Collaboration Diagram
April 11th, 2018

Interaction diagrams are models that describe how a group of objects collaborate in some behavior - typically a single use-case. The diagrams show a number of example objects and the messages that are passed between these objects within the use-case. Interaction diagrams should be used when you want to look at the behavior of several objects within a single use case. They are good at showing the collaborations between the objects but they are not so good at providing a precise definition of the behavior.

Notation
Diagrams represent a collaboration and interaction. Collaboration captures a set of objects and their interactions in a specific context. An interaction is a set of messages exchanged in a collaboration to produce a desired result. The objects are typically represented as follows:
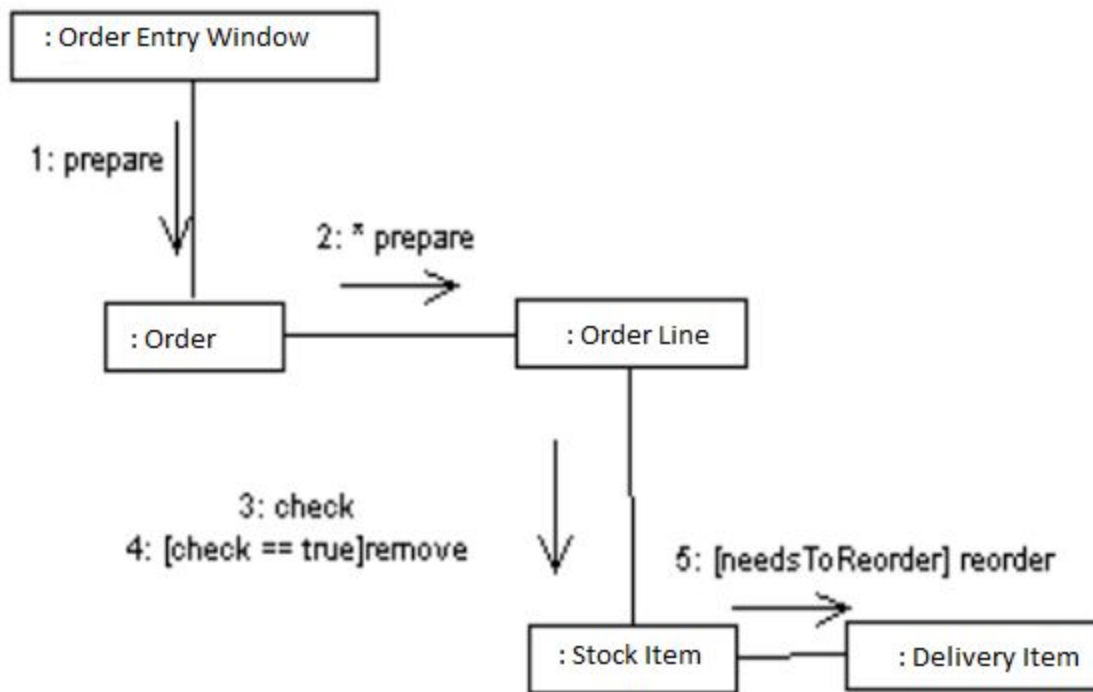- rectangles containing object signature
- object signature: object name: object class
  – object name is optional and starts with a lowercase letter
  – class name is mandatory and starts with an uppercase letter
  – objects are connected by lines
- Messages are labeled like function calls in languages like C. Message can have parameters and return values. Messages are typically written on top of an arrow that show the direction of message flow. Simple collaboration diagrams simply number the messages in sequence. More complex schemes use a decimal numbering approach to indicate if messages are sent as part of the implementation of another message. In addition, a letter can be used to show concurrent threads.
- One can specify conditions that need to be satisfied to successfully execute the message as follows:
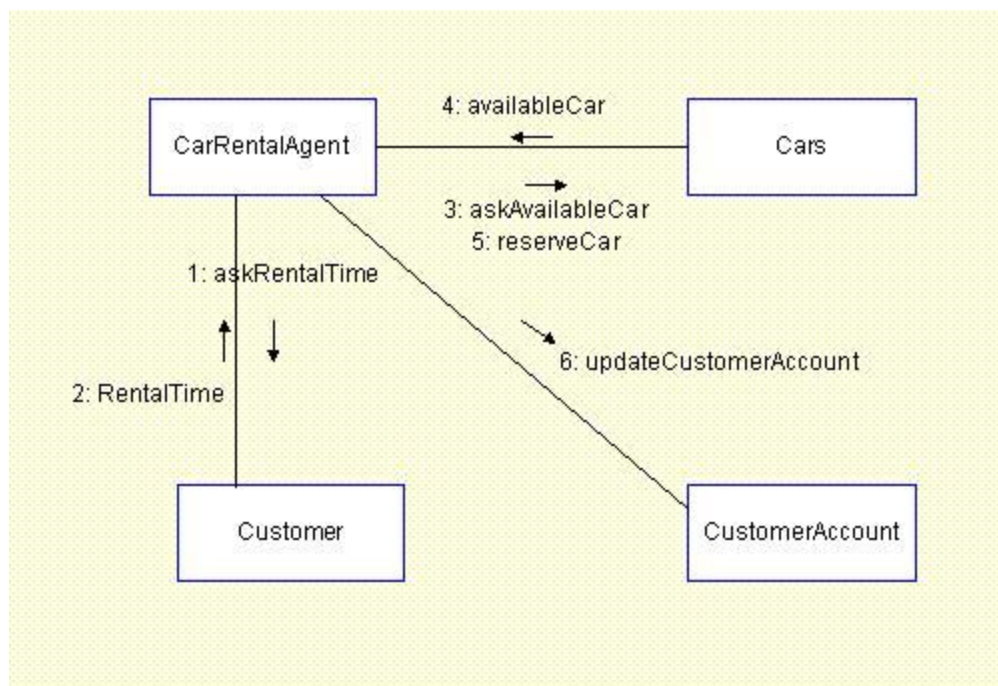 seq_no: [condition] Message.

