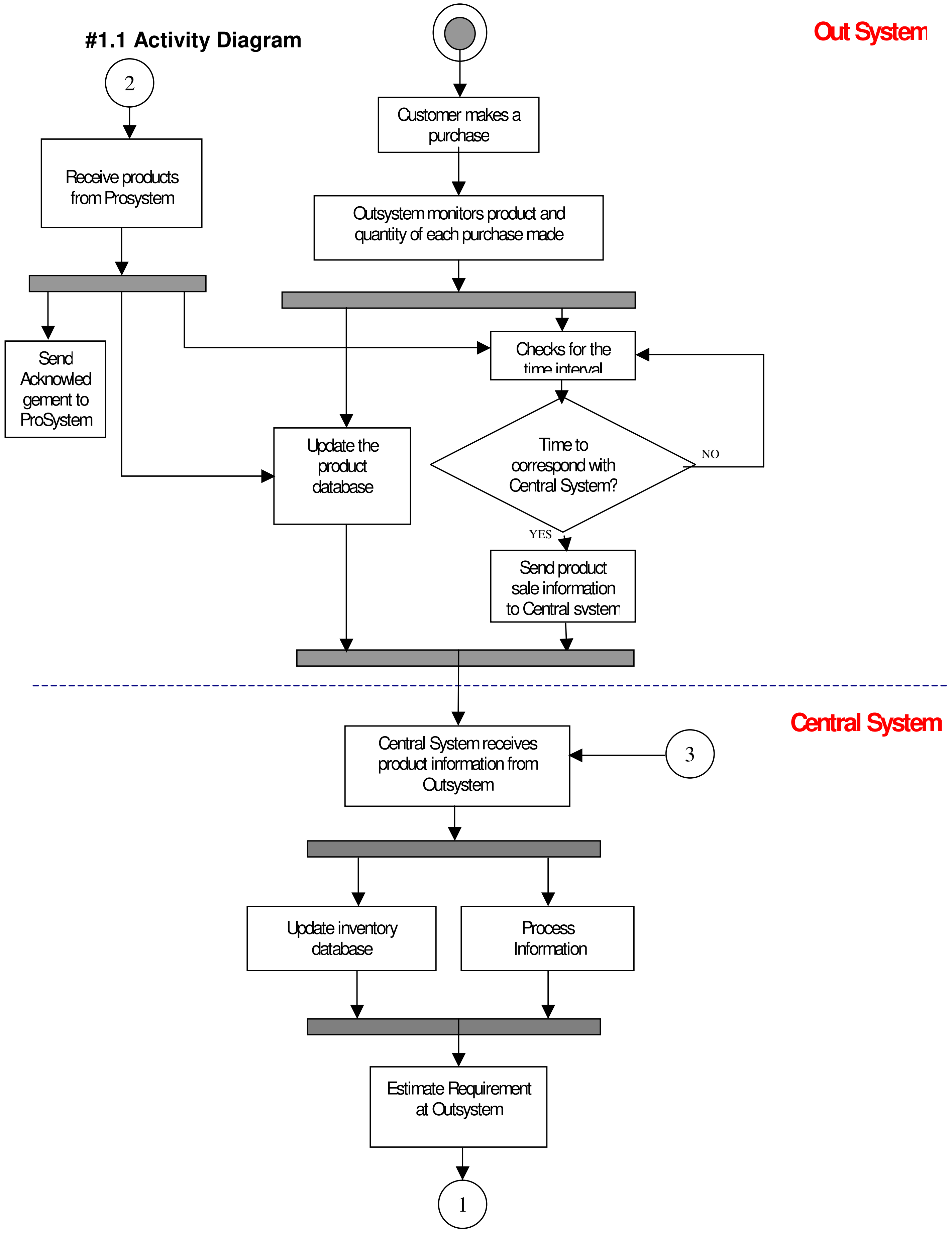
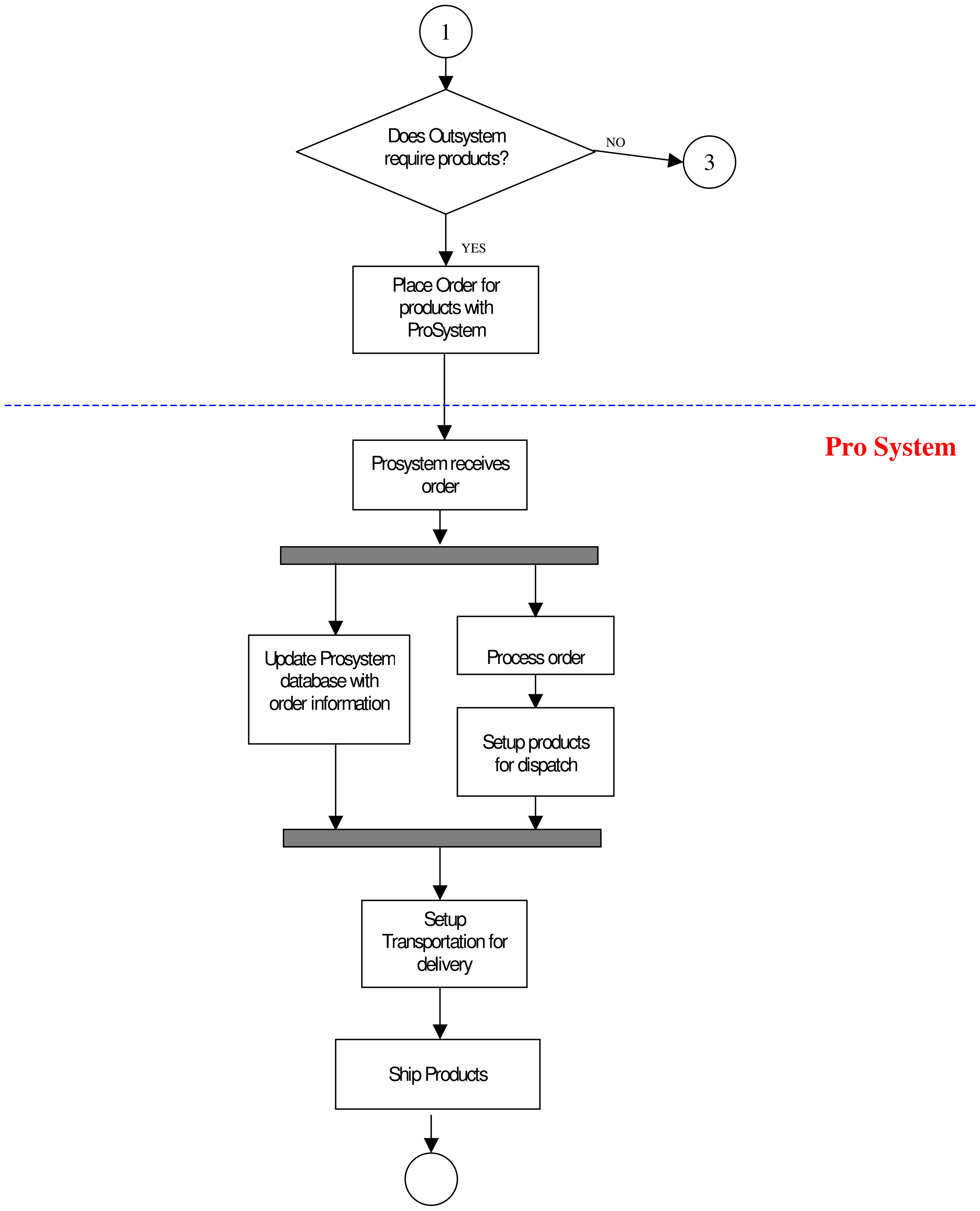
#1.1 Activity Diagram

