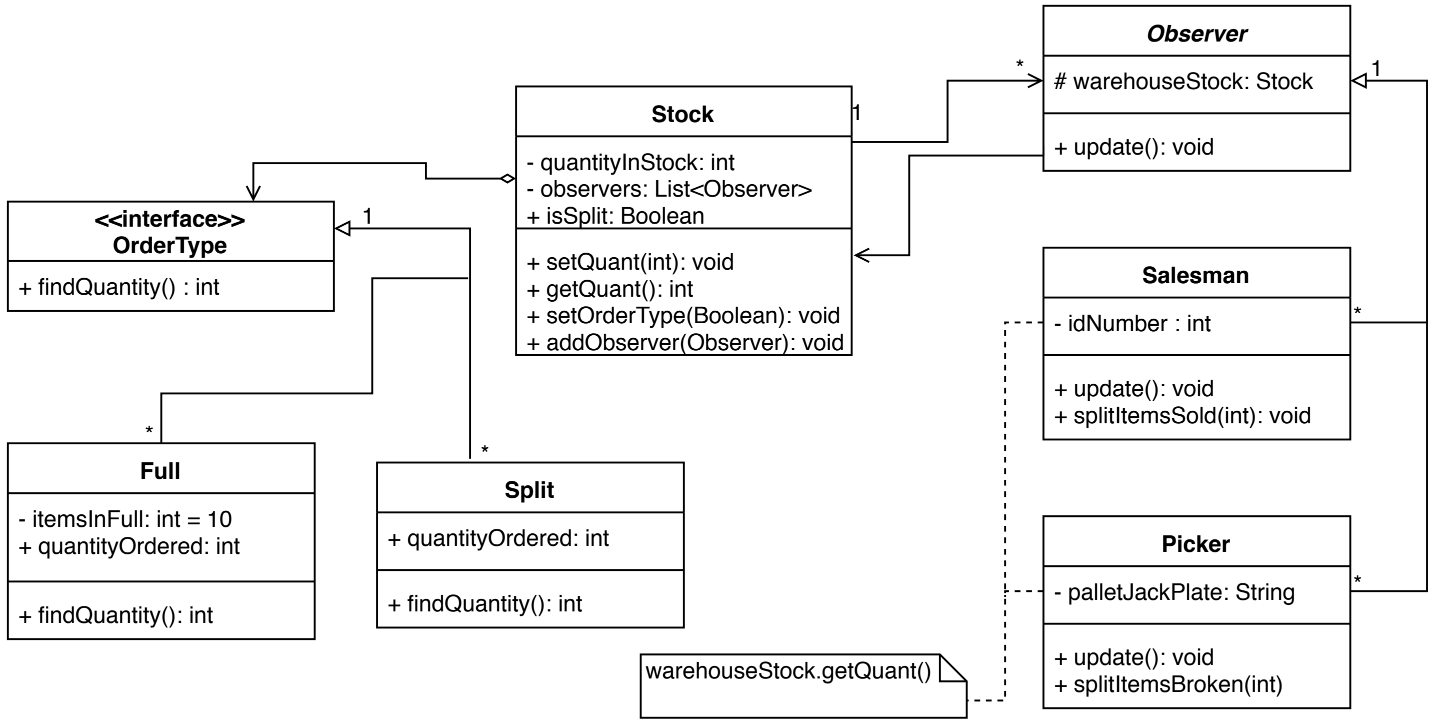
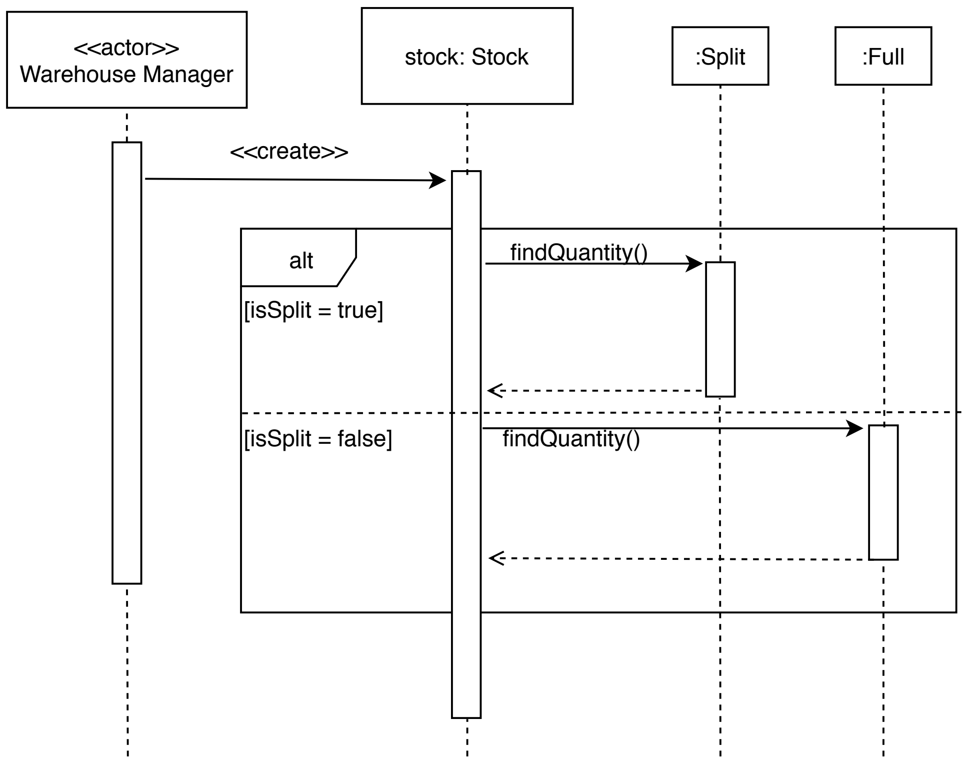
Alec VanderKolk
Assignment 3

