

---

**RASD**  
***Release 0.1***

**Giovanni Gola**

23 April 2013



<b>1</b>	<b>Introduzione</b>	<b>3</b>
1.1	Glossario . . . . .	3
<b>2</b>	<b>Sistema Proposto</b>	<b>5</b>
<b>3</b>	<b>Attori</b>	<b>7</b>
<b>4</b>	<b>Requisiti</b>	<b>9</b>
4.1	Requisiti funzionali . . . . .	9
4.2	Requisiti non funzionali . . . . .	10
<b>5</b>	<b>Specifiche</b>	<b>13</b>
5.1	Definizioni generali . . . . .	13
<b>6</b>	<b>Scenari</b>	<b>15</b>
6.1	Casi d'uso . . . . .	20
6.2	Classi . . . . .	20
6.3	Sequenza . . . . .	22
6.4	Stato . . . . .	24
<b>7</b>	<b>Modello Alloy</b>	<b>27</b>
7.1	Modello . . . . .	27
7.2	Mondi generati . . . . .	30
<b>8</b>	<b>Strumenti utilizzati</b>	<b>33</b>
	<b>Indice</b>	<b>35</b>



Il sistema richiesto è un applicativo in grado di fornire e regolamentare l'accesso di professori, assistenti e tesisti al laboratorio.

Gli utenti devono essere in grado di richiedere l'emissione, la revoca o il rinnovo dei permessi di accesso. Questi permessi (nel seguito *grants*) sono di due tipi: certificati per l'accesso alla Virtual Private Network del laboratorio, e badge per l'accesso fisico. Il sistema deve inoltre notificare lo stato delle richieste ai soggetti interessati.

La modalità di controllo dell'accesso deve essere di tipo gerarchico. Nella fattispecie, l'amministratore di sistema è l'unico ruolo abilitato ad inserire professori ed assistenti nella piattaforma. I primi sono in grado di inserire tesisti nella piattaforma e di approvare le loro richieste, mentre gli assistenti possono, previa approvazione del docente, completare le richieste.

Operazioni quali la visualizzazione delle richieste e del relativo stato, o degli utenti, sono offerte dal sistema ai ruoli che ne necessitano, a supporto della funzionalità principale di gestione degli accessi. **I** dettagli sulle operazioni richieste seguono nei successivi capitoli.



---

# Introduzione

---

## 1.1 Glossario

**VPN** Una VPN o Virtual Private Network è una rete di telecomunicazioni che offre le stesse possibilità delle linee private sfruttando reti condivise pubbliche.

**Certificato VPN** L'autenticazione nelle rete VPN avviene tipicamente tramite una coppia di credenziali username/password, o tramite l'utilizzo di certificati. Il formato dei certificati utilizzabili varia a seconda della rete VPN utilizzata.

**Badge** Tessera in materiale plastico, tipicamente delle dimensioni di una carta di credito (ID-1-ISO-7810) utilizzato a scopo identificativo e di sicurezza per l'accesso ad aree riservate.

**Grant** Letteralmente “concessione”, nel sistema proposto identifica un *badge* od un *certificato VPN*.

**Request** Letteralmente “richiesta”, nel sistema proposto identifica la richiesta effettuata da parte di uno studente per ottenere uno dei tipo di *grant* per l'accesso al laboratorio.





---

### Sistema Proposto

---

Verrà realizzata un'applicazione con interfaccia web, in grado di fornire servizi differenziati a seconda del tipo di utenza.

La richiesta, emissione e revoca dei permessi d'accesso si basa sui concetti di *grant* e *request*. In particolare, gli studenti possono creare *requests* per l'emissione di certificati VPN per la rete del laboratorio, o di badge per l'accesso fisico. Il sistema provvede a notificare i professori delle *requests* create dai loro tesisti. I professori possono consultare in qualsiasi istante la lista delle *requests* in attesa, esaminarle ed eventualmente approvarle. All'atto dell'approvazione di una *request*, il sistema provvede ad inviare una notifica agli assistenti interessati. A ciascun assistente, similmente a quanto viene offerto ai professori, può consultare la lista delle richieste approvate, esaminarne i dettagli ed infine procedere con l'erogazione del permesso richiesto. Una volta confermata da un assistente, la richiesta viene rimossa dal sistema che eroga un *grant*. L'erogazione può consistere nella generazione di un certificato VPN o nell'ordinazione di un badge per l'accesso fisico al laboratorio. In ogni caso, allo studente viene inviata notifica dell'operazione. La richiesta di revoca di uno dei suddetti *grants* segue un iter del tutto analogo. Le classi `LabGrant` e `LabRequest` hanno lo scopo di fornire un'astrazione nel trattamento dei succitati *grants* (certificati VPN e badge) e *requests*, rispettivamente.

Nel sistema proposto è inoltre presente il concetto di *thesis*. Esso servirà a rappresentare le tesi a disposizione dei tesisti, che dovranno essere opportunamente inserite nel sistema dai docenti prima che essi possano aggiungervi autori (studenti nel sistema). Pur introducendo la necessità di quest'ultima operazione extra, la presenza del concetto di tesi nel sistema permette il disaccoppiamento dei professori dagli studenti, e fornisce un contesto per gli studenti del laboratorio. Ad esempio, incapsulerà le informazioni di scadenza della tesi stessa, e dunque degli accessi dei propri autori.

Il sistema gestisce anche l'emissione di badge per l'accesso fisico al laboratorio, la sicurezza è dunque di primaria importanza. Per tale motivo l'inserimento nel sistema dei tesisti è completamente a carico del professore, garantendo ai docenti il massimo controllo sugli accessi al laboratorio.



