

COMP23420 Lecture 3

Realising Use Cases

Kung-Kiu Lau

kung-kiu@cs.man.ac.uk

Office: Kilburn 2.68

Overview

Active learning sheets (during lectures)

Generic software development process (recap)

ATM case study

Realising use cases

Workshop 2 (Functional Modelling: Use Cases)

Software Development Process



**User's
requirements**



**Software
System**

Software Development Process



www.bigstock.com · 29436170

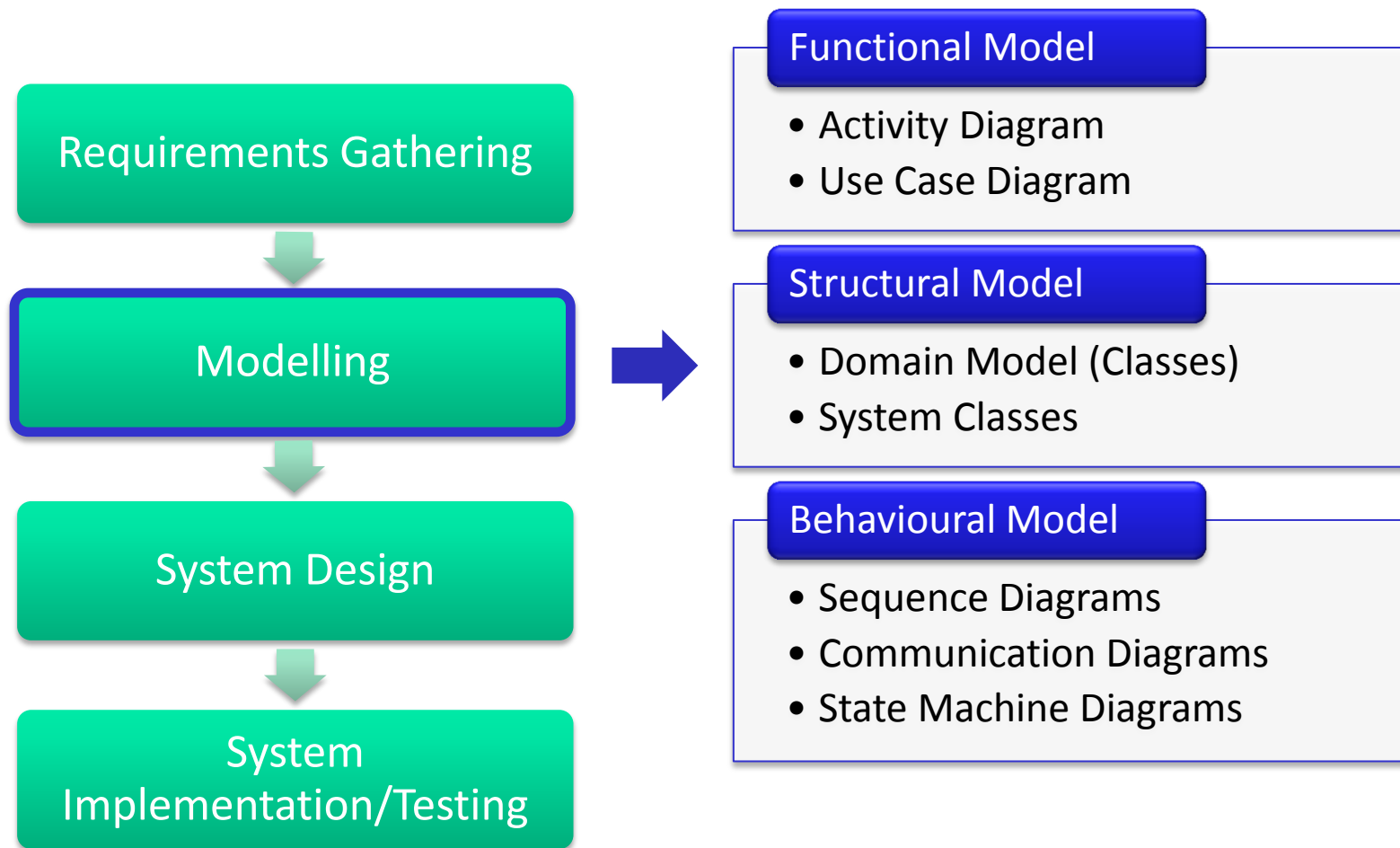


www.shutterstock.com · 57909892



gg58120616 www.gograph.com

A Generic Software Development Process



The ATM Example

Kung-Kiu Lau and Robert Stevens

kung-kiu@cs.man.ac.uk

robert.stevens@manchester.ac.uk

ATM Case Study

Contents

1	Requirements	1
2	Activity Diagram	1
3	Use Cases	3
3.1	Use Case Descriptions	4
3.1.1	Request Balance	4
3.1.2	Withdraw Money	5
3.1.3	Transfer Money	6
3.1.4	Deposit Money	7
