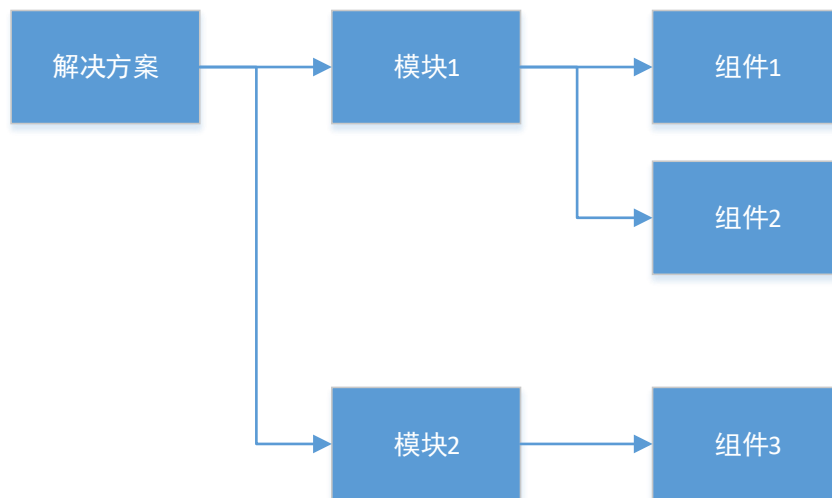


HAP 共享版-开发平台使用手册

Hap 开发平台是根据《领域驱动设计》的原理，使用模型驱动开发方式，尤其适合信息系统的开发。

模型的划分

平台是通过解决方案来代表一个开发项目，下面包含模块，模块又包含组件。

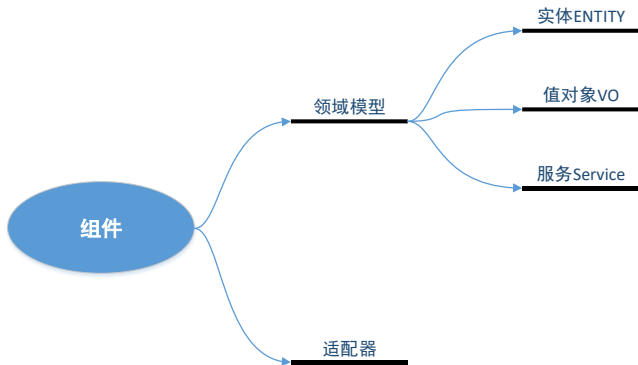


组件

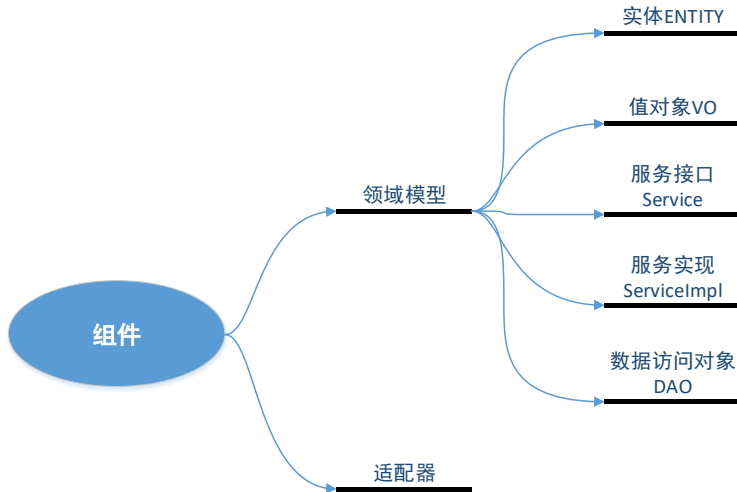
我们怎么划分一个组件呢？按照“限界上下文”的说法，当然要按照业务，具体要怎么做呢？我们根据业务建立领域模型后，就需要划分为更小的业务组件，每个业务组件会包含 1 到多个实体和服务，这些实体和服务应该是联系比较紧密的，例如：销售订单和订单行联系就很紧密，订货相关服务与销售订单和订单行就不可分割，它们就组成销售订单组件；同样发货单也是如此。

最简单的划分方法，按照经验，每一种业务单据都有可能成为一个组件，例如：采购订单、入库单、生产订单，但是不要分得太细，例如：固定资产采购订单和低值易耗品采购订单，这些采购订单的细分类别应该成为采购订单上的一个采购类型属性，而不是成为一个新的组件；第二，每种基础档案都有可能成为一个组件，例如：物料档案、仓库、销售价格表、计量单位等等，但是要注意，某些档案之间联系非常紧密，可以放在一个组件内，例如：仓库和库位，它们共同表示一个位置；计量单位组和计量单位；很多规则性的东西往往会依附在某个基础档案组件，例如：销售价格策略可以放在销售价格表中，取计量单位间转换率规则可以放在计量单位组件中。