---

### Attori

---

I possibili utilizzatori del sistema sono i seguenti:

- **Amministratore** [Admin] È il solo attore in grado di aggiungere o rimuovere utenti di tipo *Professor* o *Assistant*. Inoltre può svolgere qualsiasi attività di un *Assistant*.
- **Professore** [Professor] Può approvare le richieste dei suoi studenti e specificare la durata delle tesi. Può consultare la lista dei suoi studenti e delle richieste da essi fatte.
- **Assistente** [Assistant] È in grado, previa approvazione del *Professor*, di completare le richieste avanzate dagli *Student*, emettendo, rinnovando o revocando certificati VPN o autorizzazione d'accesso fisico al laboratorio.
- **Studente** [Student] È in grado di richiedere l'emissione di certificati VPN e di badge per l'accesso fisico al laboratorio. Può inoltre richiedere il rinnovo o la revoca di certificati o badge già in suo possesso. In ogni momento può controllare lo stato d'avanzamento delle sue richieste.



---

# Requisiti

---

## 4.1 Requisiti funzionali

Il sistema offre funzionalità differenziate a seconda del tipo di utente. In particolare, successivamente al login, gli utenti dovranno essere abilitati ad eseguire le seguenti operazioni:

- **Admin:**

- Visualizzazione di tutti gli utenti del sistema
- Creazione di nuovi utenti, di qualsiasi ruolo
- Visualizzazione di tutte le richieste inserite nel sistema
- Emissione di permessi d'accesso, come specificato nelle richieste
- Rinnovo di permessi d'accesso, come specificato nelle richieste
- Revoca di permessi d'accesso, come specificato nelle richieste

- **Professor:**

- Inserimento di uno studente come proprio tesista, associandolo ad una tesi, di cui viene indicata una data di termine
- Visualizzazione dei propri tesisti
- Visualizzazione dei propri assistenti
- Registrazione di tesi, fornendone i principali dettagli
- Visualizzazione delle richieste dei propri tesisti
- Approvazione delle richieste dei propri tesisti

- **Assistant:**

- Visualizzazione delle richieste approvate dal proprio professore di riferimento
- Emissione di permessi d'accesso, come specificato nelle richieste visualizzate
- Rinnovo di permessi d'accesso, come specificato nelle richieste visualizzate
- Revoca di permessi d'accesso, come specificato nelle richieste visualizzate

- **Student:**

- Visualizzazione dello stato delle proprie richieste
- Emissione di una richiesta di certificato VPN
- Emissione di una richiesta di badge per l'accesso fisico al laboratorio

## **4.2 Requisiti non funzionali**

### **4.2.1 Interfacce utente**

La piattaforma sfrutterà principalmente un'interfaccia web, minimale e di immediato utilizzo. L'interfaccia offerta varierà a seconda del ruolo dell'utente, esponendo le operazioni ad esso consentite.

La schermata di benvenuto conterrà i campi di login, ed eventuali informazioni importanti, ad esempio in caso di manutenzione della piattaforma. Una volta effettuato il login, il sistema presenterà un'interfaccia utente a bottoni, che permetteranno la navigazione verso le pagine dedicate a funzionalità specifiche. Sarà questa schermata *dashboard* ad essere personalizzata a seconda del ruolo dell'utente.

Per facilitare la fruizione del servizio e minimizzare il numero di clic necessari ad utilizzarne le funzionalità, la schermata principale mostrerà anche le informazioni di più frequente utilizzo, come lo stato delle richieste nel caso dello studente, o una breve anteprima delle richieste in attesa di approvazione nel caso del professore. Le funzionalità complete, che per motivi di ordine ed immediatezza non possono essere integrate direttamente nella pagina principale, saranno comunque sempre raggiungibili tramite bottoni.

### **4.2.2 Architettura**

L'intero sistema verrà realizzato utilizzando la piattaforma J2EE nella modalità three-tier, e fornirà un'interfaccia web-based. Per tale motivo per poter accedere al sistema gli utenti dovranno dotarsi di un browser web recente, con funzionalità JavaScript e cookie attivate.

### **4.2.3 Gestione degli errori**

La gestione degli errori che provocano un esito negativo di un'operazione è delegata all'utente, a cui verrà immediatamente inviata notifica visiva dell'errore, corredata dalle cause d'errore rilevate, in modo tale da permetterne una corretta gestione ed una eventuale riesecuzione dell'operazione.

Nel caso invece di perdita di dati, il sistema deve necessariamente essere in grado di assicurarne persistenza e coerenza. Verrà dunque utilizzato come backend di storage un server RDBMS, configurato per una massima affidabilità.

#### 4.2.4 Sicurezza

Nel sistema richiesto la sicurezza è di primaria importanza, poichè una violazione consentirebbe l'accesso non autorizzato al laboratorio, sia remoto che fisico. È dunque necessario abilitare adeguate misure di sicurezza, quali connessioni sicure (https), token anti-CSRF in tutti i form presentati agli utenti ed hashing/salting per il salvataggio delle password di login.

Sarà infine necessario adottare una politica di backup dell'intera base di dati, presumibilmente con base regolare giornaliera.





---

### Specifiche

---

#### 5.1 Definizioni generali

Di seguito elenchiamo le specifiche necessarie affinché il sistema sviluppato incontri i requisiti richiesti:

- Ogni professore può avere un numero illimitato di tesi, tesisti ed assistenti
- Ogni studente può possedere un numero illimitato di certificati VPN
- Ogni studente può possedere un solo badge
- La richiesta di un badge implica la revoca del badge attualmente in possesso dello studente, se esistente
- Ogni studente può essere autore di una tesi soltanto
- Ogni assistente può avere un solo professore referente
- Se una tesi ha autori, deve avere anche una data di termine
- Al termine di una tesi, i permessi d'accesso dei suoi autori devono essere revocati



---

### Scenari

---

#### **L'amministratore aggiunge un professore al sistema**

##### **Attori**

- Admin

##### **Condizioni di ingresso**

- Il Professore in questione non è ancora stato inserito nel sistema

##### **Flusso degli eventi**

- L'amministratore fa il login
- L'amministratore seleziona *Add Professor*
- Il sistema visualizza la pagina per l'inserimento dei dati
- L'amministratore inserisce i dati del Professor Beduschi e conferma
- Il sistema informa l'amministratore dell'esito dell'operazione

##### **Condizioni di uscita**

- Il sistema aggiunge il Professor Beduschi al sistema e gli invia una notifica tramite e-mail

##### **Eccezioni**

- Login errato
- L'amministratore ha inserito informazioni non valide
- Il sistema non riesce a concludere l'operazione

#### **Il Professor Beduschi inserisce una tesi nel sistema**

##### **Attori**

- Professor

##### **Condizioni di ingresso**

- Il Professor Beduschi è registrato
- La tesi in questione non è ancora stata inserita nel sistema

### **Flusso degli eventi**

- Il Professor Beduschi fa il login
- Il Professor Beduschi seleziona *Register Thesis*
- Il sistema visualizza la pagina per l'inserimento dei dati
- Il Professor Beduschi inserisce i dati e conferma
- Il sistema informa il professore dell'esito dell'operazione

### **Condizioni di uscita**

- Il sistema aggiunge la tesi al sistema.

