预警平台技术红皮书

NC-UAP 5.02

用友 NC-UAP

2009-09-10

目　　录

第一章 前言

现代企业要在激烈的市场竞争中生存并发展，就需要对自身的优势与缺陷都有一个清楚的了解，所谓知己知彼，企业要想在市场竞争中立于不败之地，就必须及时地发现自己的优势与缺陷，发挥自身的优势，弥补存在的缺陷，为企业的发展与壮大扫除障碍。

如何及时地发现自身的优势与缺陷，向来是企业界和理论界倾力研究的重点问题之一。于是 NC 的预警平台应求而生。NC 预警平台分成两种类型的预警。一为定时预警。即用户可以指定何时或者以何周期去执行某项任务，并依照设置的阈值决定是否产生预警提示,致使企业及时合理做出正确的决策。二为即时预警。顾名思义即时是立即发生，NC 中主要支持用户登录和打开节点两种，所作事情与定时预警一样。

NC 预警服务的整体示意图如图 1-1.

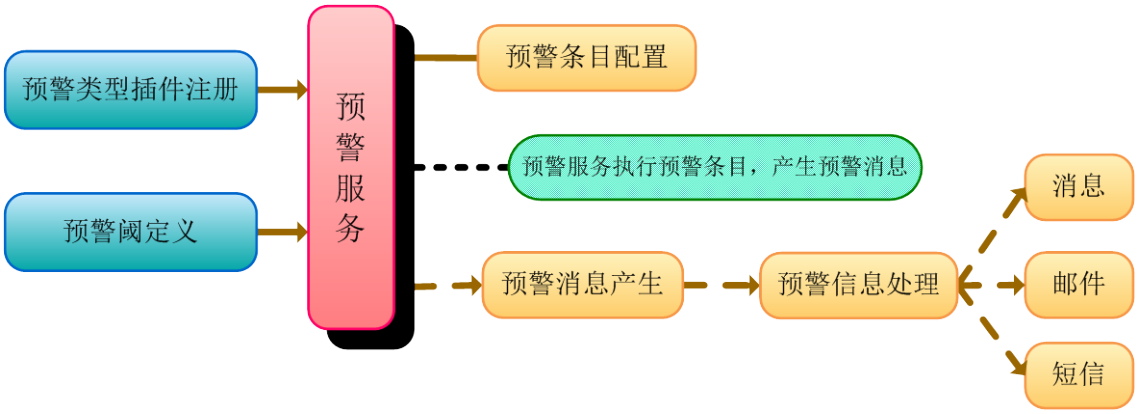

图 1-1 预警平台服务示意图

由上图可知，预警平台主要由预警类型、预警条目以及预警消息三个组成部分。于是想要使用预警平台的预警功能，需要做两步工作：1.预警类型注册；2.预警条目注册。分别在本文档第二章和第三章介绍。预警消息是在条目配置的时候配置的。

注意：从 V502 开始，预警平台功能并入任务中心。

第二章 预警类型注册

1. 功能简介

预警类型就是一种预警的一个插件类型（由开发人员开发，具体如何开发插件见 插件开发指南）。它目的是对某个业务或操作的抽象，其可以定义一系列阈值。(这里也只是定义，真正的值还是由条目来设置的)。

定义一个预警类型需要提供：名称、所属系统、业务插件、描述、阈值名称、编辑类型、参照名称(如果编辑类型为参照)。

2. 操作说明

- 打开节点:[客户化]->[预警平台]->[预警条件配置]