4	Structural Model	8
4.1	Domain Classes	8
4.2	Domain Model	11
4.3	System Classes	12
4.3.1	The Complete System Class Diagram	18
5	Behavioural Model	19
5.1	Sequence Diagrams	20
5.1.1	The Withdraw Money Use Case	20
5.1.2	The Deposit Money Use Case	22
5.1.3	The Transfer Money Use Case	22
5.1.4	The Check Balance Use Case	24
5.2	State Diagrams	25
5.2.1	CardReader	25
5.2.2	KeyPad	26
5.2.3	Display	26
5.2.4	ClientManager	27
5.2.5	TransManager	28
5.2.6	AccManager	29
A	ATM glossary	29

Complete example with UML modelling

- From requirements to behavioural model

Available on Moodle

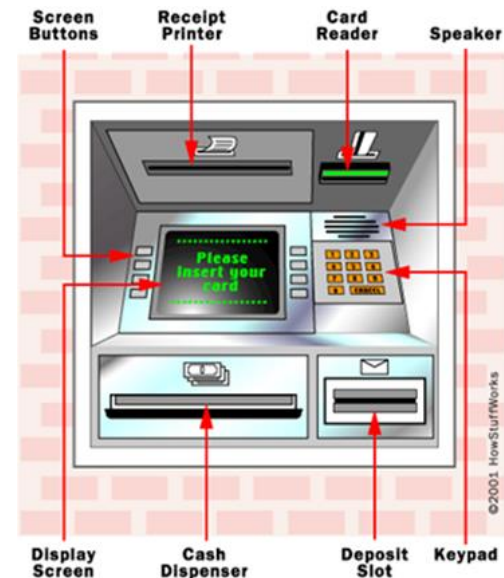
Realising Use Cases

The software development process is all about **realising use cases**.

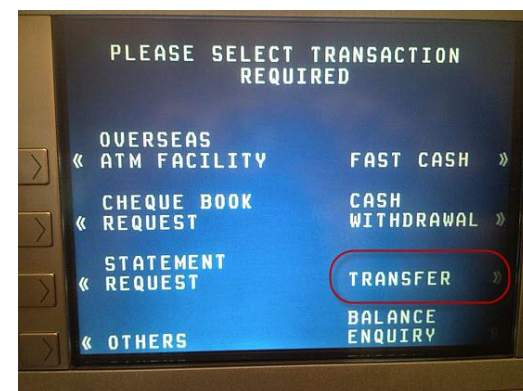
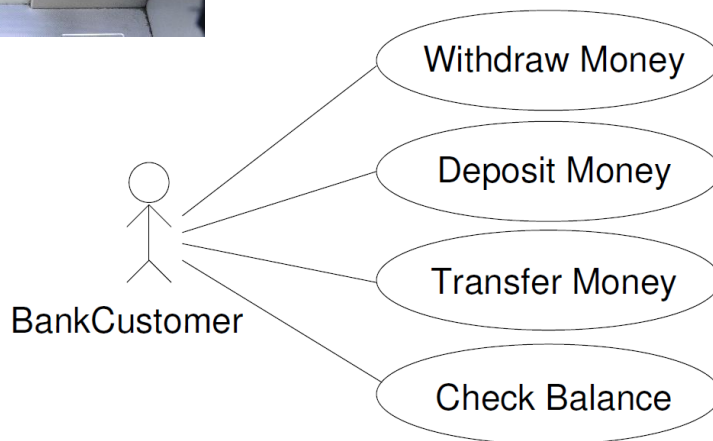
All use cases must be **realised** (to meet all user requirements).

