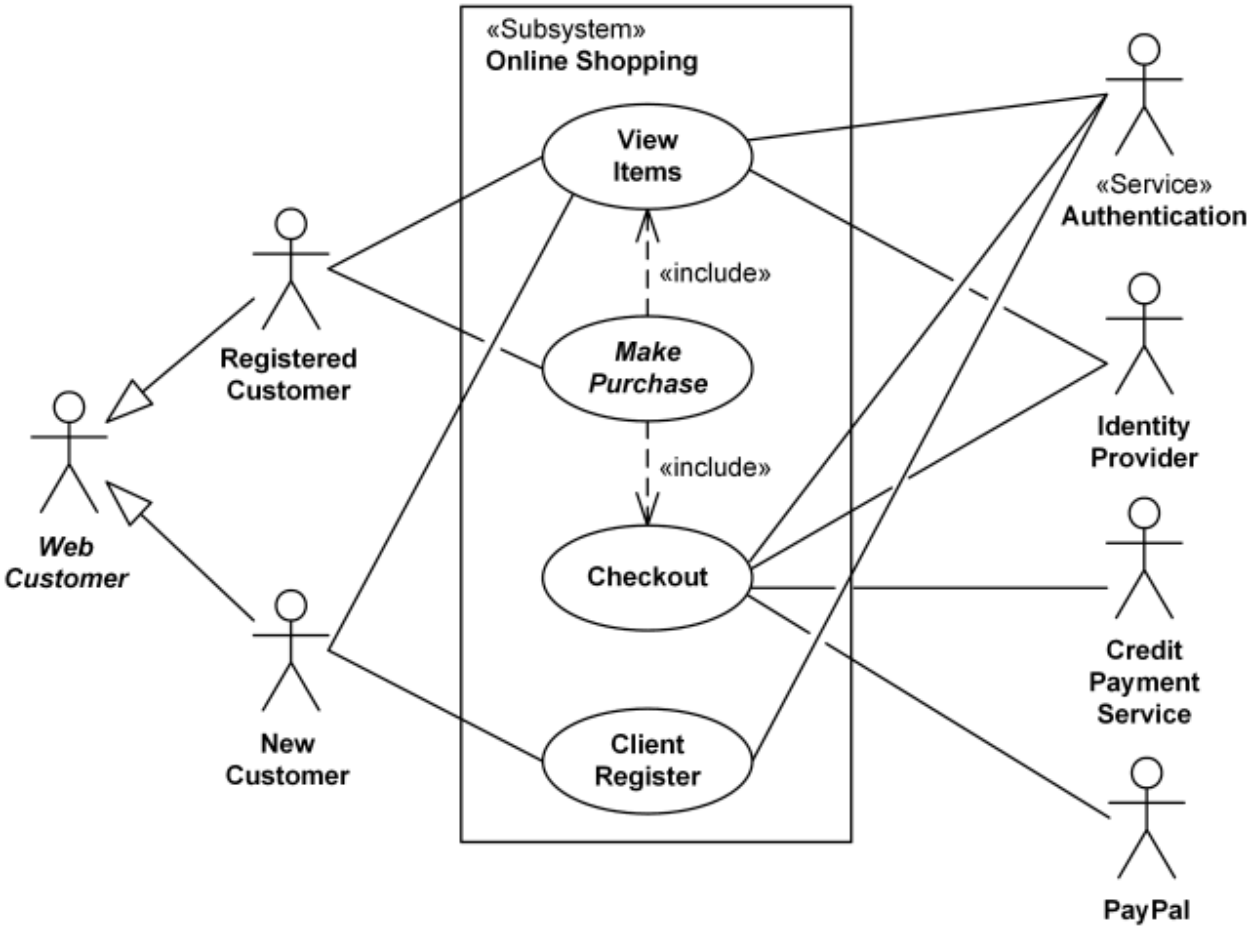
Some exercise from UML diagrams

Use case Diagram

Web Customer **actor** uses some web site to make purchases online. Top level **use cases** are **View Items**, **Make Purchase** and **Client Register**. View Items use case could be used by customer as top level use case if customer only wants to find and see some products. This use case could also be used as a part of Make Purchase use case. Client Register use case allows customer to register on the web site, for example to get some coupons or be invited to private sales. Note, that **Checkout** use case is **included use case** not available by itself - checkout is part of making purchase.
Except for the **Web Customer** actor there are several other actors which will be described below with detailed use cases.