鼠标指向“类型配置”菜单，单击增加，可以增加一个预警类型。

选中一条已注册的预警类型，单击“删除”、“编辑”，可以对已有预警类型进行删除或修改的操作。双击一条预警类型，也可以对其进行编辑。

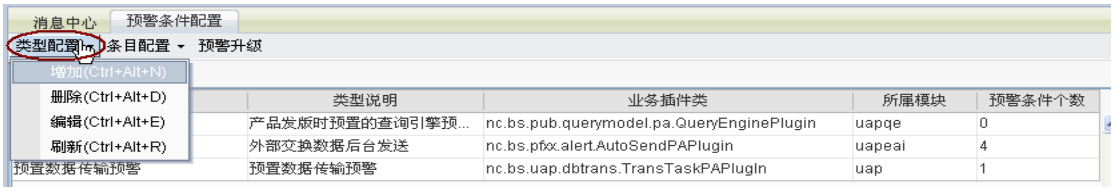

图 2-1 预警首界面

2.1 增加/编辑预警类型

增加/编辑一个预警类型，预警类型编辑界面如下图所示：

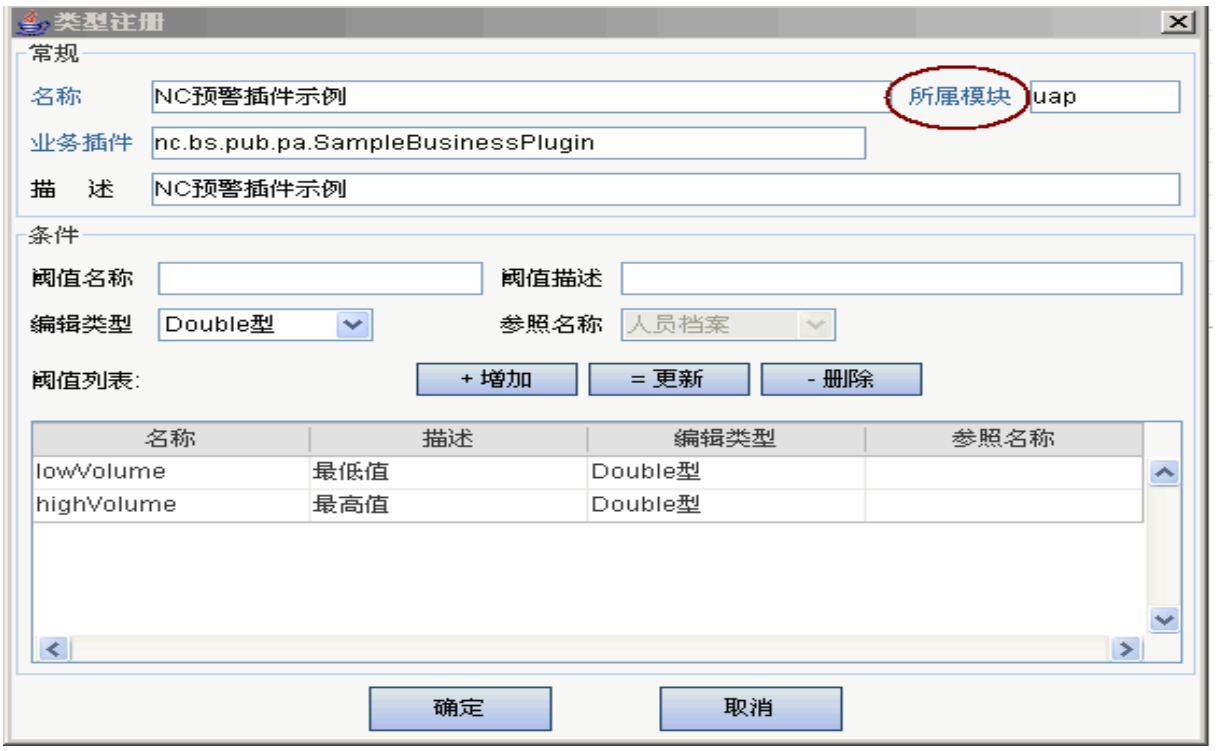

图 2-2 预警类型编辑界面

项目说明：

- ✓ 名称：输入预警类型的名称。必填项。
- ✓ 所属模块：是区分各产品组模块的标识，即通常说的模块名。这值对应的是中间件/modules 目录下的子目录。按规定模块名都是小写的。必填项。
- ✓ 业务插件：输入业务插件的类名，此项不能为空。此类中需实现预警平台接口。且可以放在业务模块中任何地方(即不限制Public/Client/Private).该类遵循开发规范见 插件开发指南。必填项。
- ✓ 描述：指对预警类型的文字描述。
- ✓ 阈值名称：也就是条件的名称。阈值含义：对阈值名称的说明。
- ✓ 编辑类型：此处定义输入阈值的样式，系统提供五个选择：字符型、逻辑型、整型、Double 型和参照基础档案类型。如果将某个阈值的编辑类型定义为逻辑型，那么在进行预警条目设置时，此阈值的值以下拉框的形式出现，有是和否两个选择；如果选择编辑类型为基础档案，还要在后面的参照名称栏中选择参照哪个基础档案，这样当用户输入该阈值时会弹出相应的基础档案参照。
- ✓ 参照名称：如果编辑类型选择为参照基础档案，那么参数名称变为可选项，提供选择的项有人员档案、部门档案、客户档案等等。

2.2 关于多语的说明

这里指对 NC 预置的插件支持多语，对于后来增加的没有这个支持。想多了解多语的这一特性，参见高级操作。

第三章 预警条目注册

1. 功能简介

预警条目一般由实施人员或用户根据具体的业务环境和需要来定义。通过选择预警类型，并设置该预警类型中需要用户设置的阈值变量，以及定义预警方式来实现具体的预警任务。预警方式包括产生方式和发送方式。产生方式控制预警的发生时间，有即时和定时两种。发送方式是定义预警以何种方式发送给用户。

预警条目是具体的预警任务，是预警平台调度执行的单位。一个预警类型可以根据不同的业务情况定义多个预警条目。

预警条目保存预警信息的产生条件及发送方式设置，基于预警类型进行编辑。预警平台的后台服务线程定时读取预警条目信息，根据条目中设置的条件，调用相应的预警类型与业务信息相比较，当符合预警条件时，就会产生预警信息，并根据相应配置进行信息发送。

条目与类型的关系其实就是一个具体与抽象的关系。如用友公司与泛化的公司一样。公司具有名称、地址等属性，用友公司与之对应的就是用友、北京上地等。

2. 操作说明

鼠标指向“条目配置”菜单，单击增加，可以增加一个预警条目。

选中一条已注册的预警条目，单击“删除”、“编辑”，可以对已有预警条目进行删除或修改的操作。双击一条预警条目，也可以对其进行编辑。与类型配置的菜单差别就是其可以复制，即把其它公司的条目复制到本公司。