### **Eccezioni**

- Login errato
- Il Professor Beduschi ha inserito informazioni errate
- Il sistema non riesce a concludere l'operazione

## **Il professor Beduschi aggiunge lo studente Gianni come proprio tesista**

### **Attori**

- Professor

### **Condizioni di ingresso**

- Il Professor Beduschi è registrato
- La tesi in questione è registrata
- Gianni non è ancora stato inserito nel sistema

### **Flusso degli eventi**

- Il Professor Beduschi fa il login
- Il Professor Beduschi seleziona *Add Student*
- Il sistema visualizza la pagina per l'inserimento dei dati
- Il Professor Beduschi inserisce i dati di Gianni
- Il Professor Beduschi seleziona la tesi su cui Gianni lavorerà
- Il Professor Beduschi seleziona il periodo di validità della tesi
- Il Professor Beduschi conferma i dati immessi
- Il sistema informa il professor Beduschi dell'esito dell'operazione

### **Condizioni di uscita**

- Il sistema aggiunge Gianni al sistema
- Il sistema registra lo studente come autore della tesi selezionata.

### **Eccezioni**

- Login errato
- Il Professor Beduschi ha inserito informazioni errate
- Il sistema non riesce a concludere l'operazione

**Lo studente Gianni richiede un certificato VPN:**

**Attori**

- Student

**Condizioni di ingresso**

- Gianni è correttamente registrato

**Flusso degli eventi**

- Gianni fa il login
- Gianni seleziona *Request VPN Certificate*
- Il sistema informa Gianni dell'esito dell'operazione

**Condizioni di uscita**

- Il sistema crea la richiesta
- Il sistema invia una notifica tramite e-mail al professore referente di Gianni

**Eccezioni**

- Login errato
- Il sistema non riesce a concludere l'operazione

**Lo studente Gianni controlla lo stato delle sue richieste d'accesso:**

**Attori**

- Student

**Condizioni di ingresso**

- Gianni è correttamente registrato

**Flusso degli eventi**

- Gianni fa il login
- Gianni seleziona *My Requests*
- Il sistema visualizza la lista delle richieste di Gianni ed il relativo stato d'avanzamento

**Eccezioni**

- Login errato
- Il sistema non riesce a concludere l'operazione

**Il Professor Beduschi approva la richiesta di Gianni**

#### **Attori**

- Professor

#### **Condizioni di ingresso**

- Il Professor Beduschi è registrato

#### **Flusso degli eventi**

- Il Professor Beduschi fa il login
- Il Professor Beduschi seleziona *Pending Requests*
- Il sistema visualizza la lista delle richieste in attesa di approvazione
- Il Professor Beduschi seleziona la richiesta di Gianni
- Il sistema visualizza i dettagli della richiesta
- Il Professor Beduschi seleziona *Approve*
- Il sistema informa il Professor Beduschi dell'esito dell'operazione

#### **Condizioni di uscita**

- Il sistema cambia lo stato della richiesta da *Pending* ad *Approved*
- Il sistema invia una notifica tramite e-mail agli assistenti del Professor Beduschi

#### **Eccezioni**

- Login errato
- Il sistema non riesce a concludere l'operazione

### **Il Dott. Bertelli soddisfa la richiesta di Gianni:**

#### **Attori**

- Assistant

#### **Condizioni di ingresso**

- Il Dott. Bertelli è registrato
- Il Dott. Bertelli è assistente del Professor Beduschi

#### **Flusso degli eventi**

- Il Dott. Bertelli fa il login
- Il Dott. Bertelli seleziona *Approved Requests*
- Il sistema visualizza la lista delle richieste approvate dal professore
- Il Dott. Bertelli seleziona la richiesta di Gianni
- Il sistema visualizza i dettagli della richiesta
- Il Dott. Bertelli seleziona *Emit Certificate*

