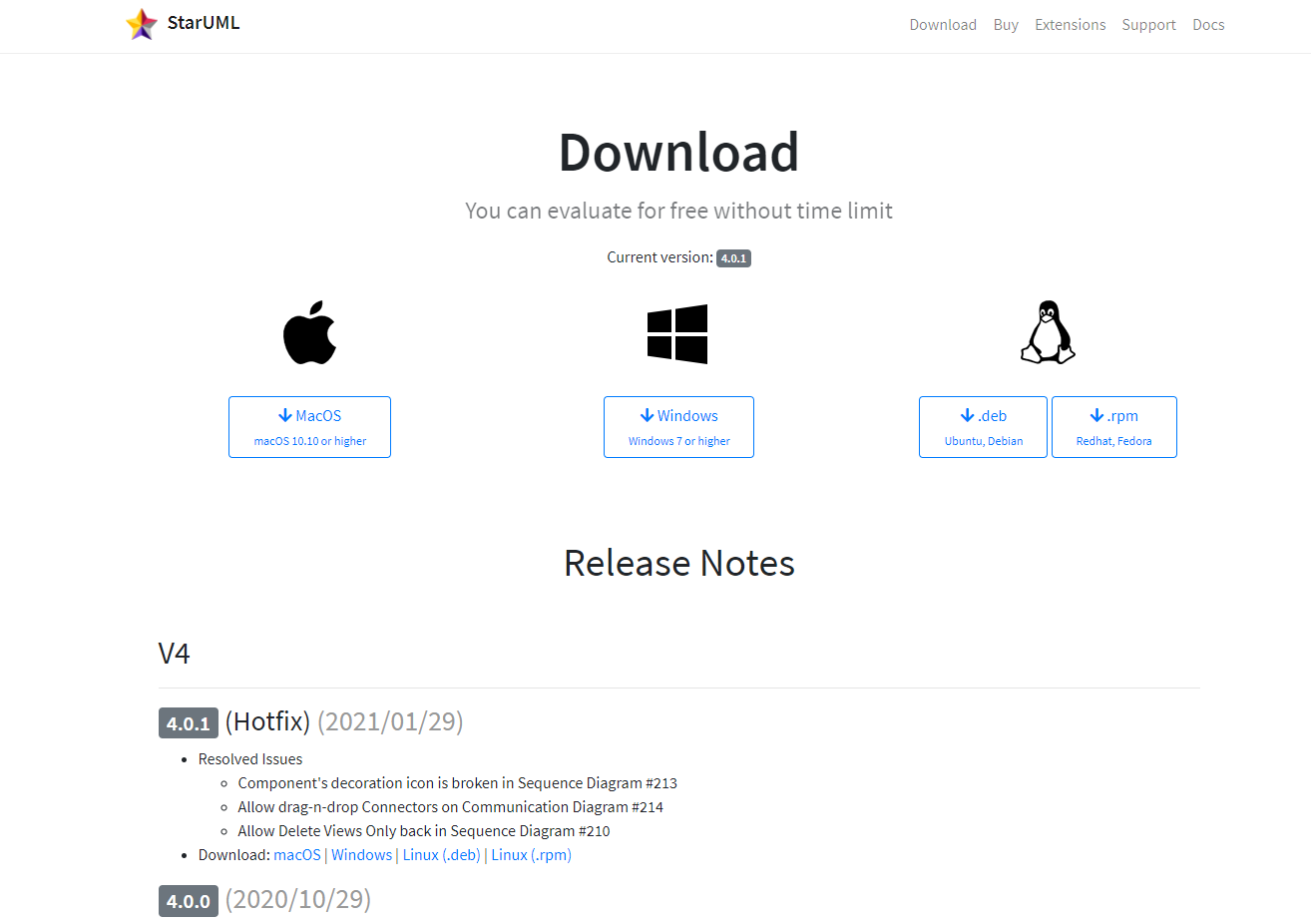
书上提到的StarUML5.0是较早的开源版本，由于版本过老，因此我组使用的工具为新版的StarUML4.0，软件需要付费，但可以无限期免费试用。

