工业数字模型驱动引擎(iDME)

开发指南

文档版本 05

发布日期 2024-12-23

华为云计算技术有限公司

商标声明


和其他华为商标均为华为技术有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为云计算技术有限公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为云计算技术有限公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

华为云计算技术有限公司

地址： 贵州省贵安新区黔中大道交兴功路华为云数据中心 邮编：550029

网址： https://www.huaweicloud.com/

目 录

1 简介

概述

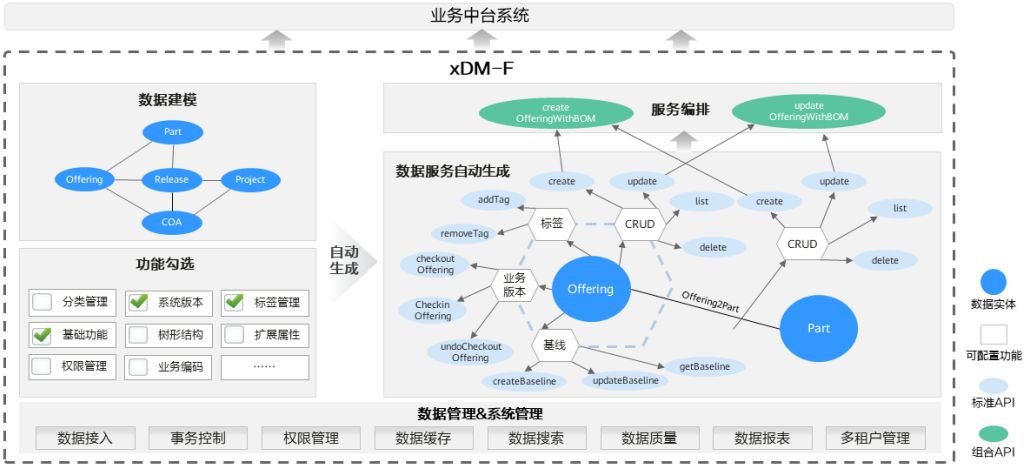
随着企业数字化转型的浪潮不断来袭，众多非数字原生企业的多年信息化建设，除了给企业带来了成长的红利，也给企业带来了数据管理和消费的历史包袱。当前，国内企业业务持续的风险不断升高，现有系统面临业务数据获取周期长、业务数据转换效率低、升级替换难等挑战，亟须破局。工业数字模型驱动引擎（Industrial Digital Model Engine，简称iDME），是基于数字化变革和数据管理优秀实践开发的创新型工业软件开发与运营平台，打造“基于模型+数据驱动”的公共底座，为协同打造新一代全栈自主可控工业软件体系提供根技术服务，助力快速构建和运营云化SaaS化工业软件，汇聚企业全场景数据，构建企业级数据图谱，提供万数互联的数据索引、追溯、交互服务。

开发指南从iDME功能实现的角度，介绍iDME的功能、使用方法、样例工程及代码介绍，提供给需要进行业务应用开发的并具备Java开发经验的开发人员使用。

xDM-F 架构

您可以华为云上管理和使用工业数字模型驱动引擎-数据建模引擎（xDM Foundation，简称xDM-F）服务，也可以将xDM-F集成到您自己的应用中。

图 1-1 xDM-F 架构



功能使用

表 1-1 xDM-F 功能使用

功能	描述	使用方式
基础数据服务	针对数据实体、关系实体、接口模型等自动生成数据服务，例如生成增、删、改、查、分页查询等原子接口，实现基础服务的调用。	• 控制台 • API
文件服务	可以在数据实例中上传、下载多种格式文件，可满足文件管理的诉求。	• 控制台 • API
安全受控	安全受控功能可以高安全、高可靠的管控数据访问权限及业务关键信息。	• 控制台 • API
主版本服务	可以对数据对象进行检出、修订、检入操作，为M-V模型数据对象中master提供父类。	• 控制台 • API
版本服务	可以为多版本对象提供M-V模型原子服务，用于业务对象追溯管理，提高追溯效率。	• 控制台 • API
系统版本	可以通过版本号标识记录修改历史记录。	• 控制台 • API
树形结构	可以指定全路径字段，为树形节点全路径绑定属性。	• 控制台 • API
权限管理	可以根据自身业务需求，对数据或功能进行授权鉴权的开发。	• 控制台 • API
业务编码生成器	业务编码生成器功能可以自定义规则，自动生成业务编码，为系统维护者提供便利。	• 控制台 • API
文件夹管理	可创建多层级的文件夹结构，在创建数据实例时可指定所属文件夹，方便对数据进行分门别类。	• 控制台 • API
分类管理	可以根据分类节点及分类属性快速检索对象，节约时间成本、提高查询效率和业务效率。	• 控制台 • API
扩展属性	可以为数据对象添加扩展属性，用于租户的定制化扩展业务。	• 控制台 • API
扩展类型	可以灵活定义对象的扩展类型，用于标识数据对象是否可被扩展。	• 控制台 • API

功能	描述	使用方式
生命周期管理	可以通过生命周期模板定义及管理，实现数据对象的全过程管理，以及对不同类型业务对象的过程追溯。	● 控制台 ● API
失效管理	可以灵活管理用户权限、业务属性（如零部件的损坏），便于用户识别。	● 控制台 ● API
所有者管理	可以通过所有者属性为数据对象授权。	● 控制台 ● API
标签管理	可以对已有数据对象从不同维度进行规划和分类管理。	● 控制台 ● API
多维视图&多维分支	可以通过不同视角（如设计、工艺、制造、服务等）对同一个数据对象（如BOM编码）进行差异化管理。	API **说明** xDM-F只能对M-V模型数据实体设置多维视图&多维分支能力，只支持通过API方式创建多维视图&多维分支数据实例。
数据分类管理	方便用户将同维度的数据实例进行分门别类管理、实例数据统一归集。	● 控制台 ● API **说明** 仅支持通过API方式将数据分类和数据分类对象进行关联。
结构化文档管理	融合文档编辑、文档管理、文档协作、知识共享等功能于一体，帮助个人和团队更便捷地创作、共享和管理文档。	● 控制台 ● API

2 准备工作

资源环境要求

- 已**开通iDME设计服务**。
- 已**购买数据建模引擎**。

权限要求

使用iDME时，需要对用户进行授权。建议您通过IAM用户使用iDME服务。IAM用户可以帮助您安全的控制华为云资源的访问。

具体操作请参见**创建iDME操作用户**和**用户管理**。

技能要求

- 了解iDME的基本概念、使用场景、使用方式等。
- 熟悉Java语言，能够编写Java语言代码。
- 熟悉Maven构建方式和使用方法。

开发环境要求

须知

开发过程中，您有任何问题可以在**华为云工业数字模型驱动引擎论坛**中发帖求助。

- 从**Oracle官网**下载并安装推荐使用的JDK版本。
  推荐使用的JDK版本：JDK 8 以上版本。
- 根据开发需要，下载并安装开发工具。
  - 从**Eclipse官网**下载并安装Eclipse IDE for Java Developers最新版本。
  - 从**IntelliJ IDEA官网**下载并安装IntelliJ IDEA开发工具。

3 构建 iDME 应用

应用即一个App，是一个实现了某种业务管理的可运行应用程序。创建应用是在iDME中开发项目的第一步，也是端到端构建软件应用的入口。在开发项目工程前，请先创建一个应用，再在应用设计态中创建数据模型，在应用运行态使用数据服务。

操作步骤

步骤1 创建应用，用于对软件应用的设计和构建。

iDME提供控制台和API两种创建方式，具体操作请参见**创建应用（控制台）**和**创建应用（API）**。

步骤2 在应用设计态构建并发布应用。

- 构建应用：对业务规则进行设计和开发，具体操作请参见**数据建模引擎用户指南**。
- 发布应用：生成相应代码包，具体操作请参见**发布应用**。

步骤3 部署应用，生成应用运行态，用于多租户应用集成测试及数据服务调用。

iDME提供控制台和API两种部署方式，具体操作请参见**部署应用（控制台）**和**部署应用（API）**。

----结束

下一步操作

当您完成应用构建后，即可在应用运行态对设计态的数据模型进行全生命周期管理和控制，或根据用户的定制化需求进行模型扩展、搜索服务定义等操作。常用场景包括：

- **使用全量数据服务API**
- **使用多维视图&多维分支功能**
- **使用生命周期管理**
- **使用搜索服务管理**

4 使用全量数据服务 API

4.1 全量数据服务概述

简介

为提升应用开发人员的开发效率，工业数字模型驱动引擎-数据建模引擎（xDM Foundation，简称xDM-F）提供对应用全量数据服务的管理功能，支持查看所有可调用API的信息，支持调用应用部署后的所有API。

全量数据服务API包括系统管理API、数据模型生成API、定制API的基本信息、参数信息和示例。应用开发人员可以在应用运行态的**全量数据服务**快速获取应用下所有类型API的清单、查看API详情、导出全量数据服务API，提升模型的消费速度和API调用速度，快速使能工业应用。

使用方法

请参见**4.2 常用全量数据服务API开发示例**。

4.2 常用全量数据服务 API 开发示例

4.2.1 创建指定实体的实例

功能介绍

在应用设计态完成模型的构建、发布（模型发布和应用发布），以及控制台部署应用后，会在应用运行态自动生成相应的CRUD接口。其中，create接口用于创建模型实例。

更多API信息请参见**全量数据服务**。

URI

- URI格式：

  POST http://**{Endpoint}**/rdm_**{appID}**_app/services/dynamic/api/**{entityName}**/create

- 参数说明：

表 4-1 URI 参数说明

参数	是否必填	参数类型	描述
Endpoint	是	String	承载REST服务端点的服务器域名或IP地址。
appID	是	String	应用ID。
entityName	是	String	实体的英文名称。

请求参数

表 4-2 请求 body 参数

参数	是否必填	参数类型	描述
id	是	String	唯一编码。
rdmVersion	是	Int	系统版本。
rdmExtensionType	否	String	实体类型。
createTime	否	Date	创建时间。
creator	否	String	创建者。
lastUpdateTime	否	Date	最后更新时间。
modifier	否	String	更新者。
tenant	否	Object	租户。

根据实体类型、功能的不同，请求body参数不同，您可以在**全量数据服务**查看API具体参数。

响应参数

返回创建实例的详细信息。

请求示例

示例背景

您已在设计态构建一个名称为“Design_document”的数据实体，并完成了实体发布、应用发布和应用部署。根据运行态的“数据服务管理 > 全量数据服务”查看到的“Design_document”数据实体Create接口信息，传入相应的参数。

示例代码

```
{
   "params": {
      "clsAttrs": null,
      "department": null,
      "income": null,
      "securityLevel": "internal",
      "kiaguid": null,
      "name": "产品设计",
      "description": null,
      "tenant": {
         "id": "-1",
         "clazz": "Tenant"
      },
      "rdmExtensionType": null,
      "id": null
   }
}
```

