

PROGRAMMER MANUAL

Blockwood Check-out and Management System

- Technical Introduction
- System Requirements
- Installation Instructions
- Technical Specifications
- Database Description
- Domain Diagram
- Sequence Diagrams
- JavaDoc API Documents

TECHNICAL INTRODUCTION

Welcome to the programmer manual for the Blockwood Video Checkout Management System. Throughout the development of this system, we have kept in mind possible future additions and modifications. We have implemented an object-oriented methodology throughout the system and hope that it is as understandable as possible. Below, you will find system requirements, installation instructions, technical specifications outlining the composition of the system, and a walkthrough of the underlying database. Finally, we have also included sequence diagrams and JavaDoc API documents for a more detailed look at the system processes.

SYSTEM REQUIREMENTS

- Windows Server 2003 with Service Pack 1
- .NET 2.0 Framework
- Java Runtime Environment 1.5.x
- PC with 550-MHz or faster processor recommended.
- 128 MB of RAM (256 MB or more recommended)
- 10 GB of hard-disk space

INSTALLATION INSTRUCTIONS

Router

If using a router, port forward port 22 and 80 to the server

Database

- Run SQL Server 2005 Express Installation
 - When prompted for the authentication method, select mixed
 - Enter a password for the SA account
- Install the management console (SQL Server 2005 SSMSEE)
- Run the SQL Server Configuration Manager
 - Select the SQL Server 2005 Network Configuration
 - Select Protocols for SQLEXPRESS
 - Right click TCP/IP and select “Enable”
 - Right click TCP/IP and select “Properties”
 - Click on the IP Addresses Tab
 - Set the TCP port to 22
- Verify that Remote Connections are allowed
 - Run the Microsoft SQL Server Management Studio Express
 - Right click on the database and select Properties
 - Select the Connections option and verify that “Allow remote connections to this server” is checked
- Restart SQL Server
 - Open a command prompt (Start → Run and type cmd and hit enter)
 - Type “net stop mssql\$sql\$express”
 - Type “net start mssql\$sql\$express”
- Create a new database
 - Right click on databases
 - Name the database “intex2” and hit ok
 - Open db.sql and click execute

Internet Information Services (IIS)

- In the “Manage Your Server” screen, check to see if the Application Server role has been installed
 - If not, select Add or Remove a Role and go through the process to add an Application Server

Tomcat

- Run the Apache Tomcat Installation
- Install the Tomcat redirector (isapi_redirect.msi)
- Configure IIS to handle new server extension
 - Go into Manage this Application Server in “Manage Your Server”
 - Select IIS Manager
 - Select Web Service Extensions

- Select Add a New Web Service Extension
 - Name: “Tomcat”
 - Path %Install Directory%\Apache Software Foundation\Jakarta Isapi Redirector\bin\isapi_redirect.dll
 - Allow
- Restart IIS
 - Go into “Manage Your Server” and select Manage this application server
 - Click on IIS Manager → Server Name → Websites and right click on the default website and select stop and then start
- Edit the following file %Install Directory%/ Apache Software Foundation\Jakarta Isapi Redirector\conf\uriworkermap.properties
- Add the following lines of code
 - /*.jsp=wlb
 - /*.actions=wlb
 - /*.css=wlb
 - /WebIntex2/images/*.gif=wlb
- Add the jar file to Tomcat
 - %Install Directory%/ Apache Software Foundation\Tomcat 5.5\webapps
 - Restart Tomcat

TECHNICAL SPECIFICATIONS

The Blockwood Video system is separated into five different layers of functionality. These layers include (a) database layer, (b) data access layer, (c) business layer, (d) control layer, and (e) display layer. Below are descriptions of each of the layers and how they interoperate with each other.

Database layer. The database layer is the grass roots of the system and runs on SQL Server 2005. This is where all data is placed for long-term storage. There are approximately 24 different tables that store information about transactions, accounts, rental videos, sale videos, refreshments, stores, and more. The database is directly accessed by the data access layer.

Data Access Layer. The purpose of the data access layer is to provide a pipe between the database layer and an object-oriented framework. It is composed of a data access object (DAO) for each table in the database layer. At the command of a system event, the data access layer retrieves raw data from the database layer through SQL statements and compiles it into business objects that represent real-life entities. The

data access layer can also be called on to save new or modified business object information to the database for long-term storage.

Business Layer. The business layer is composed of business objects (BO) that represent real-life entities such as a customer, transaction, or product. These business objects are retrieved or created by the data access layer and are capsules of information about the entity they represent. For example, a business object representing a customer contains the following attributes:

- First name (String)
- Last name (String)
- Address (String)
- City (String)
- State (String)
- Zip Code (String)
- Phone Number (String)
- Account (BO)

This is an object-oriented framework that is easy to understand and manipulate.

Control Layer. The control layer performs the grunt of the system logic. This layer is generally called on by the display layer, manipulates the business layer, and commands the data access layer. There is usually a controller for each main system process that takes place. For example, the transaction controller performs most logic related to a transaction. Its responsibilities include adding items to a transaction, removing items from a transaction, creating related journal entries, calculating totals, adjusting inventory levels, adjusting account balances, and more. Web controllers also directly interact with http requests in order to provide data to the web display layer. For easy portability, this layer of logic and control should be kept separate from the display layer as much as possible.

Display Layer. The display layer is everything the user sees. It provides the user with an easily understood interface that interacts directly with the control layer. The display layer contains minimal logic, enabling low-hassle customization and enhanced portability as features are added or modified. This layer has been created with usability in mind and provides the quickest method possible for processing transactions and performing other business functions. Both the application GUI and the web GUI are parts of the display layer.

DATABASE DESCRIPTION

The underlying database consists of 24 separate tables that retain information for long-term storage. Below, you will find a description of these tables and their relation to other tables as well.

Transaction. This table contains information relating to specific transactions. The information includes the transaction date, tax, and total of each transaction. Each transaction is related to one payment record, one to many journal entries, one customer, one store, and one to many transaction lines.

Payment. This table contains specific information concerning a transaction payment including the amount, amount tendered, and the change given. Each payment is related to one transaction.

Journal Entry. This table includes accounting information for making better business decisions and for auditing procedures. It contains information about the general ledger account, debit or credit, amount, and the date posted. A transaction will likely have many journal entries.

Customer. The table includes information about customer including their name, address, and phone number. One account can be used for many customers, but a customer must only belong to one account. Each account is owned by one customer.

Account. This table includes information about accounts including the account number, credit card information, and a balance. As noted before, one account can be used for many customers, but a customer must only belong to one account. Each account is owned by one customer. An account is related to a membership and the store that created the account.

Store. The store table contains information about each Blockwood Video store. This information contains location and contact information, the tax rate, and the minimum fee amount (how high an account balance can be while still allowing a customer to rent).

Transaction Line. The transaction line table contains information about each item on a transaction. This includes the serial number, quantity, and subtotal. Each transaction line only belongs to one transaction and is associated with one revenue source.

Revenue Source. Although not a physical table in the database, this represents an abstract superclass of several revenue source types; namely, membership, rental, fee, and product. The specific revenue source subclass is determined by the associated sku length, therefore making it unnecessary to store a table identifier in a revenue source table.

Membership. This contains information about specific account memberships. This information includes the start date, expiration date, and cancel date if applicable. Each membership is associated with one account, and a given account is associated with one active membership, though inactive past memberships are also stored for archival purposes.

Membership Type. These are the different membership types customers can choose from when they create an account. Information stored includes the membership type sku, description, price, and number of videos allowed out simultaneously. Initially, accounts are associated with a free membership type. With this membership type, they must pay for each rental but no monthly membership fees are charged. Each membership has one membership type.

Rental. This table contains information about specific rentals. Information stored includes the date the rental video was checked out, when it is/was due, and when it was checked in (if applicable). Each rental is always associated with one rental video; and, if late, damaged, or not returned, associated with a conceptual fee.

Conceptual Fee. These are fees that are applied if a rental is late, damaged, or not returned. Information stored includes a sku, description, and amount.

Fee. Fees are associations between rentals and conceptual fees. It also stores a quantity. This number will be more than one if a rental is returned several days late. In this case, the quantity reflects the number of times (days) the “late” conceptual fee will be applied.

Release Type. Release type refers to whether a video is regular or new. These are items that need to be differentiated because of the current pricing scheme at Blockwood Video. Release types are matched with video categories to create video category release types (VCRT).

Video Category. Video category refers to whether a video is a DVD, extended DVD, or VHS. Again, these are items that need to be differentiated because of the current pricing scheme at Blockwood Video. Video categories are matched with release types to create video category release types (VCRT).

Video Category Release Type (VCRT). This table contains combinations of video categories and release types. Information held in this table also include the price, rental duration, and the overdue price (for rentals over one month overdue). Video category release types are combined with conceptual videos to create video category release type conceptual videos (VCRTCVC).

Conceptual Video. Conceptual videos are titles of videos. These are not individual, specific videos, but collections of videos distinguished by title. These are combined with VCRT's to create VCRTCVC's.

Video Category Release Type Conceptual Video (VCRTCVC). This table contains combinations of video category release types (VCRT) and conceptual videos.

New Used. These are two categories that distinguish whether a sale video is new or used. These are combined with VCRTCVC's to create specific sale videos. New used records do not affect rental videos.

Rental Video. This table contains specific, physical rental videos that are matched with a specific store and VCRTCVC. Other information includes the rental video's serial number, status (in or out), time of reservation (if applicable), and reservation account (if applicable). Each time a rental video is rented, and rental record is created and associated with the rental video.

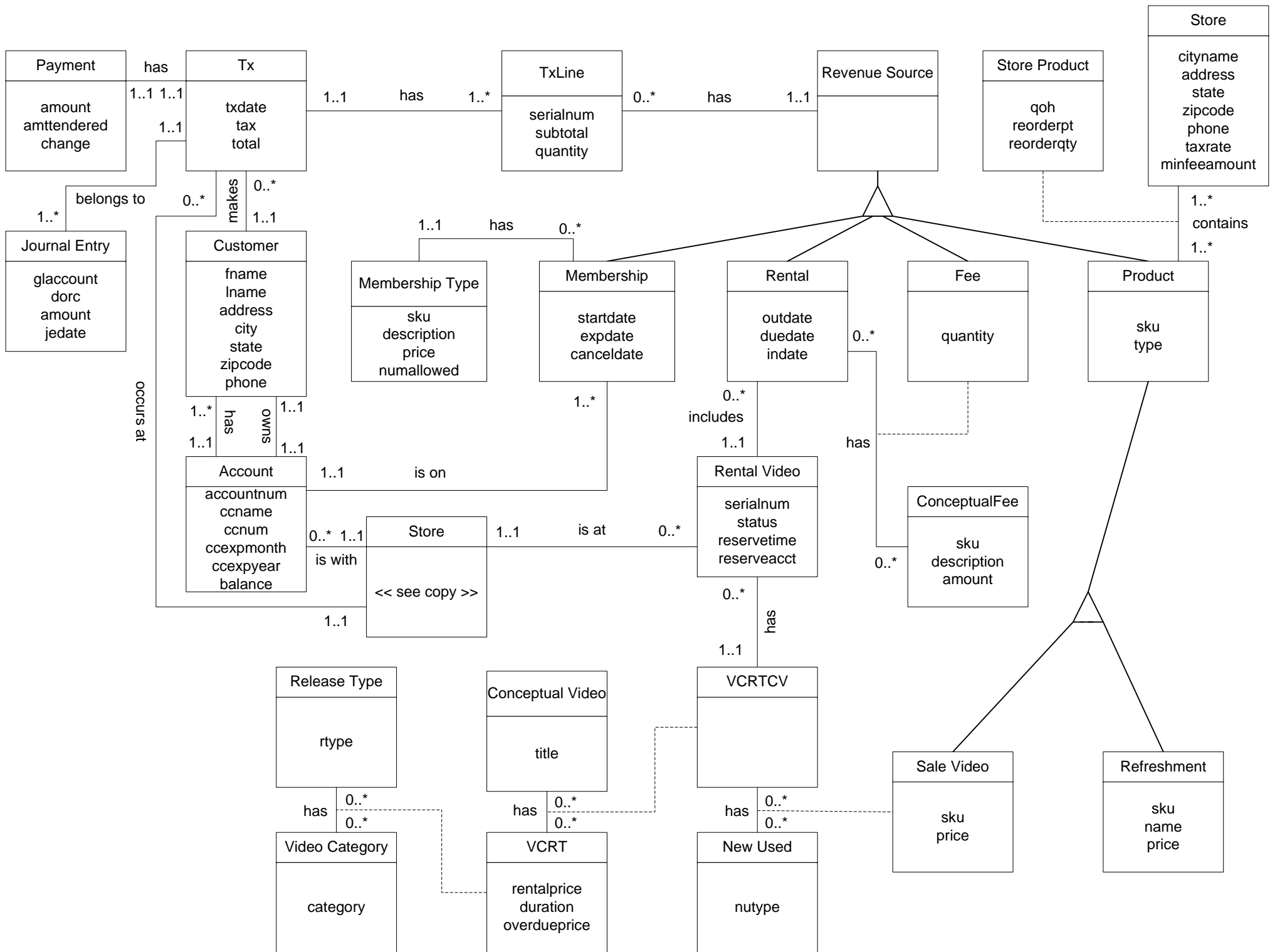
Product. A product is a physical item for sale. Currently, this includes sale videos and refreshments. The product table includes SKU's and types. These are mainly used for determining if an item is a sale video or refreshment based on its SKU. A product is also matched with a store and details of such a combination are stored in the store product table.

Store Product. Store product records contain information about a specific product line in a specific store. This includes quantity on hand, reorder point, and reorder quantity.

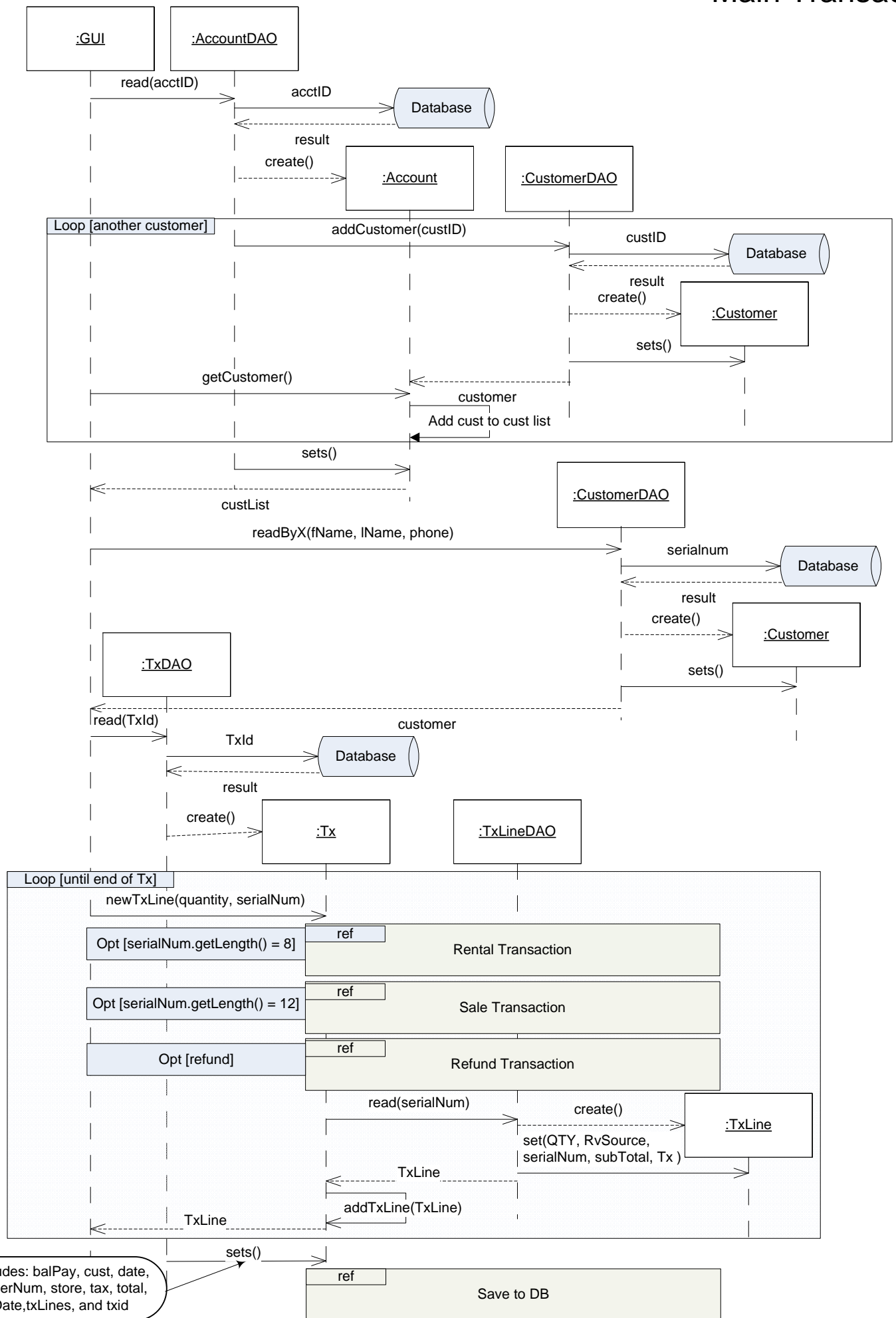
Sale Video. A sale video is a combination of a VCRTCVC and a new used record. These are specific, individual videos for sale. Information in the table include each video's SKU and price.

