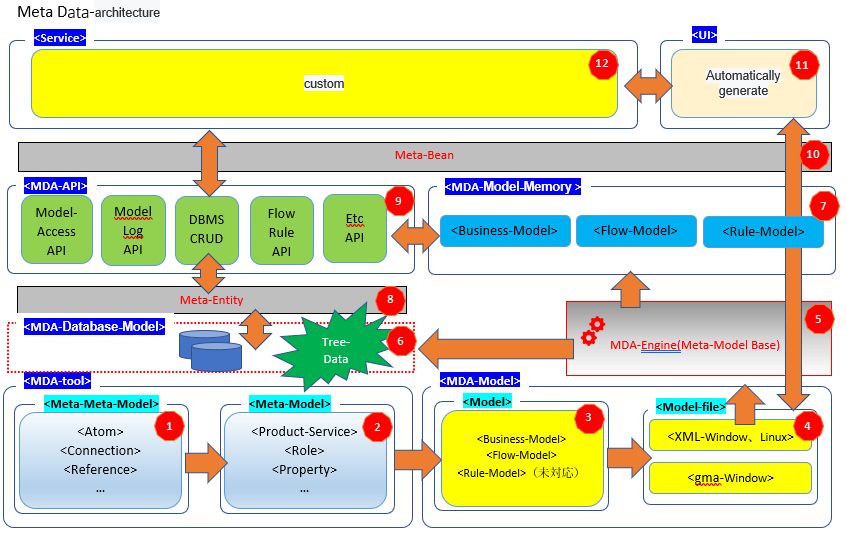
SmartKAYA

**User Guide**

(version 0.0.1)



PS: 未开放或不提供该功能。WorkFlow功能目前没有开放，RuleModel不准备开发，业务规则还是代码可扩展比较灵活，效率也更高一些。所以这个平台并不是面对没有代码基础的开发人员。

**考虑多平台开发的优势，将来也考虑推出基于Web服务器版本的开发工具，但是需要Node.js 的高级技术人员才能实现。有想投资参与的同学可以联系我（微信：chljapan）。注意不是普通的Nodejs，需要精通才行。目前已经实现部分的建模功能，但是解析部分有一些问题。**

1. **Meta-Meta-Model**

元元模型（GME提供）

1. **Meta-Model**

利用Meta-Meta-Model要素来定义元模型，构成整个平台的基础元素。

整个系统平台的核心。

1. **Model**

利用Meta-Model定义的元素，来构建我们的业务模型。

1. **Model-file**

③定义的模型文件。(XML 、MGA)。

※虽然系统平台不同，但是我们提供统一的解析引擎。

1. **MDA-Engine**

④对生成的元数据（业务模型）进行解析，生成系统平台能够调用的对象集合。

1. **MDA-Database-Model**

Tree结构的数据构成。设计充分考虑了数据的通用性和原子性。

1. **MDA-Model-Memory**

数据模型的内存集合。

1. **Meta-Entity**

通用Entity

1. **MDA-API**

MDA操作模型包括数据库UI等的API。

1. **Meta-Bean**

通用Bean

1. **UI**

画面组件信息。

1. **Service**

复杂的业务需要手动操作模型、调用MDA―API进行开发。

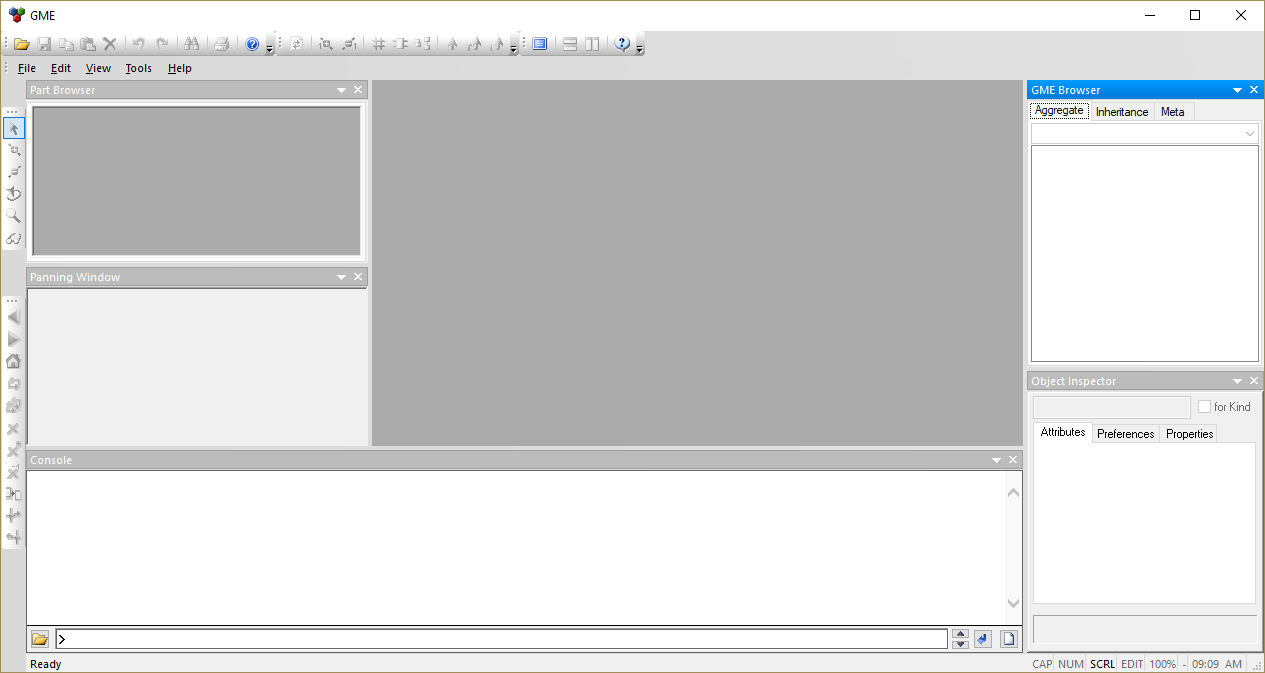
**第一章．建模**

1.1概要-KAYA的基础建模。

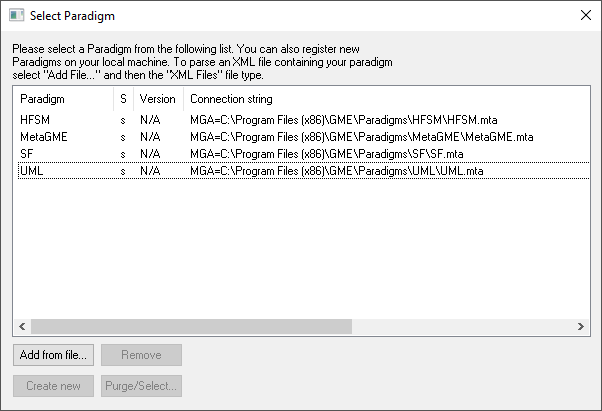
GME是范德比尔特大学（VANDERBILT UNIVERSITY）开发的用于特定领域建模研究的开源工具。下载地址：http://repo.isis.vanderbilt.edu/GME/old/（这里用的是Ver.17.12.6 ）。

有32位和64位可供选择，建议选择64位版本。

我们打开GME:



Ctrl + N 打开Select Paradigm画面。



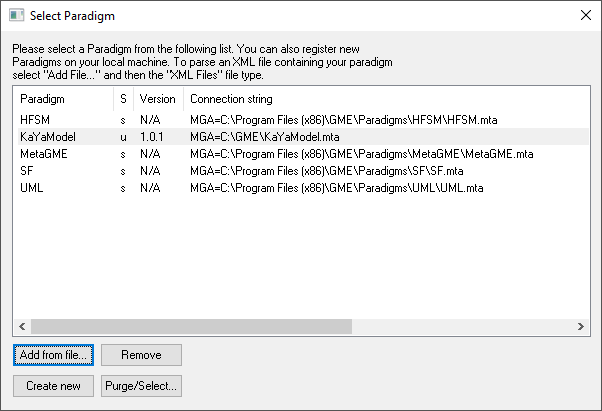
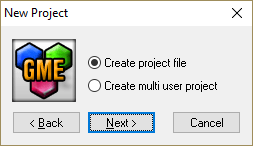
点击Add from file按钮

在对话框中选择 KaYaModel.mat文件（在压缩包中），点击Open按钮。然后如下图

*补充：KayaModel.mat是我们的元模型工具文件。是已经为大家开发好的。这也是整个Kaya模型驱动的最核心的部3.分。如需开发特殊的元模型文件和模型引擎请联系技术人员：*[*chljapan@hotmail.com*](mailto:chljapan@hotmail.com) *或搜索微信号：chljapan*

*或自行学习GME开发手册。*

我们将KaYaModel.mat选中，点击Create new按钮

点击Next> 输入文件名（KaYa001）后点击Save（保存）

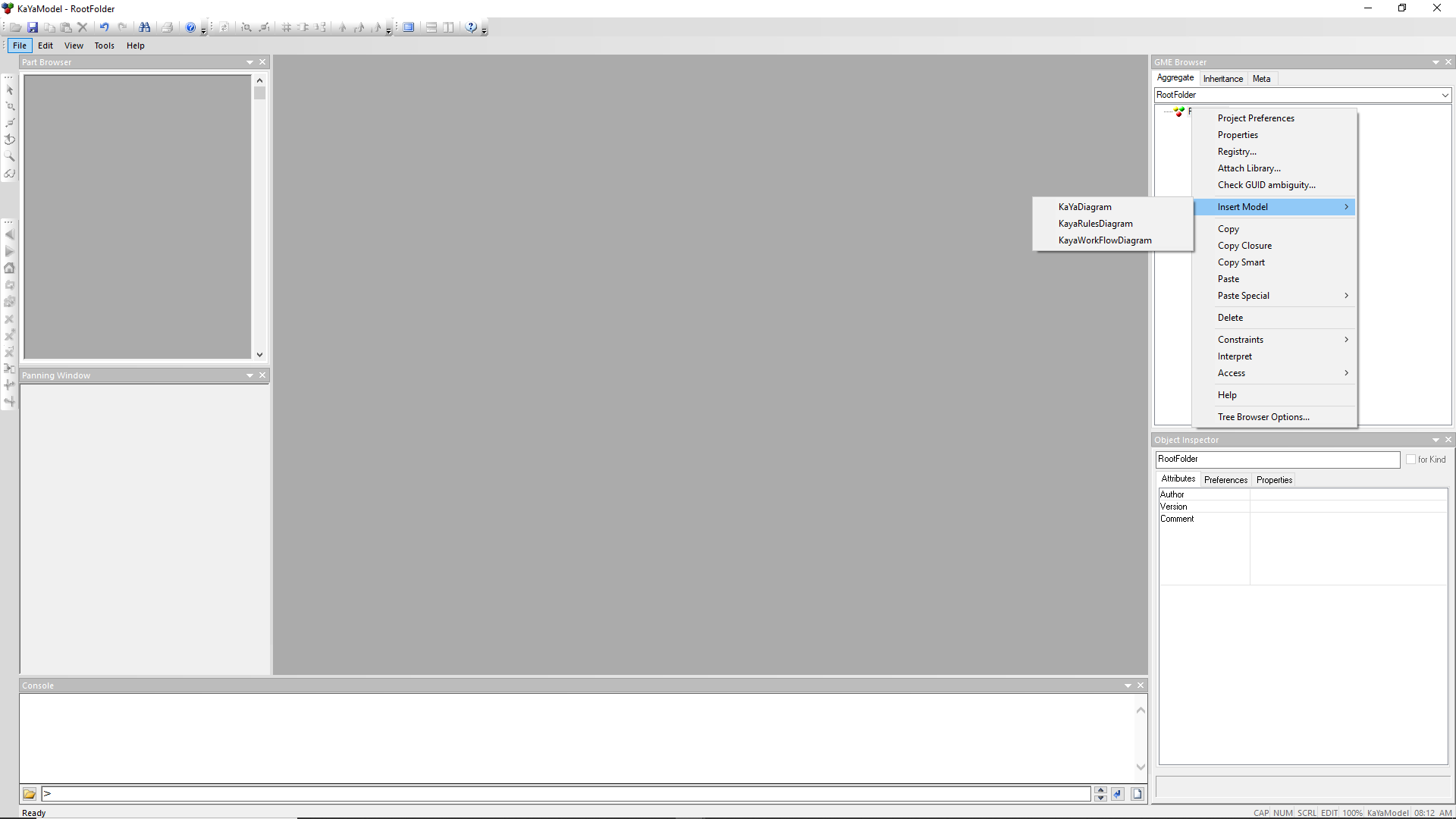
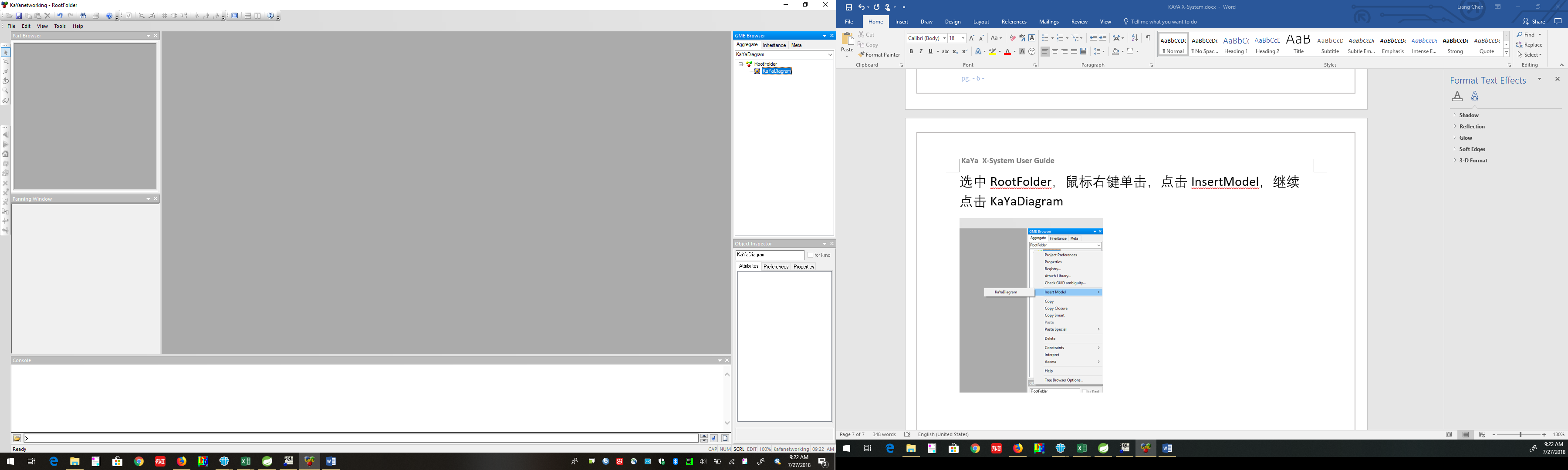


这里是你项目模型文件的开发者版本等

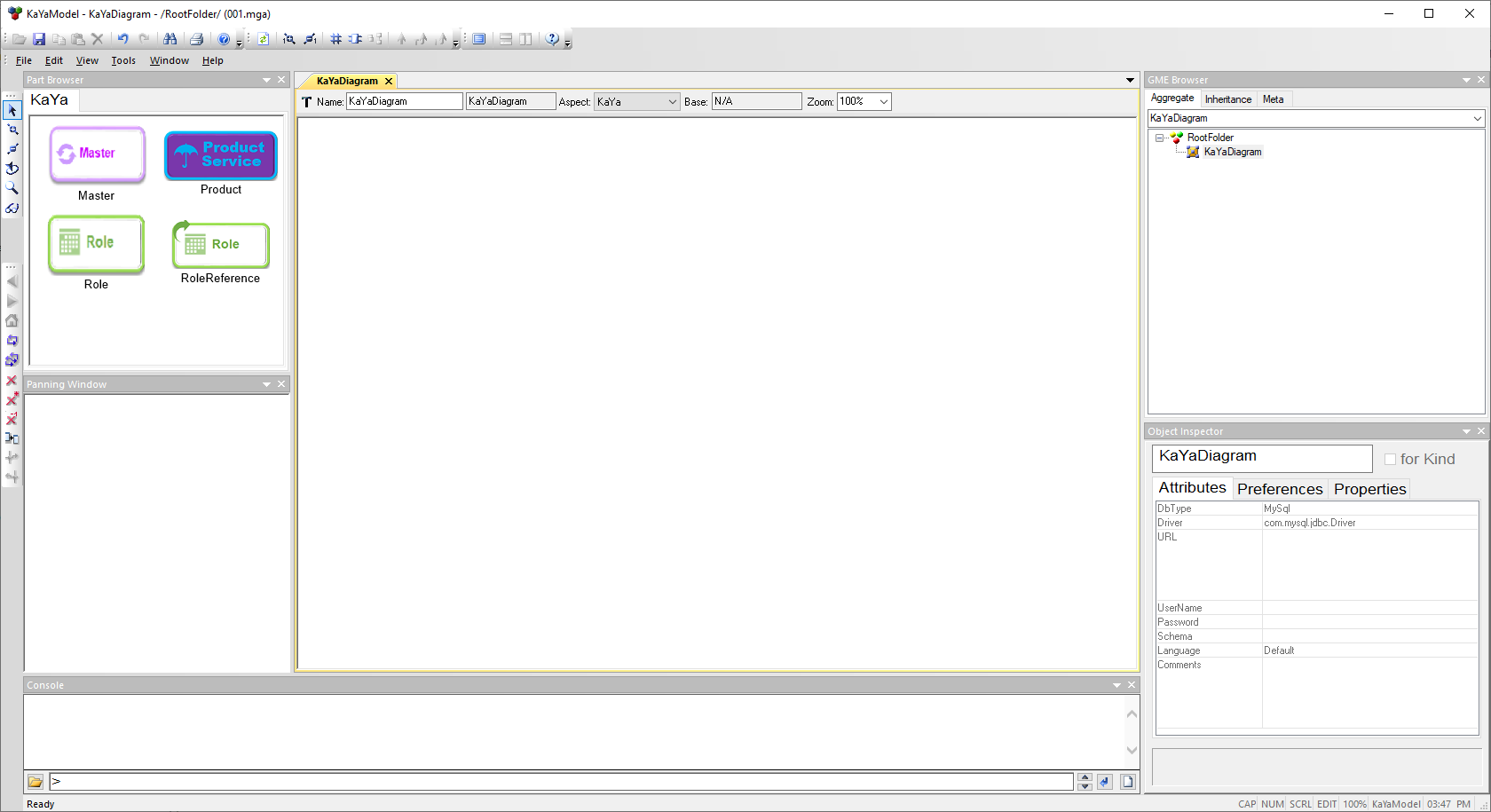
这里会生成Root Folder项目文件夹

选中Root Folder，鼠标右键单击，点击Insert Model，继续点击KaYaDiagram选项。

（不要试图添加KayaRulesDiagram和KayaWorkFlowDiagram选项，因为是开发中，所以不具有运行功能，而且会带来解析错误）

 我们会看右侧的GMEBrowser变成右图所示 

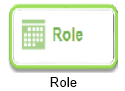
我们鼠标双击KaYaDiagram，工具的左侧会显示一些图标。



我们建模的所有操作都会在上面的界面中完成。

让我们来熟悉一下左侧的图标。

**Product**: 这个是整个系统的集合。各种表，规则，流程都会从属于这个集合。当然系统可以包含子系统。 所以Product是可以连接Product的。

****

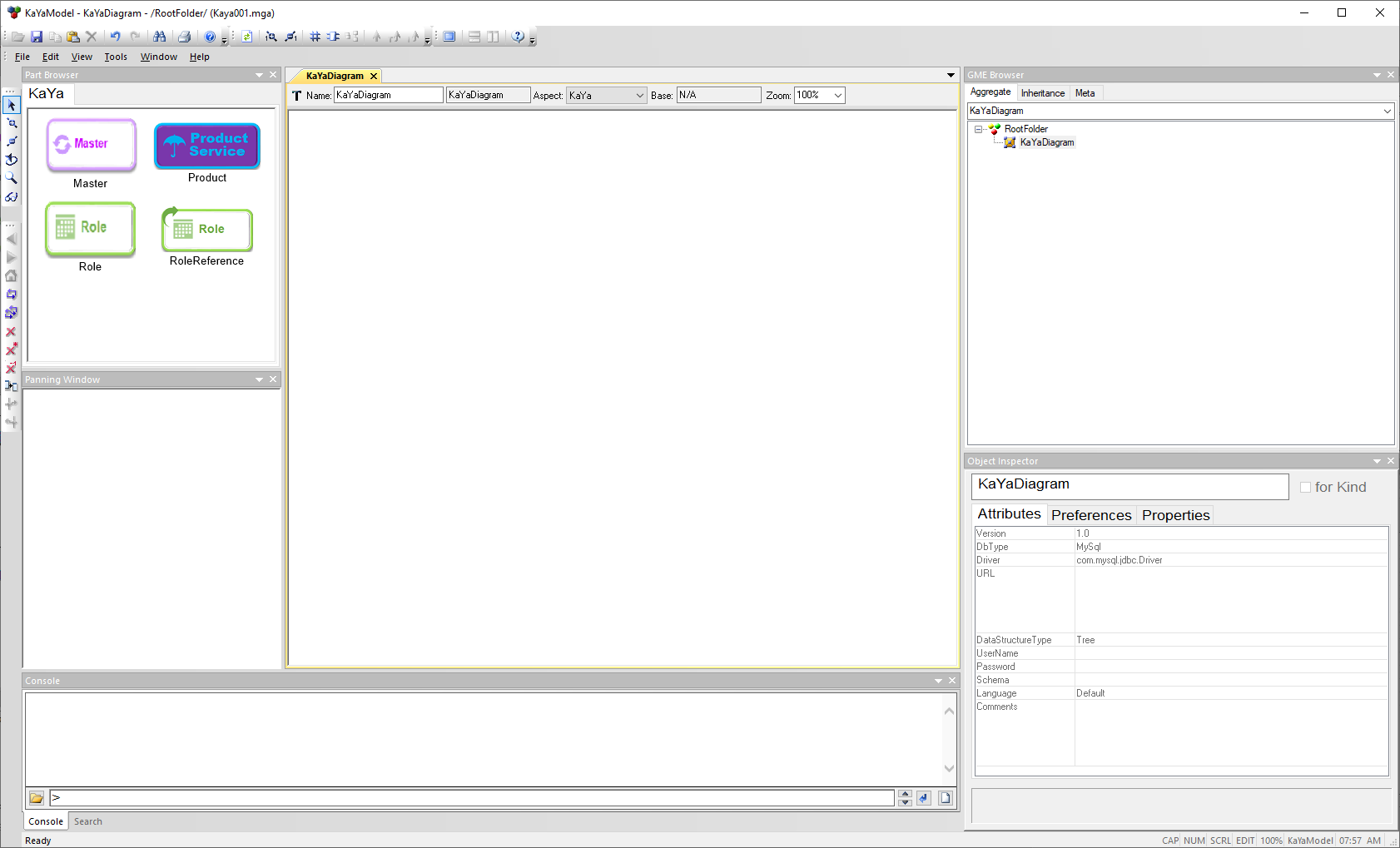
**Role**: 这个图标你可以理解成数据库中的表，也就是模型驱动中的模型。实际的业务中代表具有多种特性的某一角色。例如：员工这一角色，他又包括年龄，性别，联系方式，地址，工资等多种属性。



