实验一：软件需求分析与确认

1 实验目标

- 掌握需求分析与建模的基本方法，熟练掌握用户需求、系统需求描述的书写
- 基于UML建模方法（用例图、系统顺序图、概念类图），使用RM2PT原型自动生成工具进行需求建模，并且使用RM2PT生成原型进行需求确认

2 实验内容

- **实验准备：** 选定目标系统、工具下载与RM2PT项目创建
- **任务一：** 结构化自然语言建模
- **任务二：** UML需求建模
- **任务三：** UML需求原型化与需求确认

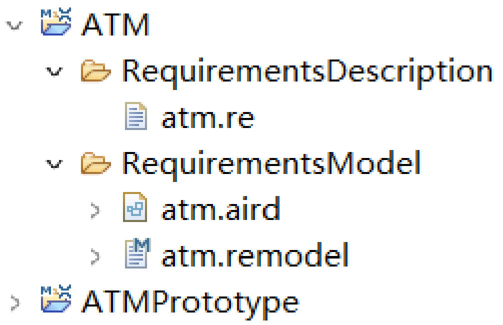
3 作业要求

3.1 模型要求

- 对于自选系统，需求模型要素数量需要满足以下要求（如果是对已有模型的扩展，则**新增**要素数量需要满足以下要求）。
- **自然语言需求部分：**
  - 用户需求数量不少于8条
  - 系统需求数量不少于15条（注：需要标注是功能需求还是非功能需求）
- **UML需求模型部分：**
  - 用例图中Actor不少于2个, 用例不少4个，用例间需要呈现`include`或者`exclude`关系
  - 系统顺序图不少于4个，系统操作总数不少12个，系统合约数量不少于12个
  - 概念类图中类不少于8个

3.2 格式要求

- 符合RM2PT项目结构（自然语言需求+UML需求模型）， 项目目录结构示例：



```
ATM
  RequirementsDescription
    atm.re
  RequirementsModel
    atm.aird
    atm.remodel
ATMPrototype
```

- 实验报告（以项目README文件形式给出），应包含建模结果截图及简要说明、模型规模说明（自然语言需求数量、Actor数量、用例数量、系统顺序图数量、系统操作总数、系统合约数量、类数量）、需求原型化结果截图
- 将整个项目上传到Github或Gitee中，仓库需要设置为Public可见，并将项目链接填写到问卷中。**注意：最晚提交时间为2025年4月8日0点！！**

4 实验准备

4.1 选定目标系统

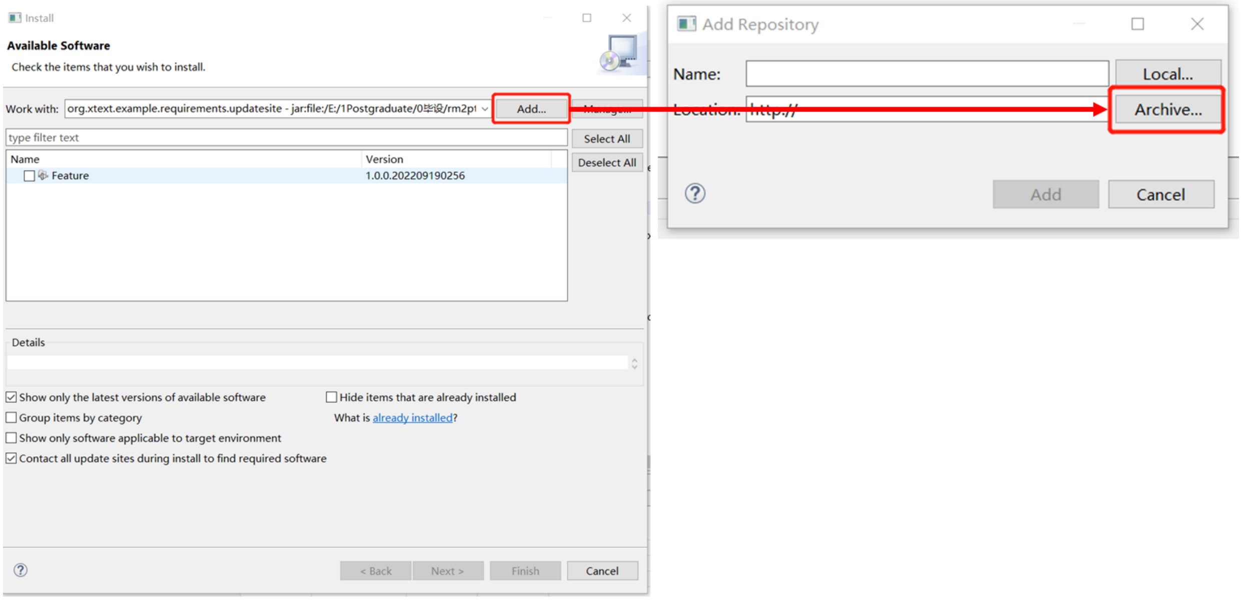
- 参考RM2PT Case Study（北航云盘），自选目标系统或者在已有案例基础上进行扩展。