Refreshment. Refreshments are tasty treats to satisfy Blockwood Video customers. Each record contains a refreshment sku, name, and price. These records do not refer

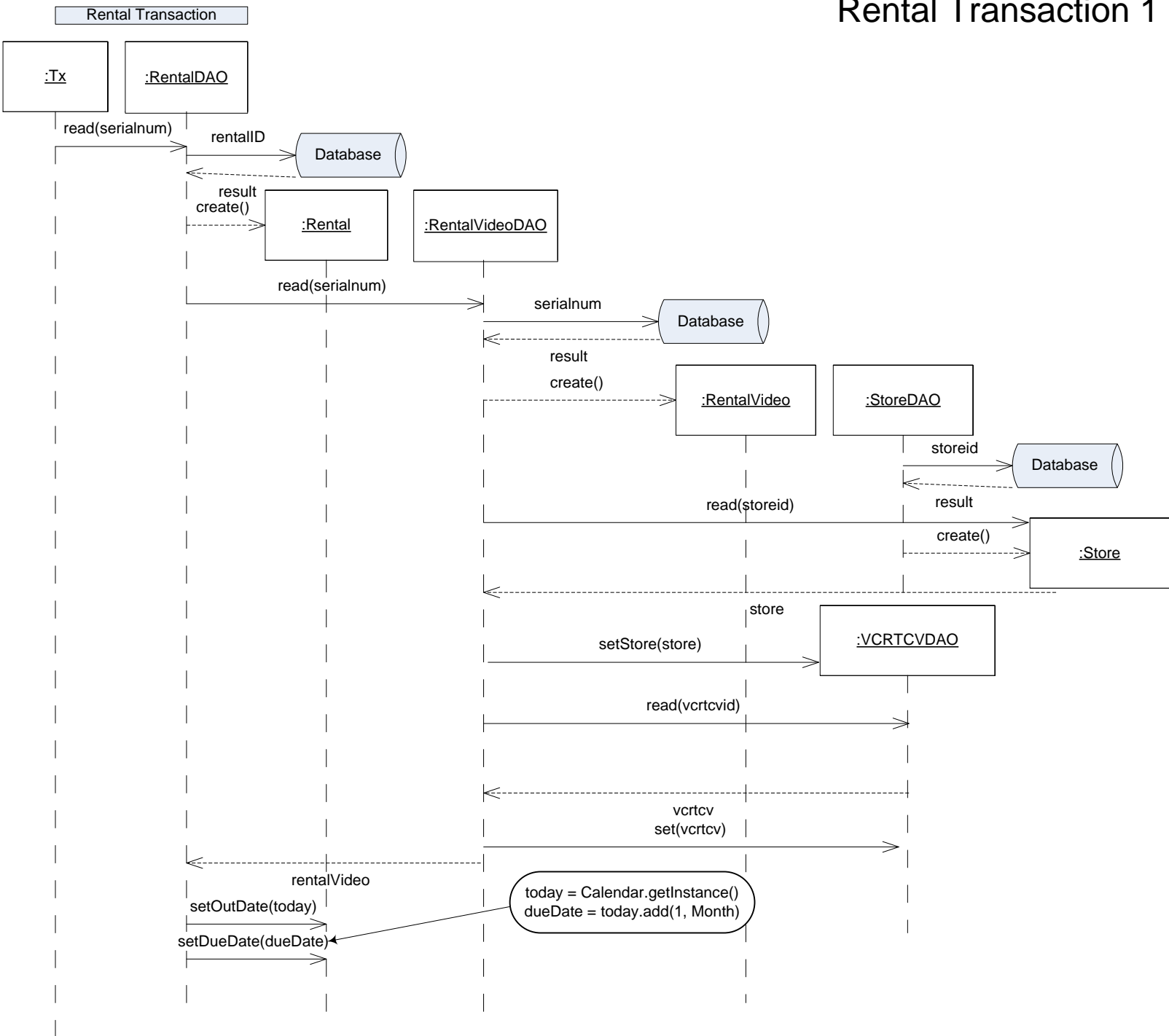
to single, individual refreshments but a line of refreshments such as Snickers candy bars.