**RoleReference**: 这个我们可以理解为角色的参照。在后续的流程引擎中，我们会进行更为详细的介绍。



**Master**: 这个是用来建Master数据的，比如性别的：男，女等，部门的：财务，生产，人事等使用时再详细介绍。



连线区工具栏

组件区

浏览器区

属性设置区

控制台输出区

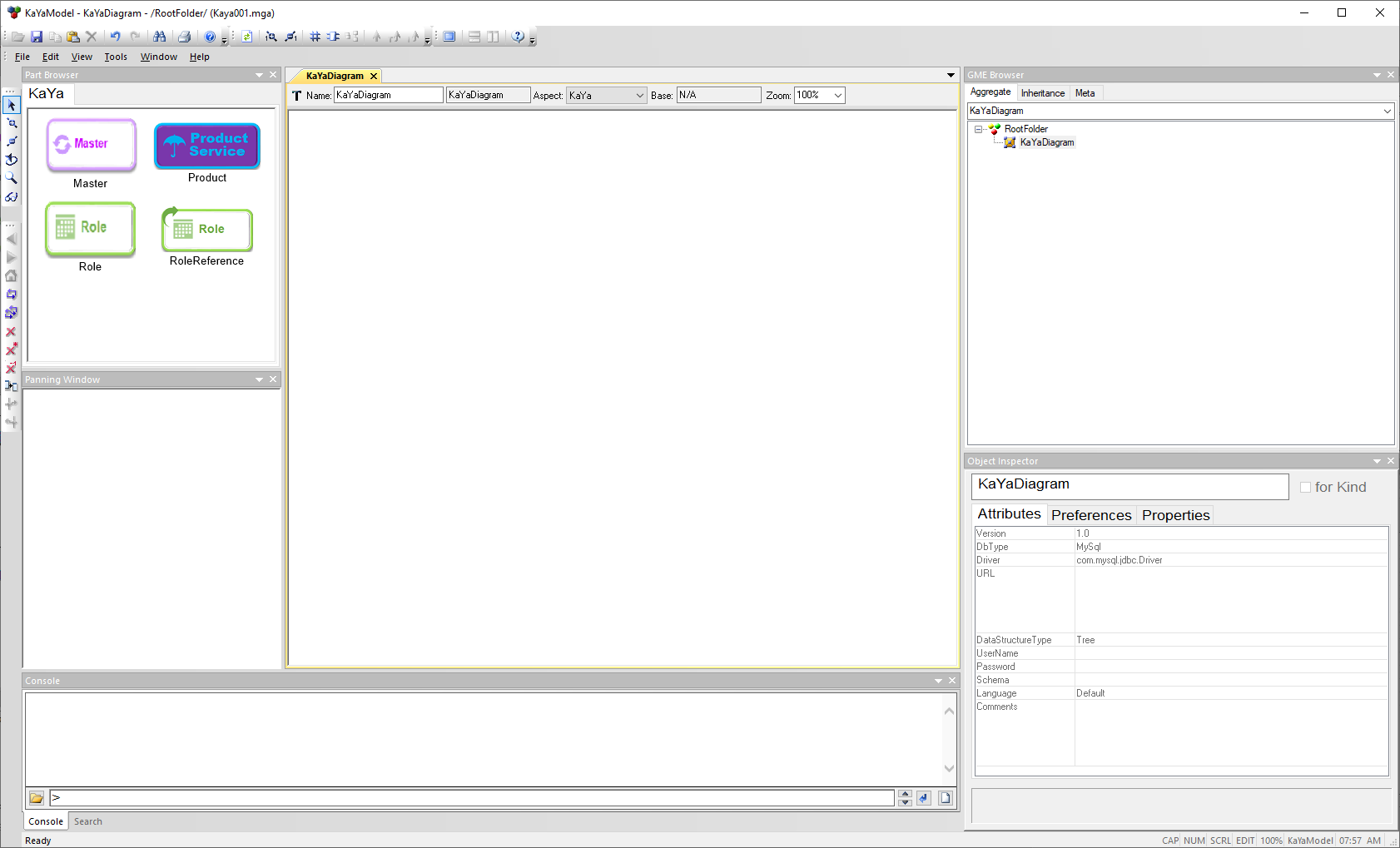
缩视图区

菜单区

主作业区

我们通过操作来熟悉一下这些组件。

新建一个模型取名为Kaya001.mga



红色区域的信息包括：

**Version**：模型版本号（用来作为模型的多版本管理）

**DbType**：可以选择数据库类型 例：MySql(目前实现了MySql，H2，PostgreSql)

**Driver**：选择对应数据库的驱动 例：com.mysql.cj.jdbc.Driver

Host: 127.0.0.1:3306

DatabaseName：best\_ecommerce

DbPropertys：(没有可以不填)

**MySql**数据库连接例：

jdbc:mysql://127.0.0.1:3306/best\_ecommerce?

serverTimezone=UTC&useSSL=true&useUnicode=true&characterEncoding=utf8

*PS: 如果你是在容器中利用KaYa，请务必关掉useSSL 。*

**DataStructureType**: 数据存储类型 例：Tree（树状数据存储和关系型数据存储），目前只支持树型数据库存储。

这也是KaYa平台的特性。

**UserName**: 数据库的用户名

**Password**：数据库登录密码

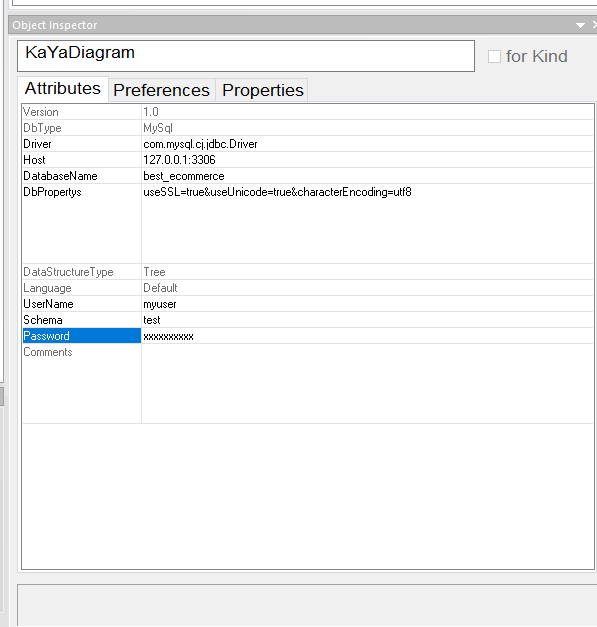
**Schema**: Schema(多租户策略的一种选择，默认提供多租户管理)

**Language**: 系统语言选择（按照你的语言来设置UI以及数据库字段显示）如果是Default这显示红框位置作为标签。

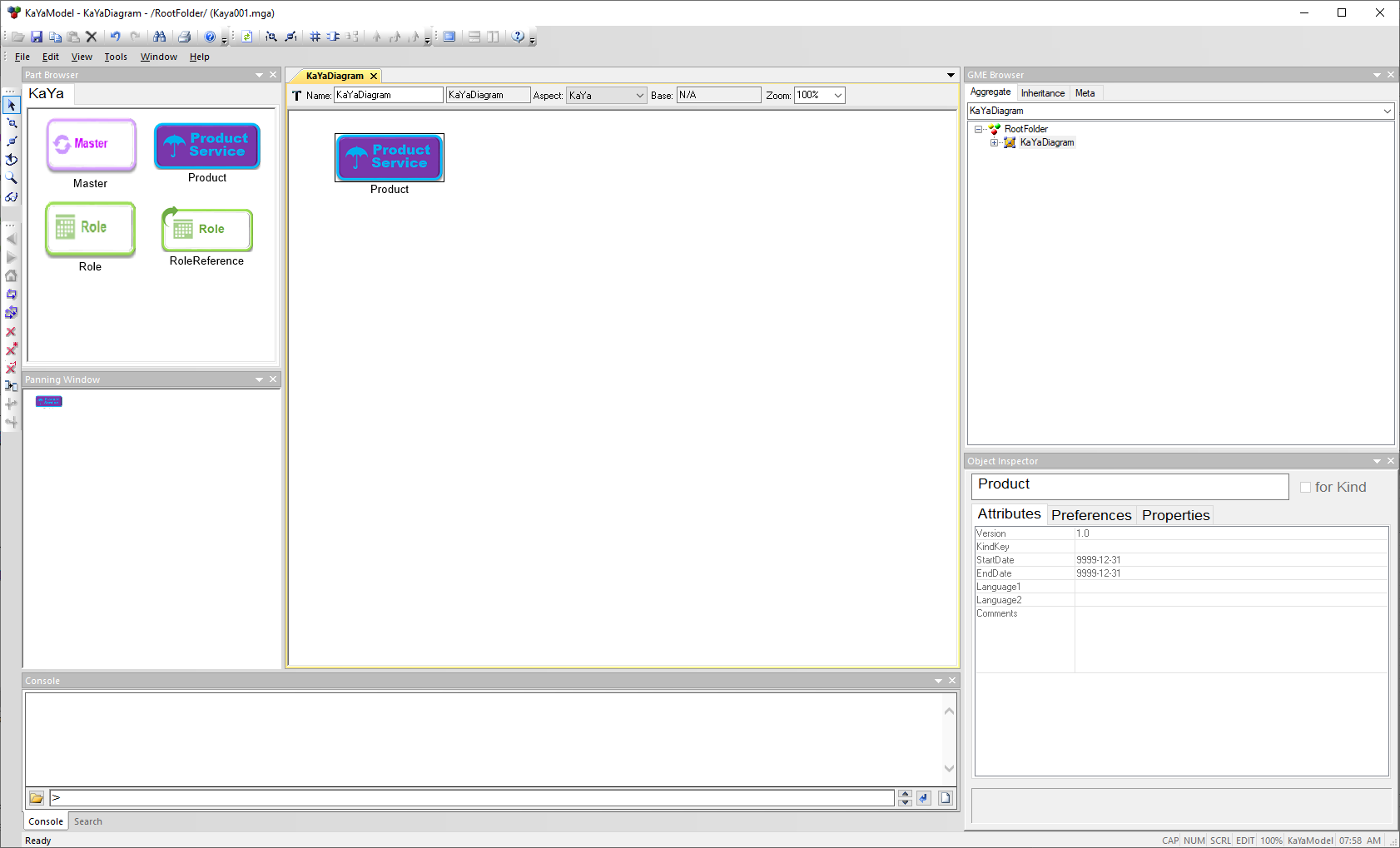
其他位置对应ZN,JP,EN等的系统语言信息。

**Comments**：补充信息，方便理解该模型元素的注释信息。

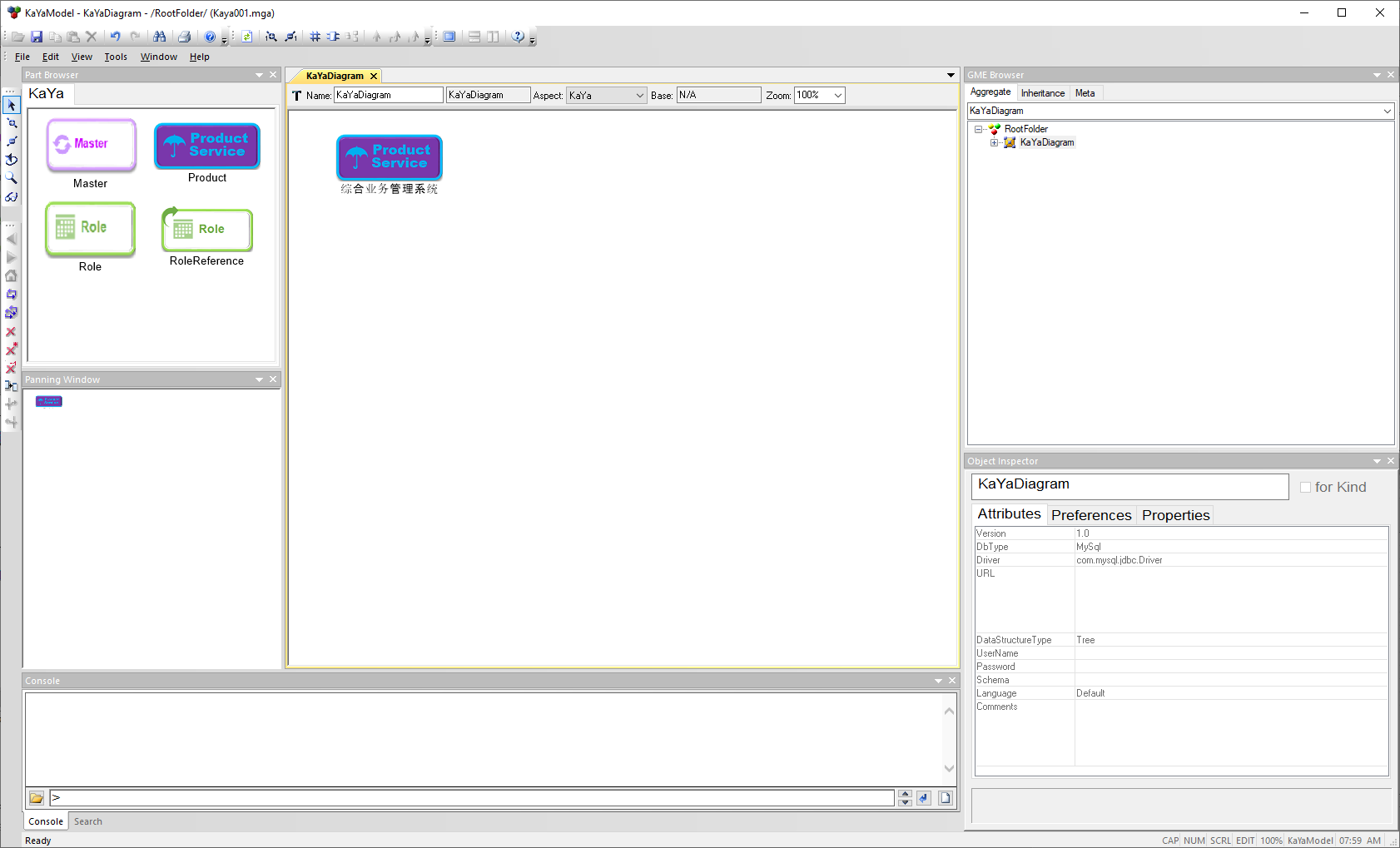
更改信息如下：



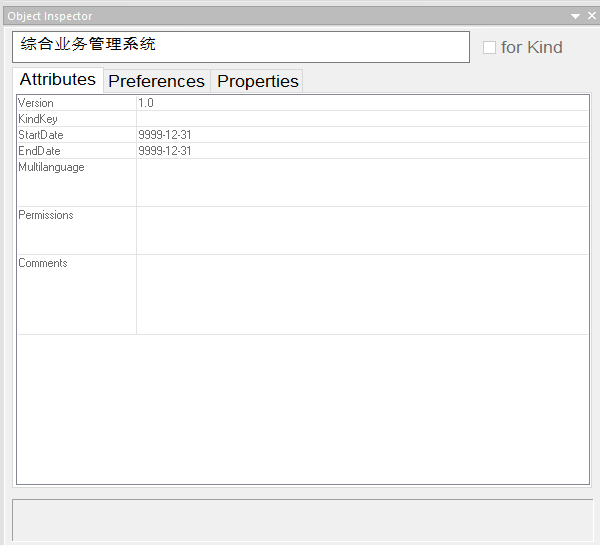
我们来添加一个系统



设置系统名称为【综合业务管理系统】（双击图标下面的Product字体位置可以直接更改该图标名称，或者直接更改红线区域信息）



这里有几个ProductService的属性，我们来熟悉一下



Version：版本管理

KindKey：关键字，用来识别该控件的唯一标识。

StartDate: 用来控制该控件的开始周期。

（默认值为9999-12-31 ，代表无效，

其他值视具体业务来规定，下同）

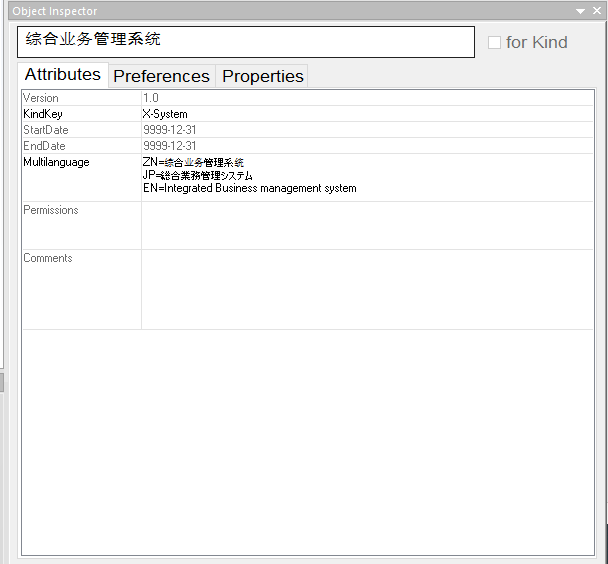
EndDate：用来控制该控件的结束周期。

Multilanguage：多语言支持(选择)

Permissions：权限管理

Comments：注释的内容可以帮助开发者理解该控件的具体业务含义代替开发文档。

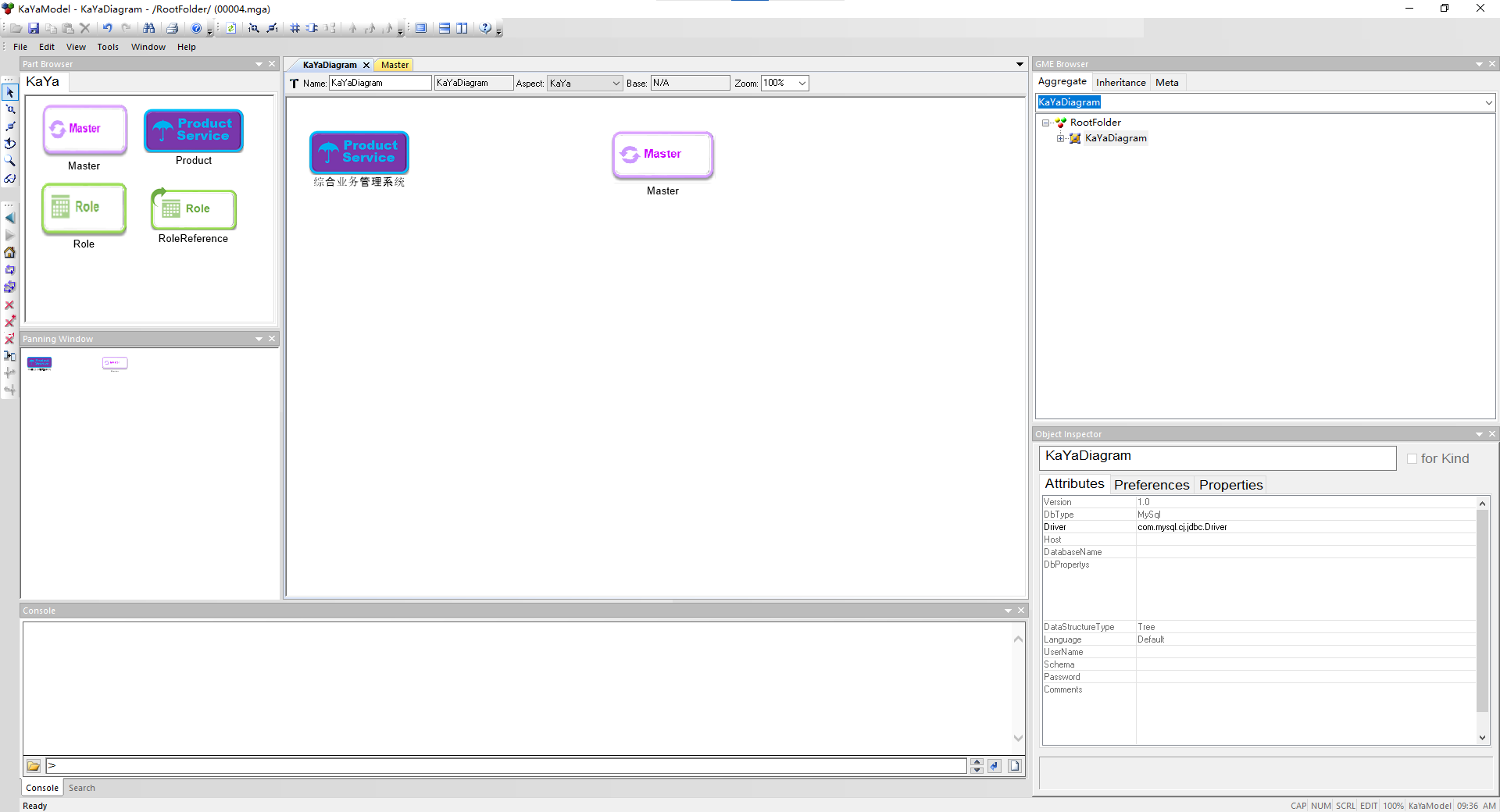
修改信息如下：



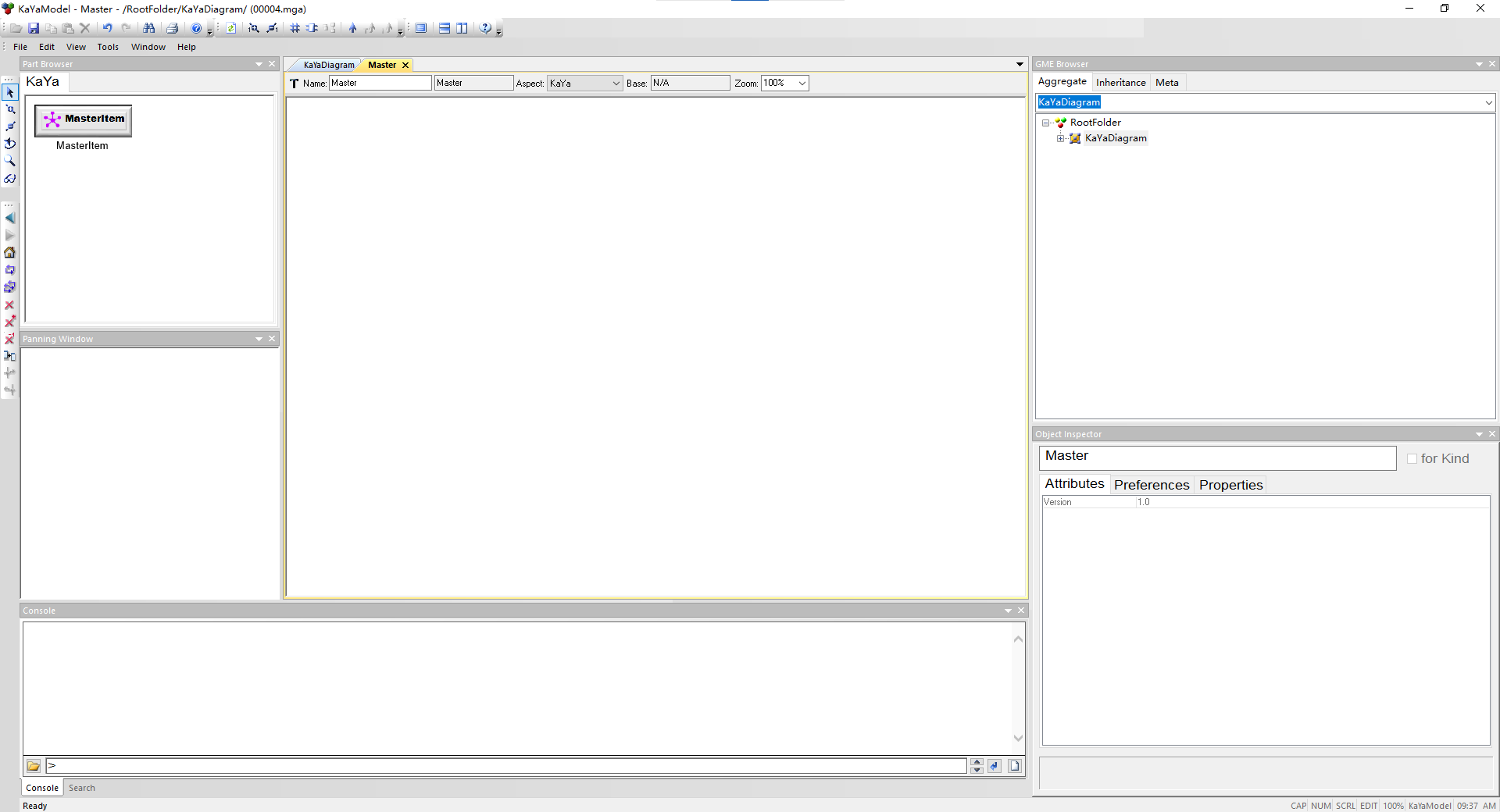
我们用员工管理系统为例，像该系统添加员工角色。但是我们需要为我们的人员添加一些属性来方便我们对员工进行管理。