4.2 安装需求原型化工具RM2PT及其插件

- 工具网站： https://ai4se.com
- 工具下载地址及安装方法：RM2PT官网下载 / 北航云盘下载
- 插件安装说明：RM2PT Developer Plugin Installing
- **自然语言插件（实验一使用）安装：**

  1. 从北航云盘下载org.xtext.example.requirements.updatesite-1.0.0-SNAPSHOT.zip
  2. 在RM2PT工具中，从主菜单栏Help → Install New Software 打开安装窗口，如下图.
  3. 点击Add，在弹出窗口中点击Archive，选择之前下载的.zip文件，点击Add确认添加
  4. 添加.zip(即updatesite文件)之后，在下方会出现Feature，勾选之后默认Next即可安装。
  5. 安装过程中，出现提示，选择Install anyway
  6. 安装完成后，软件提示重启以生效，重启即可。

7. 重启后在任意项目下新建后缀名为.re的文件，按后续介绍书写自然语言需求，关键词会出现高亮，说明插件安装成功。



Install

Available Software

Check the items that you wish to install.

Work with: org.xtext.example.requirements.updatesite - jar:file:/E:/1Postgraduate/0毕设/rm2pt

type filter text

Name	Version
□ Feature	1.0.0.202209190256

Details

☑ Show only the latest versions of available software
□ Group items by category
□ Show only software applicable to target environment
☑ Contact all update sites during install to find required software
□ Hide items that are already installed
What is already installed?

Add Repository

Name:

Location: http://

4.3 创建RM2PT项目

- 教程地址：RM2PT Tutorial

5 任务一：结构化自然语言建模

5.1 复习User Story和EARS模板

- 结合课程PPT进行复习。

5.2 撰写目标系统的结构化自然语言需求

- 在实验准备环节，已完成了RM2PT自然语言插件的安装，这里使用该插件辅助进行目标系统的结构化自然语言需求的撰写。
- 在项目中任意位置新建`.re`文件，即可进行结构化自然语言需求的书写。在书写结构化自然语言需求时，需要注意符合下面图中的格式。

结构化自然语言结构 （最外层用户故事）

ID：用例描述（用户故事形式）案例：

As a *cashier,* ***I want to*** *process sale,* ***so that*** *customers can buy goods.*

As a *cashier,* ***I want to*** *open cash desk,* ***so that*** *customers can pay.*

As a *cashier,* ***I want to*** *close cash desk,* ***so that*** *end of payment.*

该部分采用基础用户故事形式，*作为一个<用户角色>，我想要<完成活动>，以便于<实现价值>*。案例中**红色字段**为必填项，**紫色字段**为选填项，注意：案例中灰色空格不可被忽略。

主要结构 （用例描述部分）

NUM：用例内部表述（EARS形式）案例：

As a *cashier,* ***I want to*** *process sale,* ***so that*** *customers can buy goods*

