

# Travel to the Moon

## 1 Analisi dei Requisiti

### 1. Requisiti di Crociera

- 1.1 codice: Stringa
- 1.2 inizio: Data
- 1.3 fine: Data
- 1.4 associata alla Nave utilizzata
- 1.5 associata a un solo Itinerario
- 1.6 tipologie:{disj}
  - 1.6.1 Luna di Miele
    - 1.6.1.1 tipologia: Tipologia
  - 1.6.2 Famiglia
    - 1.6.2.1 bamb: Booleano

### 2. Requisiti di Nave

- 2.1 nome: Stringa
- 2.2 comfort: 3..5
- 2.3 capienza: Intero>0

### 3. Requisiti di Destinazione

- 3.1 nome: Stringa
- 3.2 continente: Continente
- 3.3 associate a diversi Posti da vedere
- 3.4 divise in:{}
  - 3.4.1 Romantiche
  - 3.4.2 Divertenti

### 4. Requisiti di Itenerario

- 4.1 nome: Stringa
- 4.2 associato ad una sequenza ordinata di Destinazioni sapendo:
  - 4.2.1 dataArrivo: OraTappa
  - 4.2.2 dataPartenza: OraTappa

### 5. Requisiti su Posto

- 5.1 nome: Stringa
- 5.2 descrizione: Stringa
- 5.3 fasciaOraria: FasciaOra [1..\*]

### 6. Requisiti di Cliente

- 6.1 nome: Stringa
- 6.2 cognome: Stringa
- 6.3 età: Intero>0
- 6.4 indirizzo: Indirizzo
- 6.5 associato alle Prenotazioni che effettua

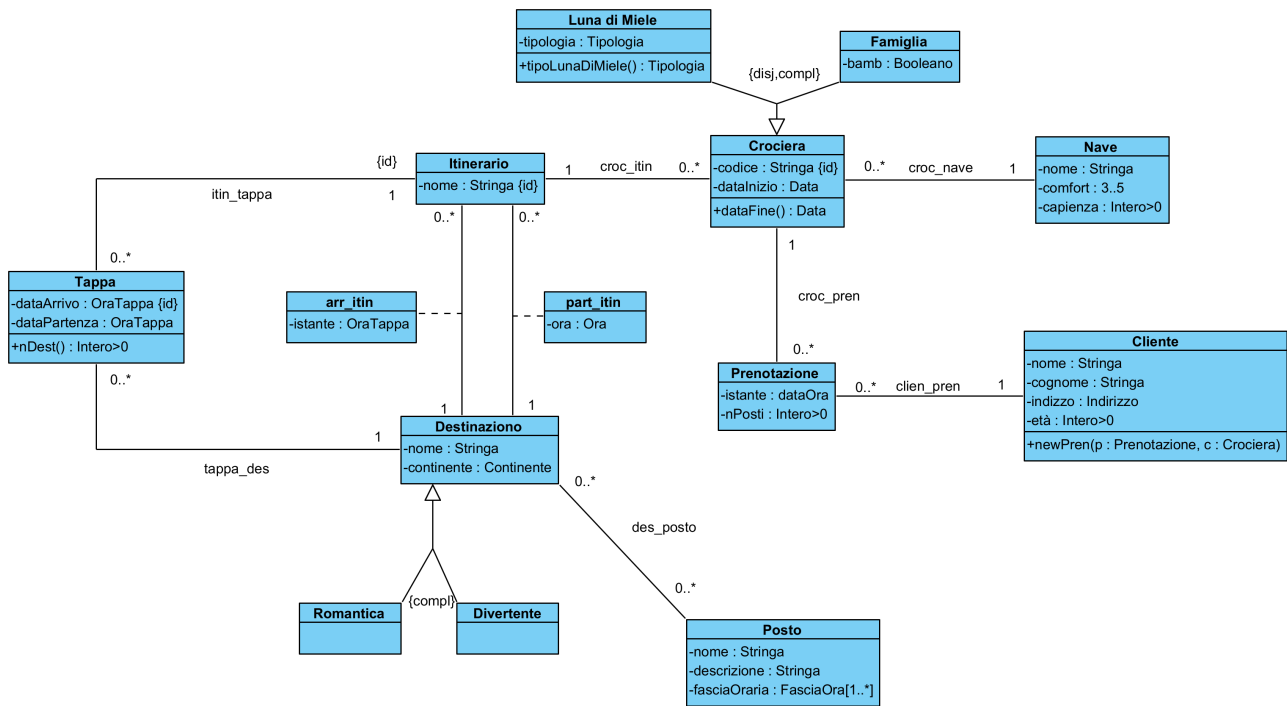
## 7. Requisiti di Prenotazioni

- 7.1 istante: DataOra
- 7.2 nPosti: Intero>0
- 7.3 associato alla Crociera prenotata

