

iTop 个性化定制

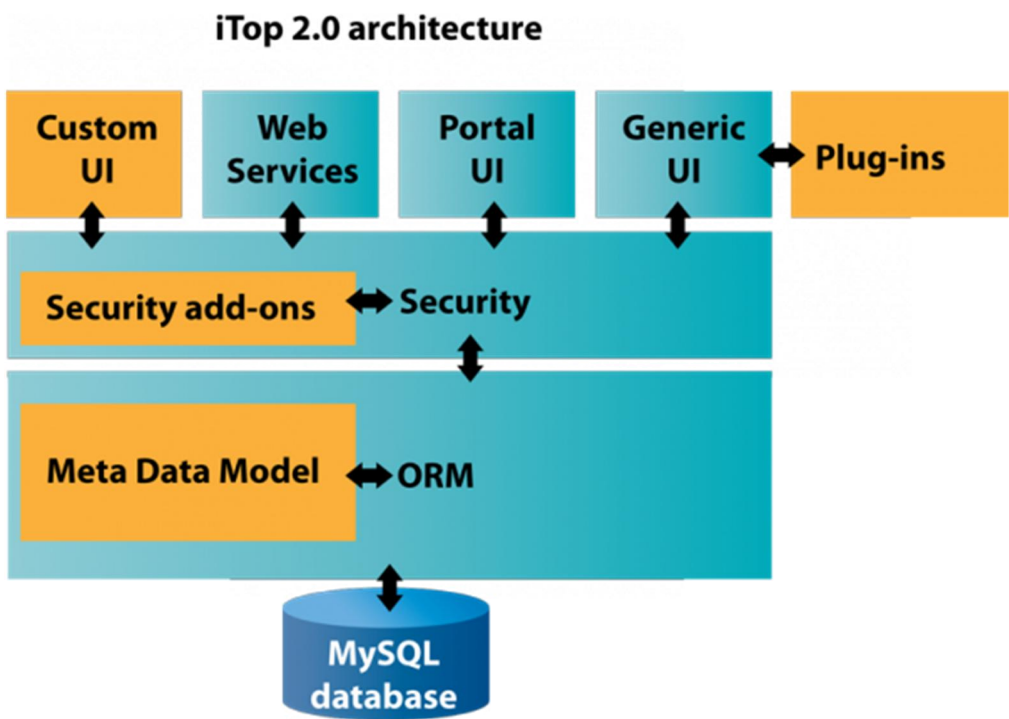
翻译: bollwarm@tom.com (ijz.me)

iTop 2.03 全中文版本下载: <http://git.oschina.net/ijz/itop203-ZHCN>

本文档基于 iTop2.0. 对于老版本的 iTop 点击相应文档链接 [the old toolkit](#) 。2.0 工具集下载地址: [iTopDataModelToolkit-2.0.zip](#)

iTop 构建与 ORM (Object Relational Mapper, 对象关系映射) 抽象层上, 基于由 php 类组成的“元数据模型 (Meta Data Model) 定义之上。” iTop2.0 的元数据模型采用 xml 文件定义, 在安装或者升级过程中被编译成 PHP 类。

iTop 架构如下图所示:



上图中橙色方框表示的部分是可定制化的模块。

定制化类型	手段	文档	预期操作者
扩展或者修改元数据模型	XML 文件	XML reference	iTop 顾问, ITIL 专家
通过插件扩展通用接口	PHP 文件	Extensions API	PHP 开发者
创建新的用户接口页面实现新的 web 服务, 特定的输出或	PHP 文件	ORM API	PHP 开发者

者源于用户接口的特殊任务			
创建自己的权限表	PHP 文件	用户权限 API	PHP 开发者

从 iTop 2.0 开始，调整元数据模型可以通过写一个简单的 xml 文件，此 xml 文件只需包含了和标准数据模型的差异。通过这种方式，你的修改保存在一个独立的文件，这样可以保证新版本下的兼容性。这样可以保持你的个性化操作在升级后还可以正常使用。

