In [ ]:

```
#Clock_One.java
import java.rmi.Naming;
import java.time.LocalTime;
import java.time.format.DateTimeFormatter;

public class Clock_One {
    public static void main(String[] args) {
        try {
            ServerImpl s1=new ServerImpl(LocalTime.parse(args[1], DateTimeF
            Naming.rebind(args[0], s1);
        } catch (Exception e) {
            System.out.println("exception - "+e);
        }
    }
}
#MainClock.java
import java.rmi.Naming;
import java.rmi.RemoteException;
import java.time.LocalTime;
import java.time.format.DateTimeFormatter;

public class MainClock {
    public static void main(String[] args) throws RemoteException {
        try {
            String clock_one = "rmi://127.0.0.1/" + args[0];
            String clock_two = "rmi://127.0.0.1/" + args[1];
            String clock_three = "rmi://127.0.0.1/" + args[2];

            LocalTime serverTime = LocalTime.parse("03:00:00", DateTimeForm
            System.out.println("Server timr = " + serverTime.toString());
            ServerIntf c1 = (ServerIntf) Naming.lookup(clock_one);
            System.out.println("clock one time =  " + c1.getTime());
            ServerIntf c2 = (ServerIntf) Naming.lookup(clock_two);
            System.out.println("clock two time =  " + c2.getTime());
            ServerIntf c3 = (ServerIntf) Naming.lookup(clock_three);
            System.out.println("clock three time =  " + c3.getTime());

            long serverNano = serverTime.toNanoOfDay();
            long c1Nano = c1.getTime().toNanoOfDay() - serverNano;
            long c2Nano = c2.getTime().toNanoOfDay() - serverNano;
            long c3Nano = c3.getTime().toNanoOfDay() - serverNano;
            long avg = (c1Nano + c2Nano + c3Nano) / 4;

            c1.adjustTime(serverTime, avg);
            c2.adjustTime(serverTime, avg);
            c3.adjustTime(serverTime, avg);
            serverTime=serverTime.plusNanos(avg);
            System.out.println("clock one updated time =  " + c1.getTime()
            System.out.println("clock two updated time =  " + c2.getTime()
            System.out.println("clock three updated time =  " + c3.getTime(
            System.out.println("updated time"+serverTime);
        } catch (Exception e) {
            System.out.println("exception - "+e);
        }
    }
}

#ServerImpl.java

import java.rmi.RemoteException;
import java.rmi.server.UnicastRemoteObject;
```

```
import java.time.LocalTime;
import java.time.format.DateTimeFormatter;

public class ServerImpl extends UnicastRemoteObject implements ServerIntf{
    private LocalTime currentTime;
    ServerImpl(LocalTime currentTime) throws RemoteException {
        this.currentTime = currentTime;
    }
    public LocalTime getTime() throws RemoteException {
        return this.currentTime;
    }
    public void adjustTime(LocalTime serverTime, long timediff) throws Remo
        long serverNano = serverTime.toNanoOfDay();
        long adjusted = timediff + serverNano;
        LocalTime adjustedTime = LocalTime.ofNanoOfDay(adjusted);
        System.out.println("Adjusted time is: " + adjustedTime.format(DateT
        this.currentTime=adjustedTime;
    }
}
#ServerIntf.java

import java.rmi.Remote;
import java.rmi.RemoteException;
import java.time.LocalTime;

public interface ServerIntf extends Remote {
    public void adjustTime(LocalTime serverTime, long timediff) throws Remo
    public LocalTime getTime() throws RemoteException;
}

RunCommand:
terminal1:javac *.java,  rmiregistry
terminal2:java Clock_One c1 03:10:00
terminal3:java Clock_One c2 03:20:00
terminal4:java Clock_One c3 02:50:00
terminal5:java MainClock c1 c2 c3
```