Travel to the moon

1 Requisiti

I dati di interesse per il sistema sono

1. Requisiti sulle **crociere**:
    - 1.1. codice
    - 1.2. data di inizio
    - 1.3. data di fine
    - 1.4. nave utilizzata (v. req. 2)
    - 1.5. itinerario (v. req. 4)
    - 1.6. il tipo, uno tra:
        - 1.6.1. luna di miele, di cui interessa:
            - 1.6.1.1. sottotipo, uno tra:
                - 1.6.1.1.1. tradizionali
                - 1.6.1.1.2. alternative
        - 1.6.2. per famiglie, di cui interessa:
            - 1.6.3. se adatte ai bambini (booleano)

2. Requisiti sulle **navi**:
    - 2.1. nome
    - 2.2. comfort (3..5)
    - 2.3. capienza

3. Requisiti sulle **destinazioni**:
    - 3.1. nome
    - 3.2. continente
    - 3.3. posti da vedere (v. req. 5)
    - 3.4. tipo, almeno uno tra:
        - 3.4.1. romantico
        - 3.4.2. divertente

4. Requisiti sugli **itinerari**:
    - 4.1. sequenza ordinata di elementi, di cui interessa:
        - 4.1.1. porto (v. req. 3)
        - 4.1.2. arrivo:
            - 4.1.2.1. il numero d'ordine del giorno (rispetto alla data di inizio della crociera)
            - 4.1.2.2. ora
        - 4.1.3. ripartenza
            - 4.1.3.1. il numero d'ordine del giorno (rispetto alla data di inizio della crociera)

