Elaborato di Ingegneria del Software

Claudia Manfredi
Mattia Pavlovic
Elena Tonini

A.A. 2021/22

Indice

Capitolo 1

Specifiche tecniche

1.1 Specifiche

Si desidera realizzare, secondo un processo di sviluppo incrementale/iterativo, un sistema software di supporto al baratto di articoli (dotati di consistenza fisica) riconducibili a un insieme prefissato (non vuoto) di categorie. L'applicazione può essere adottata da varie organizzazioni, che sovrintendono al baratto di categorie diverse di articoli e/o operano su piazze diverse.

L'applicazione prevede due tipologie di utente, il *configuratore* e il *fruitore*.

Il primo è un esponente dell'organizzazione che, attraverso l'applicazione software, sovrintende agli scambi di beni senza uso di denaro; egli è deputato alla descrizione delle categorie di articoli scambiabili e può ottenere informazioni relative agli attuali baratti potenziali o a quelli già avvenuti.

Il secondo tipo di utente è una persona che si rivolge all'applicazione per prendere visione degli articoli pubblicati da altri (fruitori) come barattabili ed eventualmente effettuare degli scambi. Egli deve compilare dei campi atti a descrivere ciascun articolo che intende barattare.

Dopo avere descritto almeno un articolo, il fruitore può (ma non deve necessariamente) avanzare la proposta di barattare un suo articolo con quello di un altro fruitore così come rispondere alla proposta di baratto avanzata da un altro fruitore.

Si assume che le due parti coinvolte in un baratto si incontrino poi di persona al fine di scambiarsi i due articoli oggetto del baratto stesso.

Si noti che un fruitore può anche non descrivere mai alcun articolo, limitandosi a scorrere le descrizioni di articoli pubblicate dagli altri fruitori.

1.2 Generalità

L'applicazione è tesa a supportare il baratto di articoli (dotati di consistenza fisica) afferenti ad alcune categorie stabilite dal configuratore, a ciascuna delle quali corrisponde un insieme specifico di campi atti a descrivere ogni articolo (che si intende barattare) appartenente alla categoria stessa.

Ogni singola categoria è dotata di un *nome* e di una *descrizione*, entrambe stringhe in linguaggio naturale volte a esplicare il significato della categoria stessa.

Per esempio, il nome di una categoria può essere "Libro" e la sua descrizione "Opera cartacea stampata".

Una categoria può articolarsi in due o più (sotto)categorie, a loro volta suddivisibili ricorsivamente, e così via, secondo una *gerarchia* ad albero.

Ad esempio, la radice di una gerarchia può essere la categoria "Libro" ed essa può articolarsi nelle (sotto)categorie "Romanzo", "Saggio", "Opera a fumetti", "Numero di un fumetto periodico"

e “Testo scolastico”. A sua volta la categoria “Romanzo” può articolarsi nelle (sotto)categorie “Romanzo di letteratura italiana” e “Romanzo di letteratura straniera”, ecc.

Il nome di ciascuna categoria è unico all'interno della gerarchia di appartenenza.

N.B. Non si deve erroneamente pensare che una gerarchia definisca ontologicamente un concetto (quello relativo alla radice), declinandolo in tutte le sfumature possibili. La gerarchia comprende invece solo le categorie che sono di interesse ai fini delle operazioni di baratto che l'organizzazione che si è dotata dell'applicazione intende sostenere.

La medesima applicazione può considerare più gerarchie. Vige il vincolo di unicità del nome di ciascuna categoria radice entro la totalità delle categorie radice di tali gerarchie.

Gli articoli da scambiare devono appartenere solo alle categorie foglia delle gerarchie considerate.

A ciascuna categoria (a qualsiasi livello essa si collochi entro la gerarchia di appartenenza) compete inoltre un insieme (eventualmente vuoto) di *campi nativi*, la compilazione di ciascuno dei quali da parte del fruitore è obbligatoria o facoltativa.

Di seguito, la compilazione di un campo che non è esplicitamente indicata come obbligatoria è da intendersi come facoltativa.

Ogni categoria radice è dotata almeno dei seguenti due campi nativi:

- Stato di conservazione: a compilazione obbligatoria. Impone al fruitore di comunicare lo stato di conservazione e usura in cui si trova l'articolo proposto in baratto
- Descrizione libera: a compilazione facoltativa consente al fruitore di descrivere l'articolo proposto in baratto mediante un testo libero

Ciascuna categoria figlio eredita poi la totalità dei campi (nativi o ereditati) della categoria padre e i campi nativi devono avere un nome distinto da quello dei campi ereditati.

Ad esempio, i campi nativi che competono alla categoria “Libro” (oltre ai due sopraindicati, comuni a tutte le categorie radice) potrebbero essere i seguenti:

- Titolo a compilazione obbligatoria
- Autore/i a compilazione obbligatoria
- Casa editrice
- Anno di stampa

L'insieme dei campi nativi della categoria “Romanzo” potrebbe essere vuoto mentre quello della categoria “Romanzo di letteratura straniera” potrebbe essere dotato di un solo campo nativo:

- Lingua di stampa a compilazione obbligatoria

Ogni articolo da barattare può essere descritto dal fruitore assegnando un valore a ciascuno dei campi (sia nativi sia derivati) che competono alla categoria foglia di appartenenza dello stesso. Ogni valore è inserito dal fruitore sotto forma di una stringa di caratteri.

Supponendo che un fruitore intenda proporre il baratto di un articolo appartenente alla categoria (foglia) “Romanzo di letteratura straniera”, egli potrebbe, ad esempio, compilare i campi come segue:

- Stato di conservazione: nuovo
- Descrizione libera: volume a copertina flessibile mai sfogliato (doppione ricevuto in regalo)
- Titolo: A farewell to arms
- Autore/i: Ernest Hemingway
- Casa editrice: ”
- Anno di stampa: ”
- Lingua di stampa: Inglese

1.3 Versioni

Di seguito la presentazione delle versioni dell'applicazione, per ognuna delle quali vengono riportati i requisiti funzionali

1.3.1 Versione 1

La prima versione dell'applicazione consente l'accesso del solo configuratore.

Ogni configuratore effettua il primo accesso sfruttando credenziali predefinite (comunicate a ciascun nuovo configuratore che intende registrarsi), che lo qualificano come appartenente al gruppo (non vuoto, eventualmente individuale) dei configuratori dell'applicazione.

Nell'ambito della sessione aperta al primo accesso egli dovrà immediatamente scegliere credenziali personali, da usare in tutti gli accessi successivi: solo a valle di tale scelta egli potrà operare sul back-end dell'applicazione. Lo username di ciascun configuratore lo individua univocamente.

La prima versione dell'applicazione consente al configuratore di:

- introdurre una nuova gerarchia di categorie
- dotare una nuova gerarchia di categorie di nome, descrizione e di tutti i suoi campi nativi, indicando l'eventuale obbligatorietà di ciascuno
- salvare in modo persistente tutte le informazioni di cui ai punti precedenti
- visualizzare ciascuna gerarchia presente e tutte le informazioni a corredo della stessa

1.3.2 Versione 2

La seconda versione dell'applicazione consente l'accesso anche al fruitore.

Ogni fruitore deve scegliere al primo accesso le sue credenziali individuali, che gli consentiranno di operare sul front-end dell'applicazione. Lo username di ciascun fruitore individua univocamente il fruitore stesso (esso non deve coincidere con lo username di alcun altro fruitore né di alcun configuratore).

La seconda versione dell'applicazione dà la facoltà al configuratore di fissare il valore dei seguenti parametri:

- Piazza: la città in cui avvengono gli scambi; per ipotesi, tale città è unica e, una volta stabilita, non può più cambiare;
- Luoghi: alcuni luoghi (al limite uno solo) in cui tali scambi sono effettuati;
- Giorni: il giorno o i giorni della settimana in cui gli scambi possono avere luogo;

- Intervalli orari: gli intervalli orari (almeno uno) entro cui effettuare gli scambi, dove i soli orari in corrispondenza dei quali si possono fissare appuntamenti finalizzati allo scambio di articoli fra le due parti coinvolte in un baratto sono quelli dello scoccare dell'ora e della mezz'ora;
- Scadenza: il numero massimo di giorni entro cui un fruitore può accettare una proposta di scambio avanzata da un altro fruitore.

Ad esempio, il configuratore può fissare i seguenti valori:

- Piazza: Brescia
- Luoghi: Piazzale Kossuth, zona nord del parcheggio
- Giorni: giovedì, venerdì
- Intervalli orari: 17.00-19.30

Si noti che l'intervallo orario sopra esemplificato implica che gli appuntamenti possano essere fissati (solo) alle ore 17.00, 17.30, 18.00, 18.30, 19.00 e 19.30.

Infine, la seconda versione dell'applicazione visualizza, a beneficio del fruitore, il nome e la descrizione delle categorie radice di tutte le gerarchie, nonché la piazza, i luoghi, i giorni e gli intervalli orari in cui sono possibili gli scambi.

1.3.3 Versione 3

La terza versione consente al fruitore di pubblicare le informazioni circa un articolo che egli intende barattare.

A tal fine il fruitore deve individuare (magari con l'aiuto dell'applicazione) la *categoria foglia* di appartenenza e compilare i campi (nativi ed ereditati) relativi alla stessa.

L'applicazione accetta la pubblicazione relativa a un articolo solo se sono stati compilati almeno i campi obbligatori di pertinenza.

Le informazioni circa la categoria di appartenenza di tale articolo e il contenuto della pubblicazione accettata (valori dei campi), nonché il collegamento fra la stessa e lo username del fruitore autore di suddetta pubblicazione, vengono salvati in modo persistente.

Una pubblicazione accettata rappresenta un'offerta di baratto che si trova nello stato di *Offerta aperta*. Il fruitore autore di una *Offerta aperta* non può modificare le informazioni fornite in merito all'articolo da barattare, egli può però ritirare tale offerta, trasformandone così lo stato in quello di *Offerta ritirata*.