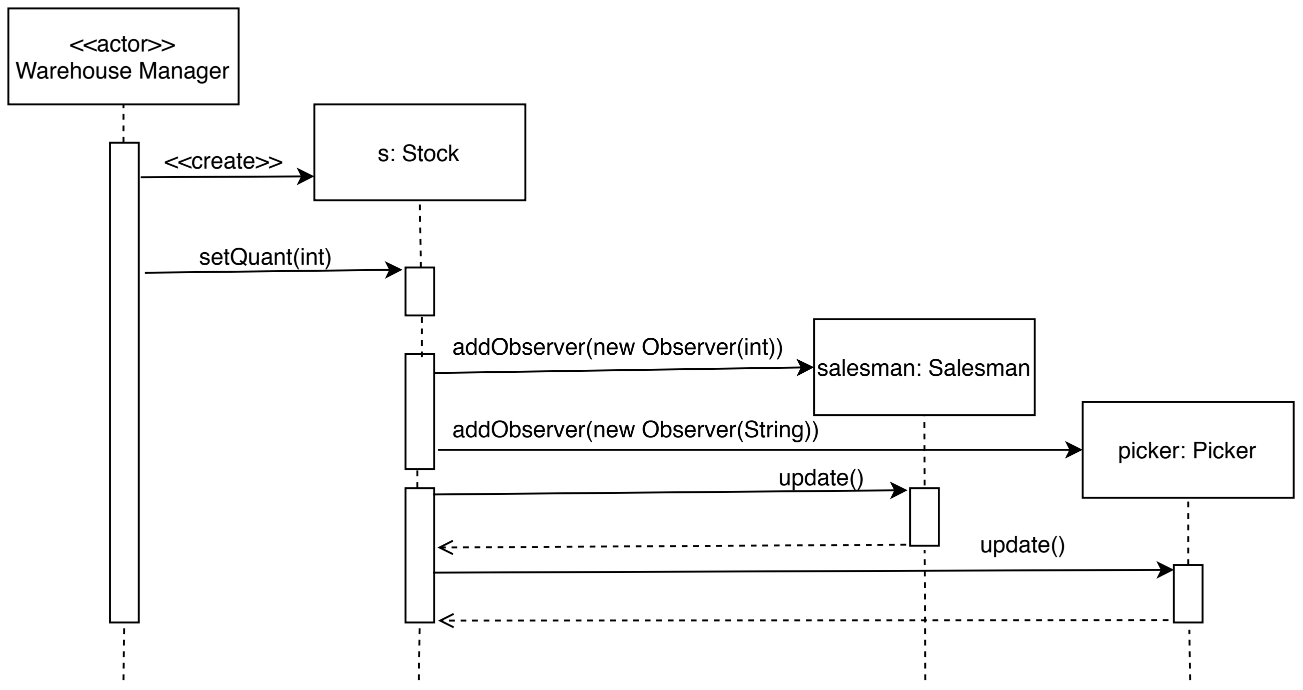
1a. The Stock class utilizes the strategy pattern and the observer pattern.



1b. Sequence diagram from perspective of strategy pattern:



1b (cont.). Sequence diagram from perspective of Observer pattern:

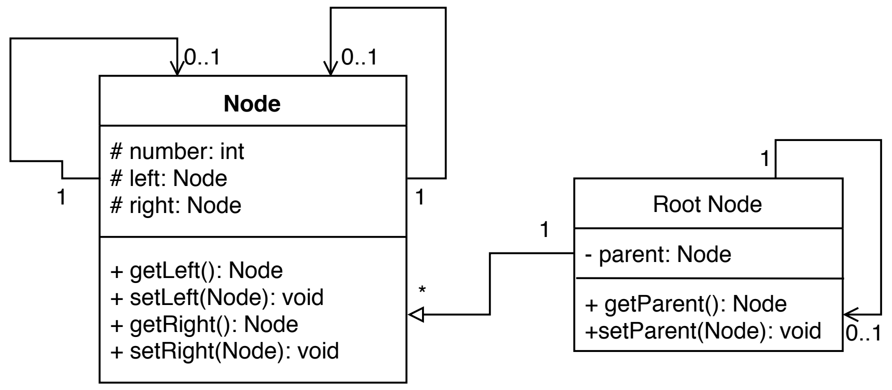


2a. previous focus factor = 32/45 = 0.71
estimated velocity = (45+15+12)*0.71=51.19
51.19 rounds down to **51 Story Points**

2b. Estimate focus factor using a value between 0.6 and 0.8. 0.6 applies to a more disorganized work space, and 0.8 applies to an efficient work place. 0.7 is used for most focus factor estimates.

