Midterm 2013

Jörg Kienzle

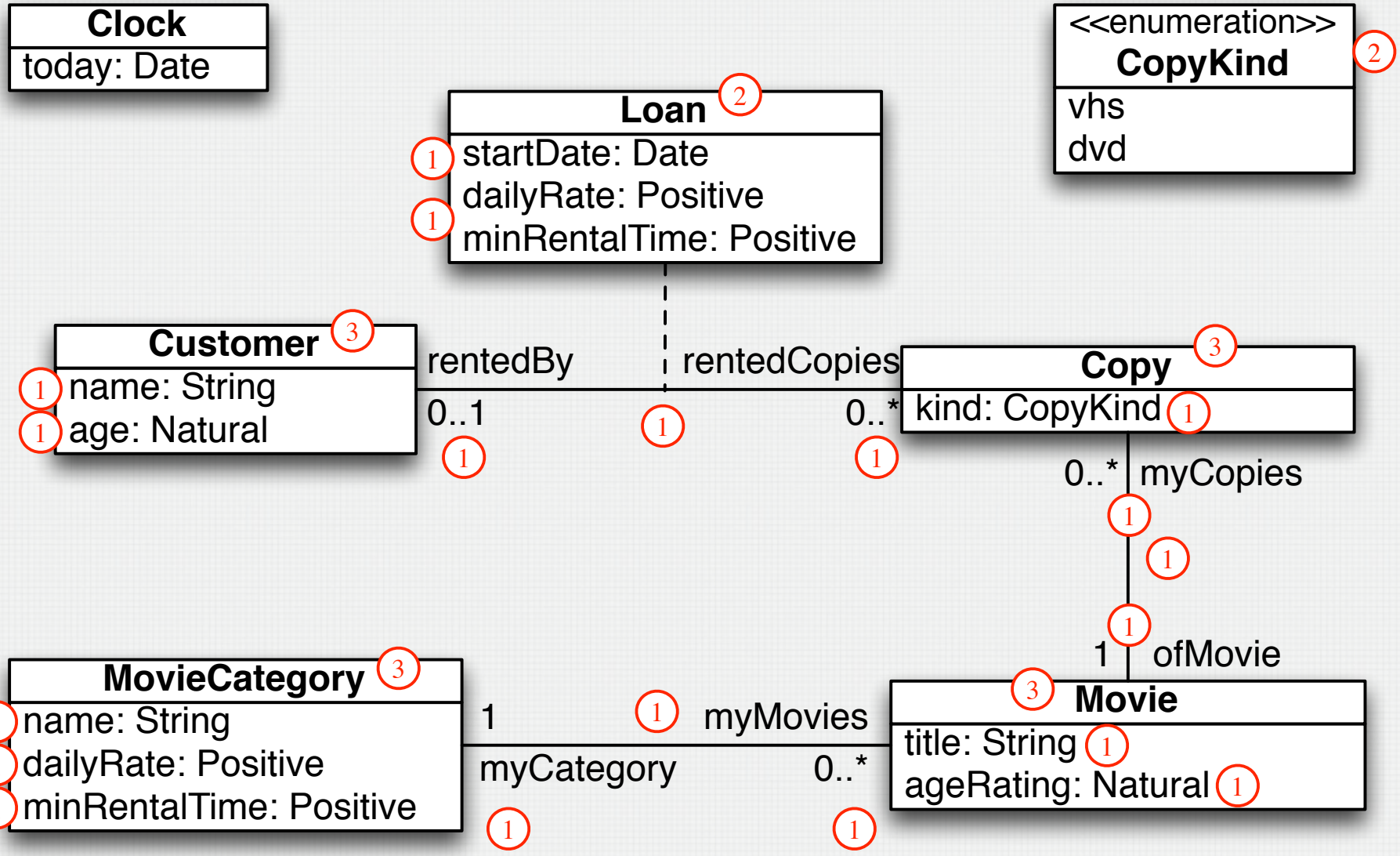
Video Store

A video store rents videos to its customers. In general, a store can have several copies of a movie in stock. Some copies might be VHS tapes, some copies are DVDs. A movie has a name and an age rating. Movies are classified according to at least four categories, e.g. new releases, classics, weekly, and specials. However, it should be possible to add new categories later, without changing the model! Also, the store may change the category of a video at any time. The daily price at which a video is rented out at depends on the category of the video (VHS and DVDs are rented out at the same dayly rate). Some categories have a minimum rental time of more than one day (e.g. weekly rentals are rented out for a minimum of 7 days, minimum rental for specials depends on the kind of special).