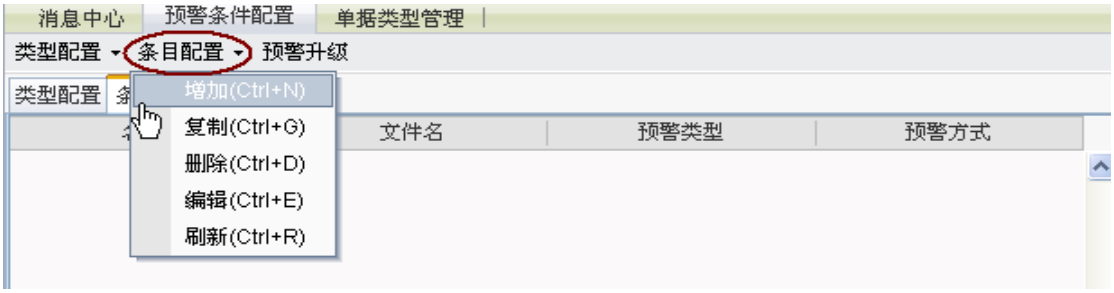

图 3-1 条目配置首界面

2.1 增加/编辑预警条目

增加/编辑一个预警条目，预警条目编辑界面如下图所示：

1) 常规属性页签

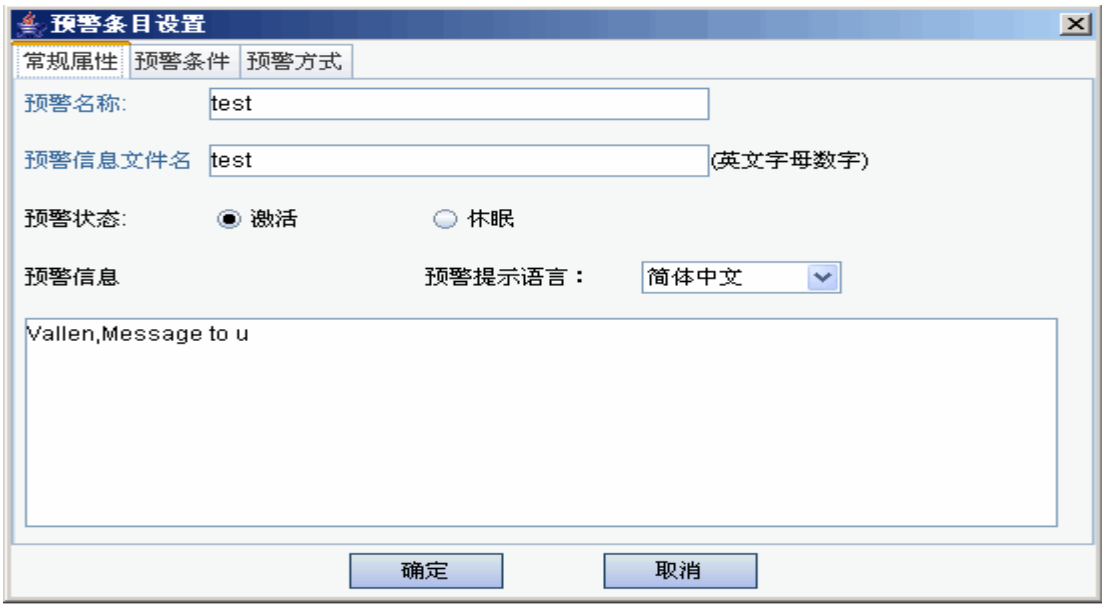

图 3-2 条目常规属性页签

- ✓ 预警名称：即该条目的名称，一个显示的标记。同公司同类型不允许条目名相同。
- ✓ 预警消息文件名：即消息生成时的 HTML 的文件名标记。(该 HTML 文件完整的名称是由它和生成时时间组成)。
- ✓ 预警状态：激活态表示该条目是有效的，反之休眠则表示此时该条目是无效的。默认为激活态。
- ✓ 预警消息：这个消息只有当消息接收配置为邮件时候才显示的邮件内容。
- ✓ 预警提示语言：这里指定是当消息生成时候，调用插件的某些显示(如HTML文件的标题)时，它的多语语种的选择。亦可参见 多语支持

2) 预警条件页签

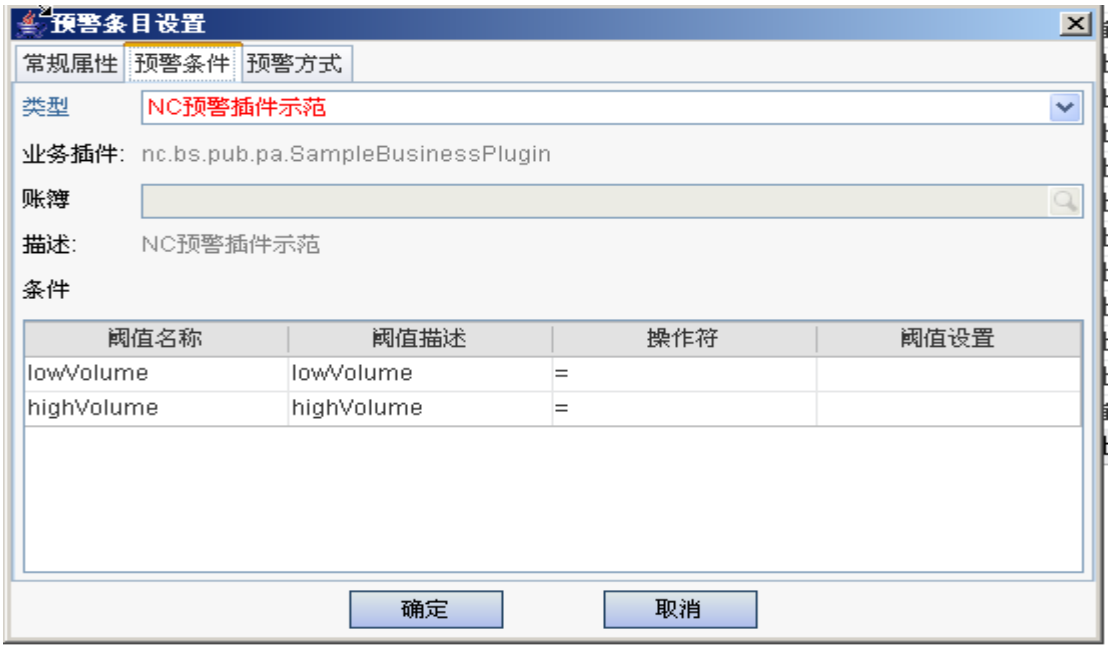

图 3-3 条目-预警条件页签

项目说明：

- ✓ 类型：即第二章所述的预警类型注册的预警类型。其以下拉框的形式显示，此处对预警类型的选择将决定此预警条目将调用的业务插件。
- ✓ “条件”列表：在此处编辑预警条目的阈值。这里的阈值是从类型定义中带过来的，这里要做的只是设置操作符合阈值设置。
- ✓ 帐簿:只是对于模块为财务和总帐的预警类型才必须设置值。其参照为主体账簿参照。注其预警类型也是必须实现 业务插件接口 3。