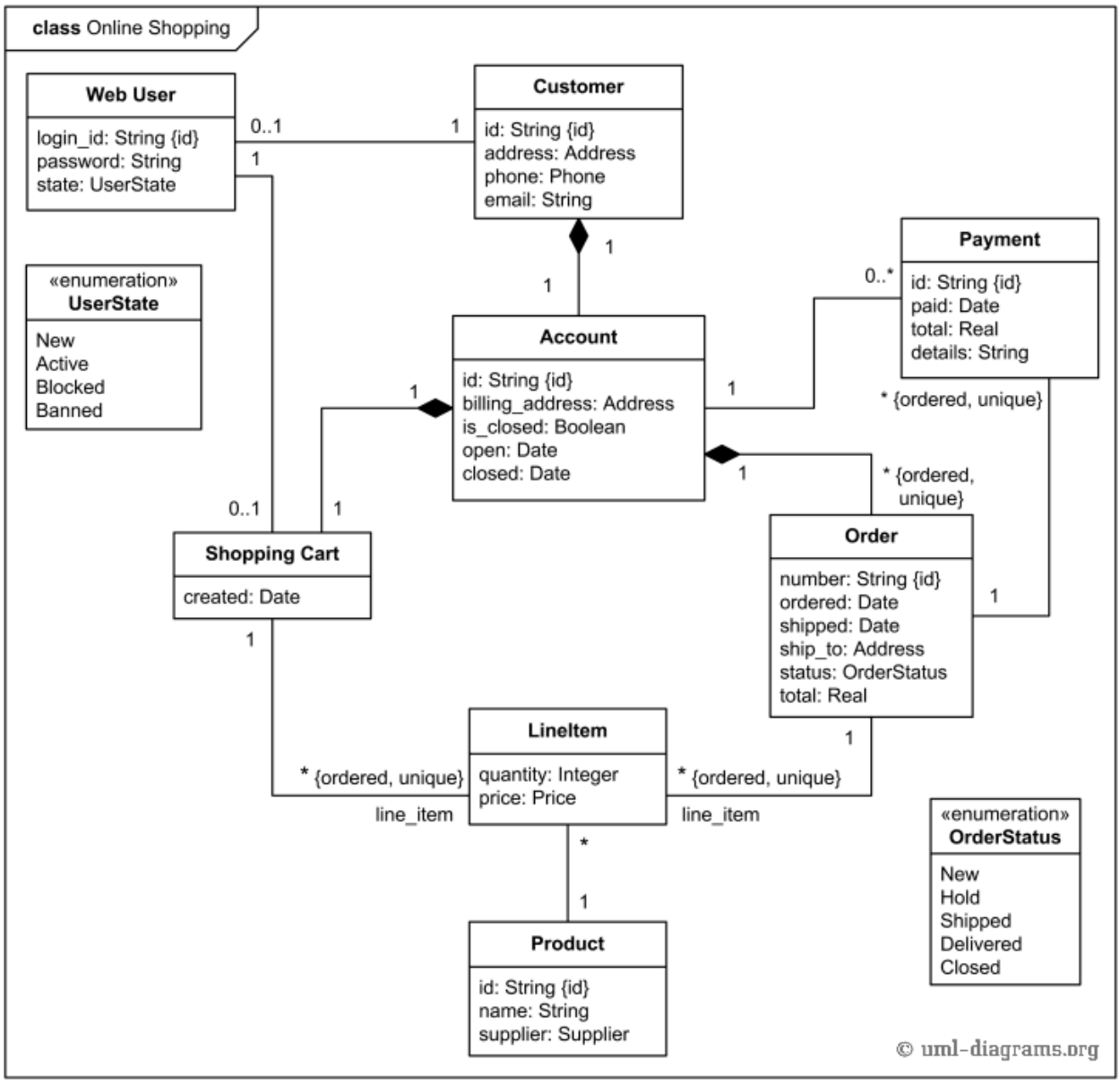
Class Diagram

class diagram which shows a domain model for online shopping. The purpose of the diagram is to introduce some common terms, "dictionary" for online shopping - Customer, Web User, Account, Shopping Cart, Product, Order, Payment, etc. and relationships between. It could be used as a common ground between business analysts and software developers.

Each customer has unique id and is linked to exactly one **account**. Account owns shopping cart and orders. Customer could register as a web user to be able to buy items online. Customer is not required to be a web user because purchases could also be made by phone or by ordering from catalogues. Web user has login name which also serves as unique id. Web user could be in several states - new, active, temporary blocked, or banned, and be linked to a **shopping cart**. Shopping cart belongs to account.

Account owns customer orders. Customer may have no orders. Customer orders are sorted and unique. Each order could refer to several **payments**, possibly none. Every payment has unique id and is related to exactly one account.

Each order has current order status. Both order and shopping cart have **line items** linked to a specific product. Each line item is related to exactly one product. A product could be associated to many line items or no item at all.