响应示例

```
{
   "result": "SUCCESS",
   "data": [
      {
         "id": "502903940749205504",
         "creator": "TestAccount2 5547b9adee54423cb**********",
         "modifier": "TestAccount2 5547b9adee54423cb**********",
         "createTime": "2023-05-24T10:00:42.978+0000",
         "lastUpdateTime": "2023-05-24T10:00:42.978+0000",
         "rdmVersion": 1,
         "rdmDeleteFlag": 0,
         "rdmExtensionType": "Design_document",
         "tenant": {
            "id": "-1",
            "creator": "xdmAdmin",
            "modifier": "xdmAdmin",
            "createTime": "2022-12-01T11:24:39.000+0000",
            "lastUpdateTime": "2022-12-01T11:24:39.000+0000",
            "rdmVersion": 1,
            "rdmDeleteFlag": 0,
            "rdmExtensionType": "Tenant",
            "tenant": null,
            "className": "Tenant",
            "name": "basicTenant",
            "description": "默认租户",
            "kiaguid": null,
            "securityLevel": "internal",
            "code": "basicTenant",
            "disableFlag": false,
            "dataSource": "DefaultDataSource"
         },
         "className": "Design_document",
         "name": "产品设计",
         "description": null,
         "kiaguid": null,
         "securityLevel": "internal",
         "income": null,
         "department": null
      }
   ],
   "errors": []
}
```

4.2.2 查询满足条件的实例数据

功能介绍

在应用设计态完成模型的构建、发布（模型发布和应用发布），以及控制台部署应用后，会在应用运行态自动生成相应的CRUD接口。其中，find接口用于分页查询实例信息。

更多API信息请参见**全量数据服务**。

URI

- URI格式：

  POST http://**{Endpoint}**/rdm_**{appID}**_app/services/dynamic/api/**{entityName}**/find/**{pageSize}**/**{curPage}**

- 参数说明：

  表 4-3 URI 参数说明

参数	是否必填	参数类型	描述
Endpoint	是	String	承载REST服务端点的服务器域名或IP地址。
appID	是	String	应用ID。
entityName	是	String	实体的英文名称。
pageSize	否	String	分页查询时，每页最多展示的记录数。
curPage	否	String	分页查询的页数。

请求参数

根据实体类型、功能的不同，请求body参数不同，您可以在**全量数据服务**查看API具体参数。为篇幅起见，这里只展示部分内容。

表 4-4 请求 body 参数

参数	是否必填	参数类型	描述
sorts	否	List	排序。
sort	否	String	排序方式（asc/desc）。

参数	是否必填	参数类型	描述
orderBy	否	String	排序字段，可填写模型自身属性、参考对象的属性、扩展属性及分类属性。
filter	否	Object	查询条件。
isNeedTotal	否	Boolean	是否需要查询总记录数（false/true）。

响应参数

返回模型所有属性、直接关联的参考对象、扩展属性、分类属性、级联的数据等。

请求示例

示例背景

- 已在设计态构建一个名称为“employee”的数据实体，该实体自定义了“age”和“gender”两个属性，并完成了实体发布、应用发布和应用部署。
- 已在运行态通过控制台或API的方式创建多个数据实例。

需要查询“gender”为“男性”的数据实例，且结果需要先按名称倒序排序，再按年龄倒序排序。

示例代码

```
{
   "params": {
      "sorts": [
         {
            "sort": "DESC",
            "orderBy": "name"
         },
         {
            "sort": "DESC",
            "orderBy": "age"
         }
      ],
      "filter": {
         "joiner": "and",
         "conditions": [
            {
               "conditionName": "gender",
               "operator": "=",
               "conditionValues": [
                  "男"
               ]
            }
         ]
      },
      "isNeedTotal": true
   }
}
```

响应示例


```
{
   "result": "SUCCESS",
   "data": [
      {
         "id": "455304645330341888",
         "creator": "test1 3c03e719256a427eb9277b64fcXXXXXX",
         "createTime": "2023-01-13T01:38:07.000+00:00",
         "modifier": "test1 3c03e719256a427eb9277b64fcXXXXXX",
         "lastUpdateTime": "2023-01-13T01:38:07.000+00:00",
         "rdmVersion": 1,
         "rdmExtensionType": "People",
         "rdmDeleteFlag": 0,
         "tenant": {
            "id": "-1",
            "clazz": "Tenant"
         },
         "className": "People",
         "name": "李四",
         "description": null,
         "kiaguid": null,
         "securityLevel": "internal",
         "gender": "男",
         "age": 20
      },
      {
         "id": "455304534248394752",
         "creator": "test1 3c03e719256a427eb9277b64fcXXXXXX",
         "createTime": "2023-01-13T01:37:40.000+00:00",
         "modifier": "test1 3c03e719256a427eb9277b64fcXXXXXX",
         "lastUpdateTime": "2023-01-13T01:37:40.000+00:00",
         "rdmVersion": 1,
         "rdmExtensionType": "People",
         "rdmDeleteFlag": 0,
         "tenant": {
            "id": "-1",
            "clazz": "Tenant"
         },
         "className": "People",
         "name": "张三",
         "description": null,
         "kiaguid": null,
         "securityLevel": "internal",
         "gender": "男",
         "age": 18
      }
   ],
   "errors": [],
   "pageInfo": {
      "curPage": 1,
      "pageSize": 20,
      "totalRows": 2,
      "totalPages": 1
   }
}
```


4.2.3 分页查询指定用户的实例数据

操作场景

本文介绍如何通过鉴权接口和分页查询接口实现数据对象的权限过滤。

假设，在cn-north-4区域创建一个应用（TestApp），应用ID为01a2b2c4764d4e00f123g345fd9baa9f，且已部署至数据建模引擎。应用中存在一个数据实体（DataEntity1），现在需要分页查询该数据实体的所有实例数据，并过滤指定用户权限的实例数据。

URI

- Find接口

  URL格式：POST http://**{Endpoint}**/rdm_**{appID}**_app/publicservices/api/**{entityName}**/find/**{pageSize}**/**{curPage}**

- batchHasAccess接口

  URL格式：POST http://**{Endpoint}**/rdm_**{appID}**_app/services/rdm/basic/api/AccessService/batchHasAccess

- 参数说明：

  表 4-5 URI 参数说明

参数	是否必填	参数类型	描述
Endpoint	是	String	承载REST服务端点的服务器域名或IP地址。
appID	是	String	应用ID。
entityName	是	String	实体的英文名称。
pageSize	否	String	分页查询时，每页最多展示的记录数。
curPage	否	String	分页查询的页数。

步骤 1：分页查询 DataEntity1 实体的所有实例数据

步骤1 在Request Header中增加“X-Auth-Token”，值为用户Token。

步骤2 在Request Header中增加“Content-Type”，值为“application/json”。

步骤3 在Request Body中传入参数如下：

```
{
   "params":{
      "sort":"DESC",
      "orderBy":"lastUpdateTime",
      "filter": {
          "joiner":"and",
              "conditions":[]
      },
      "isNeedTotal":false
   }
}
```

- sort：排序类型，也可为空。
- orderBy：填写需要按哪个字段进行排序，可填写模型自身属性、参考对象的属性、扩展属性及分类属性，也可为空。
- filter：填写过滤条件，可为空。

步骤4 发送POST“https://dme.cn-north-4.huaweicloud.com/rdm_01a2b2c4764d4e00f123g345fd9baa9f_app/publicservices/api/DataEntity1/find/20/1”。

若请求失败，会返回错误码及对应的错误信息说明。

----结束

步骤 2：基于分页查询的实例数据，指定用户过滤实例数据

步骤1 在Request Header中增加“X-Auth-Token”，值为用户Token。

步骤2 在Request Header中增加“Content-Type”，值为“application/json”。

步骤3 在Request Body中传入参数如下：

```
{
   "params":{
      "userId":"490183140267008000",
      "rdmExtensionType":" DataEntity1",
      "operations":["000000005"],
      "ids":["490930153786974208", "23213", "20230414300", "9898989898989898"]
   }
}
```

- userId：被鉴权用户在XDMUser实体实例中的ID，不可为空。
- rdmExtensionType：被鉴权实例的实体类型，不可为空。
- operations：操作列表，不可为空。
- ids：被鉴权的实例ID列表，不可为空。

步骤4 发送POST“https://dme.cn-north-4.huaweicloud.com/rdm_01a2b2c4764d4e00f123g345fd9baa9f_app/services/rdm/basic/api/AccessService/batchHasAccess”。

若请求失败，会返回错误码及对应的错误信息说明。

----结束

4.2.4 比较 M-V 模型的版本对象

操作场景

在应用设计态完成模型的构建、发布（模型发布和应用发布），以及控制台部署应用后，会在应用运行态自动生成相应的接口。本文介绍如何通过版本对象比较接口（compareBusinessVersion）实现比较不同版本对象间的基本属性、扩展属性和关联关系。

URI

- URL格式：

  POST http://**{Endpoint}**/rdm_**{appID}**_app/publicservices/api/**{entityName}**/compareBusinessVersion

- 参数说明：

表 4-6 URI 参数说明

参数	是否必填	参数类型	描述
Endpoint	是	String	承载REST服务端点的服务器域名或IP地址。
appID	是	String	应用ID。
entityName	是	String	实体的英文名称。

请求参数

表 4-7 请求 body 参数

参数	是否必填	参数类型	描述
id	是	String	主对象ID，用于标识版本对象。
basicVersion	是	String	基础版本对象的版本号。
correlationVersion	是	String	待比较的版本对象的版本号。

为篇幅起见，这里只展示部分内容。更多参数信息，您可以在**全量数据服务**进行查看。

响应参数

返回两个版本对象的对比结果。

请求示例

示例背景

您已在cn-north-4区域的TestApp应用中，构建一个名称为“TestMV”的M-V模型数据实体，并完成了实体发布、应用发布和应用部署。希望对主对象ID为“492060584863342592”的A.1版本和A.2版本进行比较。

示例代码

```
{
   "params":{
      "id":"492060584863342592",
      "basicVersion":"A.1",
      "correlationVersion":"A.2"
   }
}
```

响应示例

为篇幅起见，这里只展示部分内容。

```
{
   "result": "SUCCESS",
   "data": [
      {
        ......    //返回基础版本对象版本号A.1的所有内容
      },
      {
         ......    //返回待比较版本对象版本号A.2的基本属性的区别
         "relations": [
      ......    //返回待比较版本对象版本号A.2的关联关系的区别（所有区别都是A.2与A.1对比后A.2的区别）。如需对比A.1的区别，传参时调换A.1和A.2的顺序，将A.2作为基础版本对象版本号，A.1作为待比较的版本对象的版本号即可。
         ],
         "extAttrs": {
            ......    //返回待比较版本对象版本号A.2的扩展属性的区别
         },
         "latest": true,
         "lastUpdateTime": "2023-04-27 14:35:56"
      }
   ],
   "errors": []
}
```

4.2.5 另存版本对象的实例数据

功能介绍

版本对象的另存为接口（saveAs）用于创建一条与原版本对象实例数据相同的数据实例。该实例数据会完全复制原实例现有的数据，包括与其关联的主对象和分支对象，且新实例数据的版本号从初始值开始计算。

URI

- URL格式：

  POST http://**{Endpoint}**/rdm_**{appID}**_app/publicservices/api/**{entityName}**/saveAs

- 参数说明：

  表 4-8 URI 参数说明

参数	是否必填	参数类型	描述
Endpoint	是	String	承载REST服务端点的服务器域名或IP地址。
appID	是	String	应用ID。
entityName	是	String	实体的英文名称。