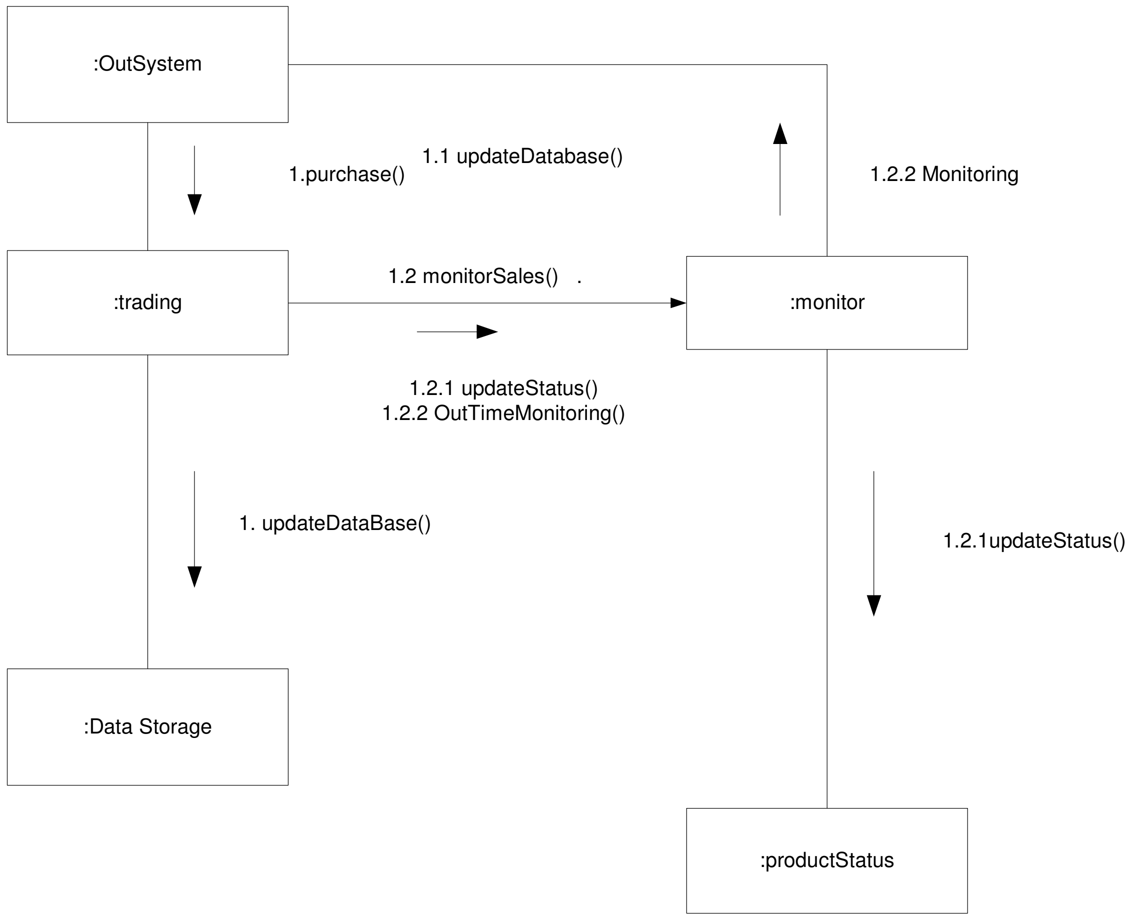


Out System
2
Receive products from Prosystem
Send Acknowled gement to ProSystem
Customer makes a purchase
Outsystem monitors product and quantity of each purchase made
Checks for the time interval
Update the product database
Time to correspond with Central System?
NO
YES
Send product sale information to Central system
Central System
Central System receives product information from Outsystem
3
Update inventory database
Process Information
Estimate Requirement at Outsystem
1

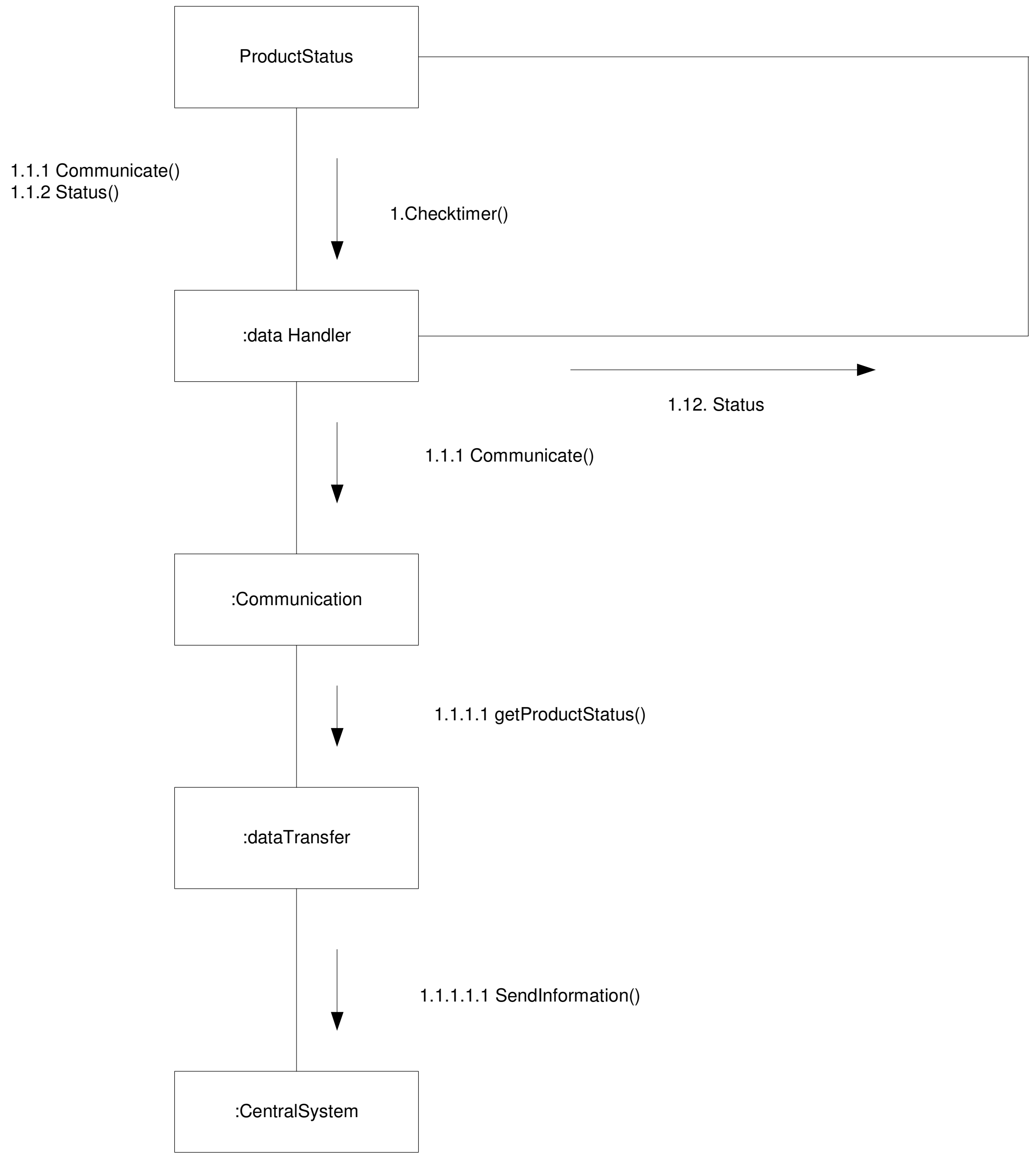

1
Does Outsystem require products?
NO
3
YES
Place Order for products with ProSystem
Pro System
Prosystem receives order
Update Prosystem database with order information
Process order
Setup products for dispatch
Setup Transportation for delivery
Ship Products