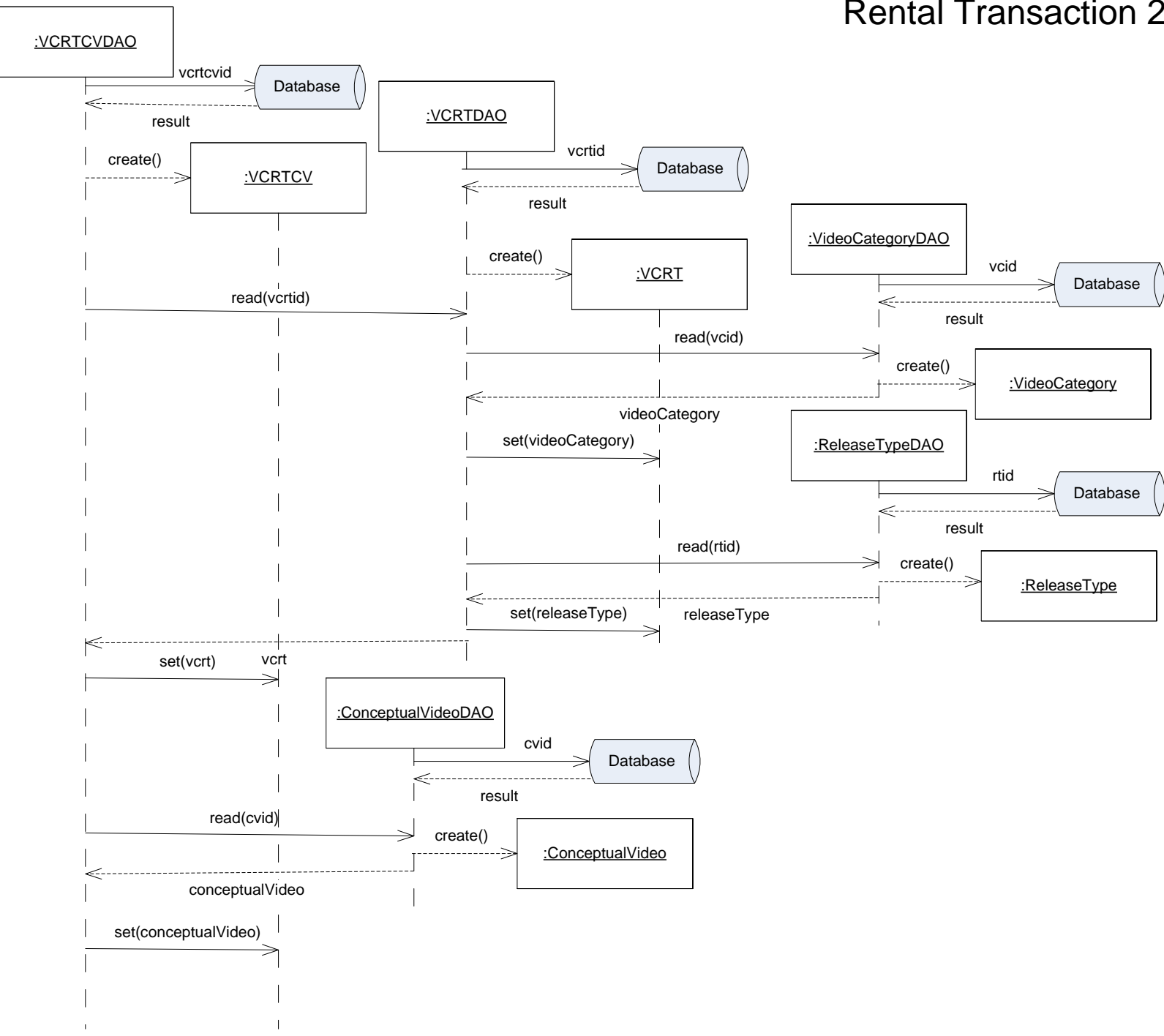
Main Transaction



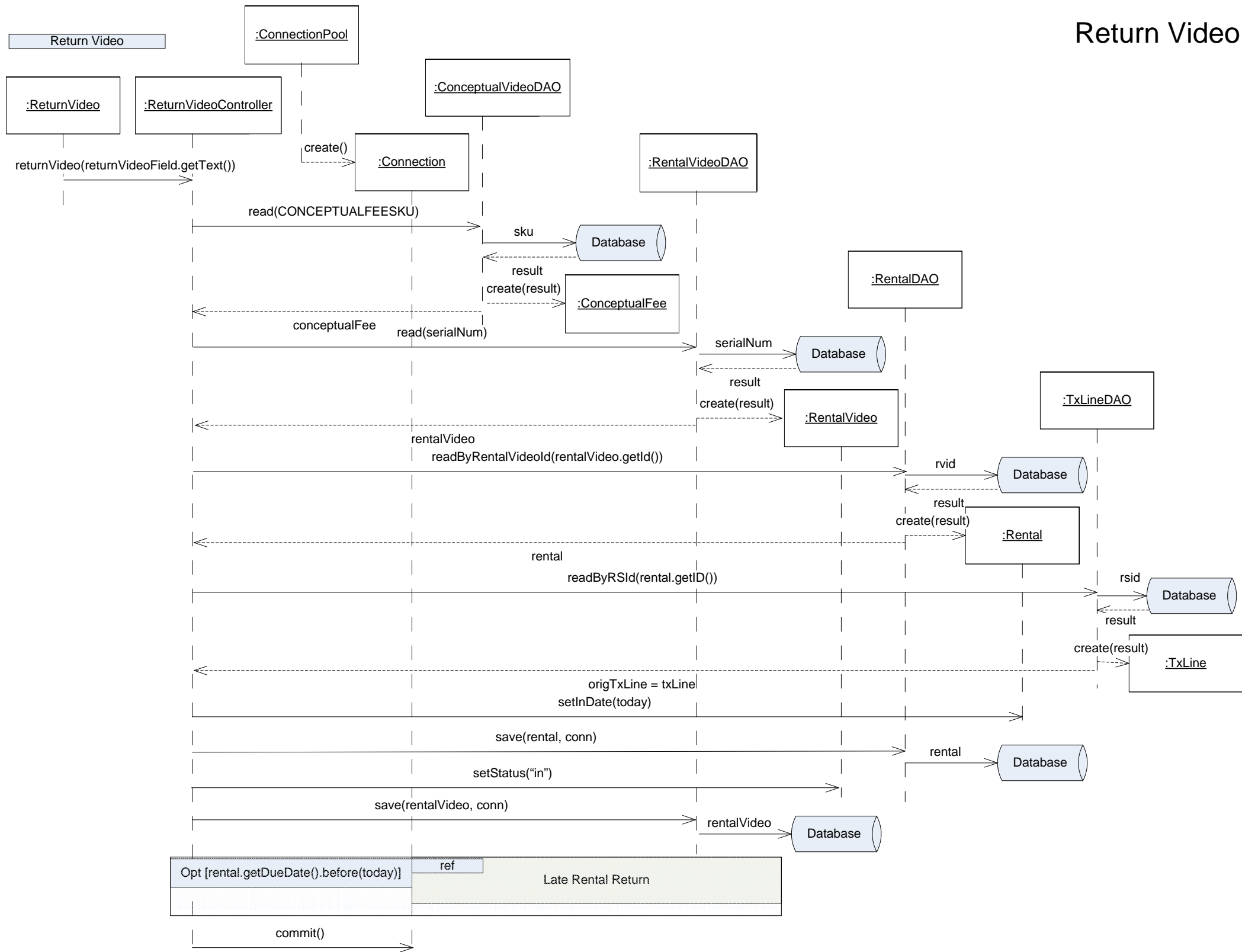
Rental Transaction 1

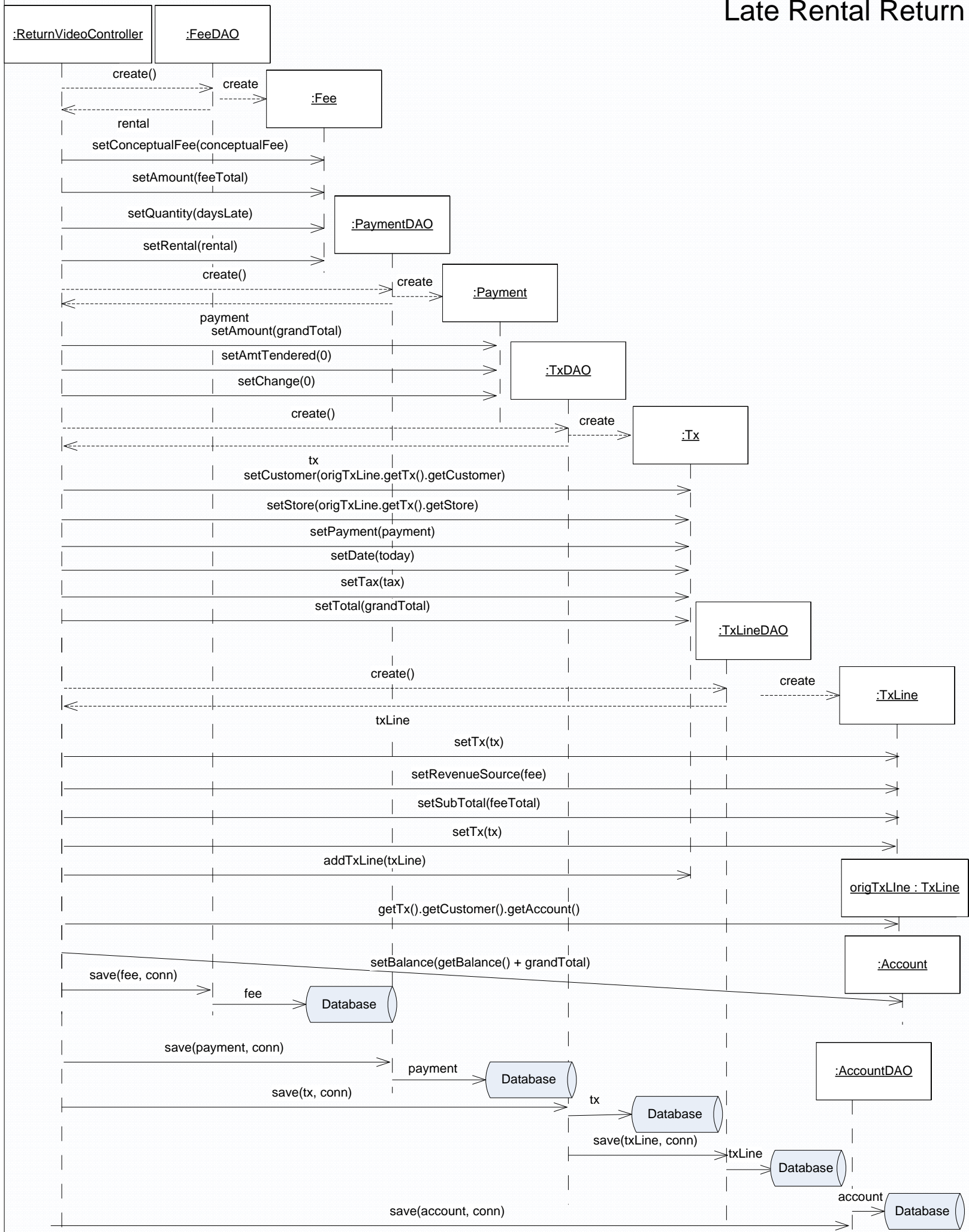


Rental Transaction 2

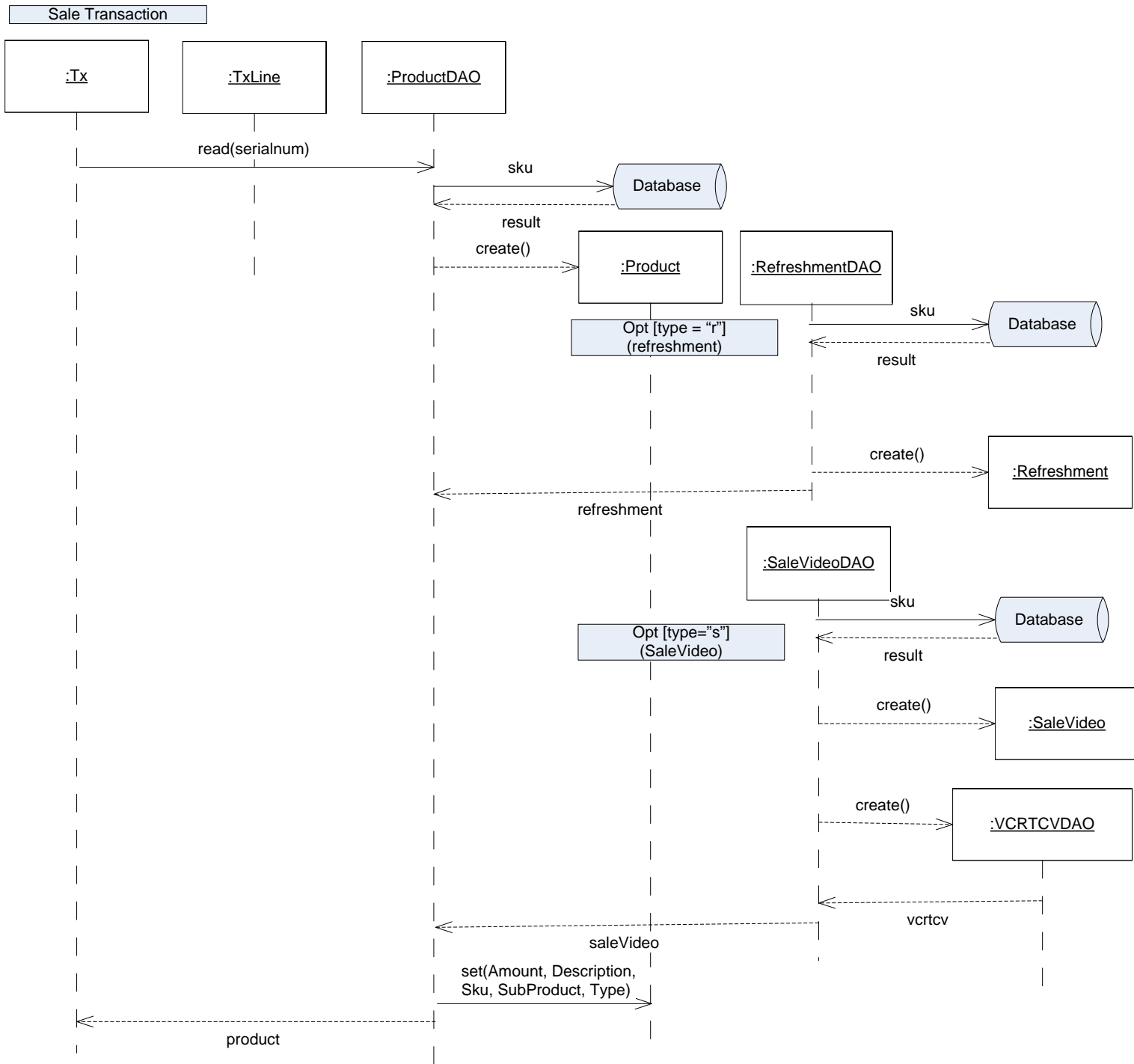


Return Video

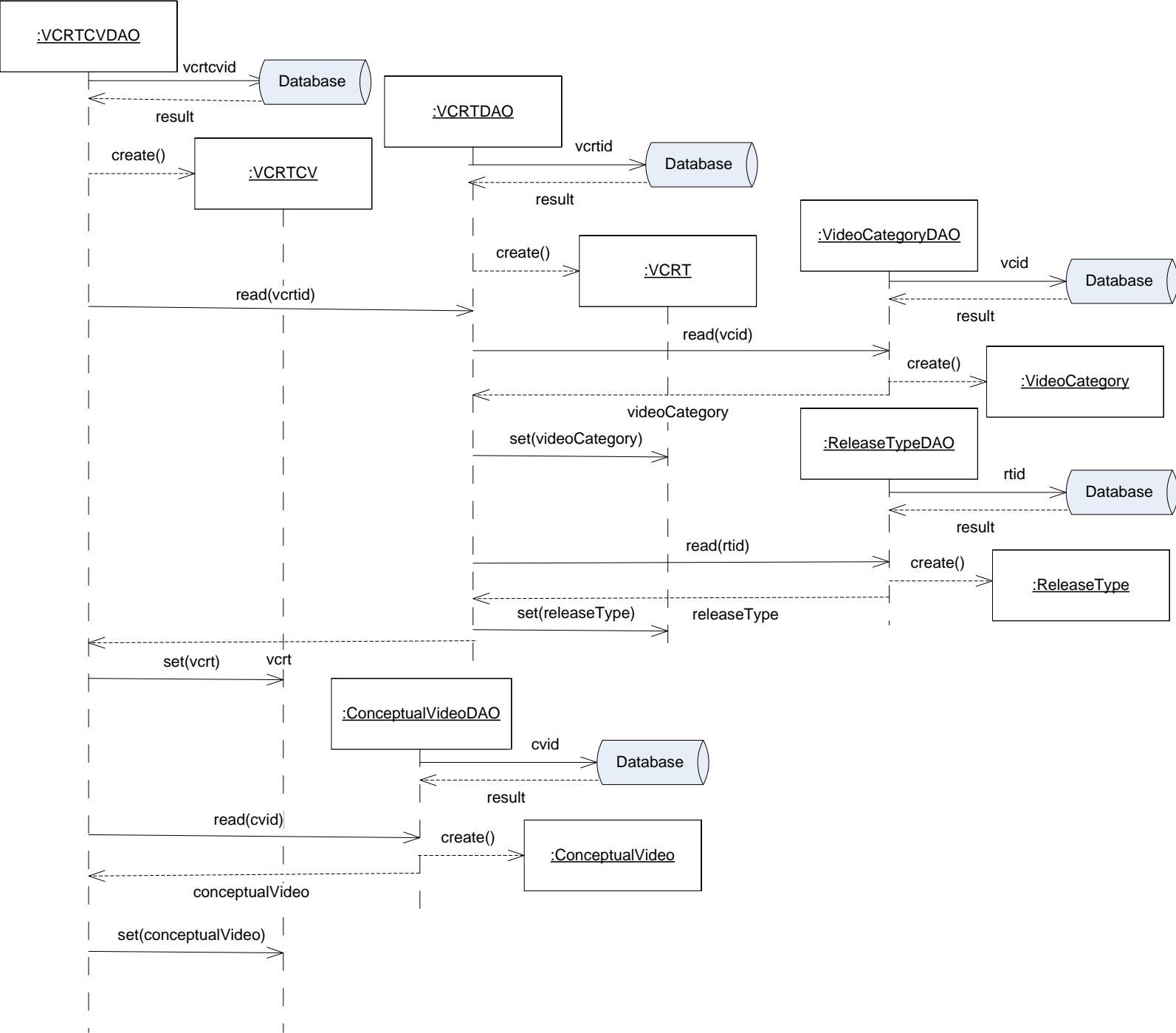


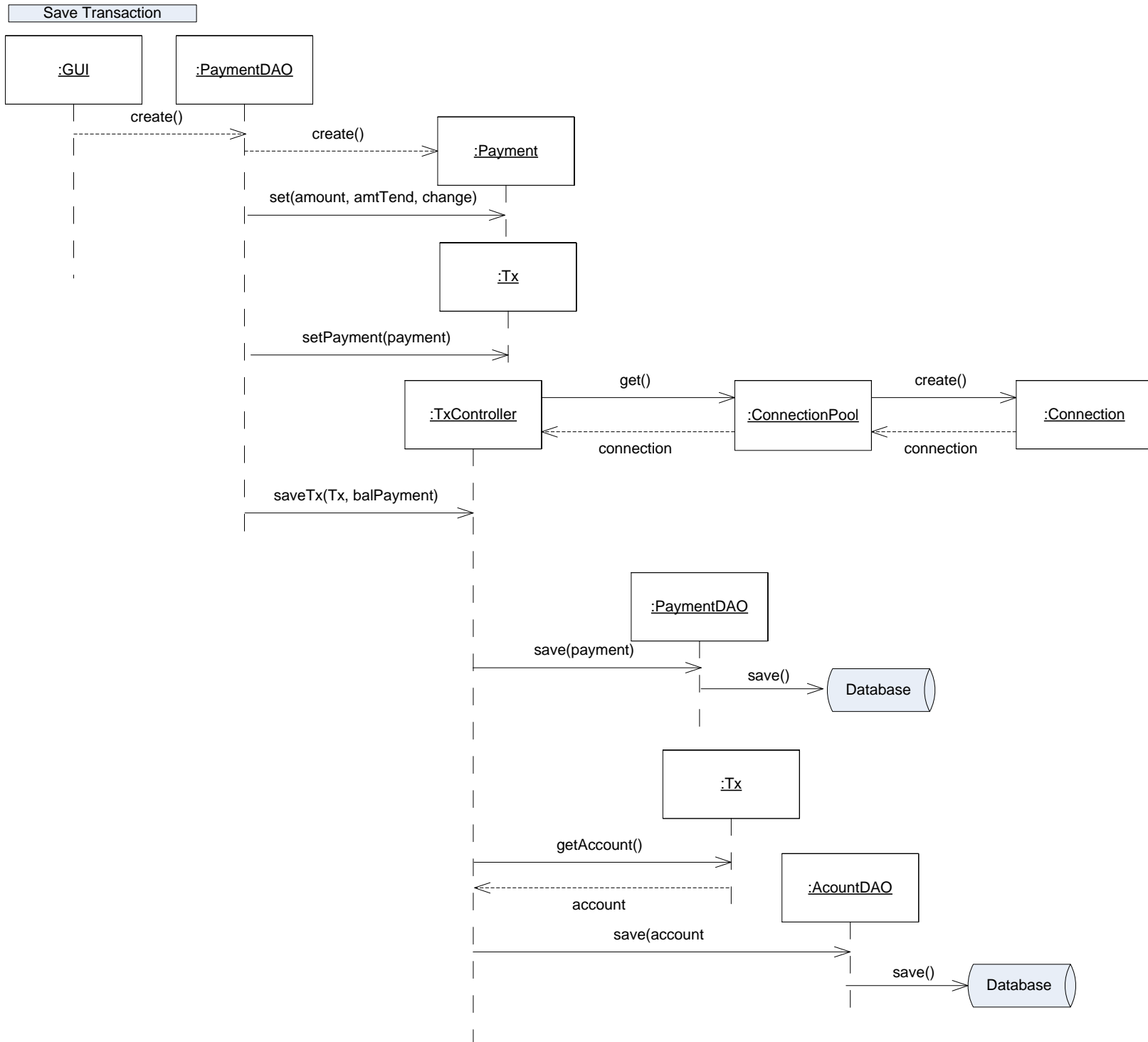


