CSSE 374 – Software Architecture and Design I

Homework 4

Objectives

Practice GRASP principles by applying three of them (Creator, Information Expert, and Controller) to designing portions of Brick Busters Video Store (BBVS).

Practice evaluating your team members using a Performance Evaluation Survey on Angel; get a sense of what is involved constructively evaluating an individual's performance.

Due Date

5 p.m., Tuesday, Week 5, January 11th, 2011.

Because of the exam, no late days are allowed on this assignment.

Tasks

The first three tasks involve a partial design for the BBVS. You do NOT need to do a complete design, just portions that demonstrate your knowledge of the GRASP principles as indicated below.

1. Review the attached Domain Model, System Sequence Diagrams and other artifacts for the BBVS. Identify three different system operations that would be sensible in this domain [e.g., selectVideoToRent(…), returnVideo(…)] where you could apply the GRASP principles Creator, Information Expert, and Controller. As illustrated in your textbook (Chapter 17, Figures 17.4 and 17.5) your partial designs should include design class diagram and/or interaction diagram segments to show where your GRASP principles are applied.
2. Give your partial design for each of the three system operations, as follows:
    a. For one operation, describe how you used the Creator pattern to decide which class created an instance of some other class.
    b. For another operation, describe how you used Information Expert.
    c. For the third operation, describe how you used Controller.
3. For each of your operation designs above, briefly describe the effects of your decision on coupling and cohesion.

The last task is on ANGEL.

4. Evaluate your team members using the Midterm Team Member Performance Evaluation survey on ANGEL. This represents 3 of the 10 points for this assignment, so please take it seriously.

As always, please provide accompanying text and/or embedded notes indicating what you did in your modeling. Please recall that scans of neatly drawn pen and paper sketches are adequate for homework (though not for projects). There is a scanner in F217.

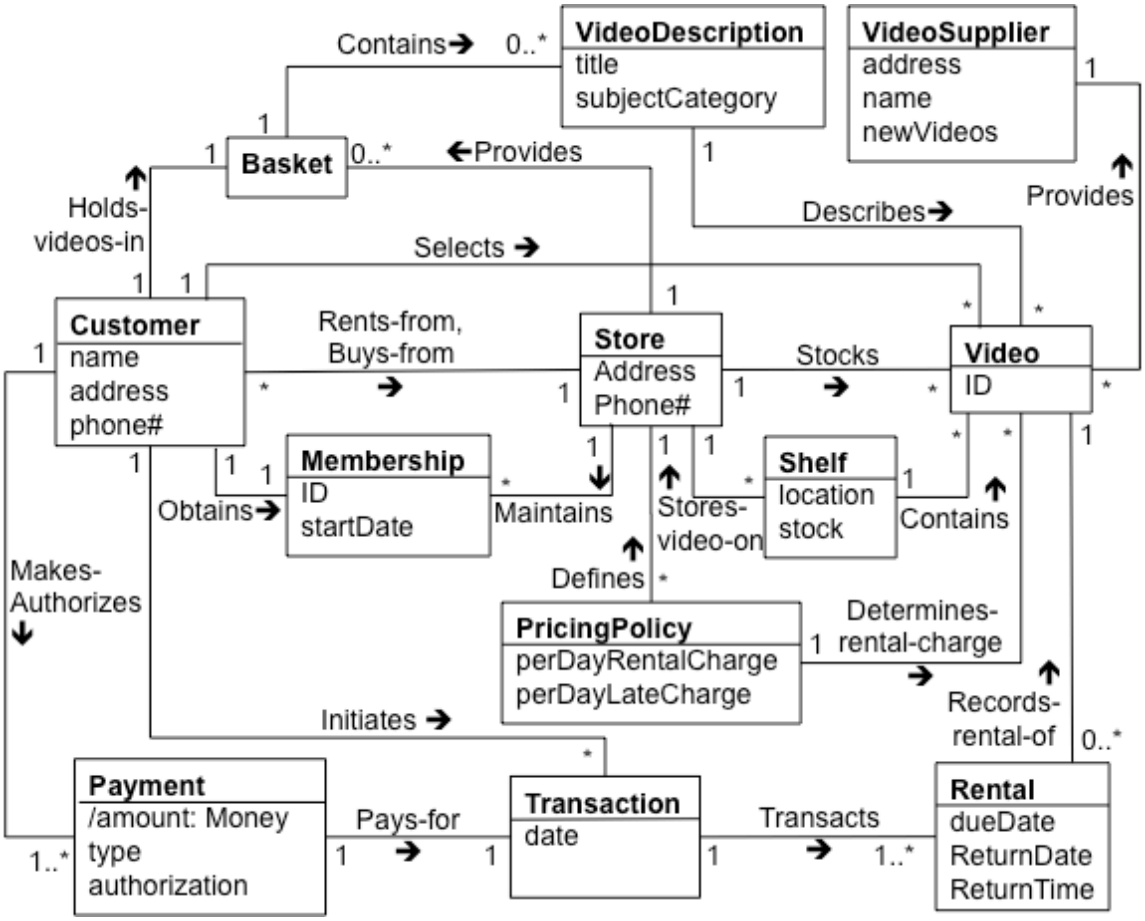
Submitting Your Work

Please submit your GRASP assignment as a single document.