#1.2 Collaboration Diagram

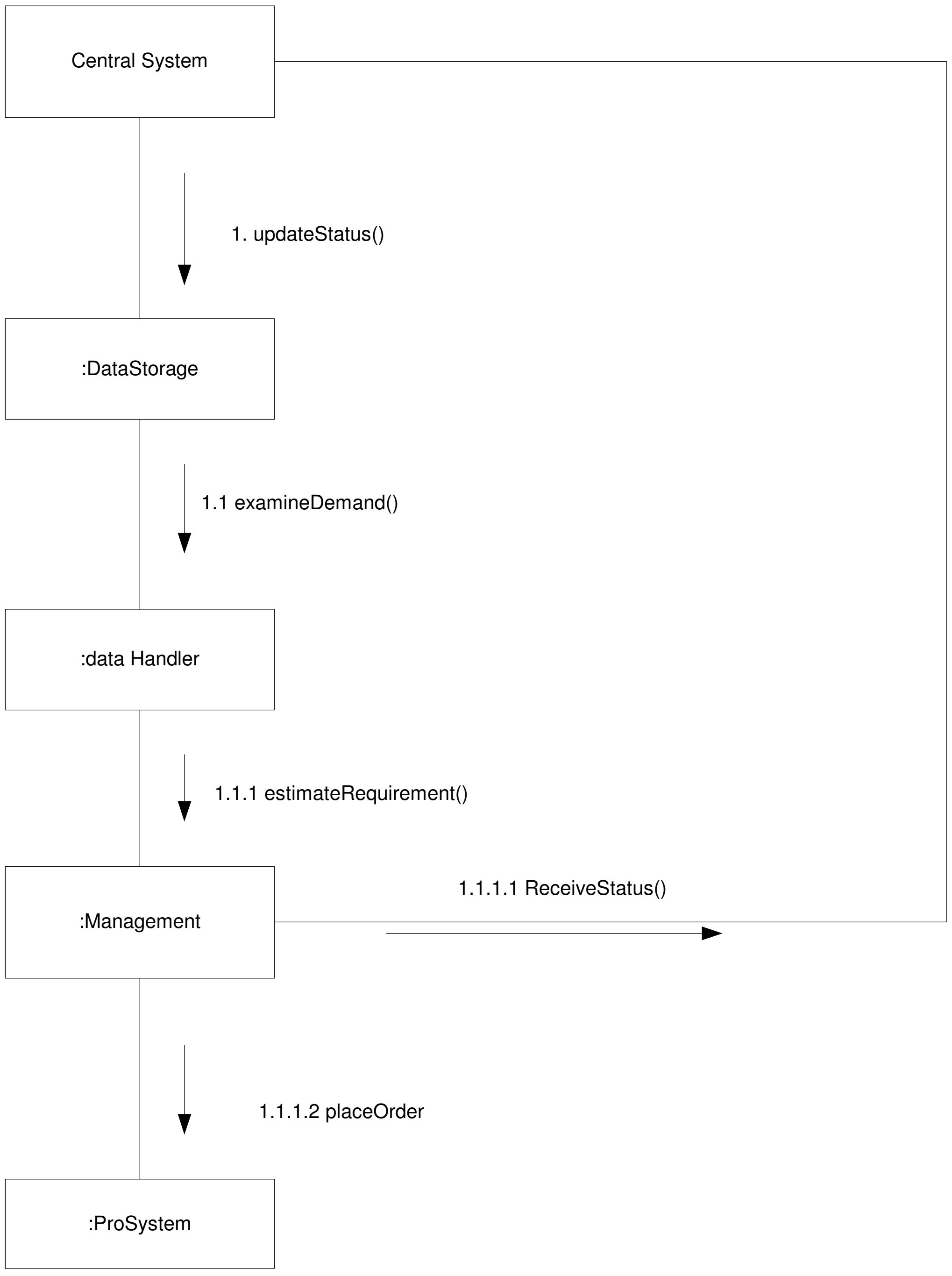
Use Case #1: Monitoring product sales



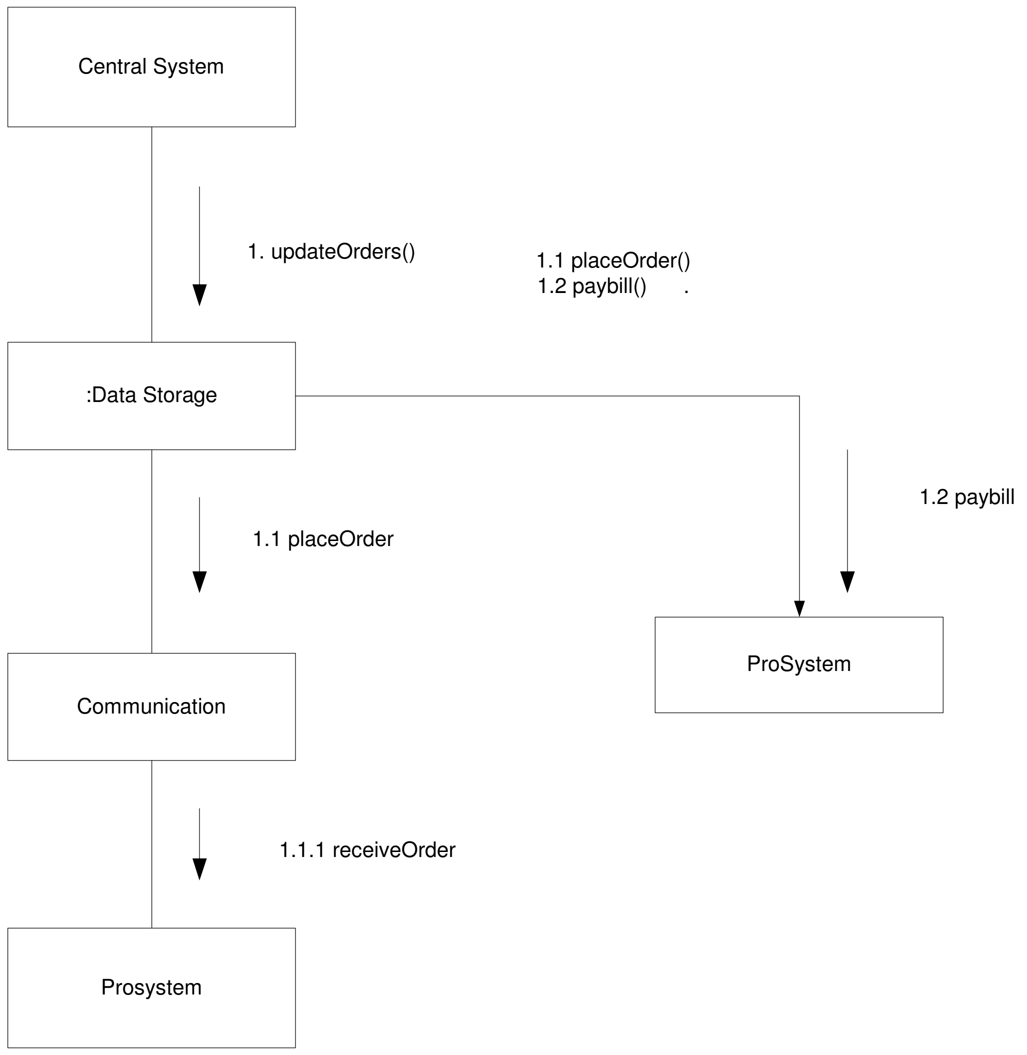
Use Case #2: OutSystem sending information



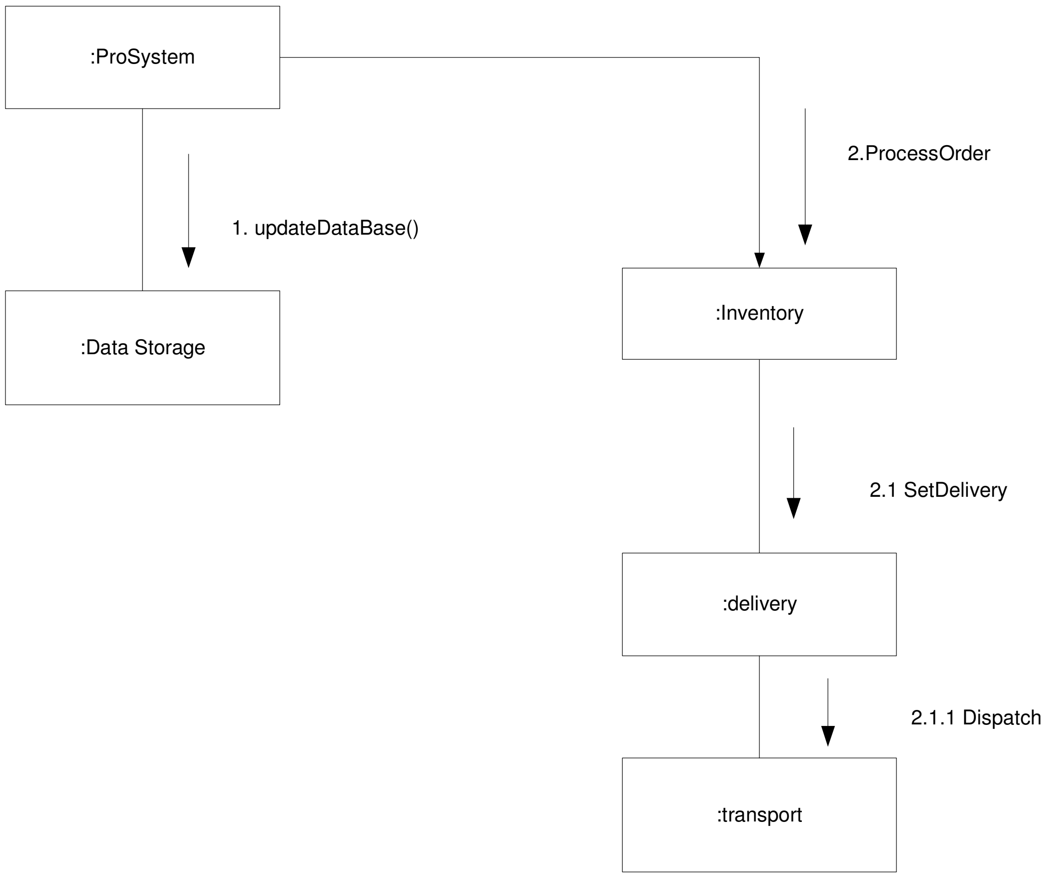
Use Case #3: CentralSystem processing incoming information



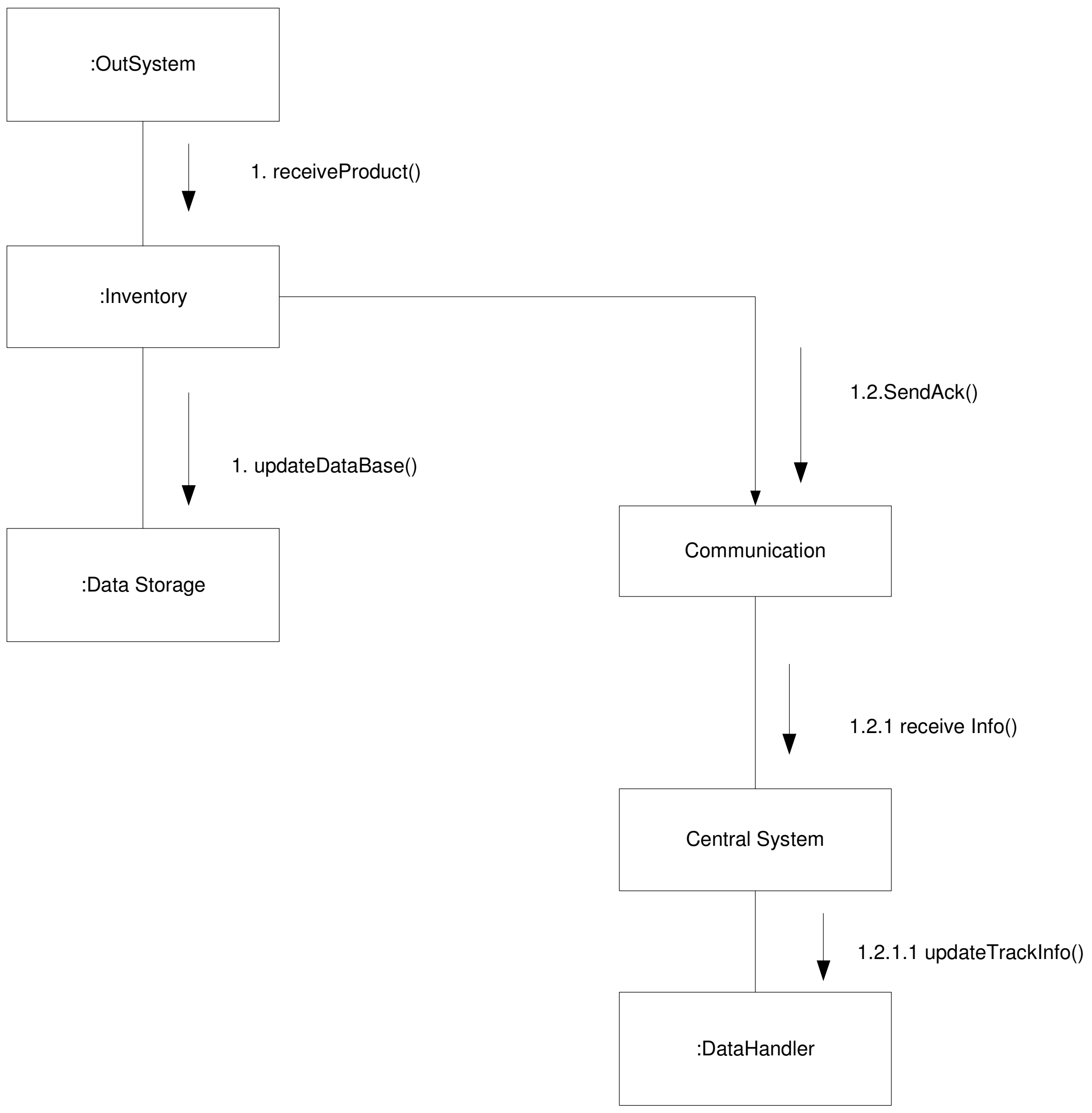
Use Case #4: CentralSystem placing order



Use Case #5: Servicing product request.