The information stored for each customer is his/her name, and his/her age, the movies that he/she is currently renting, and all the state that is needed to determine how much money a customer owes when he/she returns the rented copies.

Devise a domain model (or concept model, if you prefer) that contains the conceptual classes, attributes and relationships relevant in the context of a (single) video store.

Video Store


Clock
today: Date
Loan
startDate: Date
dailyRate: Positive
minRentalTime: Positive
<<enumeration>>
CopyKind
vhs
dvd
Customer
name: String
age: Natural
rentedBy
rentedCopies
0..1
0..*
Copy
kind: CopyKind
0..*
myCopies
1
ofMovie
MovieCategory
name: String
dailyRate: Positive
minRentalTime: Positive
1
myCategory
myMovies
0..*
Movie
title: String
ageRating: Natural

Parking Garage Use Case

The following is an informal description of how an automobilist interacts with a parking garage control system (PGCS) when parking his car. The function of the PGCS is to control and supervise the entries and exits into and out of a parking garage. The system ensures that the number of cars in the garage does not exceed the number of available parking spaces.

The entrance to the garage consists of a gate, a state display showing whether any parking space is available, a ticket machine with a ticket request button and a ticket printer, and an induction loop (i.e., a device that can detect the presence or absence of a vehicle). To enter the garage, the driver, receives a ticket indicating the arrival time upon his request. The gate opens after the driver takes the ticket. The driver then parks the car and leaves the parking garage. In case of problems, the PGCS notifies an attendant by means of an attendant call light.

Parking Garage Use Case (1)

Use Case: EnterGarage
Scope: PGCS
Level: User-Goal
Intention in Context: The Driver wants to enter the garage with his vehicle.
Multiplicity: Only one Driver can enter the garage at a given time for each entry. If there are *n* entries, then *n* EnterGarage use cases can execute at the same time.
Primary Actor: Driver
Secondary Actors: Gate, Attendant
Main Success Scenario:

Driver drives the car to the entrance and stops.

1. *Driver* informs *System* that she is requesting entry.
2. *System* delivers ticket to *Driver.*
3. *System i*s made aware that *Driver* took the ticket.
4. *System* instructs *Gate* to open.

   Driver drives car passed the gate into the garage.
5. *System* is made aware that *Driver* has left the entry and passed the gate.
6. *Gate* informs *System* that it is closed.

Parking Garage Use Case (2)

Extensions:

2a. There are no more parking spots available.

2a.1 *System* informs *Driver* that there are no more parking spots available. Use case ends in failure.

3a. There is a problem with the ticket printer.

3a.1 *System* notifies *Attendant*. Use case ends in failure.