请求参数

表 4-9 请求 body 参数

参数	是否必填	参数类型	描述
sourceInstanceId	是	String	version.唯一编码。
sourceMasterId	是	String	master.唯一编码。

传入参数时，必填参数需至少传入一个，即至少传入sourceInstanceId参数或sourceMasterId参数。为篇幅起见，这里只展示部分内容。更多参数信息，您可以在**全量数据服务**进行查看。

响应参数

返回创建的数据实例信息。

请求示例

示例背景

您已在cn-north-4区域的TestApp应用中，构建一个名称为“TestMV”的M-V模型数据实体，并完成了实体发布、应用发布和应用部署。希望可以根据指定的版本对象唯一编码（481785585908850688）或主对象唯一编码（481785532712488960），创建一个新的数据实例。

示例代码

为篇幅起见，这里只展示必填参数。

```
{
   "params":{
            ......
      "sourceInstanceId":"481785585908850688",
      "sourceMasterId":"481785532712488960"
   }
}
```

响应示例

为篇幅起见，这里只展示部分内容。

```
{
   "result": "SUCCESS",
   "data": [
      {
         "id": "515167665271083008",
         "creator": "TestAccount2 5547b9adee54423cb**********",
         "modifier": "TestAccount2 5547b9adee54423cb**********",
         "createTime": "2023-06-27T05:57:56.000+0000",
         "lastUpdateTime": "2023-06-27T06:12:22.752+0000",
         "rdmVersion": 1,
         "rdmDeleteFlag": 0,
         "rdmExtensionType": "TestMV",
         "tenant": {
            ......
         },
```

```
            "className": "TestMV",
            "master": {
                "id": "515167665275277312",
                "tenant": {
                    ......
                },
                "className": "TestMVMaster"
            },
            "branch": {
                "id": "515167665313026048",
                "creator": "TestAccount2 5547b9adee54423cb**********",
                "modifier": "TestAccount2 5547b9adee54423cb**********",
                "createTime": "2023-06-27T06:12:22.774+0000",
                "lastUpdateTime": "2023-06-27T06:12:22.774+0000",
                "rdmVersion": 1,
                "rdmDeleteFlag": 0,
                "rdmExtensionType": "TestMVBranch",
                "tenant": {
                    ......
                },
                "className": "TestMVBranch",
                "version": "A"
            },
            "latest": true,
            "versionCode": 1,
            "iteration": 1,
            "version": "A",
            "workingState": {
                "code": "CHECKED_IN",
                "cnName": "已检入",
                "enName": "checked in",
                "alias": "CHECKED_IN"
            },
            "checkOutUserName": null,
            "checkOutTime": null,
            "preVersionId": null
        }
    ],
    "errors": []
}
```


4.2.6 创建多维版本的数据实例

功能介绍

版本对象的创建视图接口（createView）和批量创建视图接口（batchCreateView）是根据M-V模型实体已有数据实例的versionId（version.唯一编码）并指定多版本属性进行创建。创建后该M-V模型实体会生成实例数据完全复制原数据实例现有信息，但“version.修订版本”和“branch.大版本”会从初始值重新开始计算的多维版本数据实例。

本章节以createView为例，如需调用batchCreateView，请前往**全量数据服务**查看。

URI

- URL格式：

  POST http://**{Endpoint}**/rdm_**{appID}**_app/publicservices/api/**{entityName}**/createView

- 参数说明：

表 4-10 URI 参数说明

参数	是否必填	参数类型	描述
Endpoint	是	String	承载REST服务端点的服务器域名或IP地址。
appID	是	String	应用ID。
entityName	是	String	实体的英文名称。

前提条件

- 已**获取用户Token**。
- 已获取Endpoint值（数据建模引擎所在域名或IP地址）和应用ID。
- 已创建一个M-V模型实体并在“功能配置”的“多维版本”中配置了多维版本属性（如“view2”），详情请参见**创建数据实体**。
- 已在应用运行态为M-V模型实体创建一个数据实例，详情请参见**创建数据实例**。

请求参数

表 4-11 请求 body 参数

参数	是否必填	参数类型	描述
versionId	是	String	原视图的versionId，即已创建数据实例的version.唯一编码。
workCopyType	否	Object	此配置项用于创建多维版本数据实例时是否继承其对应的关系实例。根据业务需求，选择类型。 • BOTH：若存在关系实例引用此数据实例作为源端实例或目标端实例，创建后的数据实例将继承这些关系实例。 • SOURCE：若存在关系实例引用此数据实例作为源端实例，创建后的数据实例将继承这些关系实例。 • TARGET：若存在关系实例引用此数据实例作为目标端实例，创建后的数据实例将继承这些关系实例。 • NONE：创建后的数据实例将不继承任何关系实例。 • CUSTOM：若指定的关系实体集合对应的关系实例引用此数据实例作为源端实例或目标端实例，创建后的数据实例将继承这些关系实例。

参数	是否必填	参数类型	描述
customLinkSet	否	List	关系实体名称的集合。 当“workCopyType”设置为“CUSTOM”时，需要设置此参数。
needSetNull	否	List	指定不复制的属性。被指定不复制的属性，其返回值将被设置为“null”。
view2	否 **说明** 如果用户在应用设计态配置“多维版本”功能配置时，将此多维版本属性设置为必填，则请求参数为必填项。	Object	自定义的多维版本属性。
modifier	是	String	更新者。

响应参数

返回创建的多维版本数据实例。

请求示例

示例背景

您已在cn-north-4区域的TestApp应用中，构建一个名称为“TestMV”、多维版本属性为“view2”的M-V模型数据实体，并完成了实体发布、应用发布和应用部署。希望可以根据已创建的M-V模型数据实例，创建一个多维版本数据实例。

示例代码

```
{
   "params": {
      "versionId": "521722330943061234",
      "modifier": "DME_Developer",
      "view2": {
         "id": "11",
         "clazz": "ViewAttr"
      }
   }
}
```

响应示例

为篇幅起见，这里只展示部分内容。其中，**"version": "A"**表示多维视图的迭代版本。

```
{
   "result": "SUCCESS",
   "data": [
```

```
    {
        "id": "523616294595981234",
        ......
        "rdmVersion": 1,
        "rdmDeleteFlag": 0,
        "rdmExtensionType": "View2",
        "tenant": {
            ......
        },
        "className": "View2",
        "name": "AS",
        "description": null,
        "kiaguid": null,
        "securityLevel": "internal",
        "master": {
            ......
            "rdmExtensionType": "View2Master",
            "tenant": {
                ......
            },
            "className": "View2Master"
        },
        "branch": {
            ......
            "rdmVersion": 1,
            "rdmDeleteFlag": 0,
            "rdmExtensionType": "View2Branch",
            "tenant": {
                ......
            },
            "className": "View2Branch",
            "version": "A"
        },
        "latest": true,
        "latestIteration": true,
        "versionCode": 1,
        "iteration": 1,
        "version": "A",
        "latestVersion": true,
        "workingCopy": false,
        "workingState": {
            "code": "CHECKED_IN",
            "cnName": "已检入",
            "enName": "checked in",
            "alias": "CHECKED_IN"
        },
        "checkOutUserName": null,
        "checkOutTime": null,
        "preVersionId": "52281****087179264",
        "viewAttr1": null,
        "viewAttr3": null,
        "viewAttr2": null
    }
  ],
  "errors": []
}
```


4.2.7 修订多维版本的数据实例

功能介绍

创建多维版本的数据实例后，您可以在需要调用接口的请求参数中传入该数据实例的多维版本属性，用于多维版本管理数据实例。

本章节以修订接口（revise）为例，指导您如何使用多维版本&多维分支功能。

URI

- URL格式：
  POST http://**{Endpoint}**/rdm_**{appID}**_app/publicservices/dynamic/api/{**entityName**}/revise
- 参数说明：

  表 4-12 URI 参数说明

参数	是否必填	参数类型	描述
Endpoint	是	String	承载REST服务端点的服务器域名或IP地址。
appID	是	String	应用ID。
entityName	是	String	实体的英文名称。

前提条件

- 已**获取用户Token**。
- 已获取Endpoint值（数据建模引擎所在域名或IP地址）和应用ID。
- 已**4.2.6 创建多维版本的数据实例**。

请求参数

表 4-13 请求 body 参数

参数	是否必填	参数类型	描述
creator	否	String	创建者。
modifier	否	String	更新者。
masterId	是	String	主对象ID。

参数	是否必填	参数类型	描述
workCopyType	否	Object	此配置项用于创建多维版本数据实例时是否继承其对应的关系实例。根据业务需求，选择类型。 ● BOTH：若存在关系实例引用此数据实例作为源端实例或目标端实例，创建后的数据实例将继承这些关系实例。 ● SOURCE：若存在关系实例引用此数据实例作为源端实例，创建后的数据实例将继承这些关系实例。 ● TARGET：若存在关系实例引用此数据实例作为目标端实例，创建后的数据实例将继承这些关系实例。 ● NONE：创建后的数据实例将不继承任何关系实例。 ● CUSTOM：若指定的关系实体集合对应的关系实例引用此数据实例作为源端实例或目标端实例，创建后的数据实例将继承这些关系实例。
workingCopy	否	Boolean	是否工作副本，默认值false。
customLinkSet	否	List	关系实体名称的集合。 当“workCopyType”设置为“CUSTOM”时，需要设置此参数。
needSetNull	否	List	指定不复制的属性。被指定不复制的属性，其返回值将被设置为“null”。
view2	否 **说明** 如果用户在应用设计态配置“多维版本”功能配置时，将此多维版本属性设置为必填，则请求参数为必填项。	Object	自定义的多维版本属性。
view2.id	是	String	多维版本属性的ID。
view2.clazz	否	String	多维版本属性的类名。

响应参数

返回修订后的多维版本数据实例。

请求示例

示例背景

- 已在cn-north-4区域的TestApp应用中，构建一个名称为“TestMV”、多维版本属性为“view2”的M-V模型数据实体，并完成了实体发布、应用发布和应用部署。
- 已在应用运行态通过可视化页面或API的方式创建一个多维版本的数据实例，其数据实例的唯一编码为“523616294595981234”。

您的业务数据存在较大变更，希望可以创建新的版本，差异化管理实例数据。

示例代码

```
{
   "params": {
      "modifier": "XDM_Developer",
      "masterId": "637652965003370496",
      "workCopyType": "BOTH",
      "view2": {
         "id": "4"
      }
   }
}
```

