|  |
| --- |
|  |
| DesignForFDAS |
|  |
|  |
| ©MetaShare Inc. |
| **5/7/2018** |

|  |
| --- |
| 本文包括第十期FDAS的设计过程。 |

目录

[1 引言 1](#_Toc399246668)

[1.1 目的 1](#_Toc399246669)

[1.2 背景 1](#_Toc399246670)

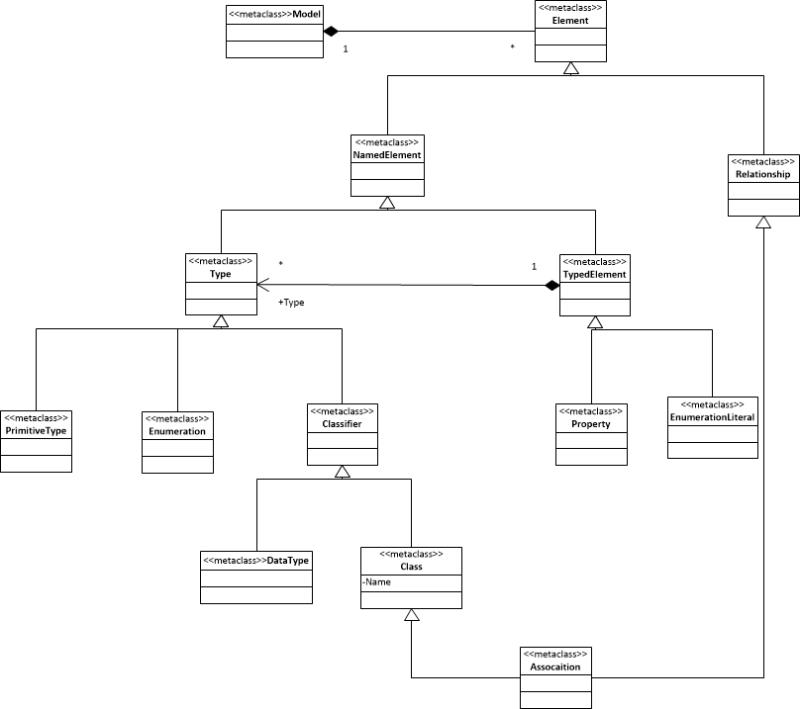
# 引言

## 目的

本文记录在FDAS开发过程中的设计过程。

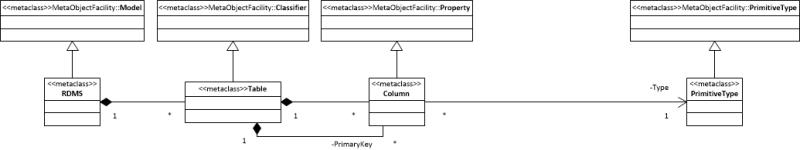
# 设计过程记录

## 20180504 初步设计

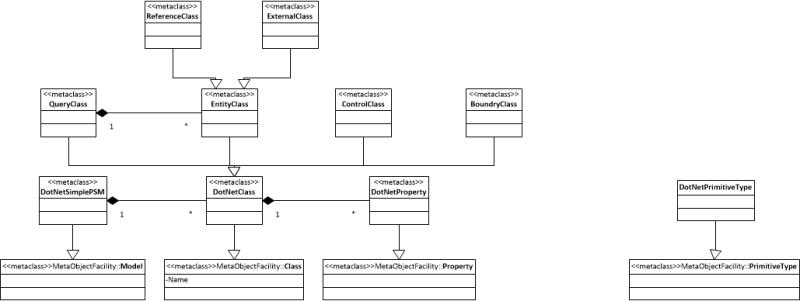