{

Basic Flow {

(User）1. ***the*** *Customer* ***shall*** *arrives at a POS checkout with goods and services to purchase.*

(User）2. ***the*** *Cashier* ***shall*** *starts a new sale.*

(User）3. ***the*** *Cashier* ***shall*** *entars item identifier.*

(System）4. ***When*** *starts a new sale,* ***the*** *CoCoME* ***shall*** *Records each sale line item and presents item description and running total.*

}

}

该部分采用基础EARS形式，**本部分整体为选填项**，当需要对用例进行内部描述时进行填写，其中NUM.后面的需要表述当前需求是用户需求还是系统需求（*User/System*），另外案例中**红色字段**为必填项，**紫色字段**为选填项，注意：案例中灰色空格不可被忽略。

主要结构

补充信息案例：

As a *cashier,* ***I want to*** *process sale,* ***so that*** *customers can buy goods*

{

Alternative Flow {

A. ***At any time,*** *System fails* ***:***

To support recovery and correct accounting ensure all mansaction sensitive state and events can be recovered from any step of scenario.

1. *Cashier restarts System logs in and requests recovery of prior state.*

2. *System reconstructs prior state.*

a2. *System detects anomalies preventing recovery.*

1. *System signals error to Cashier records error and enters a clean state.*

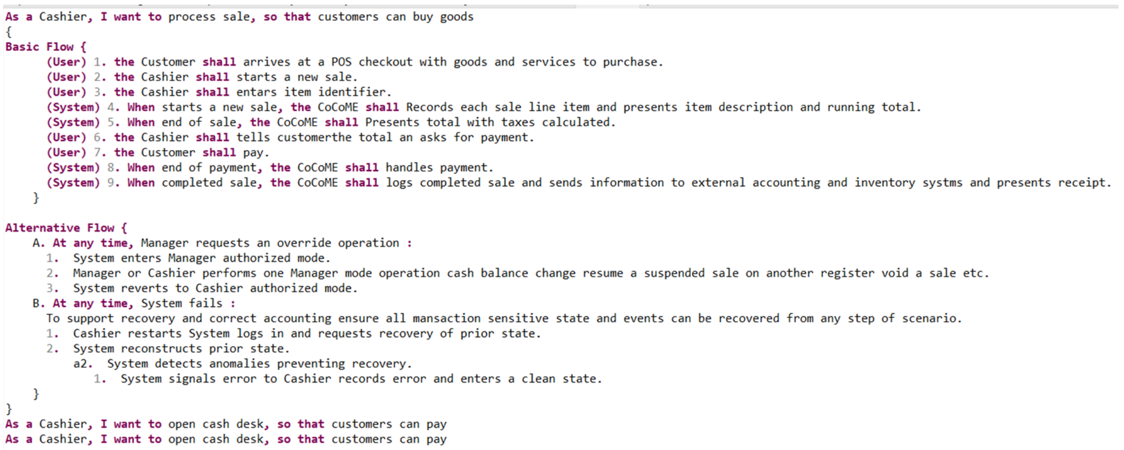
}

}

该部分采用基本的自然语言描述形式，对需求的细节进行补充，另外案例中**红色字段**为必填项，黄色背景为选填部分，对细节进行补充，**注意：**案例中灰色空格不可被忽略（由于技术原因该部分**大括号**内部的空格为两个）。