## 3.2 StarUML的安装与配置

#### 3.2.1 StarUML的安装

官网：<https://staruml.io/>

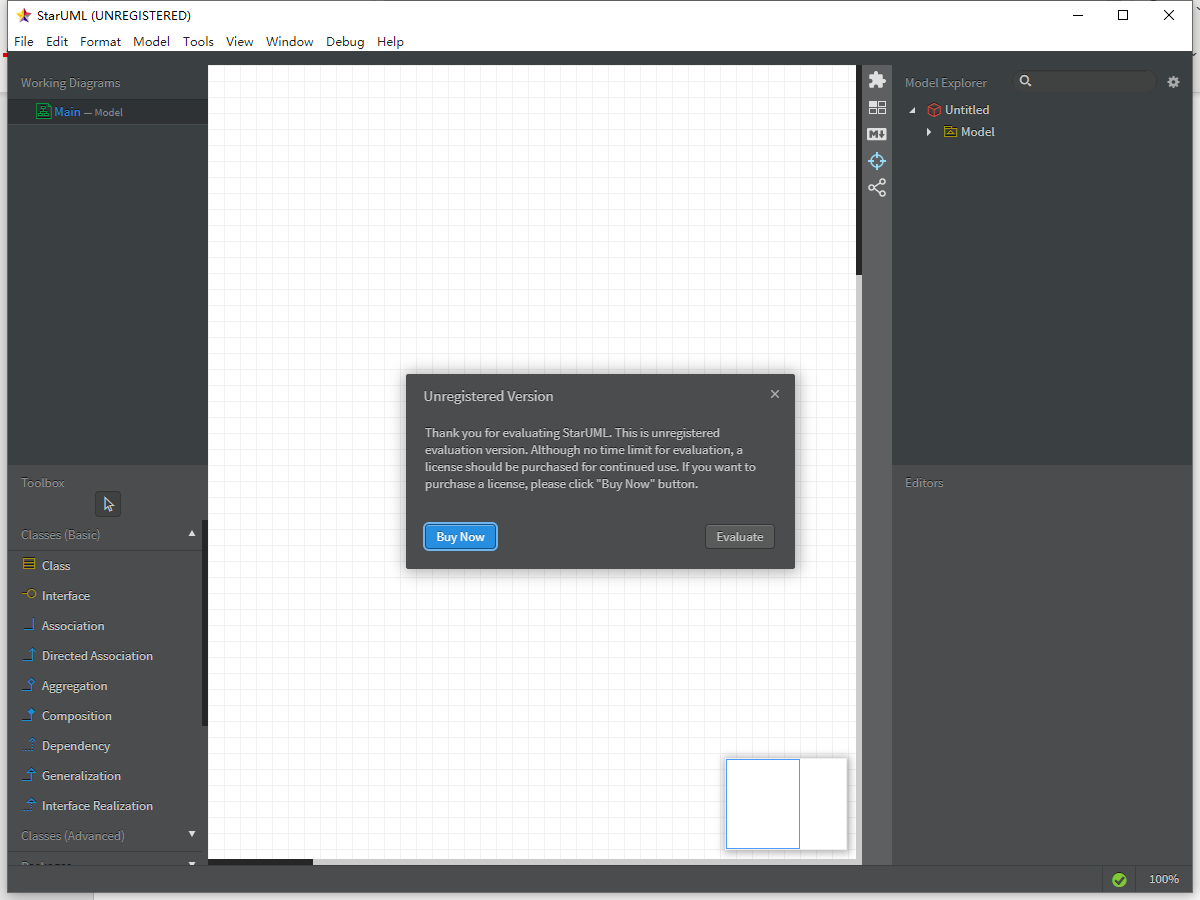
1. 访问官网，选择合适版本下载\



1. windows的下载链接得到的是安装包setup，点击运行安装



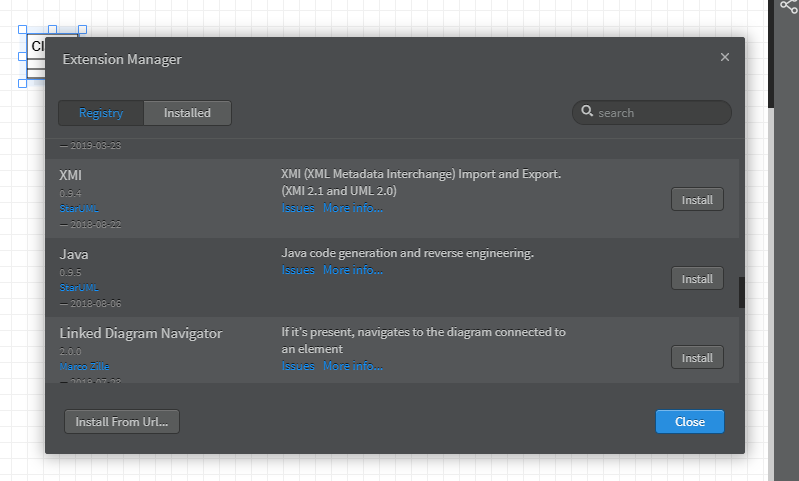
1. 安装完毕，打开软件，提示未购买证书，点Evaluate使用试用版