Use cases are realised by **domain classes**

- **conceptual classes** that represent **entities** in the **problem domain**



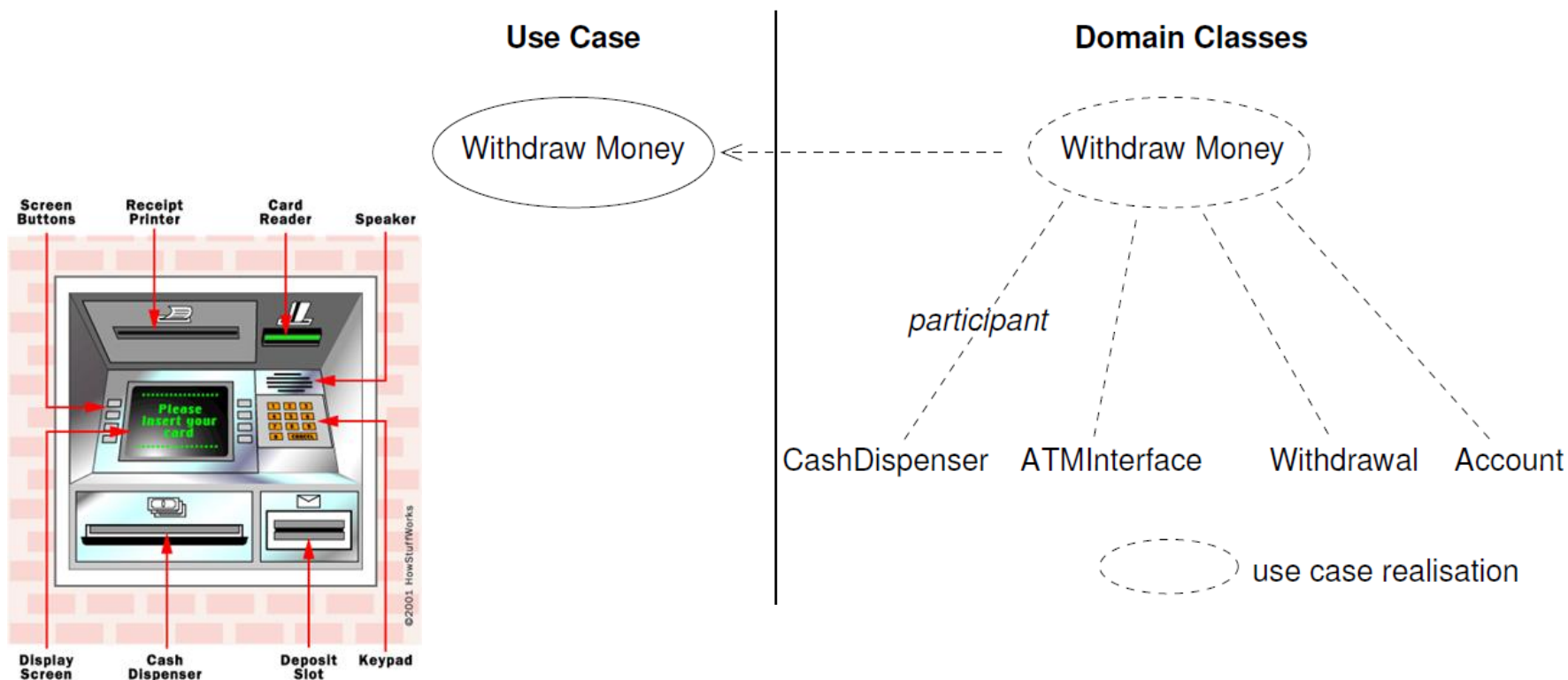
Realising Use Cases: the ATM Example



Use cases for the ATM example

Realising Use Cases: the ATM Example

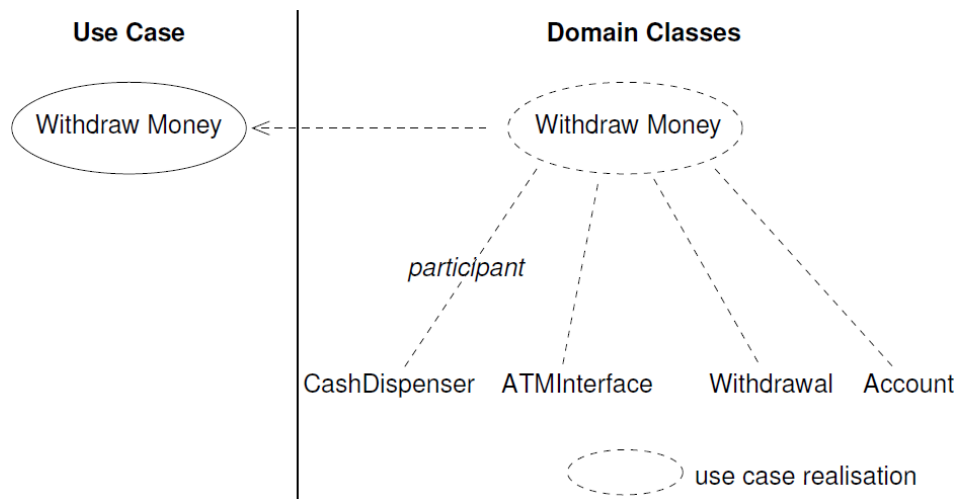
For each use case, domain classes that **realise** the use case must be identified.