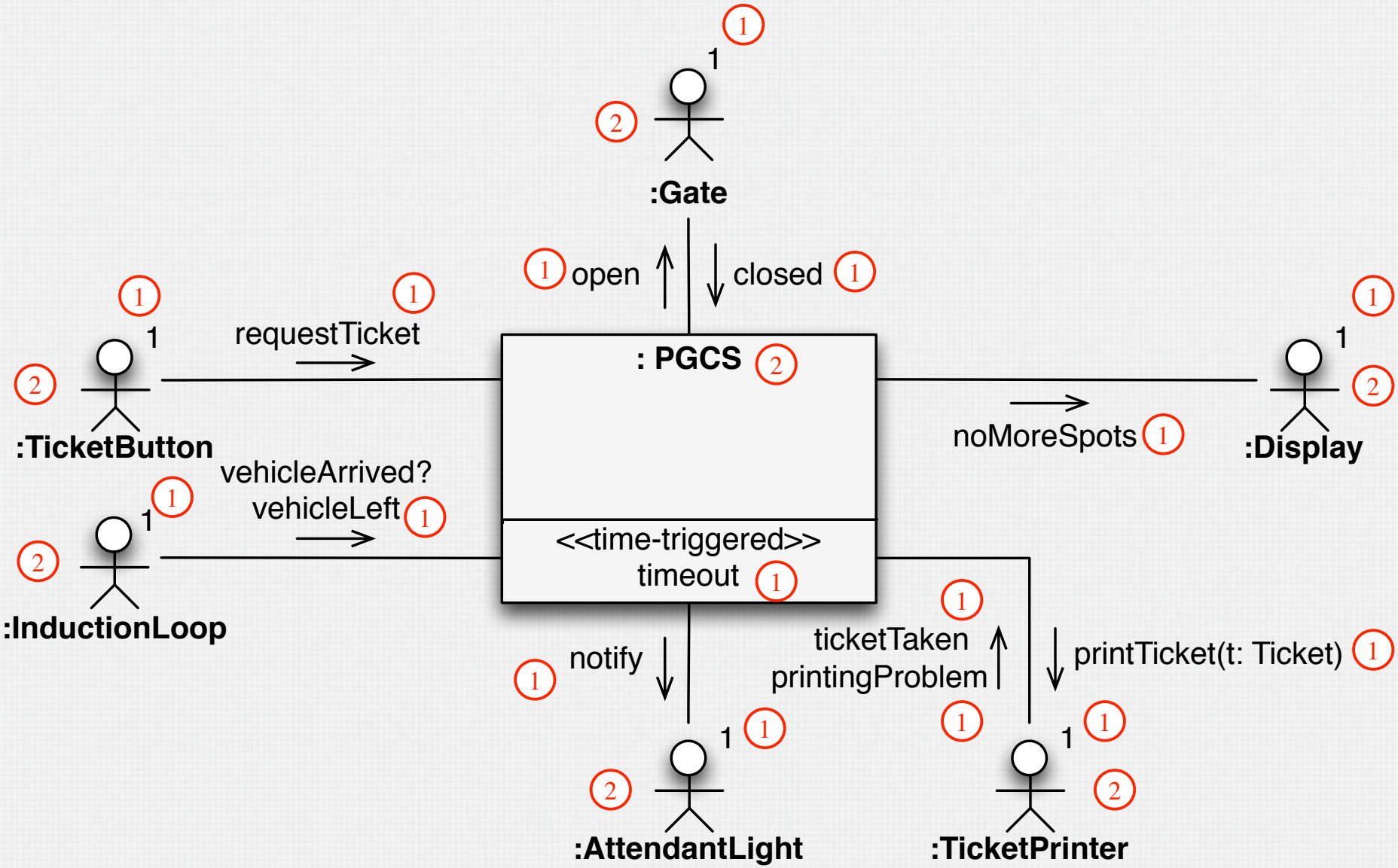
3b. Timeout: Driver has not taken the ticket.

3b.1 *System* notifies *Attendant*. Use case ends in failure.

5a. Timeout: Driver has not driven past the gate.

5a.1 *System* notifies *Attendant*. Use case ends in failure.

PGCS Environment Model



Identify Use Case

In lots of e-commerce applications, users are identified by means of a username and password. Write a subfunction-level use case “Identify User”, in which a user is successively prompted for his username and password. Three consecutive failed attempts (due to wrong password) result in blocking the user account. Make sure that the use case always ends, even if, for example, the user has forgotten his password and decides to abandon the identification process.

Identify Use Case (1)

Use Case: IdentifyUser
Scope: ECommerceSystem
Level: Subfunction-Level
Intention in Context: The User wants to authenticate with the System.

(1) **Multiplicity**: One user can only authenticate once at a given time. Multiple users might authenticate concurrently.