- 针对ATM系统撰写的结构化自然语言需求如下示例：

示例

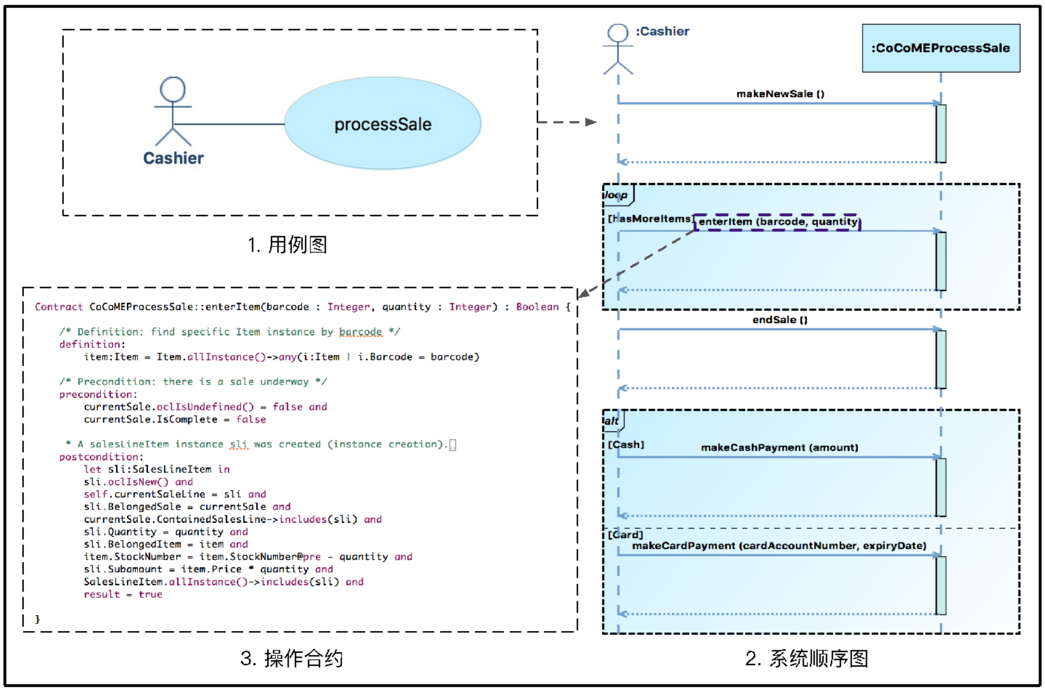


```
As a Cashier, I want to process sale, so that customers can buy goods
{
Basic Flow {
      (User) 1. the Customer shall arrives at a POS checkout with goods and services to purchase.
      (User) 2. the Cashier shall starts a new sale.
      (User) 3. the Cashier shall entars item identifier.
      (System) 4. When starts a new sale, the CoCoME shall Records each sale line item and presents item description and running total.
      (System) 5. When end of sale, the CoCoME shall Presents total with taxes calculated.
      (User) 6. the Cashier shall tells customerthe total an asks for payment.
      (User) 7. the Customer shall pay.
      (System) 8. When end of payment, the CoCoME shall handles payment.
      (System) 9. When completed sale, the CoCoME shall logs completed sale and sends information to external accounting and inventory systms and presents receipt.
    }

Alternative Flow {
    A. At any time, Manager requests an override operation :
      1.  System enters Manager authorized mode.
      2.  Manager or Cashier performs one Manager mode operation cash balance change resume a suspended sale on another register void a sale etc.
      3.  System reverts to Cashier authorized mode.
    B. At any time, System fails :
      To support recovery and correct accounting ensure all mansaction sensitive state and events can be recovered from any step of scenario.
      1.  Cashier restarts System logs in and requests recovery of prior state.
      2.  System reconstructs prior state.
          a2.  System detects anomalies preventing recovery.
             1.  System signals error to Cashier records error and enters a clean state.
    }
}
As a Cashier, I want to open cash desk, so that customers can pay
As a Cashier, I want to open cash desk, so that customers can pay
```