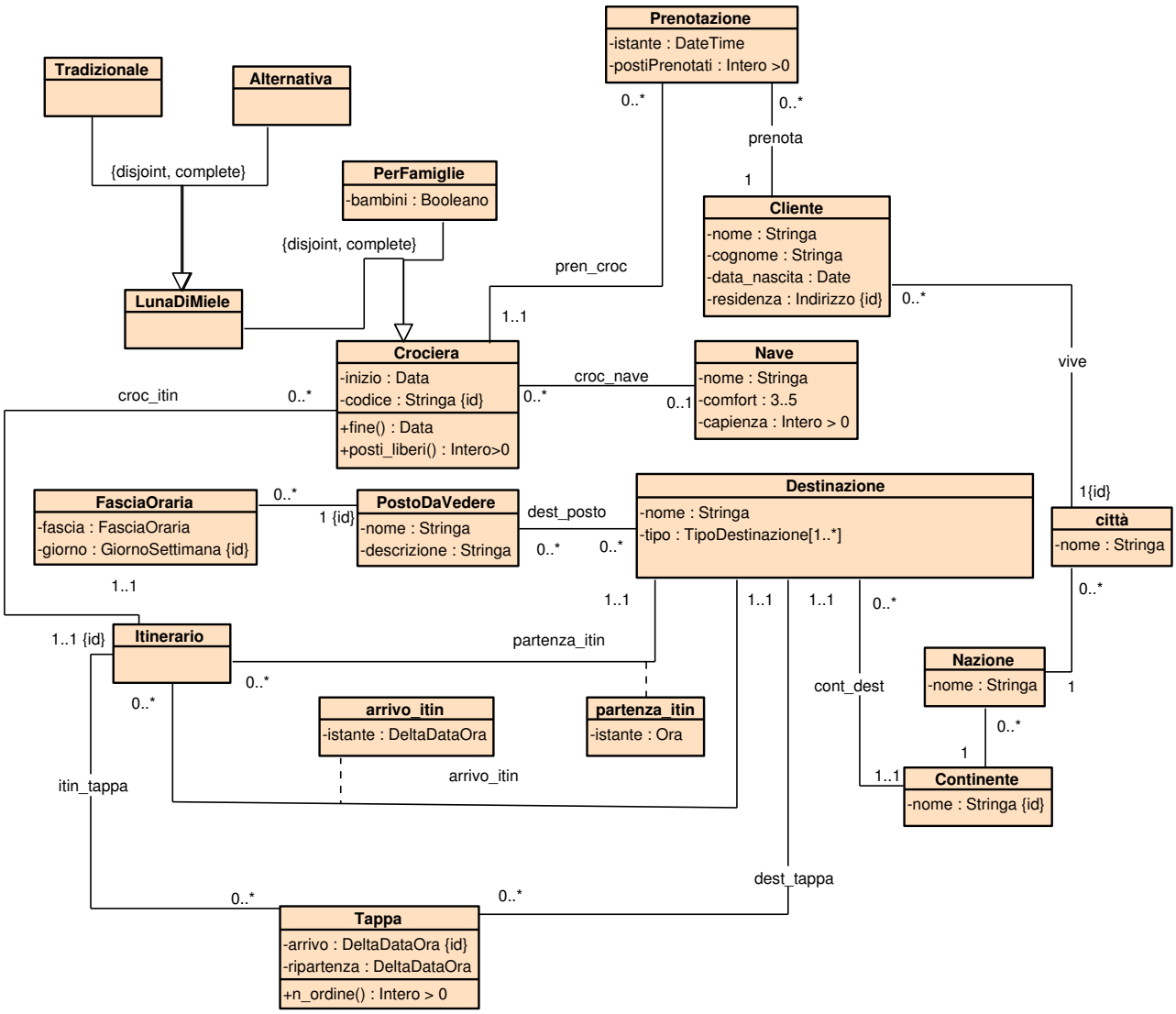
4.1.3.2. ora

5. Requisiti sui **posti da vedere**:
   5.1. nome
   5.2. descrizione
   5.3. orari di apertura, nella forma di una mappa che associa ad ogni giorno della settimana (lunedì, ..., domenica) un insieme di fasce orarie, dove ogni fascia oraria è definita in termini di una coppia di orari