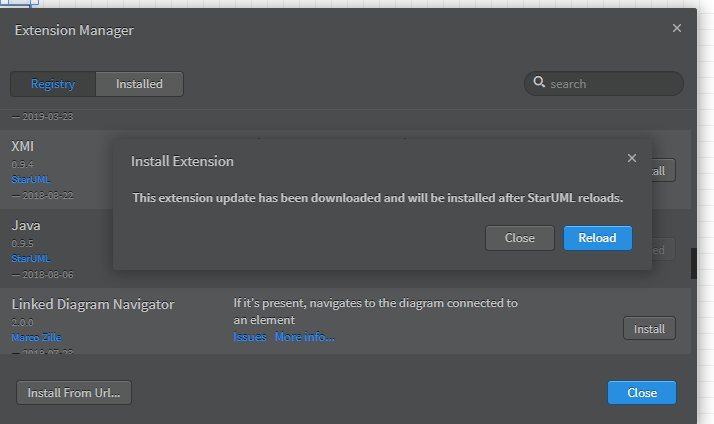
#### 3.2.2 StarUML的配置

以Java插件为例，安装Java插件，用于导入导出Java源代码

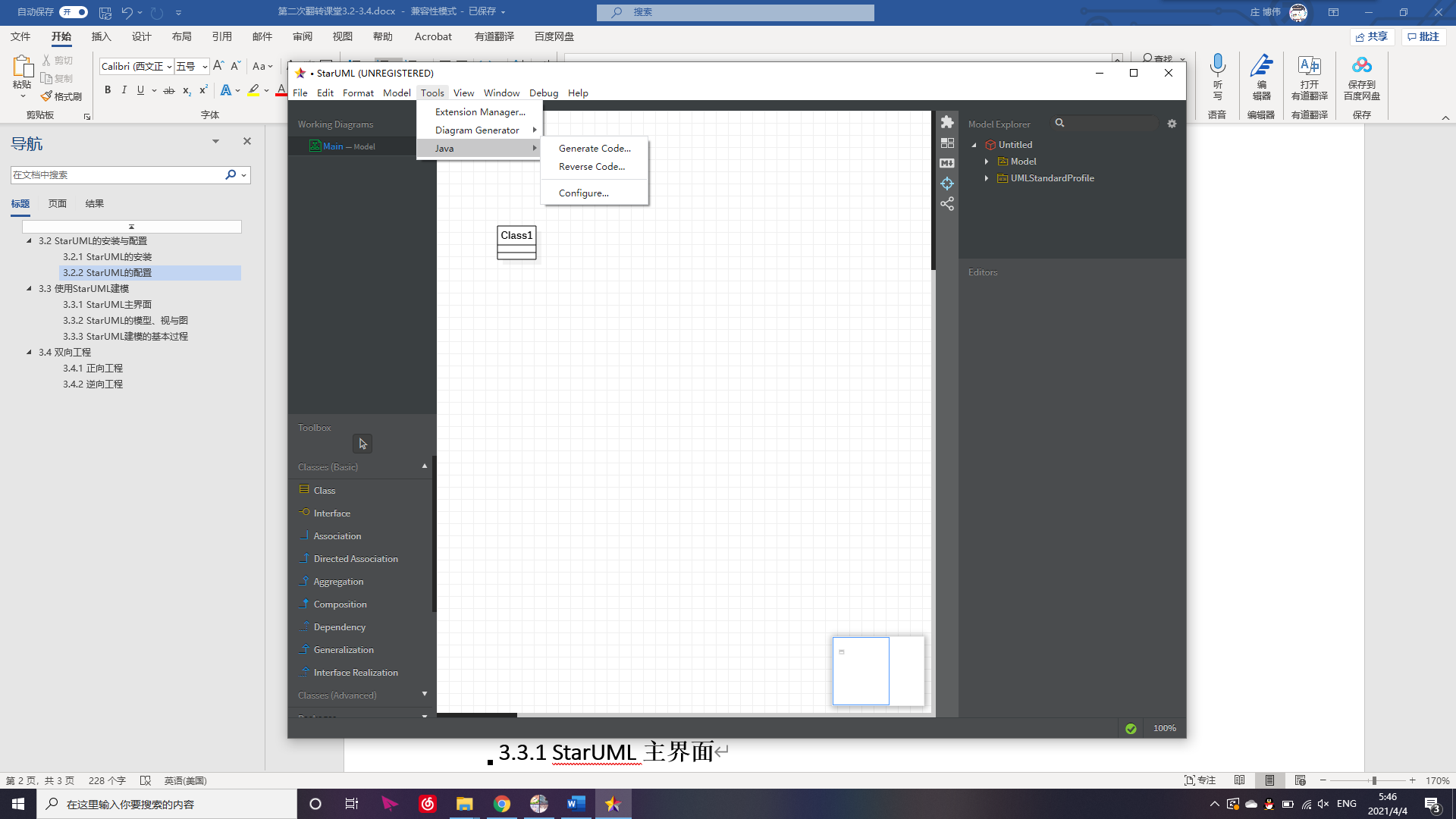
1. 点击菜单栏Tools-Extension Manager



1. 找到Java插件，点击Install安装，出现提示，重新加载StarUML

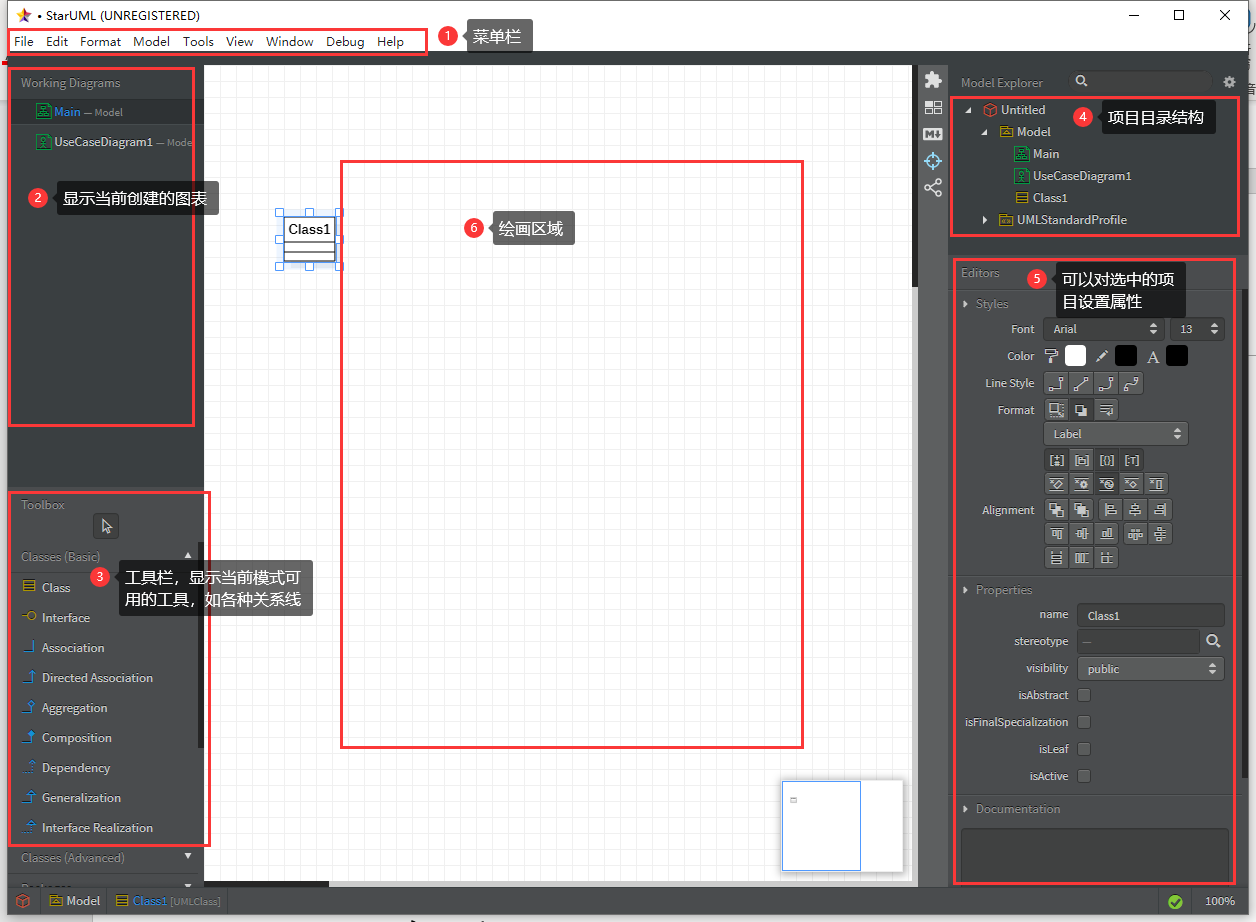