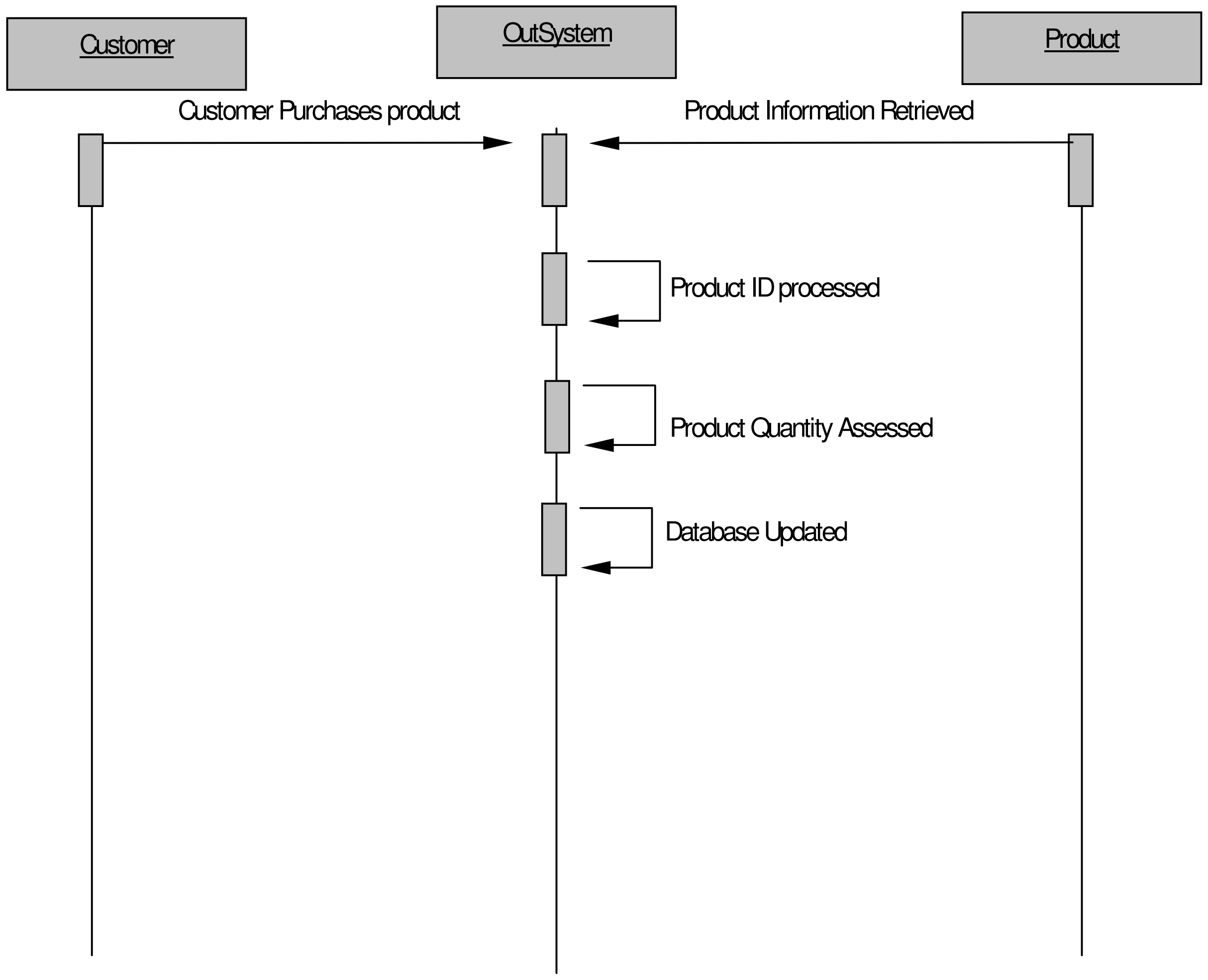


Use case #6: Order receipt acknowledgement

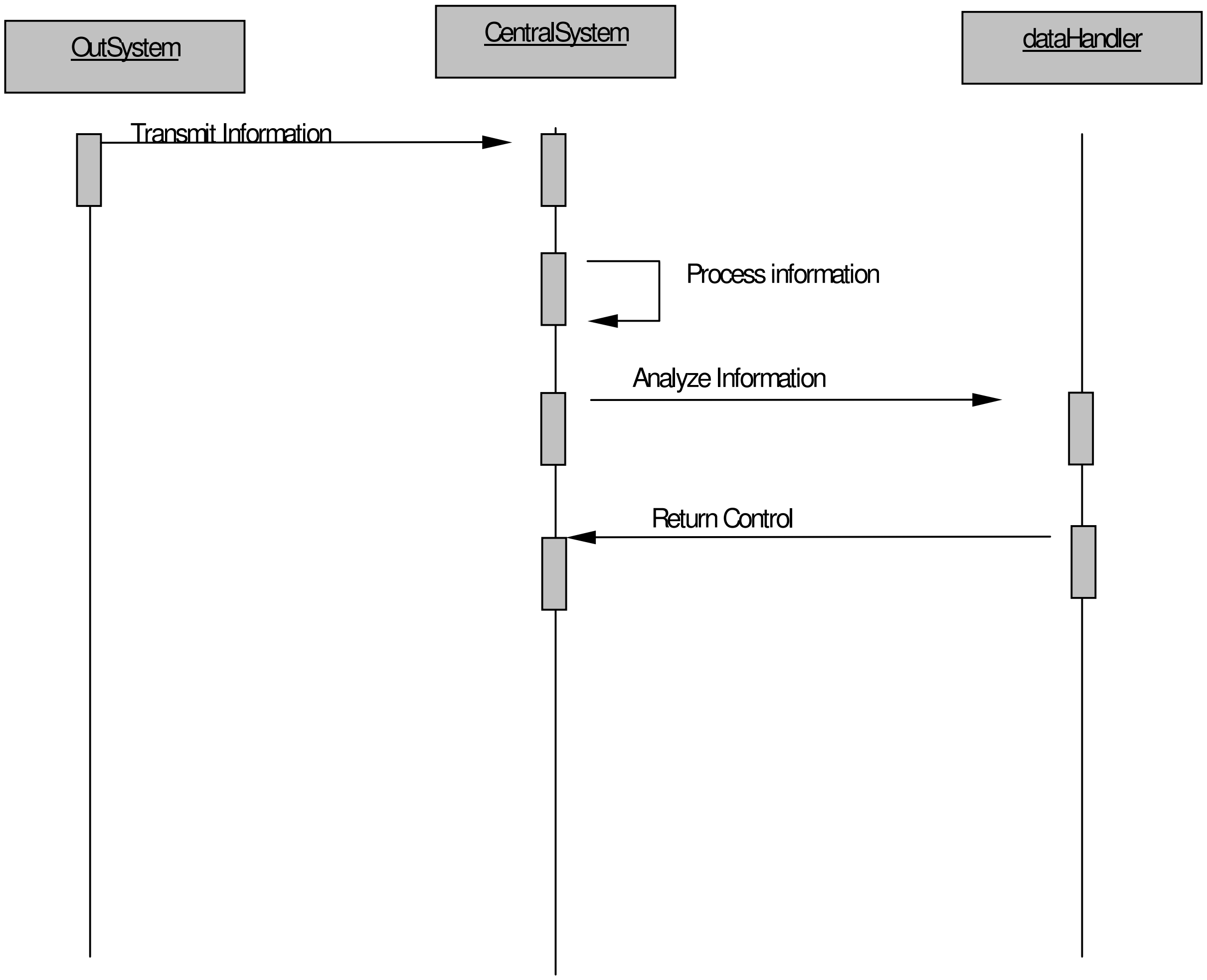


#1.3 Sequence Diagrams

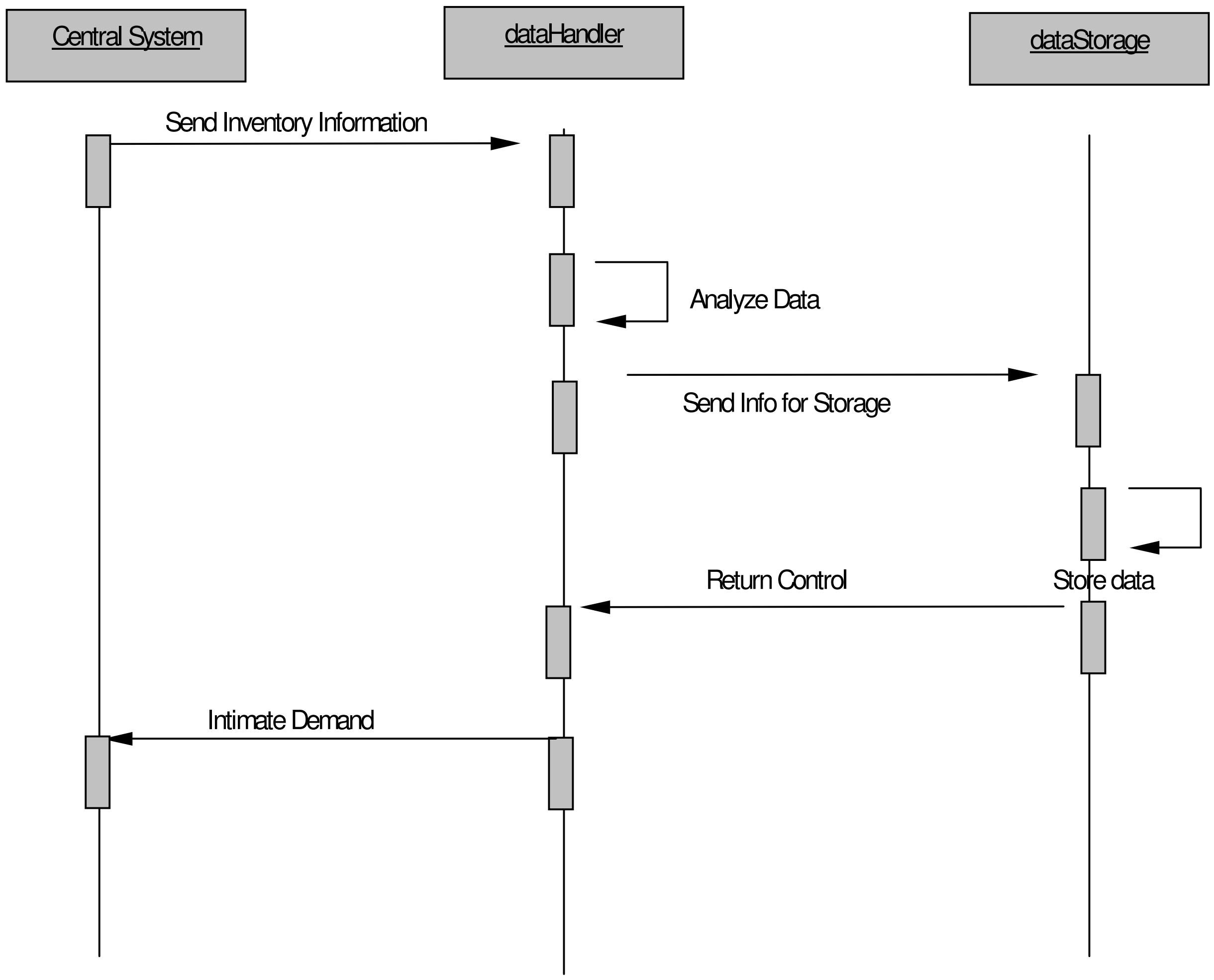
Use Case #1: Monitoring product sales


Customer
OutSystem
Product
Customer Purchases product
