现在讨论这样一种情况：我们用一套Program做了一些实验，然后根据项目要求，需要修改Program的一些条件，然后进行另外一些实验。

我们希望

1. 对条件的修改，不会影响到原来实验数据的可追溯性。

2. 为了修改条件，不需要新增Program类，而只是新增Program实例。

原本的模型有4个层级：Program，Sub, Recipe, Tester Recipe。为了简化问题以便讨论，现在假设只有两个层级Program, Recipe。一个Program包含多个Recipe，现在假设客户的项目中一个Program包含3个Recipe. 假设客户一开始在Recipe3中设置的截止电压是V1，而后希望改为V2。

现在的操作流程是这个样子的：

1. 创建3个Recipe，名字叫Recipe1, Recipe2, Recipe3。Recipe3中的截止电压设置为V1。

2. 创建1个Program，包含上面3个Recipe，名字叫Program1

3. 执行Program1，提交实验数据Log1。

------------------------------

4. 创建新的Recipe4，设定其截止电压为V2；或Copy Recipe3，修改截止电压为V2，Save As Recipe4

5. 创建新的Program2，包含Recipe1, Recipe2，Recipe4；或Copy Program1，去掉Recipe3，添加Recipe4，Save As Program2.

6. 执行Program2，提交实验数据Log2。

需要说明的是，软件设计中，实验数据没有包含Program和Recipes的全部信息，而是使用RecipeID和ProgramID两个字段，来指明使用的是哪个Program的哪个Recipe。所以Log1里面的RecipeID是1，2，3；ProgramID是1。Log2里面的RecipeID是1，2，4；Program ID是2.

随着用户不停的提出新的需求，我们就需要不断创建新的Recipe和Program。Recipe Repository和Program Repository的Item会越变越多。

如果我们不想创建那么多新的item，想换成这样一个流程：

1. 同上

2. 同上

3. 同上

------------------------------

4. 修改Recipe3的截止电压为V2

5. 执行Program1，提交实验数据Log2

我们希望在原来的基础上修改就好了，而不是新增。这样Repository里的Item不会越变越多。

1. Need to CREATE new repository item to avoid information break.

在现有的设计上，这样做的问题是，如果我问“Log1的Recipe3的截止电压是多少？”，从数据上表现出来的答案会是V2（我们在Log1中只有ProgramID和RecipeID，根据ID找到的内容，是修改后的内容）。而我们知道，正确的答案应该是V1。

为了避免这样的问题，界面上没有提供Edit或者Update功能。必须要新增Program2和Recipe4才能保证可追溯性。也就是说还是得用第一个流程。

1. Need to CRUD design for program for sure.

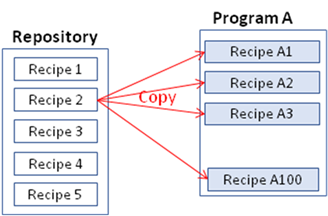
那么如何才能按流程2这样操作，同时还能保持可追溯性呢？那就是实验数据包含Program和Recipes的全部信息。

1. No, the test data (path) can be simply attach to the program recipes records, without adding any identification inside test data.

现在考虑这样一种情况：客户提了100次相同需求，没有任何需要修改的步骤。

In such case, for example, the request is to execute Recipe **R2** for 100 times in Program A:

1. Open program editor for Program A.
2. From repository, add recipe 2 into the Program A, for 100 times. Save the Program A.
3. So now Program A has 100 recipes attached to it, and those are all Recipe **R2**.
4. For the test data of each program recipe, attach to it respectively.



按照原有设计，我们创建Program1，然后执行100次。这一百个实验数据，都指向同一个Program1。

按照新设计，我们创建Program1，然后执行100次。这一百个实验数据，每个都包含一份Program1的副本。这样增加了存储开销。

所以，对于需要大量重复的Project，原设计有优势。对于一个Project执行一两次的状况，新设计会比较好。