我们需要有一定的灵活性，所以我们将引入MOF和CommonCoreDynamic并做一些适应性修改，使得我们可以动态完成一些维护工作而不用发布

为存储操作数据，我们需要RDMS模型

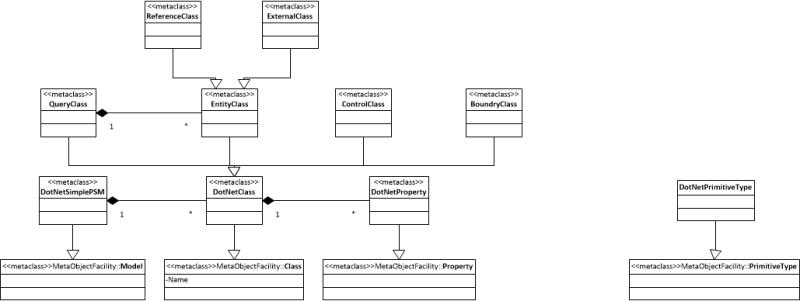


为了能反射必要的组建和方法等，我们需要一个DotNetSimplePSM

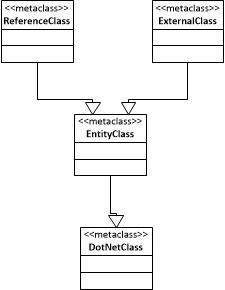


其中定义一系列的Class用于区分在应用系统中的层次

类似与分析模型



其中实体是存储的数据，为区分实体不同的操作定义了，以下结构

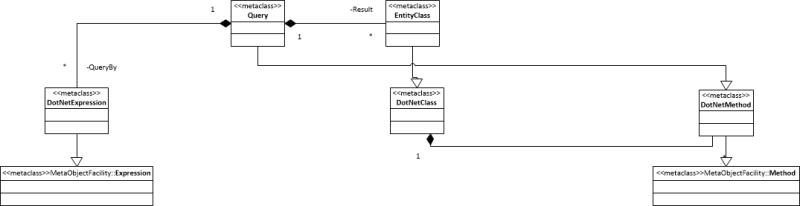
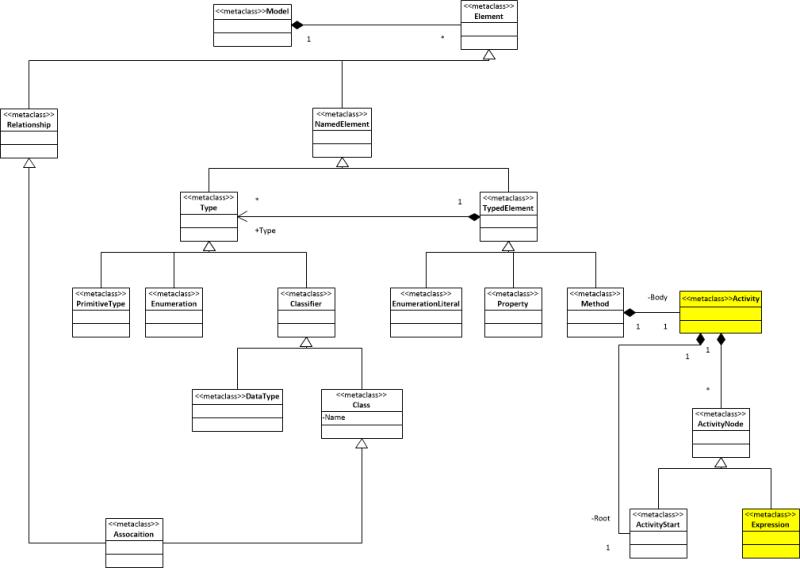