Product Information Retrieved
Product ID processed
Product Quantity Assessed
Database Updated

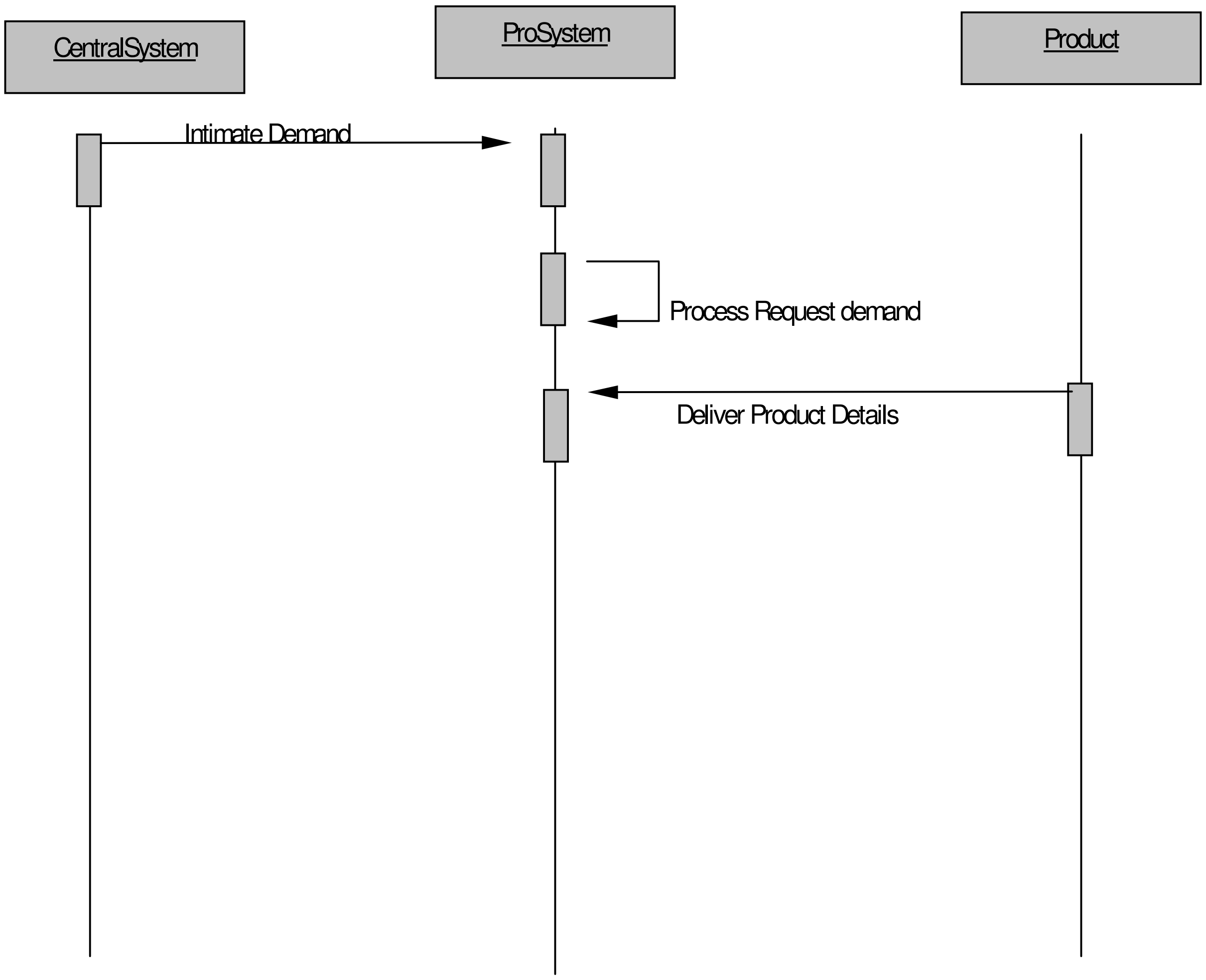
Use Case #2: OutSystem sending information



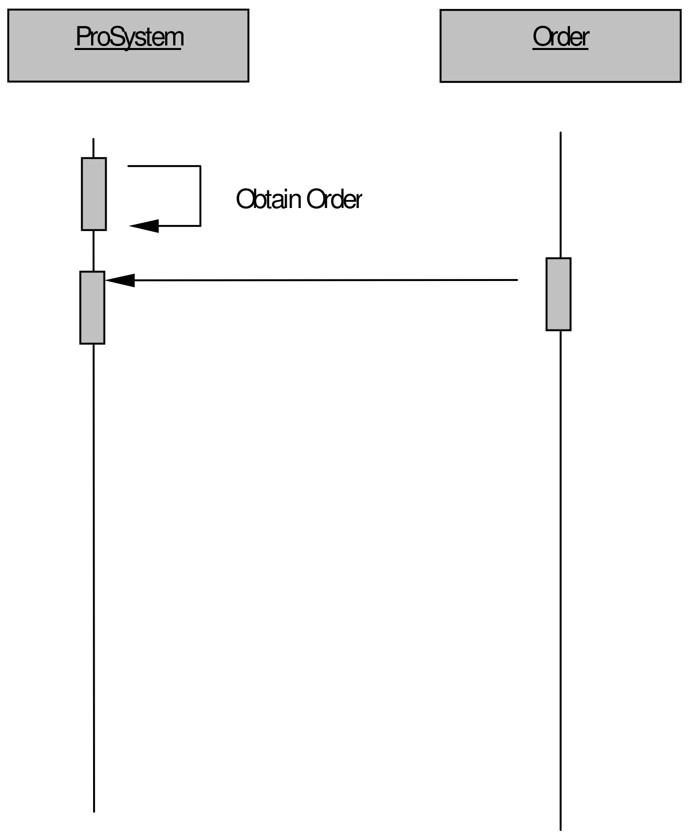
Use Case #3: CentralSystem processing incoming information