3) 预警方式页签

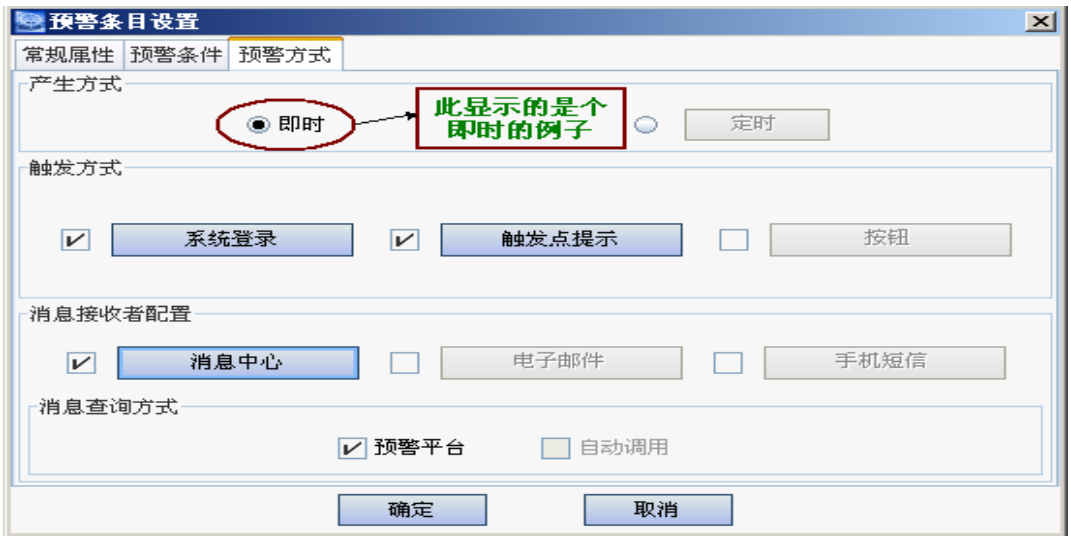

图 3-4:预警方式页签

- ✓ 产生方式：其分为即时产生与 定时产生。两者的意义上的区别可以参见:前言。如果选择的为即时产生，则系统会根据此处定义的触发方式来触发业务操作，并依据条件满足与否，来产生预警提示信息。如果选择定时产生，则预警平台会在设定的时间配置到来时进行业务检查，进行预警检查，并产生预警信息。两者产生的预警消息的如何接收都由消息接收者配置面板的来配置。关于 定时配置稍候叙述。
- ✓ 触发方式： 只有当产生方式为*即时*的时候，此组才能编辑和有效。
  - ✧ 系统登录：勾选并单击“系统登录”按钮，弹出系统登录用户选择界面。(如图 3-5)左侧为对本公司及其下级公司(通过界面的参照来切换)拥有登录权限的角色和用户。右侧为已经选择的用户。这些用户在登录NC时，如果系统有定义了该用户登录条目，并满足产生消息的条件，这时候会在消息中心的预警消息栏自动给登录用户发送一条消息，用户可以点击此来查看详细信息。但不会主动弹出IE。这点也是V5 和以前的版本的不同之处。详细可见 新特性。
  - ✧ 触发点提示：勾选后单击“触发点提示”按钮，弹出“触发点选择”界面(如图 3-6)，左侧为系统功能结点树，右侧为将触发预警的功能结点。当用户进入该公司打开已经定义有条目功能结点时，如果有符合条件的预警消息产生，则会弹出 IE 窗口来显示预警信息的详细内容。注意：集团的是不能定义功能节点触发的。
  - ✧ 按钮:只适用于 HR。即在业务单据的某个按钮点击时触发。

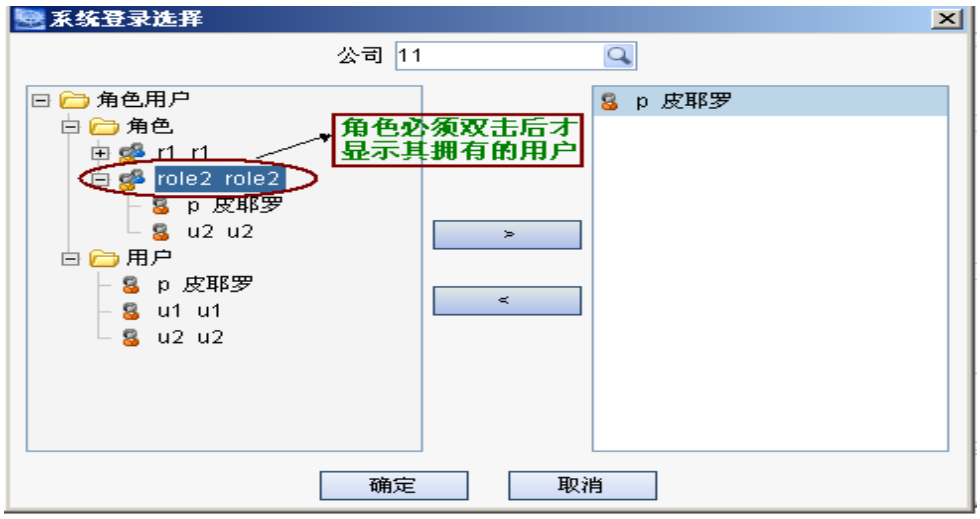

图 3-5:系统登录对话框

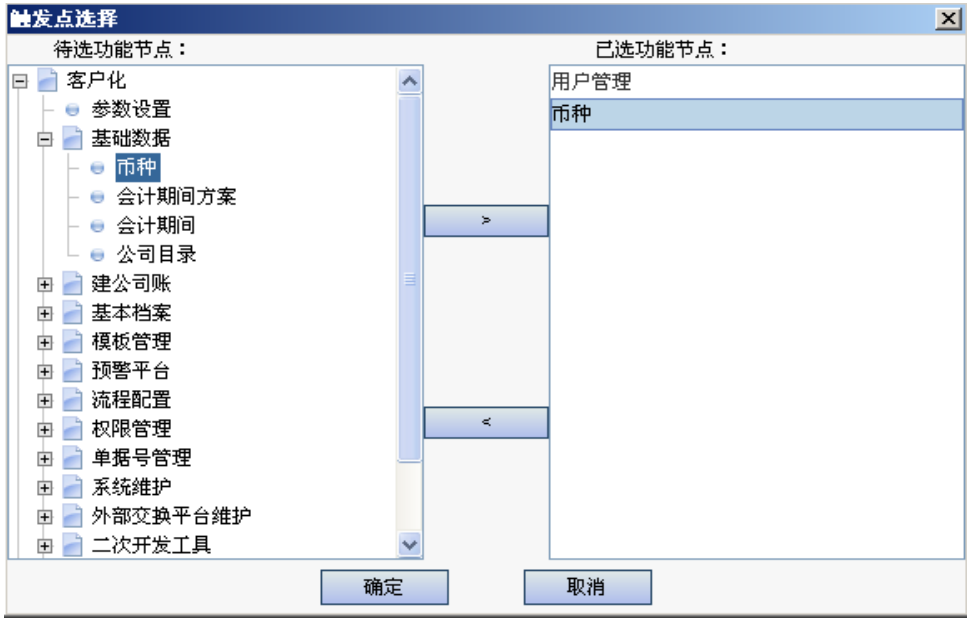

图 3-6:功能节点选折对话框

✓ 消息接收者配置：配置消息的接收者。即当预警条目触发时，除了适当的时候弹出 IE 之外的给用户的提示的配置。

✧ 消息中心：勾选后单击“消息中心”按钮，弹出消息中心配置界面。其界面相似于系统登录界面（即图 3-5）。但意义不一样。它的意义是：当一个预警条目触发时是否给用户送一条预警消息到消息中心。消息中心：即为用户登录 NC 系统时候收到的与之相关的消息，其包含公告栏、待办事务和预警消息等三栏。我们在预警所说的消息中心即为此处的消息中心的预警消息栏，以后亦同。