(1) **Primary Actor**: User

Secondary Actors (optional): Keyboard?, Display?

Main Success Scenario (could also be done in just 2 steps):

1. *User* provides username to *System*.
2. *System* prompts *User* for password.

(4)

3. *User* provides password to *System*.
4. *System* notifies *User* that identification was successful.

Identify Use Case (2)

Extensions:

1 (1-4)a. *User* informs *System* that he wishes to cancel the identification process. Use case ends in failure.

2a. System ascertains that the user name is unknown.

2a.1 *System* informs *User* that user name is unknown. Use case continues at step 1.

4a. System ascertains that the password is wrong.

1 4a.1a. *System* informs *User* that password is wrong. Use case continues at step 3.

1 4a.1b. System ascertains that User entered a wrong password for the third time in a row.

1 4a.1b.1 *System* informs *User* that his/her account is now blocked. Use case ends in failure.

Network of ATMs Protocol

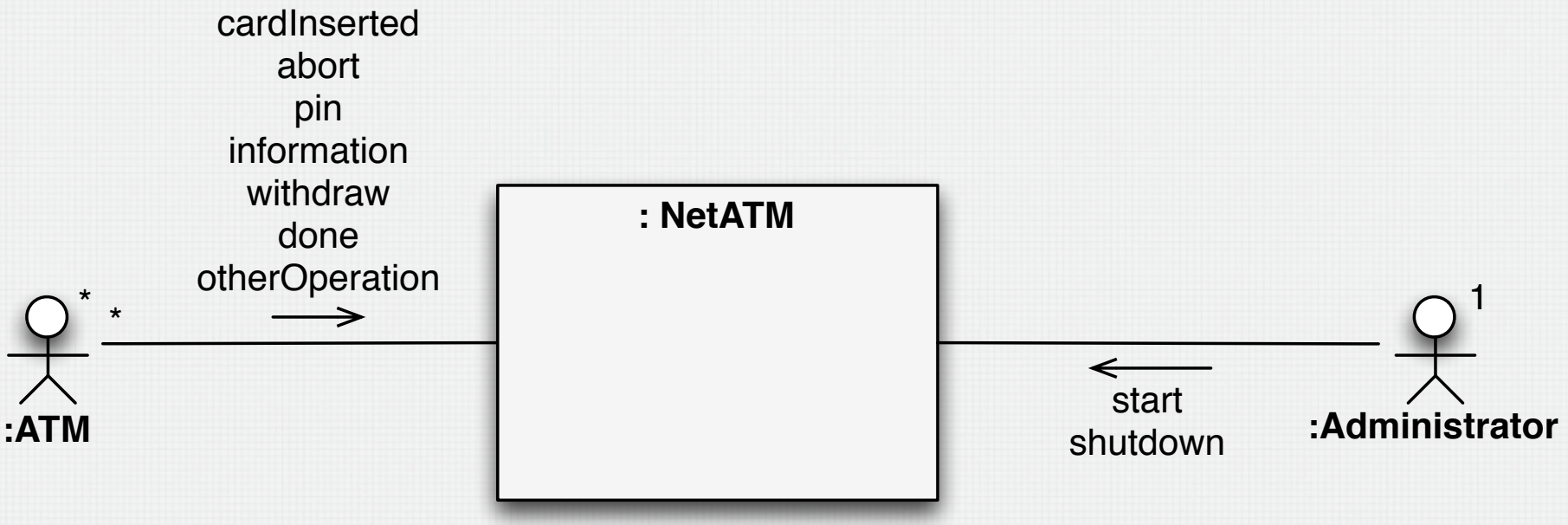
The NetATM system is started by a system administrator (start). Whenever needed, the administrator can also shut down the system (shutdown), with the consequence that all individual ATMs are shut down. Many users can access the system simultaneously, each one standing in front of a different ATM. An individual ATM works as follows:

- The user starts a session by inserting her/his card (cardInserted). (From then on, the user can abort the session whenever it pleases her/him (abort). The session then ends immediately and the card is ejected, so the user can grab it.)
- Then the user must type in her/his personal identification number (pin). Up to three trials are allowed, then the ATM swallows the card, and the session ends.
- Once authorized, the user can either ask for account information (information) or request a withdrawal (withdraw).
- If s/he requests a withdrawal, the specified amount is delivered, a receipt is printed, the card is ejected (so the user can grab it), and the session ends.
- If the user asks for account information, the information is displayed. The user is then asked if s/he wants to perform another operation. If not (done), the session ends. If the user asks for another operation (otherOperation), for security reasons, s/he is asked to provide again her/his PIN, getting again up to three trials.