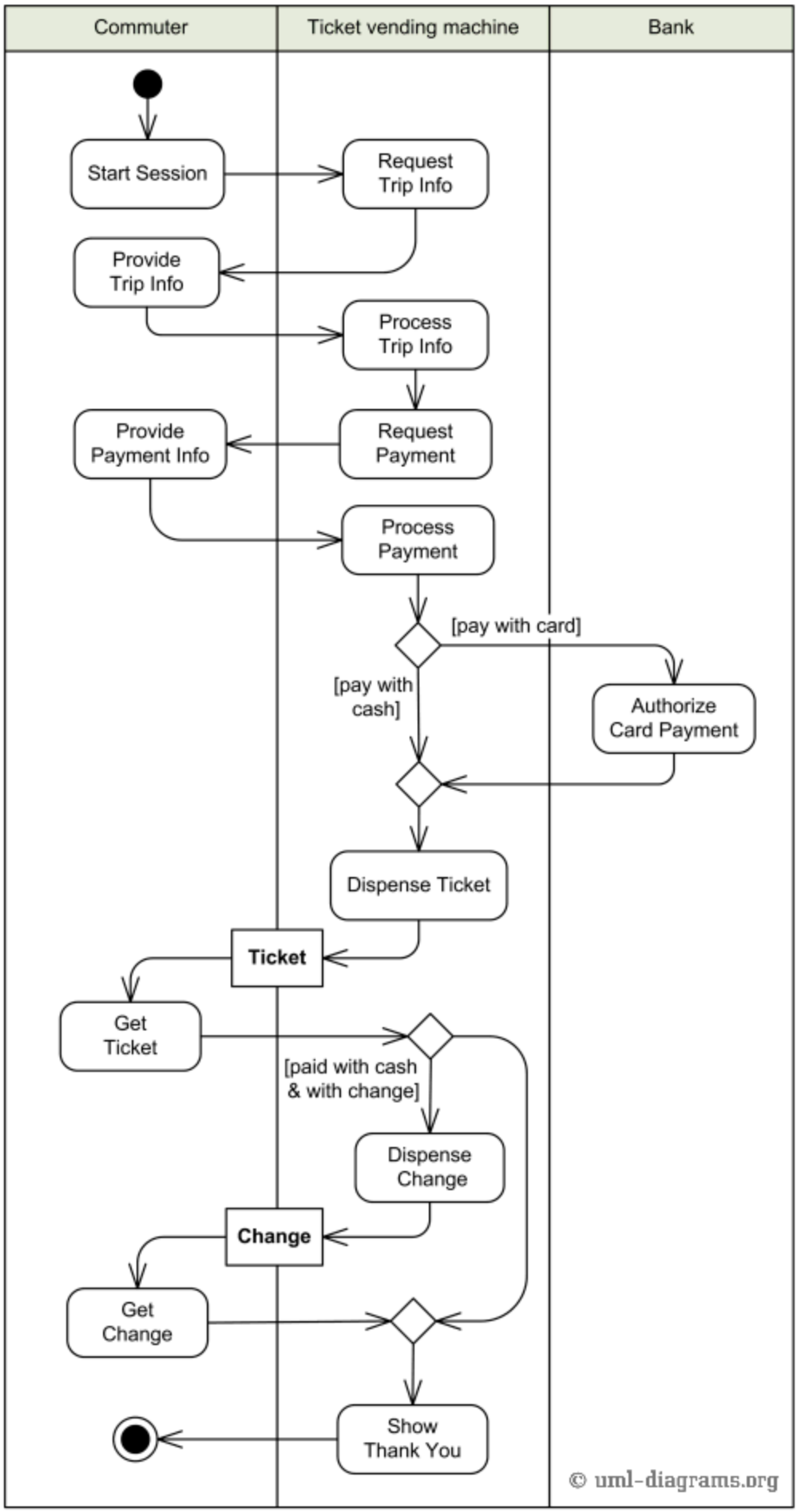

class Online Shopping
Web User
login_id: String {id}
password: String
state: UserState
0..1
1
1
Customer
id: String {id}
address: Address
phone: Phone
email: String
1
1
«enumeration»
UserState
New
Active
Blocked
Banned
Payment
id: String {id}
paid: Date
total: Real
details: String
0..*
* {ordered, unique}
Account
id: String {id}
billing_address: Address
is_closed: Boolean
open: Date
closed: Date
1
1
1
1
* {ordered, unique}
0..1
1
Shopping Cart
created: Date
Order
number: String {id}
ordered: Date
shipped: Date
ship_to: Address
status: OrderStatus
total: Real
1
1
1
LineItem
quantity: Integer
price: Price
* {ordered, unique}
line_item
* {ordered, unique}
line_item
*
1
Product
id: String {id}
name: String