L'applicazione deve mantenere traccia dei passaggi di stato subiti da un'offerta, salvando le informazioni relative a tali passaggi in modo persistente.

In ogni momento, il configuratore e così pure il fruitore può indicare una categoria (foglia) di una qualsiasi gerarchia e visualizzare tutte le attuali *Offerte aperte* relative a tale categoria. Inoltre, il fruitore può visualizzare tutte le *Offerte aperte* e le *Offerte ritirate* di cui è autore, indipendentemente dalla categoria di appartenenza.

1.3.4 Versione 4

Questa versione consente a una coppia di fruitori di accordarsi per un baratto.

Il fruitore autore di una *Proposta aperta* (relativa a un articolo A) può infatti scegliere un'altra *Proposta aperta* (relativa a un articolo B) – ma solo se di questa non è autore ed essa appartiene alla

stessa categoria foglia. In tal modo il fruitore autore della prima proposta indica la sua disponibilità ad accettare B in cambio di A. In questo caso l'offerta relativa ad A passa nello stato di *Offerta accoppiata*, lo stato dell'offerta relativa a B cambia in quello di *Offerta selezionata* e viene creato un collegamento fra le due offerte.

Se il fruitore autore di una *Offerta selezionata* non risponde alla proposta del fruitore autore della corrispondente *Offerta accoppiata* entro il numero di giorni prestabilito (pari al valore del parametro di configurazione "Scadenza"), allo scadere di tale termine entrambe le offerte ritornano allo stato di *Offerta aperta*.

Il fruitore che invece risponde (necessariamente affermativamente) alla proposta del fruitore autore dell'*Offerta accoppiata* corrispondente entro il numero di giorni prestabilito, nella risposta deve indicare gli estremi (luogo, data e ora) di un possibile appuntamento (in occasione del quale effettuare lo scambio dei due oggetti).

Nel momento in cui tale risposta è inviata, entrambe le offerte coinvolte passano allo stato di *Offerta in scambio*, mantenendo sempre traccia l'una dell'altra.

Il fruitore autore della (ex) *Offerta accoppiata* deve ora rispondere, entro il numero massimo di giorni (il solito valore del parametro di configurazione "Scadenza"), accettando tale appuntamento o scartandolo e proponendone uno alternativo (in altro luogo e/o data e/o ora, fra quelli consentiti), e così via.

Lo scambio di proposte di appuntamenti prosegue fino a quando si verifica una di queste condizioni:

- una delle due parti accetta l'appuntamento proposto dall'altra,
- la parte che deve rispondere alla proposta di appuntamento non lo fa entro i termini stabiliti.

Se si verifica la prima condizione, l'una e l'altra *Offerta in scambio* passano nello stato di *Offerta chiusa*.
Se invece si verifica la seconda condizione, l'una e l'altra *Offerta in scambio* recidono il collegamento reciproco e passano nello stato di *Offerta aperta*.

Ciascun fruitore può visualizzare tutte le offerte di cui è autore, indipendentemente dallo stato e dalla categoria di appartenenza di ognuna.
Inoltre, per ciascuna *Offerta in scambio* di cui è autore, egli può visualizzare l'ultima risposta fornita dall'autore dell'offerta a essa collegata.

Si noti che, al passaggio di un'offerta dallo stato di *Offerta aperta* a quello di *Offerta selezionata* (e, analogamente, al passaggio di un'offerta dallo stato di *Offerta accoppiata* allo stato di *Offerta in scambio*), l'autore della prima si rende conto del fatto che è invitato a rispondere al fruitore autore della seconda – e se ne rende conto proprio in virtù di tale passaggio (di cui può acquisire consapevolezza solo se accede all'applicazione e visualizza le offerte di sua pertinenza), non già perché gli viene inviato un messaggio (ad esempio, di posta elettronica).

Parimenti, quando sopra si parla di "invio" di una risposta da parte di un fruitore, si intende che tale fruitore può "allegare" la risposta, cioè la proposta di un appuntamento, a un'*Offerta selezionata* oppure a un'*Offerta in scambio* di cui è autore e che suddetta risposta è visualizzabile dal fruitore autore dell'offerta collegata. Quest'ultimo, però, si rende conto di quale appuntamento è stato proposto solo se accede all'applicazione e richiede la visualizzazione della risposta "allegata" all'offerta collegata.

Quindi, la comunicazione non avviene mai attraverso messaggi (ad esempio, di posta elettronica) inviati esplicitamente da un fruitore a un altro (i fruitori ignorano lo username degli altri

fruitori e non hanno modo di contattarli direttamente).

Il configuratore può indicare una categoria foglia di una gerarchia e visualizzare le attuali Offerte in scambio nonché le Offerte chiuse relative ad articoli di tale categoria.

1.3.5 Versione 5

La quinta versione consente al configuratore di importare gli ingressi del back-end dell'applicazione, ovvero gerarchie delle categorie e valori dei parametri di configurazione, in modalità batch (cioè attraverso uno o più file di input) anziché in modalità interattiva.

1.4 Requisiti non funzionali

- Il modello di processo da adottare è incrementale/iterativo.
- Il linguaggio di programmazione da utilizzare è Java.
- L'architettura esterna da realizzare per l'applicazione è stand alone.
- Requisito non prescrittivo ma importante in sede di valutazione è l'impiego di precondizioni, postcondizioni e invarianti di classe entro il codice Java.
- Non è richiesta la creazione di una interfaccia utente grafica (tuttavia è bene che l'architettura interna sia progettata in modo da ridurre gli effetti collaterali e lo sforzo connesso al cambiamento se in futuro il sistema di interazione testuale fosse sostituito da una GUI).
- Non è richiesto l'impiego di alcun DBMS (Data Base Management System).

1.5 Nota

Ogni versione da produrre estende le funzionalità della precedente, senza modificarle.

Nei requisiti sopra enunciati si sono ignorati i possibili aspetti legali connessi alla gestione di un'applicazione volta a supportare il baratto di articoli, all'uso della stessa e al baratto vero e proprio.

È evidente che ogni applicazione che operi nel mondo reale deve rispettare la legislazione del Paese in cui viene utilizzata. Tale legislazione potrebbe imporre, per esempio, che ciascun fruitore fornisca le sue generalità oppure che dichiari di essere maggiorenne, che autocertifichi che gli articoli proposti in baratto rientrano nella sua personale disponibilità e non sono frutto di attività illecite, ecc. Inoltre, le Offerte aperte dovrebbero essere pubblicate solo se non contengono (ad esempio, entro la descrizione dell'articolo mediante un testo libero) messaggi inaccettabili (perché offensivi o altro).

Queste considerazioni mettono in luce l'opportunità di un ulteriore tipo di utente, il moderatore, che dovrebbe effettuare controlli di questo genere (così come quelli volti a evitare truffe).

La ragione per cui questa ulteriore figura di utente e tali controlli – essenziali in una applicazione professionale – sono stati ignorati è che l'applicazione che sarà realizzata è da intendere come il frutto di un esercizio, necessariamente di dimensioni limitate, e non è destinata a essere effettivamente installata e usata nella realtà quotidiana, come sottolineato dal requisito di adottare un'architettura esterna stand alone (mentre una applicazione che supporta il baratto nel mondo

reale dovrebbe operare in rete).

Per la stessa ragione non si è mai parlato del ritorno atteso da parte dell'organizzazione che, attraverso l'applicazione, sovrintende alle operazioni di baratto. Tale ritorno esula completamente dagli scopi didattici del progetto.

Nelle pagine precedenti si è assunto che i valori dei campi atti a descrivere un articolo che un fruitore desidera scambiare siano solo stringhe di caratteri. Nel caso più generale, alcuni valori potrebbero essere di tipo diverso, talvolta anche enumerativo (con richiesta al fruitore di fornire risposte chiuse). Inoltre, potrebbe essere contemplato che il fruitore carichi opzionalmente dei file (ad esempio, una fotografia dell'articolo che intende scambiare).

Quest'ultimo è uno scenario complesso, in cui il requisito di sicurezza gioca un ruolo di rilievo. Tale requisito, sempre più importante in un lavoro professionale, non è stato considerato nei limiti dell'esercizio proposto né il gruppo di lavoro deve tenerne conto.

L'enfasi del progetto non è infatti sulla sicurezza, pertanto anche l'attribuzione di credenziali a configuratore e fruitore non è ritenuta un'operazione critica.

L'interpretazione relativa alle "credenziali predefinite" comunicate a ciascun nuovo configuratore che intende registrarsi, di cui ai requisiti funzionali della prima versione, non è univoca. Secondo l'interpretazione più semplice – e meno sicura – tali credenziali sono uniche e immutabili, stabilite al momento dell'installazione dell'applicazione. Un'interpretazione più complessa è quella secondo cui le credenziali predefinite sono diverse per ciascun configuratore che intende registrarsi. In questo caso, al primo configuratore che intende registrarsi vengono comunicate credenziali predefinite stabilite al momento dell'installazione dell'applicazione mentre a ciascuno di quelli successivi vengono comunicate nuove credenziali, fissate di volte in volta da un configuratore già registrato. Questa soluzione è più sicura ma non ne è richiesta la realizzazione.

Secondo la trattazione precedente, il baratto è consentito solo fra articoli afferenti alla medesima categoria (foglia). In futuro si potrebbe consentire il baratto anche fra articoli di categorie compatibili, dove la compatibilità è stabilita a priori dal configuratore e salvata permanentemente.

Tutte le versioni dell'applicazione attualmente previste assumono che ciascuna gerarchia di categorie, una volta introdotta dall'utente, sia immutabile (e pertanto è molto importante che tale gerarchia sia stata congegnata con attenzione). In futuro si potrebbe permettere l'estensione di una gerarchia attraverso l'aggiunta di sottoalberi. Se il padre della radice del sottoalbero da aggiungere è una categoria foglia, l'intervento sulla gerarchia comporta effetti collaterali sulle offerte non ancora chiuse né ritirate che cadono in tale categoria.