6 任务二：UML需求建模

6.1 复习UML需求模型与建模方法

- **UML需求模型-用例模型**



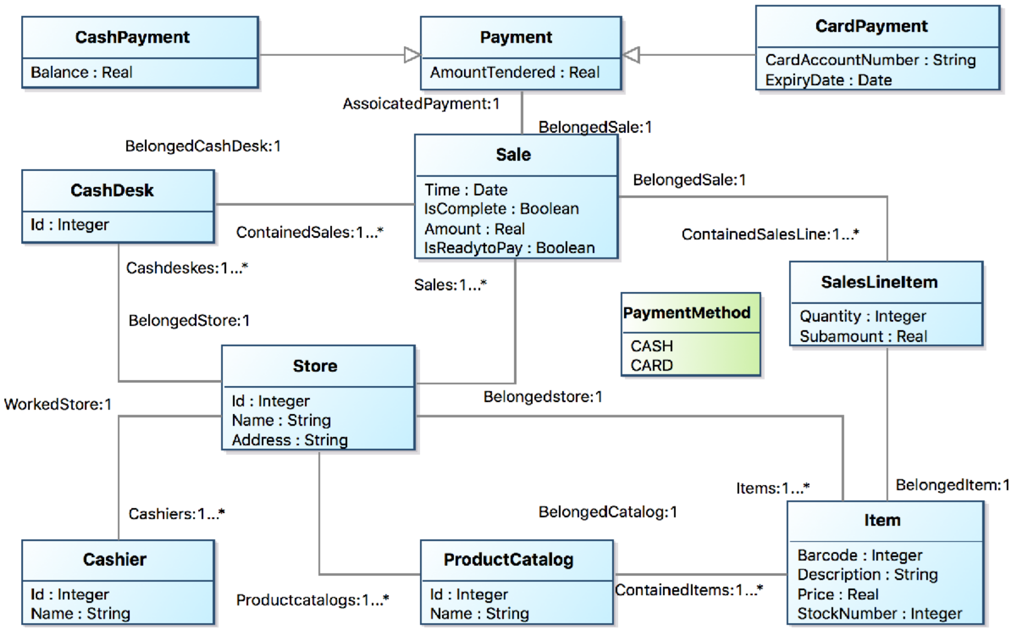
1. 用例图

```
Contract CoCoMEProcessSale::enterItem(barcode : Integer, quantity : Integer) : Boolean {

    /* Definition: find specific Item instance by barcode */
    definition:
        item:Item = Item.allInstance()->any(i:Item | i.Barcode = barcode)

    /* Precondition: there is a sale underway */
    precondition:
        currentSale.oclIsUndefined() = false and
        currentSale.IsComplete = false

     * A salesLineItem instance sli was created (instance creation).
    postcondition:
        let sli:SalesLineItem in
        sli.oclIsNew() and
        self.currentSaleLine = sli and
        sli.BelongedSale = currentSale and
        currentSale.ContainedSalesLine->includes(sli) and
        sli.Quantity = quantity and
        sli.BelongedItem = item and
        item.StockNumber = item.StockNumber@pre - quantity and
        sli.Subamount = item.Price * quantity and
        SalesLineItem.allInstance()->includes(sli) and
        result = true

}
```

