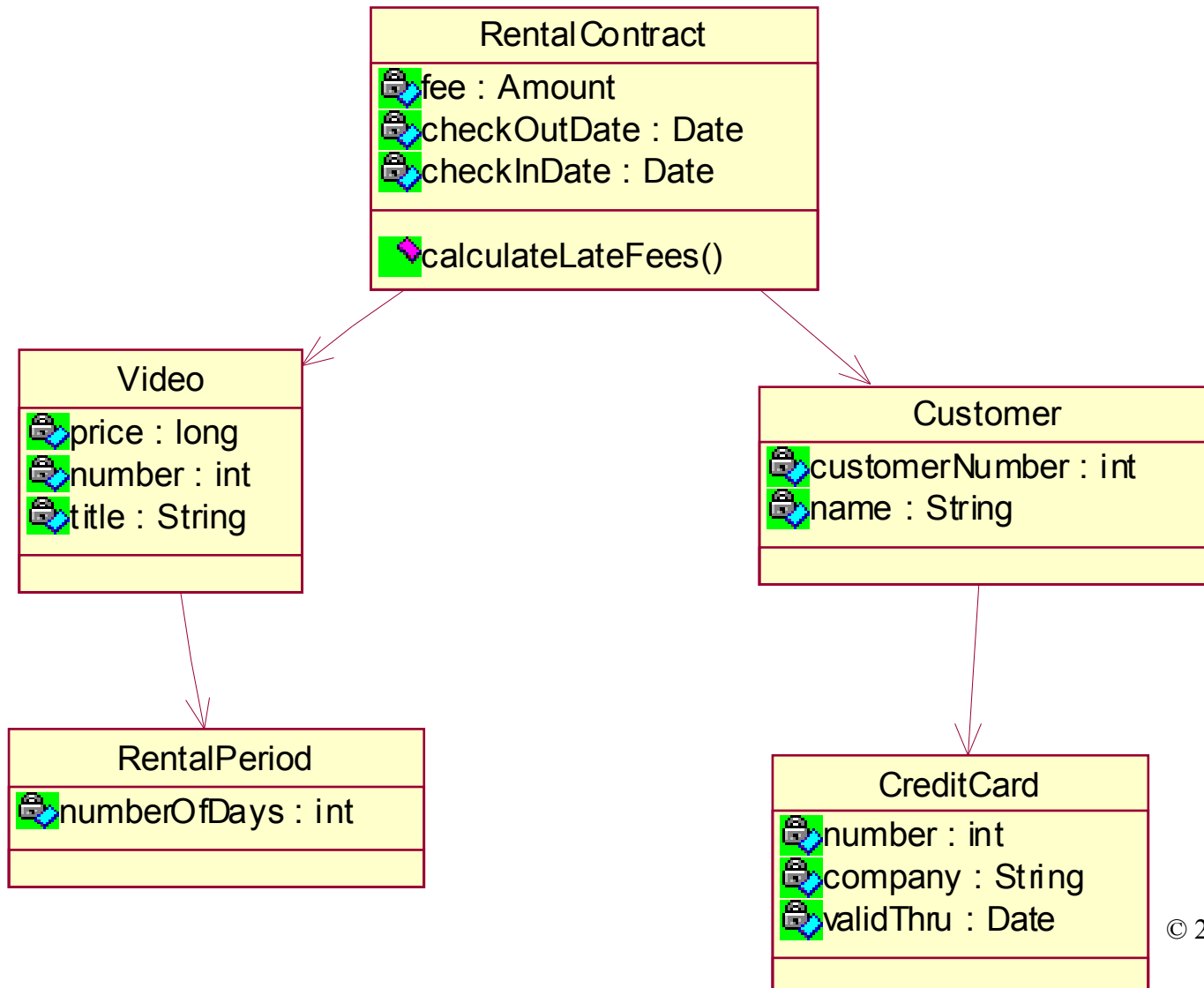


# Revision of Analysis Class Diagram



# An Additional Use Case

## Exchanging the