2c. Estimate story points based on each team member's individual ability to complete the task. Each person determines how long it would take for them to complete the task alone. The mean of those numbers is taken then divided by the number of employees. The solution would be an estimate for number of story points. This is worse than poker I believe because the individuals will probably be considerably more efficient in teams than as individuals.

2d.



2e.

```
public class Node {
        protected int number;
        protected Node left;
        protected Node right;

        public Node(){
                number = 0;
                left = null;
                right = null;
        }
        public Node(int n){
                this.number = n;
                this.left = null;
                this.right = null;
        }

        public int getNumber() {
                return number;
        }

        public void setNumber(int number) {
                this.number = number;
        }

        public Node getLeft() {
                return left;
        }

        public void setLeft(Node left) {
                this.left = left;
        }

        public Node getRight() {
                return right;
        }

        public void setRight(Node right) {
                this.right = right;
        }
}
```

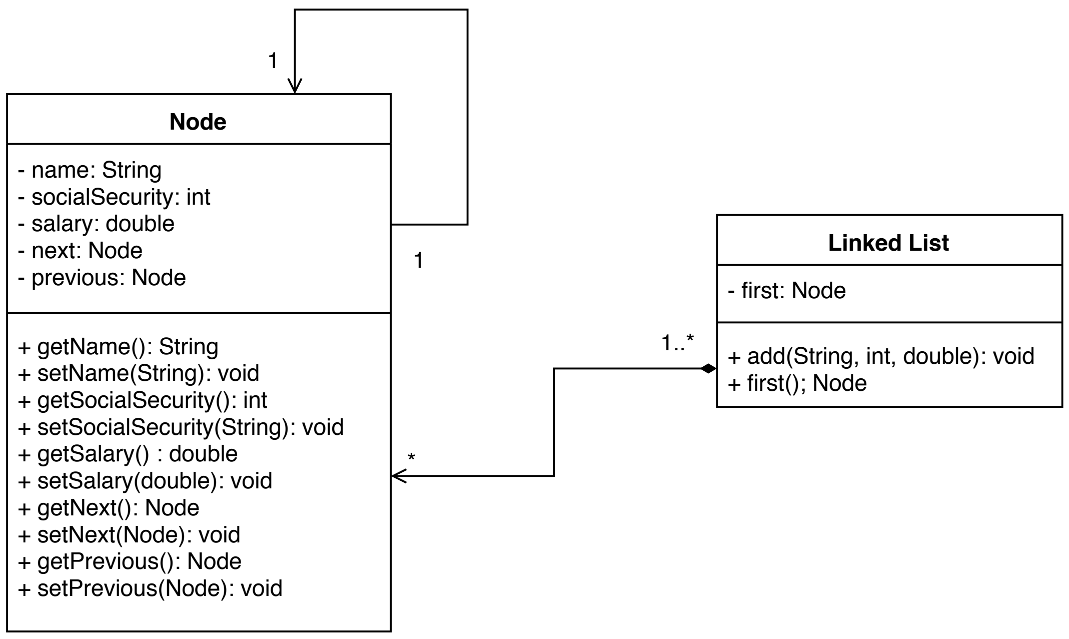
```
public class RootNode extends Node{
        private Node parent;

        public RootNode(int n, Node par){
                this.number = n;
                this.parent = par;
                this.left = null;
                this.right = null;
        }

        public Node getParent() {
                return parent;
        }

        public void setParent(Node parent) {
                this.parent = parent;
        }
}
```

2f.



2g.

```
public class Node {
	private String name;
	private int socialSecurity;
	private double salary;
	private Node next;
	private Node previous;

	public Node(String n, int ss, double sa){
		this.name = n;
		this.socialSecurity = ss;
		this.salary = sa;
		next = null;
		previous = null;
	}

	public String getName() {
		return name;
	}

	public void setName(String name) {
		this.name = name;
	}

	public int getSocialSecurity() {
		return socialSecurity;
	}

	public void setSocialSecurity(int socialSecurity) {
		this.socialSecurity = socialSecurity;
	}

	public double getSalary() {
		return salary;
	}

	public void setSalary(double salary) {
		this.salary = salary;
	}

	public Node getNext() {
		return next;
	}

	public void setNext(Node next) {
		this.next = next;
	}

	public Node getPrevious() {
		return previous;
	}

	public void setPrevious(Node previous) {
		this.previous = previous;
	}

}
```

```
public class LinkedList {
	private Node first;

	public LinkedList(){
		first = null;
	}

	public Node first(){
		return first;
	}

	public void add(String n, int s, double sal){
		Node add = new Node(n, s, sal);
		if(first==null){
			first = add;
		} else{
			first.setPrevious(add);
			add.setNext(first);
			first = add;
		}
	}
}
```