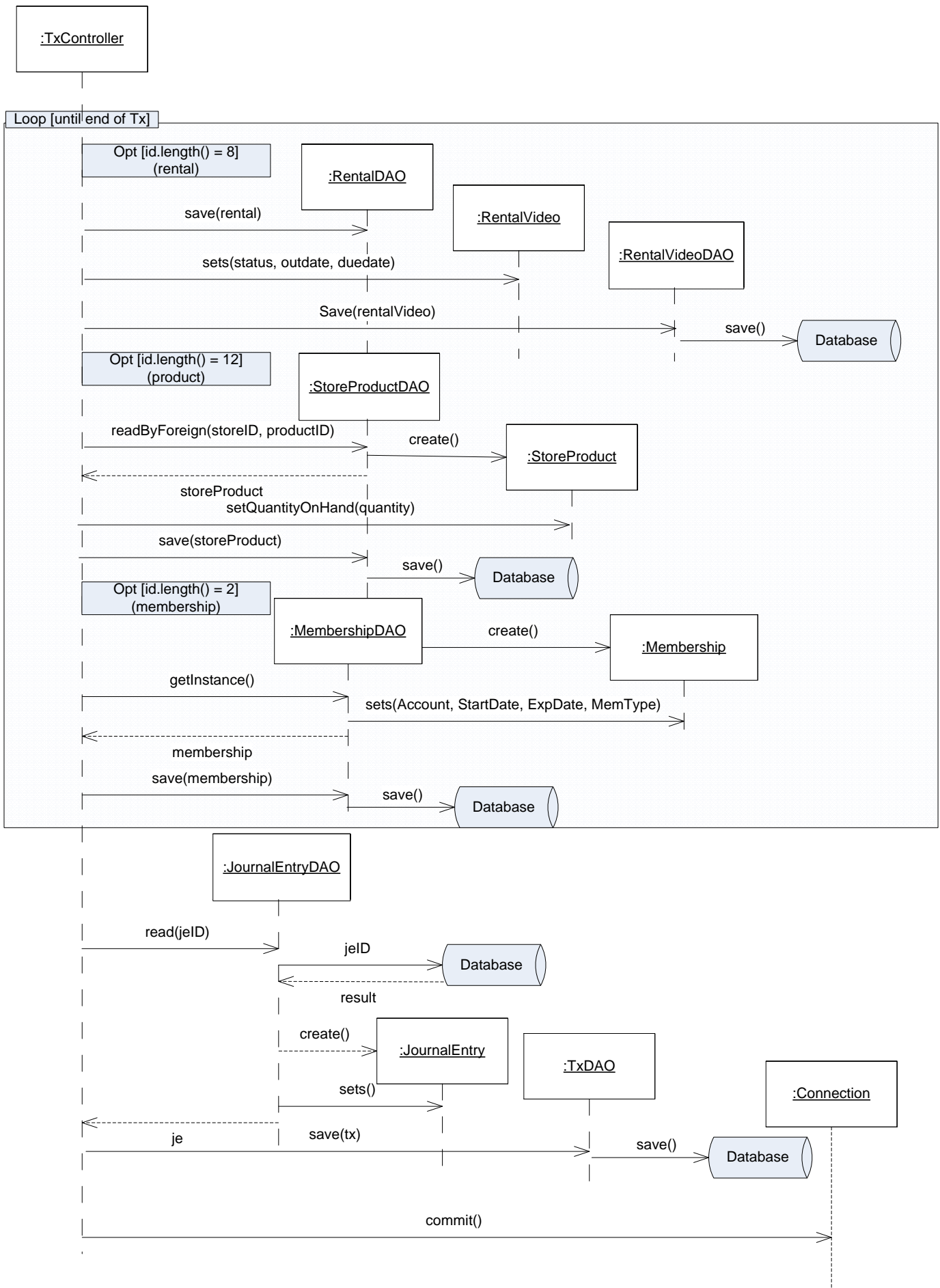
Sale Transaction 1



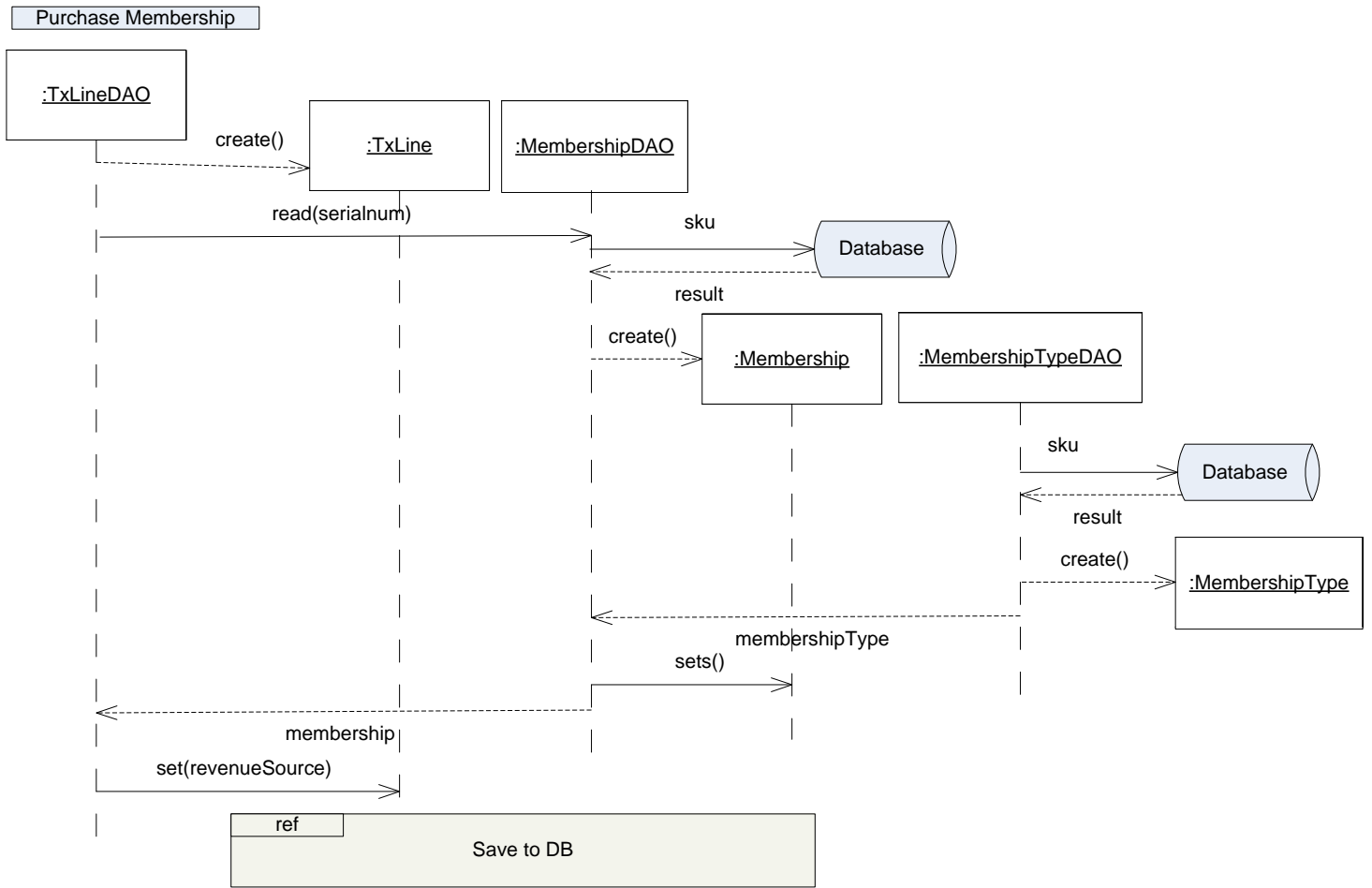
Sale Transaction 2



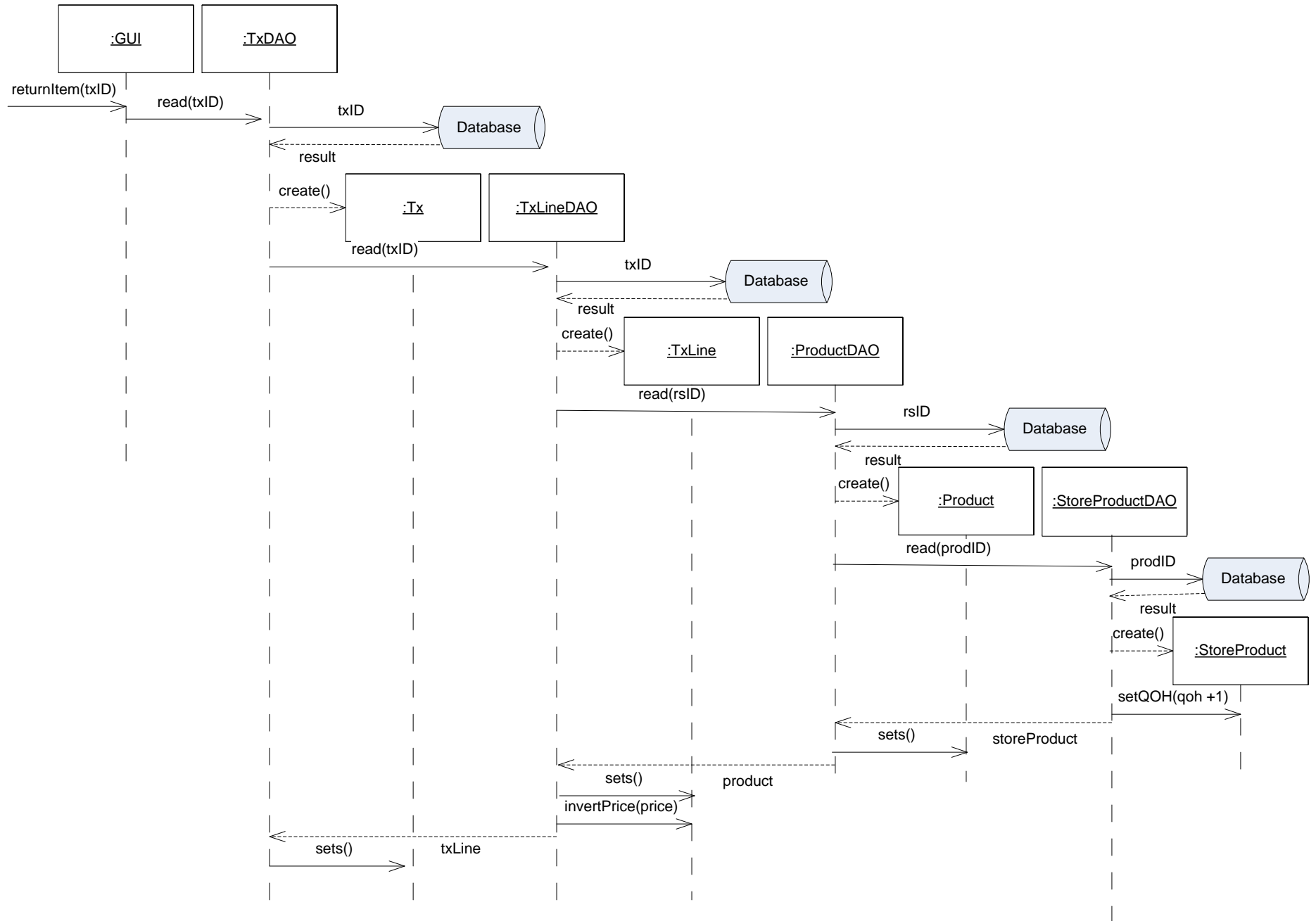




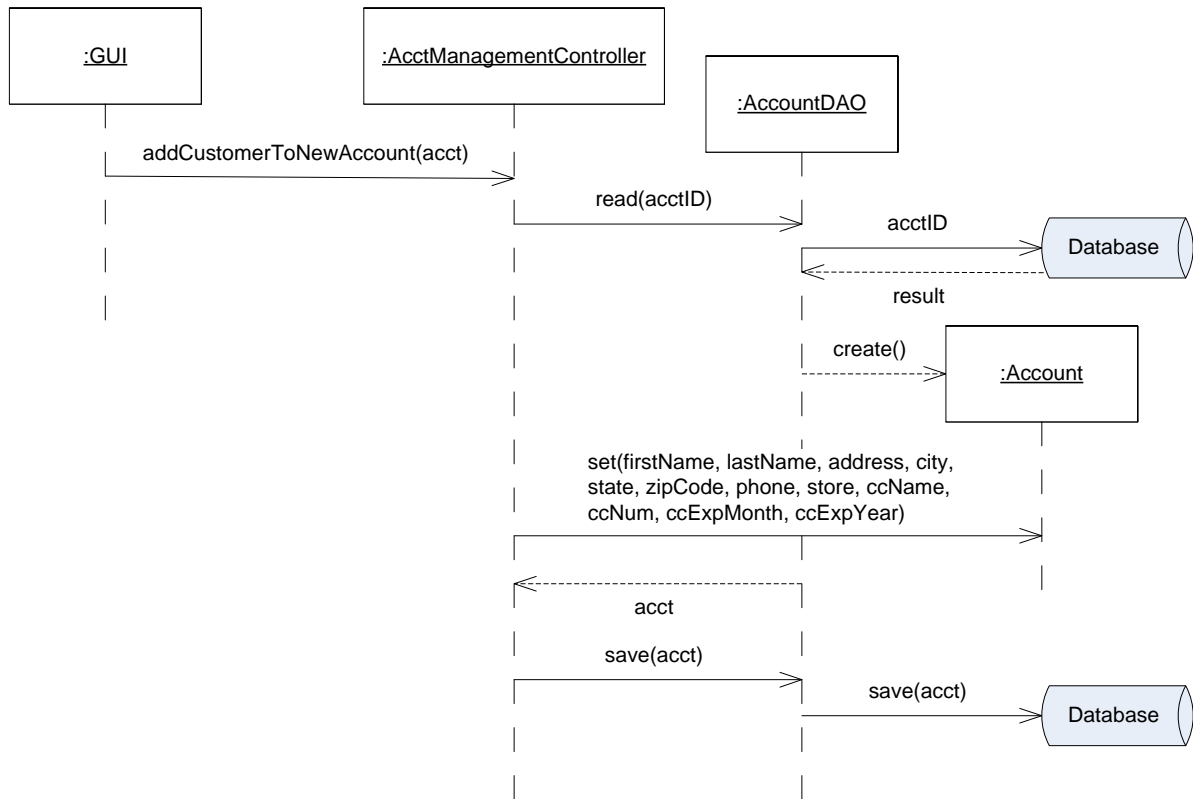
Purchase Membership



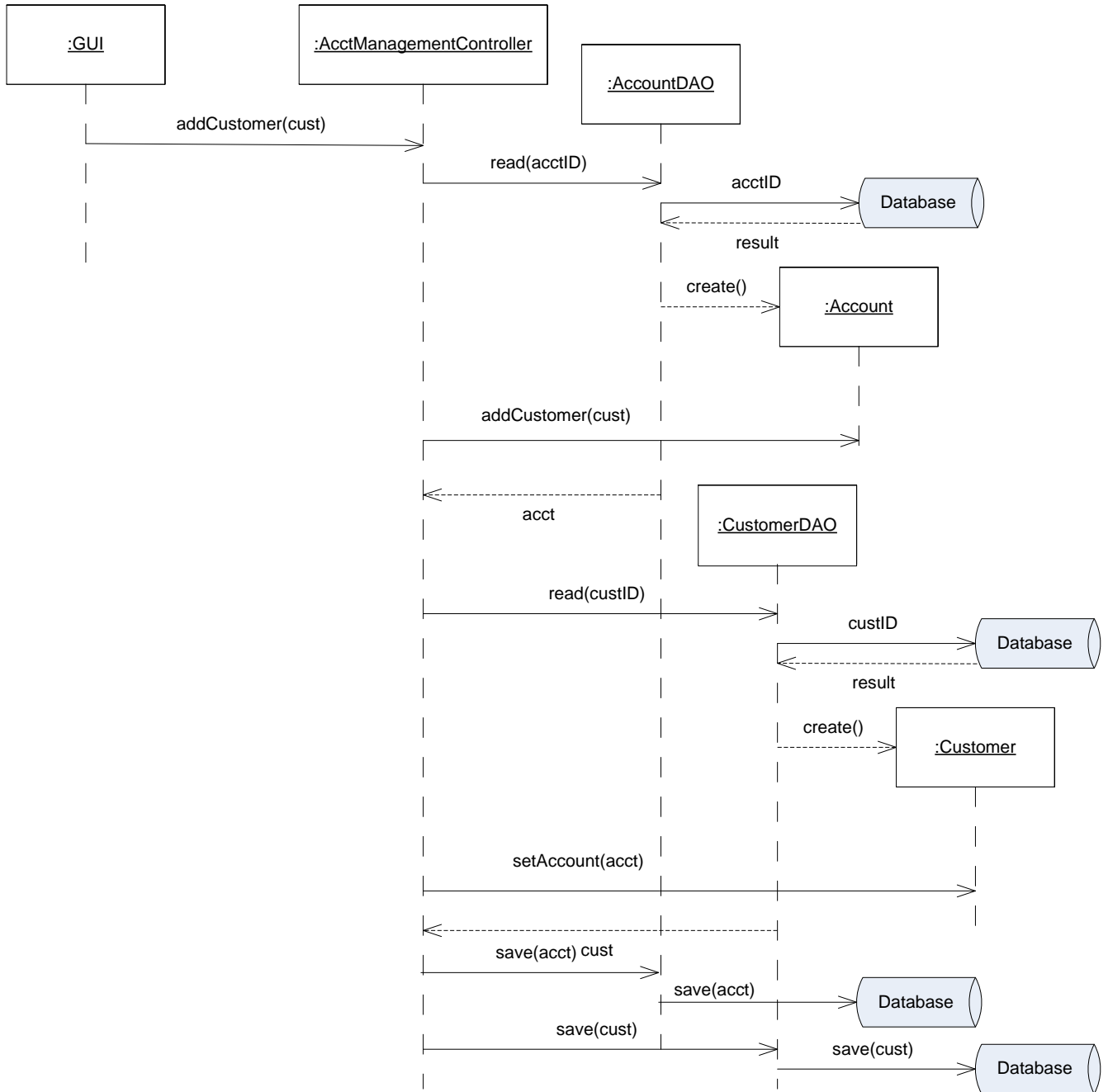
Refund Transaction



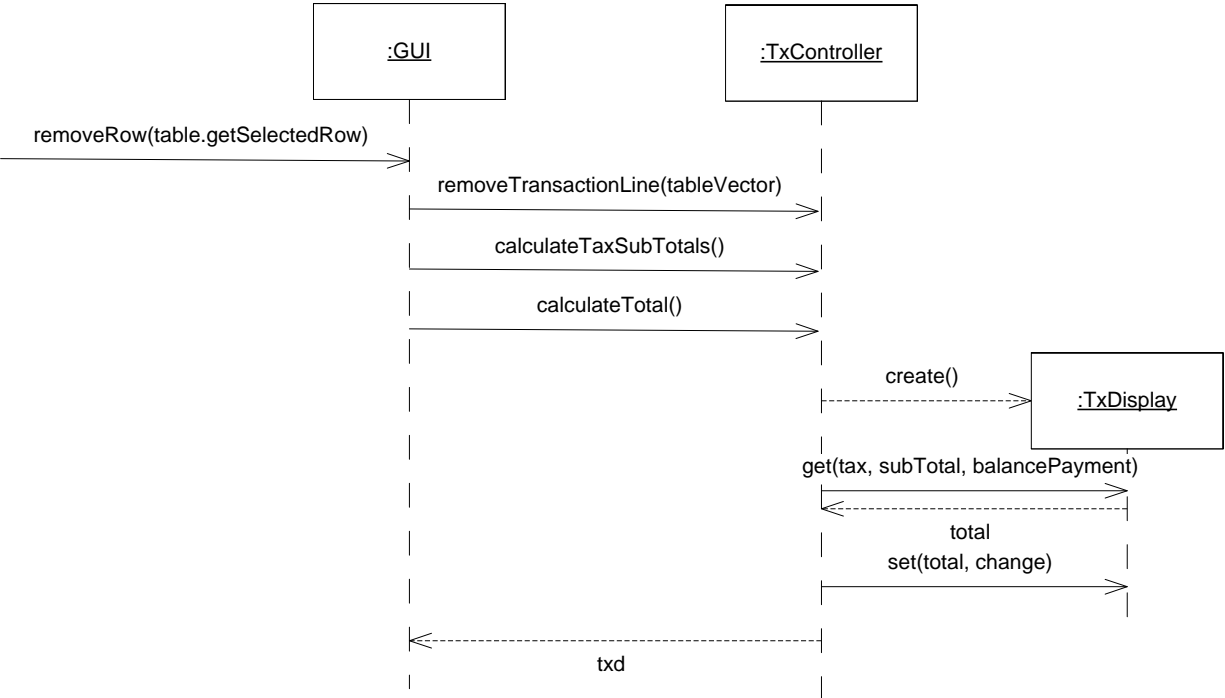
Create Account