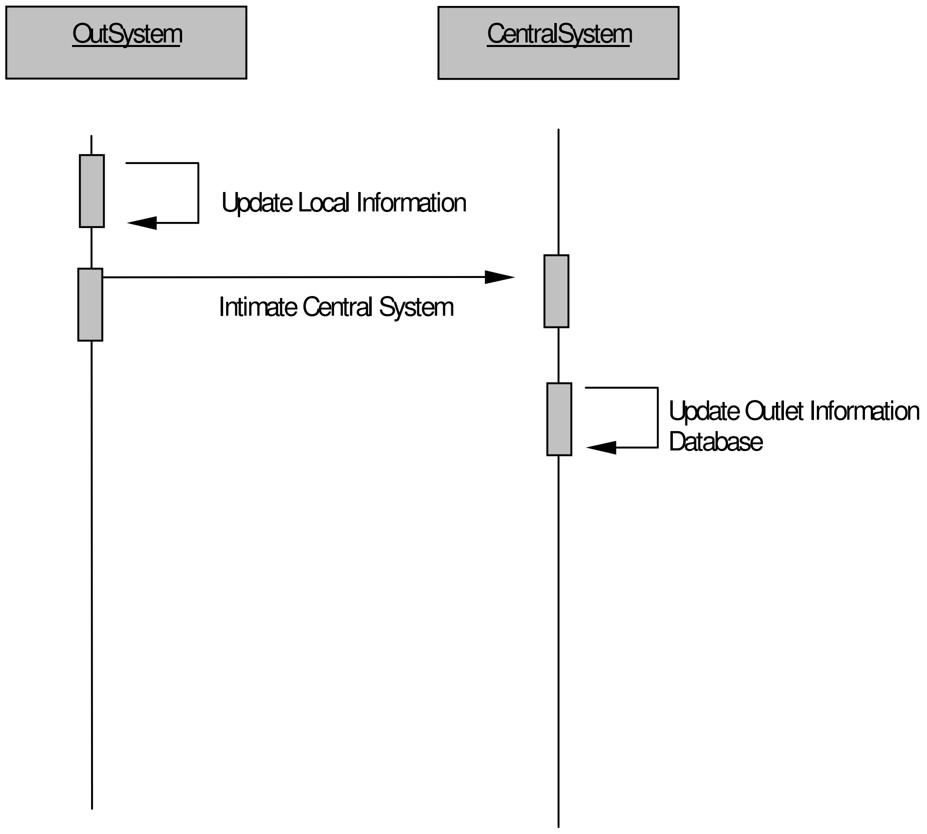
Use Case #4: CentralSystem placing order



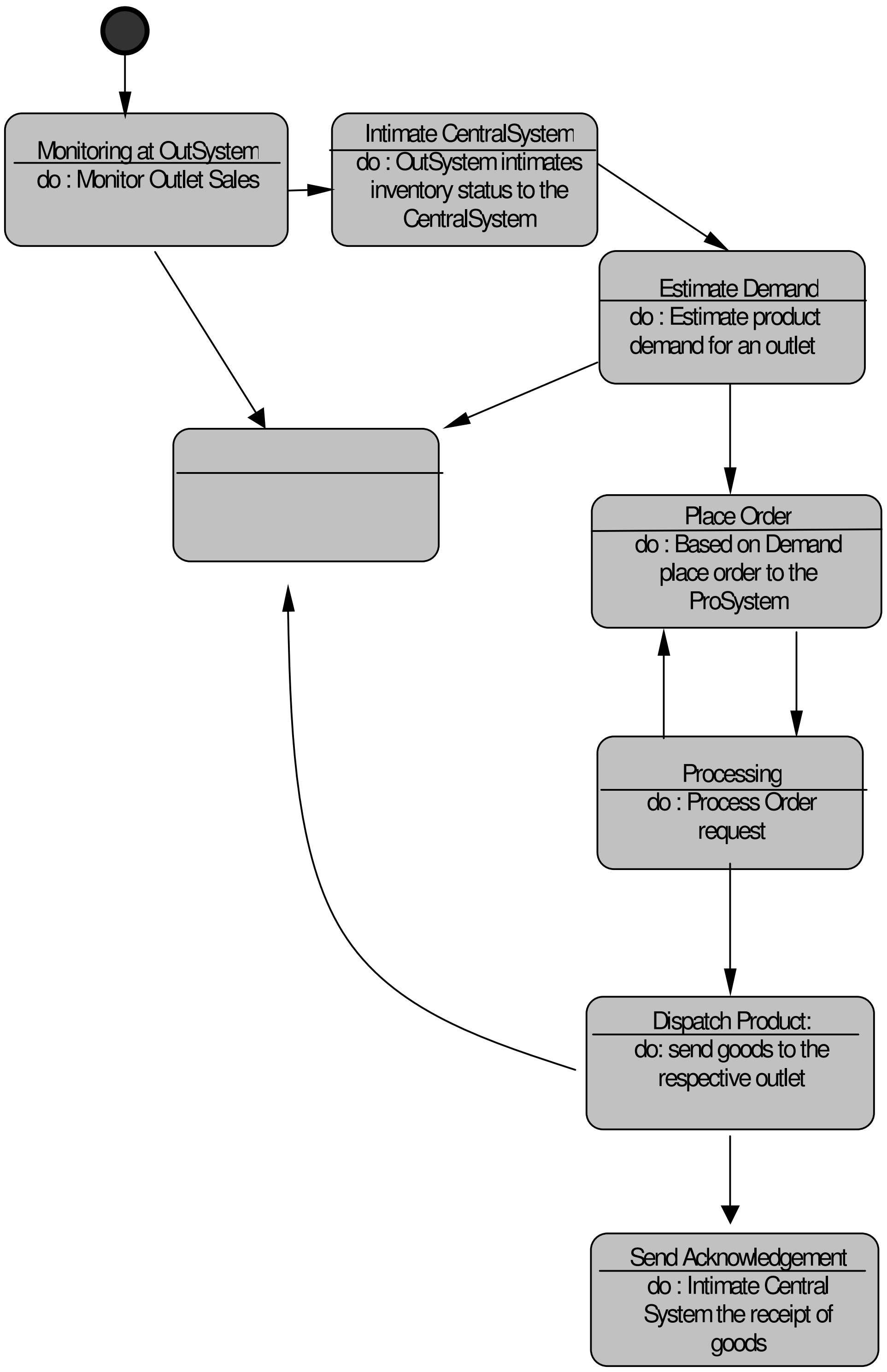
Use Case #5: Servicing product request.