Pricing System

1. Manager enters new pricing system

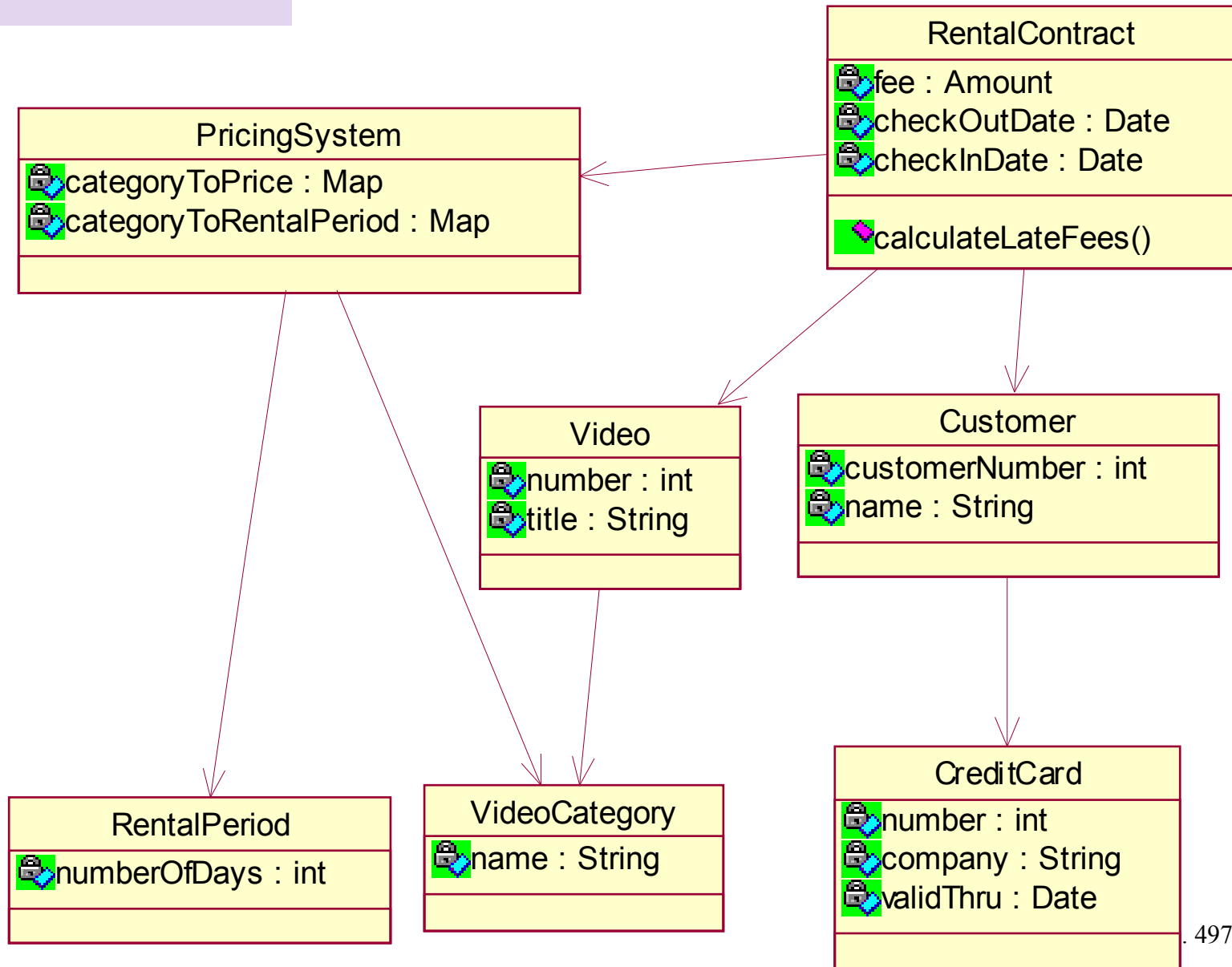
### What is a pricing system?