响应示例

```
{
   "result": "SUCCESS",
   "data": [
      {
         "id": "637773750976122880",
         "creator": "XDM_Developer",
         "modifier": "XDM_Developer",
         "createTime": "2024-05-30T14:05:11.504+0000",
         "lastUpdateTime": "2024-05-30T14:05:11.504+0000",
         "rdmVersion": 1,
         "rdmDeleteFlag": 0,
         "rdmExtensionType": "MultiViewMv",
         "tenant": {
            "id": "-1",
            "creator": "xdmAdmin",
            "modifier": "xdmAdmin",
            "createTime": "2024-05-30T01:11:34.110+0000",
            "lastUpdateTime": "2024-05-30T01:11:34.110+0000",
            "rdmVersion": 1,
            "rdmDeleteFlag": 0,
            "rdmExtensionType": "Tenant",
            "tenant": null,
            "className": "Tenant",
            "name": "basicTenant",
            "description": "默认租户",
            "kiaguid": null,
            "securityLevel": "internal",
            "nameEn": null,
            "code": "basicTenant",
            "disableFlag": false,
            "dataSource": "DefaultDataSource"
         },
         "className": "MultiViewMv",
         "name": "1",
         "description": null,
         "kiaguid": null,
         "securityLevel": "internal",
         "master": {
            "id": "637652965003370496",
            "creator": "XDM_Developer",
```

```
        "modifier": "XDM_Developer",
        "createTime": "2024-05-30T06:05:13.911+0000",
        "lastUpdateTime": "2024-05-30T06:14:31.070+0000",
        "rdmVersion": 2,
        "rdmDeleteFlag": 0,
        "rdmExtensionType": "MultiViewMvMaster",
        "tenant": {
            "id": "-1",
            "creator": "xdmAdmin",
            "modifier": "xdmAdmin",
            "createTime": "2024-05-30T01:11:34.110+0000",
            "lastUpdateTime": "2024-05-30T01:11:34.110+0000",
            "rdmVersion": 1,
            "rdmDeleteFlag": 0,
            "rdmExtensionType": "Tenant",
            "tenant": null,
            "className": "Tenant",
            "name": "basicTenant",
            "description": "默认租户",
            "kiaguid": null,
            "securityLevel": "internal",
            "nameEn": null,
            "code": "basicTenant",
            "disableFlag": false,
            "dataSource": "DefaultDataSource"
        },
        "className": "MultiViewMvMaster"
    },
    "branch": {
        "id": "637773751051620352",
        "creator": "XDM_Developer",
        "modifier": "XDM_Developer",
        "createTime": "2024-05-30T14:05:11.496+0000",
        "lastUpdateTime": "2024-05-30T14:05:11.496+0000",
        "rdmVersion": 1,
        "rdmDeleteFlag": 0,
        "rdmExtensionType": "MultiViewMvBranch",
        "tenant": {
            "id": "-1",
            "creator": "xdmAdmin",
            "modifier": "xdmAdmin",
            "createTime": "2024-05-30T01:11:34.110+0000",
            "lastUpdateTime": "2024-05-30T01:11:34.110+0000",
            "rdmVersion": 1,
            "rdmDeleteFlag": 0,
            "rdmExtensionType": "Tenant",
            "tenant": null,
            "className": "Tenant",
            "name": "basicTenant",
            "description": "默认租户",
            "kiaguid": null,
            "securityLevel": "internal",
            "nameEn": null,
            "code": "basicTenant",
            "disableFlag": false,
            "dataSource": "DefaultDataSource"
        },
        "className": "MultiViewMvBranch",
        "version": "C"
    },
    "latest": true,
    "latestIteration": true,
    "versionCode": 3,
    "iteration": 1,
    "version": "C",
    "latestVersion": true,
    "workingCopy": false,
    "workingState": {
        "code": "CHECKED_IN",
```



```
            "cnName": "已检入",
            "enName": "checked in",
            "alias": "CHECKED_IN"
        },
        "checkOutUserName": null,
        "checkOutTime": null,
        "preVersionId": "637760894251569152",
        "view2": {
            "id": "4",
            "creator": "XDM_Developer",
            "modifier": "XDM_Developer",
            "createTime": "2024-05-30T07:17:01.666+0000",
            "lastUpdateTime": "2024-05-30T07:17:01.666+0000",
            "rdmVersion": 1,
            "rdmDeleteFlag": 0,
            "rdmExtensionType": "Entity",
            "tenant": {
                "id": "-1",
                "creator": "xdmAdmin",
                "modifier": "xdmAdmin",
                "createTime": "2024-05-30T01:11:34.110+0000",
                "lastUpdateTime": "2024-05-30T01:11:34.110+0000",
                "rdmVersion": 1,
                "rdmDeleteFlag": 0,
                "rdmExtensionType": "Tenant",
                "tenant": null,
                "className": "Tenant",
                "name": "basicTenant",
                "description": "默认租户",
                "kiaguid": null,
                "securityLevel": "internal",
                "nameEn": null,
                "code": "basicTenant",
                "disableFlag": false,
                "dataSource": "DefaultDataSource"
            },
            "className": "Entity",
            "name": null,
            "description": null,
            "kiaguid": null,
            "securityLevel": "internal",
            "ref": null,
            "lifecycleTemplate": null,
            "lifecycleState": null,
            "folder": null,
            "clsAttrs": null,
            "owner": [],
            "aclEntry": null,
            "rootNode": null,
            "parentNode": null,
            "leafFlag": true,
            "fullPath": "4/",
            "rawFullPath": "4/",
            "disableFlag": false,
            "extAttrs": [],
            "extAttrMap": {}
        },
        "lifecycleTemplate": null,
        "lifecycleState": null,
        "disableFlag": false,
        "aclEntry": null,
        "clsAttrs": null,
        "owner": [],
        "extAttrs": [],
        "extAttrMap": {}
    }
  ],
  "errors": []
}
```


5 使用多维视图&多维分支功能

5.1 多维视图&多维分支概述

简介

针对同一编码承载不同时期的BOM结构等同一业务编码精细化管理不同时期的版本数据场景，工业数字模型驱动引擎-数据建模引擎（xDM Foundation，简称xDM-F）提供多维视图&多维分支功能，用以满足不同领域对版本数据的差异化管理需求。例如，在试制视图、生产发行视图和制造视图下，同一个物料编码可关联不同的BOM清单。

使用流程

1. 在应用设计态创建一个具有“多维视图&多维分支”功能的M-V模型实体，并配置了“多维版本”功能配置，具体操作请参见**创建数据实体**。
2. 依次完成“发布数据实体 > 发布应用 > 部署应用”的操作，具体操作请参见**发布数据实体**、**发布应用**和**部署应用**。
3. 通过全量数据服务API使用“多维视图&多维分支”功能，具体操作请参见**全量数据服务**。

使用方法

请参见**5.2 多维视图&多维分支开发指导**。

5.2 多维视图&多维分支开发指导

本文指导您通过全量数据服务API使用多维视图&多维分支功能。

涉及的接口

请参见**全量数据服务**。

前提条件

已在应用设计态创建一个M-V模型实体，并在“功能配置”的“多维版本”中配置了多维版本属性，详情请参见**创建数据实体**。

示例：M-V模型实体的英文名称为“TestMV”，多维版本属性的英文名称为“view2”。

为已有数据实例创建多维版本

步骤1 在应用运行态获取M-V模型实体“TestMV”的版本对象数据实例信息，详情请参见**查询数据实例**。

您也可以通过API方式获取该实体的版本对象数据实例，详情请参见**4.2.2 查询满足条件的实例数据**。

例如，获取一个“version.唯一编码”为“521722330943061234”的数据实例。

步骤2 在应用运行态获取创建多维视图的API信息，具体操作请参见**全量数据服务**。

步骤3 根据实际的业务需求，为数据实例创建多维视图，具体操作请参见**4.2.6 创建多维版本的数据实例**。

----结束

管理指定多维版本的数据实例

您可以根据实际的业务需求，调用M-V模型实体的相关API，在API的请求参数中传入指定多维版本属性，即可对数据实例进行多维版本管理。如下操作以修订接口（revise）为例。

步骤1 在应用运行态获取M-V模型实体“TestMV”的版本对象数据实例信息，详情请参见**查询数据实例**。

您也可以通过API方式获取该实体的版本对象数据实例，详情请参见**4.2.2 查询满足条件的实例数据**。

例如，获取一个“version.唯一编码”为“521722330943061234”的数据实例。

步骤2 在应用运行态获取修订该版本对象数据实例的API信息，具体操作请参见**全量数据服务**。

步骤3 根据实际的业务需求，调用API修订指定多维版本的数据实例，具体操作请参见**4.2.7 修订多维版本的数据实例**。

----结束

6 使用生命周期管理

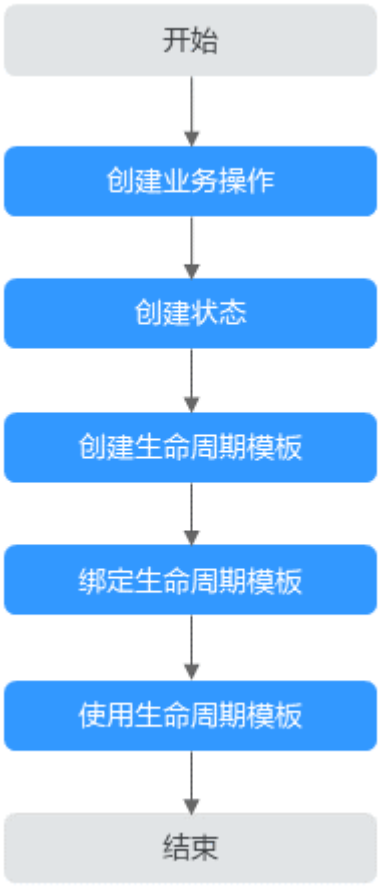
6.1 生命周期管理概述

简介

为了方便应用开发人员对不同的业务对象定义不同的生命周期模板及访问规则、方便业务设计人员跟踪/流转业务对象状态，iDME提供生命周期管理功能。用户可以通过生命周期模板定义及管理实现对象的全过程管理，实现对不同类型业务对象的过程追溯，提升系统扩展性、灵活性，降低开发的人力及时间成本；也可以根据生命周期状态定义对象的动态访问权限，实现权限的灵活定义，提升应用开发人员的体验、降低应用运维成本。

使用流程

图 6-1 生命周期管理流程



使用方法

- 使用应用运行态
  a. 根据业务对象的流程标志，依次创建业务操作，具体操作请参见**创建业务操作**。
     例如，Charter开发流程的概念决策评审点（Concept Decision Check Point，CDCP）、计划决策评审点（Plan Decision Check Point，PDCP）等。
  b. 根据业务对象的流程阶段，依次创建状态，具体操作请参见**创建状态**。
     例如，IPD（Integrated Product Development，集成产品开发）流程的概念阶段、计划阶段、开发阶段、验证阶段、发布阶段和维护阶段。
  c. 根据业务对象的生命周期操作流程，创建生命周期模板，关联相关的业务操作和状态，具体操作请参见**创建生命周期模板**。
  d. 将具有生命周期管理功能的数据实体和刚创建的生命周期模板进行绑定，具体操作请参见**修改数据实体**。
     如果应用运行态没有生命周期管理功能的数据实体，请前往应用设计态创建一个具有生命周期管理功能的数据实体，并完成“发布该数据实体 > 发布应

用 > 部署应用”的操作后，再返回应用运行态通过修改数据实体绑定生命周期模板。

e. 根据实例数据的流转状态，灵活使用生命周期模板的业务操作和状态。

- 如果您的数据实例是单实体实例，可通过编辑的方式更新实例的生命周期阶段，具体操作请参见**修改数据实例**。
- 如果您的数据实例是M-V模型实体实例，可通过检入/检出的方式更新实例的生命周期阶段，具体操作请参见**检入数据实例（M-V模型）**和**检出数据实例（M-V模型）**。