Reference是我（当前系统）经常要应用的数据，变化少，量小，ID不能变化，一般不允许删除，需要在系统启动时或隔一定时间同步

ExternalClass是我经常用到的数据，但不需要我存储

我们还可以根据使用场景再定义

为了获得进一步的灵活性，我们可以考虑把Activity加入MOF



我们不需要实现所有的Method，只需要定义比如Query，就可以足够灵活了

初步想法，有了这2个元模型，作为PSM的基础，再以EntityClass部分为基础定义一个CIM，再定义他们的变换

过程与代码生成相同，但本系统都是预先写好的模板，只由Reflector驱动变换产生接口，不需要代码生成器

这部分差不多了，我们有个大体的理解就可以

@zy @银幸元 @Bela @log 主要是你们几个需要理解，带领大家去实现。

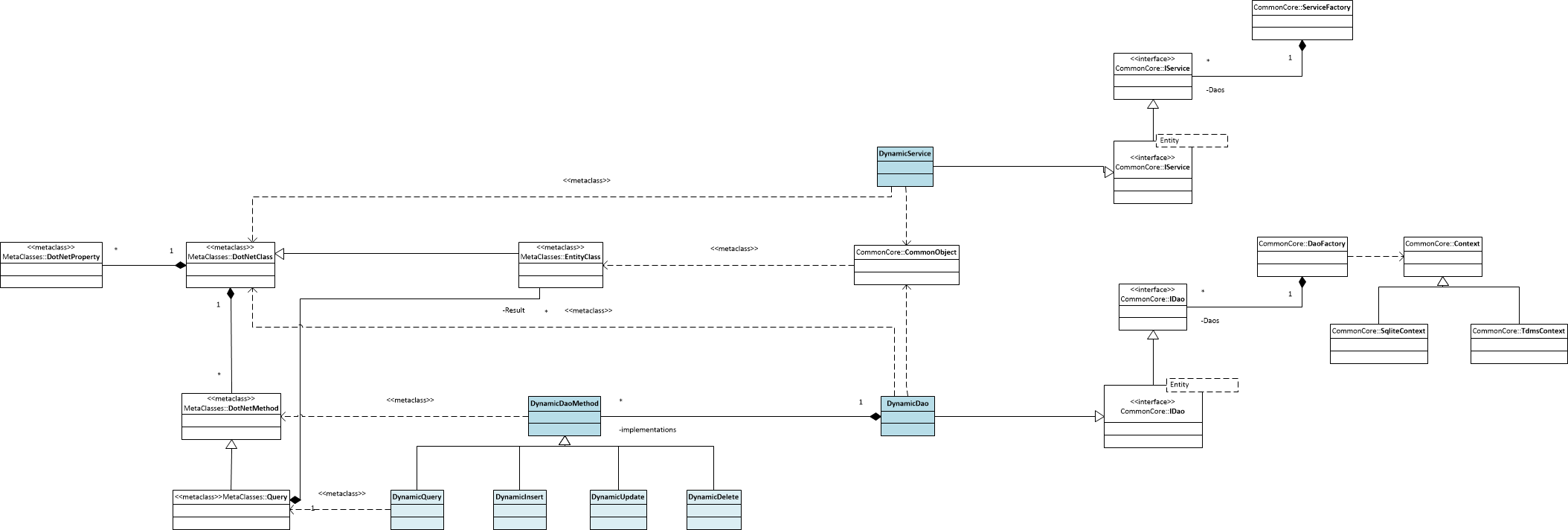
后面我在设计时会略过一些细节，不明白要及时提出了，我下周在路上也好解答

太细的设计不可能做出来。

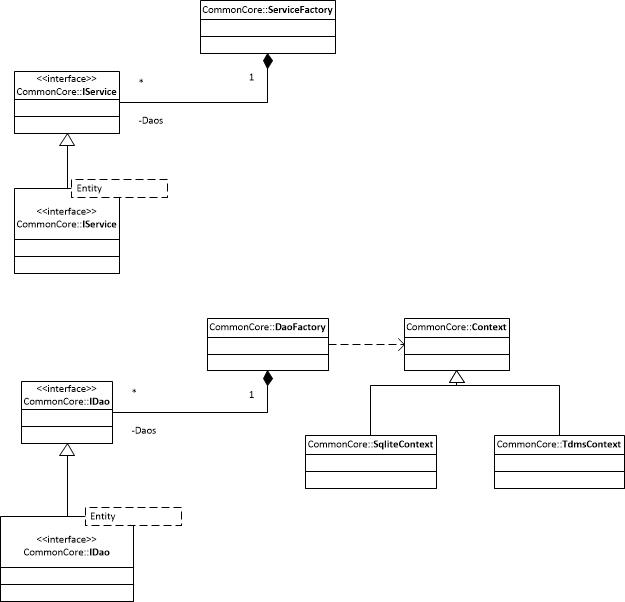
注意，这套机制也许要求你们在所有的Windows Form前台和Mobile APP中都要实现，使用统一的架构，做成产品。

做的时候，一定要使用我们已经存在的代码，不准另起炉灶

如果不知道怎么做，试着理解，不能理解的话不要动手，如果一直不能理解，就不要在这个团队里。

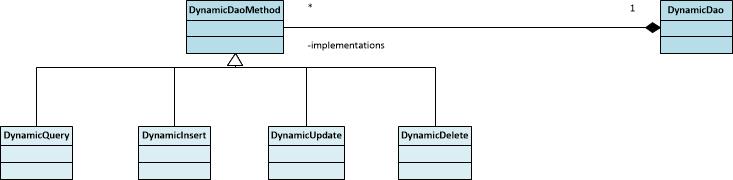


现在看看Reflector使用这些元数据元模型够不够灵活



这是我们在CommonCore定义的架构

我们在CommonCoreDynamic里面以前实现了一些，但后期方向错误，主要是凌飞当时在IService中加入过多的参数，试图通过参数让实现变得灵活，结果带来的是结构混乱，事实上他完全可是使用Builder/Prototype等模式使其变得灵活，而不是参数