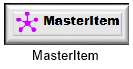
选中Master控件，然后按住鼠标左键拖拽左边的Master控件到主编辑区。放开鼠标左

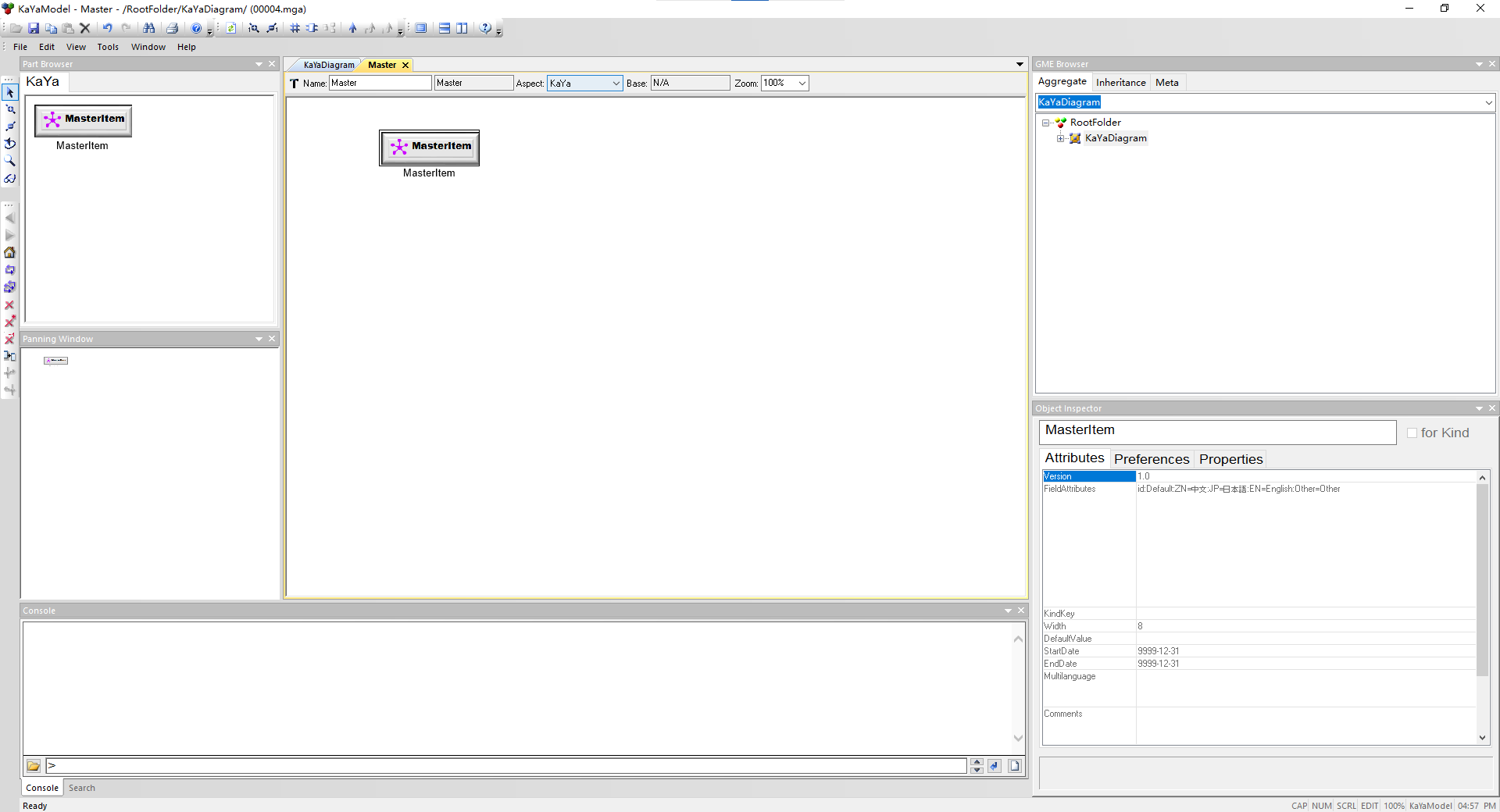
键。我们看到在主编辑区追加了图标。看起来像下图的样子：

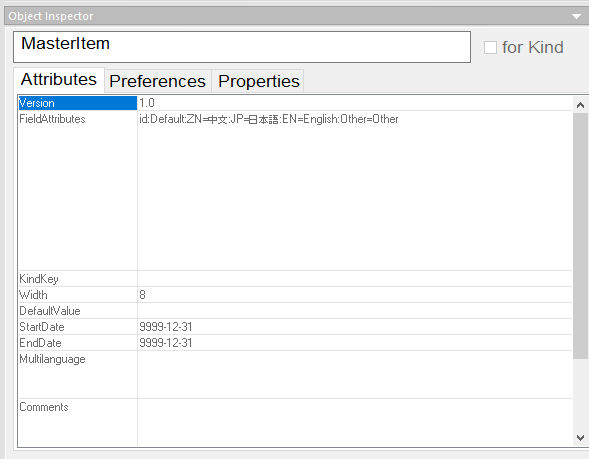
****

双击粉色图标：



拖拽左侧的图标到主作业区域，选中该元素，我们关注一下红色线标注的区域。





控件名称：标签表示信息

Version：1.0（默认）

KindKey：Maste关键字

FieldAttributes：包含的元素信息（UI属性）

DefaultValue:：默认字段。

StartDate: 2018-12-20该控件生效

EndDate：9999-12-31 永久有效

Multilanguate：多语言支持

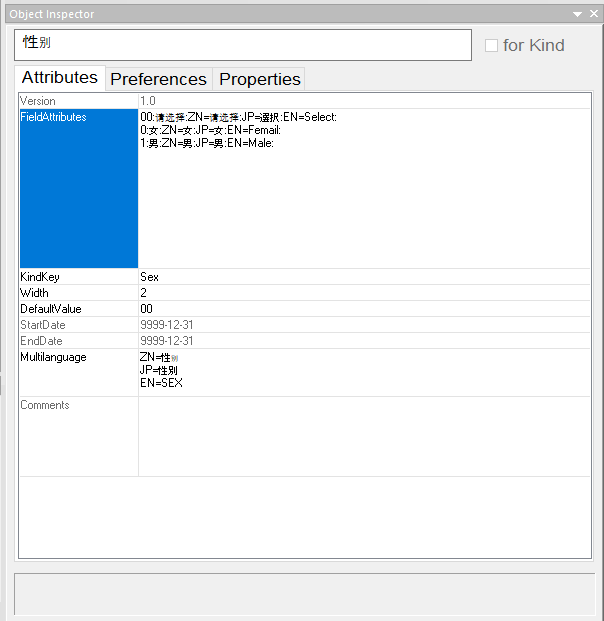
格式为： ZH=中文

其他语言换行后输入（键值请参照P7

**与Language中的选择项保持一致**）

Comments： 说明信息，默认为空

我们修改信息如下：



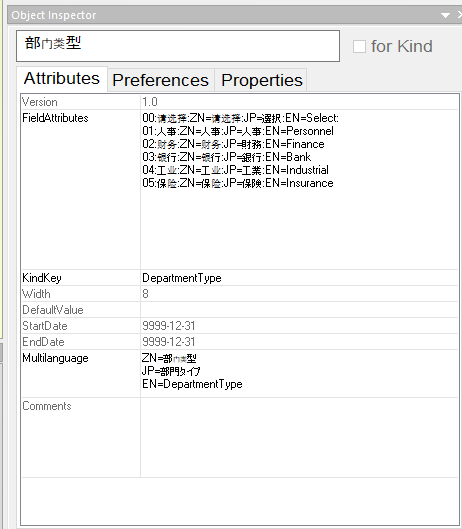
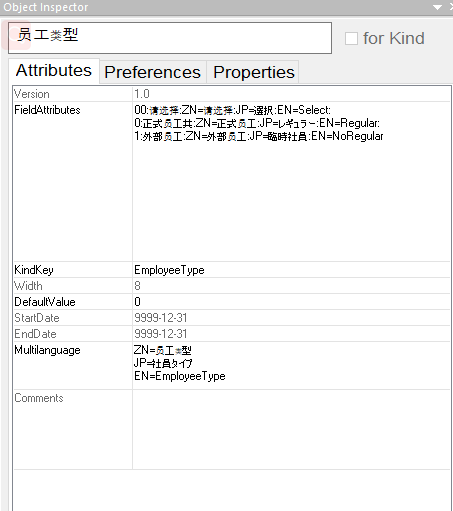
注意FieldAttributes的编辑格式：

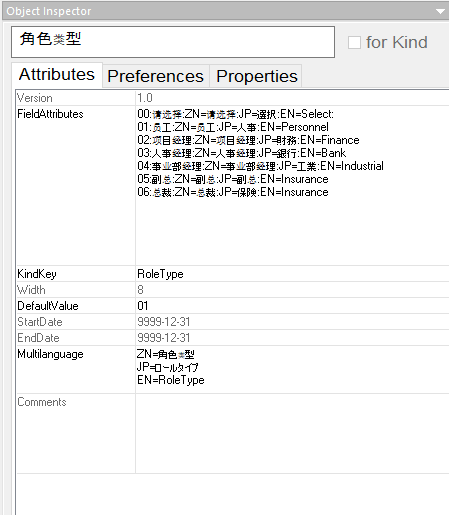
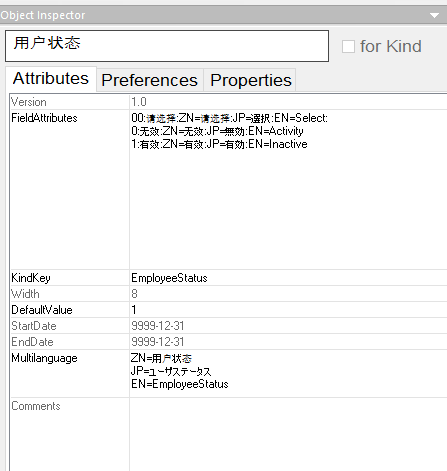
Id:DefaultValue:ZH=中文:JP=日文:EN=English:Other=Other

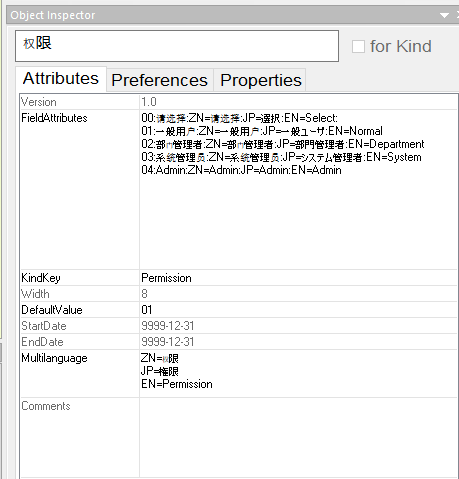
各个语言间用半角’:’隔开

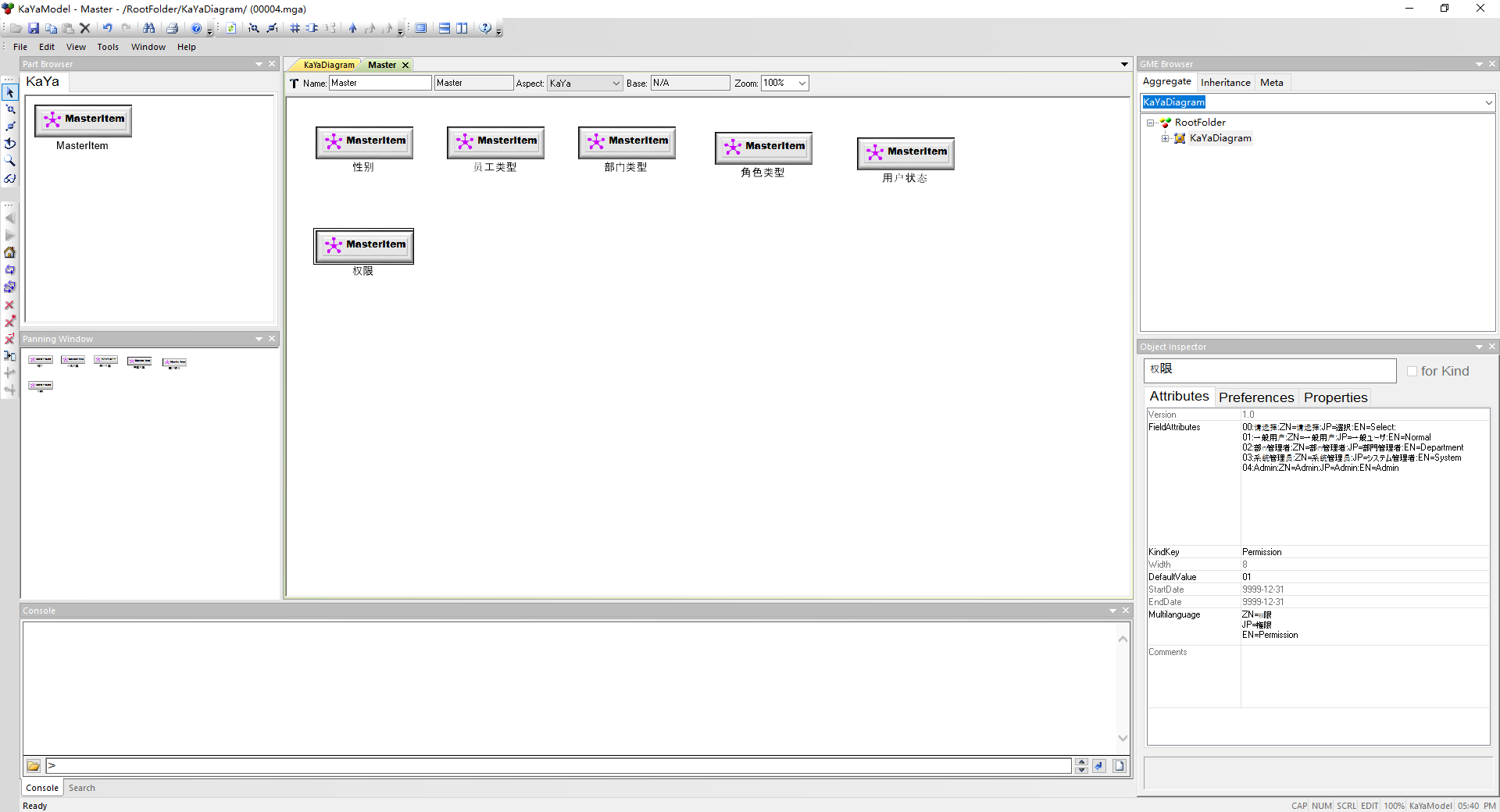
‘’