✧ 电子邮件：勾选后单击“电子邮件”按钮，弹出电子邮件地址选择界面。中间列表显示已经配置的Email.即可以根据人员档案来选择，也可以手写。如图 3-7。电子邮件要发送成功，必须正确配置邮件服务器。可以参见 系统环境配置

✧ 手机短信：勾选“手机短信”，其界面类似于登录消息中心配置界面。只是其会根据所选用户去关联其手机号而发送短信。

图 3-7 邮件选择对话框

✓ 消息查询方式：这个配置主要是方便查询。当为即时，则预警平台可以勾选掉，它的作用是当没有选择任何一个接收方式时，其便能在查询处纪录触发过的预警。同理当为定时，自动调用也是起类似的作用，所不同的是其不能勾选掉。关于查询操作详细见 第四章预警平台查询。

✓定时配置：当产生方式选择的是定时：

(1) 此时的触发方式将不能编辑。

(2) 此时的消息接收者配置与即时意义是一样的。

(3) 此时的消息查询方式意义相同，但是不能编辑，只为自动调用。

(4) 定时配置界面(如图 3-8)

如图 3-8 定时配置

- ✧ 发生频率：包含天、周，月等三个时间量纲。当为周或月时候，还能够选折对应的哪天，以及关于量纲的间隔。
- ✧ 一天内：因为不管频率制定的如何，具体到还是某一天中。这里就是具体设置某一天内的时间关系。
- ✧ 有效期：这是优先级最高的设置，即频率和一天内的设置都必须要在此有效期内。

系统会在设定的时间点进行业务检查，触发并合适地产生预警信息。

2.2 复制预警条目

如图 3-9:其左边待选树只有两层结构。一级为公司，二级即为条目。复制的原则是同一类型在同一公司下不允许同名。故选择完会一般要进行编辑，以保证满足此原则!

图 3-9 条目复制

3. 其它

在预警条件配置首界面上有[预警升级]菜单。此菜单主要用来升级 3.5 的预警条目! 详细查看 升级指南

第四章 预警平台查询

预警平台查询是为了方便用户查询当前产生的所有预警消息以及产生过的历史预警消息。预警平台查询主界面如图 4-1 所示。

其查询的依据是条目配置中的触发方式和接收方式以及查询方式。其新旧的标准是产生日期与当前日期的时间差是否在一个月内。

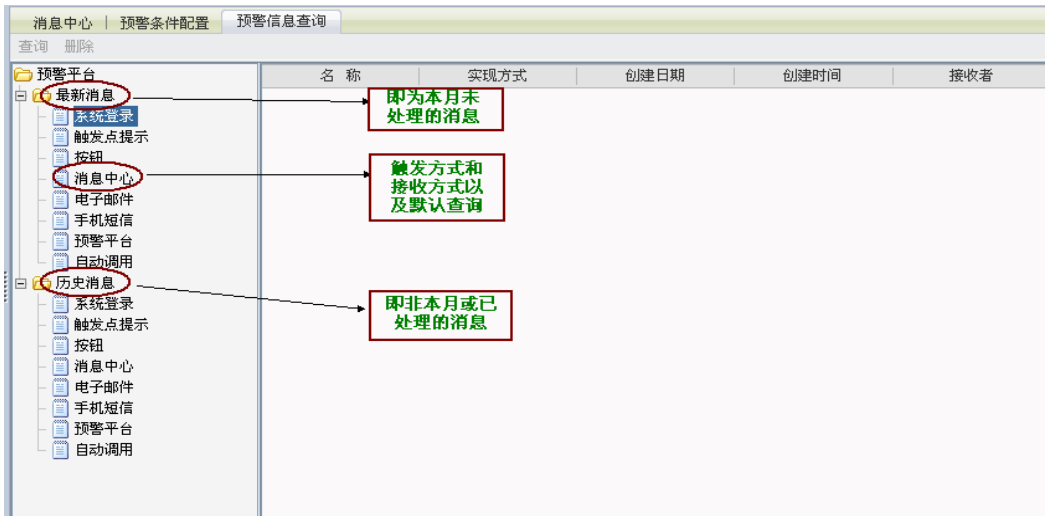

图 4-1 预警消息查询界面

此外查询处还提供了多样性的查询。如图 4-2.

图 4-2 预警消息查询对话框

多个名称或者多个接收者之间以分号分离。且两者支持模糊匹配。

第五章 插件开发指南

1. 简介及开发步骤

适用人群：本指南适用于对预警平台进行二次开发的开发人员。
开发步骤：

1) 开发人员先实现预警平台规定的接口 (如 5.2 所述)
2) 增加预警类型。（如 第二章 预警类型注册所述）
3) 增加测试预警条目。（如 第三章 预警条目注册所述）
4) 测试插件条目。依照条目定义，或打开节点，或登录系统，或定制时间。并查看消息是否如插件所意。

2. 预警平台业务插件接口

定义预警类型时必须提供做业务检查的业务插件，由开发人员编写。该业务插件必须实现预警业务插件接口。预警服务运行时，根据定义的预警条目执行业务插件的适当业务，并将产生的预警信息写入预警文件，进行企业业务预警。

2.1 业务插件接口 1

nc.bs.pub.pa.IBusinessPlugin 最普通最原始的接口。定义接口如下:

```
public interface IBusinessPlugin {
public int IMPLEMENT_RETURNMESSAGE = 0;
// 返回一个字符串信息,后台会生成只包含一个字符串的HTML文件
public int IMPLEMENT_RETURNOBJECT = 1;
// 返回一个可以序列化的对象,后台会生成一个序列化的文件
public int IMPLEMENT_WRITEFILE = 2;
// 自定义写一个文件,即插件自己去写文件.预警后台不做任何处理.(集群除外)这时候条目的文件名即为生成的文件名，不允许再拼接
public int IMPLEMENT_RETURNFORMATMSG = 3;
// 返回格式化的信息...,后台会生成一Table格式的HTML文件.

/*返回实现类型，其值为如上几个常量*/
public int getImplmentsType();

//下面的四个接口实现依照返回类型，分别选择实现之一即可，并不需要都要实现*/
```

```
    /**
     * 返回给定格式的一个接口对象...
     */
    public                                nc.bs.pub.pa.html.IAlertMessage
implementReturnFormatMsg(Key[] keys, String corpPK,
            UFDate clientLoginDate) throws BusinessException;
    /**
     * 业务插件实现此方法时，如果只需要预警，则返回一个非空的字符串.*/

    public String implementReturnMessage(Key[] keys, String corpPK,
UFDate clientLoginDate)  throws BusinessException;
    /**
     * 当业务插件实现此方法时，则如果需要预警，则返回一个非空的的对象 */
    public Object implementReturnObject (Key[] keys, String corpPK,
UFDate clientLoginDate)  throws BusinessException;

    /**
     * 业务插件如果实现此方法,则意味着预警平台将 Key 数组和 fileName 传入 业务
插件可以根据 Key[]
     * 来判断是否需要预警，如果要，则将预警信息写入到 fileName 文件中，并且一定
要返回 true，否则视为不需预警     */
    public boolean implementWriteFile(Key[] keys, String fileName,
String corpPK,UFDate clientLoginDate) throws BusinessException;
}
```