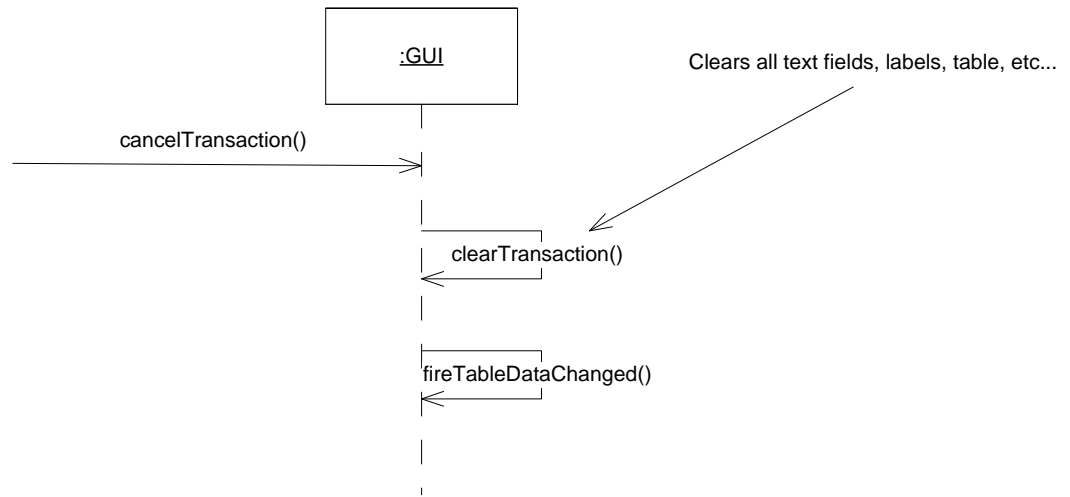
Add Customer to Account

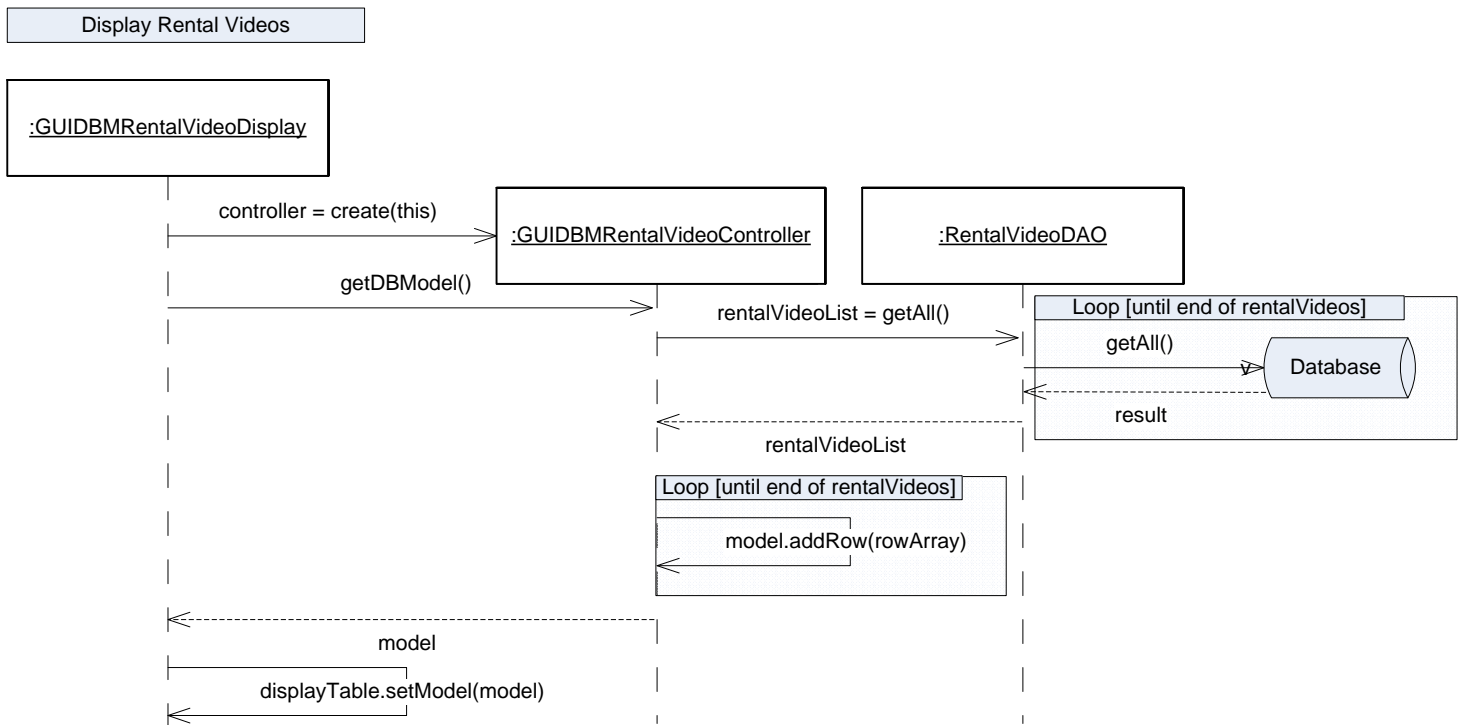


Delete Transaction Line



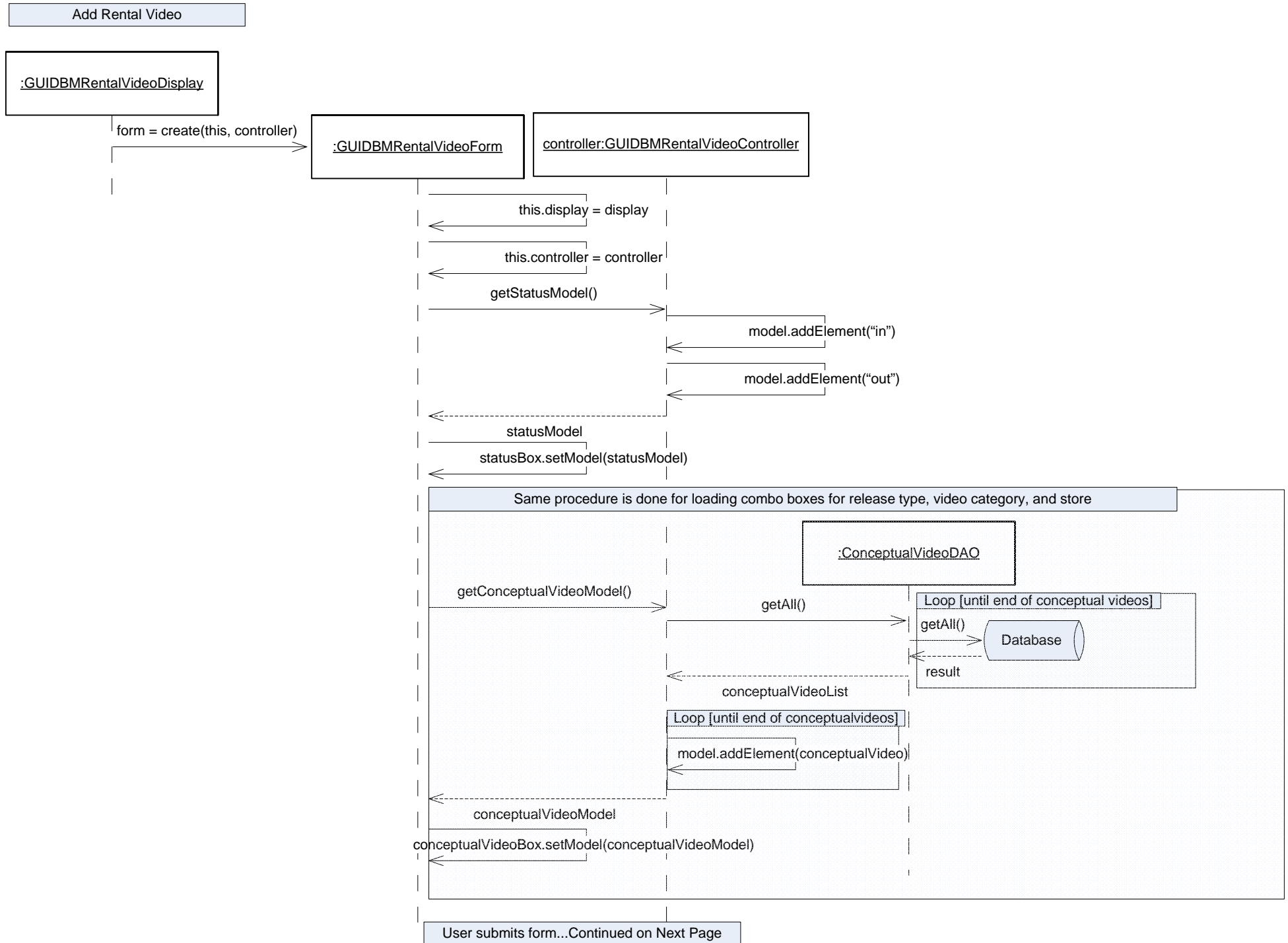
Cancel Transaction





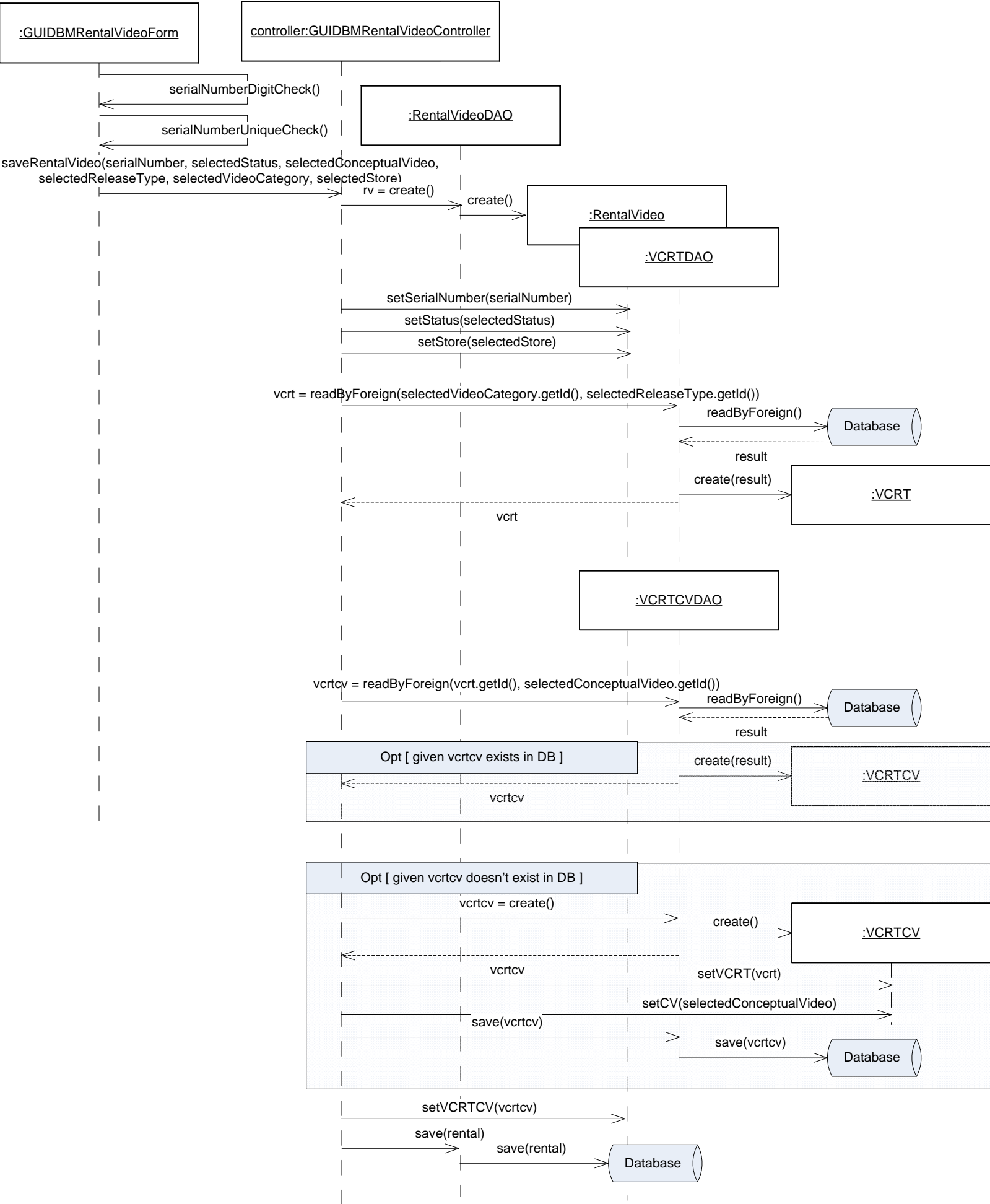
- identical for all tables -

Add Rental Video



Add Rental Video 2

User submits form...



