

TuTubi

1 Requisiti

1. Utente

- 1.1 nome
- 1.2 data iscrizione
- 1.3 playlist create
- 1.4 playlist visualizzate
- 1.5 video pubblicati
 - 1.5.1 istante pubblicazione
- 1.6 video visualizzati
 - 1.6.1 data ed ora visualizzazione
 - 1.6.2 valutazione video
 - 1.6.3 commento al video
 - 1.6.3.1 data ed ora commento

2. Playlist

- 2.1 nome
- 2.2 data creazione
- 2.3 pubblica o privata?
- 2.4 video coinvolti
 - 2.4.1 ordine del video

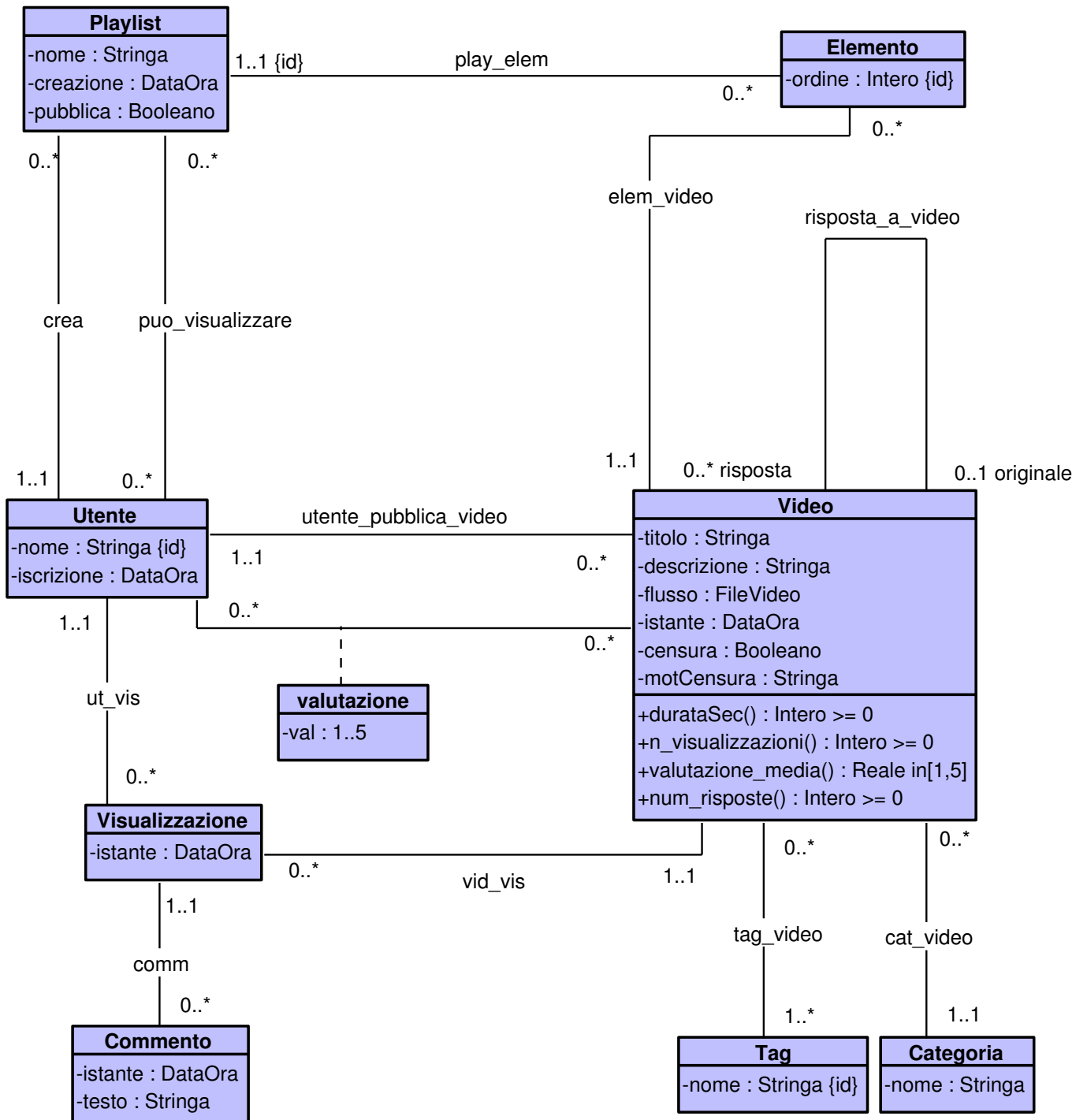
3. Video

- 3.1 titolo
- 3.2 commenti
 - 3.2.1 data ed ora commento
- 3.3 descrizione
- 3.4 file video
- 3.5 istante pubblicazione
- 3.6 video censurato?
 - 3.6.1 motivazione censura
- 3.7 statistiche
 - 3.7.1 durata del video in secondi
 - 3.7.2 numero di visualizzazioni
 - 3.7.3 valutazione media
 - 3.7.4 numero di video risposta
- 3.8 video risposta?
 - 3.8.1 video alla quale si risponde