理解 iTop 的目录结构

iTop 的目录结构层次，如下图所示：

iTop 2.0 files layout

📁 iTop	
📁 addons	User-rights management add-ons
📁 application	Generic UI (PHP classes)
📁 conf	Configuration files
📁 production	Created by the setup
📁 toolkit	Created by the toolkit
📁 core	ORM engine
📁 css	Generic UI (Cascading Style Sheets)
📁 data	Local data generated by the application
📁 datamodels	
📁 1.x	Data model for upgrading a 1.x version
📁 2.x	Data model for 2.x versions
📁 dictionaries	Generic UI (Dictionaries)
📁 documentation	Online help for the setup
📁 env-production	"Compiled" data model, generated by the setup, do not edit
📁 env-toolkit	"Compiled" data model, generated by the toolkit, do not edit
📁 extensions	Copy your iTop extension modules in this folder
📁 images	Generic UI (Images)
📁 js	Generic UI (Javascript files)
📁 lib	Third party PHP libraries
📁 log	Log files
📁 pages	Generic UI pages
📁 portal	Portal user interface
📁 setup	Setup program
📁 synchro	Data synchronization
📁 toolkit	Toolkit for modifying the data model
📁 webservices	Import/Export and SOAP web services

以下目录在 iTop 中有特殊的使用：

- **conf**：包含了配置文件，每一个环境包含在一个子目录中（见下面环境部分）

- **data:** 包含了系统生成的数据，比如对象生命周期的图片（如果系统中 graphviz 可以用的话）。
- **Datamodels:** 包含了元数据模型的定义，每一个大版本包含在一个子目录中
- **env-xxxx:** 此目录（一个环境一个）包含了“编译”的数据模型。每次重新安装或者升级 env-production 目录会重新生成。
- **extensions:** 此目录为存放 iTop 非标准版本部分的扩展模块。
- **log:** 子目录包含了系统日志：setup.log 和 error.log

所有的其他目录可以被认为是系统源代码的一部分，通常不允许修改。系统不会写数据到这些目录，他们可以被设置为对 web 服务进程只读。

环境

iTop 2.0 引入了环境的新概念。环境表示有相同代码但是不同数据模型和配置文件的 iTop 运行实例。

- 一个配置文件保存在 /conf/环境名/config-itop.php
- 一个运行时数据模型保存在 env-环境名目录

两个不同的环境可以操作同样的数据，如果他们配置文件一样的话。

为了编译 xml 数据模型并且测试其对线上环境的兼容性质，可以用 toolkit 自动的创建一个独立的环境（命名为 toolkit）。当变化没有问题，你可以知道 toolkit 把这些合法的变化应用到线上环境。

扩展模型

iTop 最基本的数据定义部件叫作模块。

模块由实现给予功能的所必需的所有文件组成：数据模块定义 xml 文件、php 类，js 和 css 文件，php 页面，图片，等等。一个模块至少包含一个文件：模块定义文件，通常命名为 module. 模块名.php。

尽管你可以通过修改源文件，但是定制 iTop 的最好的方式是创建个性化的模块。这样你创建是一个纯净的个性化包，允许容易的安装部署以及兼容性升级。

PHP vs XML 数据模块定义

在 iTop 1.x 版本中，数据模型都是通过 php 类来定义的。2.0 开始支持 php 和 XML 两种方式的数据定义。

XML 定义相对于 php 类最大的好处是：一个模块定义的 XML 可以被另一个模块修改。例如，我们可以在安装时候创建一个扩展模块，这个模块在表述“Server”类的基础上增加一个属性，“Server”类是定义在标准 iTop 数据模型中。通过这种定义的话，用扩展模块就不需要替换整个“itop-config-management”模块（Server 类定义的模块），而只需调整修改另外模块中的定义。

在 PHP 中要达到同样的效果，必须要克隆整个 itop-config-mgmt 模块，并且用你的修改过的版本替换它。当一个新版本的模块发布后，你必须重做差异对比和补丁的工作，来生成你自己的模块。由于 XML 定义是直接定义差异，升级是自动实现的。