- Il sistema informa il Dott. Bertelli dell'esito dell'operazione

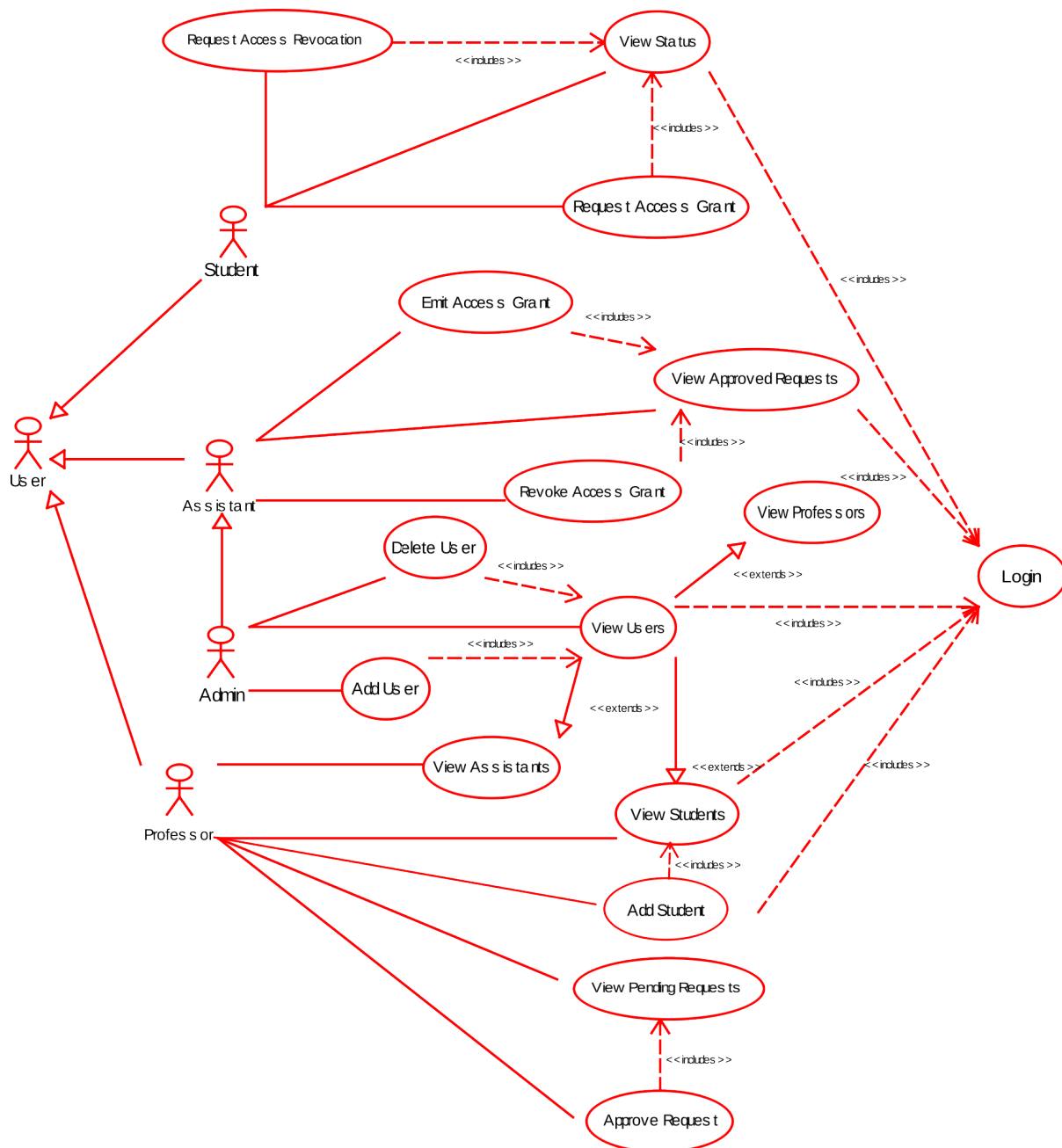
**Condizioni di uscita**

- Il sistema genera un certificato VPN legato alle credenziali di Gianni
- Il sistema invia tramite e-mail il certificato appena generato a Gianni
- Il sistema rimuove la richiesta

**Eccezioni**

- Login errato
  - Il sistema non riesce a generare un certificato
  - Il sistema non riesce a concludere l'operazione
-