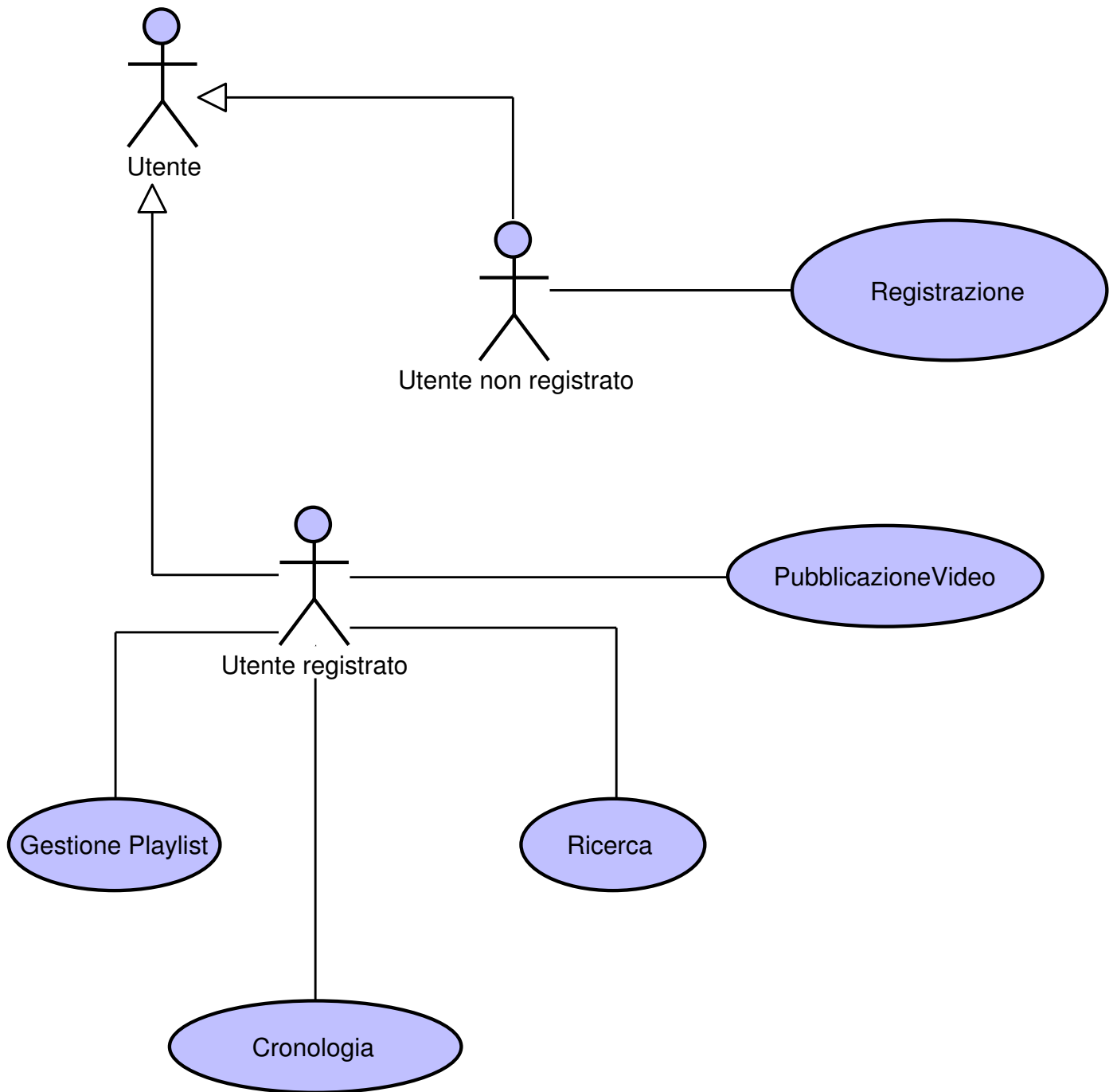
3.9 tag del video

3.10 categoria del video

2 Diagramma UML



3 Diagramma Use-Case



4 Specifiche

4.1 Specifica dei tipi di dato

FileVideo:

- sequenza di byte che codifica un flusso video
 - operazioni del tipo di dato
- $\text{durataSec}(f:\text{FileVideo}): \text{Intero} \geq 0$
pre-condizioni : nessuna
post-condizioni : result è la durata di 'f'

4.2 Specifica delle classi

4.2.1 Video

$\text{durataSec} () : \text{Intero} \geq 0$

- *pre-condizioni* : Nessuna
- *post-condizioni* : $\text{result} = \text{durataSec}(\text{this.flusso})$

$\text{n_visualizzazioni} () : \text{Intero} \geq 0$

- *pre-condizioni* : Nessuna
- *post-condizioni* : result è il numero di oggetti $v : \text{Visualizzazione}$ tali che: $(\text{this}, v) : \text{vid}_v \text{is}$

$\text{valutazione_media} () : \text{Reale} \in [1,5]$

- *pre-condizioni* : Deve esistere almeno un link $(\text{this}, \text{vis}) : \text{vid_vis}$ con $\text{vis.valutazione} \neq \text{NULL}$
- *post-condizioni* :
Sia VIS l'insieme degli oggetti $\text{vis} : \text{Visualizzazione}$ tali che $\exists (\text{this}, \text{vis}) : \text{vid_vis} \wedge \text{vis.valutazione} \neq \text{NULL}$
Si ha che

$$\text{result} = \frac{1}{|VIS|} \cdot \sum_{\text{vis} \in VIS} \text{vis.valutazione}$$

$\text{num_risposte} () : \text{Intero} \geq 0$

- *pre-condizioni* : Nessuna
- *post-condizioni* :
Sia R l'insieme dei link $r : \text{risposta_a_video}$ tali che this è coinvolto in r secondo il ruolo *originale*.
 $\text{result} = |R|$

4.3 Specifica dei vincoli esterni

[V.Video.autore_non_risponde_a_se_stesso]

Per ogni r :Video, per cui esiste v :Video tale che

(r, v) : risponde_a_video

sia:

- u :Utente tale che:

(u, r) : pubblica

Deve che (u, v) non è un link di pubblica.

Formalmente:

$$\forall r, v, u \quad Video(r) \wedge Video(v) \wedge Utente(u) \wedge \\ pubblica(u, r) \wedge \\ risponde_a_video(r, v) \rightarrow \\ \neg pubblica(u, v)$$

[V.Visualizzazione.valutare_proprio_video]

Per ogni utente u , sia VIS l'insieme degli oggetti di tipo *Visualizzazione* tale che:

- $\exists(u, vis) : ut_vis$ con $vis \in VIS$
- $\exists(vis, v) : vid_vis$ con $vis \in VIS$
- $\exists(u, v) : utente_pubblica_video$

Deve essere che $\forall vis \in Vis, \quad vis.valutazione = NULL$

Formalmente

$$\forall u, v, vis \quad Utente(u) \wedge Video(v) \wedge Visualizzazione(vis) \\ \wedge ut_vis(u, vis) \wedge vid_vis(vis, v) \\ \wedge utente_pubblica_video(u, v) \rightarrow \\ valutazione(vis, NULL)$$

[V.Playlist.no_visualizza_privata]

Per ogni utente u , se esiste $(u, p) : puo_visualizzare$ e non esiste $(u, p) : crea$, allora deve essere che $p.pubblica = True$

Formalmente

$$\forall u \forall p \quad Utente(u) \wedge Playlist(p) \wedge \\ puo_visualizzare(u, p) \wedge \neg crea(u, p) \rightarrow pubblica(p, True)$$

[V.Video.censurati]

Solo i video non censurati possono essere visualizzati, commentati, valutati o aggiunti a playlist.

Formalmente

$$\forall v, vis, e \quad Video(v) \wedge Visualizzazione(vis) \wedge Elemento(e) \\ elem_video(v, e) \vee vid_vis(vis, v) \rightarrow \\ censura(v, False)$$

[V.Valutazione.Registrati]

Se esiste $(u, v) : valutazione$, allora esiste (u, vis) e (vis, v) .