I requisiti non funzionali non impongono alcuna tecnologia da utilizzare per la memorizzazione persistente dei dati (categorie, valori di configurazione, offerte e relativi passaggi di stato) né per l'importazione (non interattiva) degli input del back-end. Per quanto riguarda la prima, la scelta dei progettisti potrebbe cadere banalmente sulla serializzazione di oggetti. Per quanto riguarda la seconda, esistono più soluzioni, fra cui l'impiego di file di testo aventi una sintassi definita appositamente dai progettisti stessi (ad esempio, file in formato CSV o JSON o XML ...). Si noti che tali file potrebbero essere adottati anche per il salvataggio.

I requisiti (funzionali e non) delle cinque versioni dell'applicazione da realizzare sono deliberatamente espressi a un alto livello di astrazione (ad esempio, non si è parlato esplicitamente di possibili conseguenze – che comunque non è necessario siano gestite – della modifica dei valori dei parametri di configurazione, né si è fissato – e non è indispensabile farlo – un numero massimo di tentativi entro cui le due parti di un baratto devono convergere nello stabilire luogo, data e ora dell'incontro in cui avviene lo scambio effettivo degli articoli; inoltre, non si è stabilita la lunghezza massima, in termini di numero di caratteri, delle stringhe attraverso cui il fruitore esprime il valore dei campi atti a descrivere un articolo) al fine di consentire agli ingegneri del software di fornire un'interpretazione personale, che comporta sempre l'aggiunta di ulteriori requisiti.

Tali aggiunte devono essere chiaramente documentate.

1.6 Richieste

Agli studenti è richiesto di realizzare evolutivamente cinque versioni software che soddisfino i requisiti sopra esposti. Ogni gruppo (costituito al più da tre persone), dovrà:

1. per ogni versione, produrre la documentazione di progetto, contenente:
   - casi d'uso (comprensivi dell'espressione di eventuali requisiti aggiuntivi), sia in forma testuale, sia in forma di diagramma UML; si invita a rendere evidenti a colpo l'occhio le integrazioni/modifiche apportate ai casi d'uso (testuali e grafici) della versione precedente per ottenere quelli della versione corrente;
   - diagramma UML delle classi;
   - diagrammi UML comportamentali (opzionali), e qualsiasi altra specifica ritenuta opportuna;
   - la documentazione relativa alle cinque versioni deve essere raccolta in un unico file;
2. per l'ultima versione, redigere un unico manuale di installazione e uso (il cui contenuto potrebbe eventualmente divenire parte dell'help in linea dell'applicazione); si sottolinea la necessità di documentare accuratamente il da farsi al fine di importare nell'applicazione le gerarchie di categorie e i dati di configurazione;
3. consegnare in formato elettronico quanto richiesto ai punti precedenti;
4. per ogni versione, consegnare codice sorgente + codice interpretabile + (preferibilmente) codice eseguibile.

Capitolo 2

Documentazione

2.1 Versione 1

2.1.1 Casi d'uso

In questa sezione sono riportati i casi d'uso relativi alla prima versione dell'applicazione, sia in forma testuale che come diagramma UML.

Nella descrizione testuale di tutti i casi d'uso eccetto "Creazione configuratore" e "Accesso configuratore" è lasciato sottinteso il fatto che l'utente debba aver effettuato l'accesso al proprio profilo prima di poter proseguire con l'interazione descritta dal caso d'uso considerato: la scelta relativa all'azione da effettuare e di conseguenza del caso d'uso a cui far riferimento avviene infatti tramite un menu presentato all'utente solo dopo aver effettuato l'accesso, pertanto non sarebbe possibile richiedere di effettuare alcuna operazione che non sia la creazione di un nuovo profilo Configuratore o l'accesso a un profilo già esistente prima che venga presentato suddetto menu.

Nella Tabella 2.7 è riportato il caso d'uso testuale relativo all'accesso dell'utente configuratore al proprio profilo: l'utente interagisce con l'applicazione inserendo il proprio username e password; in questo caso d'uso, pertanto, è incluso il caso d'uso "Creazione configuratore" che viene invocato nella situazione in cui non esista ancora alcun utente o in cui un utente decida di non voler ritentare l'accesso a seguito dell'inserimento di credenziali errate ma piuttosto di creare un nuovo profilo.

Nome	Accesso configuratore
Attore	Configuratore
Scenario principale	Precondizione: esiste almeno un profilo Configuratore già registrato 1. L'utente inserisce le proprie credenziali Postcondizione: le credenziali inserite sono corrette e l'utente ha accesso all'applicazione Fine
Scenario alternativo	Precondizione: non esiste alcun profilo già registrato 1.a. ⟨⟨include⟩⟩ Creazione Configuratore Fine
Scenario alternativo	Precondizione: le credenziali inserite sono errate 2. Viene segnalato all'utente un errore Fine

Tabella 2.1: Caso d'uso: Accesso Configuratore

Nome	Creazione configuratore
Attore	Configuratore
Scenario principale	1. Vengono comunicate all'utente le credenziali (username e password) provvisorie con cui effettuare il primo accesso Postcondizione: viene creato un profilo configuratore con le credenziali provvisorie comunicate all'utente al punto 1 2. L'utente inserisce le credenziali provvisorie Precondizione: le credenziali inserite sono corrette 3. L'utente personalizza il proprio username Precondizione: lo username personalizzato non è associato ad alcun profilo 4. L'utente personalizza la propria password Postcondizione: vengono modificate le credenziali associate al profilo Fine
Scenario alternativo	Precondizione: la creazione del profilo non è andata a buon fine Fine
Scenario alternativo	Precondizione: le credenziali inserite sono errate 3.a Viene segnalato all'utente un errore 3.b Torna al punto 2
Scenario alternativo	Precondizione: lo username personalizzato è già associato a un profilo 4.a. Viene segnalato all'utente un errore Torna al punto 3
Scenario alternativo	Precondizione: la modifica delle credenziali non è andata a buon fine Fine

Tabella 2.2: Caso d'uso: Creazione Configuratore

Nella Tabella 2.8 è riportato il caso d'uso testuale relativo alla creazione di un nuovo profilo configuratore: l'utente riceve la comunicazione delle proprie credenziali temporanee, accede al proprio profilo e modifica le proprie credenziali a piacere (rispettando il vincolo relativo all'univocità dello username).

Nella Tabella 2.9 è riportato il caso d'uso testuale relativo alla creazione di una nuova gerarchia da parte dell'utente configuratore: l'utente, dopo aver effettuato l'accesso, interagisce con l'applicazione chiedendo di aggiungere nuove categorie oppure salvare la configurazione esistente; in questo caso d'uso, pertanto, sono inclusi i casi d'uso "Aggiunta categoria" e "Salvataggio dati", a cui si demandano i compiti relativi.

Nella Tabella 2.10 è riportato il caso d'uso testuale relativo alla Visualizzazione di tutte le gerarchie presenti all'interno dell'applicazione da parte dell'utente configuratore; anche in questo caso d'uso è necessario che l'utente sia autorizzato ad accedere all'applicazione prima di poter interagire avanzando altre richieste.

Nella Tabella 2.11 è riportato il caso d'uso testuale relativo all'aggiunta di una categoria a una gerarchia; si tratta di un caso d'uso incluso in quello relativo alla creazione di una gerarchia.

Nella Tabella 2.12 è riportato il caso d'uso testuale relativo al salvataggio dei dati introdotti nell'applicazione dall'utente configuratore; anche in questo caso d'uso è necessario che l'utente sia autorizzato ad accedere all'applicazione prima di poter interagire avanzando altre richieste.

Nome	Creazione gerarchia
Attore	Configuratore
Scenario principale	1. L'utente inserisce il nome della categoria radice Precondizione: il nome inserito è unico nell'insieme delle radici delle gerarchie 2. L'utente inserisce descrizione della categoria radice ed eventuali campi nativi Precondizione: l'utente chiede di aggiungere un'altra categoria alla gerarchia 3. Viene segnalato all'utente che ogni categoria nodo deve avere almeno due sottocategorie, poi l'utente seleziona la categoria a cui aggiungere la sottocategoria 4. ⟨⟨include⟩⟩ Aggiunta categoria Precondizione: ogni categoria nodo della gerarchia contiene almeno due sottocategorie 5. Precondizione: l'utente conferma di voler salvare i dati inseriti 6. ⟨⟨include⟩⟩ Salvataggio dati Fine
Scenario alternativo	Precondizione: il nome inserito non è unico nell'insieme delle radici delle gerarchie 2.a. Viene segnalato un errore all'utente Fine
Scenario alternativo	Precondizione: l'utente non vuole aggiungere un'altra categoria alla gerarchia Torna al punto 5
Scenario alternativo	Precondizione: almeno una categoria nodo non contiene almeno due sottocategorie Torna al punto 3
Scenario alternativo	Precondizione: l'utente conferma di non voler salvare i dati inseriti Fine

Tabella 2.3: Caso d'uso: Creazione gerarchia

Nome	Visualizzazione gerarchie
Attore	Configuratore
Scenario principale	1. Viene presentato l'elenco delle gerarchie esistenti e per ogni gerarchia sono riportate tutte le informazioni a corredo della stessa (nome, descrizione, campi nativi ed ereditati obbligatori e facoltativi) Fine

Tabella 2.4: Caso d'uso: Visualizzazione gerarchie

Nome	Aggiunta categoria
Attore	Configuratore
Scenario principale	1. L'utente inserisce il nome della categoria da creare Precondizione: il nome inserito è unico all'interno della gerarchia di appartenenza della categoria 2. L'utente inserisce la descrizione facoltativa della categoria. 3. L'utente inserisce eventuali altri campi nativi e specifica se sono a compilazione obbligatoria o facoltativa Postcondizione: viene creata una nuova categoria con le caratteristiche specificate Fine
Scenario alternativo	Precondizione: il nome inserito non è unico all'interno della gerarchia di appartenenza della categoria 2.a. Viene segnalato all'utente un errore Fine
Scenario alternativo	Precondizione: la creazione della categoria non va a buon fine Fine

Tabella 2.5: Caso d'uso: Aggiunta categoria

Nome	Salvataggio dati
Attore	Configuratore
Scenario principale	Postcondizione: Viene salvata su un file la configurazione attuale della gerarchia 1. Viene comunicato all'utente che il salvataggio è andato a buon fine. Fine
Scenario alternativo	Precondizione: il salvataggio dei dati non è andato a buon fine Fine.

