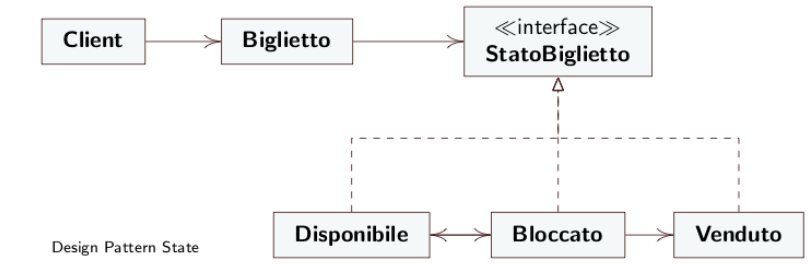

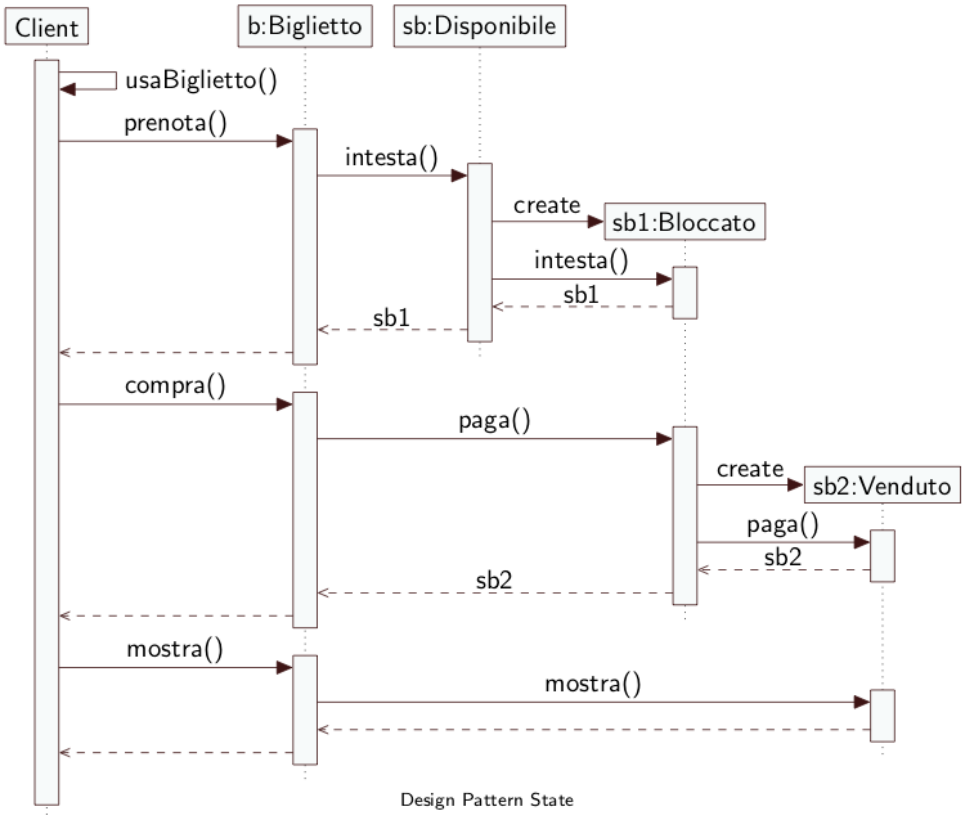


```java
// Codice Java che implementa il design pattern State

// StatoBiglietto e' uno State
public interface StatoBiglietto {
   public void mostra();
   public StatoBiglietto intesta(String s);
   public StatoBiglietto paga();
   public StatoBiglietto cancella();
}


// Disponibile e' un ConcreteState
public class Disponibile implements StatoBiglietto {
   @Override public void mostra() {    }

   @Override public StatoBiglietto intesta(String s) {
      System.out.println("DISP Cambia stato da Disponibile a Bloccato");
      return new Bloccato().intesta(s);
   }

   @Override public StatoBiglietto paga() {
```

```
        System.out.println("DISP Non si puo' pagare, bisogna prima intestarlo");
        return this;
    }

    @Override public StatoBiglietto cancella() {
        System.out.println("DISP Lo stato era gia' Disponibile");
        return this;
    }
}


// Bloccato e' un ConcreteState
public class Bloccato implements StatoBiglietto {
    private String nome;

    @Override public void mostra() {
        System.out.println("BLOC Nome: "+nome);
    }

    @Override public StatoBiglietto intesta(String s) {
        System.out.println("BLOC Inserito nuovo intestatario");
        nome = s;
        return this;
    }

    @Override public StatoBiglietto paga() {
        System.out.println("BLOC Cambia stato da Bloccato a Venduto");
        return new Venduto(nome).paga();
    }

    @Override public StatoBiglietto cancella() {
        System.out.println("BLOC Cambia stato da Bloccato a Disponibile");
        return new Disponibile();
    }
}


// Venduto e' un ConcreteState
import java.time.LocalDateTime;
public class Venduto implements StatoBiglietto {
    private final String nome;
    private LocalDateTime dataPagam;

    public Venduto(String n) {
        nome = n;
    }

    @Override public void mostra() {
        System.out.println("VEND Nome: " + nome);
    }
    @Override public StatoBiglietto intesta(String s) {
        System.out.println("VEND Non puo' cambiare il nome nello stato Venduto");
```

```
        return this;
    }
    @Override public StatoBiglietto paga() {
        if (dataPagam == null) {
            dataPagam = LocalDateTime.now();
            System.out.println("VEND Pagamento effettuato");
        } else
            System.out.println("VEND Il biglietto era gia' stato pagato");
        return this;
    }
    @Override public StatoBiglietto cancella() {
        System.out.println("VEND Non puo' cambiare stato da Venduto a Disponibile"
        return this;
    }
}


// Biglietto e' un Context
public class Biglietto {
    private String codice = "XYZ";
    private int prezzo = 100;

    private StatoBiglietto sb = new Disponibile();

    public void mostra() {
        System.out.println("Prezzo: " + prezzo + "  codice: " + codice);
        sb.mostra();
    }

    public void prenota(String s) {
        sb = sb.intesta(s);
    }

    public void cancella() {
        sb = sb.cancella();
    }

    public void compra() {
        sb = sb.paga();
    }
}


// Classe con il main che usa il Context
public class Client {
    private static Biglietto b = new Biglietto();
    public static void main(String[] args) {
        usaBiglietto();
    }

    private static void usaBiglietto() {
        b.prenota("Mario Tokoro");
```

```
        b.compra();
        b.mostra();
    }

    private static void nonUsaOk() {
        b.compra();
        b.cancella();
        b.prenota("Mario Biondi");
    }
}
```

Output dell'esecuzione

```
DISP Cambia stato da Disponibile a Bloccato
BLOC Inserito nuovo intestatario
BLOC Cambia stato da Bloccato a Venduto
VEND Pagamento effettuato
Prezzo: 100  codice: XYZ
VEND Nome: Mario Tokoro
```