组件内又包含什么呢？除了适配器（组件接口）外，内部的领域模型具体是什么呢？见下图。

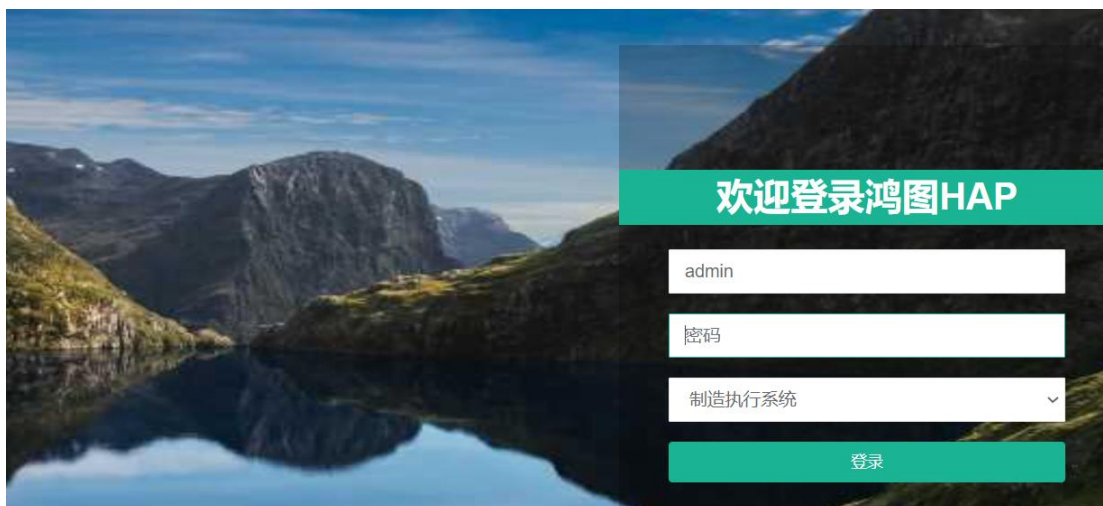


在软件开发中，业务逻辑层和数据访问层一般都会分离，实体代表数据层，所以从实现的角度，而业务逻辑层往往是接口和实现分离，又可以这样划分：



简介

登录：共享版 用户名 admin，默认密码 123456，用户可以进入后修改
输入用户名后，下方的下拉列表框显示所有的解决方案，选中后，登录后就是编辑当前解决方案的模型



进入后，主页画面如下：

- 1、最上面部分，是菜单
- 2、上面的树是模块和组件，下面的树是选中一个组件后，组件内的模型



组件模型

解决方案

通过菜单“系统/解决方案”进入

解决方案							
<div><div>退出</div><div>增加</div><div>编辑</div><div>删除</div></div>							
	编码	名称	描述	数据源类型	包名	创建时间	修改时间
<input type="radio"/>	base	基础	-	MySql	htsoft.base	2021-03-30 19:17:46	2021-06-13 10:02:06
<input type="radio"/>	mes	制造执行系统	-	MySql	htsoft.mes	2021-03-12 14:36:31	2021-06-13 11:13:46
显示第 1 到第 2 条记录，总共 2 条记录 每页显示 <div>10</div> 条记录							

增加：

增加解决方案

编码

名称

描述

数据源类型

包名

保存

取消

新增解决方案后，在登录时就可以选择，做为当前解决方案

模块

通过菜单“工具/模块定义”进入
维护当前解决方案的模块

模块定义							
<div><div>退出</div><div>增加</div><div>编辑</div><div>删除</div></div>							
	序号	编码	名称	包名	禁用	创建时间	修改时间
<input type="radio"/>	10	sm	系统管理	htsoft.mes.sm	否	2021-03-12 18:41:25	2021-06-13 11:13:46
<input type="radio"/>	20	base	基础数据	htsoft.mes.base	否	2021-04-12 15:08:20	2021-06-13 11:13:46
<input type="radio"/>	30	inv	库存管理	htsoft.mes.inv	否	2021-05-26 11:53:34	2021-06-13 11:13:46
显示第 1 到第 3 条记录，总共 3 条记录 每页显示 <div>10</div> 条记录							