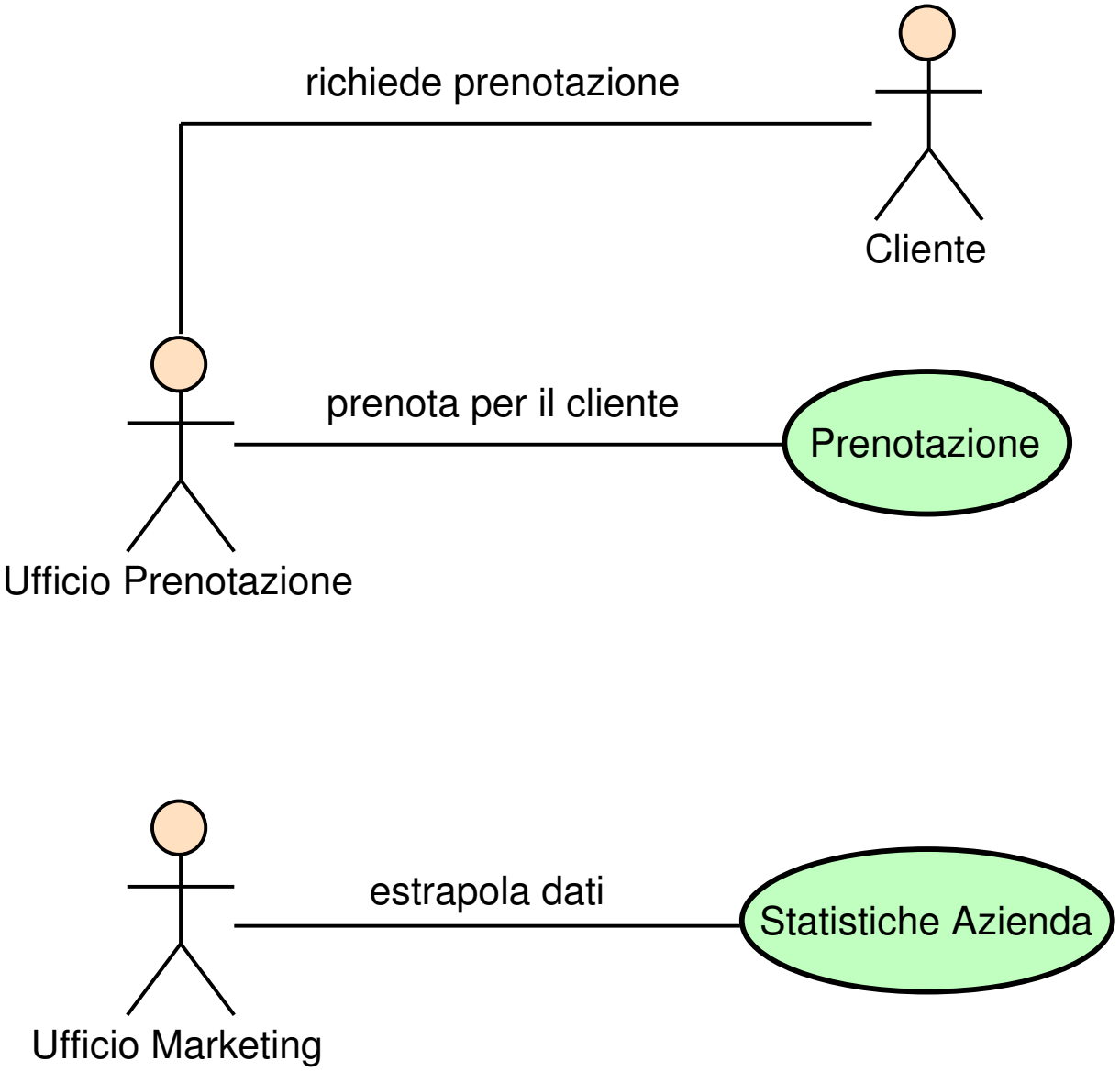
6. Requisiti sulle **prenotazioni**:
   6.1 data ed ora della prenotazione.
   6.2 cliente coinvolto.
   6.3 crociera coinvolta
   6.4 numero posti prenotati

7. Requisiti sui **clienti**:
   7.1 nome.
   7.2 cognome.
   7.3 data di nascita.
   7.4 indirizzo.

2 UML


Prenotazione
-istante : DateTime
-postiPrenotati : Intero >0
Tradizionale
Alternativa
{disjoint, complete}
PerFamiglie
-bambini : Booleano
{disjoint, complete}
LunaDiMiele
0..*
0..*
prenota
1
Cliente
-nome : Stringa
-cognome : Stringa
-data_nascita : Date
-residenza : Indirizzo {id}
0..*
pren_croc
1..1
Crociera
-inizio : Data
-codice : Stringa {id}
+fine() : Data
+posti_liberi() : Intero>0
croc_nave
0..*
0..1
Nave
-nome : Stringa
-comfort : 3..5
-capienza : Intero > 0
vive
croc_itin
0..*
FasciaOraria
-fascia : FasciaOraria
-giorno : GiornoSettimana {id}
0..*
1 {id}
PostoDaVedere
-nome : Stringa
-descrizione : Stringa
dest_posto
0..*
0..*
Destinazione
-nome : Stringa
-tipo : TipoDestinazione[1..*]
1{id}
città
-nome : Stringa
1..1
1..1 {id}
Itinerario
1..1
1..1
1..1
0..*
0..*
partenza_itin
0..*
arrivo_itin
-istante : DeltaDataOra
partenza_itin
-istante : Ora
arrivo_itin
cont_dest
Nazione
-nome : Stringa
1
0..*
1
Continente
-nome : Stringa {id}
1..1
itin_tappa
dest_tappa
0..*
0..*
Tappa
-arrivo : DeltaDataOra {id}
-ripartenza : DeltaDataOra
+n_ordine() : Intero > 0