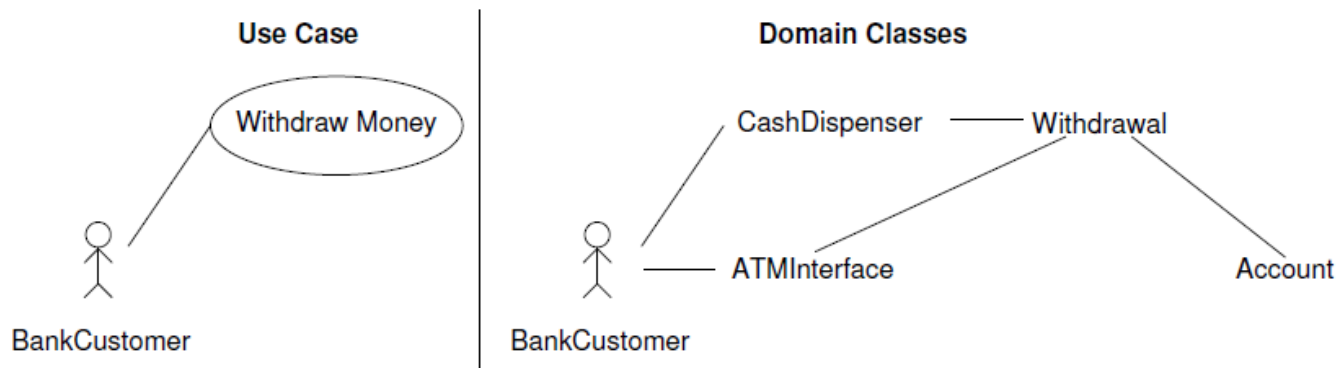
Realising the Withdraw Money use case by **domain classes**

Realising Use Cases: the ATM Example

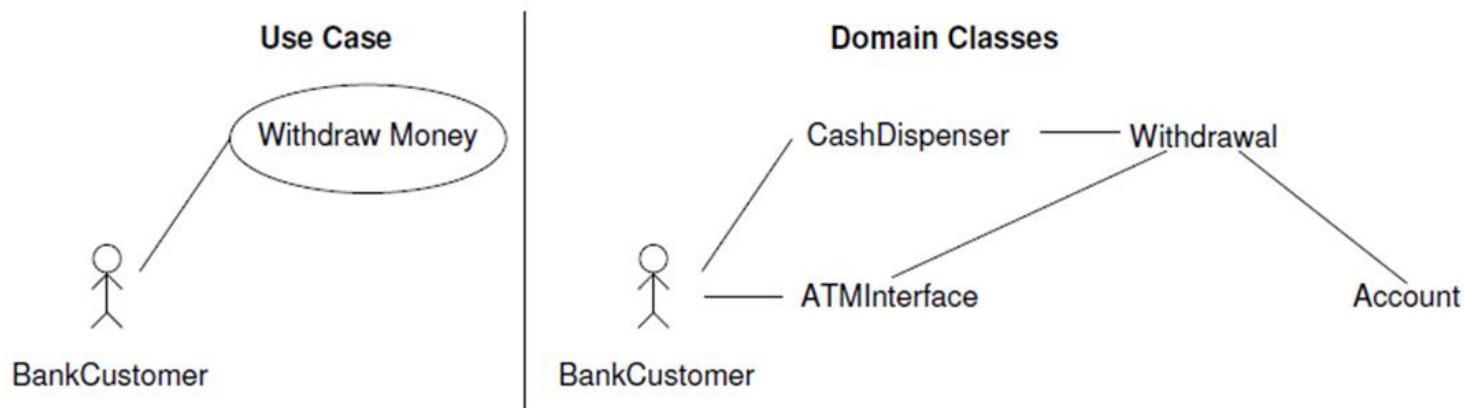
Typically a use case is realised by several **collaborating** domain classes.