新增模块： 序号是模块在左侧树中的显示顺序

增加模块

序号

编码

0

名称

保存

取消

组件

在左侧组件树中，鼠标右键点击模块，可以新增组件

增加组件

编码

名称

保存

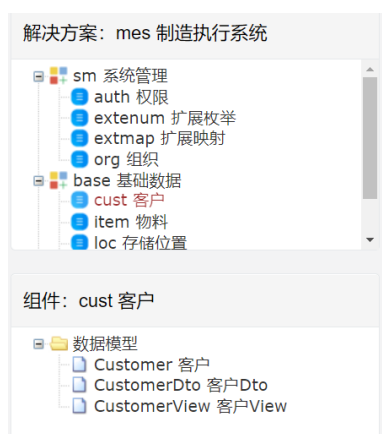
取消

在左侧组件树中，鼠标右键点击组件，可以修改组件

数据模型

在左侧组件树，选择一个组件，显示数据模型

数据模型有三种，实体、Dto、View



实体

在DDD一书中，“以标识作为其基本定义的对象称为实体”。也就是说，实体有唯一标识，就像每个人有身份证号一样，每一样物体都有一个唯一标识，那么在信息系统里，怎么

简单的表示呢？系统使用{实体编码}ID 作为每个实体的标识。

在信息系统中，经常有档案、单据实体，可以将这些作为实体的类型。另外，DDD 中有一种聚合根（Aggregate Root）实体，我们将它们命名为“主实体”，主实体聚合的实体叫做“子实体”，一个组件内可以包括 1 到多个主实体及多个子实体。主实体在维护的时候，一般是和其子实体一起进行的，例如：新增一张销售订单时，订单头和订单行一般是输入完毕后一块提交保存的，子实体的生命周期是由主实体决定的。

值对象

值对象是用于数据传递的，是在内存中的快照，断开系统后就会丢失。笔者认为在信息系统中，值对象可以分为两类，一种是服务传递用，一般是实体的一部分，例如：新增一个实体，传递的参数对象就是，系统命名为 DTO（数据传输对象），第二种是表示层显示需要，例如，显示一张订单，我们在保存这张订单时，关联数据保存的往往是实体的 ID，例如：客户 ID、物料 ID，但是在显示这张订单时，ID 是没有意义的，需要显示的是客户名称、物料名称、计量单位名称等等，所以表示层需要的值对象是一种将多个实体数据关联起来的对象，笔者命名为 View（视图对象）。

新增数据模型

在数据模型节点，鼠标右键弹出菜单，点击“新增数据模型”

增加数据模型

模型信息

保存

关闭

编码

名称

父实体

模型类型

实体类型

Entity

数据表/视图

属性

增加

编辑

删除

清除

拷贝

编码	名称	类型	值类型	可空	来源实体	来源枚举	来源映射	来源属性	类
没有找到匹配的记录									

模型类型：BaseEntity 基础实体（实体、档案、单据等等，通常不能使用），Entity 代表实体，Dto 代表数据传输对象，

实体类型：模型类型=Entity 后，可以选择实体类型

实体：具有 ID、创建时间、修改时间、创建人（当前有用户实体）、修改人（当前有用户实体）的数据表

档案：具有编码、名称的实体

单据：具有单据号、单据日期的实体

树型档案：具有上级 ID 的档案

数据表：一般不需要输入，对于实体，系统会自动设置为{模块编码}_{实体编码}；Dto 为空

新增属性

增加属性

编码

名称

值类型

普通

标准类型

数据类型

长度

精度

可空

是

缺省值

来源实体

枚举

映射

来源属性

保存

取消

扩展枚举：具有枚举项的数据字典，存储枚举值为 0，1，2，显示枚举项名称

扩展映射：具有{映射编码}={映射名称}的数据字典，存储映射编码，显示映射名称

编码：实体的编码会成为数据表字段

名称：实体的名称会成为数据表字段的备注

值类型：

普通：简单数据类型

实体：指向实体的 ID 引用字段，需要选择实体，注意：输入{实体编码}ID，会自动产生一个实体 ID 引用属性。

枚举：如果当前解决方案中有“扩展枚举”这个组件，可以选择，指向扩展枚举的引用字段，需要选择枚举，注意：输入{枚举编码}，会自动产生一个枚举引用属性

映射：如果当前解决方案中有“扩展映射”这个组件，可以选择，指向扩展映射的引用字段，需要选择映射，注意：输入{映射编码}，会自动产生一个映射引用属性

数据类型：数据库字段类型

长度：字段长度

精度：小数位数

标准类型：系统预置一个数据类型、长度、精度的组合，用于快速输入

拷贝

从其它组件的模型中复制属性

移动

当前选择属性移动到指定位置前面
选择属性，点击移动，输入位置编号，输入 0，表示最前面

显示所有属性

默认不显示一些每个实体都有的属性，如“创建人”、“创建时间”、“修改人”、“修改时间”、“版本”等，如果选中，系统会显示所有属性

关联属性

当前数据模型=View 的时候才能使用，用于选择关联的实体字段
前面打勾的是已经选择的关联实体字段，选择后会显示在当前数据模型的属性列表中，然后可以修改编码和名称