- 使用全量数据服务API

  您可以通过**全量数据服务**提供的API使用生命周期管理，具体操作请参见**6.2 生命周期管理开发指导**。

6.2 生命周期管理开发指导

本文指导您通过全量数据服务API使用生命周期管理功能。

涉及的接口

表 6-1 涉及的接口

接口名称	说明
https://**{Endpoint}**/rdm_**{appID}**_app/services/rdm/basic/api/LifecycleBusinessOperation/create	创建业务操作。
https://**{Endpoint}**/rdm_**{appID}**_app/services/rdm/basic/api/LifecycleState/create	创建状态。
https://**{Endpoint}**/rdm_**{appID}**_app/services/rdm/basic/api/LifecycleTemplate/create	创建生命周期模板。
https://**{Endpoint}**/rdm_**{appID}**_app/services/rdm/basic/api/LifecycleTemplate/updateLifecycleInfo	为实体绑定生命周期模板。
https://**{Endpoint}**/rdm_**{appID}**_app/services/dynamic/api/**{entityName}**/update	修改数据实例，更新数据实例当前的生命周期阶段。
https://**{Endpoint}**/rdm_**{appID}**_app/services/dynamic/api/**{entityName}**/checkout	检出M-V模型的数据实例，更新数据实例当前的生命周期阶段。
https://**{Endpoint}**/rdm_**{appID}**_app/services/dynamic/api/**{entityName}**/checkin	检入M-V模型的数据实例，更新数据实例当前的生命周期阶段。

步骤 1：创建业务操作

步骤1 准备工作。

在应用运行态中获取创建业务操作的API信息，具体操作请参见**全量数据服务**。

步骤2 根据实际业务对象的生命周期管理流程，调用API创建业务操作。

- 请求示例

```
POST http://{Endpoint}/rdm_{appID}_app/services/basic/api/LifecycleBusinessOperation/create

{
   "params": {
      "name": "发布",
      "nameEn": "Publish",
      "description": "",
      "descriptionEn": ""
   }
}
```

其中，{Endpoint}、{appID}表示数据建模引擎所在域名/IP地址和应用ID，“name”、“nameEn”、“description”、“descriptionEn”表示业务操作的中英文名称和中英文描述。

- 响应示例

```
{
   "result": "SUCCESS",
   "data": [
      {
         "id": "504753788985286656",
         "creator": "XDM_Developer 93172bbfd0f644**********345386",
         "modifier": "XDM_Developer 93172bbfd0f644**********345386",
         "createTime": "2023-05-29T12:31:21.166+0000",
         "lastUpdateTime": "2023-05-29T12:31:21.166+0000",
         "rdmVersion": 1,
         "rdmDeleteFlag": 0,
         "rdmExtensionType": "LifecycleBusinessOperation",
         "tenant": {
            "id": "-1",
            "creator": "xdmAdmin",
            "modifier": "xdmAdmin",
            "createTime": "2023-05-29T11:27:42.000+0000",
            "lastUpdateTime": "2023-05-29T11:27:42.000+0000",
            "rdmVersion": 1,
            "rdmDeleteFlag": 0,
            "rdmExtensionType": "Tenant",
            "tenant": null,
            "className": "Tenant",
            "name": "basicTenant",
            "description": "默认租户",
            "kiaguid": null,
            "securityLevel": "internal",
            "code": "basicTenant",
            "disableFlag": false,
            "dataSource": "DefaultDataSource"
         },
         "className": "LifecycleBusinessOperation",
         "businessCode": "LOP00000001",
         "description": "",
         "descriptionEn": "",
         "nameEn": "Publish",
         "name": "发布",
         "disableFlag": false
      }
   ],
```

```
    "errors": []
}
```

----结束

步骤 2：创建状态

步骤1 准备工作。

在应用运行态中获取创建状态的API信息，具体操作请参见**全量数据服务**。

步骤2 根据实际业务对象的生命周期管理流程，调用API创建状态。

- 请求示例

```
POST http://{Endpoint}/rdm_{appID}_app/services/basic/api/LifecycleState/create

{
    "params": {
        "name": "已发布",
        "nameEn": "Published",
        "description": "",
        "descriptionEn": "",
        "internalName": "Published"
    }
}
```

其中，{Endpoint}表示数据建模引擎所在域名或IP地址，{appID}表示应用ID，“name”、“nameEn”、“description”、“descriptionEn”表示生命周期状态的中英文名称和中英文描述。

- 响应示例

```
{
    "result": "SUCCESS",
    "data": [
        {
            "id": "504966116662059008",
            "creator": "XDM_Developer 93172bbfd0f644**********345386",
            "modifier": "XDM_Developer 93172bbfd0f644**********345386",
            "createTime": "2023-05-30T02:35:04.029+0000",
            "lastUpdateTime": "2023-05-30T02:35:04.029+0000",
            "rdmVersion": 1,
            "rdmDeleteFlag": 0,
            "rdmExtensionType": "LifecycleState",
            "tenant": {
                "id": "-1",
                "creator": "xdmAdmin",
                "modifier": "xdmAdmin",
                "createTime": "2023-05-29T11:27:42.000+0000",
                "lastUpdateTime": "2023-05-29T11:27:42.000+0000",
                "rdmVersion": 1,
                "rdmDeleteFlag": 0,
                "rdmExtensionType": "Tenant",
                "tenant": null,
                "className": "Tenant",
                "name": "basicTenant",
                "description": "默认租户",
                "kiaguid": null,
                "securityLevel": "internal",
                "code": "basicTenant",
                "disableFlag": false,
                "dataSource": "DefaultDataSource"
            },
            "className": "LifecycleState",
            "descriptionEn": "",
            "businessCode": "LCS00000003",
            "name": "已发布",
            "description": "",
            "nameEn": "Published",
```

```
            "internalName": "Published",
            "disableFlag": false,
            "extAttrs": [],
            "extAttrMap": {}
        }
    ],
    "errors": []
}
```


----结束

步骤 3：创建生命周期模板

步骤1 准备工作。

在应用运行态中获取创建生命周期模板的API信息，具体操作请参见**全量数据服务**。

步骤2 根据实际业务对象的生命周期管理流程，调用API创建生命周期模板。

- 请求示例

```
POST http://{Endpoint}/rdm_{appID}_app/services/basic/api/LifecycleTemplate/create

{
    "params": {
        "name": "产品发布流程",
        "nameEn": "Product Release Process",
        "description": "",
        "descriptionEn": "",
        "master": {
            "id": null,
            "category": "",
            "name": "产品发布流程",
            "nameEn": "Product Release Process"
        },
        "branch": {
            "id": null
        }
    }
}
```

其中，{Endpoint}表示数据建模引擎所在域名或IP地址，{appID}表示应用ID，“master”、“branch”表示生命周期模板的主对象和分支对象。

- 响应示例


```
{
    "result": "SUCCESS",
    "data": [
        {
            "id": "504971863990996992",
            "className": "LifecycleTemplate",
            "name": "产品发布流程",
            "description": "",
            "master": {
                "id": "504971863999385600",
                "className": "LifecycleTemplateMaster",
                "businessCode": "LCT00000002",
                "nameEn": "Product Release Process",
                "name": "产品发布流程",
                "disableFlag": false
            },
            "branch": {
                "id": "504971863999385601",
                "className": "LifecycleTemplateBranch",
                "version": "A"
            },
            "latest": true,
            "latestIteration": true,
            "versionCode": 1,
```



```
            "iteration": 1,
            "version": "A",
            "latestVersion": true,
            "workingCopy": false,
            "workingState": {
                "code": "CHECKED_IN",
                "cnName": "已检入",
                "enName": "checked in",
                "alias": "CHECKED_IN"
            },
            "checkOutUserName": null,
            "checkOutTime": null,
            "preVersionId": null,
            "descriptionEn": "",
            "nameEn": "Product Release Process",
            "reserved": null,
            "lifecyclePhaseList": []
        }
    ],
    "errors": []
}
```


----结束

步骤 4：绑定生命周期模板

步骤1 准备工作。

在应用运行态中获取绑定生命周期模板的API信息，具体操作请参见**全量数据服务**。

步骤2 根据实际业务对象的生命周期管理，调用API绑定生命周期模板。

- 请求示例

```
POST http://{Endpoint}/rdm_{appID}_app/services/basic/api/LifecycleTemplate/updateLifecycleInfo

{
    "params": {
        "applicationId": "1bab18b61c6f4f5bbd031d2d8e0e3fda",
        "id": "478135618698743808",
        "curLifecycleTemplateAttached": [
            {
                "id": "63c7e2aad21a4715bd8ac08d73da4805",
                "templateAttrName": "生命周期模板",
                "templateAttrNameEn": "LifecycleTemplate",
                "templateAttrDescription": "生命周期模板",
                "templateAttrDescriptionEn": "LifecycleTemplate",
                "stateAttrName": "生命周期状态",
                "stateAttrNameEn": "LifecycleState",
                "stateAttrDescription": "生命周期状态",
                "stateAttrDescriptionEn": "LifecycleState",
                "attachTemplateId": "504971863990996992",
                "attachTemplateName": "Product Release Process",
                "attachTemplateNameEn": "产品发布流程"
            }
        ]
    }
}
```

其中，{Endpoint}表示数据建模引擎所在域名或IP地址，{appID}表示应用ID，“attachTemplateId”、“attachTemplateName”、“attachTemplateNameEn”表示需要绑定的生命周期模板ID、名称和英文名称。

- 响应示例


```
{
    "result": "SUCCESS",
    "data": [
        {
```



```
        "id": "478135618698743808",
        "rdmExtensionType": "TypeDefinition",
        "className": "TypeDefinition",
        ..... // 实体详细信息
        "lifecycleTemplateAttached": [
          {
            "attachTemplateId": "477901037156446208",
            "stateAttrDescriptionEn": "LifecycleState",
            "readOnly": true,
            "stateAttrName": "生命周期状态",
            "attachTemplateNameEn": "Test0316",
            "templateAttrDescriptionEn": "LifecycleTemplate",
            "templateAttrDescription": "生命周期模板",
            "stateAttrDescription": "生命周期状态",
            "id": "63c7e2aad21a4715bd8ac08d73da4805",
            "attachTemplateName": "Test0316",
            "templateAttrNameEn": "LifecycleTemplate",
            "templateAttrName": "生命周期模板",
            "stateAttrNameEn": "LifecycleState"
          }
        ],
        "curLifecycleTemplateAttached": [
          {
            "id": "63c7e2aad21a4715bd8ac08d73da4805",
            "templateAttrName": "生命周期模板",
            "templateAttrNameEn": "LifecycleTemplate",
            "templateAttrDescription": "生命周期模板",
            "templateAttrDescriptionEn": "LifecycleTemplate",
            "stateAttrName": "生命周期状态",
            "stateAttrNameEn": "LifecycleState",
            "stateAttrDescription": "生命周期状态",
            "stateAttrDescriptionEn": "LifecycleState",
            "attachTemplateId": "504971863990996992",
            "attachTemplateName": "Product Release Process",
            "attachTemplateNameEn": "产品发布流程"
          }
        ]
      }
    ],
    "errors": []
}
```