这次我们并不去修改以前的代码，而是另外起一个分支，使用不同的实现模式。

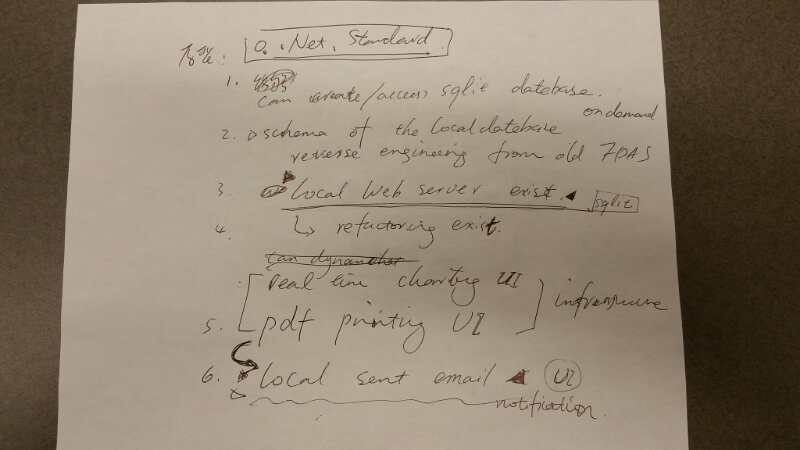
首先是规范和实现的分离

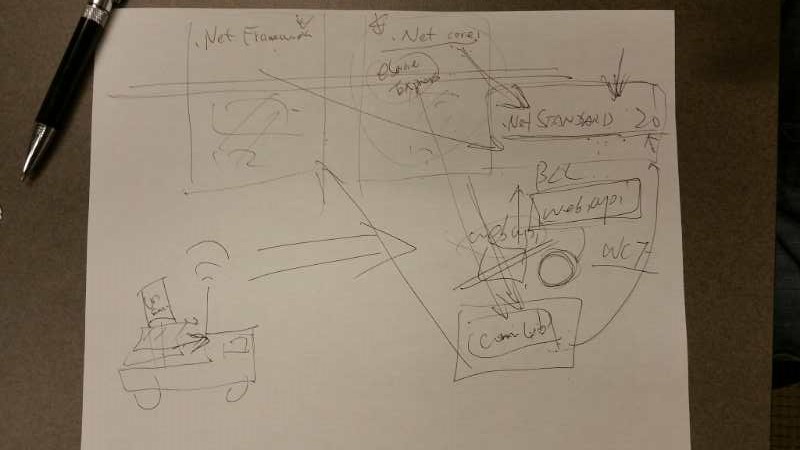
方法的实现为类，而这些方法对象可以使用Builder/Prototype等模式进行组合拼接

估计在加拿大没有时间设计其余部分了，应该还有：

1. DynamicService，可以调用DynamicDao和DynamicProxy，根据实体类型不同，有若干种组合模式
2. DynamicProxy，可以调用ExternalClass元类描述的外部数据源
3. 元数据获取及反射器，我们假定元数据的服务器组件已经用其他方法实现，
4. 服务器端与FDAS事实上是对等的，但考虑效率和其他因素，我们不认为以上组件一定是其实现，可以是手工书写，或者代码生成，而元数据可以是反向工程得到

## 20180505 确定的设计





山地时间5月4日上午11点，我和Adam在Sanjel开了一次短会，确定了以下内容：

1. 当前的设计文档已经基本足够详细，这个月的工作架构原型是比设计文档更重要的工作
2. 实现架构原型时本着这个优先级选库，即.NET Standard，.NET Core, .NET Framework。由于Toughbook/Windows成本过高，未来可能一个Linux盒子就可以完成，所以我们要提前这这方面的准备。
3. EService团队前期做了一个SQL Express的Local Web Server(@Bella 请确认是否存在？)，以此为基础进行重构。
4. 架构原型重点是3加上以下内容的框架

4.1、原型可以创建访问多个sqlite数据库，在其中创建或删除表，操作表数据。

4.2、使用数据生成图表的框架，使用Web-based的Syncfusion控件。

4.3、打印pdf格式图表的框架。

4.4、本地发送邮件进行通知的框架

1. 以上各项都是框架，要求复用能力很强，必须是架构师设计复审后再实现架构，架构实现复审后，才能进入构筑。
2. 本地数据库的Schema将由当前的FDAS反向工程得到，在Luke做完后Adam/Jason复审，没有问题交付开发团队在4完成确认后完全实现。在实现4.1时需要使用某些Schema，可以在当前设计中暂时选择一个。
3. 考虑到使用元模型进行设计实现的工程量太大难于短期内稳定，在近期仍使用静态的手写的组件。原设计中的元模型部分将作为另外一个项目处理，此部分设计将继续完成，并成为MDD在客户端的动态补充。

@Bella @佳佳 下周直接把EService Express和荧光项目中SQLITE相关部分合并，尽量保证他们使用同样组件。我们的目标是FDAS和荧光使用同一个地基