Use case #6: Order receipt acknowledgement



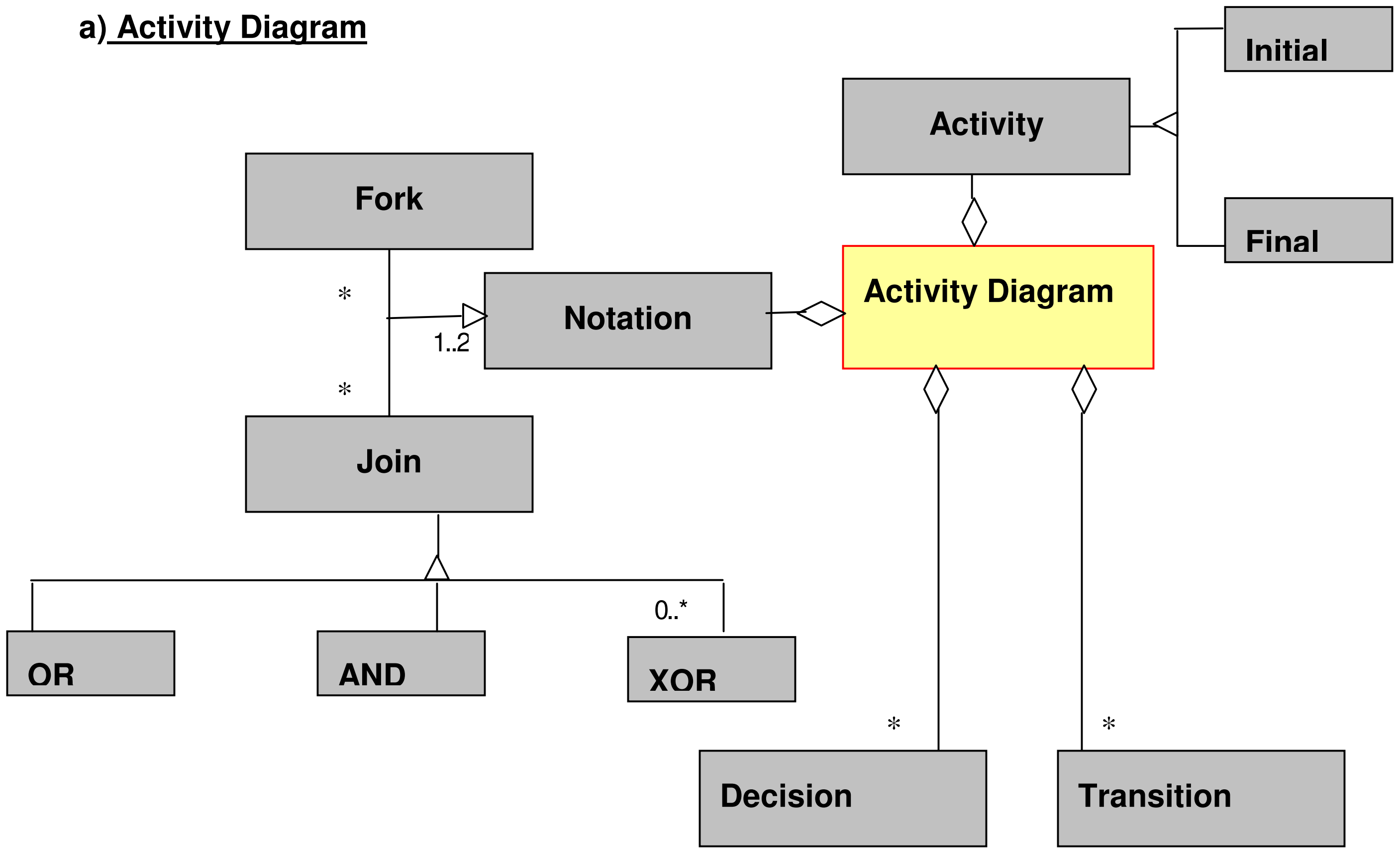
#1.3 State Transition Diagram



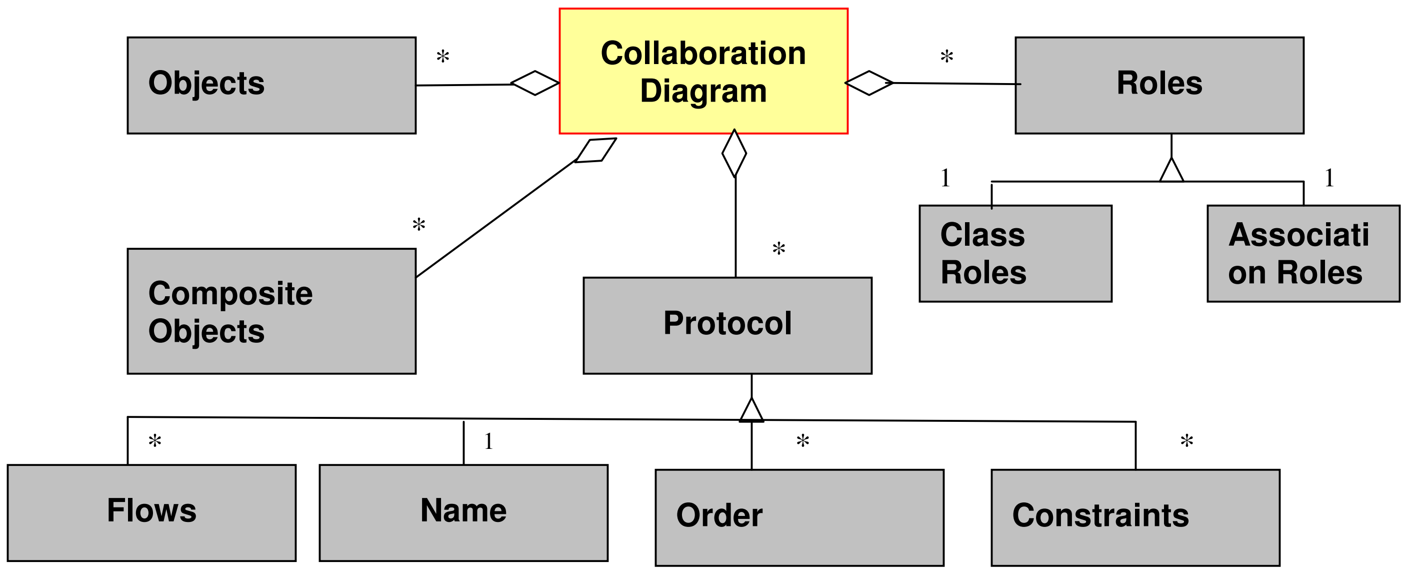
Monitoring at OutSystem
do : Monitor Outlet Sales
Intimate CentralSystem
do : OutSystem intimates inventory status to the CentralSystem
Estimate Demand
do : Estimate product demand for an outlet
Place Order
do : Based on Demand place order to the ProSystem
Processing
do : Process Order request
Dispatch Product:
do: send goods to the respective outlet
Send Acknowledgement
do : Intimate Central System the receipt of goods

#2 Metamodels

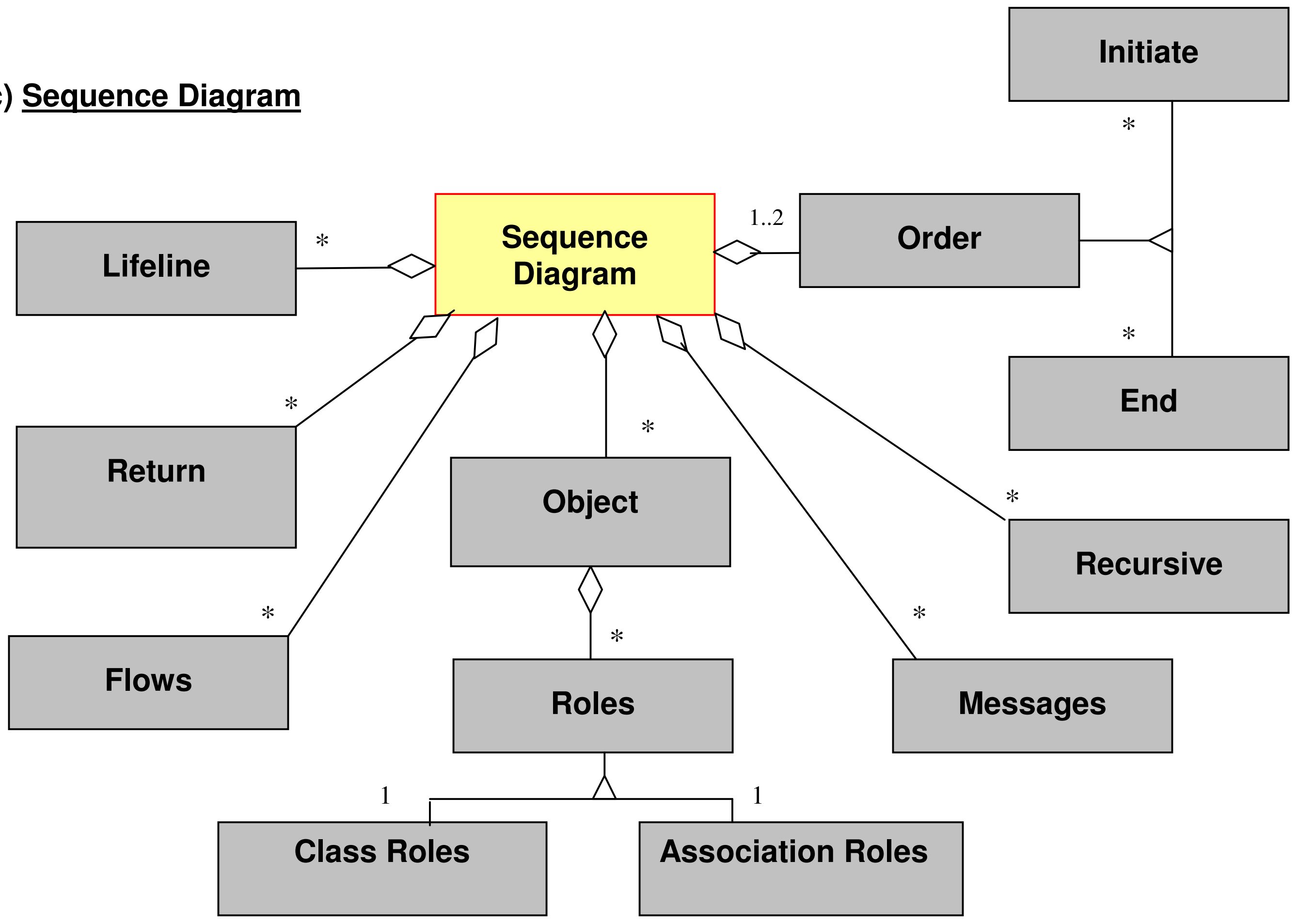
a) Activity Diagram



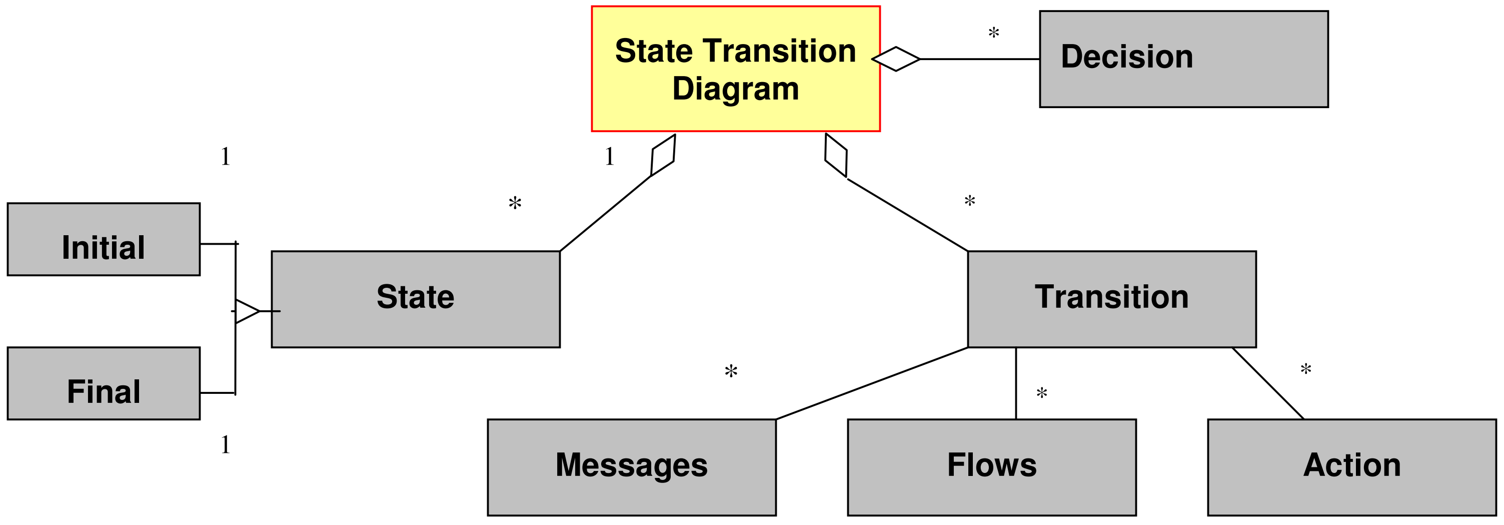
b) Collaboration Diagram



c) Sequence Diagram



c) State Transition Diagram



# 3 Value of OO design heuristics

Based on our experiences we felt the following heuristics to be useful in our exercises:

“All data should be hidden within its class” - The data elements that formed, for instance the class product and its subclasses perishable, non-perishable, automotive and electronics have data elements that are hidden within their class

“Users of a class must be dependent on its public interface, but a class should not be dependent on its users” - Using the stability model we have all the EBTs, BOs and IOs separated and hence the model is scalable which makes it independent of its users

“*Minimize the number of messages in the protocol of a class”* - The model is essentially based on a communication model, hence there is a tradeoff in this respect

“*Don’t put implementation details such as common-code private functions into the public interface of a class”* – After applying stability approaches this heuristic the common code private functions into the public interface of a class.