我们再追加其他一些Master信息：员工类型，部门类型，角色类型，用户状态，权限。

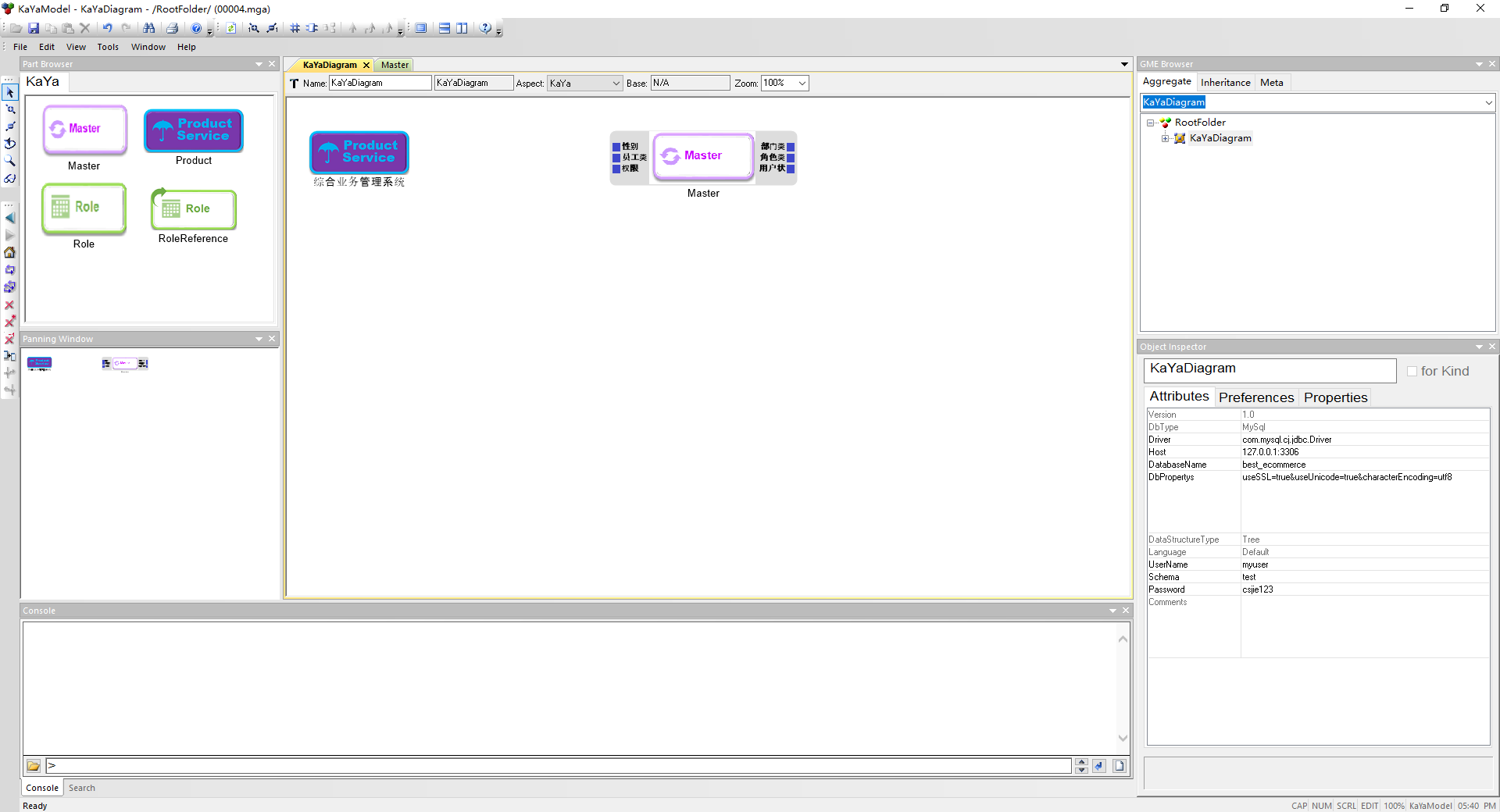






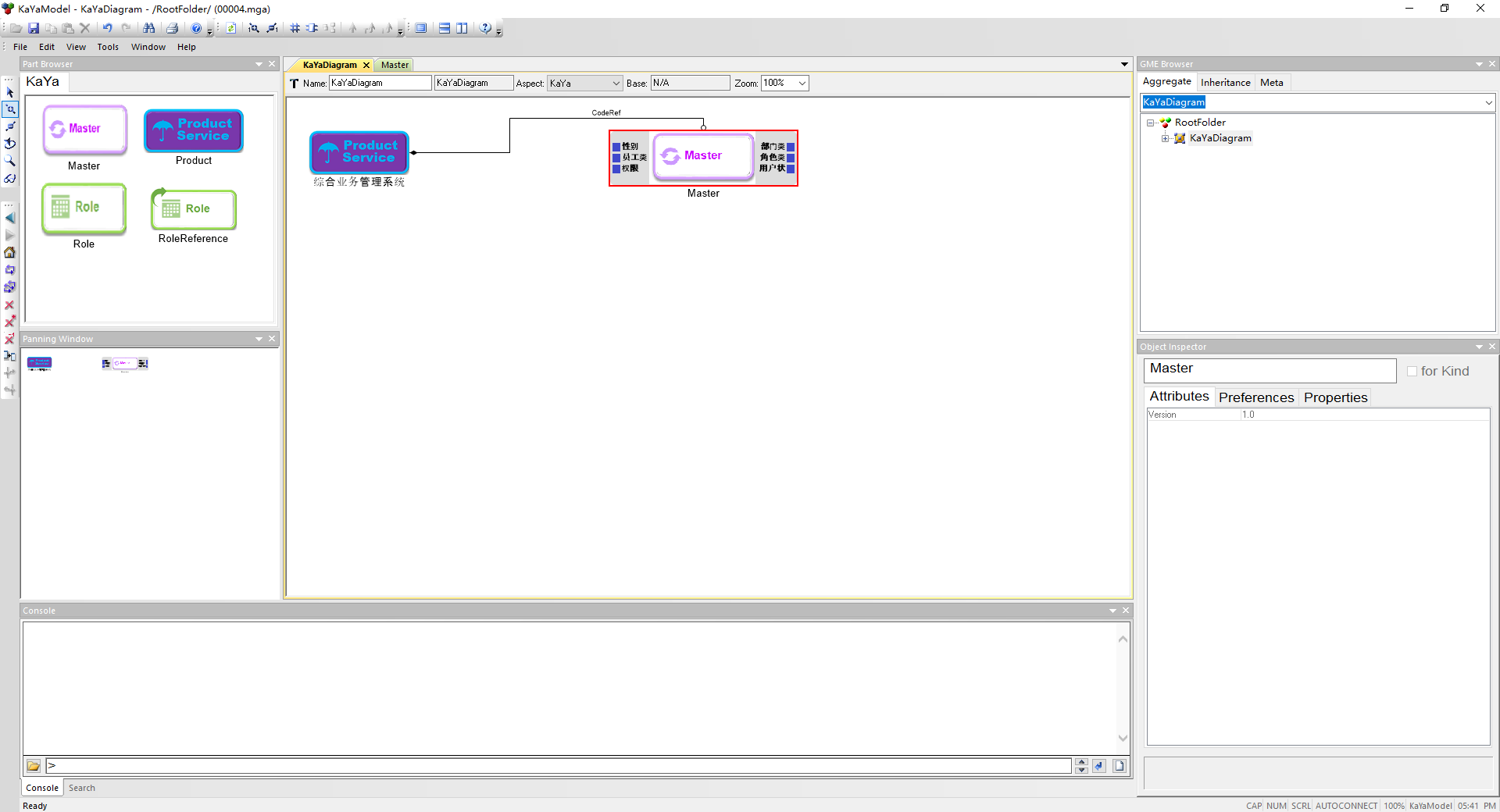


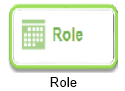
点击菜单栏：

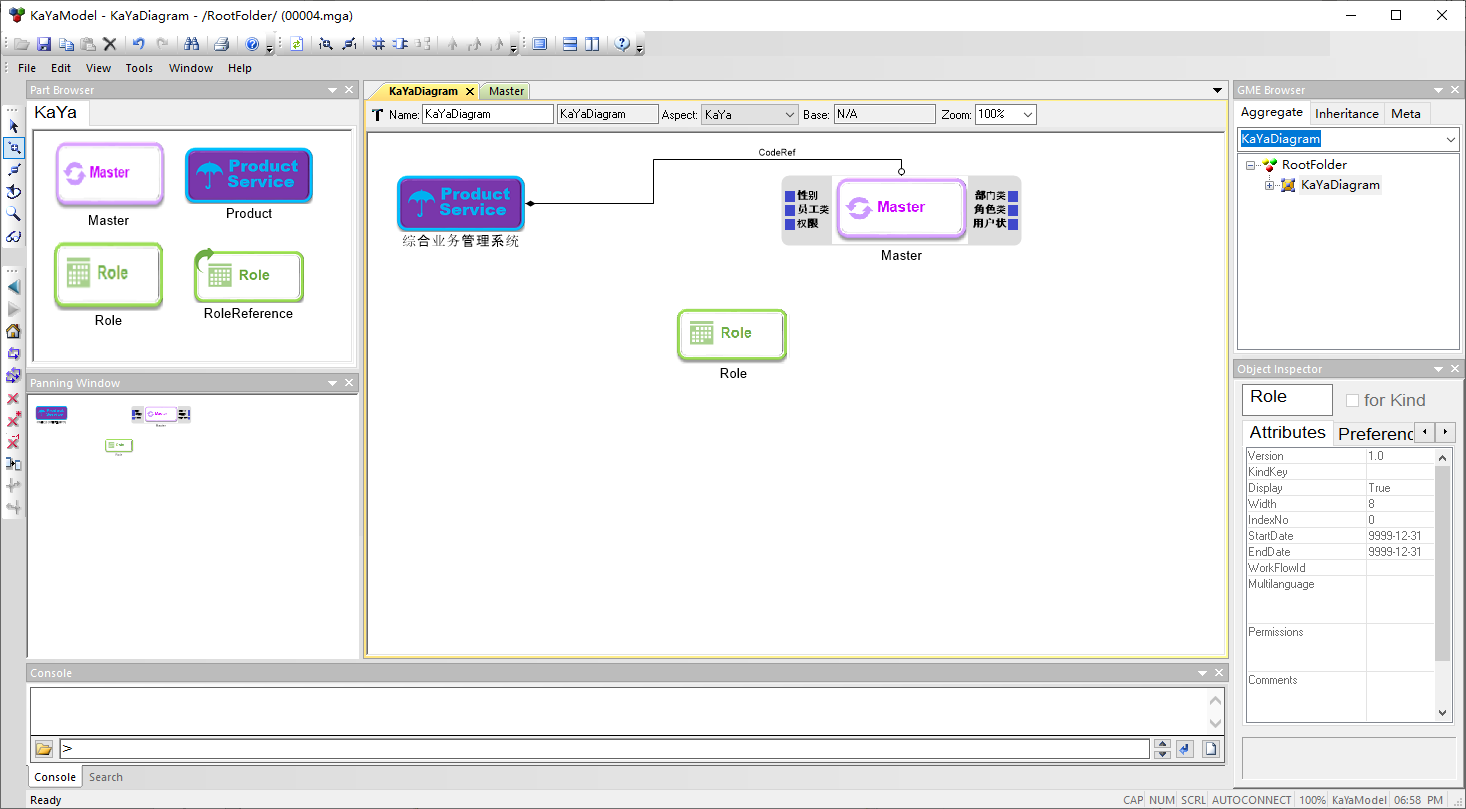


我们来建立系统与角色之间的关系。

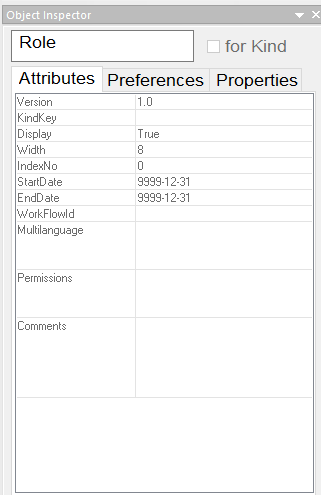
鼠标左键点击左侧连线区的图标（红色区域）。然后将鼠标指针移动到【员工信息】控件上，周围变成红色，如下图的时候，点击鼠标左键，然后再将鼠标指针移动到【综合业务管理系统】控件右半部，当其同样变成红色外框时，再次点击鼠标左键。这样我们就为该角色和系统创建了彼此的关系。（连线的位置可以指定分为上，下，左，右，就是亮起红色方块的位置为连线的起点终点）



向该系统添加员工角色。选中Role控件，然后按住鼠标左键拖拽左边的Role控件到主编辑区。放开鼠标左键。我们看到在主编辑区追加了图标。看起来像下图的样子：



在主作业区域选中鼠标左键点击Role图标，右下角的属性编辑区域我们来编辑该控件的属性。



Version：版本管理

KindKey：关键字，用来识别该控件的唯一标识。

Display: 是否显示（UI属性）

Width: 控件宽度（UI属性）

IndexNo：可以作为前台排列该控件的顺序信息。

StartDate: 用来控制该控件的开始周期。（默认值为9999-12-31 ，代表无效，其他值视具体业务来规定，下同）

EndDate：用来控制该控件的结束周期。

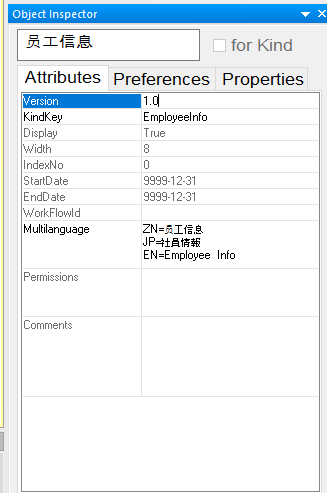
WorkFlowId： 业务流程绑定

Multilanguage：多语言支持

Permissions：访问权限

Comments：注释的内容可以帮助开发者理解该控件的具体业务含义。

我们为该控件编辑属性信息如下：

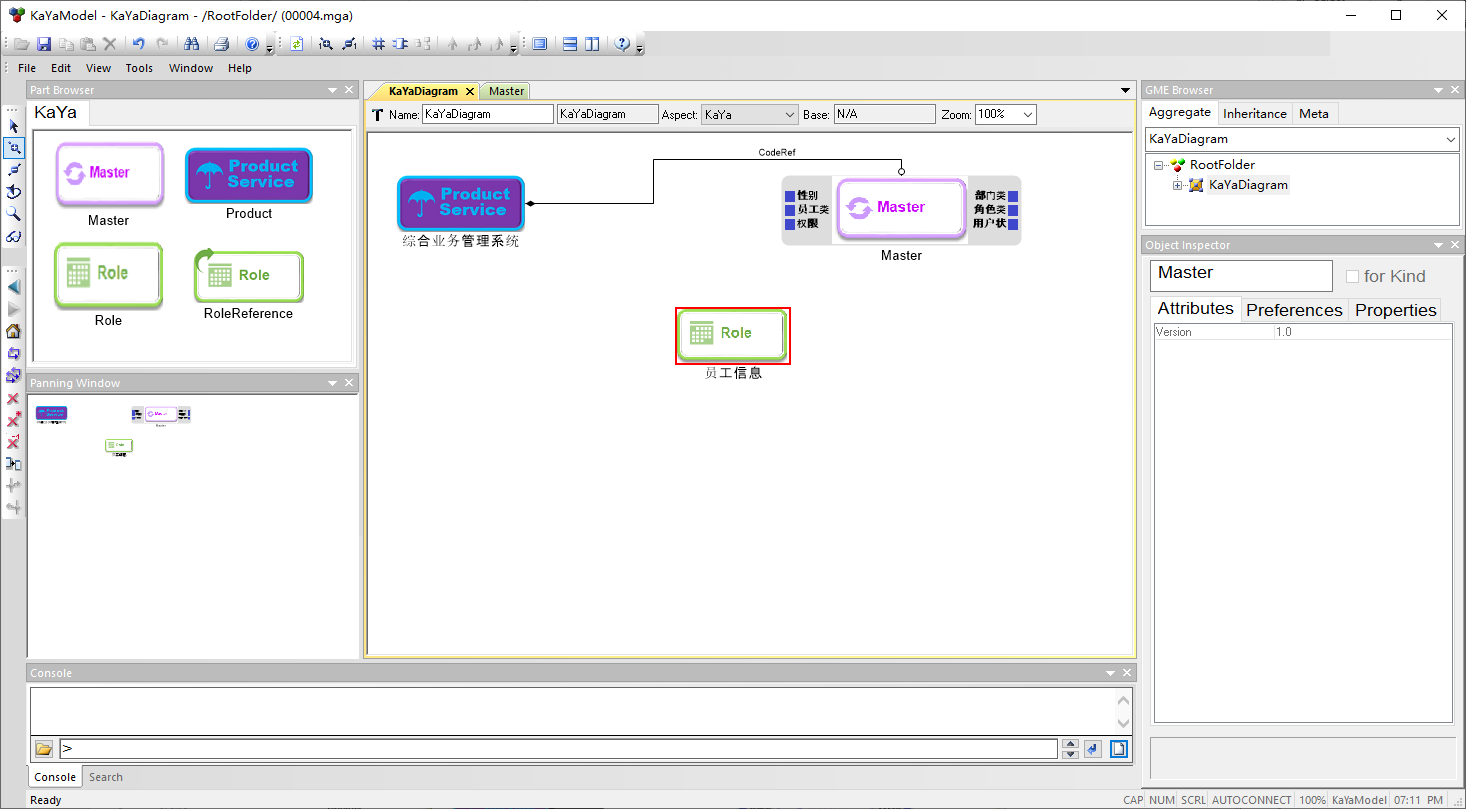


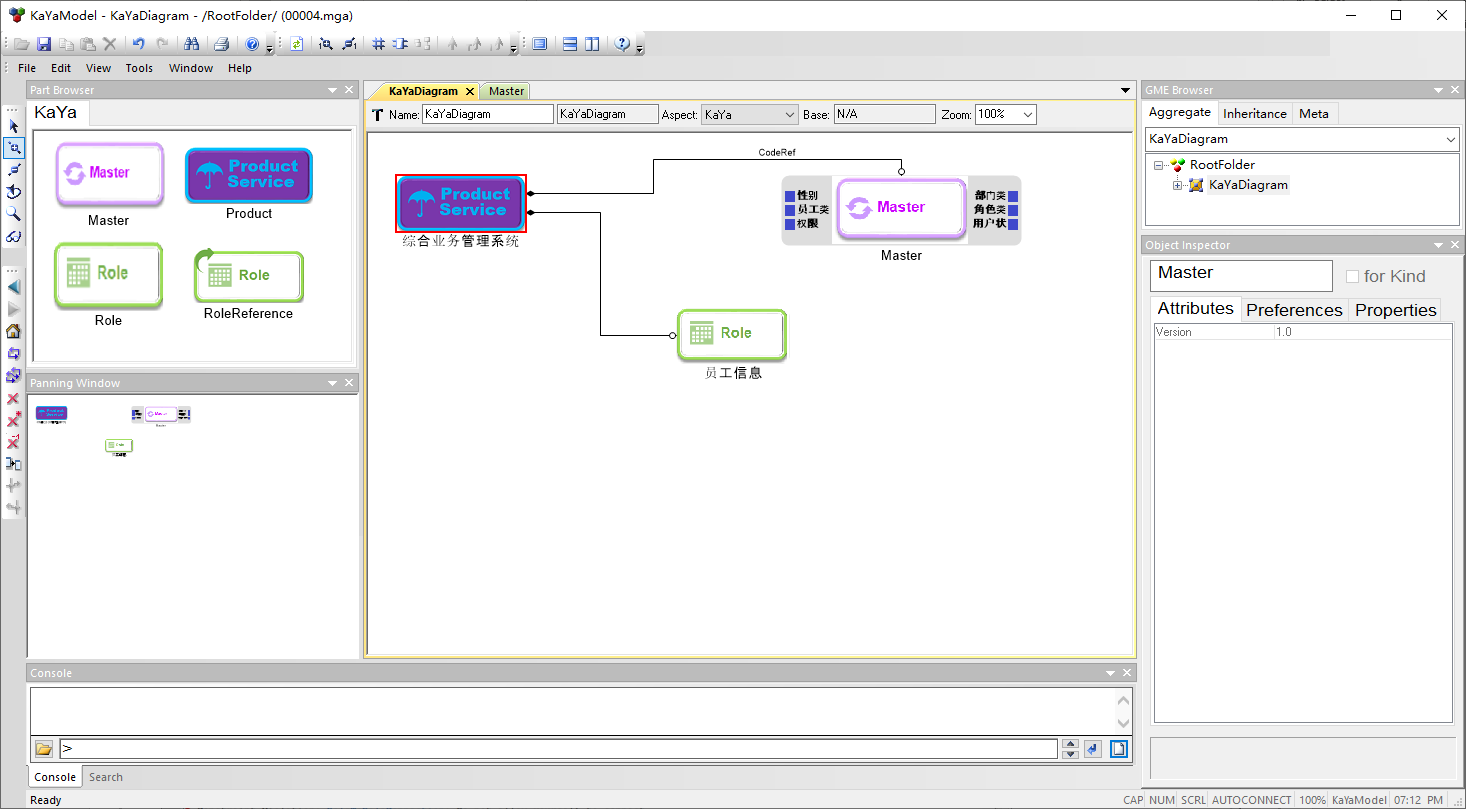
关于WorkFlowId我们会在创建业务流程的时候再讲，现在置空就可以。

Permissions我们最后会统一调整，还需要我们完善员工信息后才能够完成。

我们来建立系统与角色之间的关系。

鼠标左键点击左侧连线区的图标（红色区域）。然后将鼠标指针移动到员工信息控件左半部，变成下图的时候，点击鼠标左键，然后再将鼠标指针移动到综合业务管理系统控件右半部，当其同样变成红色外框时，再次点击鼠标左键。这样我们就为该角色和系统创建了彼此的关系。（连线的位置可以指定，就是亮起红色方块的位置为连线的起点终点）





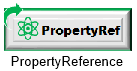
我们在用鼠标左键点击连线区的图标（红色区域），返回到编辑状态，鼠标左键双击员工信息控件。我们的模型工具会变成下面的状态。接下来我们可以为角色-员工信息来添加细节属性。

我们来熟悉一下这个工具的布局。



这个是表中的主键属性（唯一排他）。

这个代表角色的属性。

这个是角色属性的参照。

Master属性的参照

分组信息：例：兴趣（组）：体育，音乐，电影等组里面的信息

Property是整个系统的原子属性，具有唯一性（统一表内，属性的Kind Key具有排他性）。让我们来看一下它具体的一些参数。同样选中左侧的控件后，左键按住，拖拽控件到主作业区域。



同样，在主作业区域，鼠标左键选中该控件。我们在右下角可以看到该控件的属性。



Version：版本信息

KindKey：关键字，用来识别该控件的唯一标识。

Format：字段形式，用于校验（正则表达式形式）DataType：对应具体的Java数据类型（不同于数据库类型）。例：[/^[0-9]\*$/]:[请输入小于2位数字]