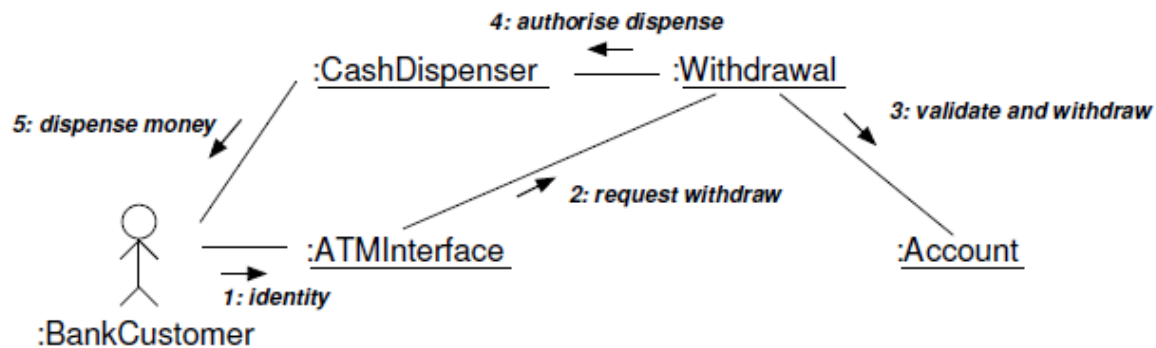
The realisation of a use case can also be modelled as follows:



Realising Use Cases: the ATM Example

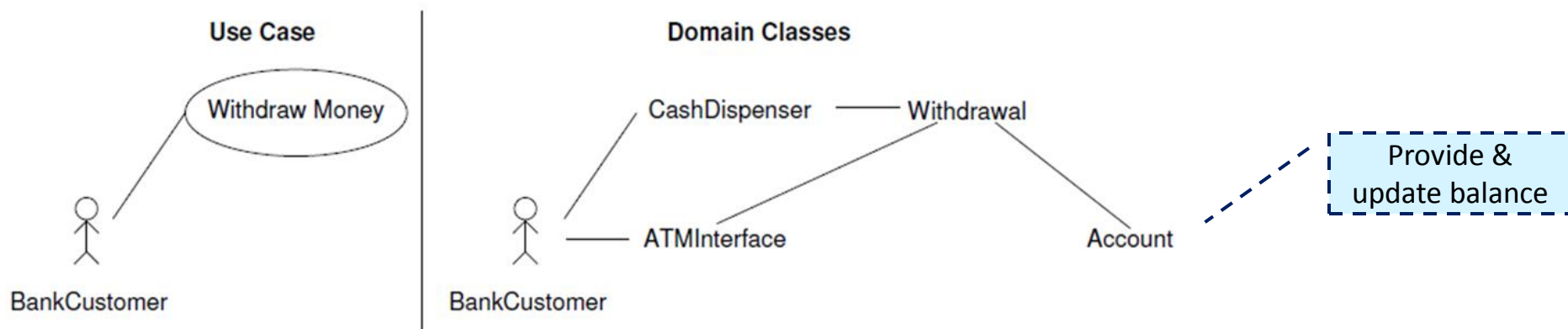


The details of a **collaboration** (of domain classes) can be described by a **communication diagram**.

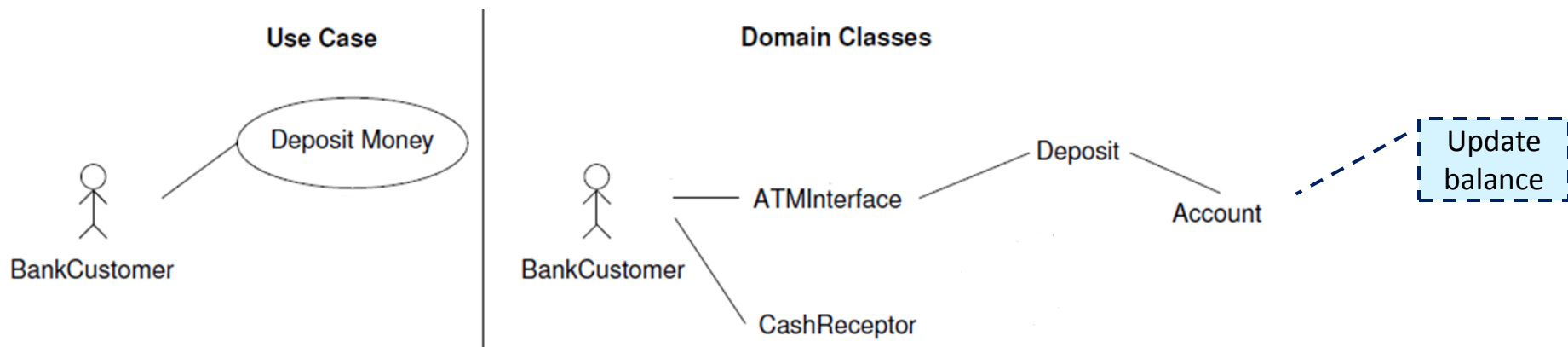


Roles and Responsibilities

A domain class can participate in **several** use case realisations.