----结束

步骤 5：使用生命周期模板

步骤1 准备工作。

在应用运行态中获取更新生命周期阶段的API信息，具体操作请参见**全量数据服务**。

步骤2 根据实际业务对象当前的生命周期阶段，调用API更新生命周期。

- 请求示例

```
POST http://{Endpoint}/rdm_{appID}_app/services/dynamic/api/{entityName}/update

{
    "params": {
        ..... // 其他实体属性
        "lifecycleTemplate": {
            "id": "455805277254459392",
            "clazz": "LifecycleTemplate"
        },
        "lifecycleState": {
            "id": "455663911463555072",
            "clazz": "LifecycleState"
        },
        "tenant": {
            "id": "-1",
```

```
            "clazz": "Tenant"
        },
        "creator": "XDM_Developer 93172bbfd0f644**********345386",
        "modifier": "XDM_Developer",
        "id": "505046194565681152"
    }
}
```

其中，{Endpoint}表示数据建模引擎所在域名或IP地址，{appID}表示应用ID，{entityName}表示实体的英文名称，“lifecycleTemplate”、“lifecycleState”表示实例需要更新的生命周期模板和状态。

- 响应示例

```
{
    "result": "SUCCESS",
    "data": [
        {
            "id": "505046194565681152",
            ..... // 其他实例属性
            "disableFlag": false,
            "lifecycleTemplate": {
                "id": "455805277254459392",
                "className": "lifecycleTemplate"
                ..... // 其他生命周期模板属性
            },
            "lifecycleState": {
                "id": "455801165657935872",
                "className": "LifecycleState"
                ..... // 其他状态属性
            }
        }
    ],
    "errors": []
}
```

----结束

7 使用搜索服务管理

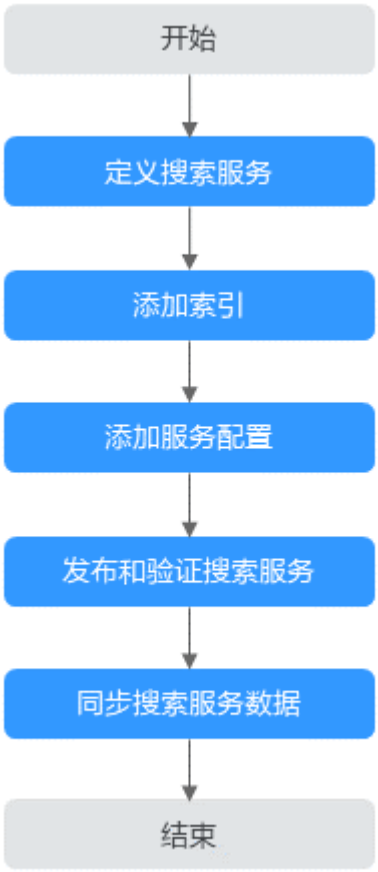
7.1 搜索服务管理概述

简介

在以往的工业应用开发场景中，常见的难题包括跨数据类型、跨数据库搜索难度大，业务数据种类多、搜索场景多、第三方搜索引擎在线应用无法在客户私有环境使用、图形搜索实现难度大等。为了满足应用开发人员研发效率的需求，工业数字模型驱动引擎-数据建模引擎（xDM Foundation，简称xDM-F）提供根据关键字查询特定目标并分页返回所有满足匹配要求的对象条目、按对象属性设置搜索条件并通过逻辑关系组合复杂搜索条件进行对象搜索的搜索服务管理功能。

使用流程

图 7-1 定义搜索服务流程



使用方法

- 使用应用运行态
  您可以通过应用运行态使用搜索服务，具体操作请参见**创建搜索服务**。
- 使用全量数据服务API
  您可以通过**全量数据服务**提供的API使用搜索服务管理，具体操作请参见**7.2 搜索服务管理开发指导**。

7.2 搜索服务管理开发指导

涉及的接口

表 7-1 涉及的接口

接口名称	说明
https://**{Endpoint}**/rdm_**{appID}**_app/services/rdm/basic/api/searchservicedefine/create	创建搜索服务。
https://**{Endpoint}**/rdm_**{appID}**_app/services/rdm/basic/api/searchServiceIndexDefinition/saveList	添加索引。
https://**{Endpoint}**/rdm_**{appID}**_app/services/rdm/basic/api/searchServColumnController/saveDataServIndexEntityList	添加服务配置。
https://**{Endpoint}**/rdm_**{appID}**_app/services/rdm/basic/api/searchservicedefine/publishSingle	发布搜索服务。
https://**{Endpoint}**/rdm_**{appID}**_app/services/rdm/basic/api/searchServ/startEsSync/**{serNumber}**	同步搜索服务。

步骤 1：定义搜索服务

步骤1 准备工作。

在应用运行态中获取定义搜索服务的API信息，具体操作请参见**全量数据服务**。

步骤2 根据实际业务需求，调用API创建搜索服务。

- 请求示例

```
POST https://{Endpoint}/rdm_{appID}_app/services/rdm/basic/api/searchservicedefine/create

{
    "params": {
        "createDTO": {
            "name": "搜索服务示例",
            "description": "搜索服务示例",
            "descriptionEn": "Sample",
            "nameEn": "Sample",
            "owner": "",
            "tenantAliasName": "TestDME"
        },
        "tags": []
    }
}
```

其中，{Endpoint}表示数据建模引擎所在域名或IP地址，{appID}表示应用ID，“name”、“nameEn”、“description”、“descriptionEn”表示搜索服务的中英文名称和中英文描述。

- 响应示例


```
{
   "result": "SUCCESS",
   "data": [
      {
         "id": "505115507221864448",
         "className": "XDMSearchServDefine",
         "name": "搜索服务示例",
         "description": "搜索服务示例",
         "kiaguid": null,
         "securityLevel": "internal",
         "master": {
            "id": "505115507255418880",
            "creator": "TestAccount2 5547b9adee**********8aac057d2c",
            "modifier": "TestAccount2 5547b9adee**********8aac057d2c",
            "createTime": "2023-05-30T12:28:41.522+0000",
            "lastUpdateTime": "2023-05-30T12:28:41.522+0000",
            "rdmVersion": 1,
            "rdmDeleteFlag": 0,
            "rdmExtensionType": "XDMSearchServDefineMaster",
            "tenant": {
               "id": "-1",
               "creator": "xdmAdmin",
               "modifier": "xdmAdmin",
               "createTime": "2022-12-01T11:24:39.000+0000",
               "lastUpdateTime": "2022-12-01T11:24:39.000+0000",
               "rdmVersion": 1,
               "rdmDeleteFlag": 0,
               "rdmExtensionType": "Tenant",
               "tenant": null,
               "className": "Tenant",
               "name": "basicTenant",
               "description": "默认租户",
               "kiaguid": null,
               "securityLevel": "internal",
               "code": "basicTenant",
               "disableFlag": false,
               "dataSource": "DefaultDataSource"
            },
            "className": "XDMSearchServDefineMaster",
            "servNumber": "SS60466196"
         },
         "branch": {
            "id": "505115507255418881",
            "creator": "TestAccount2 5547b9adee**********8aac057d2c",
            "modifier": "TestAccount2 5547b9adee**********8aac057d2c",
            "createTime": "2023-05-30T12:28:41.569+0000",
            "lastUpdateTime": "2023-05-30T12:28:41.569+0000",
            "rdmVersion": 1,
            "rdmDeleteFlag": 0,
            "rdmExtensionType": "XDMSearchServDefineBranch",
            "tenant": {
               "id": "-1",
               "creator": "xdmAdmin",
               "modifier": "xdmAdmin",
               "createTime": "2022-12-01T11:24:39.000+0000",
               "lastUpdateTime": "2022-12-01T11:24:39.000+0000",
               "rdmVersion": 1,
               "rdmDeleteFlag": 0,
               "rdmExtensionType": "Tenant",
               "tenant": null,
               "className": "Tenant",
               "name": "basicTenant",
               "description": "默认租户",
               "kiaguid": null,
```



```
                "securityLevel": "internal",
                "code": "basicTenant",
                "disableFlag": false,
                "dataSource": "DefaultDataSource"
            },
            "className": "XDMSearchServDefineBranch",
            "version": "A"
        },
        "latest": true,
        "latestIteration": true,
        "versionCode": 1,
        "iteration": 1,
        "version": "A",
        "latestVersion": true,
        "workingCopy": true,
        "workingState": {
            "code": "INWORK",
            "cnName": "工作中",
            "enName": "inwork",
            "alias": "INWORK"
        },
        "checkOutUserName": "TestAccount2 5547b9adee54423cbc05978aac057d2c",
        "checkOutTime": "2023-05-30T12:28:41.514+0000",
        "preVersionId": null,
        "owner": "",
        "descriptionEn": "Sample",
        "esindexId": null,
        "dmentityVersion": null,
        "dmentityVersionName": null,
        "step": null,
        "nameEn": "Sample",
        "needRefresh": false,
        "graphId": null,
        "lifecycleTemplate": null,
        "lifecycleState": null,
        "disableFlag": false
    }
  ],
  "errors": []
}
```


----结束

步骤 2：添加索引

步骤1 准备工作。

在应用运行态中获取添加索引的API信息，具体操作请参见**全量数据服务**。

步骤2 根据实际业务需求，调用API添加索引。

- 请求示例

```
POST http://{Endpoint}/rdm_{appID}_app/services/rdm/basic/api/searchServiceIndexDefinition/saveList

{
  "params": {
    "searchServId": "505115507221864448",
    "searchServName": "搜索服务示例",
    "indexServColuVOList": [
      {
        "id": null,
        "indexName": "SampleIndex",
        "indexDesc": "",
        "indexType": "TEXT",
        "operator": null,
        "inputSeparator": null,
        "segMethod": "NOWORD",
```



```
            "segOption": "NOTINVOLVED",
            "searchUsage": true,
            "keywordUsage": true,
            "displayUsage": true,
            "matchType": "FUZZY",
            "key": "indexDefinition43"
        }
    ]
  }
}
```

其中，{Endpoint}表示数据建模引擎所在域名或IP地址，{appID}表示应用ID，“indexName”、“indexDesc”、“indexType”表示索引名称、描述和类型。

- 响应示例

```
{
  "result": "SUCCESS",
  "data": [
    {
      "id": "505115594543079424",
      "rdmExtensionType": "XDMSearchIndexEntity",
      "className": "XDMSearchIndexEntity",
      "searchServId": "505115507221864448",
      "indexName": "SampleIndex",
      "matchType": {
        "code": "fuzzy",
        "cnName": "模糊匹配",
        "enName": "fuzzy",
        "alias": "FUZZY"
      },
      "displayUsage": true,
      "keywordUsage": true,
      "operator": null,
      "segOption": {
        "code": "notInvolved",
        "cnName": "不涉及",
        "enName": "notInvolved",
        "alias": "NOTINVOLVED"
      },
      "indexMaxFieldSize": null,
      "indexType": {
        "code": "Text",
        "cnName": "文本",
        "enName": "Text",
        "alias": "TEXT"
      },
      "indexDesc": "",
      "searchUsage": true,
      "segMethod": {
        "code": "noWord",
        "cnName": "不分词",
        "enName": "noWord",
        "alias": "NOWORD"
      },
      "searchServName": "搜索服务示例",
      "inputSeparator": null,
      "relationIndexEntityList": null
    }
  ],
  "errors": []
}
```