- It says what category of videos can be rented how long for what price
- For example:
  - new movies: 3 days, \$2.50
  - kids' movies: 5 days, \$2.00

# What does this change?

- We need something to represent a pricing system
- We need categories of videos
- Price can't be a property of the Video any longer
- RentalContract needs to be aware of the pricing system that was valid at time of rental

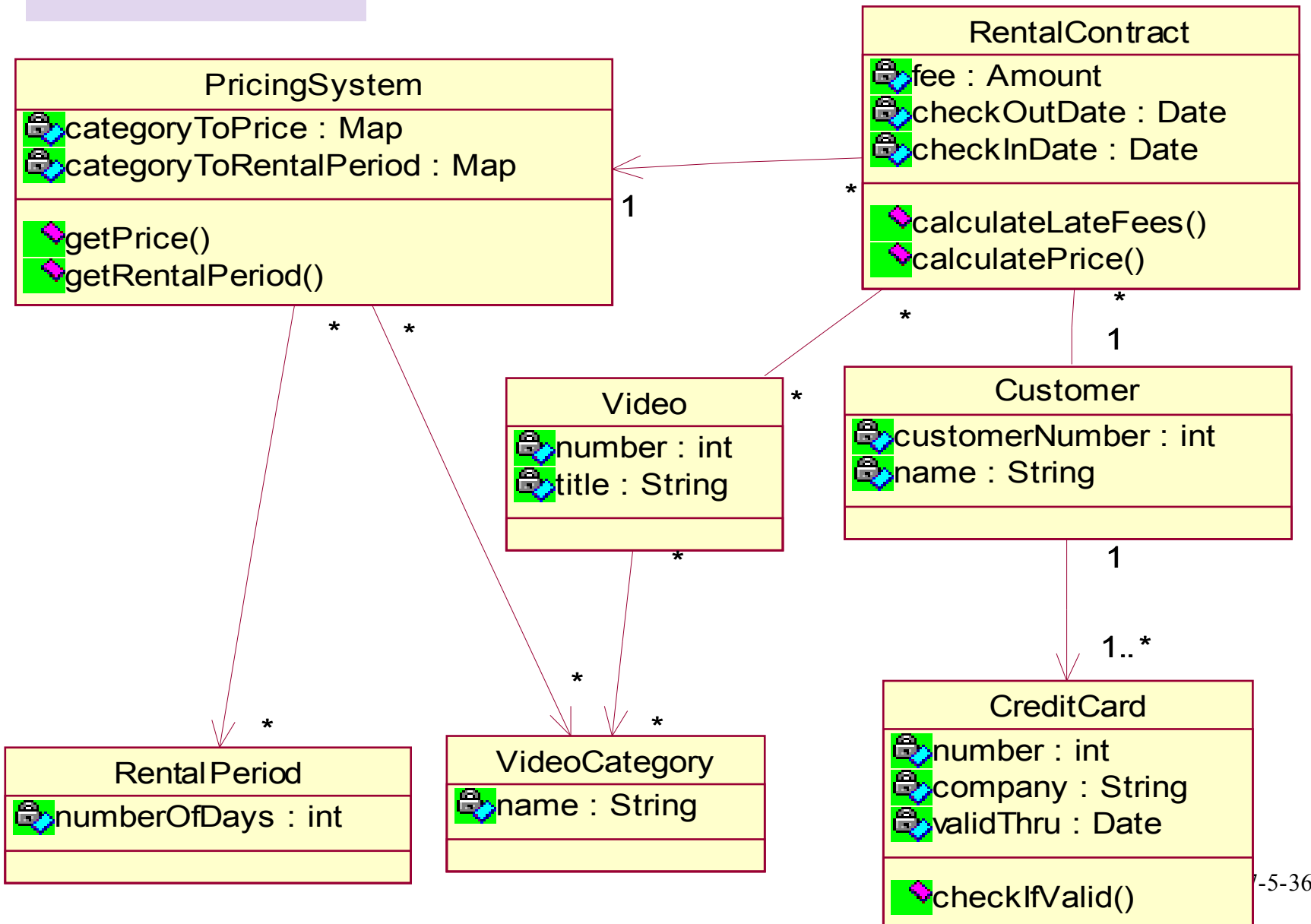
# Version 3 of the Video Store



# Video Store: Finishing Up

- Let's add in multiplicities:
  - How many PricingSystems per RentalContract?
  - How many Videos per RentalContract?
  - How many VideoCategories per Video?
  - How many CreditCards per Customer?
- Some role names to make it clearer?
  - Can't think of any
- Any operations that make it clearer?
  - calculate price of a RentalContract
  - check validity of credit card
  - get rental period of a video