Modify Rental Video

Modify Rental Video

:GUIDBMRentalVideoDisplay

form = create(this, controller, rowNum)

:GUIDBMRentalVideoForm

controller:GUIDBMRentalVideoController

rentalVideo = getRentalVideoList().get(rowNumber)

rentalVideo

this.display = display

this.controller = controller

getStatusModel()

model.addElement("in")

model.addElement("out")

statusModel

statusBox.setModel(statusModel)

Same procedure is done for loading combo boxes for release type, video category, and store

:ConceptualVideoDAO

getConceptualVideoModel()

getAll()

Loop [until end of conceptual videos]

getAll()

Database

result

conceptualVideoList

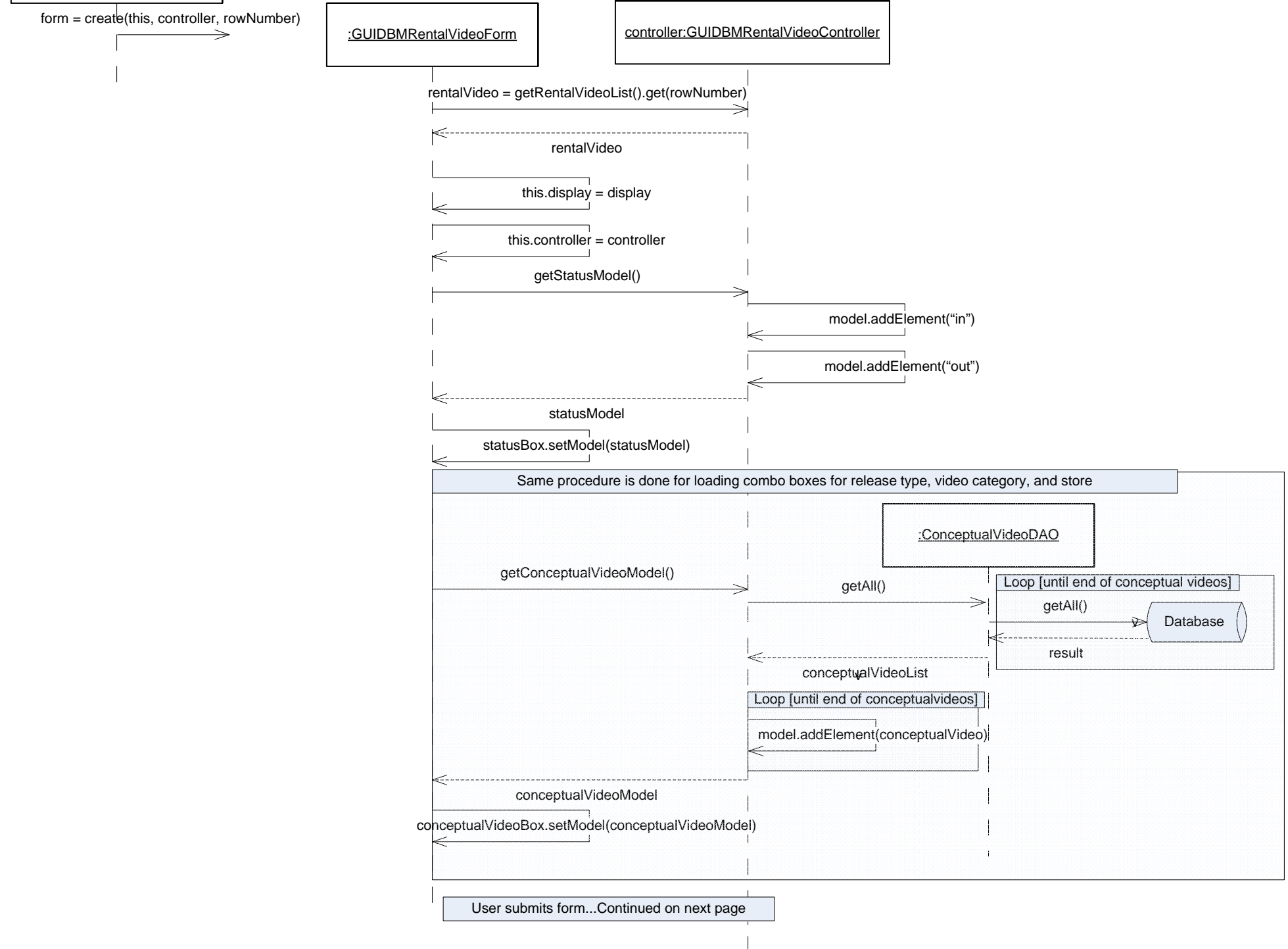
Loop [until end of conceptual videos]

model.addElement(conceptualVideo)

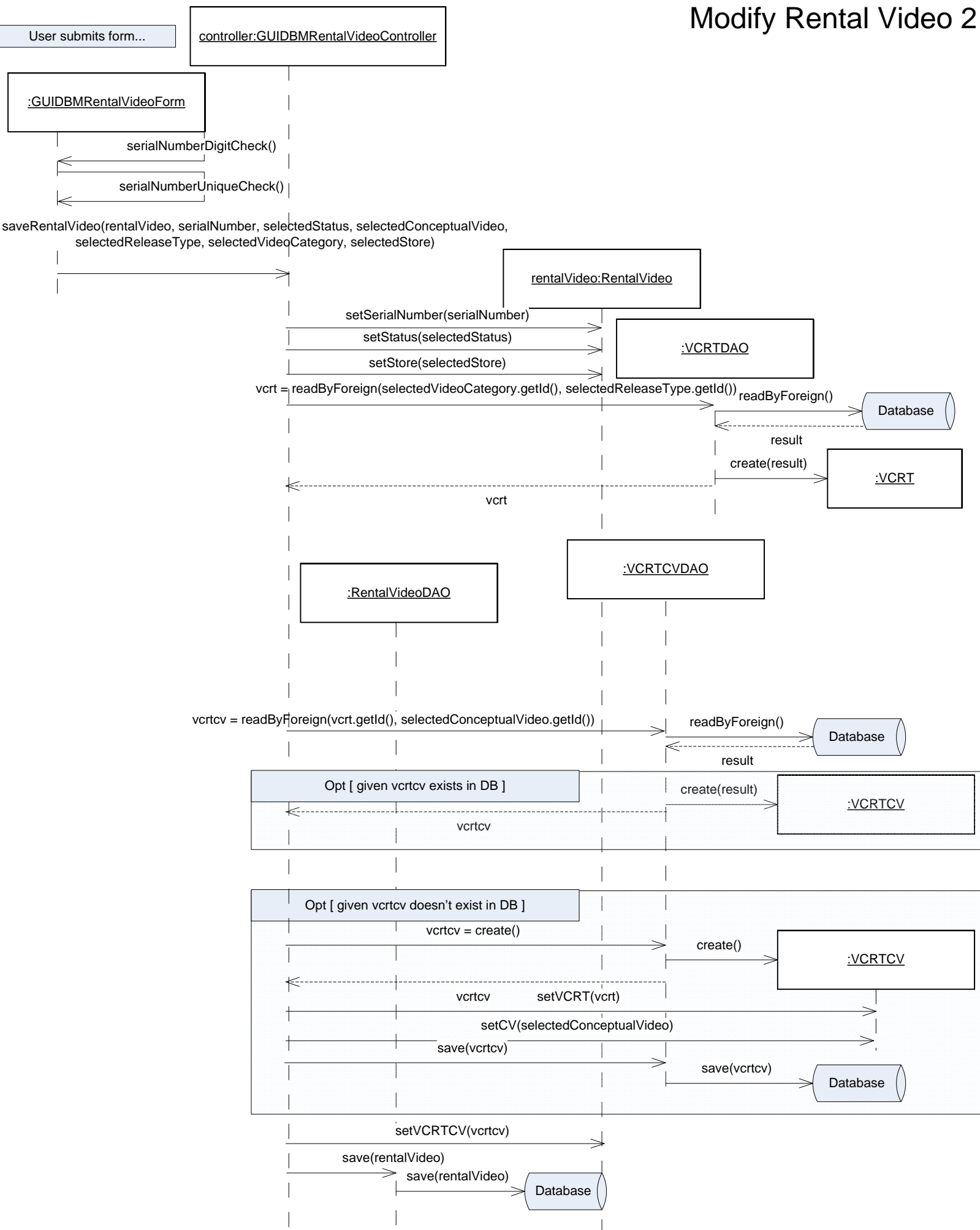
conceptualVideoModel

conceptualVideoBox.setModel(conceptualVideoModel)

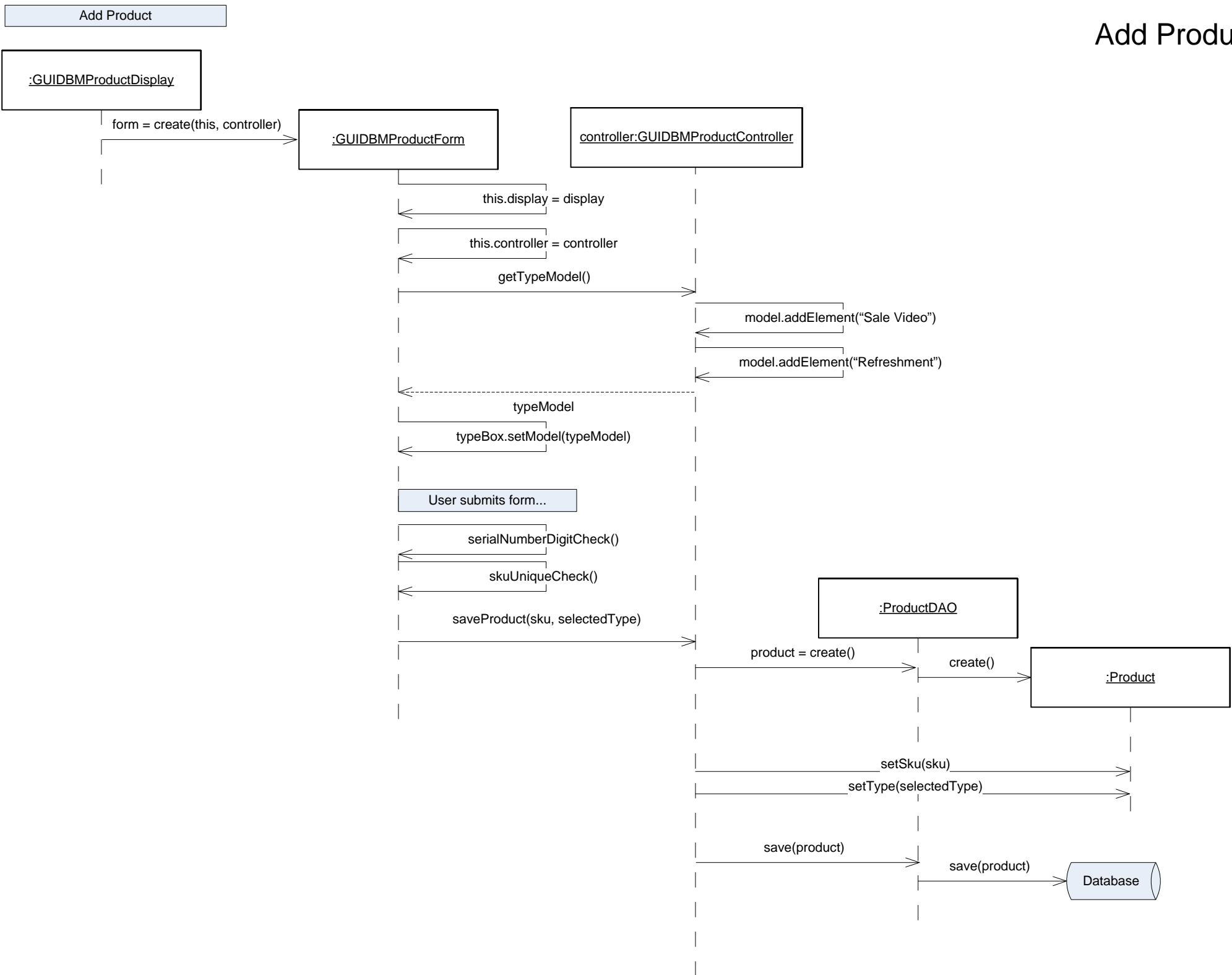
User submits form...Continued on next page