模块的构成

如果 my-module 为你模块的名称，模块目录将包含以下的文件：

文件名	描述
module. <i>my-module</i> .php	模块定义文件。必须地。包含模块的描述（名称版本，和其他类的依赖性，等）和他的组件。
datamodel. <i>my-module</i> .xml	XML 的数据定义。基于此 XML 定义，通过“编译”生成 model.xxxxxx.php 文件。XML 文件包含类、菜单和配置的定义。
model. <i>my-module</i> .php	如果你选择直接通过 php 类来定义数据模型，则用这个文件来进行定义。
main. <i>my-module</i> .php	PHP 代码和应用。有些模块包含一些 PHP 代码，可以容易把 PHP 代码抽取出来，存放到这个独立的文件中编辑，然后将其嵌入 XML。
images	把图片（类图标，等等）存放到他们的子目录是很好的实践。

itop-前缀的模块名为 iTop 包官方模块保留字。为了避免和其他扩展命名冲突建议命名你个性化模块的名称前缀为你公司名。例如 Cobodo 发布的个性化模块命名为 *combodo-xxxx*。

创建一个模块

我们以一个空模块作为个性化模块创建例子作为教程的开始。首先在 Toolkit 工具中填写如下信息，点击生成按钮，生成此模块。

模块名	<div>sample-modu</div>	（必须独特。“itop-” 和 “combodo-”前缀的名称为 Combodo 保留字。）
模块标签	<div>Sample Modu</div>	（在安装过程中显示）

模块版本	<input type="text" value="1.0.0"/>	
分类	<input type="text" value="business"/>	
依赖关系	<input type="text"/>	(模块名版本列表，用逗号隔开多个模块)

安装 Toolkit

- 下载 toolkit 压缩包: [iTopDataModelToolkit-2.0.zip](#)
- 解压压缩包的文件到开发实例根目录/, 生成一个 toolkit 目录。
- 通过浏览器访问这个: http://<your_itop>/toolkit

开发流程

1. 创建一个空的模块
2. 安装一个 iTop 开发实例，载入 “extensions” 目录你的空模块。
3. 在你的开发实例中安装 toolkit
4. 编辑你的扩展模块并用 toolkit 工具验证修改。
5. 把你扩展模块中的变化发布到正式环境。
6. 用实例数据测试你的模块。如果有问题，重复 4-6 步骤。

每次你对 XML 文件的修改都必须被编译成 PHP 文件，这就是 toolkit 所要做的工作。除此之外，你模块中的其他文件都是从 “extension/your_module” 目录简单复制到 “env-production/your_module”。为了加速编译过程，在 linux 下你可以通过文件符号链接的形式链接到他们源文件。这样源文件目录的任何修改的都对 iTop 立即起作用，你只要刷新浏览器即可。