Tabella 2.6: Caso d'uso: Salvataggio dati

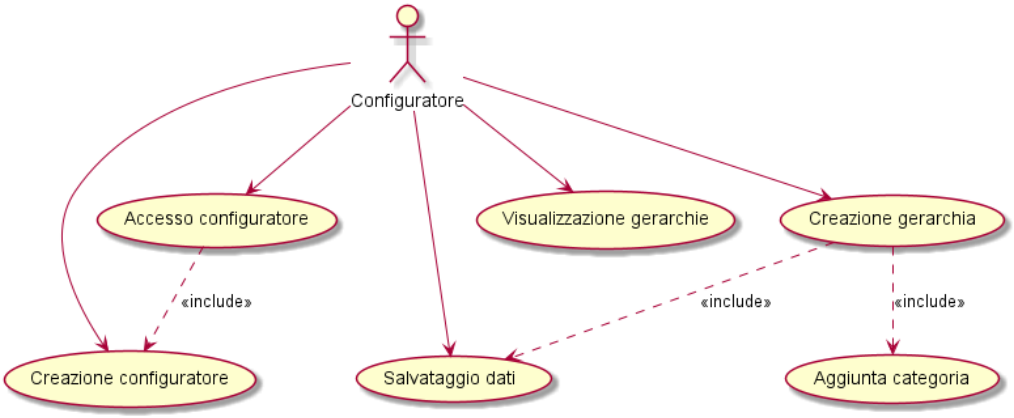

Figura 2.1: Diagramma UML dei casi d'uso - Versione 1

In Figura 2.1 è riportato il diagramma UML dei casi d'uso relativo alla prima versione dell'applicazione. Dall'immagine possiamo ricavare facilmente che i casi d'uso *sea-level*, ossia quelli corrispondenti agli obiettivi per cui l'utente interagisce con l'applicazione, sono:

- Creazione di un nuovo profilo configuratore
- Accesso a un profilo configuratore già esistente
- Creazione di una gerarchia di categorie
- Visualizzazione di tutte le gerarchie di categorie presenti nell'applicazione
- Salvataggio dei dati dell'applicazione

Alcuni di questi casi d'uso *sea-level* sono interpretabili come casi d'uso *fish-level* laddove vengano inclusi da un altro caso d'uso al fine del raggiungimento dell'obiettivo per cui l'utente interagisce con l'applicazione.

L'unico caso d'uso che ha natura esclusivamente *fish-level* è quello relativo all'aggiunta di una nuova categoria a una gerarchia, in quanto esso è incluso nel caso d'uso "Creazione gerarchia". Inoltre, i casi d'uso "Creazione configuratore", "Accesso configuratore" e "Salvataggio dati" sono inclusi in altri casi d'uso *sea-level*, pertanto possono essere visti anche come casi d'uso *fish-level* oltre che *sea-level*.

2.1.2 Diagramma UML delle classi

In questa sezione sono stati riportati tre diagrammi delle classi: il primo, in Figura 2.2, rappresenta il diagramma delle classi realizzato come prototipo per la progettazione dell'applicazione, il secondo, in Figura 2.3, rappresenta il diagramma delle classi effettivamente ottenuto a seguito della stesura del codice dell'applicazione, mentre il terzo, in Figura 2.9 rappresenta una versione semplificata del diagramma delle classi. In quest'ultima versione, in particolare, sono state rimosse alcune dipendenze in modo tale da rendere la comprensione di alcune relazioni tra classi visivamente più agevole.

Si tenga inoltre conto del fatto che le classi:

- CampoNativo
- Categoria

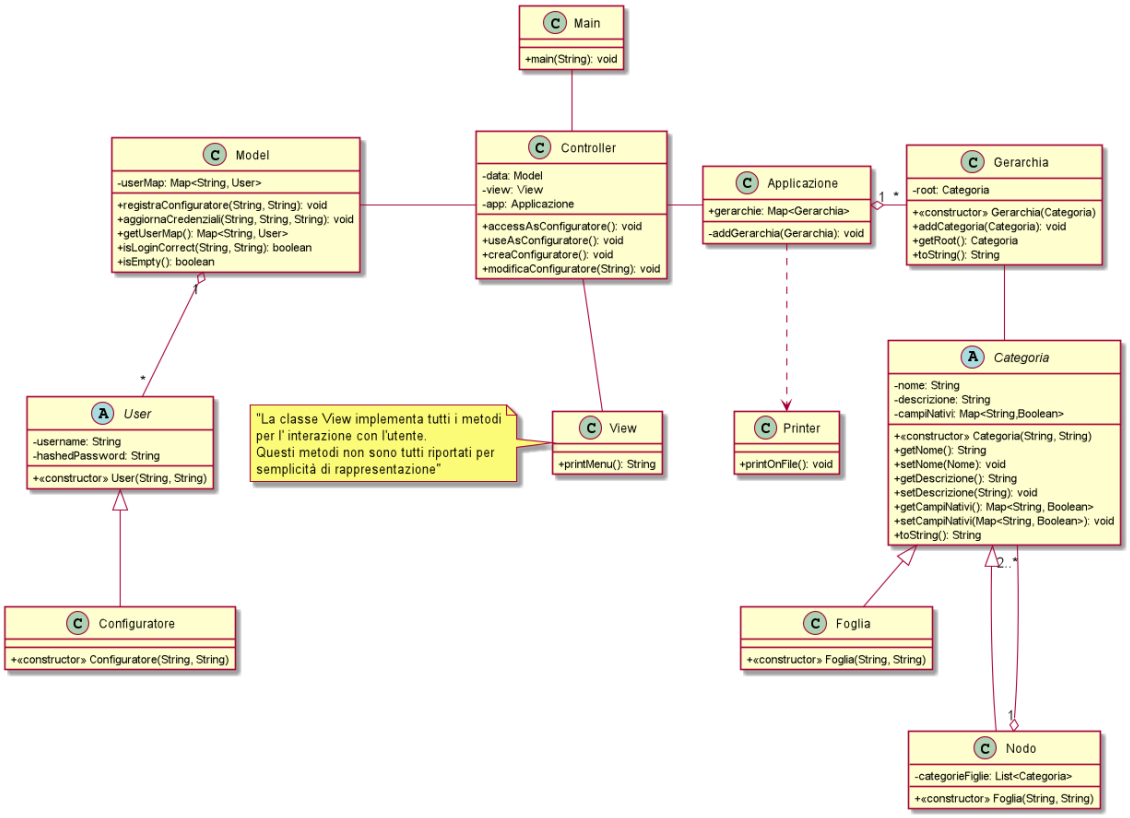

Figura 2.2: Prototipo di diagramma UML delle classi - Versione 1

- Configuratore
- Gerarchia
- User
- UserDataStore

implementano l'interfaccia `Serializable`, nonostante tale relazione non sia esplicitamente riportata nei diagrammi delle classi al fine di semplificare il layout dei diagrammi stessi.

La classe `View` contiene tutti i metodi (non indicati per esteso nel diagramma delle classi realizzato prima dell'effettiva stesura del codice) necessari per interagire e comunicare con l'utente, ricevendo i dati da lui inseriti oppure comunicando messaggi da parte del sistema.

La classe `Model`, presente con questo nome nell'UML iniziale ma rinominata `UserDataStore` nel codice finale, contiene i metodi necessari per la gestione dei dati relativi agli utenti che dovranno poi essere salvati in modo persistente. La classe ha il compito di gestire la registrazione di un nuovo utente Configuratore, l'aggiornamento delle credenziali e la verifica della correttezza delle informazioni introdotte in fase di login, oltre al salvataggio e al caricamento dei dati.

La classe `User` è una classe astratta estesa dalla classe `Configuratore`: ogni Configuratore sarà dotato di username e password. La password, in particolare, viene salvata direttamente dopo aver effettuato l'hashing, in modo da garantire un livello di sicurezza basilare relativamente alle informazioni private degli utenti. La classe `User` è dotata di metodi per l'autenticazione dell'utente e per la modifica delle credenziali.