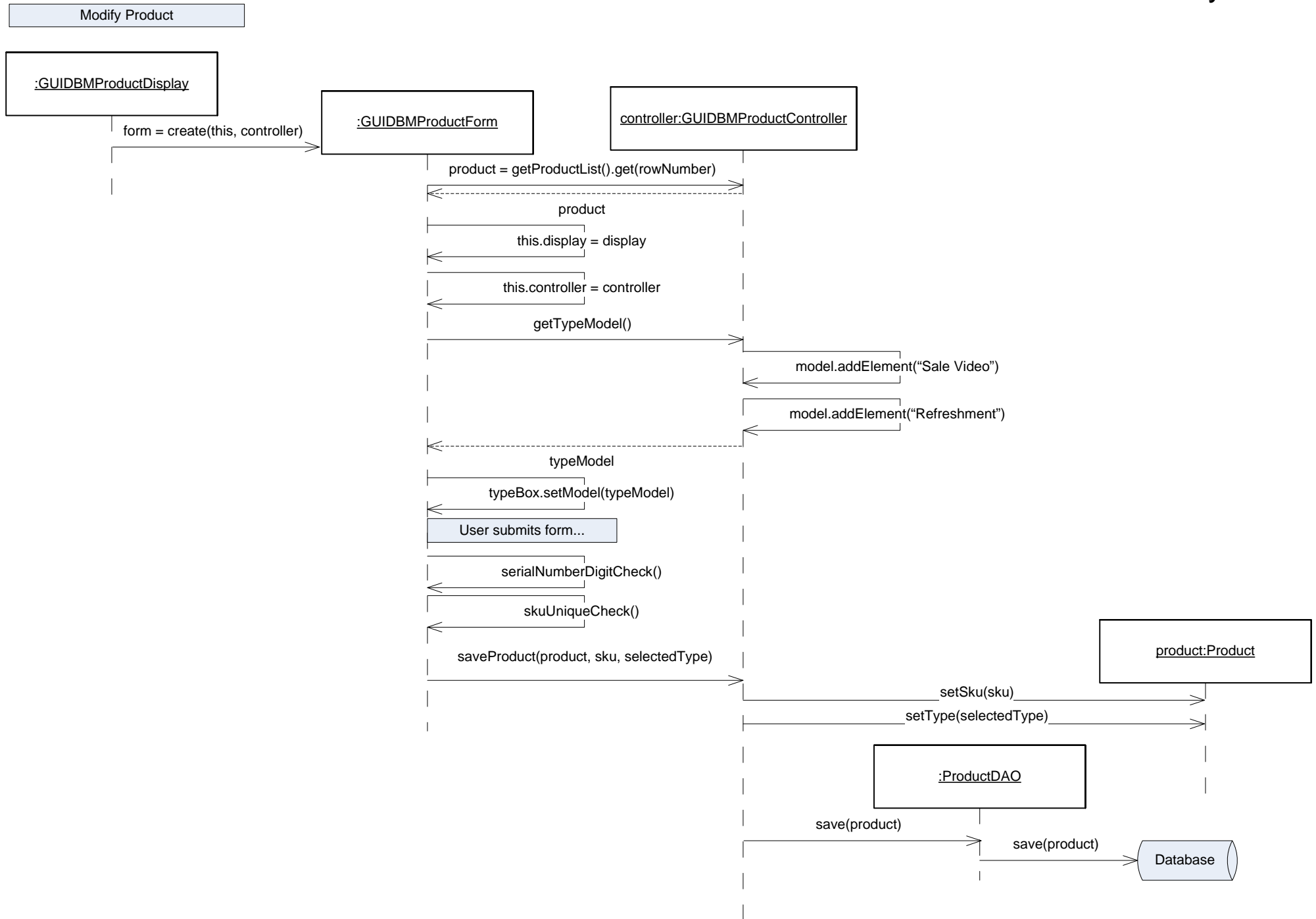
Modify Rental Video 2



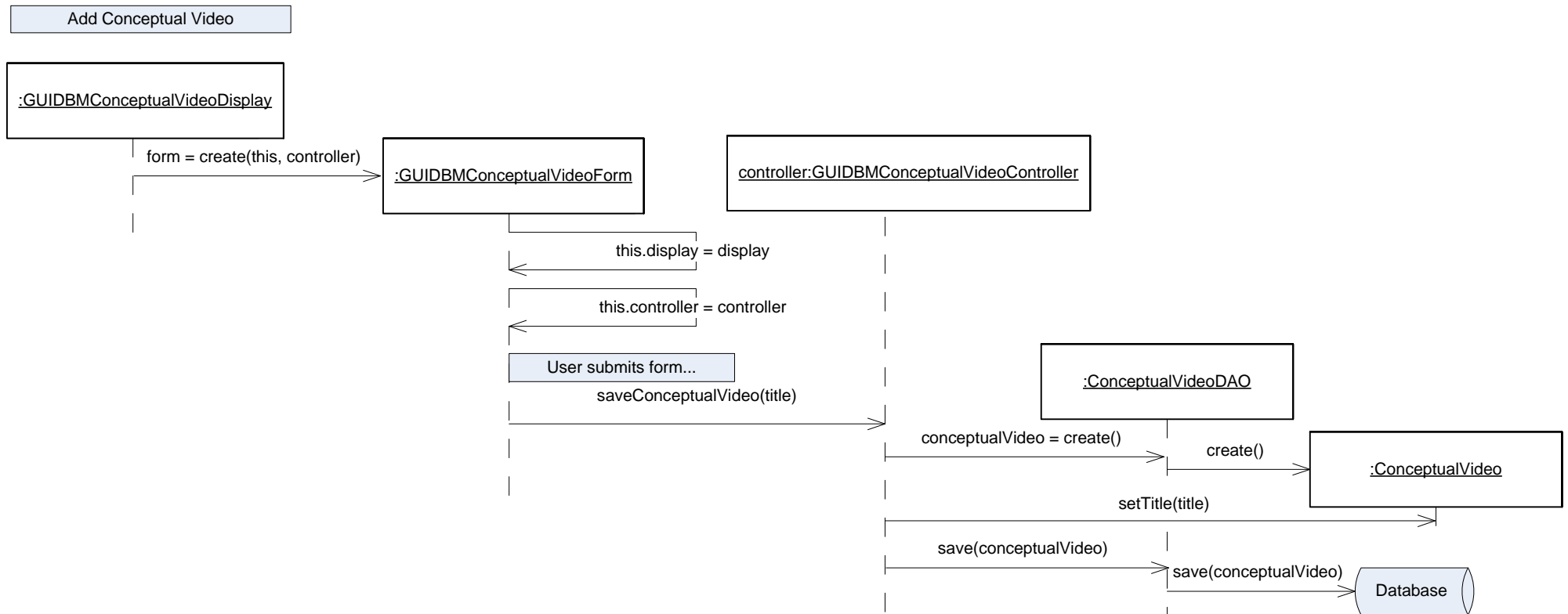
Add Product



Modify Product



Add Conceptual Video



- identical for Add Store -

Modify Conceptual Video

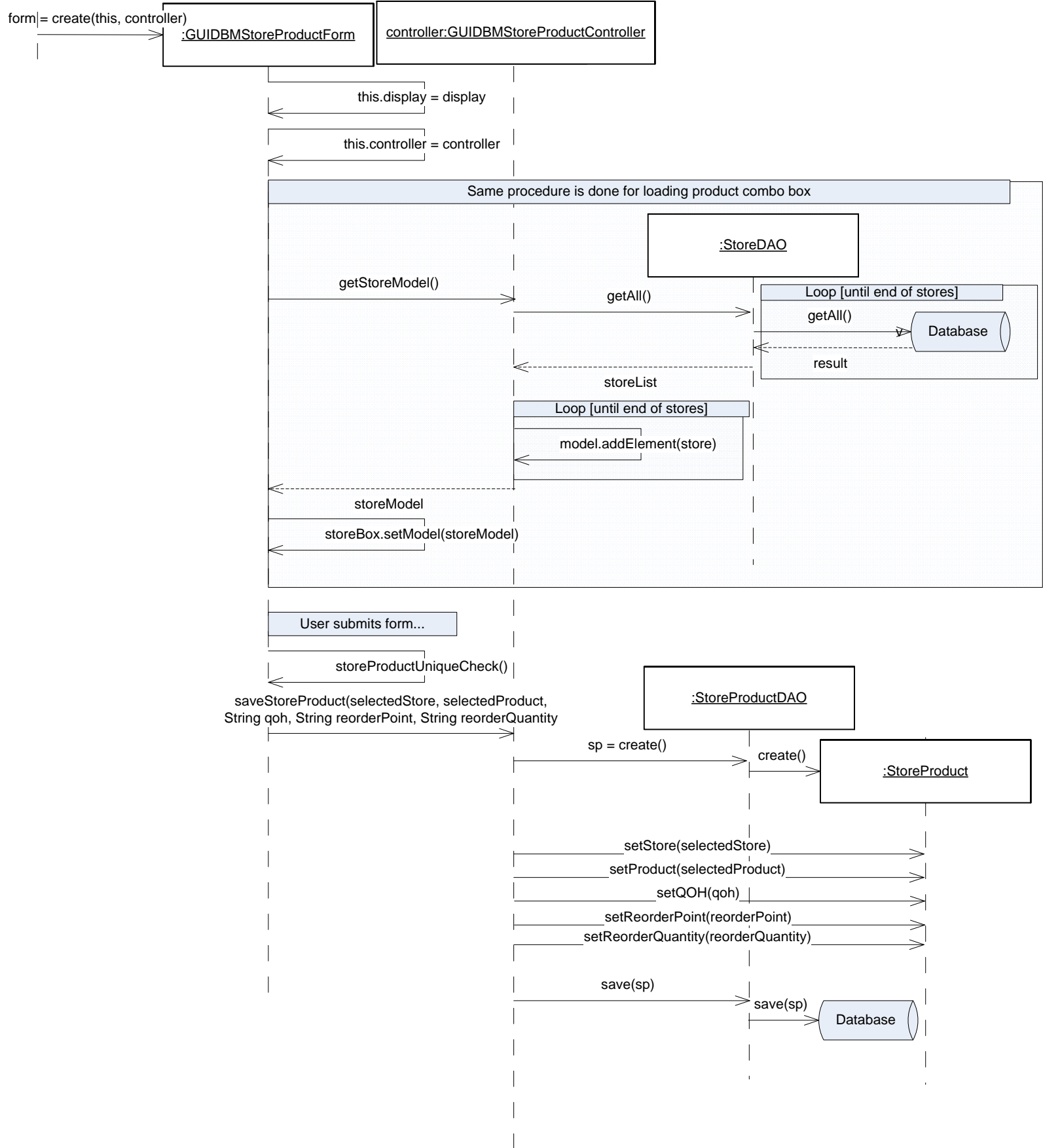


- identical for Modify Store -

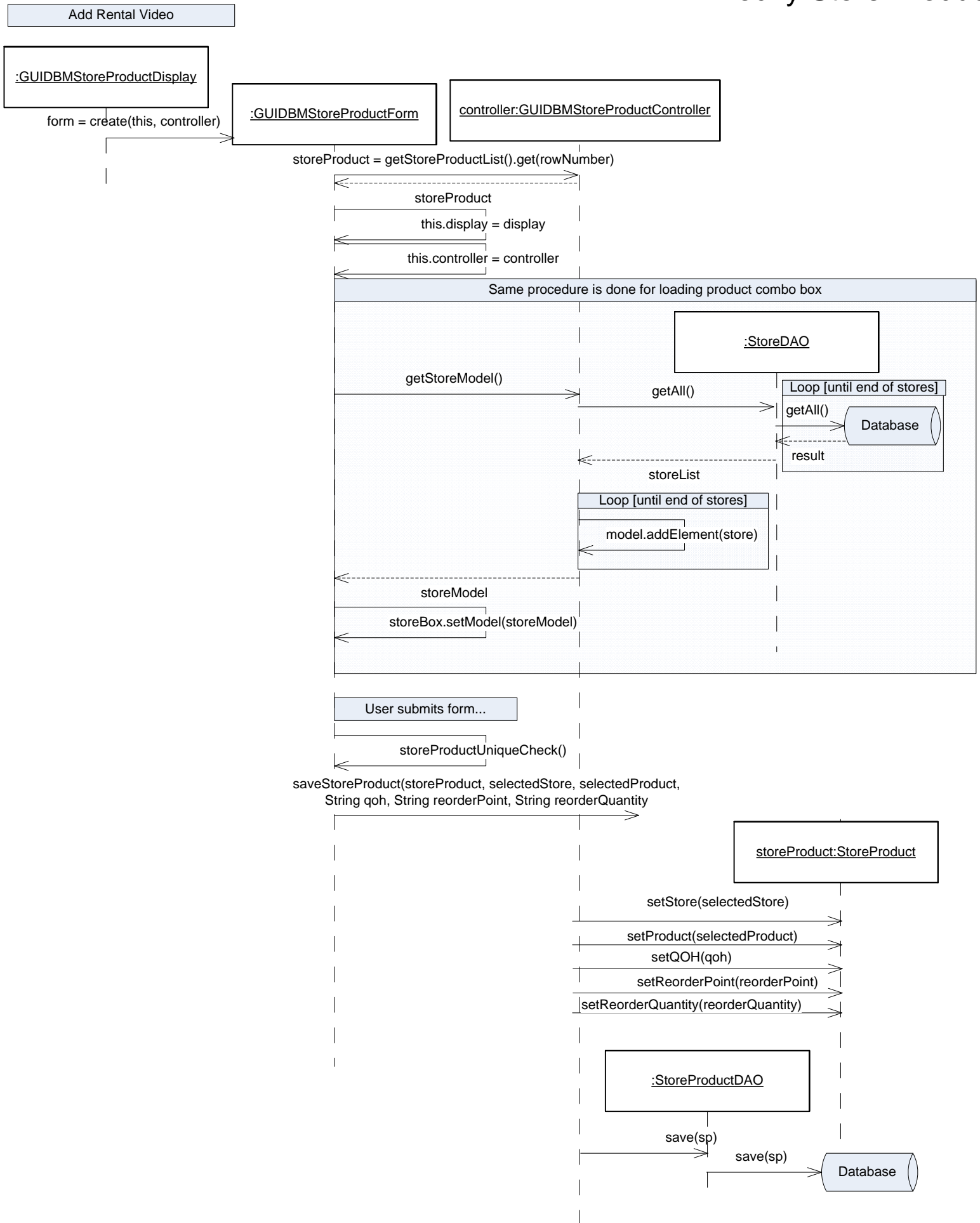
Add Store Product

Add Rental Video