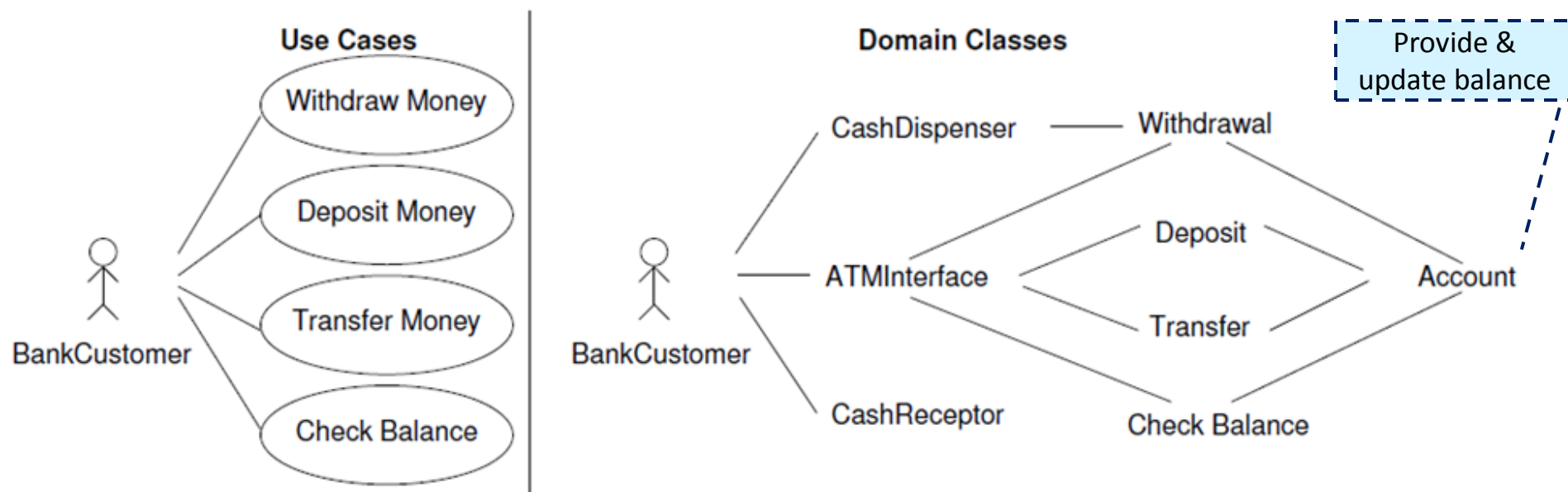


A domain class plays different **roles** in different use case realisations,
i.e. it has different **responsibilities**.



Realising All Use Cases

All use cases must be realised.

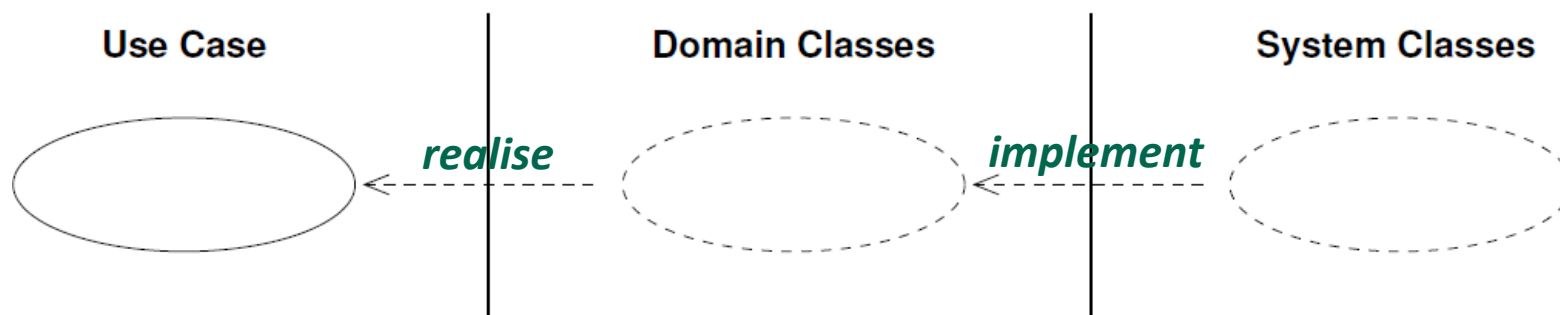


The **responsibilities** of a domain class are the **sum total** of its **responsibilities** in **all the use case realisations** in which it plays a role.