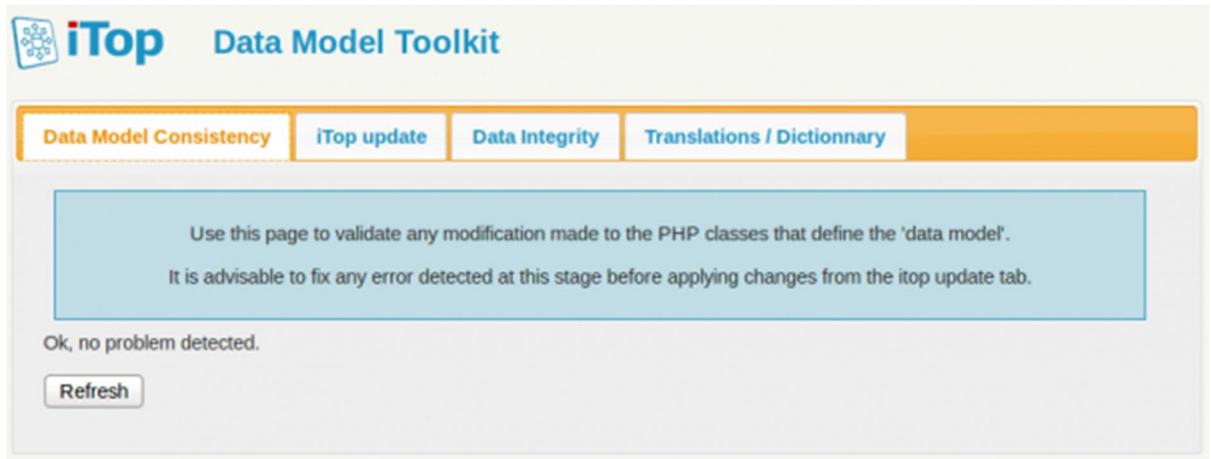
当你扩展调试完好，你可以把它们发布到生产系统：

1. 复制你扩展模块到生产环境的 “extensions” 目录。
2. 给予配置文件读写权限。
3. 运行安装程序，选择你扩展列表中你的模块中。

使用 Toolkit

当 toolkit 安装好后，在浏览器中输入 `http://<your_itop>/toolkit`。

第一个标签页进行一些基础冲突检查和数据模型定义验证。每次当你修改数据模型定义后，使用“refresh（刷新）”按钮执行数据验证。需要说明的是此处验证的过程都是在特殊的“toolkit”环境下，对实际中生产环境下的 iTop 实例没有任何的影响。



XML 文件加载的顺序非常重要（特别当某个 XML 文件修改了另一文件的定义的模型时候）。加载的顺序是通过模块定义文件中的依赖定义来执行的。

如果收到如下的错误信息：

```
XML datamodel loader: could not find node for class/XXXX
```

这大概就是因为你的模块缺少了依赖定义，该依赖定义了 XXX 类。

当 setup 运行编译过后，加载的顺序都被写入了配置文件。要注意的是 toolkit 不会自动调整加载的顺序。因此，如果你缺少了一个依赖定义，在调整模块定义文件后，需要执行安装过程，修改配置文件夹。

当第一个标签页不提示任何错误后，你可跳到第二标签页操作：

- 检查数据库结构的变化
- 通过“Udpate iTop Code”执行数据库结构变化和数据模型定义到 iTop 生产环境

Use this page to preview the changes in the format of the database, and update iTop. For example if you add a new field to an object, this new field must be added into the database as well.

Note: the current version of the tool does not remove unused fields!

Ok, the database format is compliant with the data model. (Note: the views have not been checked)

Refresh

Update iTop code

SQL commands to copy/paste:

```
ALTER VIEW `view_ActionEmail` AS SELECT DISTINCT `_priv_action_email`.`id` AS `id`,
`_priv_action`.`name` AS `name`, `_priv_action`.`description` AS `description`,
`_priv_action`.`status` AS `status`, `_priv_action_email`.`test_recipient` AS
`test_recipient`, `_priv_action_email`.`from` AS `from`, `_priv_action_email`.`reply_to`
AS `reply_to`, `_priv_action_email`.`to` AS `to`, `_priv_action_email`.`cc` AS `cc`,
`_priv_action_email`.`bcc` AS `bcc`, `_priv_action_email`.`subject` AS `subject`,
`_priv_action_email`.`body` AS `body`, `_priv_action_email`.`importance` AS `importance`,
`_priv_action`.`realclass` AS `finalclass`, CAST(CONCAT(COALESCE(`_priv_action`.`name`,
'')) AS CHAR) AS `friendlyname` FROM `priv_action_email` AS `_priv_action_email` INNER JOIN
(`priv_action` AS `_priv_action` ) ON `_priv_action_email`.`id` = `_priv_action`.`id`
WHERE 1 ;
```

第三个标签页用来更新数据同步源（如果任何数据模型变化有影响的话），并且检查和基础数据模型类层级继承键的差异。此页的操作直接影响生产环境。

Use this page to check the integrity of the data in the iTop database.

Synchronization Data Sources

There are no Data Sources in the database.
Ok.

Refresh

Hierarchical Keys

The attribute parent_id from Organization is a hierarchical key.
Ok, the attribute parent_id from class Organization seems up to date.
The attribute parent_id from Group is a hierarchical key.
Ok, the attribute parent_id from class Group seems up to date.

Rebuild HKeys Anyway

Refresh

生命周期图

对象的生命周期图在 iTop 数据模型页显示（标签 Life-cycle”）。如果你修改一个对象的生命周期，你需要在 web 服务器安装 Graphviz 来重新生成这个图。关于安装 Graphviz 的更多信息，见 iTop 安装文档。