1. 新建一个StarUML项目，可以看到插件安装成功



## 3.3 使用StarUML建模

#### 3.3.1 StarUML主界面



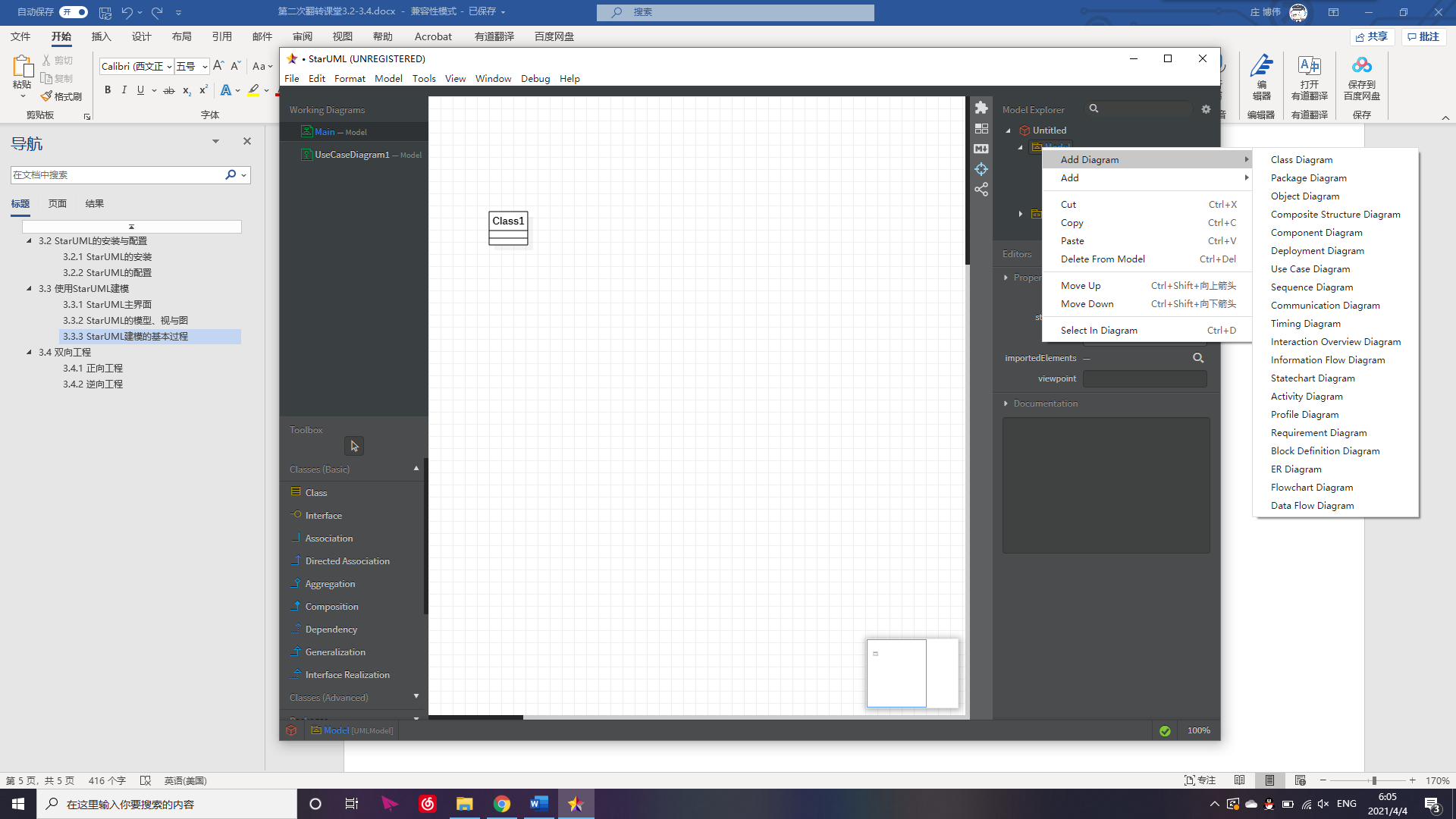
#### 3.3.2 StarUML的模型、视与图

StarUML中清晰地区分了模型（Model）、视（View）与图（Diagram）的概念。模型是包含软件模式信息的元素，视则是模型中信息的可视表达法，图则是表示用户特定设计思想的可视元素的集合。[1] (zzg注意这里ppt加脚注，我把引用放最后面)