Dsplay：该控件的显示与否控制开关（需要自己前台实现）

ComponentType:组件类型（根据前端UI的控件类型手动填写）

Width: 控件的宽度

DefaultValue：默认值

DataLength：数据长度（存储长度）

IsSearchkey: 是否作为检索项目，默认为False：不作为检索项目

DataType: 数据类型Auto,String,byte,int,short,long,char,boolean,Date,

Float,double,BigDecimal,Calendar,Integer,Double 其中Auto为自增字段18（利用系统时间生成的非重复18位数字）

Required: 必须填写项，默认为False即非必须输入项

IndexNo: 当该数据在前端显示时，影响其字段的显示顺序。

StartDate: 用来控制该控件的开始周期。（默认值为9999-12-31 ，代表无效，其他值视具体业务来规定，下同）

EndDate：用来控制该控件的结束周期。

Multilanguage: 多语言对应，参照前面的例子

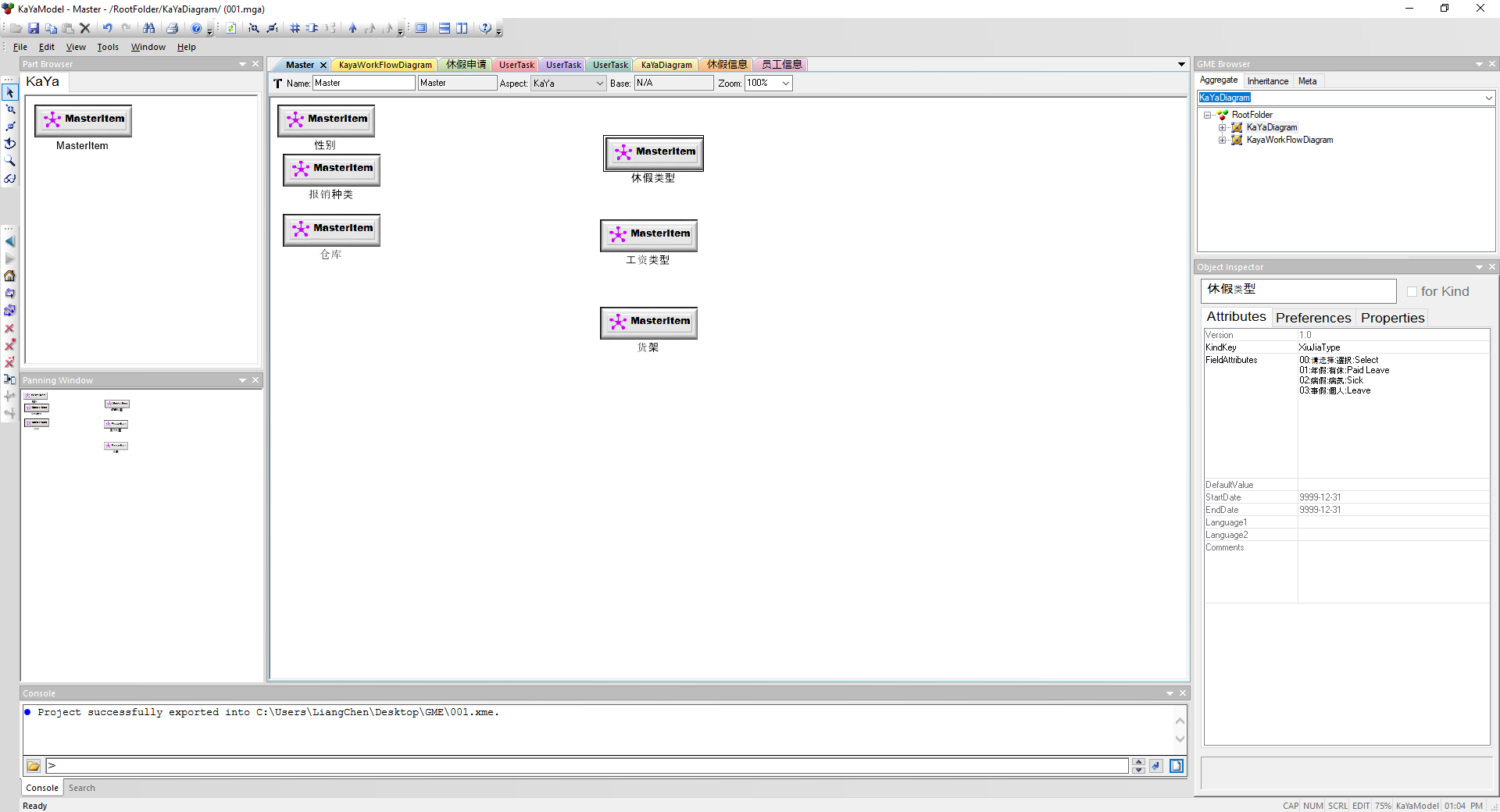
Comments：注释的内容可以帮助开发者理解该控件的具体业务含义。

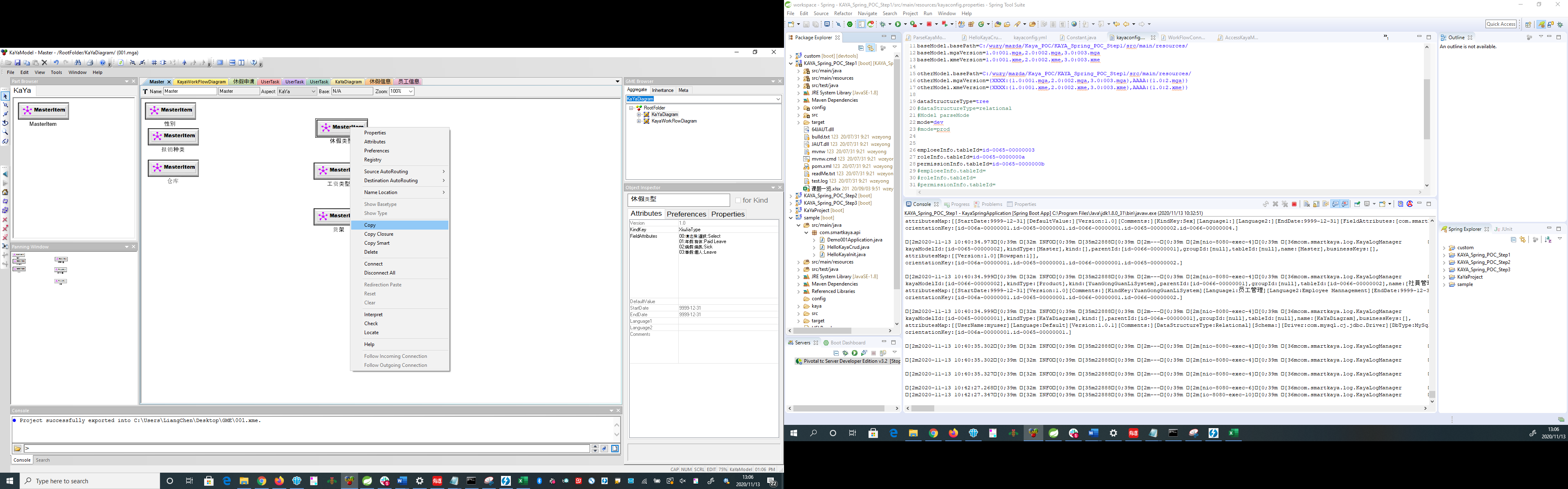
其他元素的属性基本类似，这里不再详述。

其中MasterReference需要到Master里面拷贝相应的属性，按照 参照的方式粘帖过来就可以。

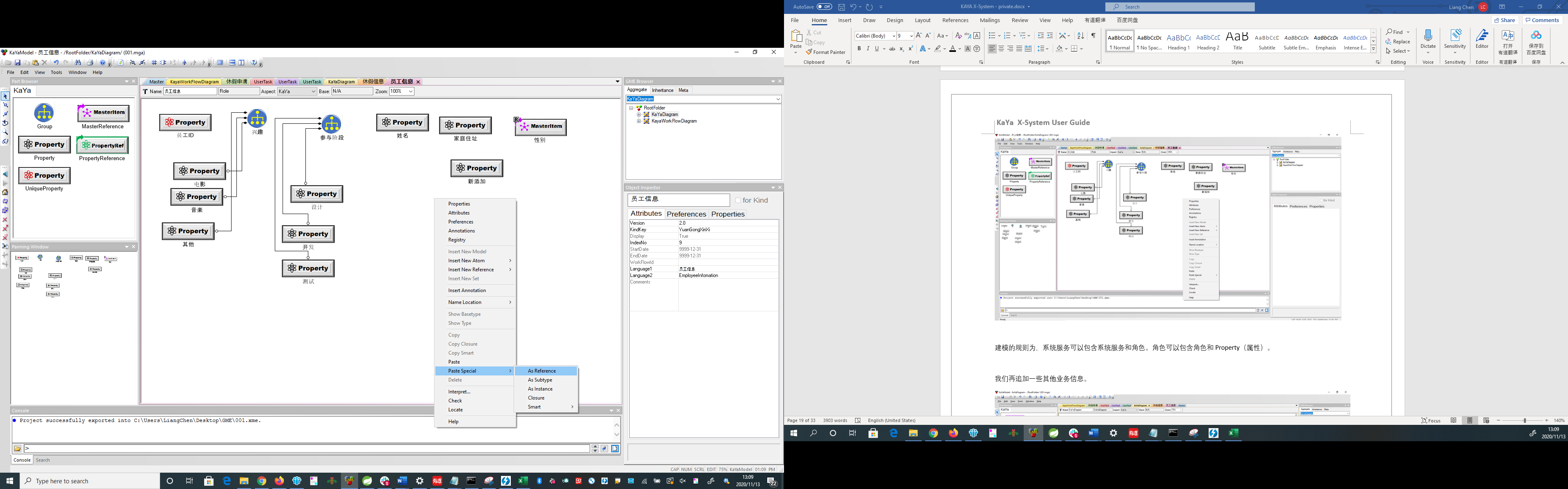
具体操作：

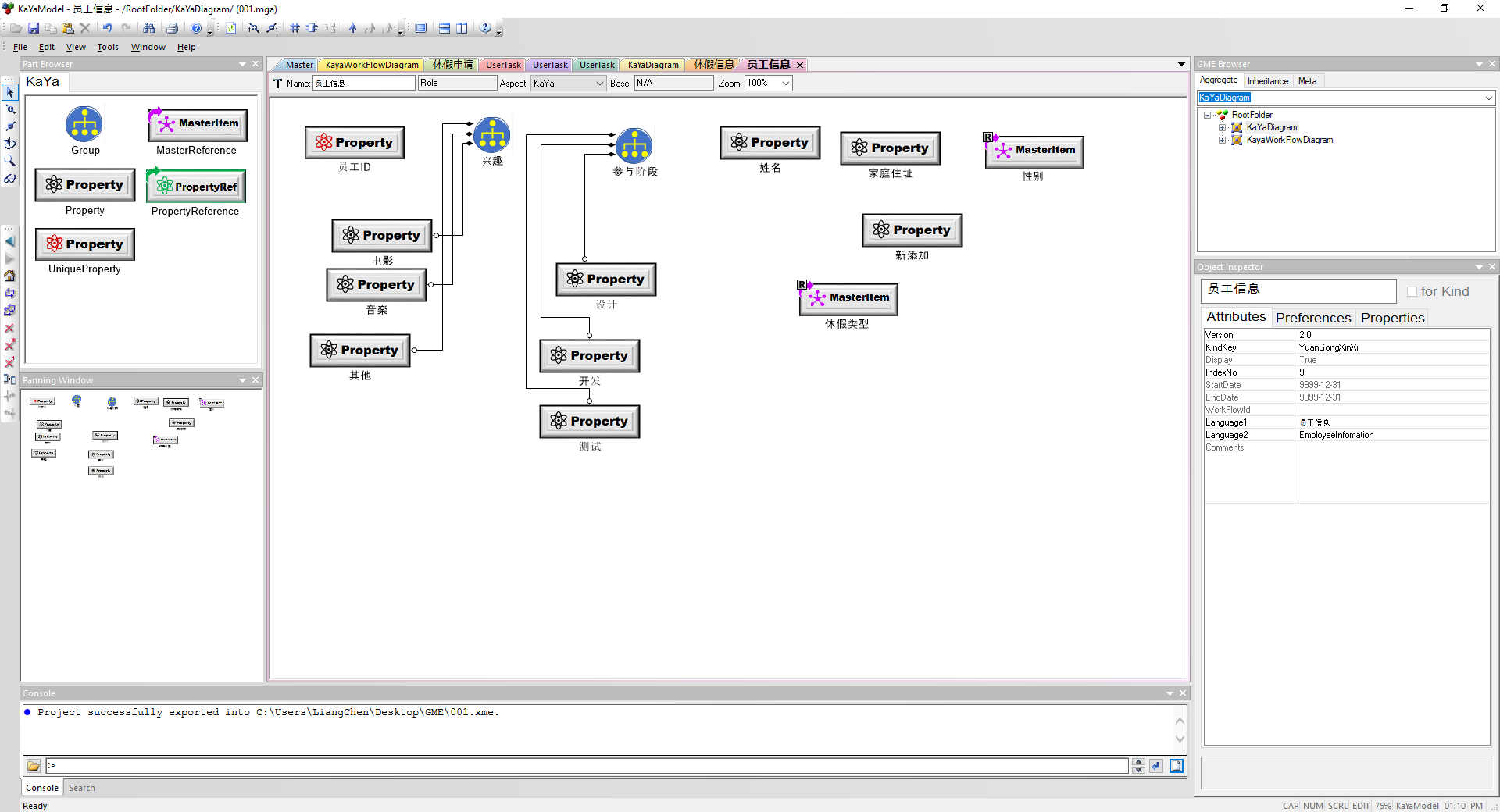
在Master操作区选中对象





在员工信息区域空白处点击右键：Paste Special -> As Reference

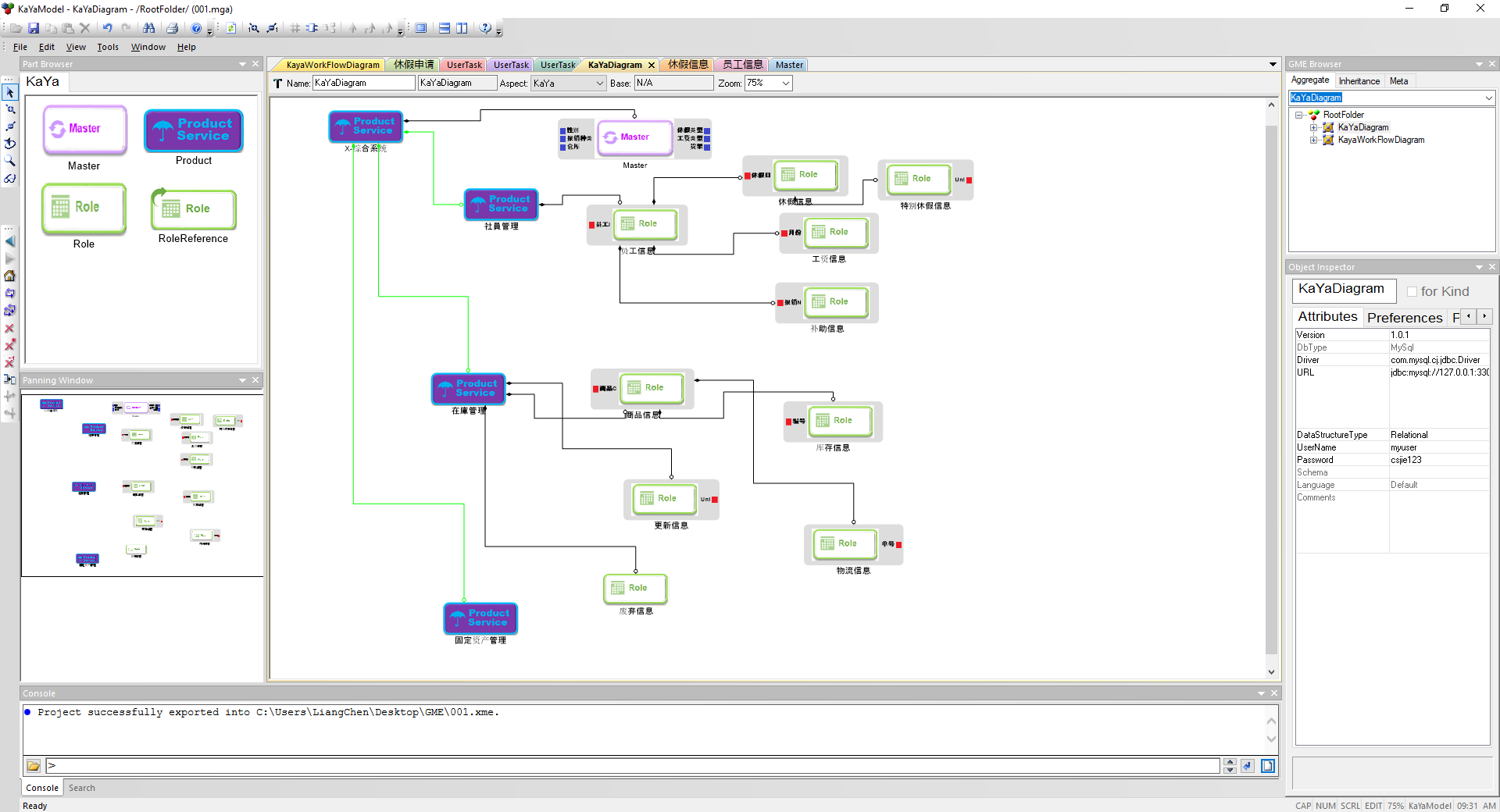




我们再完善好该元素的属性信息后，就完成了Master数据的参照操作，在模型中，我们只要改Master里面的相关信息，那么整个系统的参照就自动被更新了，是不是很方便？

建模的规则为，系统服务可以包含系统服务和角色。角色可以包含角色和Property（属性）。

我们再追加一些其他业务信息。



这样，我们的元数据模型就建好了。

打开菜单File，选择Export XML，选择保存位置后，点击Save按钮。保存模型文件。

在Console窗口会输入如下类似信息：

Project successfully exported into C:\Users\xxx\Desktop\GME\Kaya001.xme.

如果要让我们的模型跑起来，我们需要下面的配置。

1. 安装GME建模软件
2. 按照下面的方式配置文件

Project（项目文件夹，名字任意）

|\_ JAUT.dll

|\_ KAYA.jar

Config

|\_ application.properties

|\_ kayaconfig.properties

其中 kyaconfig.properties 文件配置如下

#Log Path

kayaMetaModelParseLogPath=C:/workspace/KayaModel.log

kayaMetaModelSqlLogPath=C:/workspace/KayaModelSql.log

kayaMetaModelConnectionLogPath=C:/workspace/KayaModelConnection.log

#KayaModelFile path

#Model Path

baseModel.basePath=C:/Users/LiangChen/Desktop/GME/

baseModel.mgaVersion=3.0:003.mga

baseModel.xmeVersion=3.0:003.xme

#Model parseMode

#Window下的开发模式（二进制文件）

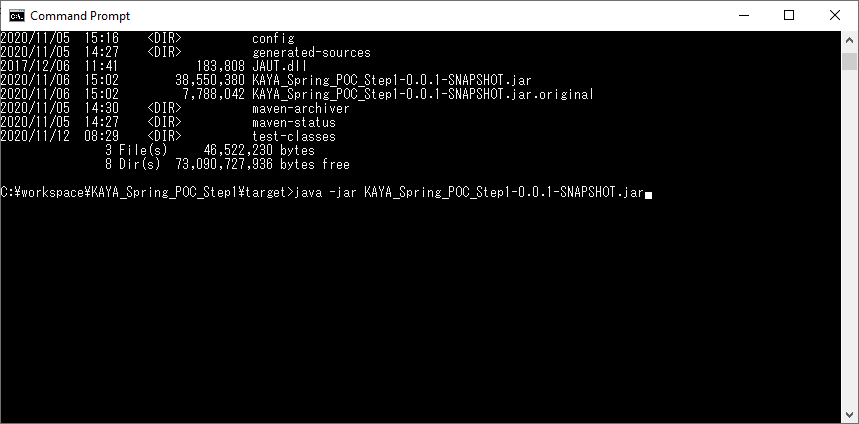
mode=dev

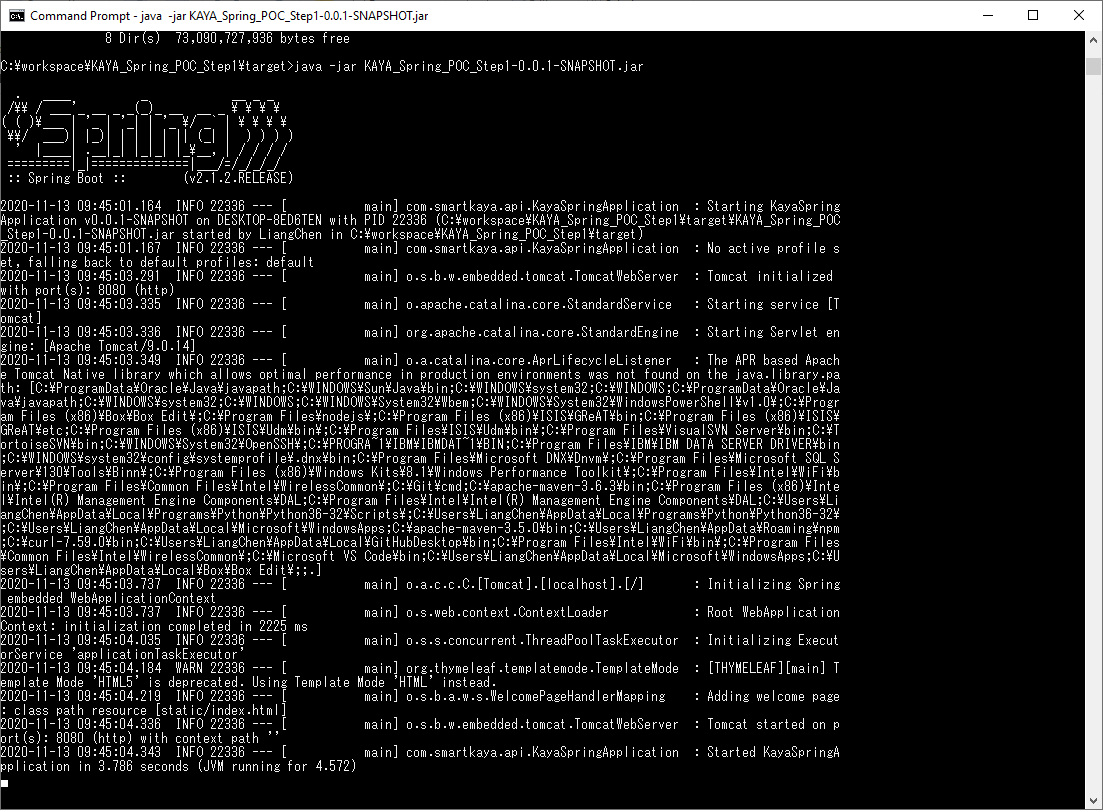
#生产环境 跨平台模式

#mode=prod

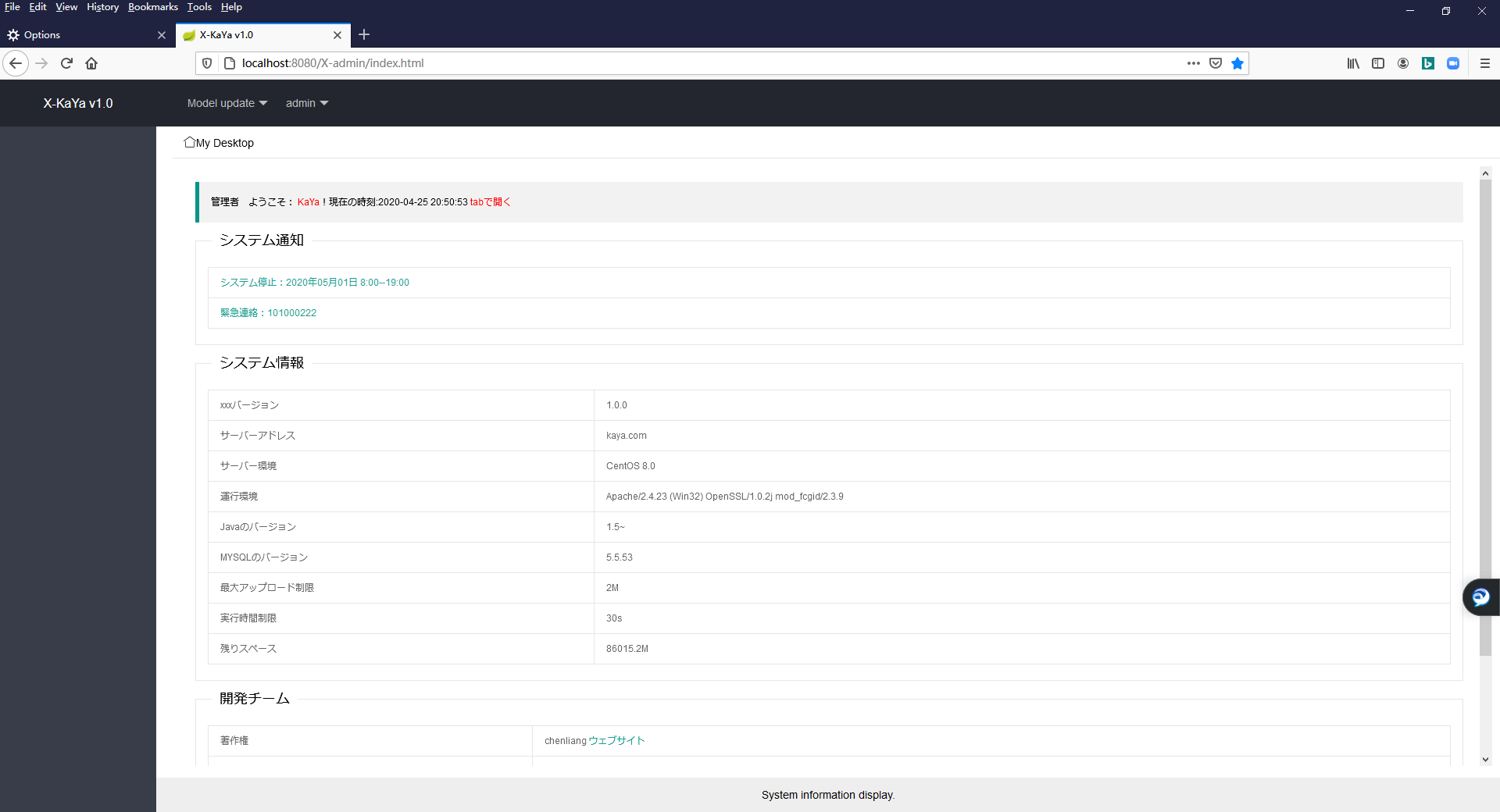
PS: 需要注意的是：我提供的是32位的链接库文件，你需要安装Java 32位的运行时组件。