3. 操作合约

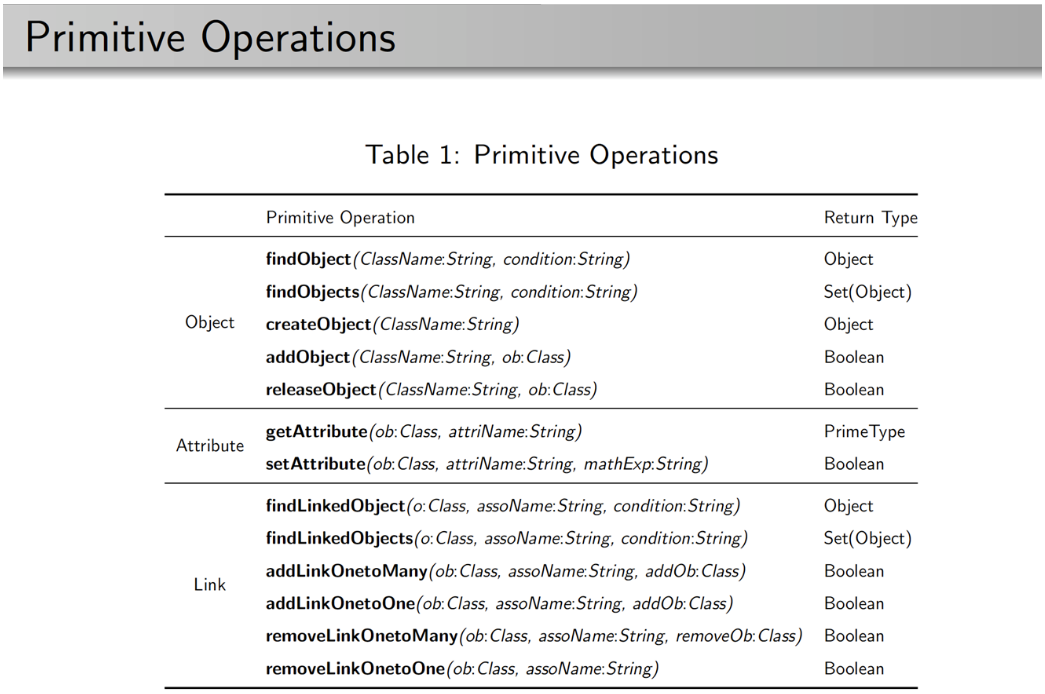
2. 系统顺序图

- **UML需求模型-领域模型**



4. 概念类图

- **面向对象基本操作**



Primitive Operations

Table 1: Primitive Operations

	Primitive Operation	Return Type
Object	**findObject**(*ClassName:String, condition:String*)	Object
	findObjects(*ClassName:String, condition:String*)	Set(Object)
	createObject(*ClassName:String*)	Object
	addObject(*ClassName:String, ob:Class*)	Boolean
	releaseObject(*ClassName:String, ob:Class*)	Boolean
Attribute	**getAttribute**(*ob:Class, attriName:String*)	PrimeType
	setAttribute(*ob:Class, attriName:String, mathExp:String*)	Boolean
Link	**findLinkedObject**(*o:Class, assoName:String, condition:String*)	Object
	findLinkedObjects(*o:Class, assoName:String, condition:String*)	Set(Object)
	addLinkOnetoMany(*ob:Class, assoName:String, addOb:Class*)	Boolean
	addLinkOnetoOne(*ob:Class, assoName:String, addOb:Class*)	Boolean
	removeLinkOnetoMany(*ob:Class, assoName:String, removeOb:Class*)	Boolean
	removeLinkOnetoOne(*ob:Class, assoName:String*)	Boolean

- **OCL合约**

```
//Signature
Contract CoCoMEProcessSale::enterItem
    (barcode : String, quantity : Real) : Boolean {

//Definition Section
definition:
    //Find Object
    item:Item = Item.allInstance()->any(i:Item | i.Barcode = barcode)

//Pre-condition Section
precondition:
    currentSale.oclIsUndefined() = false and
    currentSale.IsComplete = false and
    item.oclIsUndefined() = false and
    item.StockNumber > 0

//Post-condition Section
```

```
postcondition:
    //Create an Object
    let sli:SalesLineItem in
    sli.oclIsNew() and
    //Add Links
    self.currentSaleLine = sli and
    sli.BelongedSale = currentSale and
    currentSale.ContainedSalesLine->includes(sli) and
    sli.BelongedItem = item and
    //Modify Attributes
    sli.Quantity = quantity and
    sli.Subamount = item.Price * quantity and
    item.StockNumber = item.StockNumber@pre - quantity and
    //Add an Object
    SalesLineItem.allInstance()->includes(sli) and
    result = true
}
```

6.2 使用RM2PT工具进行目标系统UML需求模型构建

- 教程地址：Requirements Modeling in 4 steps

7 任务三：UML需求原型化与需求确认

7.1 使用RM2PT工具进行需求原型化及需求确认

- 教程地址：Requirements Validation through Auto-Prototyping