其接口中其它在此未列出的接口，在 v5.0 种都没有用， 现在只是为了产品的向下兼容性而而保留，开发人员之需要返回 null 即可。

关于 key 值的说明：Key 其实就阈值的描述，但并不是 5.0 中对应的数据库表结构，由于也是为了向下兼容，而进行了转换，但对二次开发任意是透明的。

2.2 业务插件接口 2

nc.bs.pub.pa.IBusinessPlugin2

它是 nc.bs.pub.pa.IbusinessPlugin 的子类，适用于当业务插件的实现时候需要客户端的登录信息参数。

```
/**
     * 返回给定格式的一个接口对象...  */
    public nc.bs.pub.pa.html.IAlertMessage[]
    implementReturnFormatMsg (Key[] keys, Object currEnvVO,
    UFDate clientLoginDate) throws BusinessException;
    ……
    //其它的接口实现相识，故省略之
```

差别如上框线阴影可见，预警平台会传给插件一个currEnvVO.这个参数其实就是一个nc.vo.pub.pa.CurrEnvVO对象.其中包含了登录公司PK，登录用户Pk等信息。

2.3 业务插件接口 3

nc.bs.pub.pa.IBusinessPlugin3
也是 nc.bs.pub.pa.IbusinessPlugin 的子类。适用于插件的实现需要传递主体账簿的接口。

```
    /**
     * 返回给定格式的一个接口对象... */
    public
nc.bs.pub.pa.html.IAlertMessage    implementReturnFormatMsg(Key[]
keys, String corpPK,
String accountPk,UFDate clientLoginDate) throws BusinessException;
    ……
//其它的接口实现相识，故省略之
```

差别如上框线阴影可见，预警平台会传给插件一个 accountPk.这个参数意义是主体账簿的 Pk。

诚如 条目设置所述，其现在只支持财务(fa和总账(gl). 故其它模板即使实现了此接口也没用，因为到时调用时传过来的值就是null了。

3. 返回格式化消息接口(可选)

3.1 实现切入点

如果插件类返回类型为格式化信息对象。即插件实现类的返回实现类型方法返回格式化消息。如下所示：

```
public int getImplmentsType() {
        return IBusinessPlugin.IMPLEMENT_RETURNFORMATMSG;
}
```

那么在其实现类的实现方法中必须实现如下接口方法：(其它的接口方法不用实现)。

```
public IAlertMessage implementReturnFormatMsg(…)
    throws BusinessException {
   // 具体实现… …---
  /*
   return  null;(那么其不生成消息文件,只是执行一些动作和事情)
   或者 return一个实现了IalertMessage的对象
  */
}
```

3.2 术语解析

格式消息：即要求返回的是满足一定要求的一定格式的方便展示的消息，我们此处即为HMTL，而核心又在 HMTL 中的表格展现。

3.3 接口声明

✓ 格式化消息也有三个主要接口 nc.bs.pub.pa.html.IalertMessage;

nc.bs.pub.pa.html.IalertMessage1;nc.bs.pub.pa.html.IalertMessage2
后两者都是 IalertMessage 的子类，它们的接口方法差别如下表所示：

Interface	IAlertMessage	IAlertMessage1	IAlertMessage2 V5.0 增加新接口(**推荐使用**)
公有的	详细见下		
接口 2 特有		//表格宽度 String getTableWidth();	
接口 3 特有			int[] getBodyColumnType(); String getNullPresent(); String getOmitPresent(); 详细各接口方法意义见下

表 5-1 格式化消息各个接口的接口方法差异

✓ nc.bs.pub.pa.html.IalertMessage 的接口声明如下

```
public interface IAlertMessage extends java.io.Serializable {
    //… 省略若干常量的定义，推荐返回对应值时使用此定义的常量
    /** 得到消息体的域．通常为字段名称 */
    String[] getBodyFields();
    /** 得到消息体的值．通常为各字段的值．即数据部分*/
    Object[][] getBodyValue();
    /** 得到消息体的各个域的宽．通常为各字段宽即列宽． */
    float[] getBodyWidths();
    /** 得到消息格式底部值 */
    String[] getBottom();
    /** 得到消息格式的标题 */
    String getTitle();
    /** 得到消息格式的底部值．即临近Title下的内容 */
    String[] getTop();
    }
```

✓ . V5.0 新增接口(**推荐使用此接口**)

```
public interface IAlertMessage2 extends IAlertMessage {
    /** 定制各个列的类型。其相关类型见IAlertMessage中的定义 */
    int[] getBodyColumnType();
    /** 定制空值时的显示.<br> 实现接口者可以返回null,此时默认的为空格(""). */
    String getNullPresent();
    /** 定制缺省时的显示.<br> 实现接口者可以返回null,此时默认的为空格(""). */
    String getOmitPresent();
    }
```

4. 多语支持

这里的多语的意思主要是有两个方面：

4.1 预警类型注册的多语

在预警类型注册时候，我们只对预置的类型多语，这是需要对库直接操作，填充对应表的对应多语字段。此可参见 预警平台相关表。注意产品组的多语放在多语目录的prealerttype下。自己新建文件，必须符合多语规范了！

对于界面的操作，不能影响到多语。因为我们假设用户在何语言环境下增加类型就是在何语言环境下使用。

4.2 预警消息文件的多语

预警平台本身不知道该如何去多语，而是要插件自身去多语，而返回给预警平台，预警平台会根据条目配置中的语言选择，设置当前的线程的语言环境，然后去调用适当的多语。这不仅针对格式化消息，而是所有的消息都一样。

如实现接口 IalertMessage 的，多语代码应该用服务端的多语。

```
    public String getTitle(){
    return   NCLangResOnserver.getInstance().getStrByID("101502",
"UPP101502-000261");// "预警测试样例"
    }
```