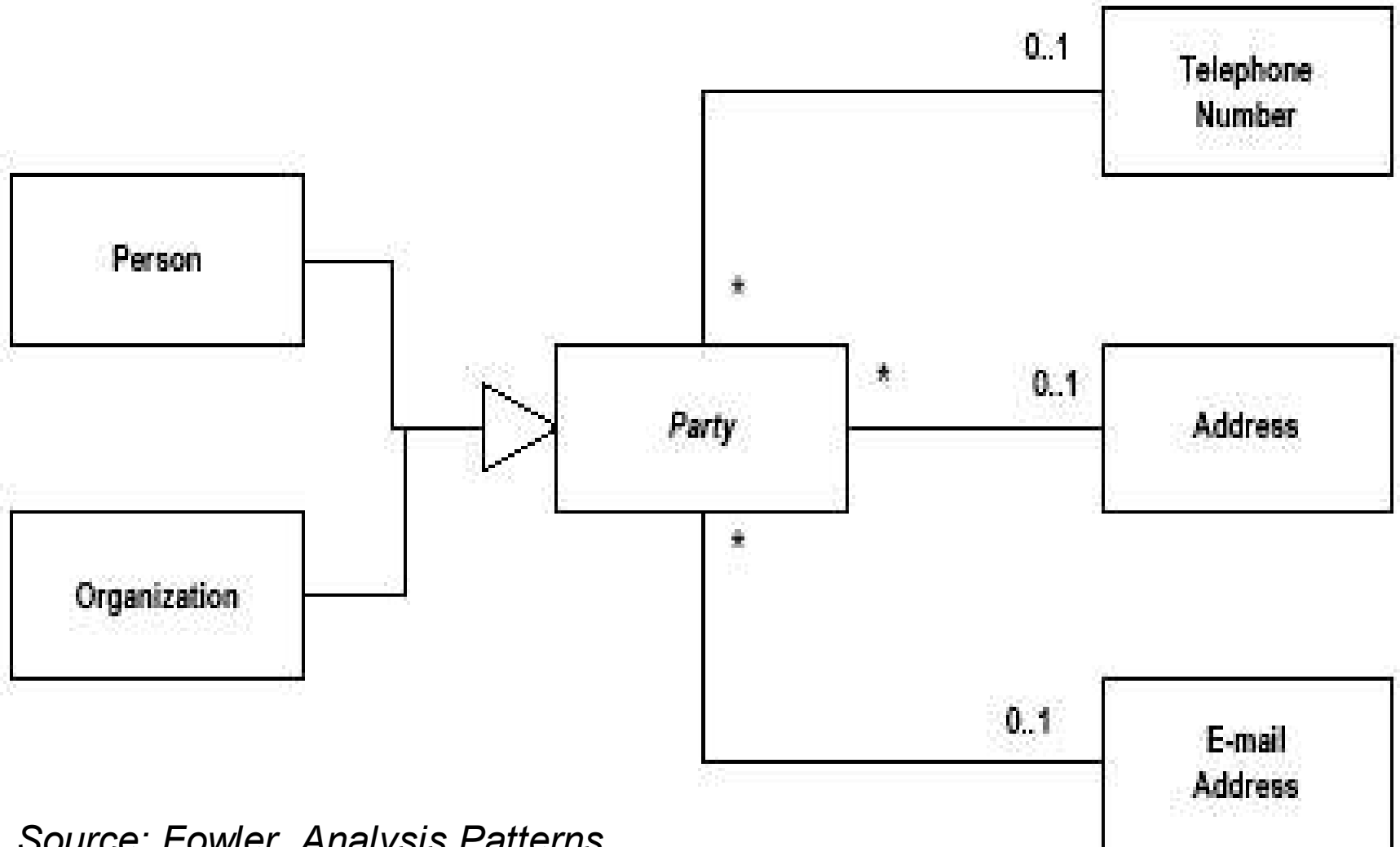
# Video Store – Version 4



## 4.2 Analysis Patterns

- Pattern: A solution to a common problem
- Most famous: Design Patterns
- Analysis Patterns
  - describe problems that often occur during analysis
  - give approximate solutions
  - solution often in form of a UML class diagram
- Disadvantage of analysis patterns:
  - analysis is domain dependent, and so are often the patterns

# Analysis Pattern: Party



Source: Fowler, *Analysis Patterns*