Refining Domain Classes

Domain classes are **refined** into **system classes**

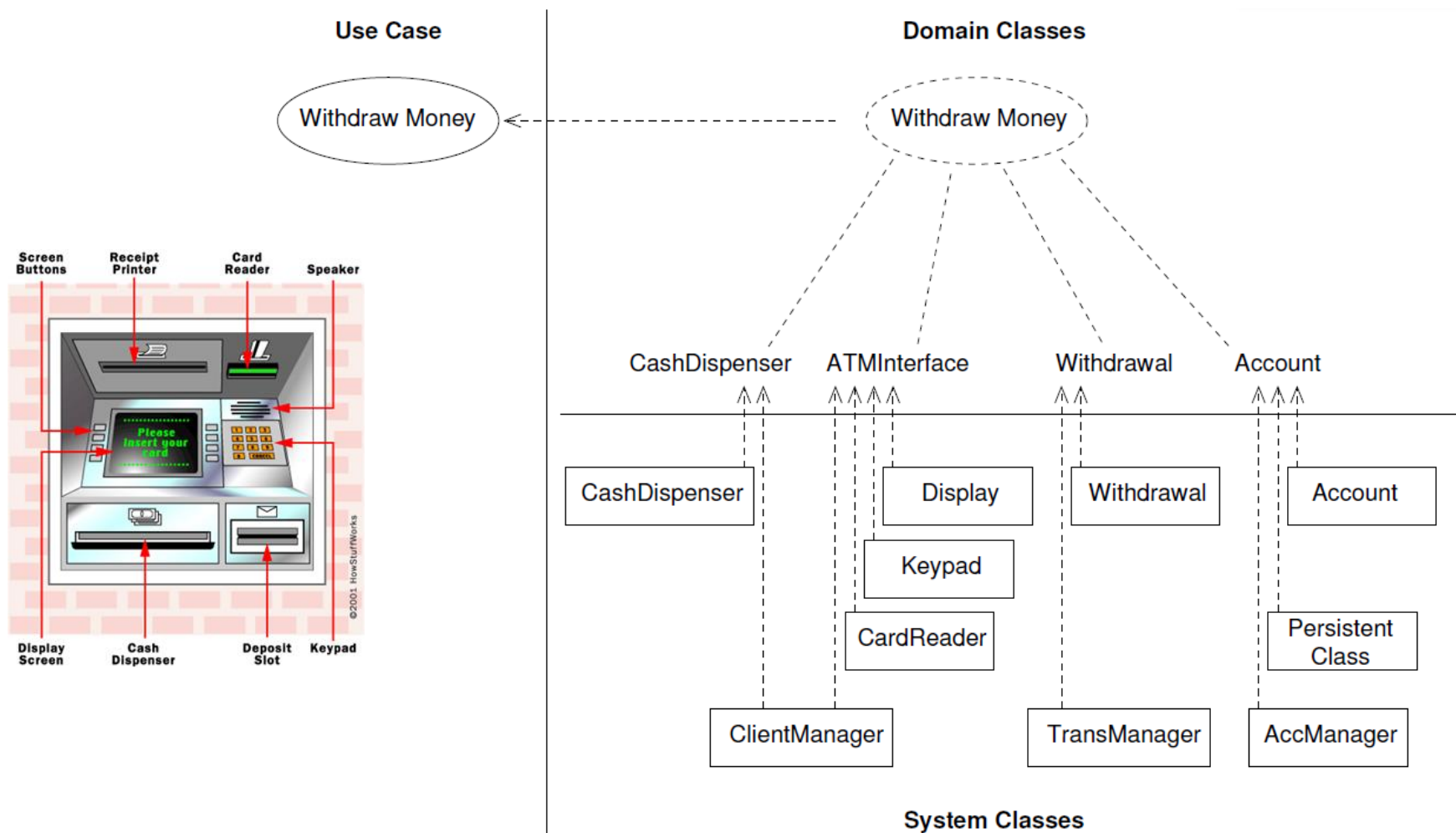
- **real classes** used for **implementing** the system