上面的链接库仅仅开发阶段使用。实际的生产环境我们是不需要的。





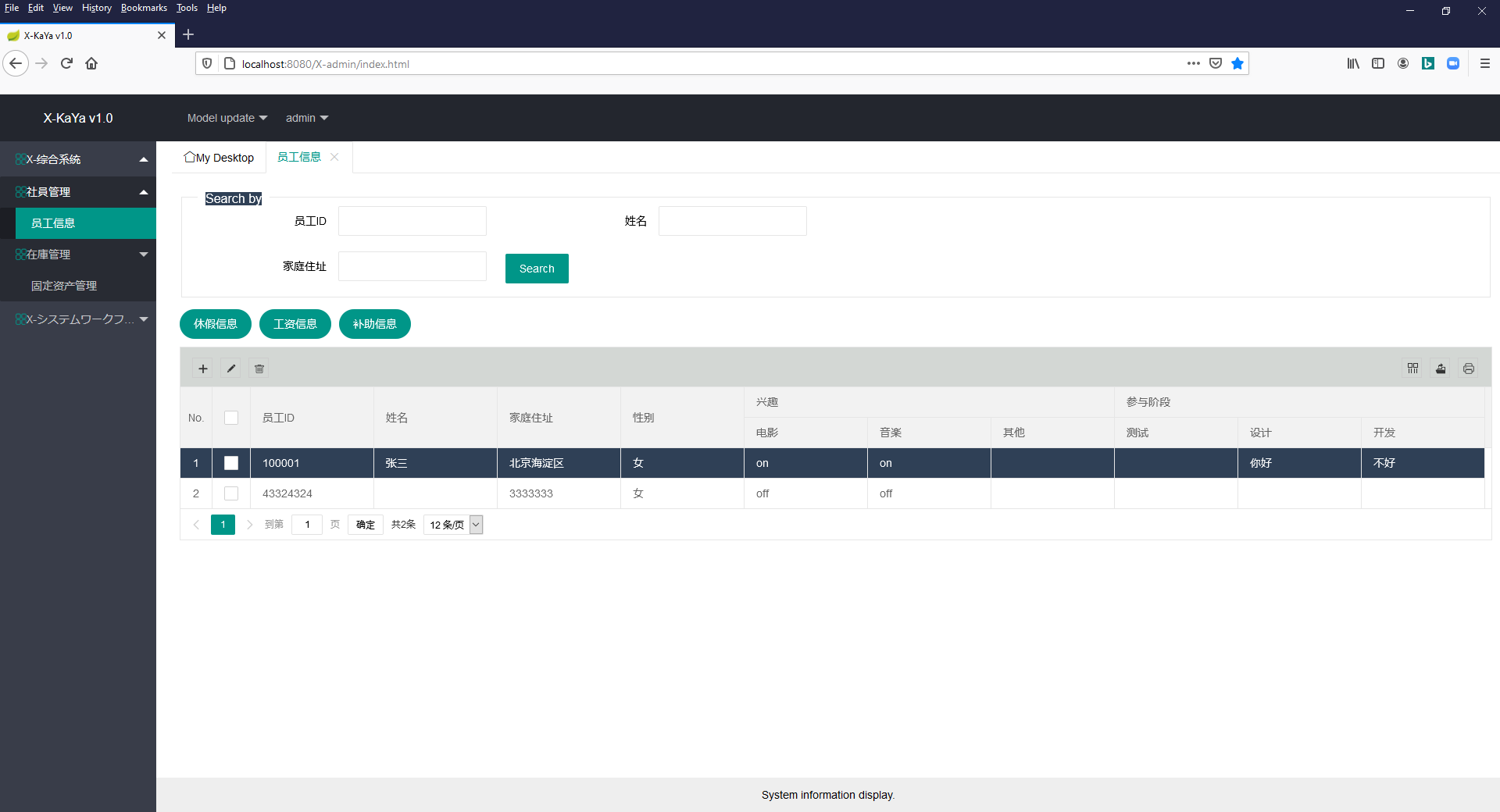
我们通过浏览器访问如下地址：<http://localhost:8080/X-admin/index.html>