新增子模型

用于增加子实体，注意：子实体一般和主实体一起被维护，例如：新增入库单时，入库单头和入库单明细都要一起增加，然后一起提交，它们是聚合关系。

如果是普通的关联关系，不建议使用子实体，例如：物料档案和物料供应商关联，新增物料档案的时候，不需要考虑和供应商的关系，它们的关系是后来建立的。

点击模型，鼠标右键弹出菜单，点击“新增子模型”

增加子数据模型

模型信息

保存 关闭

编码 *

名称 *

父实体

模型类型

实体类型

InStoreDoc

Entity

数据表/视图

属性

+增加 编辑 删除 清除 拷贝

编码	名称	类型	值类型	可空	来源实体	来源枚举	来源映射	来源属性	类
没有找到匹配的记录									

新增子模型后，父实体和模型类型都已经固定，只能选择实体类型

保存模型后，系统会在属性列表中增加对父实体的引用

修改数据模型

保存 关闭

+ 基本信息 属性

+增加 编辑 删除 清除 关联属性 拷贝 移动 显示所有属性

	行号	编码	名称	类型	值类型	可空	来源实体	来源枚举	来源映射	来源属性
<input type="radio"/>	1	InStoreDocLineID	入库单行ID	key	普通	否	-	-	-	-
<input type="radio"/>	2	InStoreDocID	入库单ID	parentid	实体	否	InStore...	-	-	-
<input type="radio"/>	3	LineNo	行号	normal	普通	否	-	-	-	-
<input type="radio"/>	4	LocationID	库位ID	normal	实体	是	Location	-	-	-
<input type="radio"/>	5	ItemID	物料ID	normal	实体	否	Item	-	-	-
<input type="radio"/>	6	BatchNo	批号	normal	普通	是	-	-	-	-
<input type="radio"/>	7	UomID	计量单位ID	normal	实体	否	Uom	-	-	-
<input type="radio"/>	8	SnList	序列号列表	normal	普通	是	-	-	-	-
<input type="radio"/>	9	Qty	数量	normal	普通	否	-	-	-	-

控制器方法

维护主实体的控制器方法，控制器方法是主实体产生的 Restful 格式的 http 调用方法，非常适合 web 前端和微服务调用

系统会自动生成一个增删改查的控制器方法，用户也可以新增方法

控制器方法列表

操作类型

关闭

增加

编辑

删除

	编码	名称	url	操作类型	表单	返回类型	权限控制	数据权限	列权限
<input type="radio"/>	instoredoc_add	保存新增入库单	/instoredoc/add	post	-	ActionResult	是	否	否
<input type="radio"/>	instoredoc_delete	删除入库单	/instoredoc/delet...	delete	-	ActionResult	是	否	否
<input type="radio"/>	instoredoc_save	保存入库单	/instoredoc/save/...	post	-	ActionResult	是	否	否
<input type="radio"/>	instoredoc_update	保存修改入库单	/instoredoc/updat...	put	-	ActionResult	是	否	否
<input type="radio"/>	instoredocs_findlist	按条件查询入库单	/instoredocs/findlist	post	-	ActionResult	是	是	是

显示第 1 到第 5 条记录，总共 5 条记录 每页显示 10 条记录

其它功能

枚举定义

维护枚举定义

前提：当前解决方案中有“扩展枚举”这个组件

通过菜单“工具/枚举定义”进入

枚举定义

关闭

增加

编辑

删除

复制

	编码	名称	允许增加	允许编辑	创建时间	修改时间
<input type="radio"/>	InStoreStatus	入库状态	否	否	2021-05-26 11:...	-
<input type="radio"/>	ItemForm	物料形态	否	否	2021-05-06 14:...	-
<input type="radio"/>	OutStoreStatus	出库状态	否	否	2021-05-27 14:...	2021-05-27 14:...
<input type="radio"/>	TraceType	追踪类型	否	否	2021-05-25 10:...	-