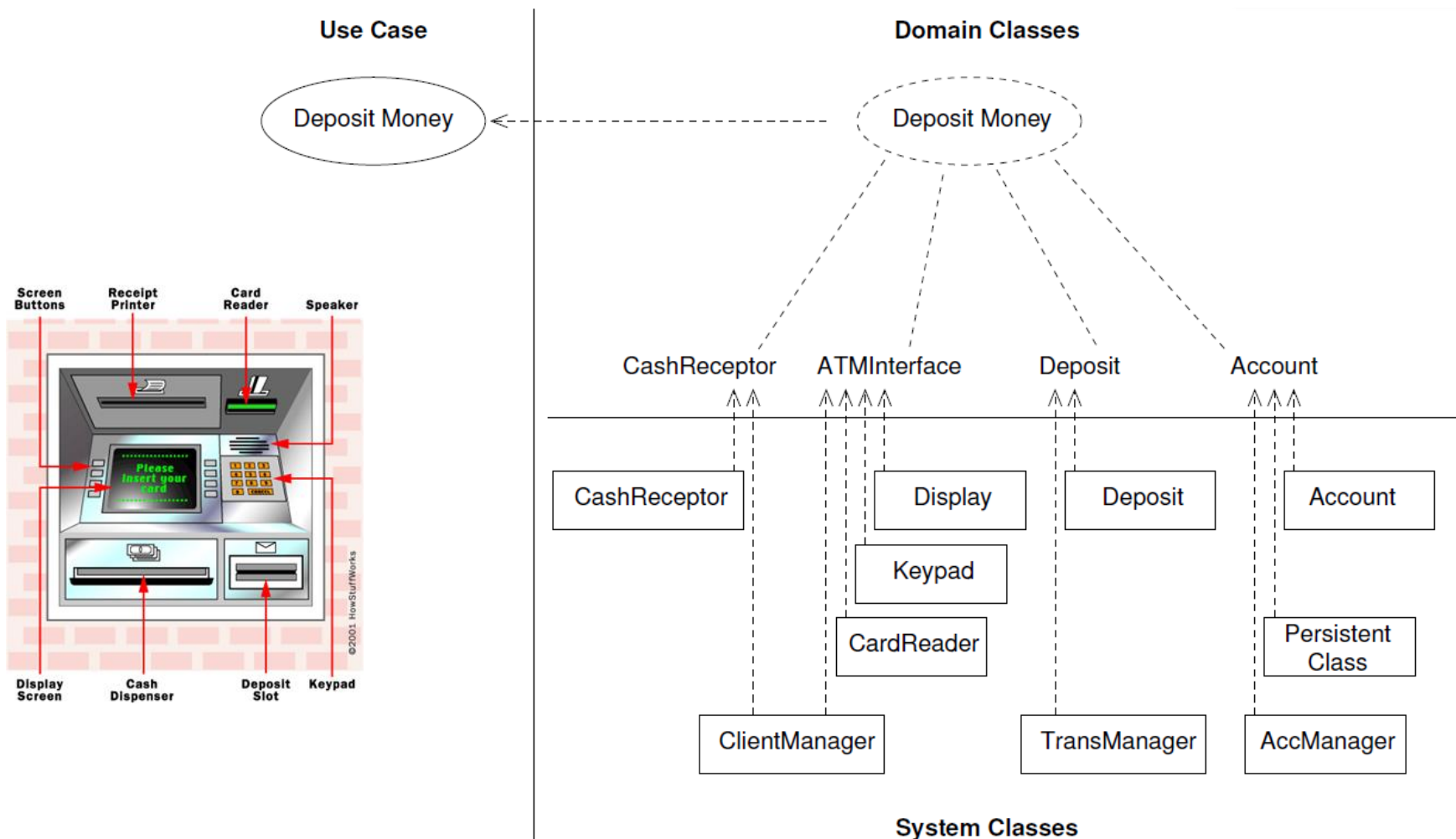
Use case realisation by domain classes and system classes.

During the **system design phase** (of the development process), system classes may be **further refined** into classes used in the **final implementation**.

Refining Domain Classes: the ATM Example



Refining Domain Classes: the ATM Example



Summary

Realising use cases is what it's all about

Use cases are realised by domain classes (and system classes)

A domain class can participate in the realisation of several use cases

Domain classes are refined into system classes

System classes may be further refined into classes used in the final system implementation

Workshop 2

Functional Modelling: Use Cases

Identify use cases

Draw use case diagrams

Write use case descriptions