La classe `Applicazione` permette di tenere traccia delle gerarchie contenute nell'applicazione. Essa contiene infatti un metodo che verifica se il nome di una categoria radice di una gerarchia è già utilizzato dalla categoria radice di un'altra gerarchia, uno che permette di aggiungere una

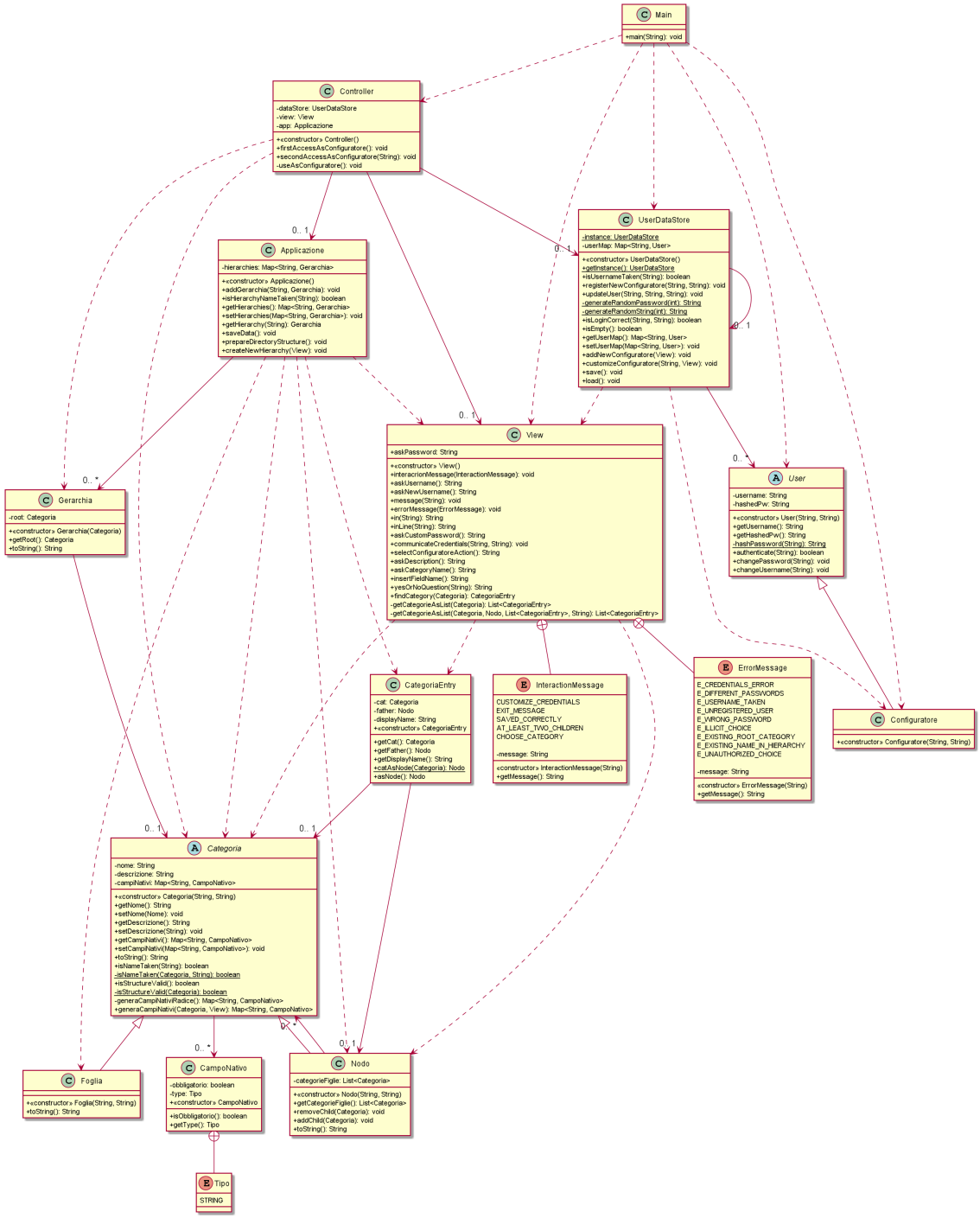

Figura 2.3: Diagramma UML delle classi - Versione 1

nuova gerarchia, metodi che permettono di ottenere le gerarchie presenti nell'applicazione o di assegnare loro i valori desiderati, metodi che permettono di salvare i dati contenuti nelle gerarchie e di caricare i dati salvati durante gli utilizzi precedenti dell'applicazione.

La classe `Gerarchia` permette di tenere traccia del contenuto di ogni gerarchia: ogni gerarchia è caratterizzata da una categoria radice che, a seconda che sia istanza della classe `Foglia` o `Nodo`, può avere o meno delle sottocategorie (a loro volta istanze della classe `Foglia` o `Nodo`).

La classe `Categoria` è una classe astratta che è estesa dalla classe `Foglia` e dalla classe `Nodo`. Ogni categoria è dotata di nome obbligatorio, descrizione facoltativa e un insieme di campi nativi (due dei quali presenti di default per la categoria radice di una gerarchia) ereditati poi da ogni categoria figlia di un nodo. La classe `Nodo`, in particolare, è dotata di una lista di categorie figlie che possono a loro volta essere foglie o nodi. Per questo motivo la classe `Nodo` è dotata di metodi che permettono di aggiungere o rimuovere una categoria figlia e di un metodo che permette di ottenere tutte le categorie figlie.

La classe `CampoNativo` tiene traccia dell'obbligatorietà della compilazione di un campo nativo o ereditato relativo a una categoria e del tipo del campo (di default si tratta di stringhe).

La classe `CategoriaEntry` tiene traccia dell'associazione tra una categoria e la sua categoria padre; essa contiene pertanto i metodi necessari a recuperare le informazioni relative a tale relazione (percorso dalla categoria radice alla categoria corrente, categoria padre e categoria corrente) e dei metodi per trasformare una categoria da Foglia a Nodo e associarla alla propria categoria padre.

La classe `Controller` contiene metodi per consentire l'accesso a un profilo Configuratore, differenziando tra il caso di registrazione e quello di accesso standard. Ognuno di questi metodi richiama poi uno o più ulteriori metodi che permettono l'utilizzo delle funzionalità a cui l'utente è autorizzato ad accedere.

La classe `Main` contiene il solo metodo main che permette di interagire con l'applicazione selezionando l'azione da compiere tra accesso, registrazione o uscita dall'applicazione.

2.1.3 Diagrammi UML comportamentali

In questa sezione sono riportati alcuni diagrammi comportamentali a corredo dei diagrammi dei casi d'uso e UML delle classi che illustrano il funzionamento della prima versione dell'applicazione.

I diagrammi UML riportati di seguito sono altamente descrittivi, in modo da essere utilizzabili per mostrare all'utente in maniera intuitiva il funzionamento dell'applicazione, pur rispettando le direttive UML.

Diagramma di sequenza: Accesso Configuratore

In Figura 2.5 è riportato il diagramma di sequenza che rappresenta gli step che caratterizzano l'interazione dell'utente con l'applicazione per consentire all'utente Configuratore di accedere all'applicazione.
Si presenta di seguito la descrizione ad alto livello delle azioni che si verificano in corrispondenza di ogni numero riportato sulle frecce:

1. L'utente richiede di accedere al proprio profilo
2. L'applicazione richiede all'utente di inserire il proprio username nell'interfaccia presentatagli
3. L'utente inserisce il proprio username
4. L'interfaccia comunica al Controller dell'applicazione lo username inserito dall'utente
5. Il Controller verifica l'esistenza dello username all'interno dell'elenco degli utenti