3 Use Case


richiede prenotazione
Cliente
prenota per il cliente
Prenotazione
Ufficio Prenotazione
estrapola dati
Statistiche Azienda
Ufficio Marketing

4 Specifiche

4.1 Specifica dei Tipi di Dato

Indirizzo = { via:Stringa, civico : Intero>0, CAP : Intero>0 }

GiornoSettimana = {Lunedì...,Domenica}

FasciaOraria = (inizio_fascia : Ora, fine_fascia : Ora)

TipoDestinazione = romantica, divertente

DeltaDataOra = (
    giorno: Intero > 0,
    orario: Ora
)

Operazioni del tipo di dato DeltaDataOra:

$<$ (x:DeltaDataOra, y:DeltaDataOra) : Boolano
    pre: nessuna
    postcondizioni:
    result = true se e solo se:
    (x.giorno $<$ y.giorno)
    oppure
    (x.giorno = y.giorno e x.orario $<$ y.orario)

4.2 Specifica delle Classi

4.2.1 Crociera

Operazione `posti_liberi () : Intero` ≥ 0

- *pre-condizioni* : L'oggetto *this* deve essere coinvolto in almeno un link di tipo croc_nave con un oggetto di tipo Nave.
- *post-condizioni* : Sia P l'insieme delle prenotazioni p:Prenotazione tale che $\exists$ un link di tipo croc_pren con *this*. Sia $n = \sum_{p \in P} p$.postiPrenotati, sia b:Nave l'oggetto di tipo Nave per il quale esiste il link (this,b) di tipo croc_nave. result = b.capienza-n.

Operazione `fine() : Data`

- *pre-condizioni* : nessuna
- *post-condizioni* : Sia i:Itinerario tale che (this, i):croc_itin.

  Sia x:DeltaDataOra il valore dell'attributo "istante" dell'unico link di assoc. arrivo_itin in cui "i" è coinvolto.

  result = this.inizio + x.giorno "giorni".

Vincoli esterni :

[V.Crociera.date]
Per ogni oggetto c:Crociera deve essere: c.inizio $\leq$ c.fine

4.2.2 Itinerario

Vincoli esterni :

[V.Itinerario.arrivo_dopo_ultima_tappa]
Per ogni i:Itinerario siano:

- T l'insieme delle istanze 't' di classe Tappa tali che (i,t):itin_tappa
- x il valore dell'attributo 'istante' dell'unico link di associazione arrivo_itin in cui 'i' è coinvolto

Per ogni t:T, deve essere: t.ripartenza $<$ x.

[V.Itinerario.prima_tappa_dopo_partenza]
Per ogni i:Itinerario siano:

- T l'insieme delle istanze 't' di classe Tappa tali che (i,t):itin_tappa e t.arrivo.giorno = 1
- x il valore dell'attributo 'istante' dell'unico link di associazione partenza_itin in cui 'i' è coinvolto

Per ogni t:T, deve essere t.arrivo.orario $>$ x.

[V.Itinerario.arrivo_dopo_partenza_se_senza_tappe]
Per ogni i:Itinerario tale siano:

- T l'insieme delle istanze 't' di classe Tappa tali che (i,t):itin_tappa
- x il valore dell'attributo 'istante' dell'unico link di associazione partenza_itin in cui 'i' è coinvolto
- y il valore dell'attributo 'istante' dell'unico link di associazione arrivo_itin in cui 'i' è coinvolto

se T = vuoto e y.giorno = 1, allora deve essere x $<$ y.orario.
Per ogni t:T, deve essere t.arrivo.orario $>$ x.

4.2.3 Tappa

Operazione `n_ordine() : Int > 0`

- *pre-condizioni* : nessuna
- *post-condizioni* : No side-effect. Sia i l'itinerario della tappa this; Sia TappePrecedenti l'insieme degli oggetti t:Tappa tali che:

- (i,t): itin_tappa

- t.arrivo < this.arrivo;

result = |TappePrecedenti| + 1.