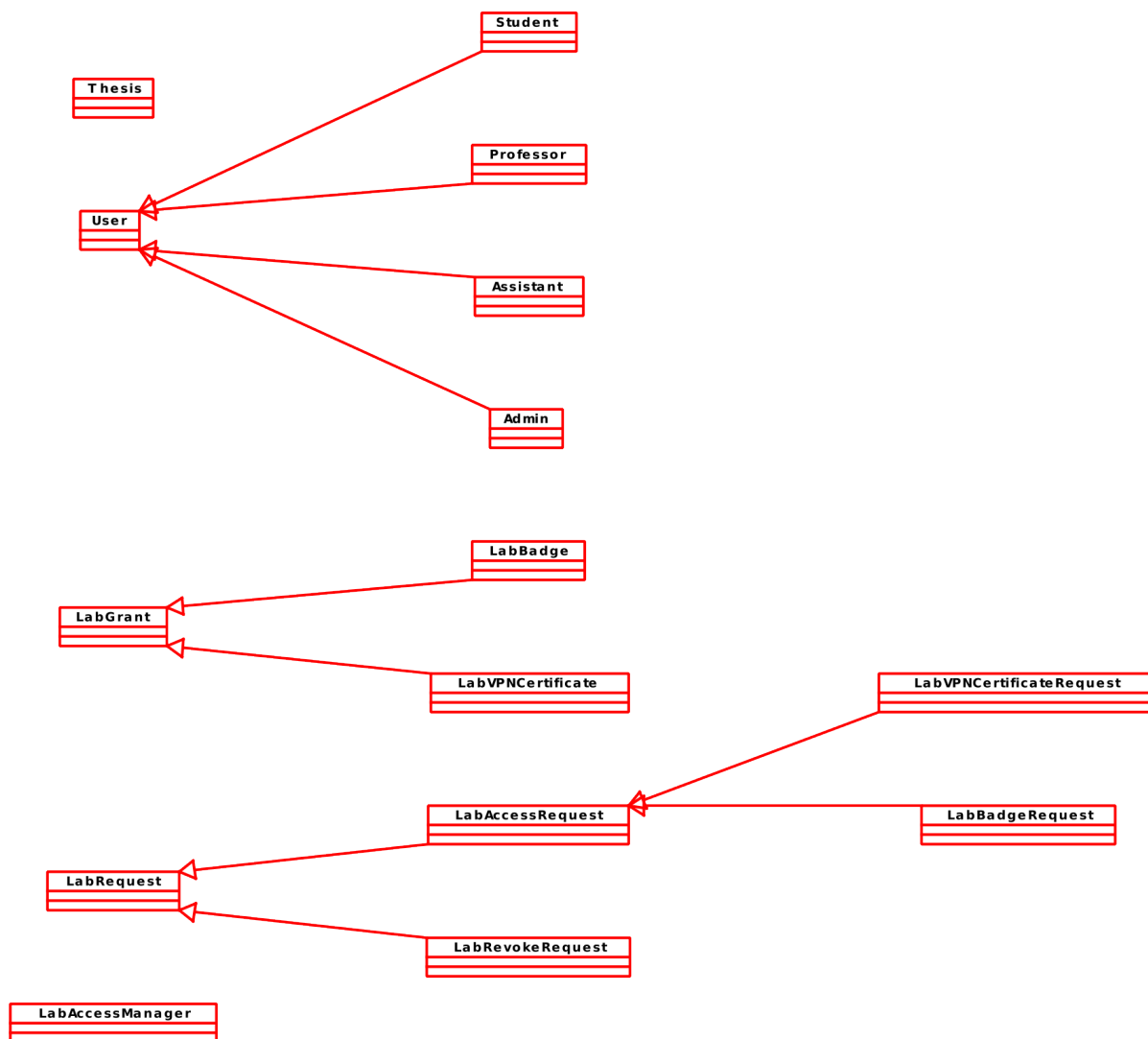
## 6.1 Casi d'uso



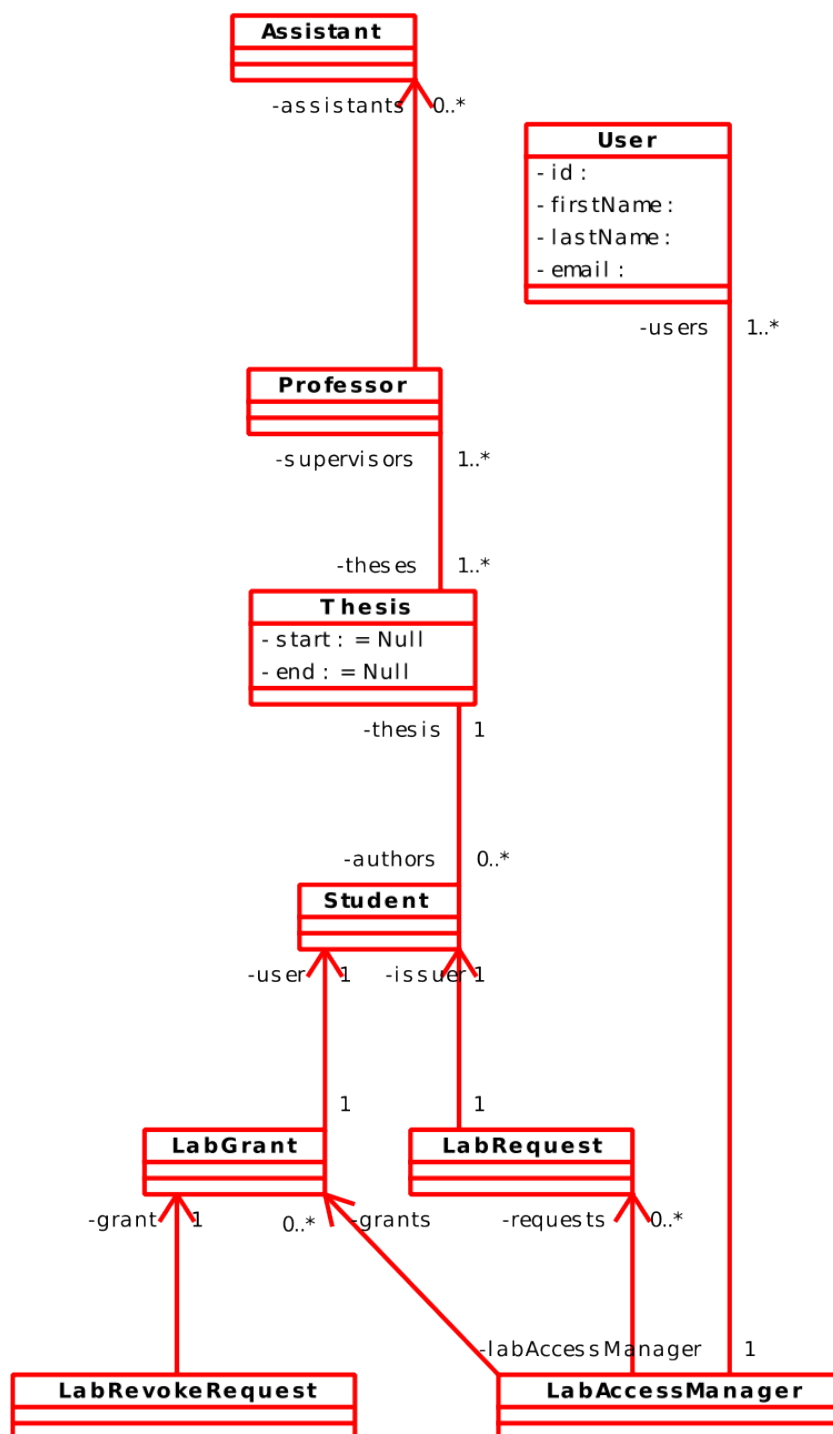
## 6.2 Classi

Per il sistema proponiamo la seguente gerarchia delle classi:





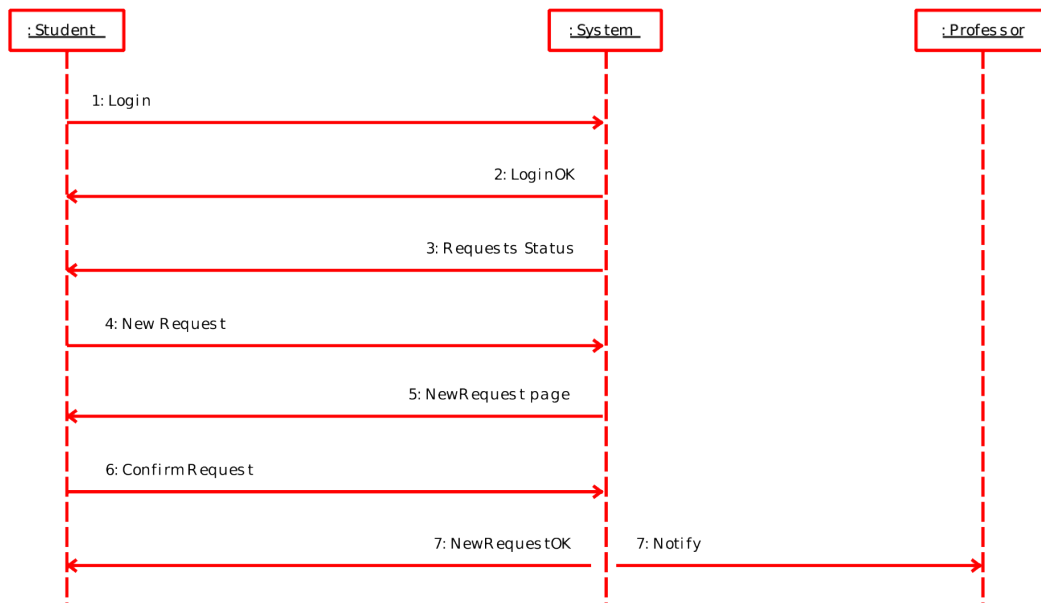
Per maggiore chiarezza, le associazioni tra le classi proposte vengono riportate separatamente nel seguente schema:



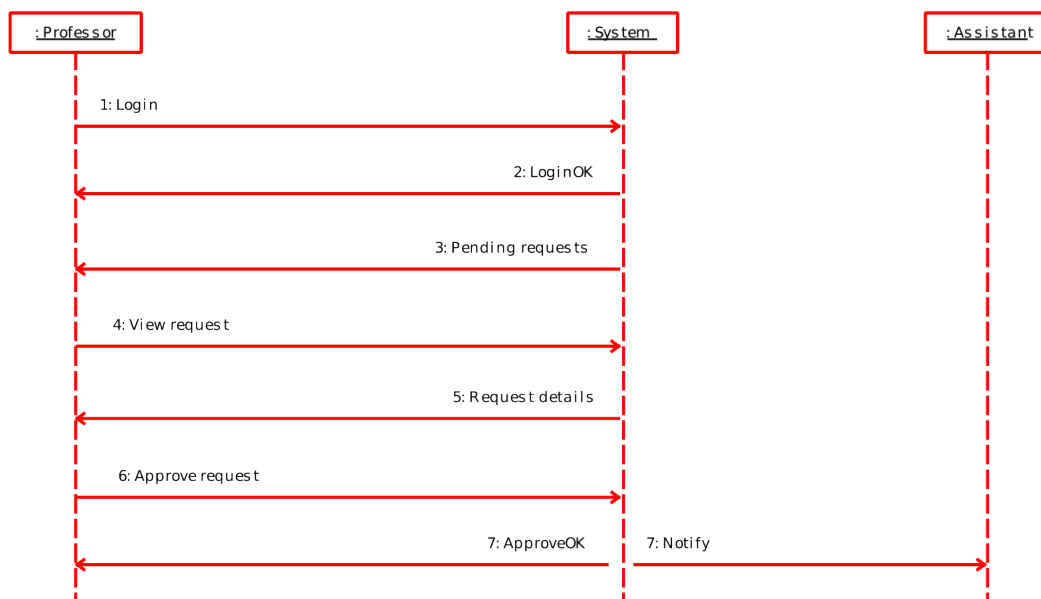
## 6.3 Sequenza

Prendiamo in considerazione un utilizzo tipico del sistema:

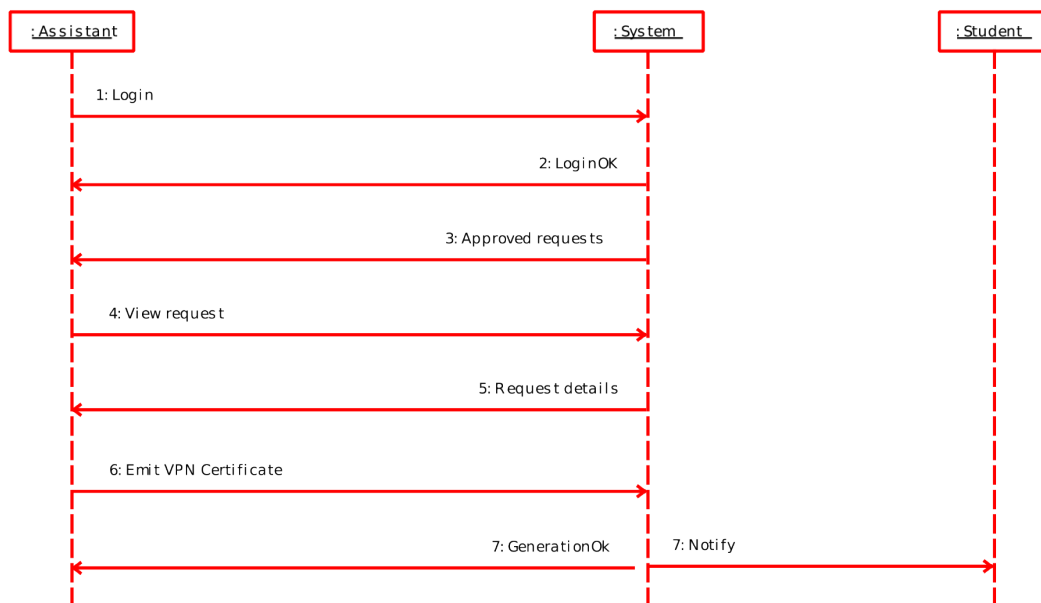
- Uno studente richiede l'emissione di un certificato. Nel caso in esame l'esito dell'operazione è positivo ed il sistema invia una notifica al professore di riferimento.



- Il professore accede al sistema, ricevendo notifica delle richieste in sospeso emesse dai suoi studenti, e ne approva una. Nel caso in esame l'esito dell'operazione è positivo ed il sistema invia una notifica ai suoi assistenti.



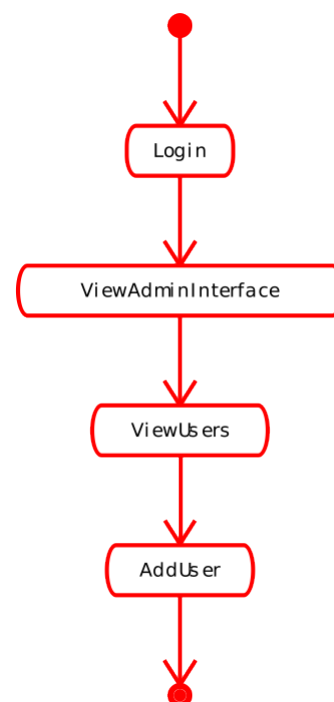
- Uno degli assistenti prende in consegna la richiesta approvata e richiede al sistema la generazione di un certificato VPN. L'esito è positivo ed il sistema invia il certificato allo studente che ne aveva fatto richiesta.