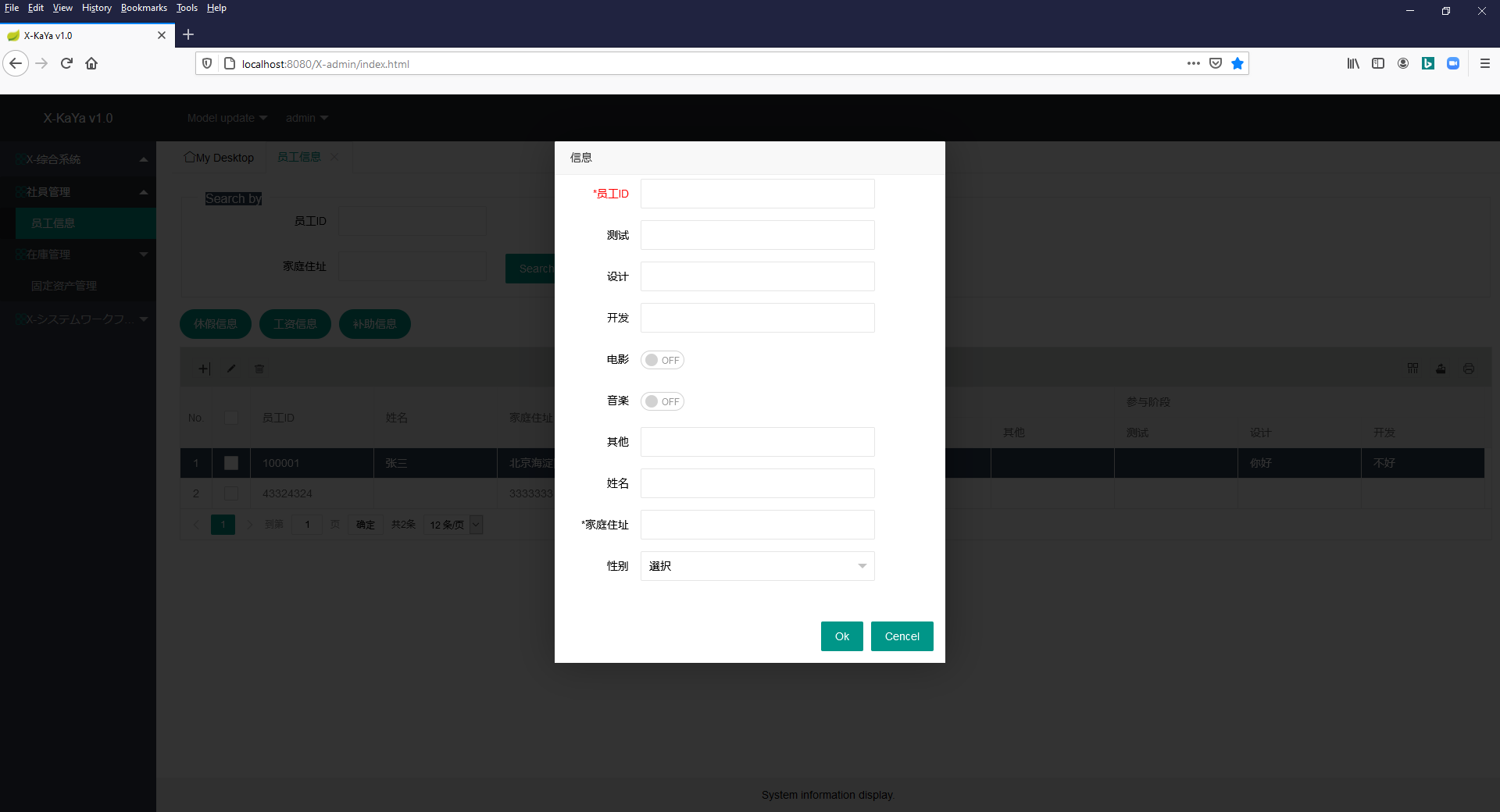


**在页面的Model update 下拉框中选择Load Model**

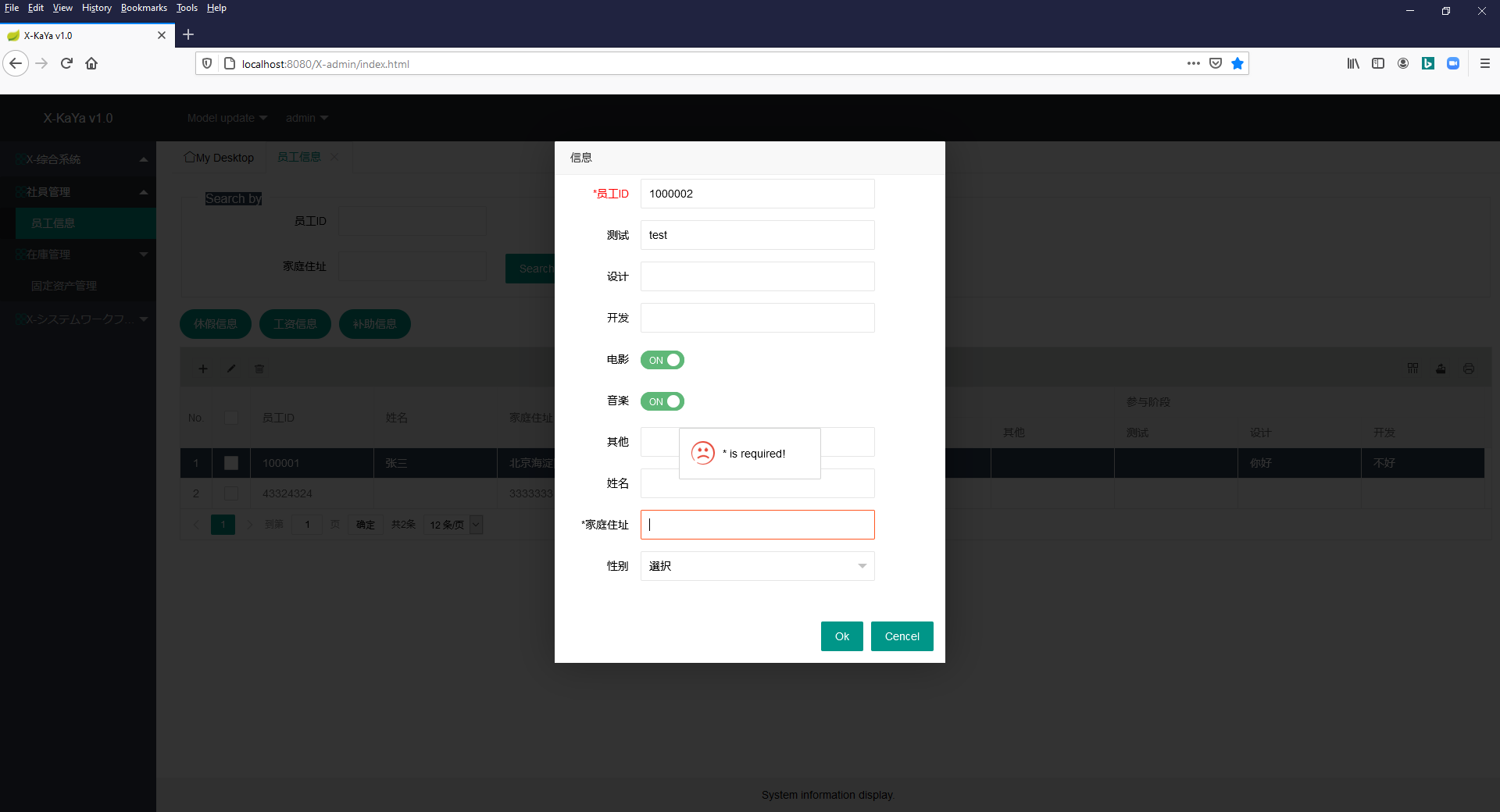




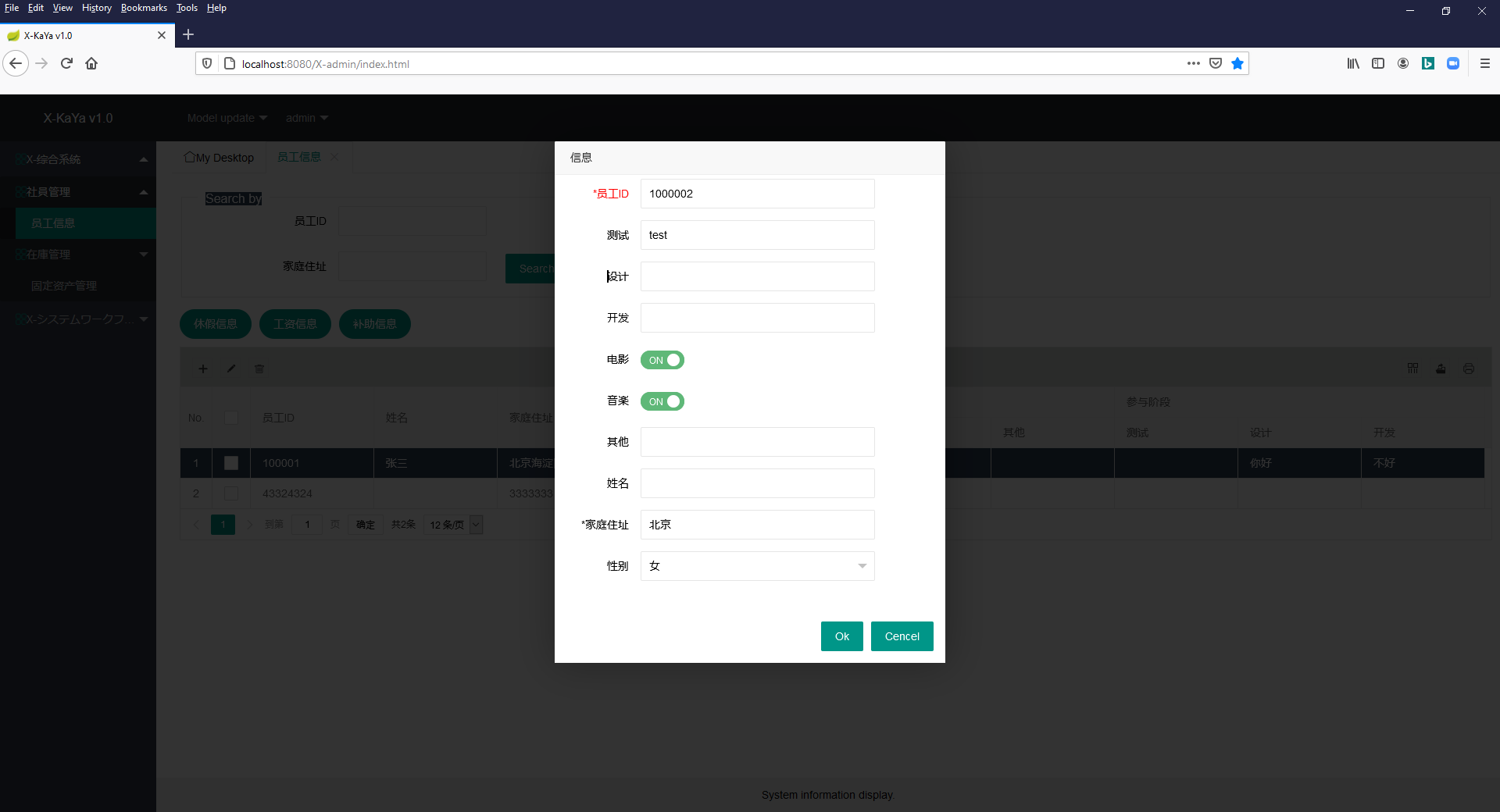


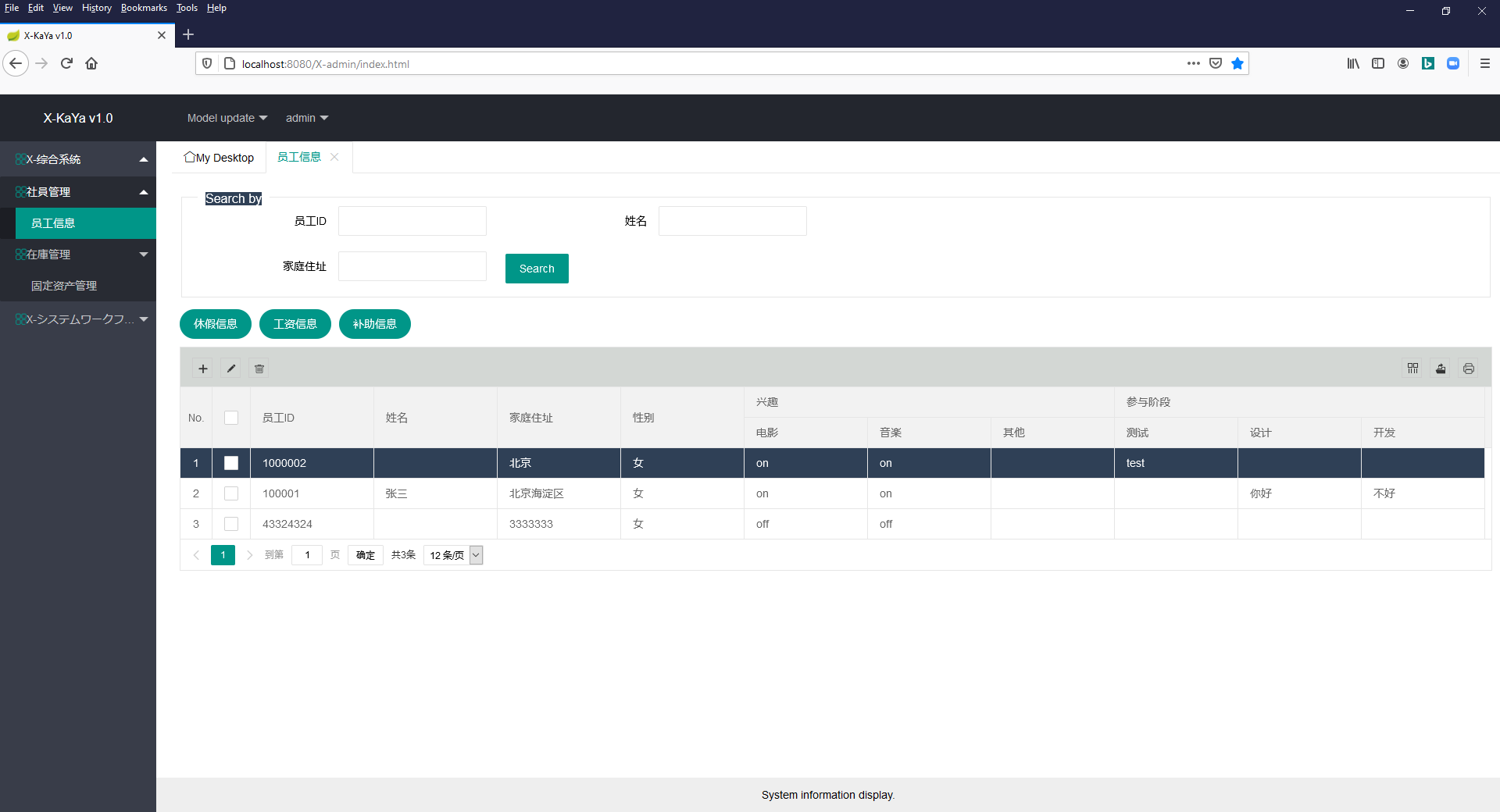


自动验证必须填写项目等

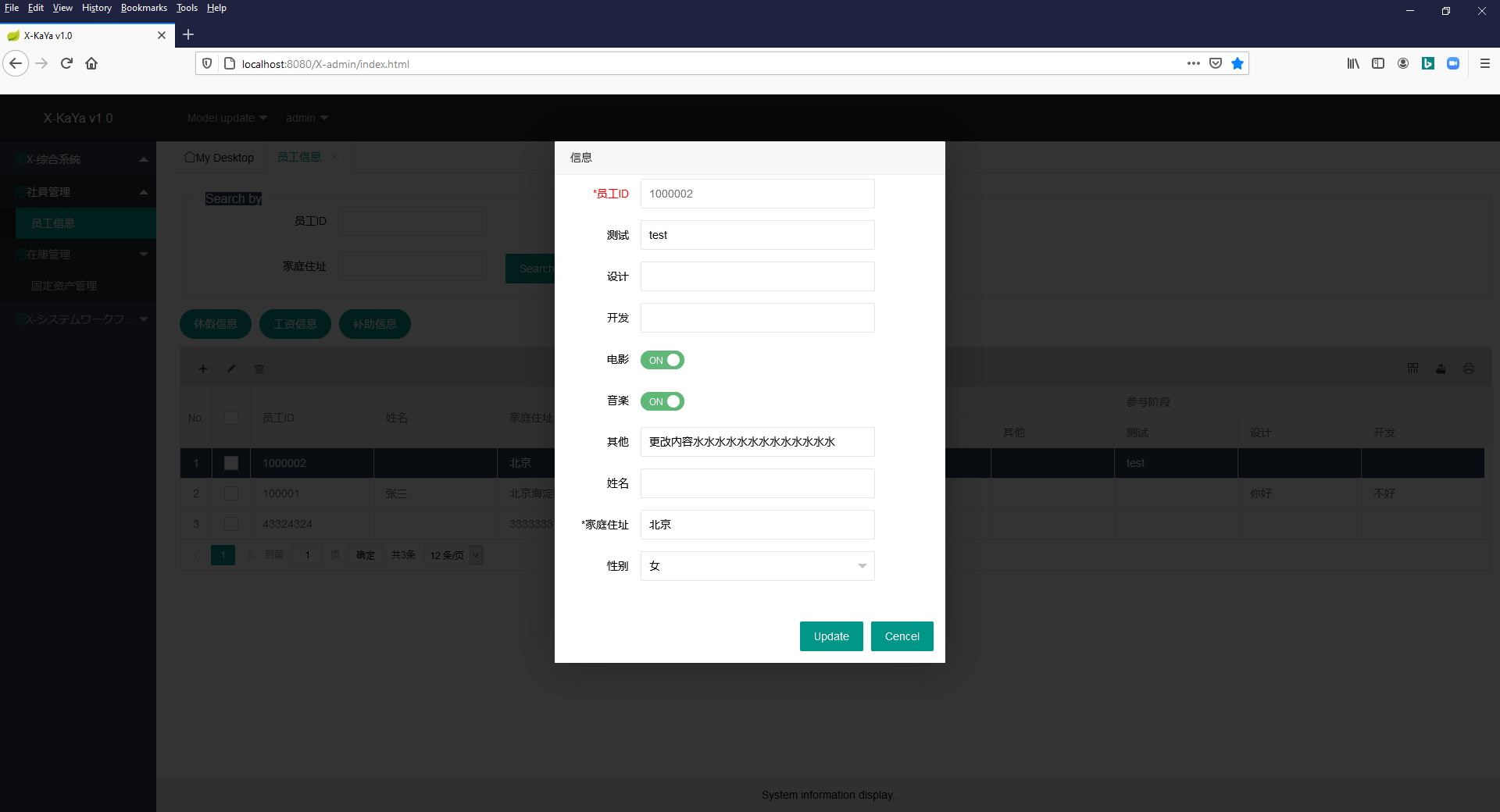


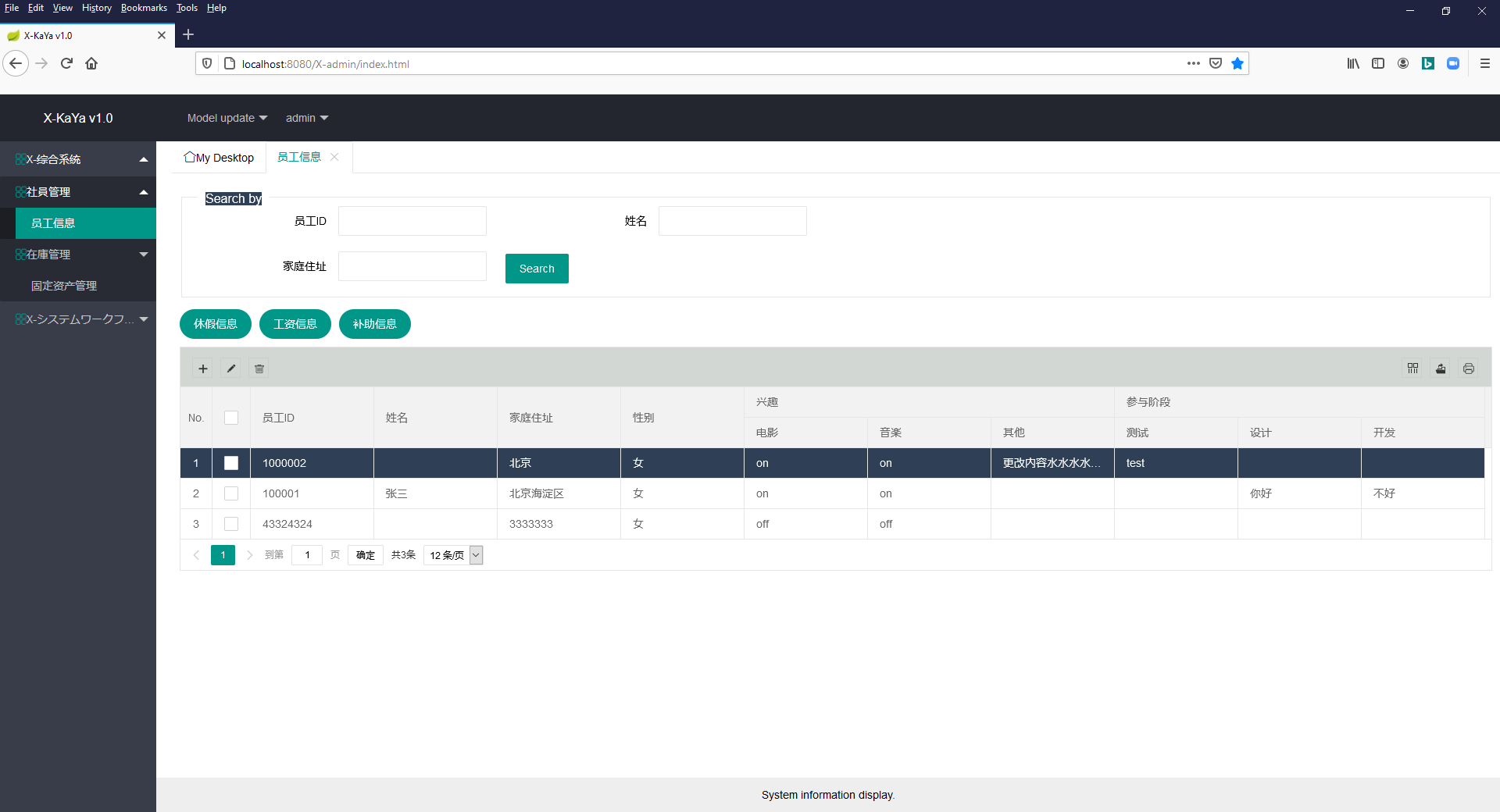
插入数据



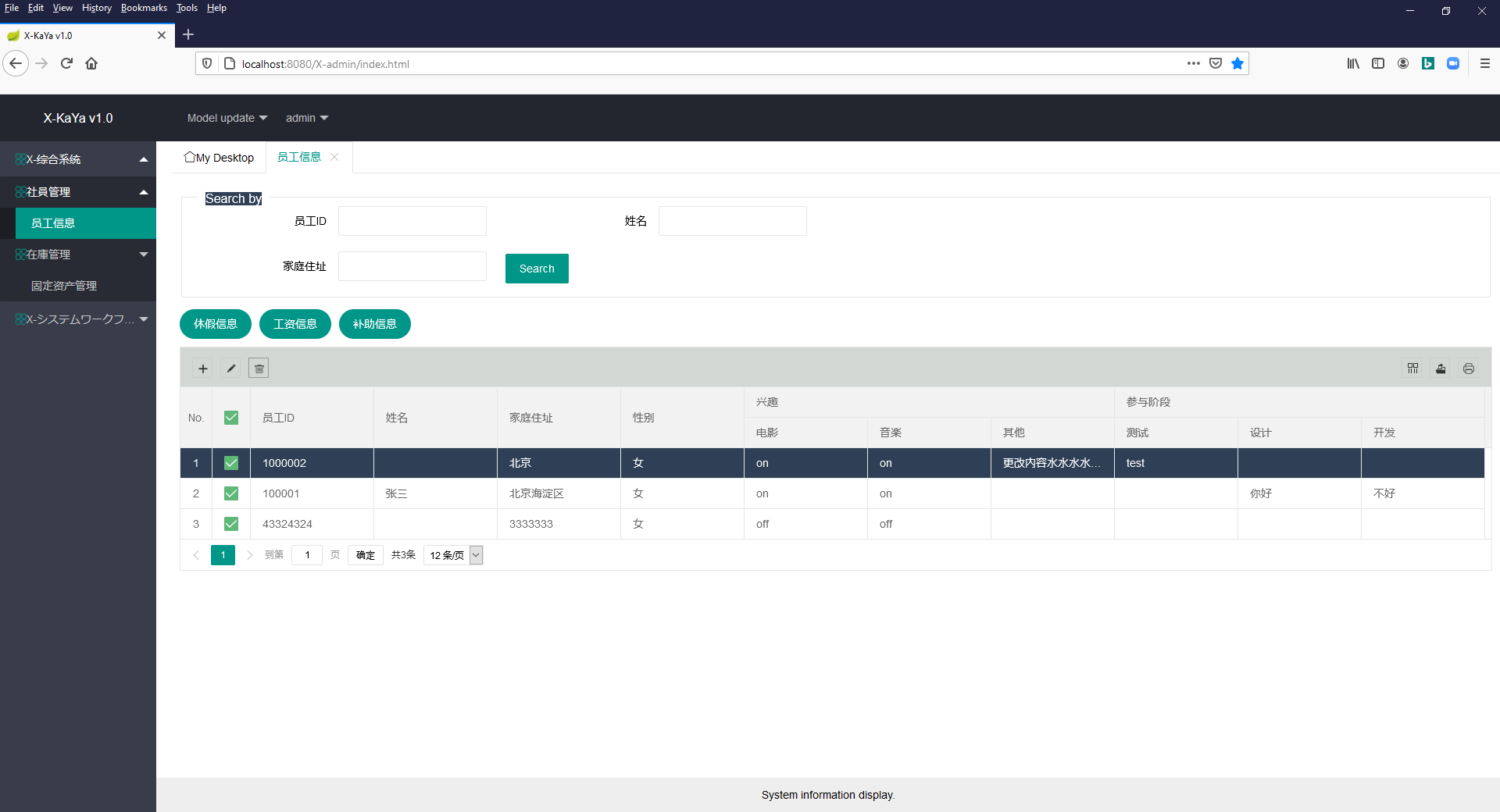


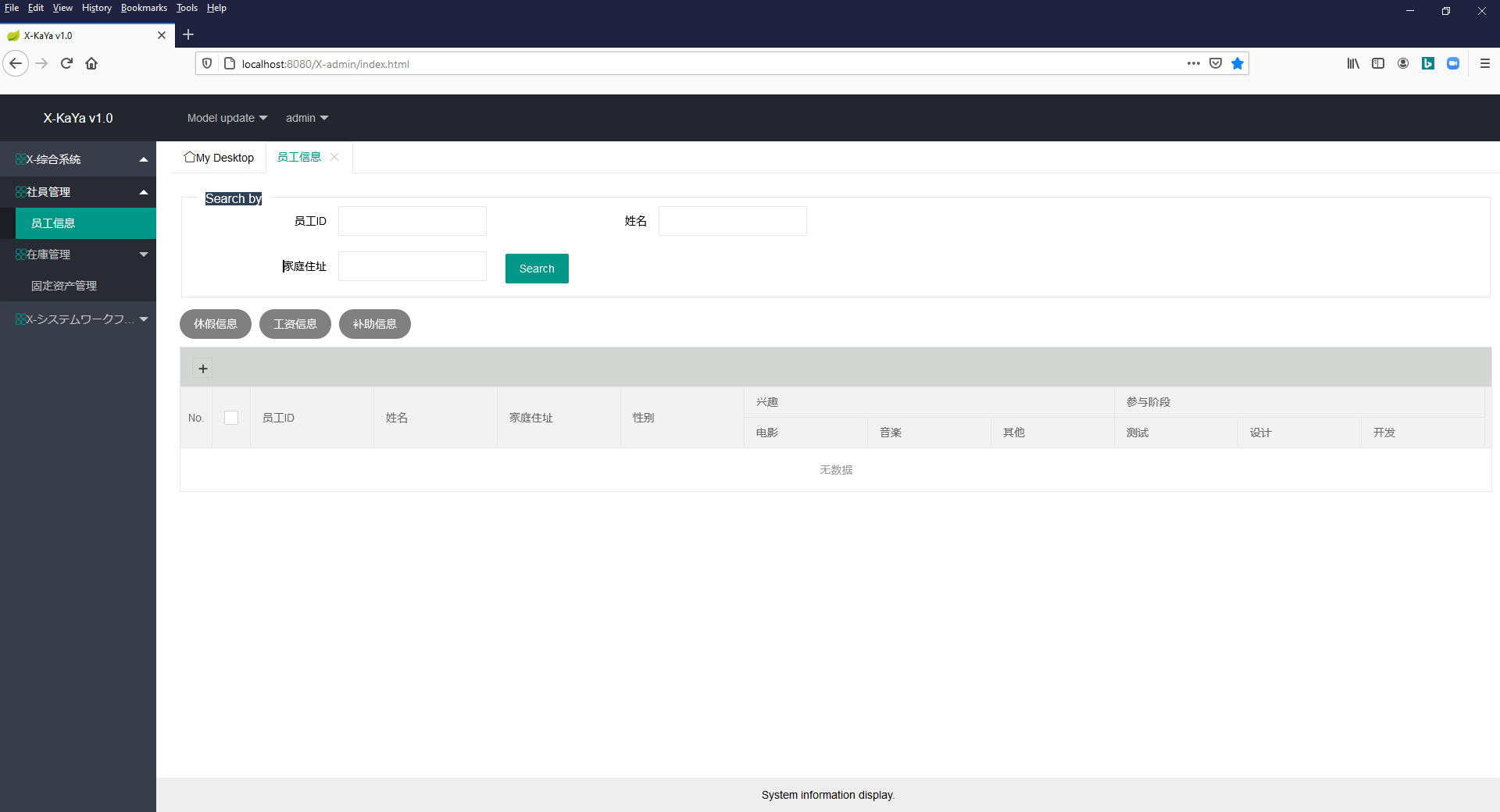
更改数据



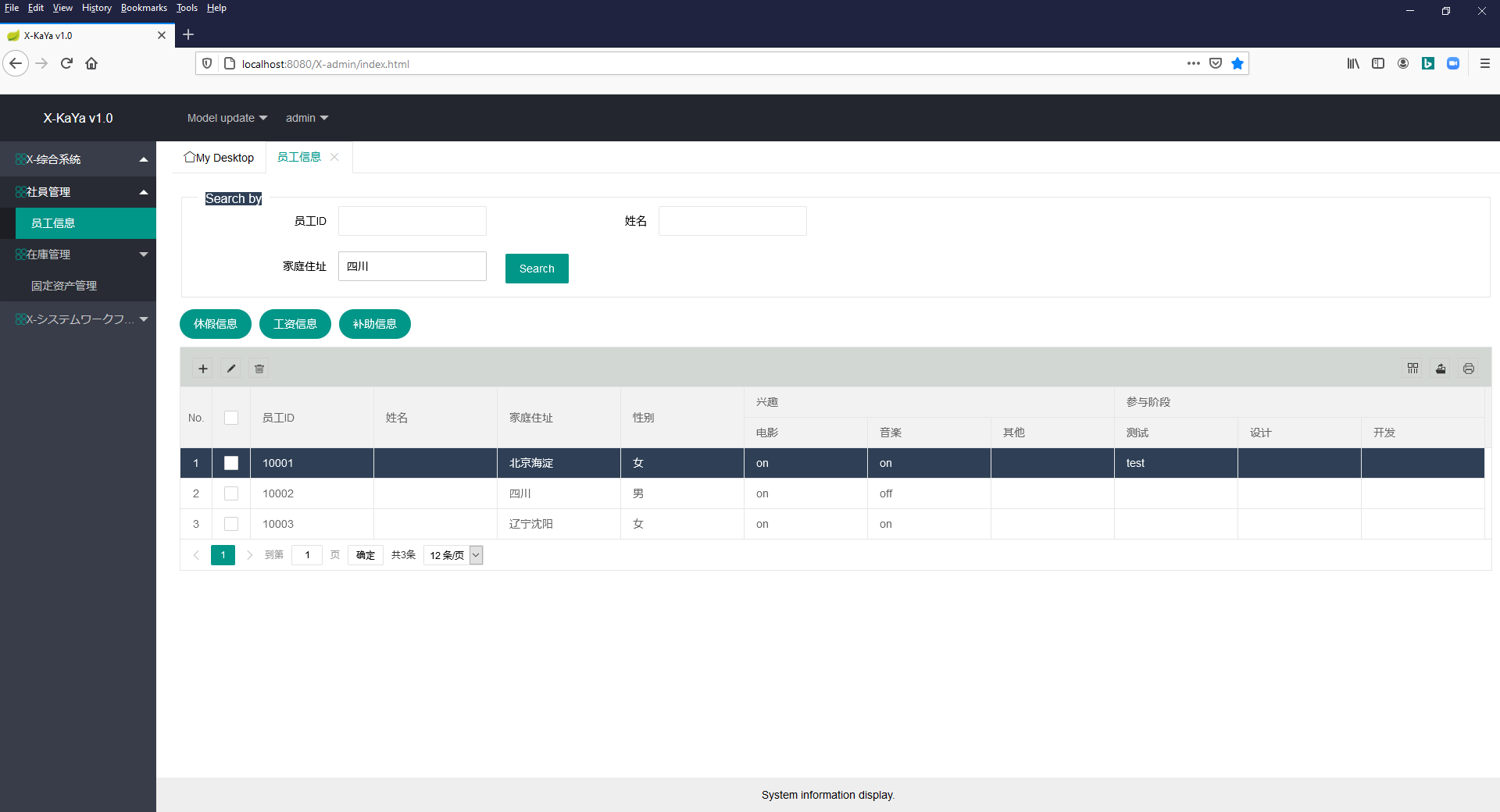


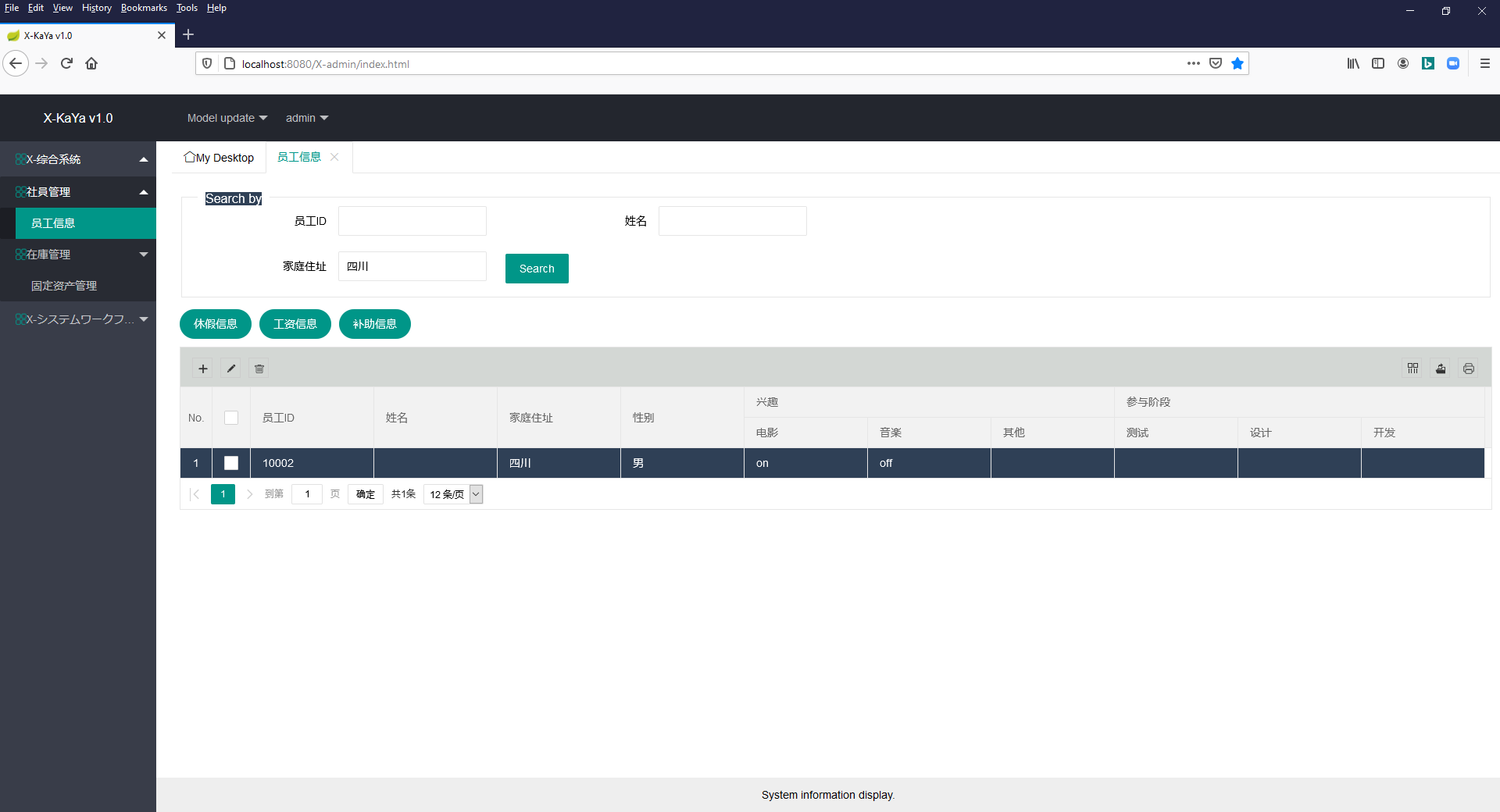
删除数据



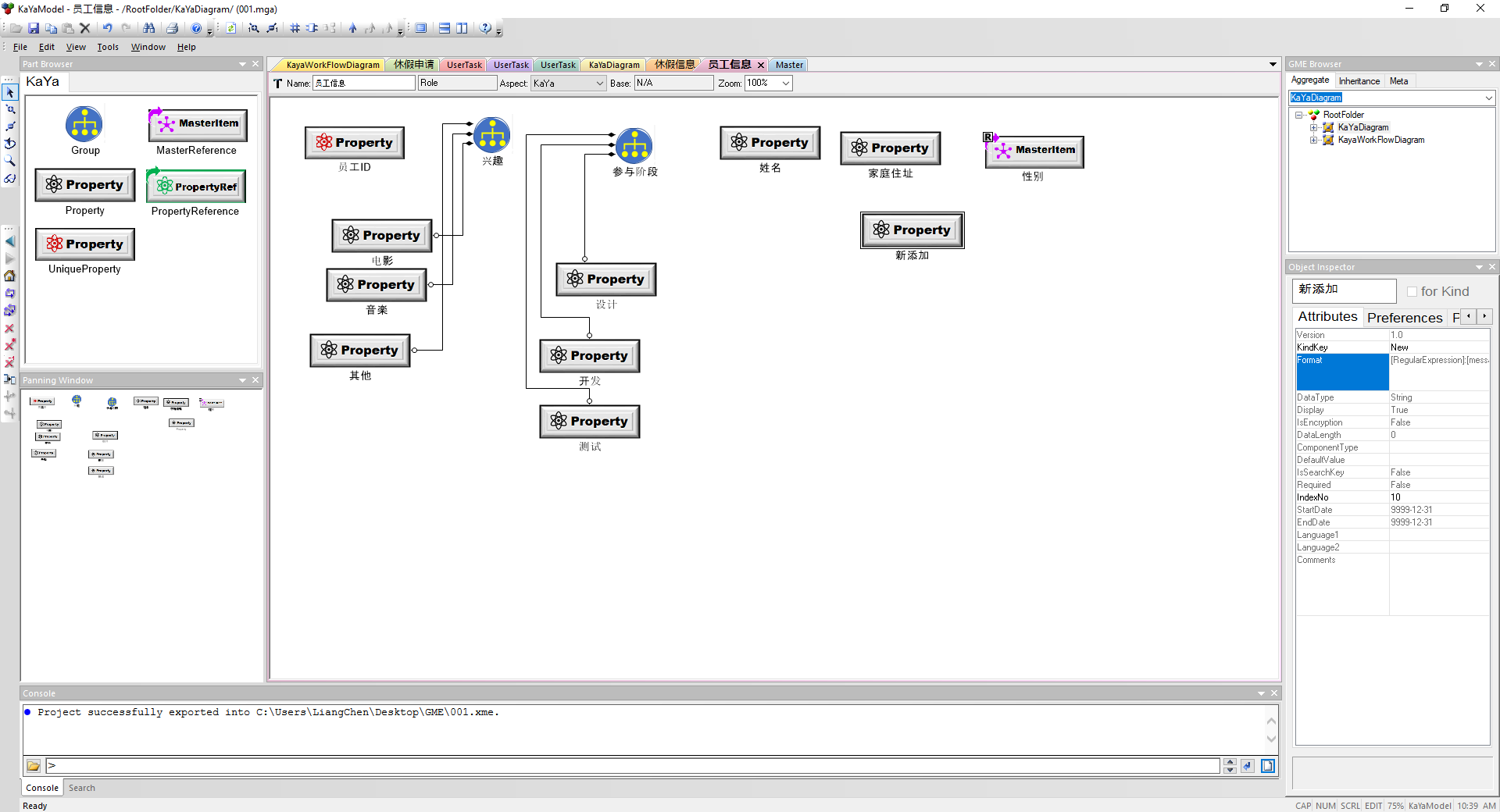


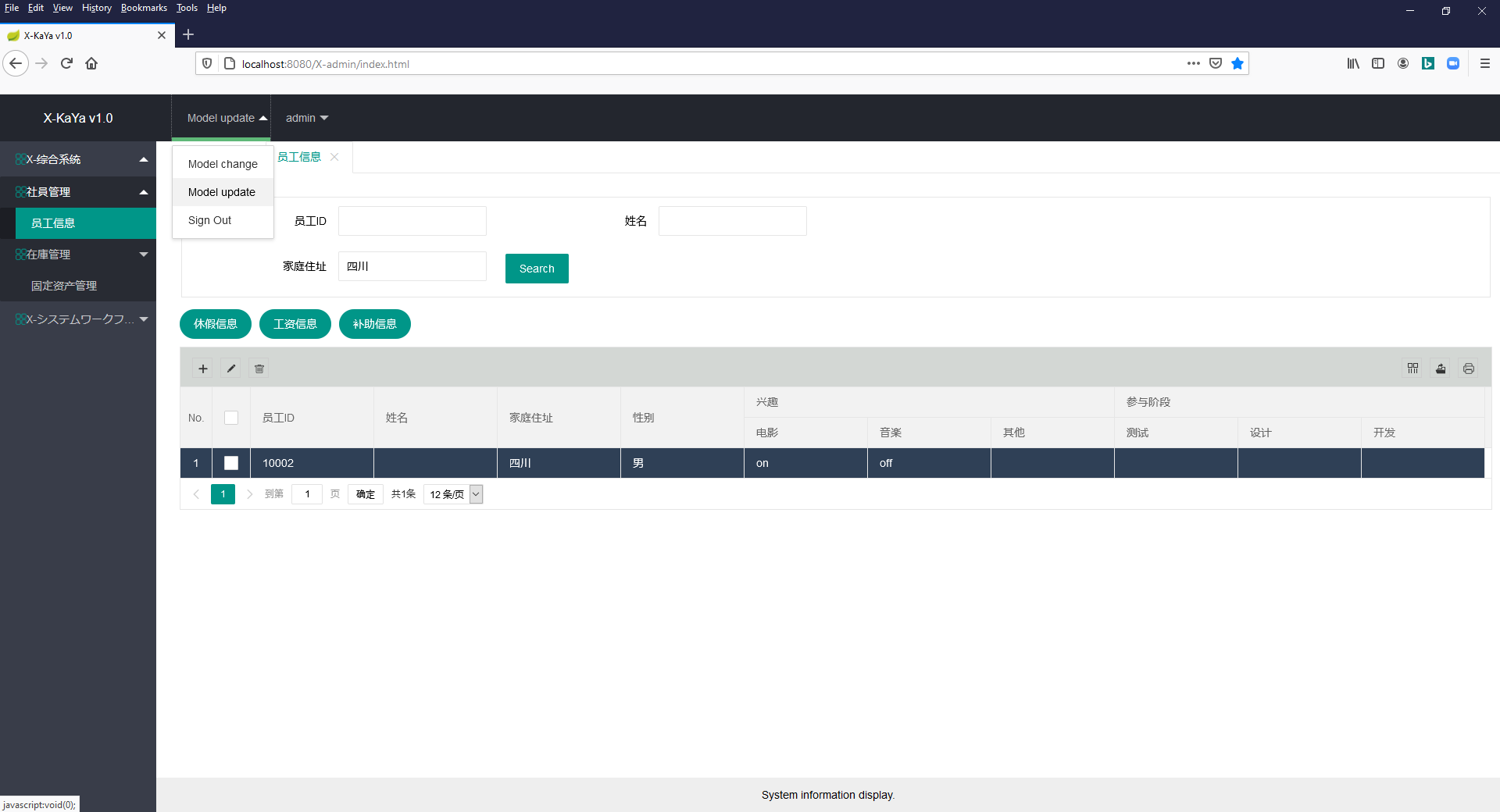
检索





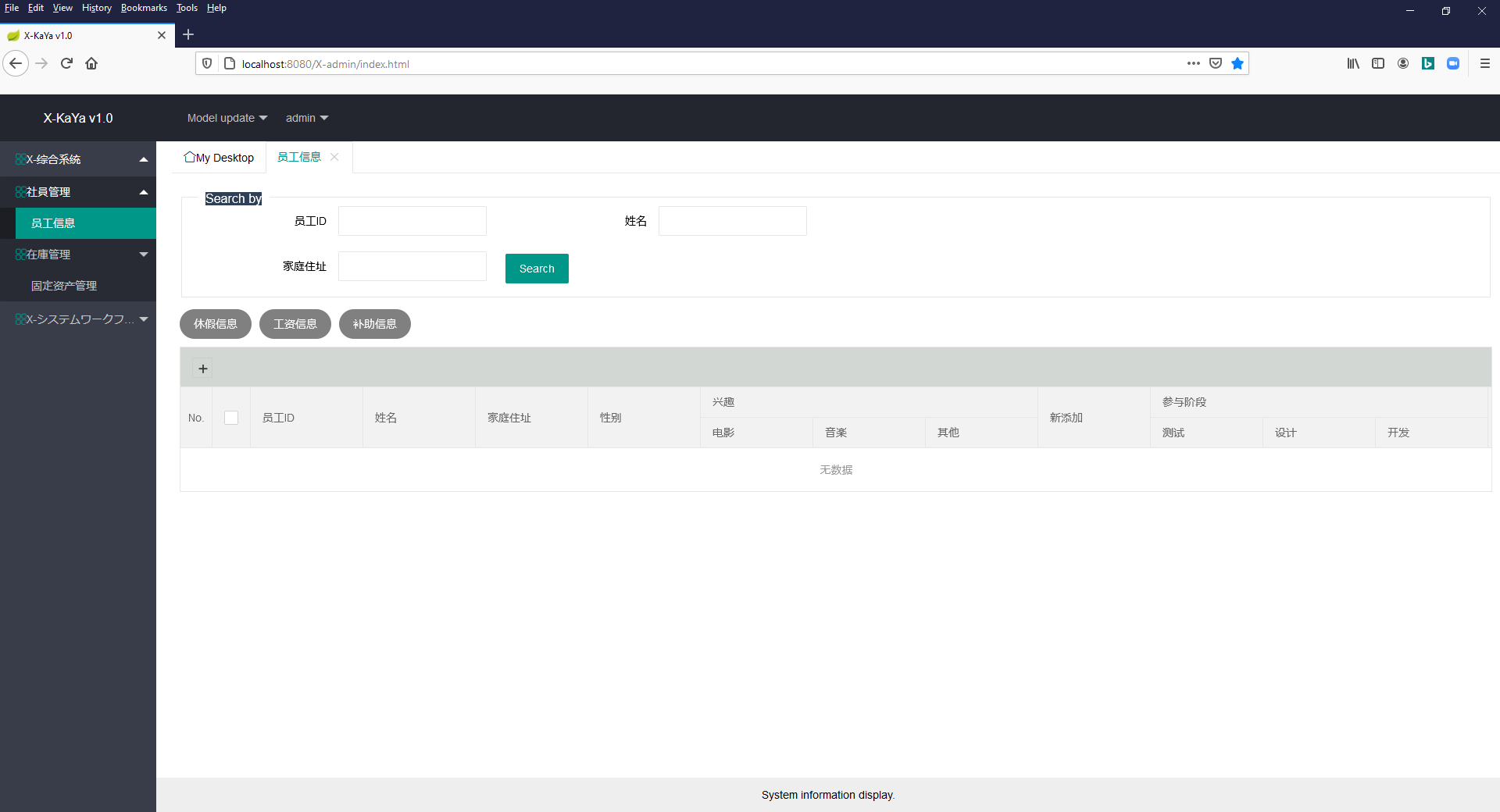
可以动态添加字段

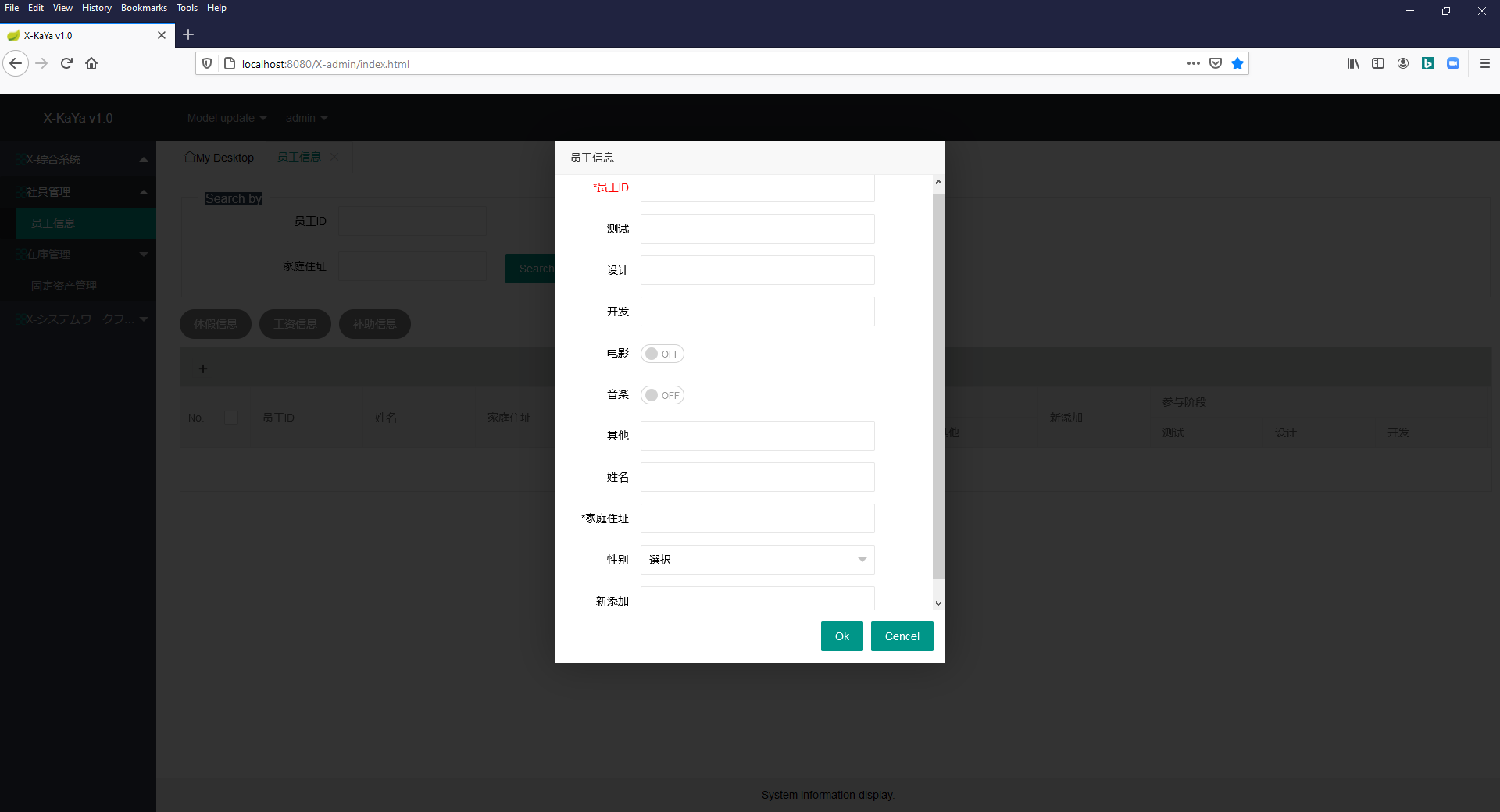




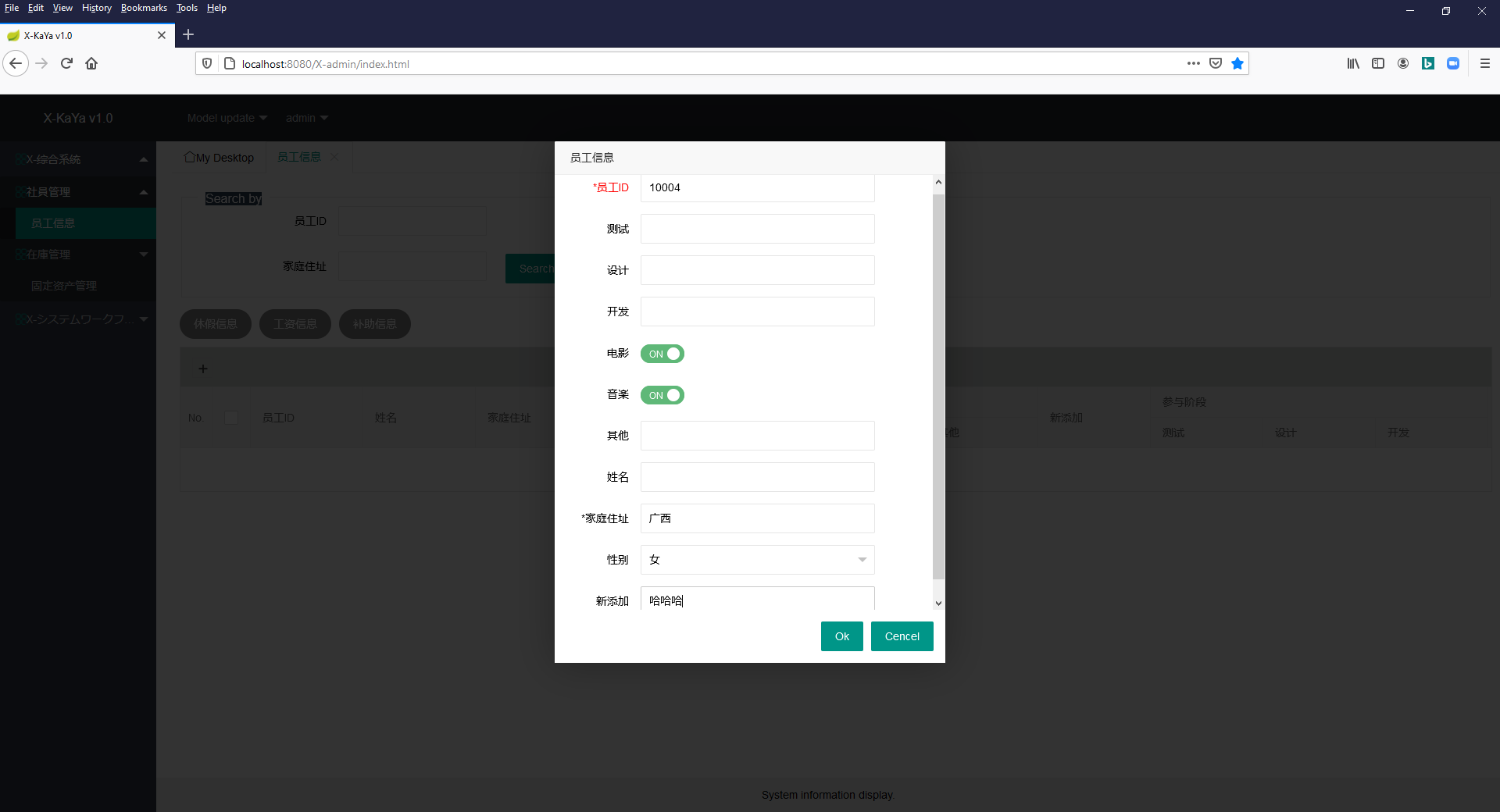
点击Model update

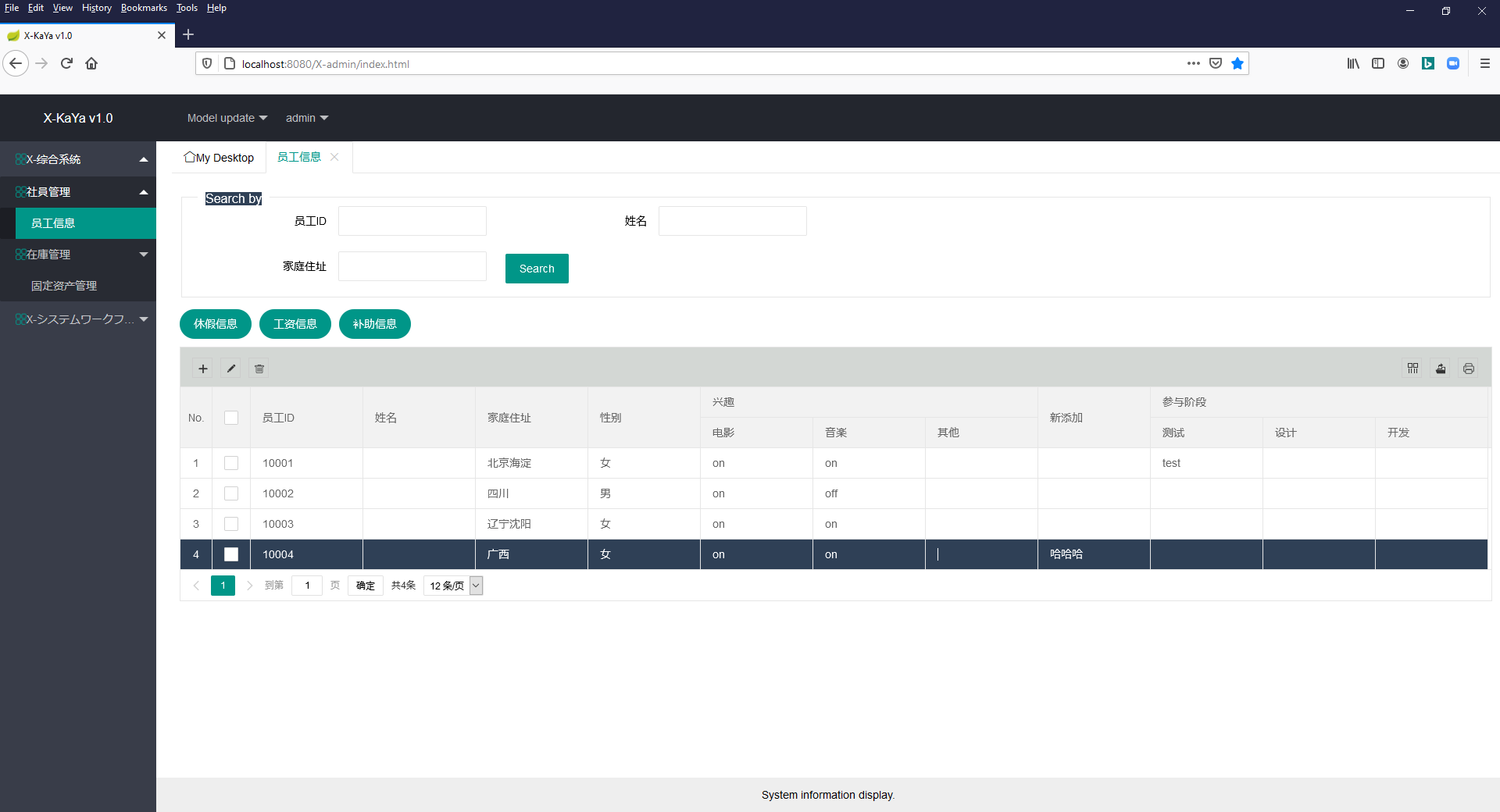
我们进到对应菜单





我们新加一条数据试试：





没问题，顺利插入新的字段，增删改查也是没有问题的，这里就不一一截屏了。

检索部分是可以通过模型属性随意增减的。

这个是前后端分离的，前段的UI可以选择你自己喜欢的平台。VUE，EsayUI，LayUI等，甚至HTML5都没问题。

系统提供的默认API很简单，目前只有如下：

1. CRUD（/kayaselect, /kayainsert,/kayaupdate, /kayadelete 4个）
2. UI API（/Menu，/Init 2个）这个是为了匹配UI ,实际可以根据前端的技术选择做调整，可扩展，高可用。

上面所有的API代码大约400行左右，就完成了整个系统基本的管理需求。

这就是元数据平台带来的优势。