Formalmente

$$\forall ut \forall vid \quad valutazione(ut, vid) \rightarrow \exists vis \\ [Visualizzazione(vis) \wedge ut_vis(ut, vis) \wedge vid_vis(vis, vid)]$$

4.4 Specifica degli use-case

4.4.1 Cronologia

`cronologia(): Visualizzazione [0..*]`

- *pre-condizioni* : Nessuna
- *post-condizioni* : Sia $u: Utente$ l'oggetto che rappresenta l'attore che ha invocato l'operazione. *result* è l'insieme dei $v: Visualizzazione$ tali che: $(u, v) : ut_vis$

4.4.2 PubblicazioneVideo

`nuovoVideo(t:String, d:String, f:FileVideo, c:Categoria, T:Tag [1..*]) : Video`

- *pre-condizioni* : Nessuna
- *post-condizioni* : Viene creato e restituito *result:Video* tale che:
 - *result.titolo* = *t*
 - *result.descrizione* = *d*
 - *result.flusso* = *f*
 - *result.istante* = adesso

Viene creato il link $(c, result) : cat_video$.

Per ogni $t : T$, viene creato il link $(t, result) : tag_video$.

Sia $u : Utente$ l'oggetto che rappresenta l'attore che ha invocato l'operazione.

Viene creato il link $(u, result) : utente_pubblica_video$.

4.4.3 Registrazione

`nuovoUtente(nome:String) : Utente`

- *pre-condizioni* : $\nexists u : Utente$ tale che $u.nome = nome$
- *post-condizioni* : viene creato e restituito *result : Utente* tale che:
 - result.nome* = *nome*
 - result.iscrizione* = *now*

4.4.4 Gestione Playlist

`crea_playlist(nome:String, privacy : Booleano) : Playlist`

- *pre-condizioni* : Nessuna
- *post-condizioni* : Viene creato un oggetto $p : Playlist$, per cui *result* = *p*, e tale che
 - *p.nome* = *nome*
 - *p.pubblica* = *privacy*
 - *p.creazione* = *now*

Viene creato un link $(u, p) : crea$ dove $u : Utente$ è l'utente che ha chiamato l'invocazione.

aggiungi_elemento(p:Playlist, v:Video, ord:Inter):Elemento

- *pre-condizioni* : Sia $u : Utente$ l'utente che ha chiamato l'invocazione, deve esistere $(u, p) : crea$.
Sia E l'insieme di oggetti $e : Elemento$ tali che $\exists(p, e) : play_elem$, deve essere che $\forall e \in E \ e.ordine \neq ord$
Deve essere che $v.censura = False$
- *post-condizioni* :
Viene creato un oggetto $e : Elemento$, per cui $e.ordine = ord$
Viene creato il link $(p, e) : play_elem$
Viene creato il link $(e, v) : elem_video$

rimuovi_elemento(p:Playlist, ord:Inter):Elemento

- *pre-condizioni* : Sia $u : Utente$ l'utente che ha chiamato l'invocazione, deve esistere $(u, p) : crea$.
Sia E l'insieme di oggetti $e : Elemento$ tali che $\exists(p, e) : play_elem$, deve essere che $\exists e \in E \ e.ordine = ord$
- *post-condizioni* :
Sia $e : Elemento$ l'oggetto tale che $\exists(p, e) : play_elem \wedge e.ordine = ord$
L'oggetto e e tutti i link ad esso associati vengono eliminati.

4.4.5 Ricerca

ricerca_filtri (cat:Categoria, tag:Tag [0..*], v:Reale $\in [1,5]$) : Video [0..*]

- *pre-condizioni* : Nessuna
- *post-condizioni* : Sia V l'insieme di oggetti $v : Video$ tale che, $\forall v \in V$
 - $v.censura = False$
 - $\exists(v, c) : cat_video$ con $c = cat$
 - sia T l'insieme di $t.Tag$ tali che $\exists(v, t) : tag_video$, deve essere che $Tag \cap T \neq \emptyset$.
 - Si verifica una delle due condizioni seguenti
 - * $\nexists vis : Visualizzazione$ tale che $\exists(vis, v)$ e $vis.valutazione \neq NULL$
 - * $v.valutazione_media() \geq v$

$result = V$

ricerca_discussione (cat:Categoria) : Video [0..*]

- *pre-condizioni* : Nessuna
- *post-condizioni* : Sia V l'insieme di oggetti v tale che
 - $\exists(v, c) : cat_video$ con $c = cat$
 - $v.censura = False$
Sia v_{max} l'elemento di V tale che, $\forall v \in V, \ v_{max}.num_risposte() \geq v.num_risposte()$
Sia V' il sottoinsieme di V tale che $\forall v \in V' \ v_{max}.num_risposte() = v.num_risposte()$
 $result = V'$