supplier: Supplier
«enumeration»
OrderStatus
New
Hold
Shipped
Delivered
Closed
© uml-diagrams.org

Activity Diagram

Activity is started by Commuter **actor** who needs to buy a ticket. Ticket vending machine will request trip information from Commuter. This information will include number and type of tickets, e.g. whether it is a monthly pass, one way or round ticket, route number, destination or zone number, etc.

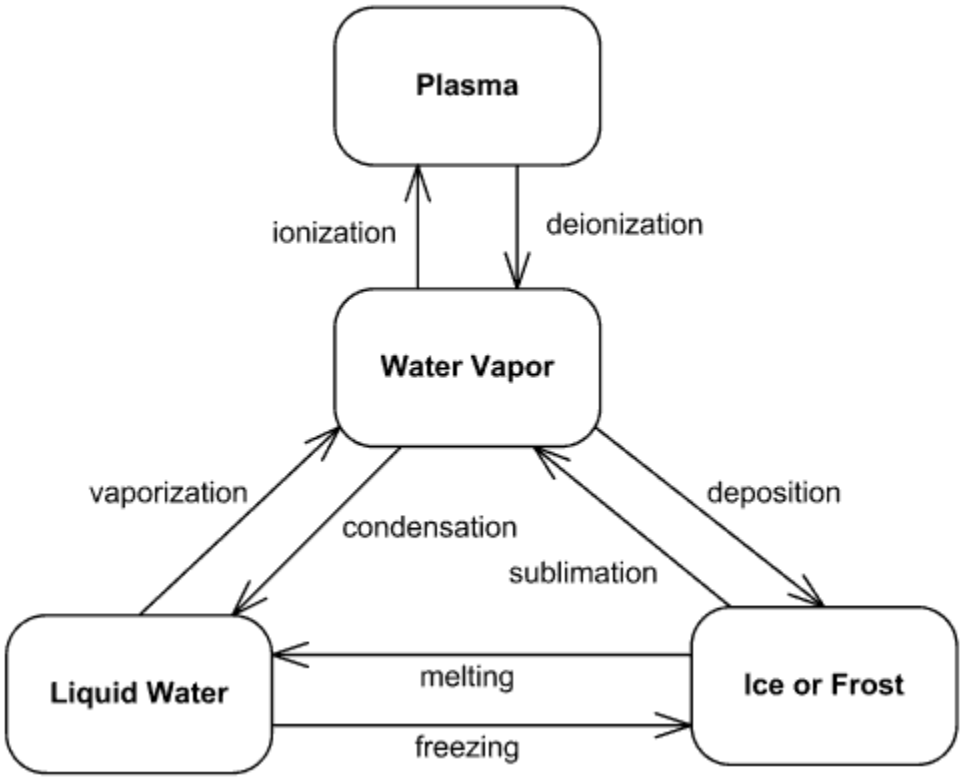
Based on the provided trip info ticket vending machine will calculate payment due and request payment options. Those options include payment by cash, or by credit or debit card. If payment by card was selected by Commuter, another actor, Bank will participate in the activity by authorizing the payment.

