范围之内。

## 尚未解决的问题

无

# 系统配置管理模块（UI）设计说明

## 功能

配置网管的基本信息，包括告警联系人管理，告警规则管理，告警规则与联系人管理，上级网管管理，业务系统告警阀值管理，业务子代理配置管理等。通知网管站相关信息做了变动。

告警联系人管理，对告警管理人信息进行管理，与网管站进行信息交互

告警联系人添加：添加新的告警联系人信息，通知网管站；

告警联系人修改：修改原有的告警联系人信息，通知网管站；

告警联系人删除：删除原有的告警联系人信息，通知网管站；

告警联系人查询：查询告警联系人信息；

告警规则管理，对告警规则信息进行管理，与网管站进行信息交互

告警规则添加：添加新的告警规则，通知网管站；

告警规则修改：修改原有的告警规则，通知网管站；

告警规则删除：删除原有的告警规则，通知网管站；

告警规则查询：查询告警规则信息；

告警级别与联系人管理，对告警级别与联系人的关联信息进行管理，与网管站进行交互

告警级别与联系人添加：添加新的告警级别与联系人关联信息，通知网管站；

告警级别与联系人修改：修改原有的告警级别与联系人关联信息，通知网管站；

告警级别与联系人删除：删除原有的告警级别与联系人关联信息，通知网管站；

告警级别与联系人查询：查询告警级别与联系人关联信息，通知网管站；

上级网管管理，对上级网管的配置信息进行管理，与网管站进行交互

上级网管信息添加：添加新的上级网管信息，通知网管站；

上级网管信息修改：修改原有的上级网管信息，通知网管站；

上级网管信息删除：删除原有的上级网管信息，通知网管站；

上级网管信息查询：查询上级网管信息，通知网管站；

业务系统子代理管理，对业务系统子代理的配置信息进行管理，与网管站进行交互

业务系统子代理信息添加：添加新的业务系统子代理信息，通知网管站；

业务系统子代理信息修改：修改原有的业务系统子代理信息，通知网管站；

业务系统子代理信息删除：删除原有的业务系统子代理信息，通知网管站；

业务系统子代理信息查询：查询业务系统子代理信息，通知网管站；

## 性能

确保稳定性，正确性和操作速度。

## 输入项

## 输出项

## 算法

无

## 流程逻辑



图 4-1 操作时序图



图4-2：业务处理流程图

## 模块的伪代码

## 接口

## 存储分配

## 注释设计

## 限制条件

用户操作限制在其所拥有的权限范围之内。

## 尚未解决的问题

# 告警查询管理模块（UI）设计说明

## 功能

每个业务系统都对告警信息提供查询操作，可以按照时间、告警类型、告警级别等，进行对告警信息的联合条件查询。

* 业务系统告警信息包括：

1、程序异常信息，数据库异常信息，连接异常信息的告警；

2、当前业务量超过阀值告警、数据进来的流量超过阀值告警；

3、出去的流量超过阀值告警，实际的处理量超过阀值告警。

## 性能

确保稳定性，正确性和操作速度。

## 输入项

业务系统：告警类型包括数据库告警、业务系统告警、主机告警、告警级别包括严重告警、主要告警、次要告警、警告告警

## 输出项

## 算法

## 流程逻辑



图 5-1：业务流程图

## 模块的伪代码

## 接口

## 存储分配

无

## 注释设计

## 限制条件

用户操作限制在其所拥有的权限范围之内。

## 尚未解决的问题

# 历史查询管理模块（UI）设计说明

## 功能

本模块主要用于对网管站采集的历史数据进行查询。

* CPU历史数据查询：对指定主机的CPU数据进行查看。
* 系统内存历史数据查询：对指定主机的系统内存历史数据进行查看。
* 历史业务数据查询：对指定业务系统的历史业务数据进行查看。