“Don’t clutter the public interface of a class with items that users of that class are not able to use or are not interested in using “– Using the stability model the Business (EBTs), the plan (BOs) and the execution of the plan (IOs) are created as frameworks that can be unplugged and scaled as and when the need arises.

“A class should capture one and only one key abstraction” – If you look at our model each and every class has only function or role associated with it.

“Spin off non-related information into another class (i.e., non-communicating behavior)” – Unlike our previous models the later ones have flushed out the unimportant ones which makes the system to be an efficient one.

As we learned in one of the lectures: “An OO design heuristic is a “rule-of-thumb”, not a rule and a design heuristic is something which makes a design “feel right” to us that we used as a guide to select the appropriate design choice from many possibilities

# 4 Rewritten Problem Statement

Background

Design Problem

Block Diagram

Use Cases

Background

With recession round the corner and profits dwindling, new technologies are a must to survive in the new economy. The validity of this statement holds good particularly in the FMCG (Fast Moving Consumer Goods) sector. The success of Wal-Mart and the decline of K-Mart render support to this statement.

A solution, if not an elixir, to this bottleneck is to apply *Supply Chain Management* techniques using new technologies. In simple terms Supply Chain Management System can be defined as follows: "*A set or a chain of actions or methods, which aids business activities among business partners, starting from the purchase of raw materials to the delivery of a finished product to the customer*". Information technology, particularly the Internet is opening a new gateway in this direction. To stay competitive and successful companies are embracing or rather have to embrace this concept.

In this article we come up with a system framework to carry out the operations of a supply chain management system effectively.

Problem Domain

Design of an Internet based system, which controls and manages the supply chain in a chain of FMCG retail stores.

FMCG markets rely greatly on information exchange, particularly using latest technologies, to stay competitive. Our system aims at providing an Internet based solution to implement traditional supply chain management techniques.

Components of IBSCMS:

1. OutSystem (Outlet – Supply Chain Management System)
2. CentralSystem (Central – Supply Chain Management System)
3. ProSystem (Provider– Supply Chain Management System)

OutSystem (Outlet – Supply Chain Management System):

This forms one end of the supply chain management system. The outlet represents the actual "point of sale" where the goods are transferred or sold out of the system. The OutSystem keeps track of the inventory at a particular outlet and there exists one OutSystem for each outlet. Each such OutSystem communicates with the CentralSystem to share information and to relay the present state of its inventory. The relay of this information is carried out on regular intervals, whenever the quantity of products reaches a threshold value. The updation is also done fortnightly or a chosen time interval, as decided by the administration

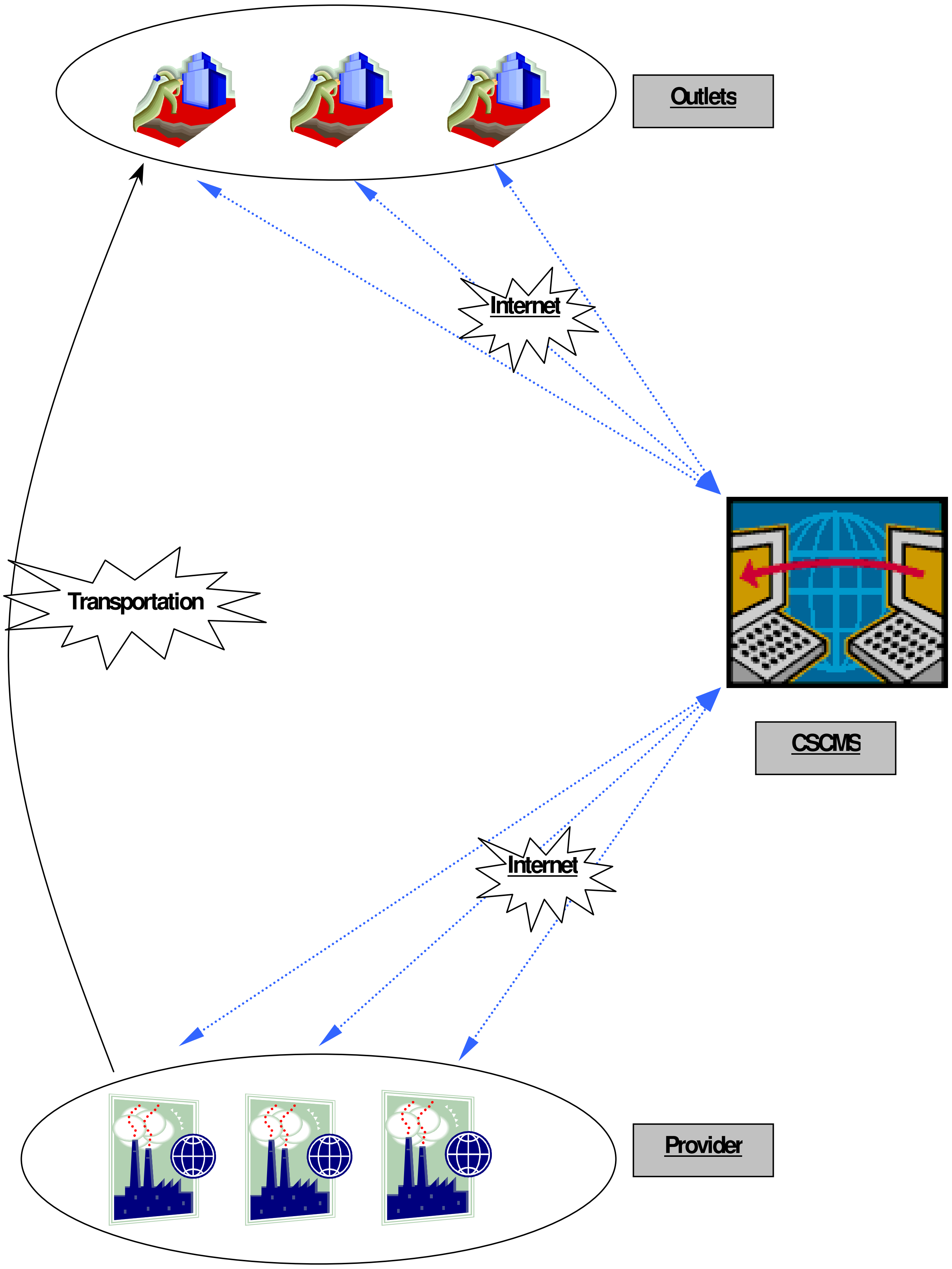
ProSystem (Provider – Supply Chain Management System):

This is the other end of the supply chain management system. This is the *supplier* or the *provider* end. The provider responds and caters to the requests of the CentralSystem and dispatches new inventory to the outlets as per the demand. The ProSystem also performs the same updations similar to the OutSystem. This helps to have a clear picture about the existing inventory status.

CentralSystem (Central – Supply Chain Management System):

This forms the core or the central System of IBSCMS. This setup is in charge of processing the information from different OutSystem and relaying requests to the appropriate ProSystem to send new inventory to the particular outlets. Processing of information includes the calculation of lead-time, the threshold level and the amount of inventory to be sent to the outlet. This forms the heart of the entire system and controls all the operations of the system.

Block Diagram



Use Cases

Use Case #1: Monitoring product sales.

Goal in Context: The OutSystem, viz. the supply chain system at the outlet, monitors product sales for changes and updates the Outlet – Database.

Actors: Customer, OutSystem, and Product.

Description: When these products are billed, the OutSystem gathers and updates the Outlet database for the product ID, quantity sold, and other related information about the purchase. A threshold quantity is defined in the system to carry out communication.

Use Case #2: OutSystem sending information.

Goal in Context: The OutSystem sends the appropriate inventory information to the CentralSystem periodically.

Actors: OutSystem, CentralSystem and dataHandler.

Description: The OutSystem keeps track of the all the products by updating the product information database each time a product is sold and checked out of the store. Periodically appropriate inventory information of all the products for an outlet is sent to the Central Supply Chain Management System, the CentralSystem. This helps the outlet store to keep the CentralSystem informed about the latest status of all the products in the outlet store and the products needed.

Use Case #3: CentralSystem processing incoming information.

Goal in Context: The CentralSystem processes the incoming information to take delivery decisions.

Actors: CentralSystem, dataHandler.

Description: The CentralSystem updates its Central – Database (CDB) and processes the incoming information about the latest inventory status of the products from each outlet. It then examines the product demand and comes up with the threshold level, which it uses to deduce when and how much inventory has to be sent to a particular outlet.

Use Case #4: CentralSystem placing order.

Goal in Context: The CentralSystem, based upon the product requirement at the outlet, places order for the respective quantity for the appropriate products, to the appropriate outlet.

Actors: CentralSystem, ProSystem and Product.

Description: The CentralSystem, after determining the current demand for each product at each outlet, places an order to the corresponding provider by delivering information which includes products required, quantity and the outlet location. After placing the order for the products, the CentralSystem updates the CDB and keeps track of the total orders placed for each product and for each outlet.

Use Case #5: Servicing product request.

Goal in Context: The ProSystem processes the product request submitted by the CentralSystem and dispatches the requested quantity of products to the corresponding outlet.

Actors: ProSystem and Order.

Description: The ProSystem processes the product request submitted by the CentralSystem. It services the order by setting up appropriate quantities for all the requested products.

Use case #6: Order receipt acknowledgement.

Goal in Context: The Outlet intimates the order receipt to the CentralSystem.

Actors: OutSystem and CentralSystem

Description: When the OutSystem receives the order it updates its local database with the information regarding the new inventory and also sends an acknowledgement to the informing it about the inventory received.