其实意思就是StarUML对每种UML图都有对应的一套模式和绘画视图，即画不同种类的UML图时，左下角的工具栏内的内容不同。

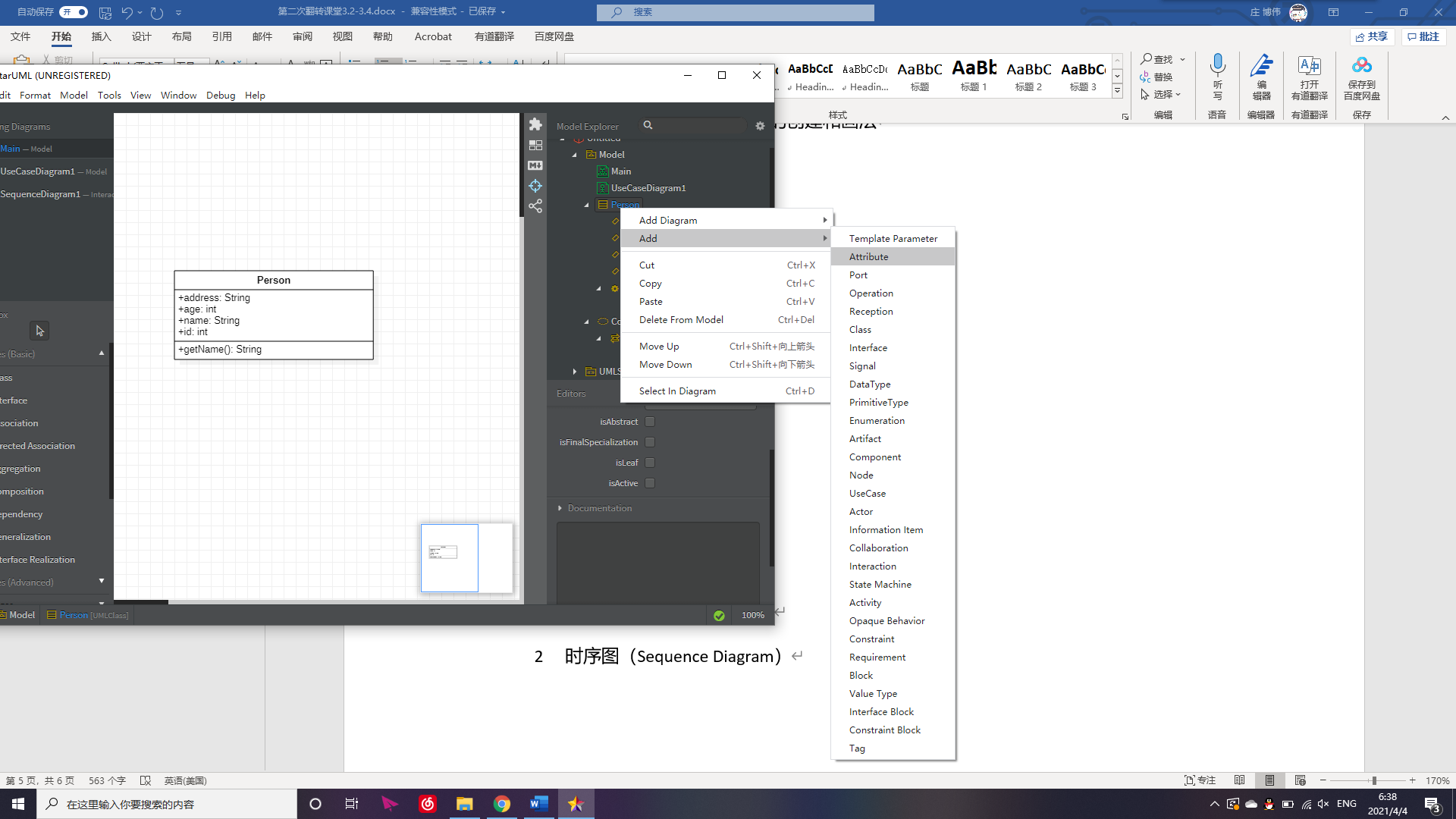
#### 3.3.3 StarUML建模的基本过程

在工作目录区右键-Add Diagram，即可创建不同类型的新图表

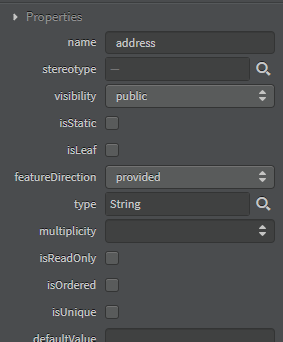


下面以几种常用图举例介绍各种图的创建和画法

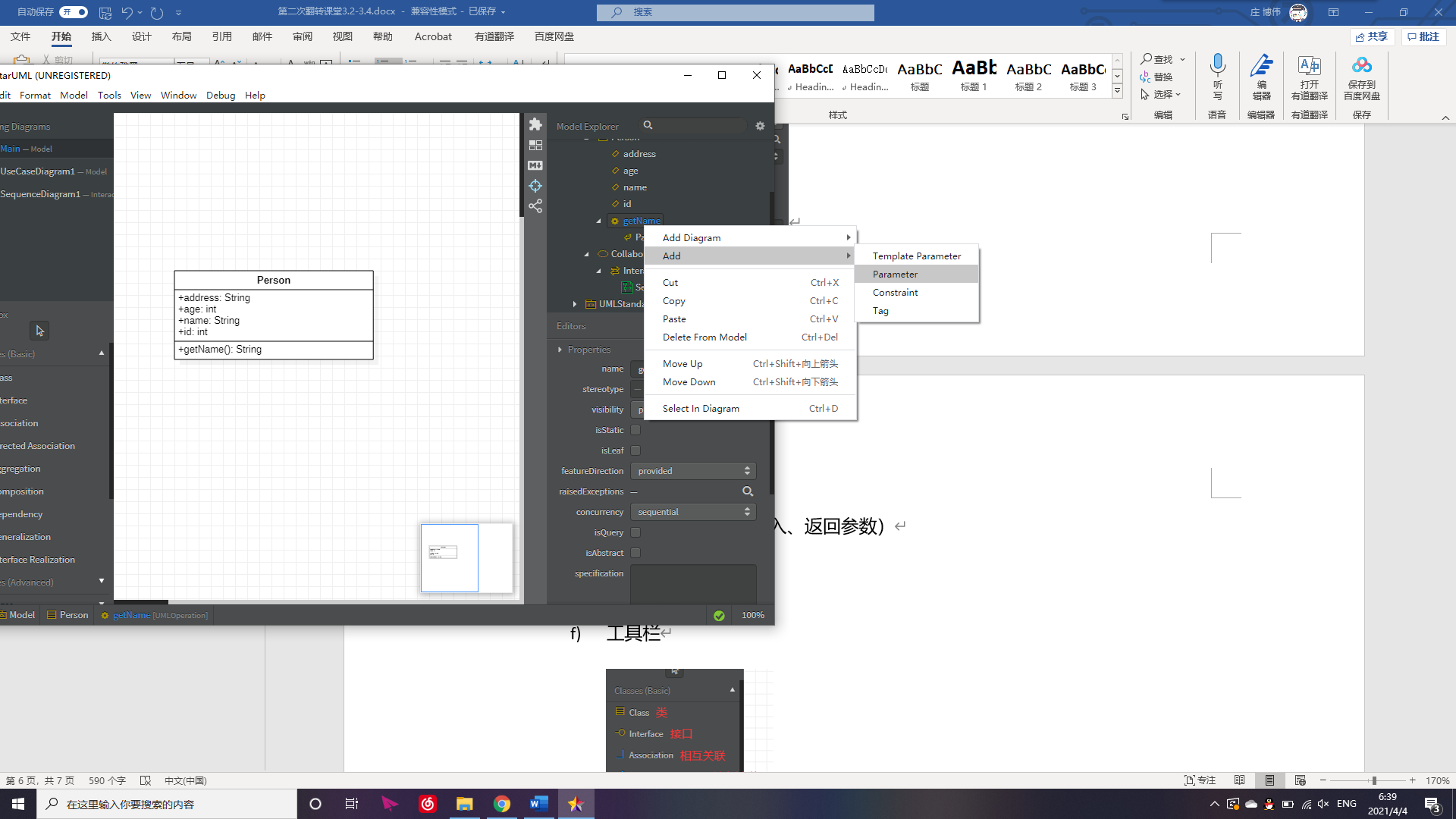
1. 类图（Class Diagram）
   1. 添加属性与方法