业务数据包括：

1、程序异常信息，数据库异常信息，连接异常信息等的报表分析

2、当前业务量、数据进来的流量，出去的流量，实际的处理量等的报表分析。

* 数据库表空间历史数据查询：对指定主机的数据库表空间历史数据进行查看。

## 性能

无特殊要求。

## 输入项

* CPU历史数据查询： 主机名、时间段
* 系统内存历史数据查询： 主机名、时间段
* 历史业务数据查询：

业务系统：日志输出信息、告警信息、采集返回信息（主机、数据库、业务系统）。

* 数据库表空间历史数据查询：主机名、时间段

## 输出项

输出项为指定条件范围内的历史数据包括CPU历史数据、系统内存历史数据、业务系统历史数据、表空间历史数据等。

## 算法

无

## 流程逻辑



图6-1 历史查询管理模块时序图

流程说明：

* + 前台用户下发统计报表操作请求，Action控制器接收请求，然后通过分析配置文件，查找请求所对应的历史查询的用户Ation，将请求转发。
  + OperationAction进行业务逻辑验证，当业务逻辑验证错误时，则直接返回视图。通过验证后，记录操作日志，调用业务逻辑处理方法，分析前台传过来的查询条件，组装成查询语句，进行数据库操作（记录操作日志），将满足条件的数据提取出来，返回给用户Ation调用，将获得的数据库记录进行处理，保存在请求范围内.
  + 正常处理用户的请求之后，将历史信息通过浏览器显示给用户

## 模块的伪代码

**private** QueryHistory queryHistory;

**public** String queryHistoryExecute() {

// 进行业务逻辑验证

......

**if** (业务逻辑验证成立) {

// 修改前进行日志记录

**this**.addOperationLog(getText(beforeSelectQueryHistory));

List<QueryHistory> li;

**try** {

// 设置用户输入的查询条件

reportForms.setConditions(request.getParameter());

// 业务逻辑处理

li = (ArrayList<QueryHistory>)Service.getList(QueryHistory.**class**, queryHistory);

// 操作后进行日志记录

**this**.addOperationLog(getText(afterSelectQueryHistory));

// 处理数据库记录

for(QueryHistory queryHistory : li){

......

}

} **catch** (BusinessException e1) {

e1.printStackTrace();

}

......

// 返回浏览器视图

**return** SUCCESS;

}

}

## 接口

## 存储分配

无

## 注释设计

## 限制条件

无。

## 尚未解决的问题

无。

# 统计报表模块（UI）设计说明

## 功能

根据用户选择的报表类型和需要进行数据统计的时间范围或数据范围将对应的图表展示给用户。

* 业务总量统计报表：对指定时间段内的业务总量进行统计，生成柱状图。
* 业务量比例统计报表：对指定时间段内的业务量所占的比例进行统计，生成饼 状图。
* 告警数量统计报表：对指定时间段内的告警数量进行统计，生成柱状图。
* 告警比例统计报表：对指定时间段内的告警比例进行统计，生成饼状图。
* 告警级别统计报表：对指定时间段内的告警级别进行统计，生成柱状图。
* 告警级别比例统计报表：对指定时间段内的告警级别比例进行统计，生成饼状图。

## 性能

无特殊要求。

## 输入项

输入相应的业务系统、图表类型、产生图表数据的时间范围或数据范围。

## 输出项

输出相应的图表信息。

## 算法

无

## 流程逻辑



图7-1 统计报表模块时序图

流程说明：

1. 前台用户下发统计报表操作请求，Action控制器接收请求，然后通过分析配置文件，查找请求所对应的生成统计报表的用户Ation，将请求转发。
2. OperationAction进行业务逻辑验证，当业务逻辑验证错误时，则直接返回视图。通过验证后，记录操作日志，调用业务逻辑处理方法，分析前台传过来的查询条件，组装成查询语句，进行数据库操作（记录操作日志），将满足条件的数据提取出来，返回给用户Ation调用，将获得的数据库记录进行处理，并将其传递给JFreeChart组件生成相应的图表。
3. 正常处理用户的请求之后，将图表信息通过浏览器显示给用户。

## 模块的伪代码

**private** ReportForms reportForms;

**public** String reportFormsExecute() {

// 进行业务逻辑验证

......