*注意：这套前台的UI仅仅是为了验证smartKaYa而开发的。个人精力有限，不完善的地方请自己添加更改kaya.js代码相关部分。*

*如果有技术问题请联系：*[*chljapan@hotmail.com*](mailto:chljapan@hotmail.com) *或搜索微信号：chljapan*

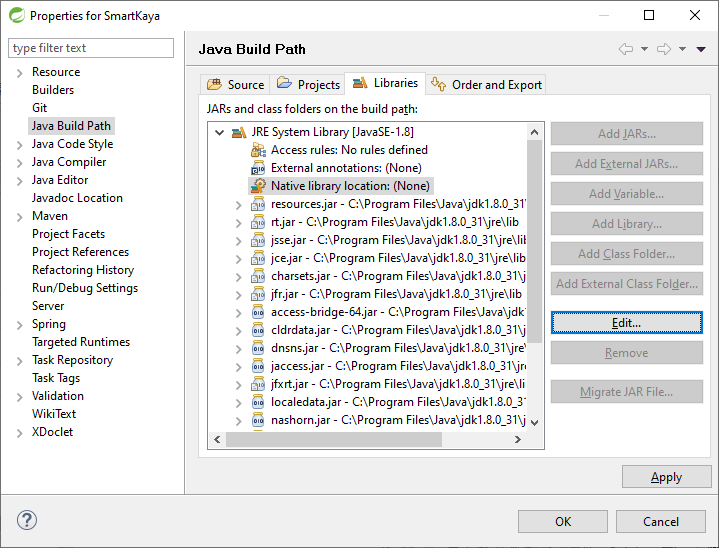
第二章 开发环境配置

1.导入smartkaya\_1.0.jar

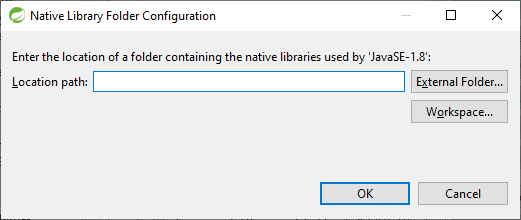
2.在项目pom.xml同目录下配置mga.dtd文件

3.在src/main/java/resources下配置JAUT.dll文件，然后在Build Path中配置Configure Build Path中的Libraries中

点击JRE System Library 中配置Native library location:到src/main/java/resources



点击Edit…



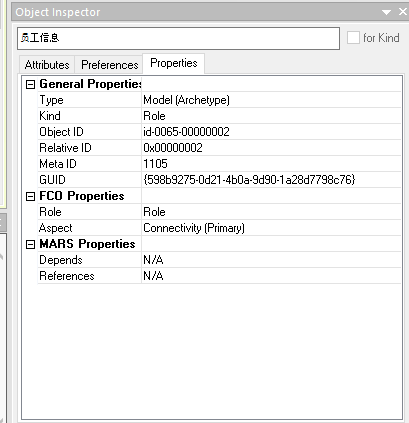
在Location Path 中填入JAUT.dll 文件所在的位置。

点击OK关闭配置。

4.配置kayaconfig.properties

然后你就可以边编辑模型，边开发了，模型和开发代码是实时的，你可以不需要导出文件，就可以看到系统的模样。

注意：下面用到的类似 id-xxxx-xxxxxxxx 利用下面的方法查看属于哪个元素。



※左侧的信息在KAYA模型概念中非常重要，这个ID是全局唯一，你可以通过这个ID在程序中随时找到这个元素的所有信息，或者设置该元素的所有信息。

第三章：程序篇

1.提供了完整的模型访问API

2.提供了多表操作的完整事务处理。

3.做到了模型和平台完全剥离，零代码生成。

4.如果是简单的系统，你基本不需要写代码就可以使用这个系统了。

5.支持多种数据库。（Oracle,Mysql,PostgreSql,H2,DB2等等 因为底层用的是基本的SQL语法，原则上支持大部分数据库）

6.提供业务流引擎。你可以单独利用它，也可以和元数据模型一起使用，同样是可视化的，非常容易使用。



7.如果你只用核心部分，Spring-Boot不是必须的。

8.Java部分代码大小为 1.2M（如果只是运行环境的代码会更小），前端代码layUI 13.8M

9.支持的Java版本从1.5～以上都支持。

通过OpenShift云环境中测试，部署更新维护等很方便，你仅仅需要维护你的模型就可以，想进一步了解KAYA扩展工作方式，可以联系技术人员。