Steve's Section: You can submit either a **pdf** file **or** a **Word** document. Name your document *HW4-GRASP*, with the appropriate extension (*pdf*, *doc*, or *docx*). Please include your name on the first page of the document.

Shawn's Section: Please submit a **pdf** file with a cover page containing your Name, Assignment Title, Date, and Campus Mail number. Please name the document: <your last name>HW4-GRASP.pdf (e.g., Bohner-HW3-LogicalArch.pdf)

Domain Model for BrickBuster Video System (BBVS):

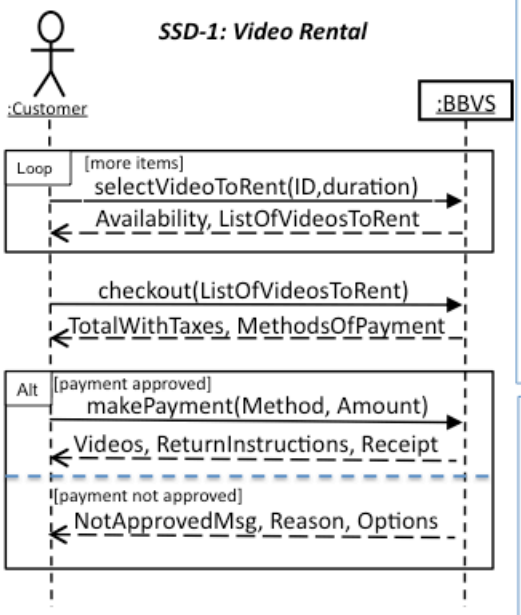


The Domain Model depicted above provides a basic framework from which the Interaction Diagrams, Design Class Diagrams, and Logical Architecture packages/components can be derived. In the figure above, the Customer is a key class that many of the other classes revolve around. The Customer Selects Videos, Buys-from or Rents-from the Store those Videos, and Initiates Transactions, including Making and Authorizing Payments.

New Videos are Provided by Video Suppliers while older releases are sold to customers. The Customer selects from Videos (in the store) from the stock on Shelves, and holds them a Basket – actually, like in a store, these just contain the empty case with the Video Description. The selected videos are then rented or purchased in the rental transaction. Hence, Transaction also plays a central role - it is Paid-for by a customer Payment, and it Transacts the Rental of the Videos. The Rental also is key - it Records-rental-of Videos. The Store is in multiple activities, in that it is where the Customer Rents-from, and it also Stocks the Videos.

System Sequence Diagrams and Operation Contracts

These SSDs and OCs represent key areas of the application of interest, within the domain.



Operation: selectVideoToRent(ID:VideoID, Duration:Days)
Cross-References:
- Use Case 1, SSD-1, Domain Model

Preconditions:
- Customer "Cust" is a member
- Cust item is in basket
- Cust item is in stock
- Cust is done selecting videos

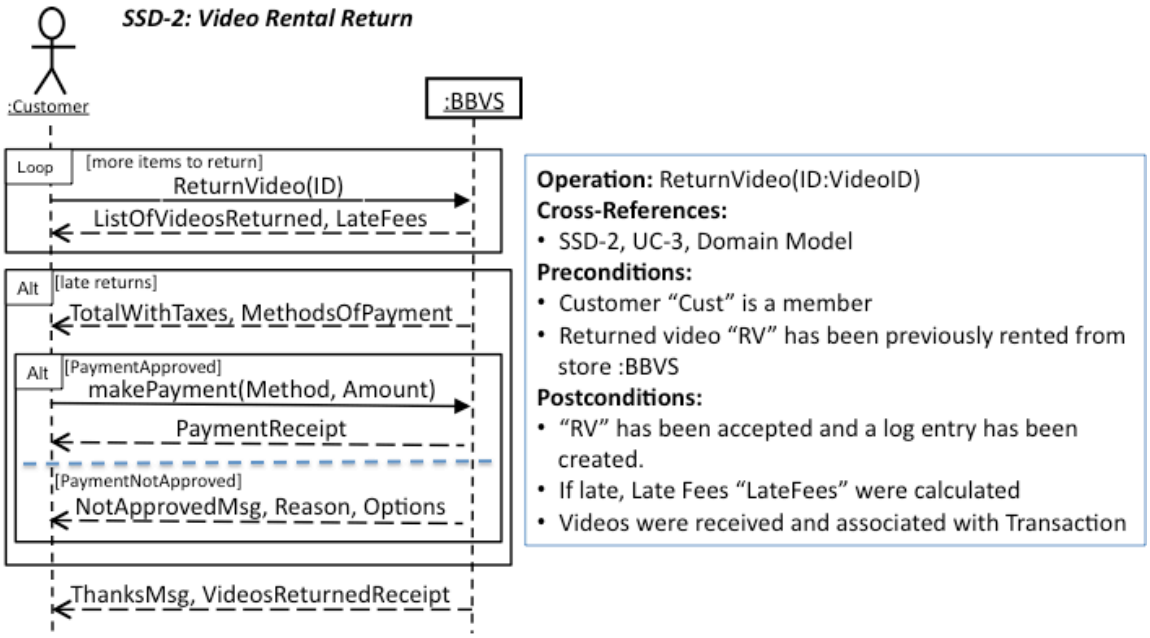
Postconditions:
- Total with taxes was calculated and Payment Attribute amount was updated
- Payment was received and associated with Transaction
- Videos were packaged and associated with Transaction
- Cust item was associated with this sale by this customer
- Rental Transaction was Completed