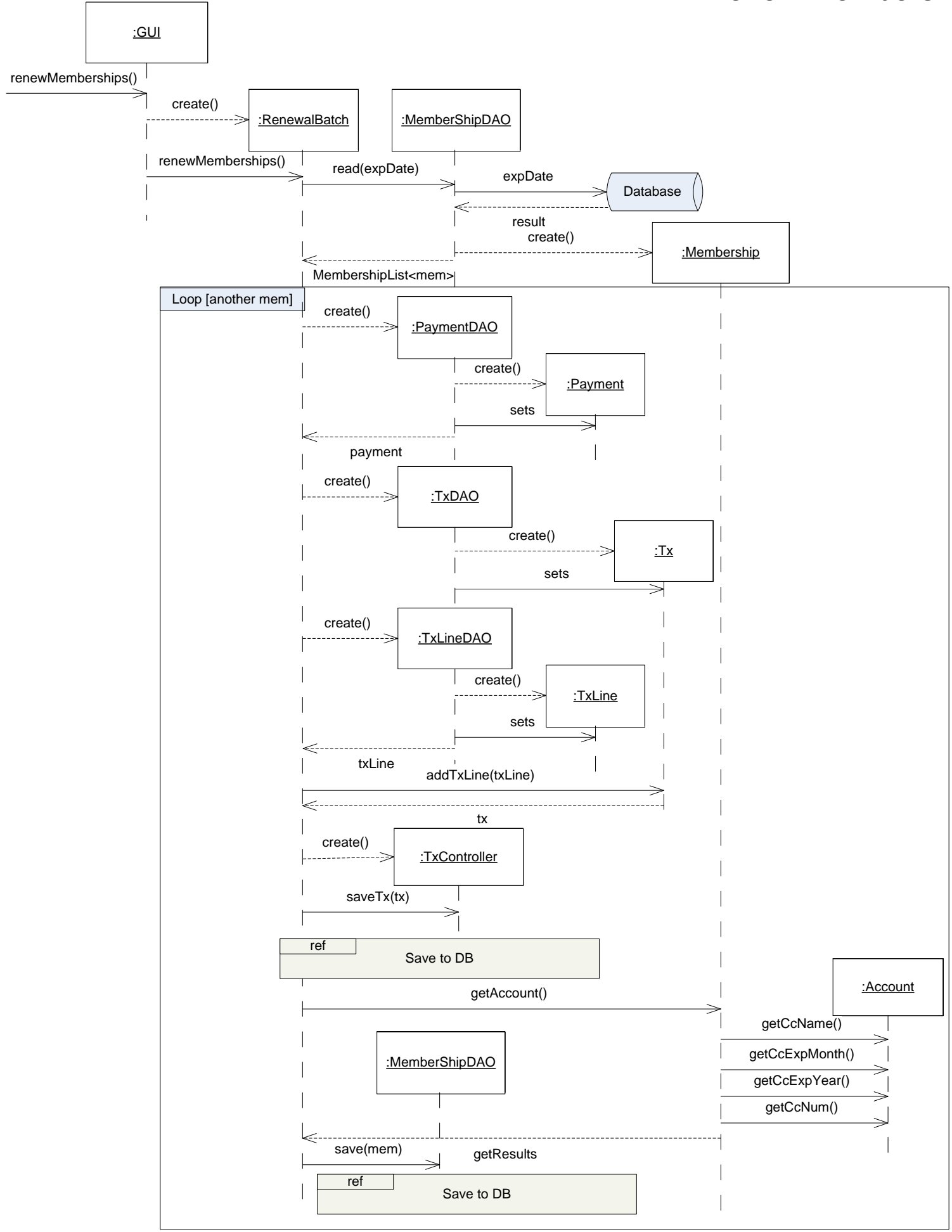
:GUIDBMStoreProductDisplay

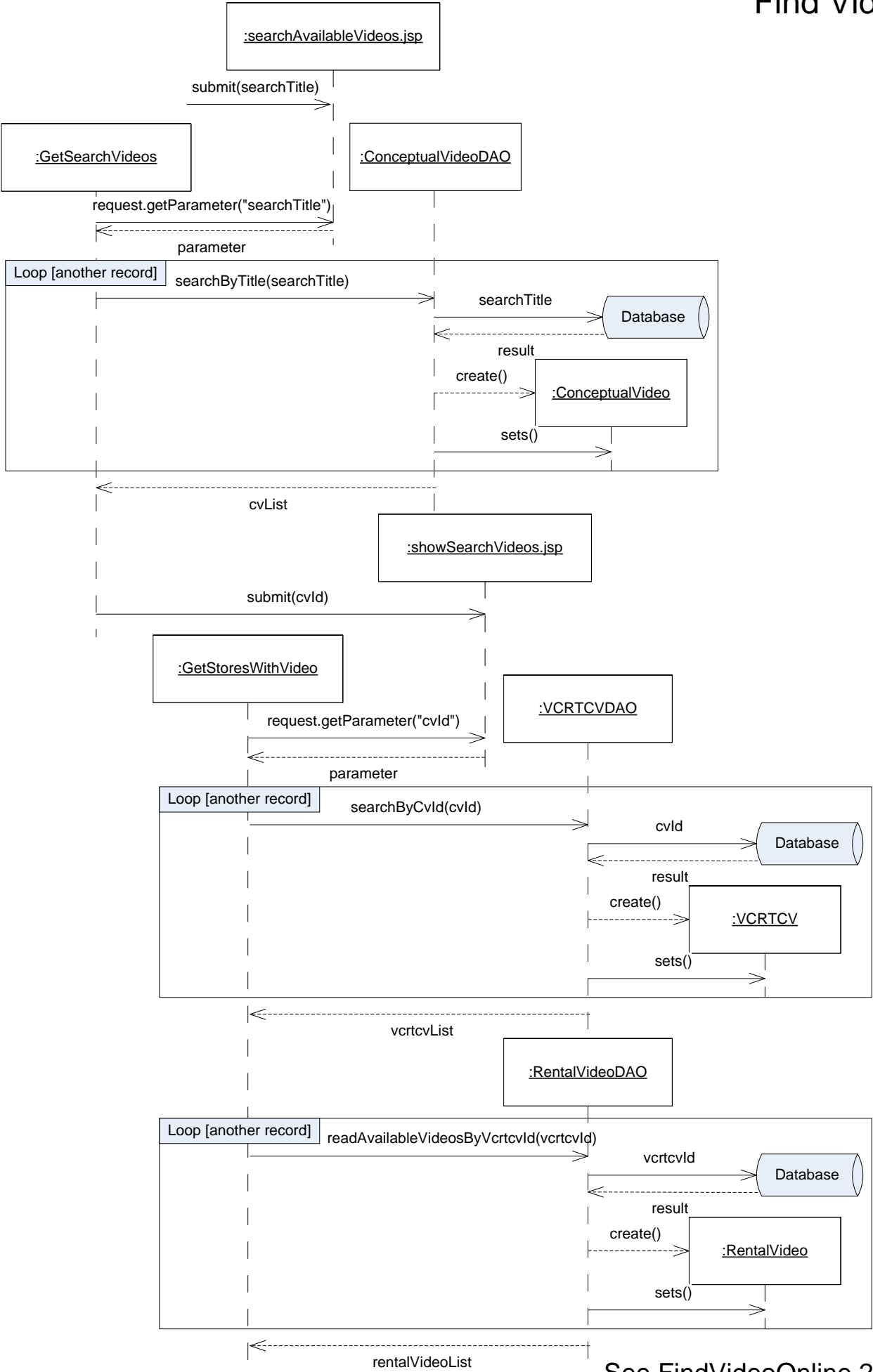


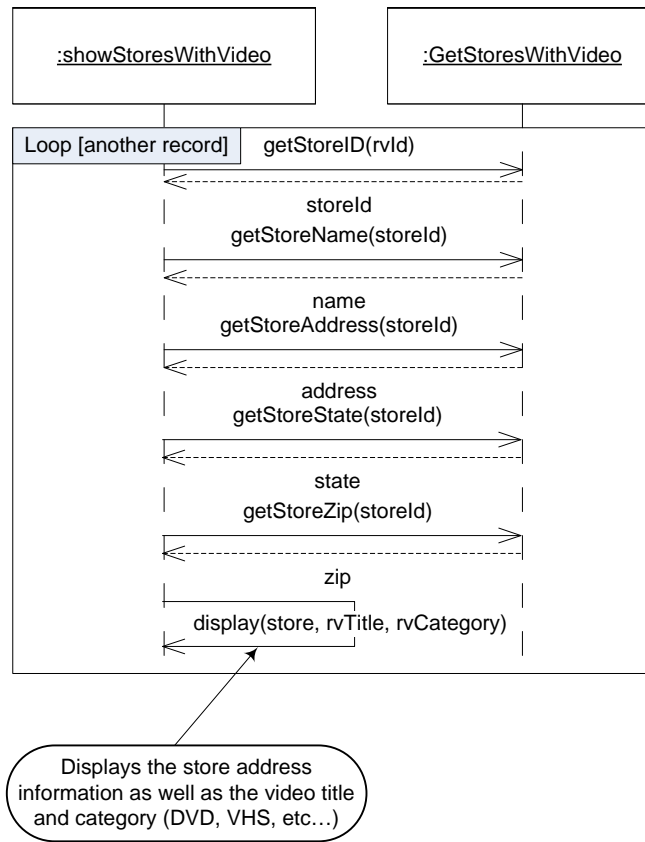
Modify Store Product



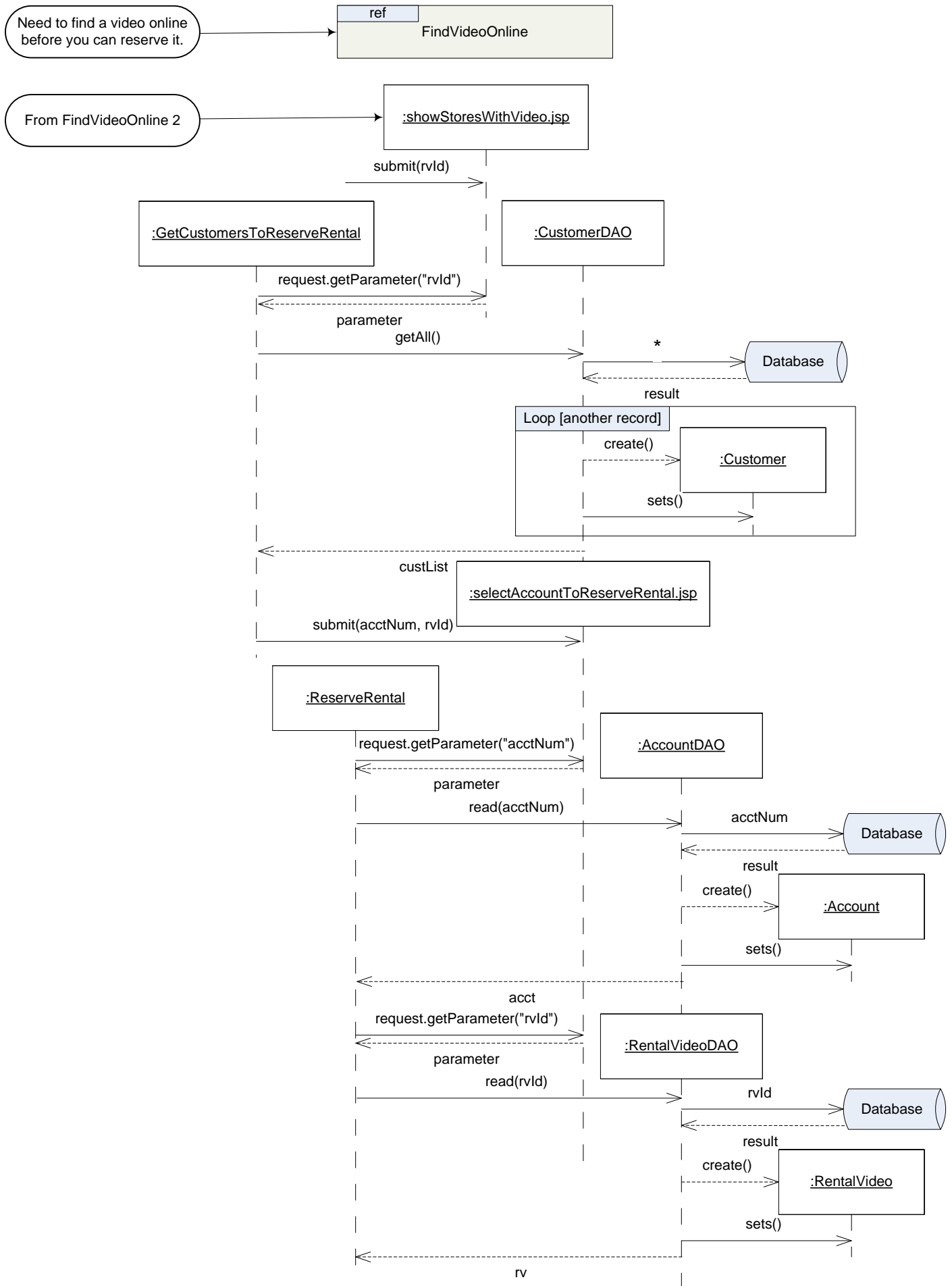
Renew Membership



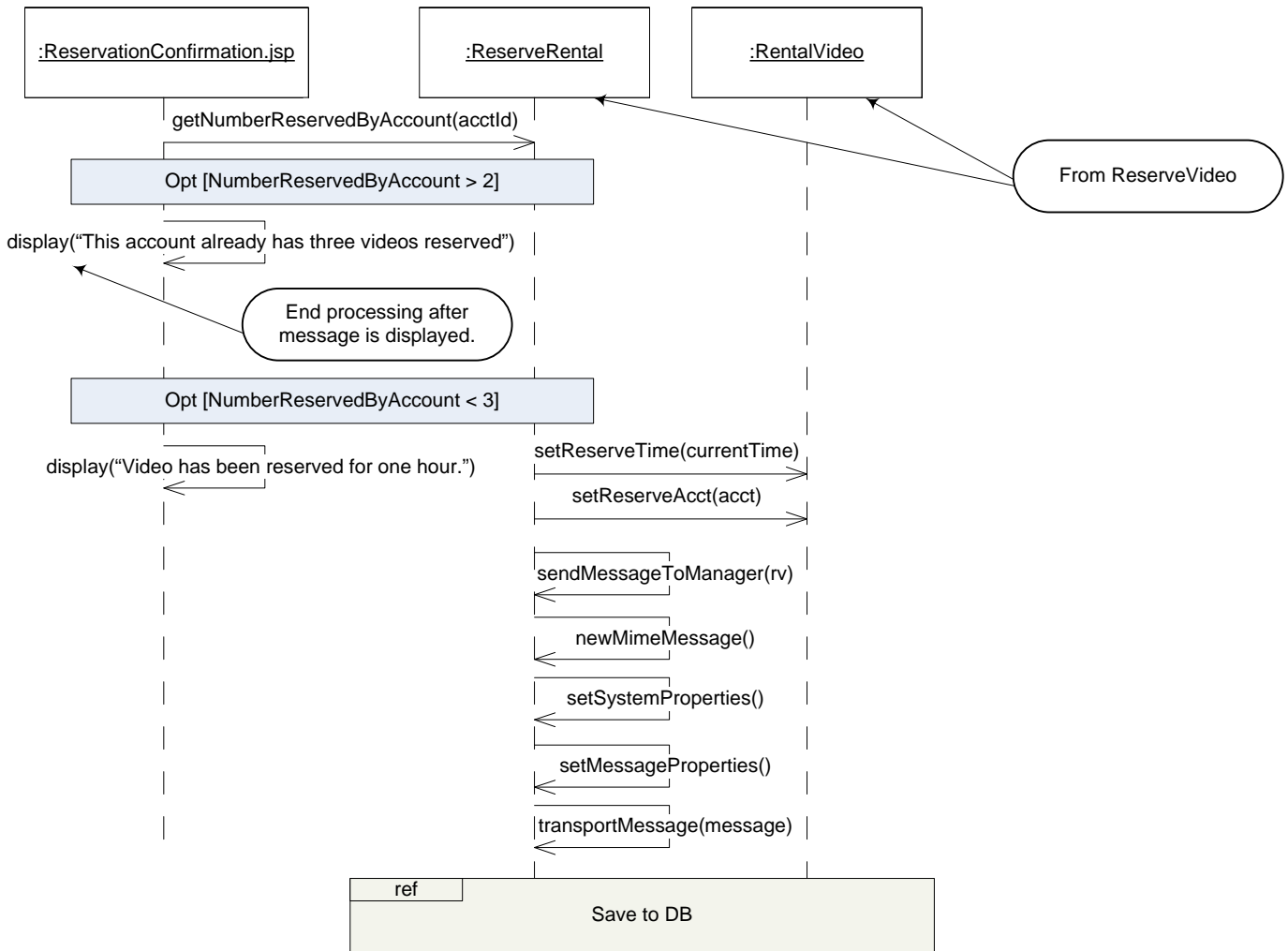




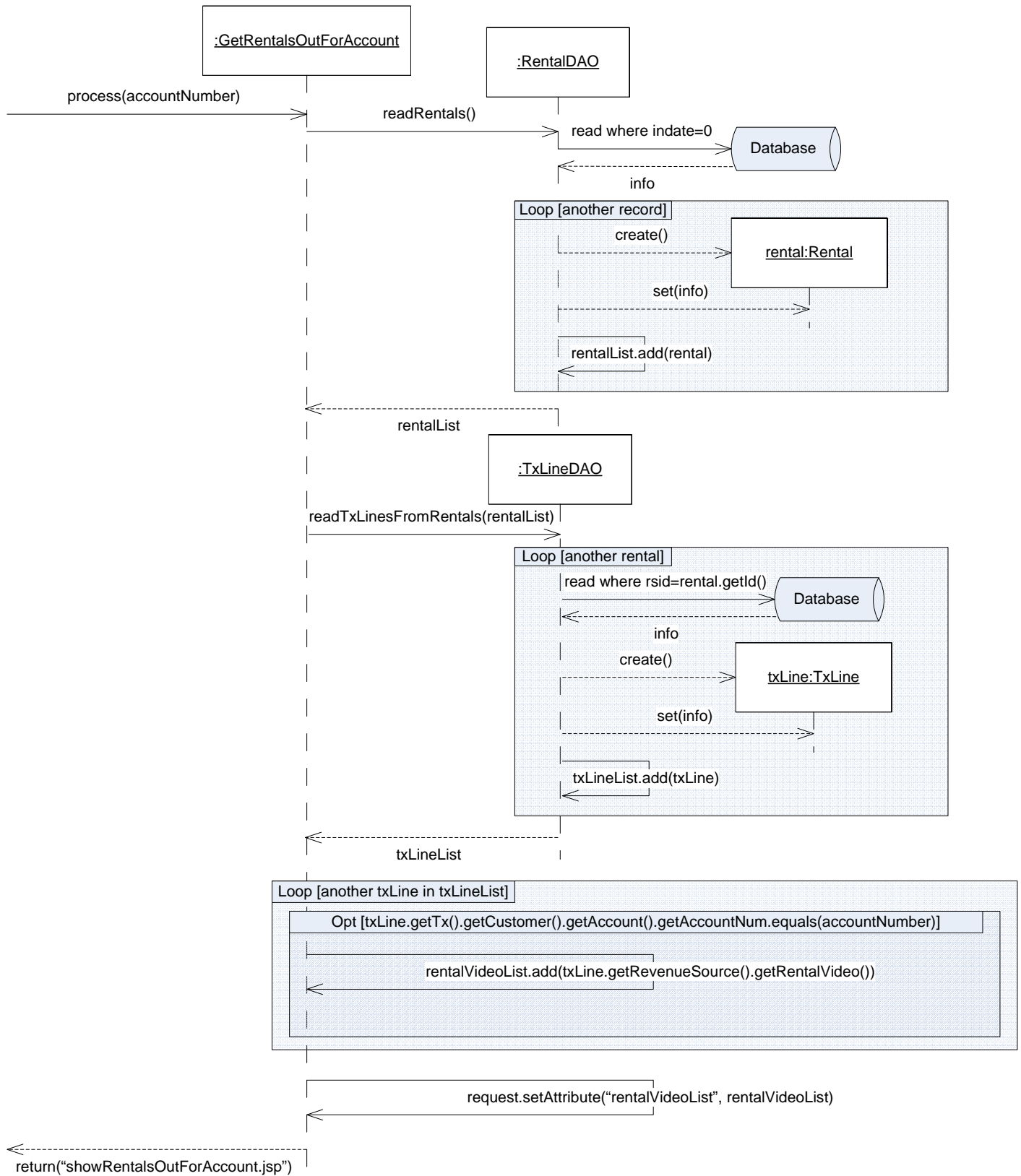
Reserve Video



Reserve Video 2



Get Checked-out Video



Assess Full-cost Fees