Examples
Order Processing: In this behavior, the order entry window sends a prepare message to an order. The order then sends prepare to each order line on the order. The order line first checks the stock item, and if the check returns true, it removes stock from the stock item. If stock item falls below the reorder level, it requests a new delivery. (see Fig. 1)
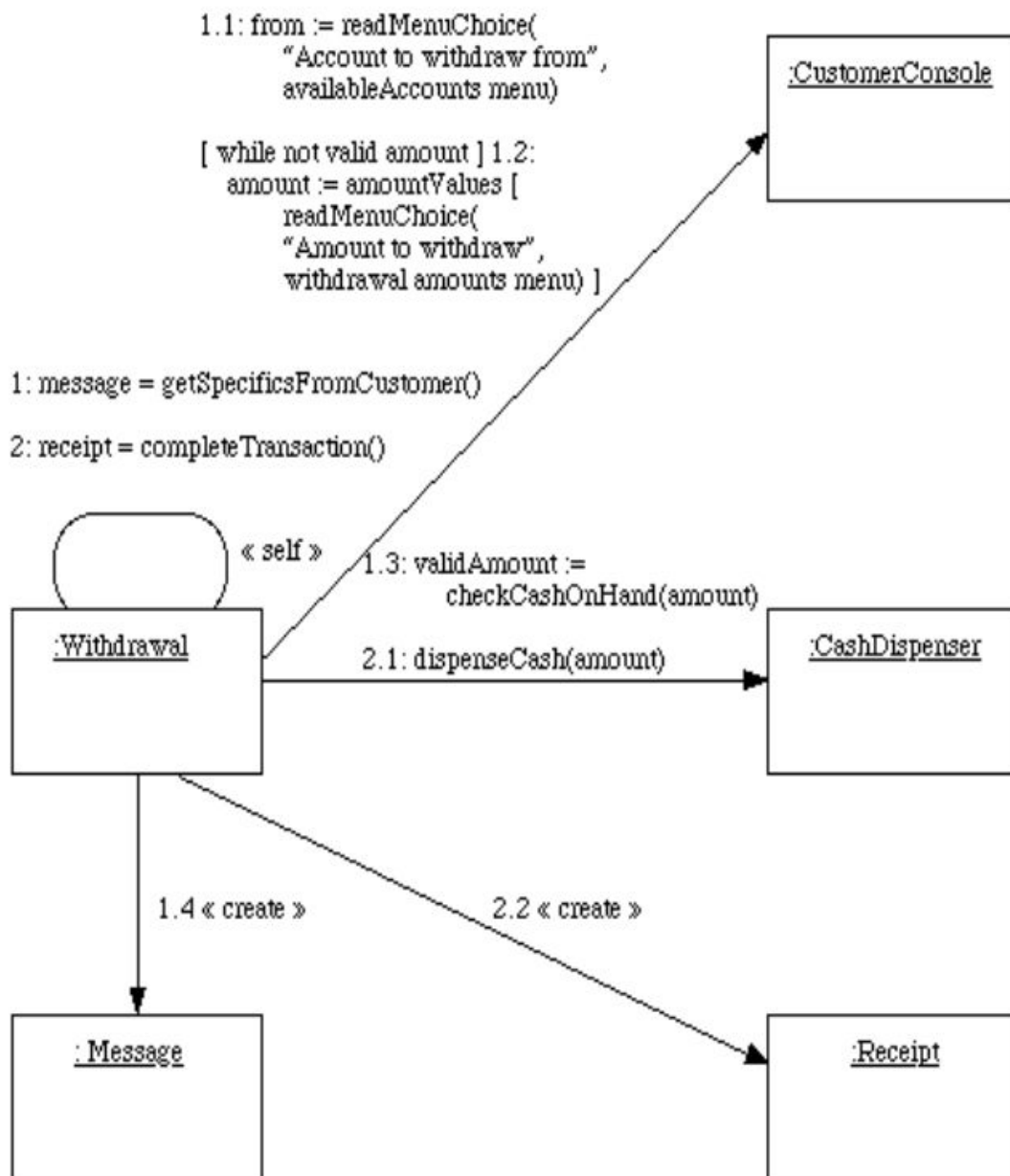**Car Rental** (see Fig 2)
Bank: Withdrawal transaction (see Fig. 3)

Figure 1: Order Processing

Figure 2: Car Rental

1.1: from := readMenuChoice(
        "Account to withdraw from",
        availableAccounts menu)

[while not valid amount] 1.2:
    amount := amountValues [
        readMenuChoice(
        "Amount to withdraw",
        withdrawal amounts menu)]

1: message = getSpecificsFromCustomer()

2: receipt = completeTransaction()

« self »

:Withdrawal

1.3: validAmount :=
        checkCashOnHand(amount)

2.1: dispenseCash(amount)

:CustomerConsole

:CashDispenser

1.4 « create »

2.2 « create »

: Message

:Receipt

Figure 3: Withdrawal Transaction Collaboration

******