第六章 程序实现举例

1. 业务插件举例

定义一个实现 IBussinessPlugin 接口的预警类型业务插件类，比如现在创建 **库存存量预警处理类 nc.bs.pub.pa.SampleBusinessPlugin**

```
package nc.bs.pub.pa;

import nc.bs.logging.Logger;
import nc.bs.pub.pa.html.IAlertMessage;
import nc.bs.pub.pa.html.SampleAlertMessage;
import nc.vo.pub.BusinessException;
import nc.vo.pub.lang.UFDate;
import nc.vo.pub.pa.Key;

/**
 * 预警平台插件接口实现示范类。兼预警平台的测试类.。
 *
 * @author huangzg 2006-10-10
 */

public class SampleBusinessPlugin implements IBusinessPlugin {

    public int getImplmentsType() {
//      return IBusinessPlugin.IMPLEMENT_RETURNMESSAGE;
        return IBusinessPlugin.IMPLEMENT_RETURNFORMATMSG;
    }

    // since v5.0 此接口方法没有用!返回null亦可。
    public Key[] getKeys() {
        return null;
    }

    // since v5.0 此接口方法没有用!返回null亦可。
    public String getTypeDescription() {
        return null;
    }

    // since v5.0 此接口方法没有用!返回null亦可。
    public String getTypeName() {
```

```
        return null;
    }

// 这里虽然实现了，但是由于返回的是格式化消息，所以等于没有起作用
    public String implementReturnMessage(Key[] keys, String corpPK,
UFDate clientLoginDate)
            throws BusinessException {
//业务实现。如果要返回格式化的HTML消息,请参考nc.bs.pub.pa.html.IAlertMessage
        double testValue = 10;
        double lowStorageVolume = -1, highStorageVolume = -1;
        if (keys != null && keys.length > 0) {
            for (int i = 0; i < keys.length; i++) {
                if (keys[i].getName().equals("lowVolume")) {
                    lowStorageVolume =
        new Double(keys[i].getValue().toString()).doubleValue();
                } else if (keys[i].getName().equals("highVolume")) {
                    highStorageVolume                =                new
Double(keys[i].getValue().toString()).doubleValue();
                }
            }
        }

        if (lowStorageVolume == -1 || highStorageVolume == -1) {
            Logger.error("预警类型配置未完成");
            return null;
        }
        if (testValue < lowStorageVolume) { return "预警平台测试示例:库存
安全最低量超过限制"; }
        if (testValue > highStorageVolume) { return "预警平台测试示例:库存
安全最高量超过限制"; }

        return null;
    }

    public boolean implementWriteFile(Key[] keys, String fileName, String
corpPK,UFDate clientLoginDate) throws BusinessException {
        return false;
    }

    public Object implementReturnObject(Key[] keys, String corpPK, UFDate
clientLoginDate)throws BusinessException {
        return null;
    }
```

```
    public IAlertMessage implementReturnFormatMsg(Key[] keys, String
corpPK, UFDate clientLoginDate) throws BusinessException {
        // FIXME 直接返回 一个格式化文件
        return new SampleAlertMessage();
    }
}
```

这个插件例子，是返回一个格式化的消息，直接 new 了一个格式化消息的实例类。而对于其中实现的方法 implementReturnMessage 没有用途，这里只是个示范。

1.1 格式消息举例

```
package nc.bs.pub.pa.html;

import nc.bs.ml.NCLangResOnserver;
import nc.vo.pub.lang.UFBoolean;
import nc.vo.pub.lang.UFDate;
import nc.vo.pub.lang.UFDateTime;
import nc.vo.pub.lang.UFDouble;

/**
 * 预警信息接口实现类示例。
 *
 * @author: huangzg 2006-10-10
 */
public class SampleAlertMessage implements IAlertMessage2 {
    /**
     * SampleAlertMessage 构造子注解。
     */
    public SampleAlertMessage() {
        super();
    }

    public java.lang.String[] getBodyFields() {
        String[] fields = { "I", "II", "III", "IV", "V", "VI" };
        return fields;
    }

    public java.lang.Object[][] getBodyValue() {
        Object[][] value = {
        { new String("XXXXX"), new UFDate(new java.util.Date()), new
Integer(0), new UFDouble(0),new UFBoolean(false), new FDateTime() },
```

```
        { new String("XX"), new UFDate(new java.util.Date()), new
Integer(1111),new    UFDouble(1111),    new    UFBoolean(true),    new
Long(3289324) },
        { new String("XXXXXXXXX"), new UFDate(new java.util.Date()), new
Integer(442452442),  new UFDouble(), new UFBoolean(true) },
        { new String("x"), new UFDate(), new UFDouble(4235435.54235), new
UFBoolean(true)  },{  new  String("XXXXXXXX"),  new  UFDate(),  new
UFDouble(54325234.5243), new UFBoolean(false) },
    { new  String("XXXX"),  new  UFDate(),  new  Integer(234),  new
UFDouble(345234543.54235),new UFBoolean(true) },
    { null,  new  UFDate(),  new  Integer(324),  new  UFDouble(),  new
UFBoolean(false) },
    { new  String("XXXXXXX"),  null,  new  Integer(423541153),  new
UFDouble(), new UFBoolean(true) }};
        return value;
    }

    public float[] getBodyWidths() {
        float[] widths = { 0.2F, 0.15F, 0.25F, 0.3F, 0.1F, 0.1F };
        return widths;
    }



    public java.lang.String[] getBottom() {
        String[] bottom = { "AAAAAAAAA", "", "BBBBB", "CCCCC", "DDDDD",
"EEEE", "FFF", "GGGG", "HHHH",
                "II", "JJJJJJJJJ" };
        return bottom;
    }

    public String getTitle() {
        // return "预警信息示例";
    //多语
    return       NCLangResOnserver.getInstance().getStrByID("101502",
"UPP101502-000261");// "预警信息示例";
    }

    public java.lang.String[] getTop() {
        String[] top = { "AAAAA", "PPPPPPP", "BBBBB", "CCCC", "DDDDD",
"EEEEEE", "FFFFFFFFF","GGGGGGGGMMMMM", "NN" };
        return top;
    }

    public int[] getBodyColumnType() {
```

```
        return new int[] { 0, 1, 6, 0, 0, 0 };
    }

    public String getNullPresent() {
        return "null";
    }

    public String getOmitPresent() {
        return "omitted";
    }
}
```

这个例子只是一个例子，并没有什么查询表或者业务逻辑等。在实践环境中可以插件类同时实现业务插件接口和格式消息接口。

第七章 系统环境配置

1. 调度引擎的配置

这里配置 NC 服务器启动时，加载一些需要调度的任务。对于预警平台就是是否需要加载定时的条目。这个在 ncSysConfig 中也能配置

文件路径：.\ierp\bin\scheduleengine.xml

```
<autoLoader>
<classQualifiedName>nc.bs.pub.pa.PreAlerLoader</classQualifiedName>
      <moduleName>uap</moduleName>
      <priority>1</priority>
      <ignoreDataSource>false</ignoreDataSource>
      <enabled>true</enabled>
    </autoLoader>
```

enabled：是否服务器启动时候加载。默认为 true.

ignoreDataSource：如果忽略数据源，则设置之为 true，只调度一次，否则对不同数据源调用多次.