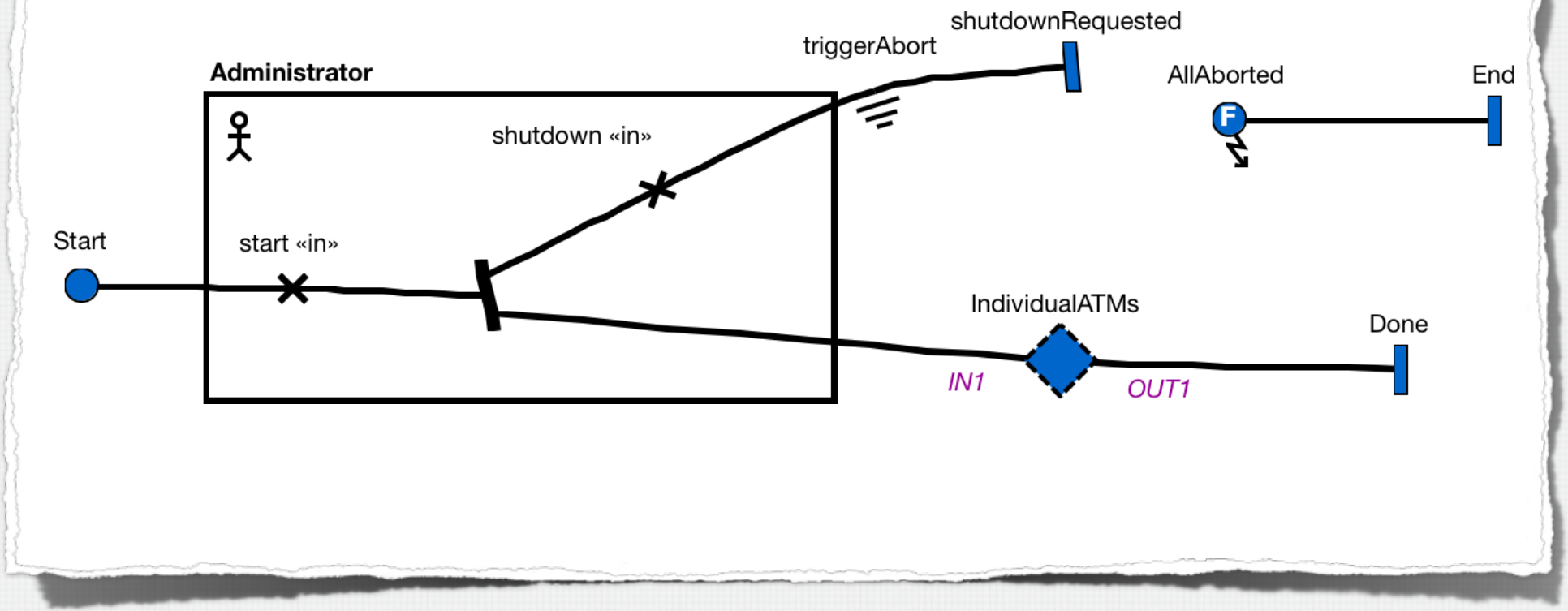
You are asked to provide a Protocol Model for the NetATM system. In particular, your model must take into account:

- that the system administrator can shut down the system at any time,
- that there are many ATMs,
- that a user can abort “her/his” session at any time.

Net-ATM Environement Model



Net-ATM Protocol (1)



Net-ATM Protocol (2)