Operation: makePayment(method: methods, amount: Money]
Cross-References:
- Use Case 1, SSD-1, Domain Model

Preconditions:
- Customer "Cust" has valid Credit card "CC"
- Cust has indicated he done selecting
- There is at least one item in Cust's sale

Postconditions:
- CC authorizer has Authorized Cust's Transaction
- Cust was given receipt and rental return instructions

From Use Case 1, we have the typical scenario where a customer rents multiple videos, and the box around that part is to indicate this loop. The OC's detail two important parts of the transaction, the renting of individual items (videos), and the acceptance of payment, after which the customer can watch their videos.



From Use Case 3, we have the scenario where a customer returns multiple videos, and the box around that part is to indicate this loop. The OC details when the video is returned, it is logged, and if it is late, late fees are tallied. The rest represents the alternatives of if there is a late fee assessed, how that fee is collected.

Use Cases:

In getting these use cases from the customer, we tried to make them separate from the way the customer now does business. That is, we wanted them to be "how they wanted the new system to work." At the same time, we wanted to keep the use cases general enough that they might apply to more than one way of running a video business.

UC1: Customer rents videos

Preconditions: Customer has a membership, has selected videos they want to rent, and made the system aware of their choices.
Actors: Customer (if self-service or remote), or store associate (if in a store).

Main flow:

1. Actor indicates to rent first item (e.g., clicking "rent" on a networked device, or scanning it physically in a store).
2. System verifies immediate availability, waits for actor to make next option.
3. Actor indicates they are done selecting.
4. System shows total, prompts for payment.
5. Actor selects method of payment, entering additional data if needed (e.g., credit card number).
6. System verifies the payment has gone through, schedules the goods for rental (e.g., sets up a window to click on to view the video remotely, or tells the store clerk where to find the DVD).

Alternate flows (among many):

2a. System tells actor that the video is not currently available, and provides information on when it will be.
3a. Actor buys additional items, in the same way, if desired, returning to step 3 after each.
6a. System rejects method of payment, asks actor for alternative.

Postconditions: Rental transaction is complete.

UC2: Customer selects videos to rent

Preconditions: None
Actors: Customer

Main flow:

1. "Store" shows videos available for rental. (This could be in stages, such as indicating areas of the store for customer to browse, or responding to a search request.)
2. Customer picks a video to rent.

3. System shows items in "basket," suggests checkout.

Alternate flows:

1a. Item is not in stock. "Store" shows when it is available, if possible.
2a. Customer continues in this loop, selecting multiple videos.

Postconditions: Customer has products ready to rent.

UC3: Customer returns videos to store

Preconditions: Customer has a membership, and Video(s) rented within last 30 days (assumption that after that, the customer has been billed).
Actors: Customer (if self-service or remote), or store associate (if in a store).

Main flow:

1. Customer provides video(s) for return.
2. System provides indicator of each video returned, and any accumulated late fees.
3. System acknowledges the return of all the videos returned this time and provides a receipt. A “thank you” message concludes the transaction.

Alternate flows:

2a. Late return: System shows late fee total, prompts for payment.
2b. Actor selects method of payment, entering additional data if needed (e.g., credit card number).
2c. System verifies the payment has gone through, and a payment receipt is provided.
2c.1. Payment not approved: Customer is alerted that the payment was not approved and

Postconditions: Video(s) returned to stock, return transaction recorded, and customer gets receipt. Video return transaction complete.