----结束

步骤 3：添加服务配置

步骤1 准备工作。

在应用运行态中获取添加服务配置的API信息，具体操作请参见**全量数据服务**。

步骤2 根据实际业务需求，调用API添加服务配置。

- 请求示例

```
POST http://{Endpoint}/rdm_{appID}_app/services/rdm/basic/api/searchServColumnController/saveDataServIndexEntityList

{
    "params": {
        "searchServId": "505115507221864448",
        "searchServIndexEntityVOList": [
            {
                "entityName": "TestData",
                "entityNameEn": "TestData",
                "dataServRelationVOList": [
                    {
                        "attrName": "Code",
                        "attrDesc": null,
                        "searchIndexId": "505115594543079424",
                        "indexType": "TEXT",
                        "dataModelVersionNameEn": "Tenant",
                        "extendAttr": false,
                        "attrType": "STRING",
                        "sortNo": 1
                    }
                ]
            }
        ]
    }
}
```

其中，{Endpoint}表示数据建模引擎所在域名或IP地址，{appID}表示应用ID，“searchServIndexEntityVOList”表示服务配置中选中的实体和属性。

- 响应示例

```
{
    "result": "SUCCESS",
    "data": [],
    "errors": []
}
```

----结束

步骤 4：发布和验证搜索服务

步骤1 准备工作。

在应用运行态中获取发布搜索服务的API信息，具体操作请参见**全量数据服务**。

步骤2 完成搜索服务的配置后，调用API发布搜索服务。

- 请求示例

```
POST http://{Endpoint}/rdm_{appID}_app/services/rdm/basic/api/searchservicedefine/publishSingle

{
    "params": "505115507221864448"
}
```

其中，{Endpoint}表示数据建模引擎所在域名或IP地址，{appID}表示应用ID，“params”表示待发布的搜索服务ID。

- 响应示例

```
{
    "result": "SUCCESS",
    "data": [],
    "errors": []
}
```

----结束

步骤 5：同步搜索服务数据

步骤1 准备工作。

在应用运行态中获取同步搜索服务的API信息，具体操作请参见**全量数据服务**。

步骤2 根据实际业务需求，调用API同步搜索服务数据。

- 请求示例

  PUT http://**{Endpoint}**/rdm_**{appID}**_app/services/rdm/basic/api/searchServ/startEsSync/**{serNumber}**

  其中，{Endpoint}表示数据建模引擎所在域名或IP地址，{appID}表示应用ID，“serNumber”待同步的搜索服务ID。

- 响应示例

```
{
   "result": "SUCCESS",
   "data": [],
   "errors": []
}
```

----结束

8 在已有项目中部署数据管理 API 代理器

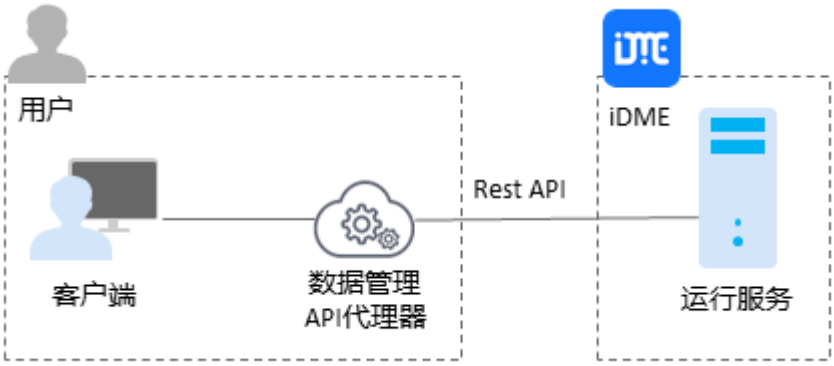
8.1 方案概述

应用场景

随着互联网化和数字化的深入，越来越多的业务被迁移到互联网上，从而产生了大量的业务需求和新技术的挑战。为了快速提升企业数据开放共享能力，工业数字模型驱动引擎-数据建模引擎（xDM Foundation，简称xDM-F）专门设计用于处理和转发API请求的数据管理API代理器，帮助企业和开发者简化API操作，提高系统的可扩展性和安全性，降低运维成本。

方案架构

图 8-1 方案架构图



该解决方案会部署如下资源：

- 开通一个iDME设计服务，用于在设计态完成数据模型的构建和发布，提供数据管理API代理器。
- 开通一个数据建模引擎，用于部署应用和生成运行态，对外提供应用的访问、数据模型动态扩展、数据管理功能配置相关能力。

8.2 资源和成本规划

该解决方案部署如下资源，以下费用仅供参考，具体请参考华为云官网**价格详情**，实际收费以账单为准。

表 8-1 成本预估（仅供参考）

华为云服务	计费说明	每月花费
工业数字模型驱动引擎 iDME	• 区域：华北-北京四 • 计费模式：按需计费 • 产品类型：iDME设计服务 • 价格：0.00元/小时	0.00元
工业数字模型驱动引擎 iDME	• 区域：华北-北京四 • 计费模式：包年包月 • 产品类型：数据建模引擎 • 部署位置：公有云 • 用户数量：100 • 价格：15000元 • 购买量：1月	15000.00元
工业数字模型驱动引擎 iDME	• 区域：华北-北京四 • 计费模式：包年包月 • 产品类型：数据建模引擎 • 部署位置：公有云 • 节点数量：2 • 价格：25200元 • 购买量：1月	25200.00元
工业数字模型驱动引擎 iDME	• 区域：华北-北京四 • 计费模式：包年包月 • 产品类型：数据建模引擎 • 增量包：结构化数据容量 • 价格：50元 • 购买量：1月	50.00元

华为云服务	计费说明	每月花费
工业数字模型驱动引擎 iDME	• 区域：华北-北京四 • 计费模式：包年包月 • 产品类型：数据建模引擎 • 增量包：文件数据容量 • 价格：68元 • 购买量：1月	68.00元
合计	-	40318.00元

8.3 实施步骤

8.3.1 获取与安装

下载数据管理 API 代理器

请参阅**应用发布**，下载对应版本的数据管理API代理器。

说明

为了避免不必要的繁琐代码排查与程序定位问题，如果您初次使用iDME或者数据量不大，推荐使用更加友好、图形化界面的**数据建模引擎**。

JAR 包结构

数据管理API代理器提供的JAR包目录结构如下：

```
|--- core-sdk-api-1.0.0-SNAPSHOT：包含接口的入参、出参、工具类等。
|--- rdm-common-1.0.0-SNAPSHOT：包含数据建模引擎（xDM Foundation，简称xDM-F）的公共类。
|--- rdm-delegate-1.0.0-SNAPSHOT：包含代理请求应用运行态的封装类。
|--- releaseNote.md：包含xDM-F的版本发布说明。
|--- 应用英文名称（小写）.api-1.0.0-optimization-SNAPSHOT：包含用户模型生成的代码（客户端），例如entity、DTO、service、delegator等相关代码。
    |--- com.huawei.innovation.rdm.应用英文名称（小写）：用户模型生成代码。
        |--- bean：生成的模型对象。
            |--- entity：数据实体。
            |--- enumerate：枚举。
            |--- interfaces：接口模型。
            |--- relation：关系实体。
        |--- delegator：模型的代理请求，使用调用代码函数的方式代替调API。
        |--- dto：数据传输实体。
            |--- entity：数据实体。
            |--- relation：关系实体。
        |--- service：模型服务层定义，仅做参考使用。
    |--- com.huawei.innovation.rdm.xdm：系统模型生成代码。
        |--- bean
        |--- delegator
        |--- dto
        |--- service
|--- 应用英文名称（小写）.impl-1.0.0-optimization-SNAPSHOT：包含用户模型实现的功能代码（服务端）。如果您已在华为云上部署应用，生成应用运行态，安装数据管理API代理器时需忽略此JAR包。
|--- 应用英文名称（小写）.controller.api-1.0.0-{应用环境标识}-SNAPSHOT：包含用户模型生成的controller API代码（客户端）。
```

```
    |--- com.huawei.innovation.rdm.应用名称：用户模型生成Controller。
    |--- com.huawei.innovation.rdm.xdm：系统模型生成Controller。
|--- 应用英文名称（小写）.controller.impl-1.0.0-{应用环境标识}-SNAPSHOT：包含用户模型生成的controller实现类代码（客户端）。如果您已在华为云上部署应用，生成应用运行态，安装数据管理API代理器时需忽略此JAR包。
    |--- com.huawei.innovation.rdm.应用名称：用户模型生成Controller。
    |--- com.huawei.innovation.rdm.xdm：系统模型生成Controller。
```

安装数据管理 API 代理器

须知

开发过程中，您有任何问题可以在**华为云工业数字模型驱动引擎论坛**中发帖求助。

步骤1 解压缩下载的JAR包。

步骤2 将解压缩后的JAR包引入到已有的Maven工程中。

以IntelliJ IDEA开发工具为例。

1. 在“resources”目录下创建一个“lib”目录，并将JAR包存放至该目录下。
2. 单击“File > Project Structure”。
3. 在弹出的窗口中，选择“Modules > Dependencies”，单击“+ > JARs or Directories…”。
4. 选中引入的JAR包，单击“OK”。
5. 单击“Apply”。

步骤3 在pom.xml文件的“<dependencies>”节点中添加如下依赖。

如下配置以“publishtest”应用为例，请将“publishtest”替换为您实际应用英文名称的小写。

```
<!--iDME core-sdk-api jar包依赖-->
<dependency>
    <groupId>com.huawei.isc.xdm-f</groupId>
    <artifactId>core-sdk-api</artifactId>
    <version>1.0.0-SNAPSHOT</version>
    <scope>system</scope>
    <systemPath>${project.basedir}/src/main/resources/lib/core-sdk-api-1.0.0-SNAPSHOT.jar</systemPath>
</dependency>
<!--iDME rdm-common jar包依赖-->
<dependency>
    <groupId>com.huawei.isc.xdm-f</groupId>
    <artifactId>rdm-common</artifactId>
    <version>1.0.0-SNAPSHOT</version>
    <scope>system</scope>
    <systemPath>${project.basedir}/src/main/resources/lib/rdm-common-1.0.0-SNAPSHOT.jar</systemPath>
</dependency>
<!--iDME rdm-delegate jar包依赖-->
<dependency>
    <groupId>com.huawei.isc.xdm-f</groupId>
    <artifactId>rdm-delegate</artifactId>
    <version>1.0.0-SNAPSHOT</version>
    <scope>system</scope>
    <systemPath>${project.basedir}/src/main/resources/lib/rdm-delegate-1.0.0-SNAPSHOT.jar</systemPath>
</dependency>
<!--iDME 应用英文名称的小写.api jar包依赖-->
<dependency>
    <groupId>com.huawei.isc.xdm-f</groupId>
    <artifactId>publishtest.api</artifactId>
    <version>1.0.0-optimization-SNAPSHOT</version>
```

```
    <scope>system</scope>
    <systemPath>${project.basedir}/src/main/resources/lib/publishtest.api-1.0.0-optimization-SNAPSHOT.jar</
systemPath>
</dependency>
<!--iDME 应用英文名称的小写.impl jar包依赖-->
<!--如果您已在本地部署iDME，可使用下方代码引用应用英文名称的小写.impl jar包-->
<!--<dependency>-->
<!--    <groupId>com.huawei.isc.xdm-f</groupId>-->
<!--    <artifactId>publishtest.impl</artifactId>-->
<!--    <version>1.0.0-optimization-SNAPSHOT</version>-->
<!--    <scope>system</scope>-->
<!--    <systemPath>${project.basedir}/src/main/resources/lib/publishtest.impl-1.0.0-optimization-
SNAPSHOT.jar</systemPath>-->
<!--</dependency>-->
<!--jsr validation-api包依赖-->
<dependency>
    <groupId>javax.validation</groupId>
    <artifactId>validation-api</artifactId>
    <version>2.0.1.Final</version>
</dependency>
<!--apache httpclient包依赖-->
<dependency>
    <groupId>org.apache.httpcomponents</groupId>
    <artifactId>httpclient</artifactId>
    <version>4.5.13</version>
</dependency>
<!--mikesamuel json-sanitizer包依赖-->
<dependency>
    <groupId>com.mikesamuel</groupId>
    <artifactId>json-sanitizer</artifactId>
    <version>1.2.3</version>
</dependency>
<!--hibernate hibernate-core包依赖-->
<dependency>
    <groupId>org.hibernate</groupId>
    <artifactId>hibernate-core</artifactId>
    <version>5.6.9.Final</version>
</dependency>
```