**if** (业务逻辑验证成立) {

// 修改前进行日志记录

**this**.addOperationLog(getText(beforeSelectReportForms));

List<ReportForms> li;

**try** {

// 设置用户输入的查询条件

reportForms.setConditions(request.getParameter());

// 业务逻辑处理

li = (ArrayList<ReportForms>)Service.getList(ReportForms.**class**, reportForms);

// 操作后进行日志记录

**this**.addOperationLog(getText(afterSelectReportForms));

// 处理数据库记录

for(ReportForms re : li){

......

}

// 生成报表

JFreeChart.makeReports();

} **catch** (BusinessException e1) {

e1.printStackTrace();

}

......

// 返回浏览器视图

**return** SUCCESS;

}

}

## 接口

## 存储分配

无

## 注释设计

## 限制条件

无

## 尚未解决的问题

无

# 系统日志管理模块（UI）设计说明

## 功能

* 记录系统日志：对登录到系统的用户所进行的增加、删除、修改等影响系统的操作进行日志记录，记录的信息包括用户ID、IP地址、操作时间、操作描述等。
* 查看系统日志：查询某个用户在某段时间所进行的系统操作，查看的信息包括用户ID、IP地址、操作时间、操作描述等。

## 性能

无

## 输入项

查询系统日志：用户名、时间段

## 输出项

查询系统日志：输出满足条件的相关日志信息

## 算法

无

## 流程逻辑



图8-1 系统日志管理模块时序图

流程说明：

1. 前台用户下发操作请求，Action控制器接收请求，然后通过分析配置文件，查找请求所对应的用户Ation，将请求转发。
2. OperationAction进行业务逻辑验证，若是验证有误，则直接返回视图，不在进行业务处理。通过验证后，对所进行的操作日志进行记录，接下来执行正常的业务逻辑处理，在正常执行相关操作之后，若有必要，可以在一次进行记录日志操作，如修改密码操作，可以将修改前和修改后的信息分别作为日志内容进行记录，方便查看操作日志时的对比。
3. 正常处理用户的请求之后，将响应信息返回视图。

## 模块的伪代码

**public** String updatePwdExecute() {

//进行业务逻辑验证

......

**if** (业务逻辑验证成立) {

//修改前进行日志记录

**this**.addOperationLog(getText(beforeUpdateUserPassword));

ModifyUserPwd modifyUserPwd = **new** ModifyUserPwd();

**try** {

//业务逻辑处理

Service.update(ModifyUserPwd.**class**, modifyUserPwd);

//修改后进行日志记录

**this**.addOperationLog(getText(afterUpdateUserPassword));

} **catch** (BusinessException e1) {

e1.printStackTrace();

}

......

//返回浏览器视图

**return** SUCCESS;

}

}

## 接口

## 存储分配

无

## 注释设计

## 限制条件

无。

## 尚未解决的问题

无

# 采集管理模块（NM）设计说明

## 功能

采集管理模块主要负责采集得到各个数据源的产生的数据，然后对数据进行解析并且将数据保存在数据库中。需要实现两部分功能:

采集数据：向各个子代理发送SNMP请求，并获取数据,把数据存入到缓冲区中。

数据解析：把采集到在缓冲区中的数据进行整理，整理成可以直接入库的SQL语句。

## 性能

采集和解析的速度不能低于每秒50条（条：单位按照入库数据条数计算）。

## 输入项

* 待采集数据源相关配置：
  + 数据源的IP和snmp监听端口；
  + 数据源呆采集的OID信息；
  + 采集的时间间隔
* 数据源产生的数据：通过snmp协议，向数据源发送get请求，并获取到结果。