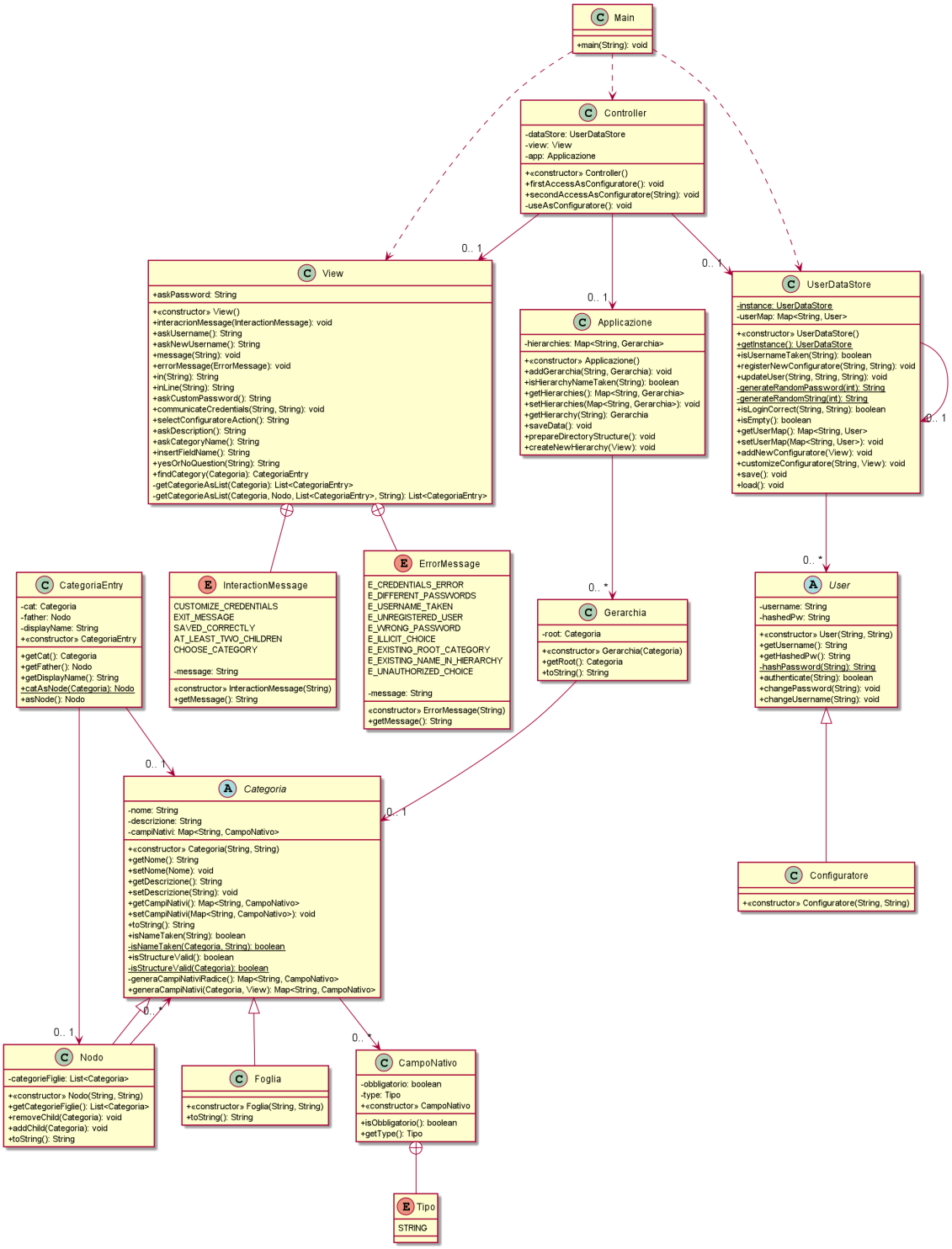

Figura 2.4: Diagramma UML delle classi semplificato - Versione 1

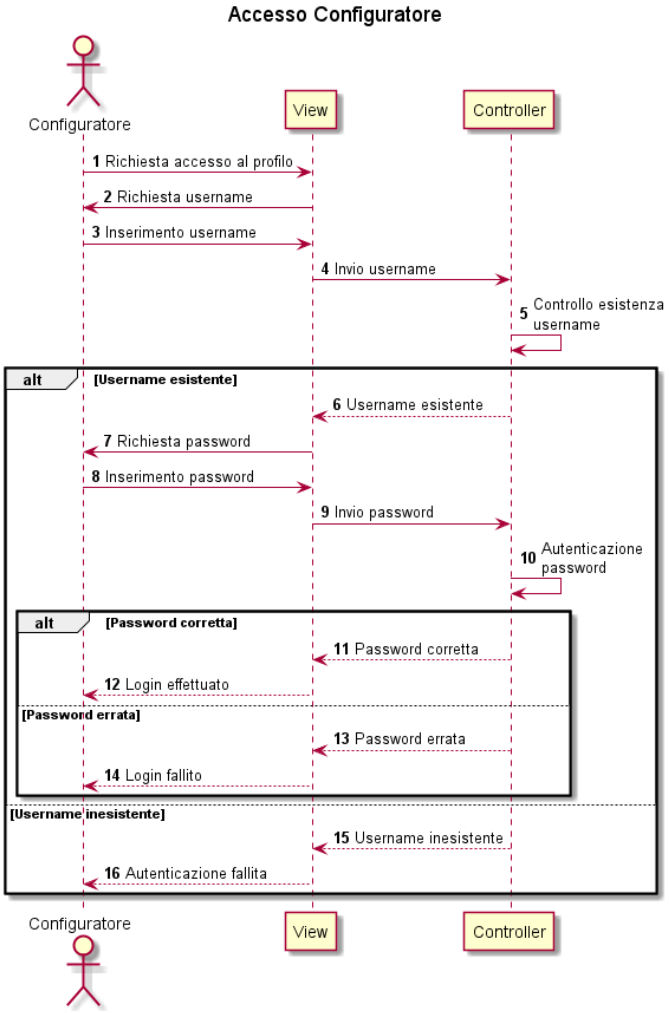

Figura 2.5: Diagramma di sequenza: Accesso configuratore - Versione 1

6. Se lo username esiste all'interno dell'elenco degli utenti il Controller manda un messaggio di conferma della correttezza dei dati inseriti all'interfaccia
7. L'interfaccia dell'applicazione richiede all'utente la password
8. L'utente inserisce la propria password
9. L'interfaccia comunica al Controller dell'applicazione la password inserita dall'utente
10. Il Controller verifica la correttezza della password
11. Se la password è corretta il Controller comunica all'interfaccia che il login è stato effettuato correttamente
12. L'interfaccia comunica all'utente che il login è stato effettuato correttamente
13. Se la password è errata il Controller comunica all'interfaccia che il login è fallito
14. L'interfaccia comunica all'utente che il login è fallito
15. Se lo username è inesistente il Controller comunica all'interfaccia che il login è fallito
16. L'interfaccia comunica all'utente che il login è fallito

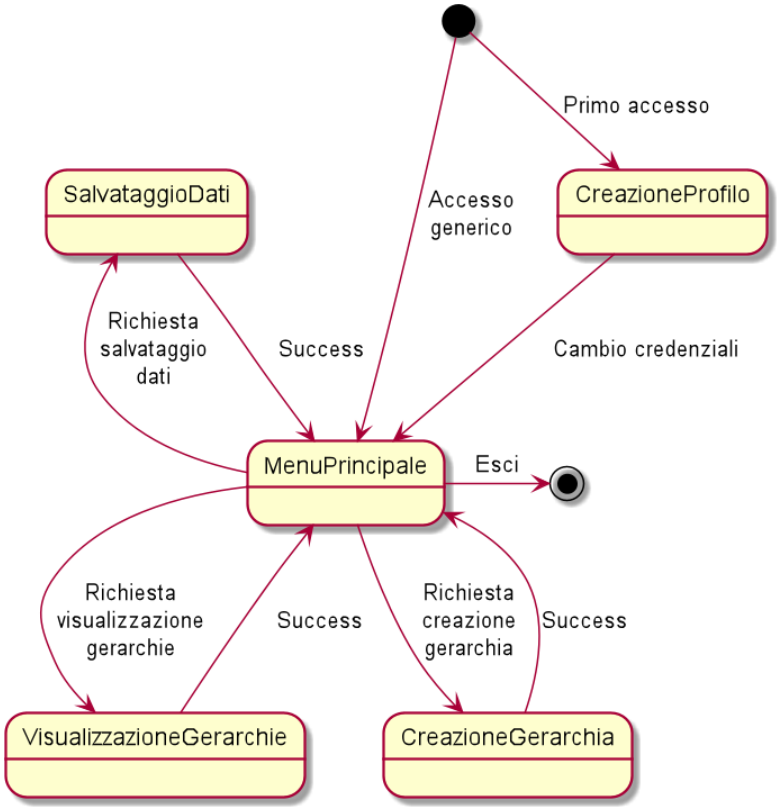

Figura 2.6: Ciclo di vita del Configuratore - Versione 1

Diagramma degli Stati: Ciclo di vita dell'utente Configuratore

In Figura 2.6 è rappresentato il ciclo di vita dell'utente Configuratore: a seconda dell'azione che esso intraprende nei confronti dell'applicazione il Configuratore si trova in un determinato stato. L'accesso all'applicazione può avvenire o in qualità di primo accesso oppure come accesso ordinario: nel caso si tratti di un primo accesso il configuratore si riconduce allo stato "CreazioneProfilo" in cui avviene la creazione di un nuovo profilo Configuratore a cui segue un necessario cambio delle credenziali, mentre nel caso si tratti di un login ordinario l'utente accede direttamente al menu principale.

In tutti i passaggi di stato si considera la situazione ideale in cui non vi siano errori nell'esecuzione delle operazioni. Nel caso in cui si verifichino errori di esecuzione allora è lasciato sottinteso il passaggio attraverso uno stato di Errore prima di tornare al menu principale.

2.2 Versione 2

2.2.1 Casi d'uso

In questa sezione sono riportati i casi d'uso relativi alla seconda versione dell'applicazione, sia in forma testuale che come diagramma UML.

In questa sezione sono riportati anche i casi d'uso che non sono soggetti a modifiche nella transizione dalla prima alla seconda versione dell'applicazione, al fine di presentarne una descrizione completa della struttura e del funzionamento.

I casi d'uso specifici della seconda versione e non presenti nella prima sono:

- Creazione fruitore
- Accesso fruitore
- Visualizzazione Informazioni app
- Configurazione parametri

Nella descrizione testuale di tutti i casi d'uso eccetto "Creazione configuratore", "Accesso configuratore", "Creazione fruitore" e "Accesso fruitore" è lasciato sottinteso il fatto che l'utente debba aver effettuato l'accesso al proprio profilo prima di poter proseguire con l'interazione descritta dal caso d'uso considerato: la scelta relativa all'azione da effettuare e di conseguenza del caso d'uso a cui far riferimento avviene infatti tramite un menu presentato all'utente solo dopo aver effettuato l'accesso, pertanto non sarebbe possibile richiedere di effettuare alcuna operazione che non sia la creazione di un nuovo profilo Configuratore o Fruitore o l'accesso a un profilo già esistente prima che venga presentato suddetto menu.

Nella Tabella 2.7 è riportato il caso d'uso testuale relativo all'accesso dell'utente configuratore al proprio profilo: l'utente interagisce con l'applicazione inserendo il proprio username e password; in questo caso d'uso, pertanto, è incluso il caso d'uso "Creazione configuratore" che viene invocato nella situazione in cui non esista ancora alcun utente o in cui un utente decida di non voler ritentare l'accesso a seguito dell'inserimento di credenziali errate ma piuttosto di creare un nuovo profilo.

Nome	Accesso configuratore
Attore	Configuratore
Scenario principale	Precondizione: esiste almeno un profilo Configuratore già registrato 1. L'utente inserisce le proprie credenziali Postcondizione: le credenziali inserite sono corrette e l'utente ha accesso all'applicazione Fine
Scenario alternativo	Precondizione: non esiste alcun profilo già registrato 1.a. ⟨⟨include⟩⟩ Creazione Configuratore Fine
Scenario alternativo	Precondizione: le credenziali inserite sono errate 2. Viene segnalato all'utente un errore Fine

Tabella 2.7: Caso d'uso: Accesso Configuratore

Nome	Creazione configuratore
Attore	Configuratore
Scenario principale	1. Vengono comunicate all'utente le credenziali (username e password) provvisorie con cui effettuare il primo accesso Postcondizione: viene creato un profilo configuratore con le credenziali provvisorie comunicate all'utente al punto 1 2. L'utente inserisce le credenziali provvisorie Precondizione: le credenziali inserite sono corrette 3. L'utente personalizza il proprio username Precondizione: lo username personalizzato non è associato ad alcun profilo 4. L'utente personalizza la propria password Postcondizione: vengono modificate le credenziali associate al profilo Fine
Scenario alternativo	Precondizione: la creazione del profilo non è andata a buon fine Fine
Scenario alternativo	Precondizione: le credenziali inserite sono errate 3.a Viene segnalato all'utente un errore 3.b Torna al punto 2
Scenario alternativo	Precondizione: lo username personalizzato è già associato a un profilo 4.a. Viene segnalato all'utente un errore Torna al punto 3
Scenario alternativo	Precondizione: la modifica delle credenziali non è andata a buon fine Fine

Tabella 2.8: Caso d'uso: Creazione Configuratore

Nella Tabella 2.8 è riportato il caso d'uso testuale relativo alla creazione di un nuovo profilo configuratore: l'utente riceve la comunicazione delle proprie credenziali temporanee, accede al proprio profilo e modifica le proprie credenziali a piacere (rispettando il vincolo relativo all'univocità dello username).