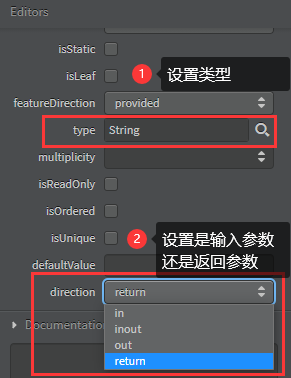


* 1. 设置属性类型

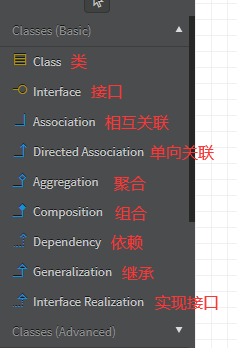


* 1. 为方法添加参数（输入、返回参数）

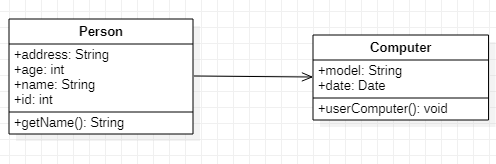




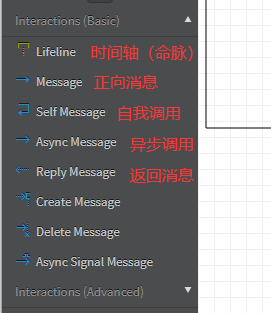
* 1. 工具栏



* 1. 随便画的类图



1. 时序图（Sequence Diagram）



## 3.4 双向工程

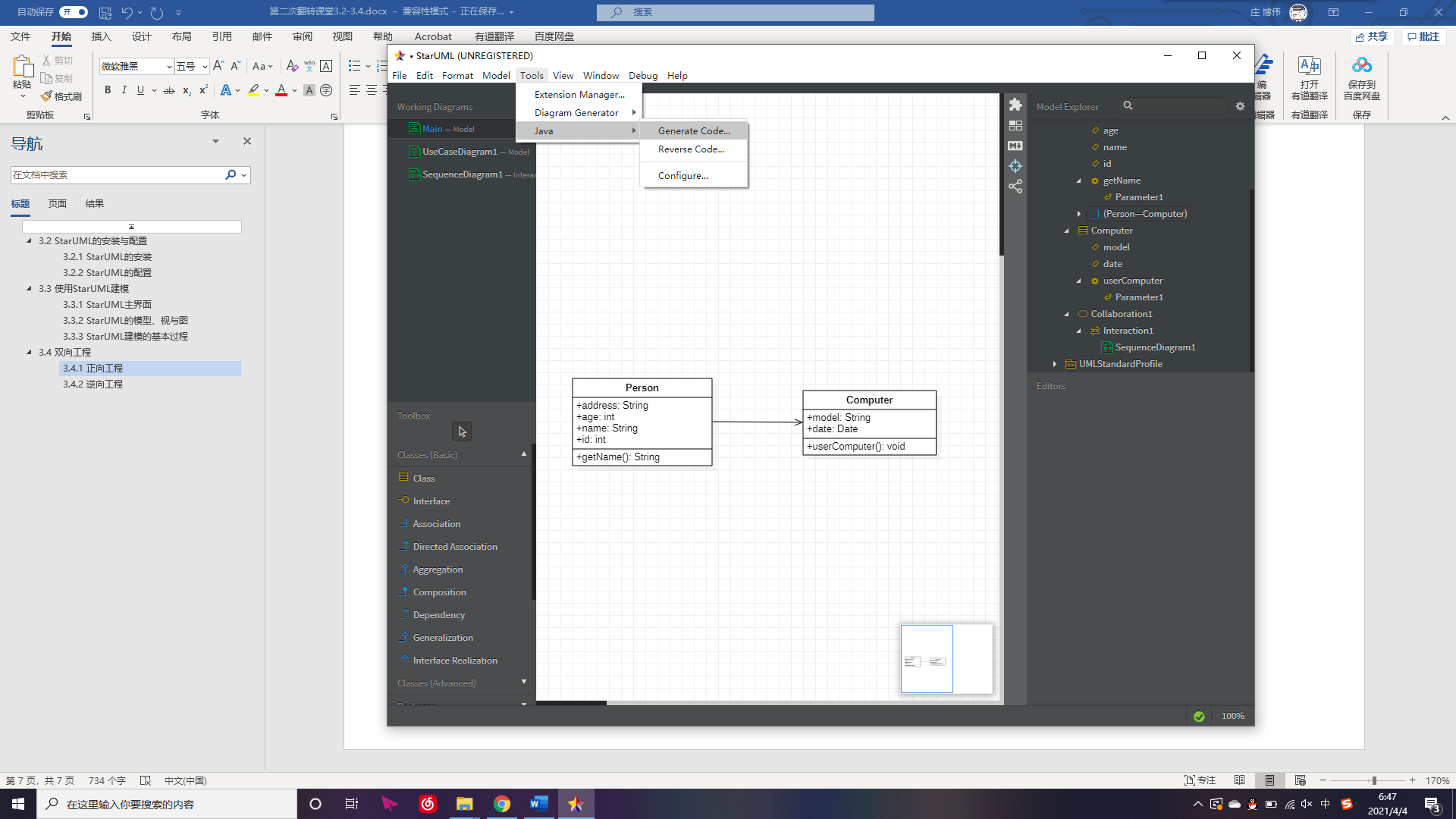
无论是从模型生成代码还是从代码生成模型，都是一项非常复杂的工作。StarUML将正向和逆向工程结合在了一起，并且提供了一种在描述系统的架构或设计和代码的模型之间进行双向交换的机制。[1]（脚注！！）

#### 3.4.1 正向工程

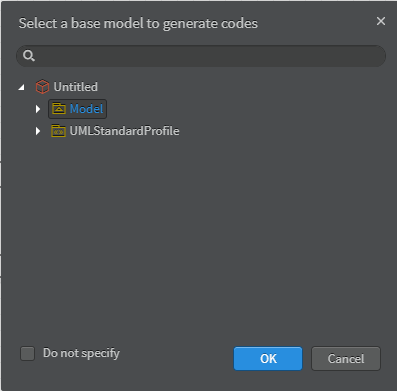
正向工程（代码生成）是指从模型直接产生一个代码框架，这将为程序员节约很多用于编写类、属性、方法代码的琐碎工作时间。但是这不等于不用编写代码了，而是存在一个框架，这个框架可以使开发人员思路更清晰。在StarUML中，可以将模型中的一个或多个类图转换为Java、C++、C#源代码。[1]