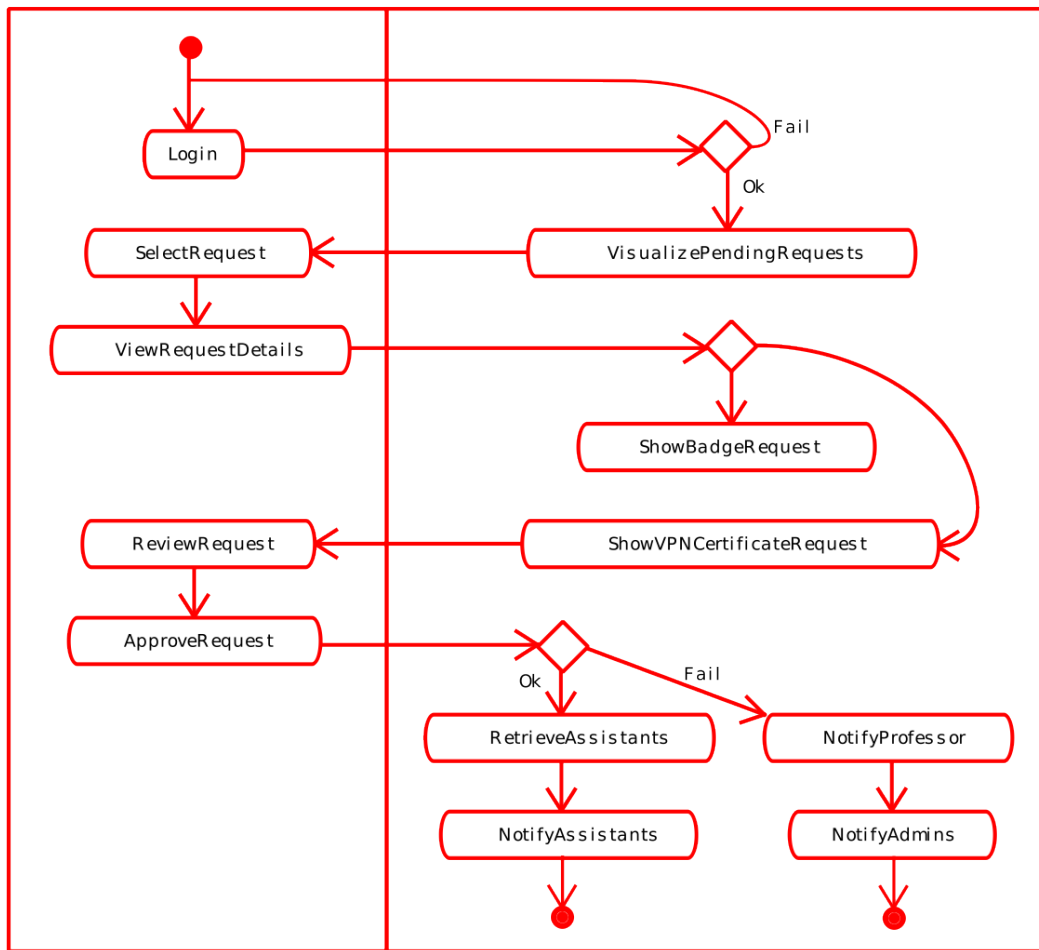
## 6.4 Stato

il flusso delle operazioni del sistema è stato modellato utilizzando i seguenti statechart:

- Un amministratore aggiunge un utente al sistema



- Un professore approva una richiesta d'accesso al laboratorio





---

## Modello Alloy

---

### 7.1 Modello

Inizialmente vengono definiti i tipi di dato necessari. In particolare un tipo `UInt` viene definito come numero intero non negativo, poichè servirà come base per i tipi `DateTime` ed `IDType`. il modello Alloy proposto è dunque il seguente:

```
module SALAME

sig UInt {
  value: one Int
} { value >= 0 }
sig DateTime extends UInt {}
sig IDType extends UInt {}

abstract sig Status {}
one sig Pending extends Status {}
one sig Approved extends Status {}

sig Thesis {
  supervisors: some Professor,
  authors: set Student,
  start: one DateTime,
  end: one DateTime
} { start.value < end.value }

abstract sig User {
  id: one IDType,
  labAccessManager: one LabAccessManager
}

some sig Admin extends User {}
sig Professor extends User {
  theses: some Thesis,
  assistants: set Assistant
}
sig Student extends User {}
```

```
sig Assistant extends User {}

abstract sig LabRequest {
  issuer: one Student,
  status : one Status
}
abstract sig LabAccessRequest extends LabRequest {}
sig LabRevokeRequest extends LabRequest {
  grant: one LabGrant
}
sig LabVPNCertificateRequest extends LabAccessRequest {}
sig LabBadgeRequest extends LabAccessRequest {}

abstract sig LabGrant {
  user: one Student
}

sig LabVPNCertificate extends LabGrant {}
sig LabBadge extends LabGrant {}

one sig LabAccessManager {
  users: set User,
  requests: set LabRequest,
  grants: set LabGrant
}

fact UniqueUserIDs {
  no disj u1, u2: User | u1.id.value = u2.id.value
}

fact NoOrphanUsers {
  all u: User | u in LabAccessManager.users
}

fact EveryStudentHasAThesis {
  all s: Student | some t: Thesis | s in t.authors
}

fact EveryAssistantHasAProfessor {
  all a: Assistant | some p: Professor | a in p.assistants
}

fact NoOrphanIDs {
  all i: IDType | i in User.id
}

fact NoOrphanDateTimes {
  all dt: DateTime | dt in (Thesis.start + Thesis.end)
}

fact NoOrphanRequests {
```



```
    all r: LabRequest | r in LabAccessManager.requests
}

fact NoOrphanGrants {
    all r: LabGrant | r in LabAccessManager.grants
}

fact SymmetryProfessorThesis {
    all p: Professor, t: Thesis | t in p.theses <=> p in t.supervisors
}

fact OneBadgePerStudent {
    no disj b1, b2: LabBadge | b1.user = b2.user
}

fact OneVPNRequestPerStudent {
    no disj r1, r2: LabVPNCertificateRequest | r1.issuer = r2.issuer
}

fact OneBadgeRequestPerStudent {
    no disj r1, r2: LabBadgeRequest | r1.issuer = r2.issuer
}

fact OneRevokeRequestPerGrant {
    no disj rr1, rr2: LabRevokeRequest | rr1.grant = rr2.grant
}

fact CanRevokeMyGrantsOnly {
    all rr: LabRevokeRequest | rr.issuer = rr.grant.user
}

fun getStudents(p: Professor) : set Student {
    { s: Student | some t: p.theses | s in t.authors }
}

pred IssueVPNCertificateRequest(s: Student, r: LabVPNCertificateRequest,
    l, l': LabAccessManager) {
    r.status = Pending
    r.issuer = s
    l'.users = l.users
    l'.grants = l.grants
    l'.requests = l.requests + r
}

run IssueVPNCertificateRequest for 5

pred IssueBadgeRevokeRequest(s: Student, r: LabRevokeRequest, b: LabBadge,
    l, l': LabAccessManager) {
    r.status = Pending
    r.issuer = s
    r.grant = b
}
```

```

    l'.users = l.users
    l'.grants = l.grants
    l'.requests = l.requests + r
  }
run IssueBadgeRevokeRequest for 5

pred ApproveVPNCertificateRequest(r, r': LabVPNCertificateRequest) {
  r'.issuer = r.issuer
  r.status = Pending => r'.status = Approved
}

run ApproveVPNCertificateRequest for 5