Nella Tabella 2.9 è riportato il caso d'uso testuale relativo alla creazione di una nuova gerarchia da parte dell'utente configuratore: l'utente, dopo aver effettuato l'accesso, interagisce con l'applicazione chiedendo di aggiungere nuove categorie oppure salvare la configurazione esistente; in questo caso d'uso, pertanto, sono inclusi i casi d'uso "Aggiunta categoria" e "Salvataggio dati", a cui si demandano i compiti relativi.

Nella Tabella 2.10 è riportato il caso d'uso testuale relativo alla Visualizzazione di tutte le gerarchie presenti all'interno dell'applicazione da parte dell'utente configuratore; anche in questo caso d'uso è necessario che l'utente sia autorizzato ad accedere all'applicazione prima di poter interagire avanzando altre richieste.

Nella Tabella 2.11 è riportato il caso d'uso testuale relativo all'aggiunta di una categoria a una gerarchia; si tratta di un caso d'uso incluso in quello relativo alla creazione di una gerarchia.

Nella Tabella 2.12 è riportato il caso d'uso testuale relativo al salvataggio dei dati introdotti nell'applicazione dall'utente configuratore; anche in questo caso d'uso è necessario che l'utente sia autorizzato ad accedere all'applicazione prima di poter interagire avanzando altre richieste.

Nella tabella 2.13 è riportato il caso d'uso testuale relativo all'impostazione da parte dell'utente Configuratore dei parametri e delle informazioni di scambio. Le informazioni vengono salvate in

Nome	Creazione gerarchia
Attore	Configuratore
Scenario principale	1. L'utente inserisce il nome della categoria radice Precondizione: il nome inserito è unico nell'insieme delle radici delle gerarchie 2. L'utente inserisce descrizione della categoria radice ed eventuali campi nativi Precondizione: l'utente chiede di aggiungere un'altra categoria alla gerarchia 3. Viene segnalato all'utente che ogni categoria nodo deve avere almeno due sottocategorie, poi l'utente seleziona la categoria a cui aggiungere la sottocategoria 4. ⟨⟨include⟩⟩ Aggiunta categoria Precondizione: ogni categoria nodo della gerarchia contiene almeno due sottocategorie 5. Precondizione: l'utente conferma di voler salvare i dati inseriti 6. ⟨⟨include⟩⟩ Salvataggio dati Fine
Scenario alternativo	Precondizione: il nome inserito non è unico nell'insieme delle radici delle gerarchie 2.a. Viene segnalato un errore all'utente Fine
Scenario alternativo	Precondizione: l'utente non vuole aggiungere un'altra categoria alla gerarchia Torna al punto 5
Scenario alternativo	Precondizione: almeno una categoria nodo non contiene almeno due sottocategorie Torna al punto 3
Scenario alternativo	Precondizione: l'utente conferma di non voler salvare i dati inseriti Fine

Tabella 2.9: Caso d'uso: Creazione gerarchia

Nome	Visualizzazione gerarchie
Attore	Configuratore
Scenario principale	1. Viene presentato l'elenco delle gerarchie esistenti e per ogni gerarchia sono riportate tutte le informazioni a corredo della stessa (nome, descrizione, campi nativi ed ereditati obbligatori e facoltativi) Fine

Tabella 2.10: Caso d'uso: Visualizzazione gerarchie

Nome	Aggiunta categoria
Attore	Configuratore
Scenario principale	1. L'utente inserisce il nome della categoria da creare Precondizione: il nome inserito è unico all'interno della gerarchia di appartenenza della categoria 2. L'utente inserisce la descrizione facoltativa della categoria. 3. L'utente inserisce eventuali altri campi nativi e specifica se sono a compilazione obbligatoria o facoltativa Postcondizione: viene creata una nuova categoria con le caratteristiche specificate Fine
Scenario alternativo	Precondizione: il nome inserito non è unico all'interno della gerarchia di appartenenza della categoria 2.a. Viene segnalato all'utente un errore Fine
Scenario alternativo	Precondizione: la creazione della categoria non va a buon fine Fine

Tabella 2.11: Caso d'uso: Aggiunta categoria

Nome	Salvataggio dati
Attore	Configuratore
Scenario principale	Postcondizione: Viene salvata su un file la configurazione attuale della gerarchia 1. Viene comunicato all'utente che il salvataggio è andato a buon fine. Fine
Scenario alternativo	Precondizione: il salvataggio dei dati non è andato a buon fine Fine.

Tabella 2.12: Caso d'uso: Salvataggio dati

Nome	Configurazione parametri
Attore	Configuratore
Scenario principale	Precondizione: non sono presenti informazioni di scambio nell'applicazione 1. L'utente inserisce il parametro "Piazza", ossia la città in cui avvengono gli scambi; viene segnalato che tale informazione non sarà più modificabile in futuro. 2. L'utente inserisce almeno un luogo in cui tali scambi sono effettuati. 3. L'utente inserisce almeno un giorno della settimana in cui gli scambi possono avere luogo. 4. L'utente inserisce almeno un intervallo orario entro cui effettuare gli scambi. 5. L'utente inserisce la scadenza, ossia il numero massimo di giorni entro cui un fruitore può accettare una proposta di scambio avanzata da un altro fruitore. 6. Viene data conferma all'utente che i dati sono stati salvati correttamente Fine
Scenario alternativo	Precondizione: sono presenti informazioni di scambio nell'applicazione 1.a. Vengono presentate all'utente le informazioni di scambio attualmente presenti Precondizione: l'utente conferma di voler modificare le informazioni presenti Torna al punto 2
Scenario alternativo	Precondizione: l'utente sceglie di non modificare le informazioni presenti 2.a.a Viene data conferma all'utente che i dati sono stati salvati correttamente Fine

Tabella 2.13: Caso d'uso: Configurazione parametri

Nome	Creazione fruitore
Attore	Fruitore
Scenario principale	1. L'utente inserisce il proprio username Precondizione: lo username inserito dall'utente non è presente nel sistema 2. L'utente inserisce la password da associare allo username appena inserito. Postcondizione: viene creato un nuovo profilo fruitore. Fine
Scenario alternativo	Precondizione: Lo username inserito dall'utente è già presente nel sistema 2.a. Viene segnalato che è già presente un profilo assegnato allo username appena inserito Fine
Scenario alternativo	Precondizione: il profilo non viene creato correttamente Fine

Tabella 2.14: Caso d'uso: Creazione Fruitore

modo permanente, in modo da consentirne la lettura ed eventuale modifica ad accessi successivi.

Nella Tabella 2.14 è riportato il caso d'uso testuale relativo alla creazione di un nuovo profilo fruitore: l'utente interagisce con l'applicazione inserendo il proprio username e password.

Nella Tabella 2.15 è riportato il caso d'uso testuale relativo all'accesso dell'utente fruitore al proprio profilo: l'utente interagisce con l'applicazione inserendo il proprio username e password, la cui correttezza viene verificata dall'applicazione.

Nome	Accesso fruitore
Attore	Fruitore
Scenario principale	1. L'utente inserisce il proprio username Precondizione: lo username inserito dall'utente è già presente nel sistema 2. L'utente inserisce la propria password Precondizione: la password inserita è corretta 3. Il fruitore ha accesso all'applicazione Fine
Scenario alternativo	Precondizione: lo username inserito dall'utente non è presente nel sistema 2.a. Viene segnalato un errore Fine
Scenario alternativo	Precondizione: la password inserita è errata 3.a. Viene segnalato all'utente un errore Fine

Tabella 2.15: Caso d'uso: Accesso Fruitore

Nella Tabella 2.16 è riportato il caso d'uso testuale relativo alla visualizzazione da parte di un utente fruitore delle informazioni di scambio e di nome e descrizione delle categorie radice delle gerarchie presenti nell'applicazione a seguito dell'inserimento da parte degli utenti configuratori.