步骤4 在application.yml文件中添加delegate信息。

```
delegate:
  subAppId: rdm_e22c66fb1d05453fa********e3cc9c0_app
  domain: https://dme.cn-north-4.huaweicloud.com
  userName: DME_validation
  password: ********
  endpoint: https://iam.cn-north-4.myhuaweicloud.com
  domainName: hwstaff_IDT_SRE
  regionName: cn-north-4
  serviceType: services/dynamic
  ssl: true
  trustStore: classpath:uat.jks
  trustStorePassword: ********
```

- subAppId：rdm_应用ID_app。应用ID可前往当前应用设计态的“应用中心 > 应用发布”获取，详情请参见**应用发布**。
- domain：iDME服务所在的域名或IP。
- userName：IAM用户名。
- password：IAM用户密码。
- endpoint：IAM用户所在的域名或IP。
- domainName：租户名/原华为账号。
- regionName：当前iDME服务所在的地域。
- serviceType：URI格式的配置，默认为“publicservices”。如果您购买的数据建模引擎为基础版，则需要将“publicservices”修改为“services/dynamic”。

- ssl：是否开启SSL认证。默认为false，不开启SSL认证。如果开启SSL认证，则需要配置trustStore和trustStorePassword参数。
- trustStore：SSL证书的安装路径。
- trustStorePassword：SSL证书的私钥密码。

----结束

8.3.2 本地应用注入 iDME 注解示例

示例场景

对于拥有本地应用的用户，在本地安装数据管理API代理器后，只需要在启动类中注入**@ComponentScan(basePackages = {"com.huawei.innovation"})**注解，运行启动类即可。

操作步骤

在启动类上方添加**@ComponentScan(basePackages = {"com.huawei.innovation"})**注解，引入iDME的JAR包。

```
package com.singzai;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.annotation.CompinentScan;

@SpringBootApplication
@ComponentScan(basePackages = {"com.huawei.innovation"})
public class MainApplication{
    public static void main(String[] args) { SpringApplication.run(MainApplication.class, args); }
}
```

异常处理

在启动类注入iDME注解并运行启动类时，本地调试可能会提示如下错误：

```
java.security.cert.CertificateException: No subject alternative DNS name matching xxx found
```

您可以参考如下操作进行修复。

1. 在项目中创建一个类，并添加**RestTemplateBuilder**和**RestTemplate**方法。
2. 分别在**RestTemplateBuilder**和**RestTemplate**方法上注入**bean**。

   例如，您在“src/main/java/com/xdm/configuration/”路径下创建“RestTemplateConfiguration.java”，其示例代码如下：

   ```
   package com.xdm.configuration;

   import org.apache.http.conn.ssl.NoopHostnameVerifier;
   import org.apache.http.impl.client.CloseableHttpClient;
   import org.apache.http.impl.client.HttpClients;
   import org.springframework.boot.web.client.RestTemplateBuilder;
   import org.springframework.context.annotation.Bean;
   import org.springframework.context.annotation.Configuration;
   import org.springframework.http.client.HttpComponentsClientHttpRequestFactory;
   import org.springframework.web.client.RestTemplate;

   import javax.net.ssl.SSLContext;
   import javax.net.ssl.TrustManager;
   import javax.net.ssl.X509TrustManager;
   import java.security.KeyManagementException;
   import java.security.NoSuchAlgorithmException;
   ```

```
import java.security.cert.X509Certificate;

@Configuration
public class RestTemplateConfiguration {

    @Bean
    public RestTemplateBuilder restTemplateBuilder() {
        return new RestTemplateBuilder();
    }

    @Bean
    public RestTemplate restTemplate(RestTemplateBuilder builder) throws
NoSuchAlgorithmException, KeyManagementException {

        TrustManager[] trustAllCerts = new TrustManager[] {
                new X509TrustManager() {
                    public java.security.cert.X509Certificate[] getAcceptedIssuers() {
                        return new X509Certificate[0];
                    }
                    public void checkClientTrusted(
                            java.security.cert.X509Certificate[] certs, String authType) {
                    }
                    public void checkServerTrusted(
                            java.security.cert.X509Certificate[] certs, String authType) {
                    }
                }
        };
        SSLContext sslContext = SSLContext.getInstance("SSL");
        sslContext.init(null, trustAllCerts, new java.security.SecureRandom());
        CloseableHttpClient httpClient = HttpClients.custom()
                .setSSLContext(sslContext)
                .setSSLHostnameVerifier(NoopHostnameVerifier.INSTANCE)
                .build();
        HttpComponentsClientHttpRequestFactory customRequestFactory = new
HttpComponentsClientHttpRequestFactory();
        customRequestFactory.setHttpClient(httpClient);
        return builder.requestFactory(() -> customRequestFactory).build();
    }

}
```

3. 在**操作步骤**添加的**@ComponentScan**注解中，添加**1**创建的类的扫描路径。

```
package com.singzai;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.annotation.CompinentScan;

@SpringBootApplication
@ComponentScan(basePackages = {"com.huawei.innovation","com.xdm"})
public class MainApplication{
    public static void main(String[] args) { SpringApplication.run(MainApplication.class, args); }
}
```

8.3.3 创建数据实例示例

示例场景

本文对使用数据管理API代理器方式创建数据实例进行示例说明。

使用数据管理API代理器时，需要在代码中注入Delegator和RestTemplate。下面以创建XZTEST数据实体的数据实例为例。

操作步骤

在代码中注入Delegator和RestTemplate，并调用create方法创建数据实例。

```
import com.huawei.innovation.rdm.publishtest.delegator.XZTESTDelegator;
import org.junit.jupiter.api.Test;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.web.client.RestTemplate;

@SpringBootTest
public class DMETest {

    @Autowired
    XZTESTDelegator xztestDelegator;
    @Autowired
    RestTemplate restTemplate;

    @Test
    public void createTest() {
        TenantUtil.setTenantId(444082253783236608L);    // 设置当前操作的租户ID
        XZTESTCreateDTO createDTO = new XZTESTCreateDTO();  // 创建XZTEST数据实体的CreateDTO
        createDTO.setUid("123456");
        createDTO.setUvalue("abc");
        createDTO.setXzdate(new Timestamp(1672888888000L));
        System.out.println(createDTO);
        XZTESTViewDTO viewDTO = xztestDelegator.create(createDTO);  //采用实体代理类
（xztestDelegator），调用create方法创建实例
        TenantUtil.clearTenantId();  // 清除租户信息
        System.out.println(viewDTO);  //打印实例的所有信息
    }
}
```

8.3.4 查询某实体的实例数据示例

示例场景

本文对使用数据管理API代理器方式查询某实体所有实例数据进行示例说明。

使用数据管理API代理器时，需要在代码中注入Delegator和RestTemplate。下面以查询XZTEST数据实体的指定实例数据为例。

操作步骤

在代码中注入Delegator和RestTemplate，并调用find方法查询UID含有“123”的数据实例。

```
import com.huawei.innovation.rdm.publishtest.delegator.XZTESTDelegator;
import org.junit.jupiter.api.Test;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.web.client.RestTemplate;

import java.util.List;

@SpringBootTest
public class DMETest {

    @Autowired
    XZTESTDelegator xztestDelegator;
    @Autowired
    RestTemplate restTemplate;

    @Test
    public void findTest() {
        QueryRequestVo queryRequestVo = new QueryRequestVo();
        QueryCondition queryCondition = new QueryCondition();
        queryRequestVo.setFilter(queryCondition);
        List<String> values = new ArrayList<>();
        values.add("123");
        queryCondition.setConditionValues(values);    // 设置查询条件：where uid like 123
```

```
        queryCondition.setConditionName("uid");
        queryCondition.setOperator("like");
        RDMPageVO rdmPageVO = new RDMPageVO();  // 分页查询
        rdmPageVO.setCurPage(1);
        rdmPageVO.setPageSize(20);
        List<XZTESTViewDTO> xztestViewDTOS = xztestDelegator.find(queryRequestVo, rdmPageVO);
        System.out.println(xztestViewDTOS);  //打印查询结果
    }
}
```

8.4 关于数据管理 API 代理器的多维视图&多维分支相关接口说明

自2024年10月1.24.100版本开始，工业数字模型驱动引擎-数据建模引擎（xDM Foundation，简称xDM-F）数据管理API代理器提供的多维视图&多维分支相关接口使用对应数据模型特有的子类（例如CBMVFuntionTest01VersionViewCreateDTO），其子类继承了VersionViewCreateDTO（公共的父类），以便于在调用多维视图&多维分支相关接口时传入多视图属性。

1.24.100版本的多维视图&多维分支相关接口包括：getAllVersionsV2、logicalDeleteLatestVersionV2、deleteLatestVersionV2、getVersionByMasterV2、batchUpdateVersionV2、compareBusinessVersionV2、deleteBranchV2、logicalDeleteBranchV2、batchDeleteBranchV2、batchLogicalDeleteBranchV2、batchDeleteV2、batchLogicalDeleteV2、logicalDeleteV2、deleteV2和createViewV2。

接口说明

1.24.100以前版本的多维视图&多维分支相关接口将于2025年01月起逐步下线。如果您的系统正在调用1.24.100以前版本接口，请尽快更新至1.24.100版本接口，避免因相关接口下线而导致系统不可用。

1.24.100以前版本的多维视图&多维分支相关接口包括：getAllVersions、logicalDeleteLatestVersion、deleteLatestVersion、getVersionByMaster、batchUpdateVersion、compareBusinessVersion、deleteBranch、logicalDeleteBranch、batchDeleteBranch、batchLogicalDeleteBranch、batchDelete、batchLogicalDelete、logicalDelete、delete和createView。

示例代码对比

本章节以createView/createViewV2接口为例。

表 8-2 示例代码对比

1.24.100以前版本	1.24.100版本
public void test() { *// 创建VersionViewCreateDTO公共父类的dto对象* VersionViewCreateDTO dto = new VersionViewCreateDTO(); dto.setVersionId(1L); cbmvFuntionTest01Delegator.createView(dto); }	public void test() { *// 创建CBMVFuntionTest01VersionViewCreateDTO子类的dto对象* CBMVFuntionTest01VersionViewCreateDTO dto = new CBMVFuntionTest01VersionViewCreateDTO(); dto.setVersionId(1L); cbmvFuntionTest01Delegator.createViewV2(dto); }