## 输出项

经过解析的采集到的数据源产生的数据。

补充说明：如果数据要保存到数据库中，那么应将数据解析为可以直接运行的sql语句。

## 算法

无

## 流程逻辑

整个采集的流程由统一网管框架来实现。

### 采集数据流程逻辑

采集数据时，网管框架会为每个需要采集的模块建立单独线程，并发进行采集。采集过程会调用根据各自业务实现的采集实现，下图为位置服务平台的采集实现类流程图：



### 数据解析流程逻辑

解析操作在全部的采集模块都采集结束之后进行，对缓冲区中的数据进行解析，生成可以直接入库的sql语句，程序流程如下图：



## 模块的伪代码

无

## 接口

## 存储分配

无

## 注释设计

无

## 限制条件

由于配置管理部分是依靠于外部数据来执行的，所以外部数据的必须保证正确和完整，否则配置管理部分将无法正常运行。

## 尚未解决的问题

无

# 告警管理模块（NM）设计说明

## 功能

主要功能：监听端口，解析发向这个端口的snmp Trap 包。并把Trap的内容存入到数据库中。

次要功能：发送邮件和短信功能，告警转发上级网管功能，告警过来规则

## 性能

## 输入项

* 从告警源发出的snmp告警信息
* 从管理界面配置的告警规则

## 输出项

* 向转发地址转出snmp告警
* 告警信息入库保存

## 算法

无

## 流程逻辑

图12-1为规则匹配流程图，在业务逻辑里需要实现多次规则匹配：级别设置规则。这些规则匹配都使用相同的流程逻辑：



图12-1

## 模块的伪代码

## 接口

## 存储分配

无

## 注释设计

无

## 限制条件

## 尚未解决的问题

无

# 日志设计说明

* 日志的使用

系统使用Log4j支持日志的输出，Log类通过LogFactory获得。所有调试用日志均需使用debug()方法，监控各个模块运行状况、重要操作的开始结束均使用info()方法，异常的错误日志均使用warn()方法，对于会影响系统正常运行的异常输出日志时使用error()方法。

* 日志格式

[日期]+[级别]+[类名]+[-]+[日志内容]

* 日志内容

操作开始：C + [ - ] + XXX 开始

操作结束：C + [ - ] + XXX 结束

关键参数：P + [ - ] + [参数中文解释] + [参数名] + [：] + [参数值]

错误日志：E + [ - ] + XXX 发生异常！ + [Cause：] + [异常产生原因]

调式日志：D + [ - ] + [调试日志内容]

* 日志级别

|  |  |
| --- | --- |
| 级别 | 说 明 |
| DEBUG | 代码调试时使用 |
| INFO | 跟踪系统运行状况时使用 |
| WARN | 系统运行发生异常时使用 |
| ERROR | 系统运行环境（如：虚拟机、数据库等）发生异常 |

* 日志输出要求：

系统的正常操作在开始及结束时要输出日志，如：采集管理模块采集数据的操作、处理告警的操作、上级网管交互模块刷新数据的操作、数据入库操作及动态更新操作等。

对系统运行的关键参数要输出日志。

扑获异常时要有对异常产生原因描述的日志输出，同时还要有异常信息的日志输出。

补充说明：操作的开始日志与结束日志必须成对出现，如果在开始和结束之间有发生异常的可能，那么在输出错误日志后要追加结束日志的输出。例如：

* 日志分析

通过INFO级别的日志能够查看出系统各个模块的各个主要功能的运行状态及系统运行时参数。

通过WARN及ERROR级别的日志能够查看出系统各个模块产生的异常，帮助定位异常产生原因。

# 错误处理说明

各模块扑获异常后均须将异常转换为系统指定的异常（见下表）并重新抛出且要有详细的异常信息描述，原则上各模块内部的子模块不准许消灭异常。异常的处理将统一由各模块的管理器负责。

补充说明：如果有特殊需要可以创建自己的异常，但需要继承于系统指定的异常。

# 注释设计

无

# 数据库物理设计

参见《数据库设计报告》