```

L'esecuzione dei predicati ne indica la coerenza:

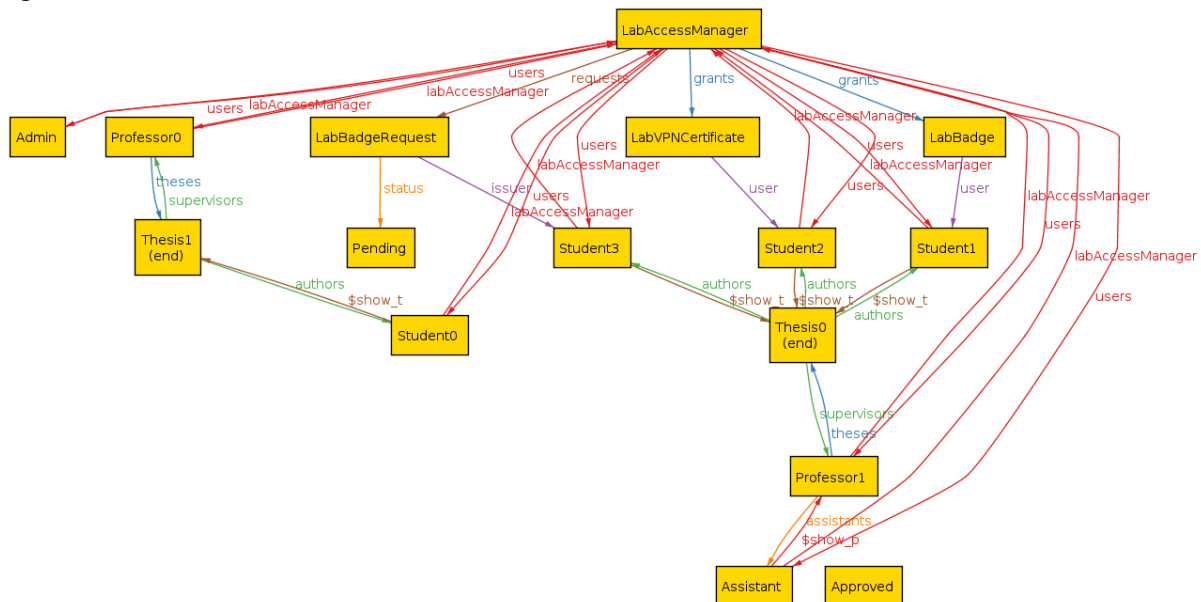
```

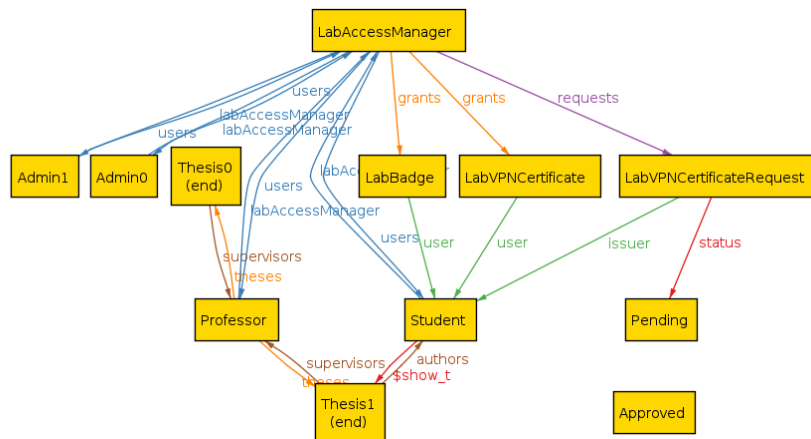
3 commands were executed. The results are:
#1: .IssueVPNCertificateRequest is consistent.
#2: .IssueBadgeRevokeRequest is consistent.
#3: .ApproveVPNCertificateRequest is consistent.

```

## 7.2 Mondi generati

Negli esempi seguenti sono stati omessi gli elementi di UInt, IDType e DateTime, per agevolarne la lettura:







---

### Strumenti utilizzati

---

Per la compilazione di questo documento d'analisi dei requisiti sono stati utilizzati i seguenti strumenti:

- **Alloy Analyzer 4:** creazione e analisi del modello Alloy
- **Umbrello UML Modeller:** diagrammi UML
- **Sphinx:** compilazione documentazione



## B

Badge, [3](#)

## C

Certificato VPN, [3](#)

## G

Grant, [3](#)

## R

Request, [3](#)

## V

VPN, [3](#)