## 2 Tipi di Dato

1. Continente: {Europa, Asia, Sud America, Nord America, Africa, Oceania, Antartide}
2. OraTappa: (giorno=Intero>0, orario=Ora)
  - 2.1  $< (x:\text{OraTappa}, y:\text{OraTappa})$ : **Booleano**
    - Pre-condizioni:
    - Post-condizioni:
      - $\text{result} = (x.\text{giorno} < y.\text{giorno}) \vee (x.\text{giorno} = y.\text{giorno} \wedge x.\text{orario} < y.\text{orario})$
3. Giorno: {Lunedì, Martedì, Mercoledì, Giovedì, Venerdì, Sabato, Domenica}
4. FasciaOra: (giorno=Giorno, oraInizio=Ora, oraFine=Ora)
5. Indirizzo: (via=Stringa, civico=Intero>0)
6. Tipologia: {Tradizionale, Alternativa}

### 3 Diagramma UML



## 4 Specifica delle Classi

### 1. Specifica della classe Crociera:

**fine(): Data**

- Pre-condizioni:
- Post-condizioni:
  - il result si calcola:
    - \* Sia i:Itinerario tale che (this,i):croc\_itin
    - \* Sia x:OraTappa il valore "istante" del link (i,y):arr\_itin
    - \*  $result = this.inizio + x.giorno$

**tipoLunaDiMiele():Tipologia**

- Pre-condizioni:
  - this:LunaDiMiele
- Post-condizioni:
  - Sia i:Itinerario tale che (i, this):croc\_itin
  - Sia T l'insieme delle t:Tappa tali che (i, t):itin\_tappa
  - Sia D l'insieme delle d:Destinazione tali che (d,t):dest\_tappa
  - Sia DR l'insieme delle d:Romantica in D
  - Sia DD l'insieme delle d:Divertente in D
  - $result = Tradizionale$  se  $|DR| \geq |DD|$  altrimenti  $result=Alternativa$

**[V.Crociera.date]**

Per ogni c:Crociera:

$c.inizio \leq c.fine$

**[V.Crociera.capienza\_max]**

Per ogni c: Crociera :

- Sia P l'insieme delle p:Prenotazione tali che (c, p):croc\_pren
- Sia N la somma dei valori "nPosti" degli elementi di P

Per n:Nave tale che (this, n):croc\_nave:  $N < n.capienza$

## 2. Specifica della classe Itinerario:

[V.Itinerario.arrivo\_dopo\_ultima\_tappa]

Per ogni  $i$ :Itinerario:

- Sia  $T$  l'insieme delle  $t$ :Tappa tali che  $(i,t):itin\_tappa$
- Sia  $x$ :OraTappa il valore "istante" del link  $(i,...):arr\_itin$

Per ogni  $t$  in  $T$ :  $t.dataPartenza < x$

[V.Itinerario.prima\_tappa\_dopo\_partenza]

Per ogni  $i$ :Itinerario:

- Sia  $T$  l'insieme delle  $t$ :Tappa tali che  $(i,t):itin\_tappa$  e  $t.dataArrivo.giorno=1$
- Sia  $x$ :OraTappa il valore "istante" del link  $(i,...):arr\_itin$

Per ogni  $t$  in  $T$ :  $t.dataArrivo.orario > x$

[V.Itinerario.arrivo\_dopo\_partenza\_se\_senza\_tappe]

Per ogni  $i$ :Itinerario:

- Sia  $T$  l'insieme delle  $t$ :Tappa tali che  $(i,t):itin\_tappa$
- Sia  $x$ :Ora il valore "istante" del link  $(i,...):part\_itin$
- Sia  $y$ :OraTappa il valore "istante" del link  $(i,...):arr\_itin$

Se  $|T|=0$ :  $x < y.orario$

## 3. Specifica della classe Tappa:

nDest():Intero > 0

- Pre-condizioni:
- Post-condizioni:
  - il result si calcola:
    - \* Sia  $i$ :Itinerario tale che  $(this,i):itin\_tappa$
    - \* Sia  $T$  l'insieme delle  $t$ :Tappa tali che  $(i,t):itin\_tappa$  e  $t.arrivo < this.arrivo$
    - \*  $result = |T|+1$

[V.Tappa.date]

Per ogni  $t$ :Tappa:  $t.arrivo < t.ripartenza$

## 4. Specifica della classe Posto:

[V.Posto.fasce\_orarie\_sovrapposte]

Sia  $F$  l'insieme delle  $this.fasciaOraria$

Per ogni coppia  $f_1, f_2$  in  $F$  se  $f_1.giorno = f_2.giorno$ :

$(f_1.oraFine \leq f_2.oraInizio) \vee (f_1.oraInizio \geq f_2.oraFine)$

5. **Specifica della classe Cliente:**

**newPren(p:Prenotazione, c:Crociera)**

- Pre-condizioni:
- Post-condizioni:
  - crea un link (p, this):clien\_pren
  - crea un link (p, c):croc\_pren