操作方法：

1. 点击Tools-Java-Generate Code



1. 选择要转换的源目录与生成的源代码存放目录



1. 转换成功

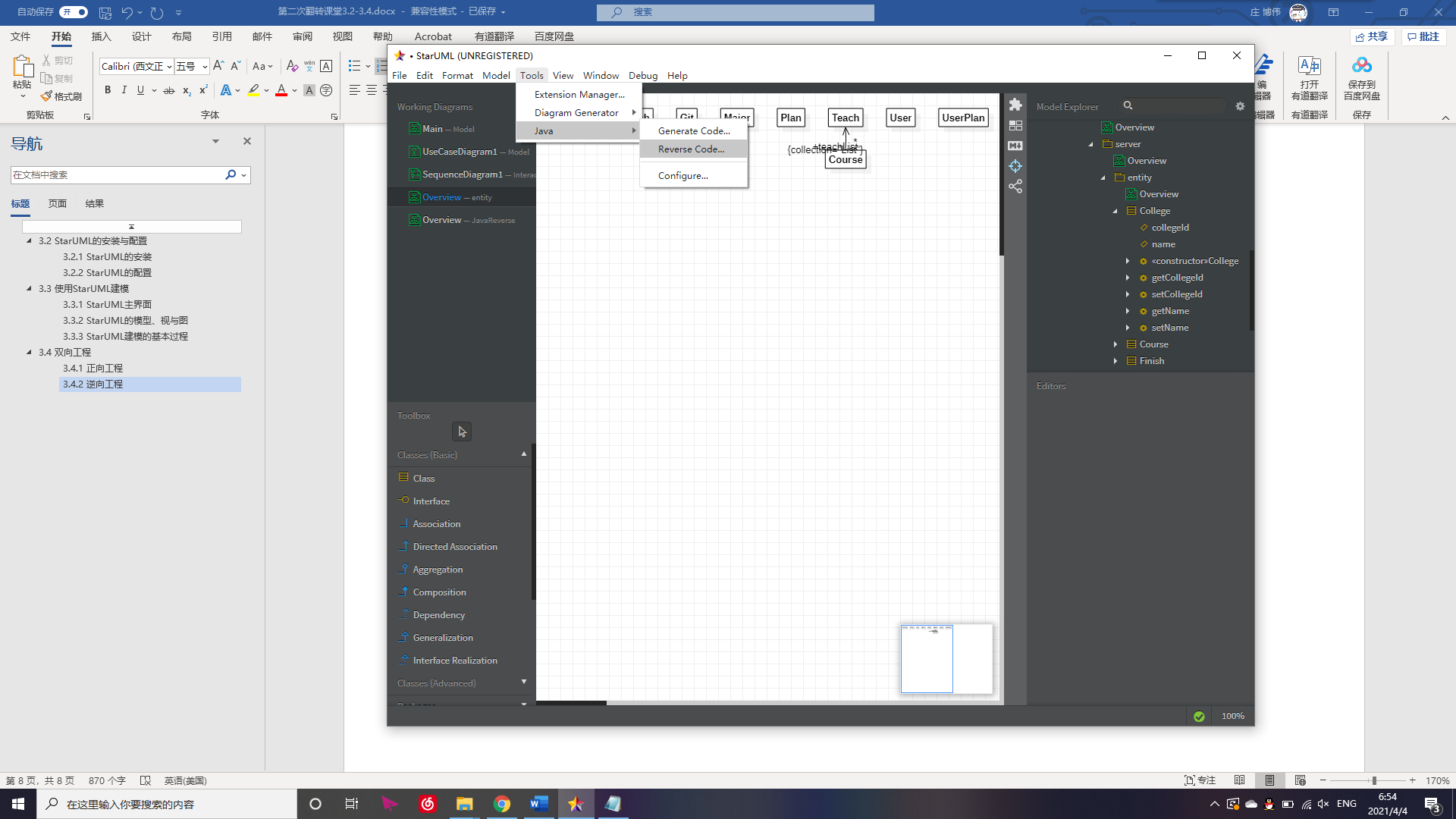


#### 3.4.2 逆向工程

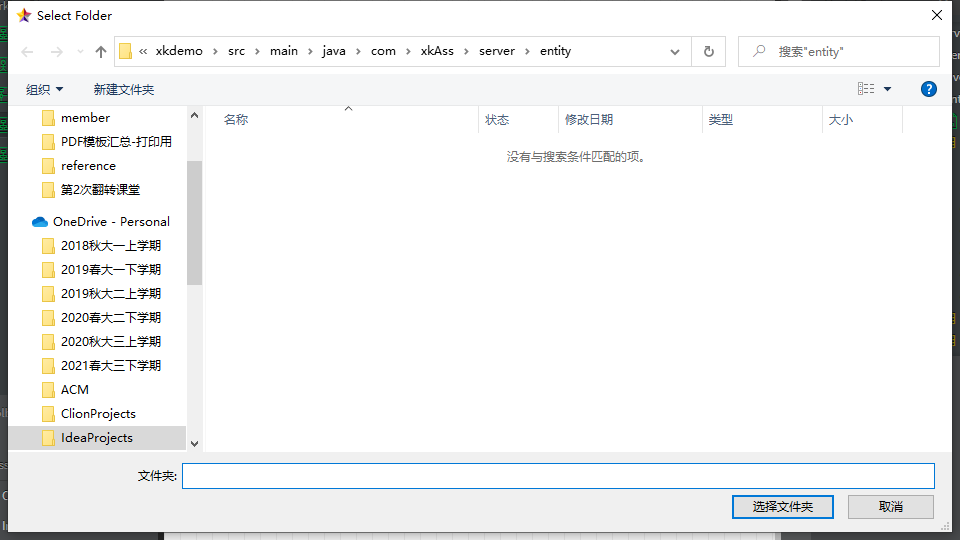
逆向工程是分析Java代码，然后将其转换到模型的类的过程。StarUML可以从现有的Java代码创建一个类图，这被称为Reverse Engineering，当从现有的代码生成图表，或者修改了生成的代码并且想在图表中反映出来时，就要启用逆向工程了。[1]

操作方法：

1. 点击Tools-Java-Reverse Code



1. 选择要分析的Java源代码目录



1. 转化成功，查看工作区