Vincoli esterni :

[V.Tappa.date]
Per ogni istanza t:Tappa, deve essere: t.arrivo < t.ripartenza

4.2.4 Prenotazioni

Vincoli esterni :

[V.Prenotazioni.date]
Per ogni istanza p:Prenotazione, ed ogni link r fra p:Prenotazione e c:Crociera, deve essere: p.data_prenotazione<c.indirizzo

4.2.5 Fascia Oraria

[V.FasciaOraria.inizio_fascia_fine_fascia]
Per ogni f:FasciaOraria, f.fascia.inizio_fascia < f.fascia.fine_fascia

[V.FasciaOraria.sovrapposizione_fasce]
Per ogni f:FasciaOraria con |f.fascia| > 1 , presi f.fascia = x e f.fascia' = y , non esiste y tale che: (x.inizio_fascia ≤ y.inizio_fascia < x.fine_fascia)

4.3 Specifica degli Use-Case

4.3.1 Prenotazione

Operazione
`effettua_prenotazione (cl:Cliente, cr:Crociera, nPosti : Intero>0) : Prenotazione`

- *pre-condizioni* : cr.Inizio > now ∧ nPosti ≤ cr.posti_liberi().
- *post-condizioni* : Crea oggetto p : *Prenotazione* tale che:
  - p.istante = now
  - p.postiPrenotati = nPosti
  - Viene creato un link di tipo croc_pren fra cr e p
  - Viene creato un link di tipo prenota fra p e cl.

  result = p

4.3.2 Statistiche Azienda

Operazione `media_dest_esotica(`$d_i : Data, d_f : Data$`) : Reale`

pre: nessuno

post: Sia $\mathcal{C}$ l'insieme formato da ogni $c : Crociera$ tale che $\exists (p, c)$ con $p : Prenotazione$ e $d_i \leq p.istante \leq d_f$.
Sia $\mathcal{M}$ l'insieme contente le $c \in \mathcal{C}$ tali che $\exists\ i : Itinerario, d : Destinazione, t : Tappa$ con $[(c, i) \wedge (i, d)] \vee [(c, i) \wedge (i, t) \wedge (t, d)]$ e $cont : Continente | (d, cont) \wedge cont.nome \neq$ "$Europa$".

Se $\mathcal{M} \neq \emptyset \Rightarrow \mathcal{P} := \{c : Cliente | \exists (m, p) \wedge (p, c)$ con $m \in \mathcal{M}$ e $p : Prenotazione\}$.

$$result = (\sum_{c \in \mathcal{P}} now - c.data\_nascita)/|\mathcal{P}|$$

altrimenti $result = -1$

$Con\ (\alpha, \beta) \longrightarrow link\ tra\ \alpha\ e\ \beta.$

Operazione `percentuale_gettonate(`$d_i : Data, d_f : Data$`): Reale`

pre: nessuna

post: Sia $\mathcal{L} := \{l : LunaDiMiele |\quad d_i \leq l.inizio \leq d_f\}$ e
$\mathcal{F} := \{f : PerFamiglie |\quad d_i \leq f.inizio \leq d_f\}$.
Sia $\varphi(x)$:= proprietà di $x : Destinazione$ vera $\Leftrightarrow \exists$ almeno 10 link del tipo : $(x, i)^* \wedge (i, l)$ oppure almeno 15 del tipo: $(x, i)^* \wedge (i, f)$ con $i : Itinerario$,$l \in \mathcal{L}$ e $f \in \mathcal{F}$.

Consideriamo $\mathcal{D}$ l'insieme di tutte le destinazioni raggiunte tra d_i e d_f, $\mathcal{D}'$ l'insieme di tutte le destinazioni per cui è vera $\varphi(x)$.

$$result = \frac{|\mathcal{D}'|}{|\mathcal{D}|} \cdot 100$$

$Con\ (x, i)^* \rightarrow [(x, i) \vee ((x, t) \wedge (t, i))]$