After payment is complete, ticket is dispensed to the Commuter. Cash payment might result in some change due, so the change is dispensed to the Commuter in this case. Ticket vending machine will show some "Thank You" screen at the end of the activity.


Commuter
Ticket vending machine
Bank
Start Session
Request Trip Info
Provide Trip Info
Process Trip Info
Provide Payment Info
Request Payment
Process Payment
[pay with card]
[pay with cash]
Authorize Card Payment
Dispense Ticket
Ticket
Get Ticket
[paid with cash & with change]
Dispense Change
Change
Get Change
Show Thank You
© uml-diagrams.org

State Diagram

Water can exist in several states - liquid, vapor, solid, and plasma. Several transitions are possible from one state to another.


Plasma
ionization
deionization
Water Vapor
vaporization
condensation
sublimation
deposition
Liquid Water
melting
freezing
Ice or Frost

Every company having customers maintains customer accounts and supports a complete life cycle of the account from its creation until it is closed. There are differences in what are the stages (states) in the account's life cycle, and what are conditions or events causing account to change its state.

Here we provide an example of user account life cycle in the context of online shopping, shown as UML **protocol state machine** diagram.

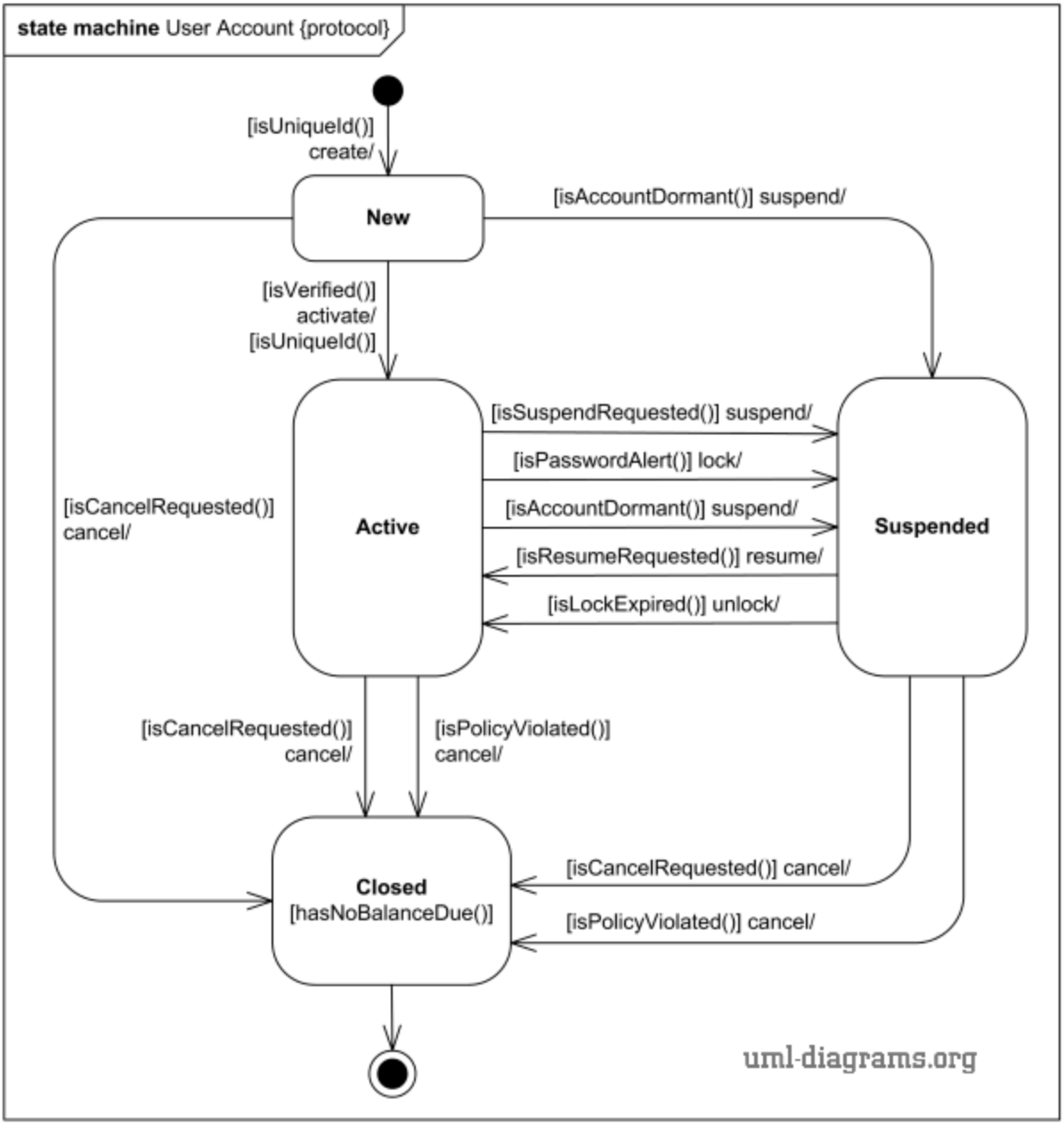
For the user account to be created, it has to meet some initial requirements. For example, user id (used as a login name) must be unique, at least for the existing accounts. After account was created, it might need to be verified. Verification depends on the company and could include e-mail, phone, and/or address verification. If account was not verified during some predefined period of time, that account could be moved to the suspended accounts.

New, active, or suspended accounts could be cancelled at any time by client's request. Note, the precondition for this usually includes payment of any outstanding balances, and might require some separate account state or substate to handle this case.

User account might be suspended for security reasons, manually or automatically. For example, website intrusion detection system locks user account for predefined period of time, if there were several unsuccessful login attempts using incorrect account password. After account lock times out, account is activated back automatically.

Some user accounts could be inactive for a long period of time. Company policy or business rules could require moving such dormant for a year or two accounts to the suspended state.

Once we listed user account states and specified all possible transitions from one state to another, we can review the diagram with other subject matter experts to see if anything is missing or needs further clarifications.


state machine User Account {protocol}
[isUniqueId()]
create/
New
[isAccountDormant()] suspend/
[isVerified()]
activate/
[isUniqueId()]
Active
[isSuspendRequested()] suspend/
[isPasswordAlert()] lock/
[isAccountDormant()] suspend/
[isResumeRequested()] resume/
[isLockExpired()] unlock/
Suspended
[isCancelRequested()]
cancel/
[isCancelRequested()]
cancel/
[isPolicyViolated()]
cancel/
Closed
[hasNoBalanceDue()]
[isCancelRequested()] cancel/
[isPolicyViolated()] cancel/
uml-diagrams.org