2. 邮件发送相关的配置

这个是与 NC 中所有的邮件配置是统一的.即在系统配置(ncSysConfig.bat)中有 UI 可以配置。这里我们也可以直接操作文件。

文件路径：.\ierp\bin\ message4pf.xml..

```
<smtp>
    <user>huangzg</user>
    <password>ajidfjaijfaidjdfi</password>
    <sender>Show Name</sender>
    <smtp>mail1.ufida.com.cn</smtp>
    <pop3>mail1.ufida.com.cn</pop3>
  </smtp>
```

Password:是密文，所以最好是通过 UI 配置了

Sender:只是显示发送时候显示的发送人名字。

其它的详细就不用解释了吧!

3. 消息文件存放路径

文件目录路径：..\webapps\nc_web\PreAlart\Messages\

4. 预警日志

日志配置：..\ierp\bin\logger-config.properties 配置文件中，
设置预警的日志级别 `prealert.level=DEBUG`
日志查看：`nclogs\pa-log.log` 即为预警的日志文件

第八章 V5 新特性及与以前版本区别

1. 系统登录的即时预警

v3.5：系统登录会弹出 IE 提示

v5.0：系统登录不会弹 IE，只会给触发此预警的人发一条消息。此显示显示在消息中心的预警消息栏。此时候你亦可以双击察看。此时可以对这条消息标记处理，删除等，注意这里删除并不是真正的删除，只是把其扔进回收站，用户可以打开消息管理来对其进行管理(左边的页签)。详情可以查看消息管理使用文档。

2. 升级(从 3.5-5.0)

2.1 升级的必要性

- ✓ 储存机制调整

由于 V5 最大的改变是将预警原来 XML 文件存储机制改成数据库存储，这样重构不仅是因为系统集群部署的需求，而且也是为了解决原来的一些性能问题。所以此次的升级主要的也是把原来的 XML 文件转换为对应的数据库中表。

- ✓ 由于 V5 的账套升级不能自动升级预警的条目，于是条目必须手动升级。

2.2 升级先决条件

如果对于想升级非预置的类型的条目，那么用户必须先增加对应类型，然后再执行升级。

执行升级操作最好是在服务器本机上操作，或者在其它客户端，这时候应该先把预警的原先设置目录在网络上共享。

2.3 升级操作

其可以执行两步操作，分别升级预警条目，和预警消息文件。其首界面如图 8-1 所示。还是路径指定问题：如先决条件里所述，如果是在服务器不在本机的客户端操作，即使欲升级的文件是在本机也请共享并填入共享路径。

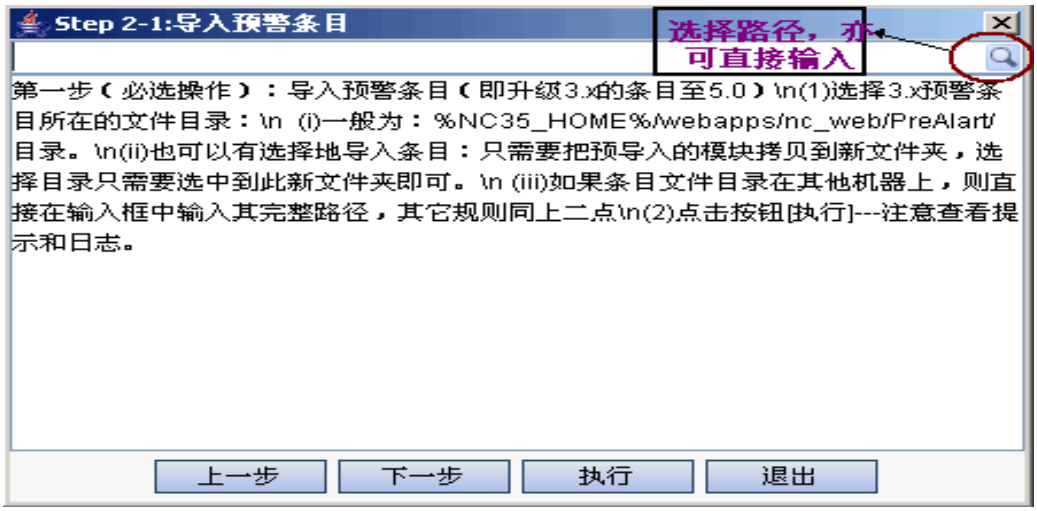
Step 2-1:导入预警条目
选择路径，亦可直接输入
第一步（必选操作）：导入预警条目（即升级3.x的条目至5.0）\n(1)选择3.x预警条目所在的文件目录：\n (i)一般为：%NC35_HOME%/webapps/nc_web/PreAlart/目录。\n(ii)也可以有选择地导入条目：只需要把预导入的模块拷贝到新文件夹，选择目录只需要选中到此新文件夹即可。\n (iii)如果条目文件目录在其他机器上，则直接在输入框中输入其完整路径，其它规则同上二点\n(2)点击按钮[执行]---注意查看提示和日志。
上一步
下一步
执行
退出

图 8-1 升级操作

附录

1. 预警平台相关表

预警类型	pub_alerttype	预警类型主表
	pub_alerttype_b	预警类型子表(阈值)
预警条目	pub_alertregistry	预警条目主表
	pub_alerttypeconfig	对应类型配置(阈值配置)表
	pub_alertsendconfig	发送发式配置表
	pub_timeconfig	时间配置(主要是定时)
预警消息	pub_alertmessage	预警消息索引表
引用其它平台表	Pub_messageinfo	消息中心