显示第 1 到第 4 条记录，总共 4 条记录 每页显示 10 条记录

编辑类型定义

保存

取消

编码

InStoreStatus

名称

入库状态

允许增加

否

允许编辑

否

枚举项

+增加

编辑

删除

	编码	名称
<input type="radio"/>	0	提交
<input type="radio"/>	1	审核

允许增加：允许应用中增加枚举项
允许修改：允许应用中修改枚举项名称

映射定义

维护映射定义

前提：当前解决方案中有“扩展映射”这个组件

通过菜单“工具/映射定义”进入

映射定义

关闭

+增加

编辑

删除

复制

	编码	名称	允许增加	允许编辑	创建时间	修改时间
<input type="radio"/>	OutStoreMap	出库类型	否	否	2021-05-27 14:...	-
<input type="radio"/>	InStoreMap	入库类型	否	否	2021-05-26 11:...	-

显示第 1 到第 2 条记录，总共 2 条记录 每页显示 10 条记录

编辑映射定义

保存

取消

编码

名称

允许增加

允许编辑

OutStoreMap

出库类型

否

否

映射项

+增加

编辑

删除

	编码	名称
<input type="radio"/>	sale	销售出库
<input type="radio"/>	issue	领料出库
<input type="radio"/>	checkout	盘亏出库
<input type="radio"/>	other	其它出库
<input type="radio"/>	purchase return	采购退货

允许增加：允许应用中增加映射项
允许修改：允许应用中修改映射项名称

代码生成

根据定义的数据模型，生成组件代码
通过菜单“系统/代码生成”进入

代码生成

模块

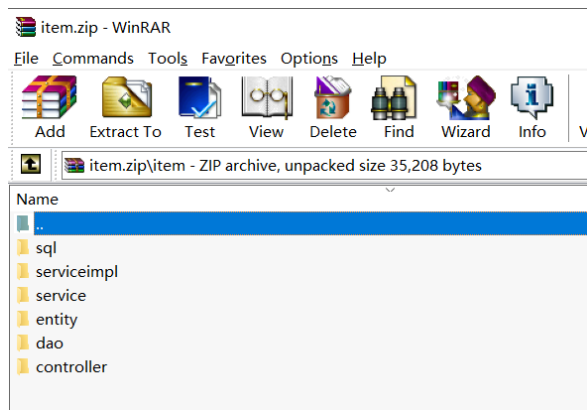
关闭

查询

+创建代码

	模块编码	模块名称	组件编码	组件名称	包名	文件	创建时间
<input type="radio"/>	base	基础数据	cust	客户	htsoft.mes.base....	-	-
<input type="radio"/>	base	基础数据	item	物料	htsoft.mes.base....	下载 删除	2021-07-11 17:41:07
<input type="radio"/>	base	基础数据	loc	存储位置	htsoft.mes.base....	-	-
<input type="radio"/>	base	基础数据	uom	计量单位	htsoft.mes.base....	-	-
<input type="radio"/>	inv	库存管理	instore	入库单	htsoft.mes.inv.in...	-	-

选择组件，点击“创建代码”，创建后，文件中出现链接，点击下载，下载代码包



sql 目录：包含创建数据表的脚本
controller：生成的控制器代码
entity：生成的实体代码
dao：生成的数据访问接口代码
service：生成的服务接口代码
serviceimpl：生成的服务实现代码

数据导出

导出系统定义的枚举定义和映射定义
通过菜单“系统/数据导出”进入

数据导出

关闭 生成

	数据	文件	创建时间
<input type="radio"/>	枚举定义	下载 删除	2021-06-10 10:15:07
<input type="radio"/>	映射定义	下载 删除	2021-06-23 15:13:11

显示第 1 到第 0 条记录，总共 0 条记录 每页显示 10 条记录

选择项目，点击生成，然后文件中出现链接，下载文件

枚举定义和枚举项的数据脚本
映射定义和映射项的数据脚本

项目运行

- 1、通过代码生成，下载所有的组件代码，并解压
- 2、通过数据导出，下载枚举定义和映射定义
- 3、创建项目数据库，执行组件代码中的 sql，执行枚举定义脚本和映射定义脚本
- 4、仿照示例项目 testbase，创建 springboot 项目，将组件代码复制到项目中，注意需要引用 platform-base-1.0.0.jar（平台基础运行框架包）
- 5、配置项目文件，运行

正式版

参见：[htplatform.cn](http://platform.cn)

QQ 群：**894953310**

特点：

- 1、实时数据库和数据表创建，创建数据模型时，自动创建数据表和视图
- 2、定义模型后，实时创建代码
- 3、定义表单，所见即所得，实时创建页面
- 4、定义项目菜单
- 5、提供前端组件框架（bootstrap）和应用 js 库，方便快速开发前端页面
- 6、后端提供 mybatis/jpa 两种框架代码生成
- 7、代码生成模板可以维护，用户可以定制自己的代码生成模板
- 8、多用户访问，项目权限控制