Nome	Visualizzazione informazioni app
Attore	Fruitore
Scenario principale	Precondizione: sono già stati impostati le informazioni di scambio e il contenuto delle gerarchie 1. Vengono presentati il nome e la descrizione di ogni gerarchia contenuta nell'applicazione e le informazioni di scambio Fine
Scenario alternativo	Precondizione: non sono ancora state impostate le informazioni di scambio o il contenuto delle gerarchie 1.a. Viene segnalato che non sono presenti informazioni di scambio o gerarchie a seconda di quale delle due è mancante (possono mancare entrambe) Fine

Tabella 2.16: Caso d'uso: Visualizzazione Informazioni App

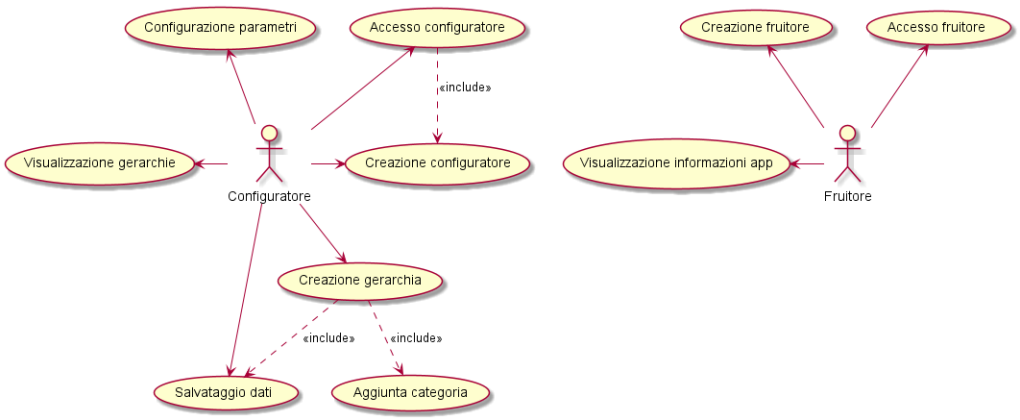

Figura 2.7: Diagramma UML dei casi d'uso - Versione 2

In Figura 2.7 è riportato il diagramma UML dei casi d'uso relativo alla seconda versione dell'applicazione. Dall'immagine possiamo ricavare facilmente che i casi d'uso *sea-level*, ossia quelli corrispondenti agli obiettivi per cui l'utente interagisce con l'applicazione, sono:

- Creazione di un nuovo profilo configuratore
- Accesso a un profilo configuratore già esistente
- Creazione di una gerarchia di categorie
- Visualizzazione di tutte le gerarchie di categorie presenti nell'applicazione
- Salvataggio dei dati dell'applicazione
- Configurazione delle informazioni di scambio
- Creazione di un nuovo profilo fruitore
- Accesso a un profilo fruitore già esistente
- Visualizzazione delle informazioni di scambio

Alcuni di questi casi d'uso *sea-level* sono interpretabili come casi d'uso *fish-level* laddove vengano inclusi da un altro caso d'uso al fine del raggiungimento dell'obiettivo per cui l'utente interagisce con l'applicazione.

L'unico caso d'uso che ha natura esclusivamente *fish-level* è quello relativo all'aggiunta di una nuova categoria a una gerarchia, in quanto esso è incluso nel caso d'uso "Creazione gerarchia". Inoltre, i casi d'uso "Creazione configuratore", "Accesso configuratore" e "Salvataggio dati" sono inclusi in altri casi d'uso *sea-level*, pertanto possono essere visti anche come casi d'uso *fish-level* oltre che *sea-level*.

2.2.2 Diagramma UML delle classi

In questa sezione sono stati riportati due diagrammi delle classi: il primo, in Figura 2.8, rappresenta il diagramma delle classi dotato di tutte le dipendenze tra le classi, mentre il secondo, in Figura 2.9, rappresenta il diagramma delle classi privato di alcune dipendenze esclusivamente al fine di renderne la comprensione più agevole.

Si tenga inoltre conto del fatto che le classi:

- CampoNativo
- Categoria
- Configuratore
- Fruitore
- Gerarchia
- Giorno
- InfoScambio
- IntervalloOrario
- Orario
- User
- UserDataStore

implementano l'interfaccia `Serializable`, nonostante tale relazione non sia esplicitamente riportata nei diagrammi delle classi al fine di semplificare il layout dei diagrammi stessi.

La classe `View` contiene tutti i metodi (non indicati per esteso nel diagramma delle classi realizzato prima dell'effettiva stesura del codice) necessari per interagire e comunicare con l'utente, ricevendo i dati da lui inseriti oppure comunicando messaggi da parte del sistema.

La classe `UserDataStore` contiene i metodi necessari per la gestione dei dati relativi agli utenti che dovranno poi essere salvati in modo persistente. La classe ha il compito di gestire la registrazione di un nuovo utente, l'aggiornamento delle credenziali (se si tratta di un profilo Configuratore) e la verifica della correttezza delle informazioni introdotte in fase di login, oltre al salvataggio e al caricamento dei dati.

La classe `User` è una classe astratta estesa dalla classe `Configuratore` e dalla classe `Fruitore`: ogni utente sarà dotato di username e password. La password, in particolare, viene salvata direttamente dopo aver effettuato l'hashing, in modo da garantire un livello di sicurezza basilare relativamente alle informazioni private degli utenti. La classe `User` è dotata di metodi per l'autenticazione dell'utente e per la modifica delle credenziali.

La classe `Applicazione` permette di tenere traccia delle gerarchie contenute nell'applicazione. Essa contiene infatti un metodo che verifica se il nome di una categoria radice di una gerarchia è già utilizzato dalla categoria radice di un'altra gerarchia, uno che permette di aggiungere una nuova gerarchia, metodi che permettono di ottenere le gerarchie presenti nell'applicazione o di assegnare loro i valori desiderati, metodi che permettono di salvare i dati contenuti nelle gerarchie e di caricare i dati salvati durante gli utilizzi precedenti dell'applicazione.

La classe `Gerarchia` permette di tenere traccia del contenuto di ogni gerarchia: ogni gerarchia è caratterizzata da una categoria radice che, a seconda che sia istanza della classe `Foglia` o `Nodo`, può avere o meno delle sottocategorie (a loro volta istanze della classe `Foglia` o `Nodo`).

La classe `Categoria` è una classe astratta che è estesa dalla classe `Foglia` e dalla classe `Nodo`. Ogni categoria è dotata di nome obbligatorio, descrizione facoltativa e un insieme di campi nativi (due dei quali presenti di default per la categoria radice di una gerarchia) ereditati poi da ogni categoria figlia di un nodo. La classe `Nodo`, in particolare, è dotata di una lista di categorie figlie che possono a loro volta essere foglie o nodi. Per questo motivo la classe `Nodo` è dotata di metodi che permettono di aggiungere o rimuovere una categoria figlia e di un metodo che permette di ottenere tutte le categorie figlie.

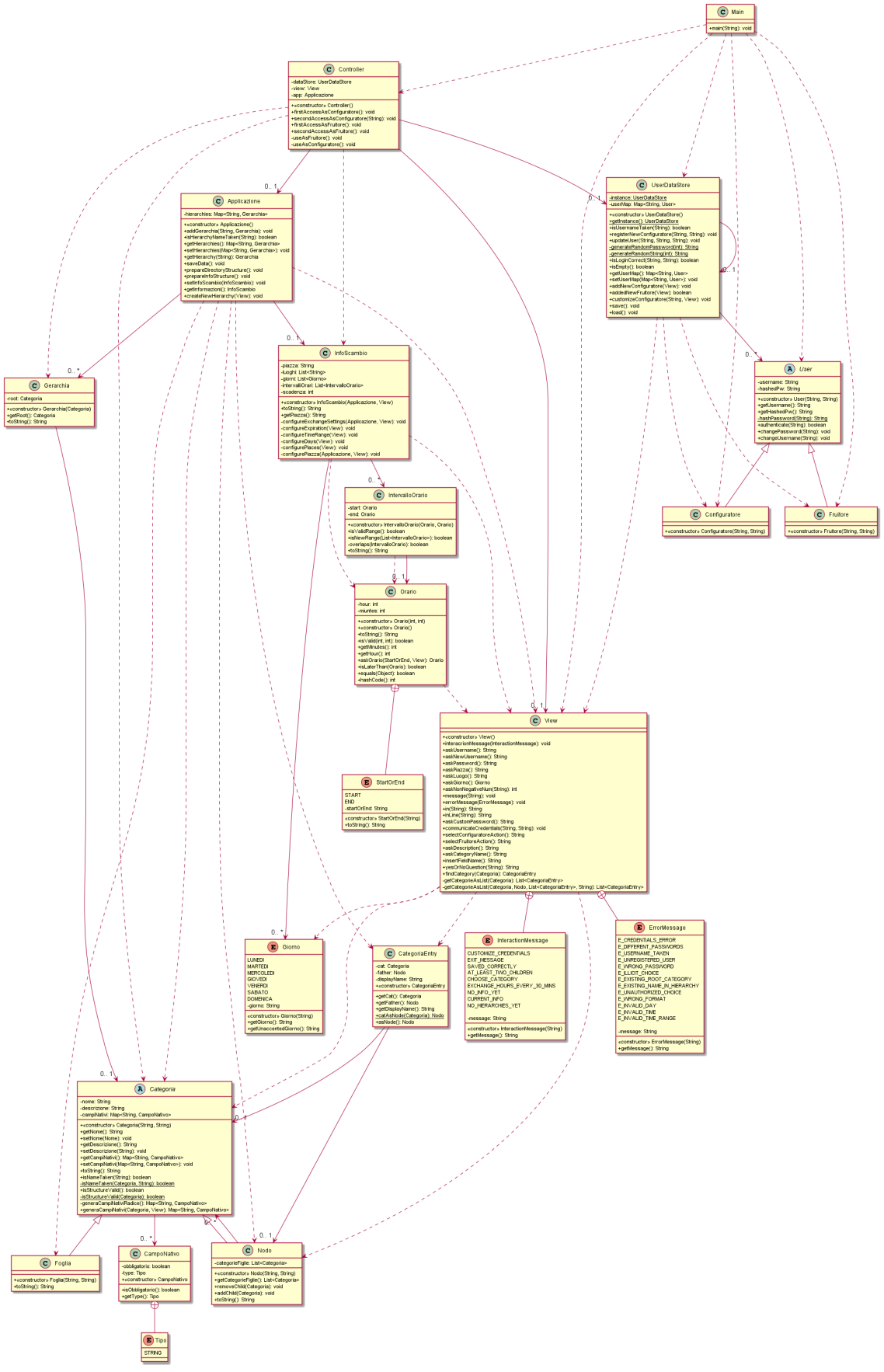

Figura 2.8: Diagramma UML delle classi - Versione 2

La classe `CampoNativo` tiene traccia dell'obbligatorietà della compilazione di un campo nativo o ereditato relativo a una categoria e del tipo del campo (di default si tratta di stringhe).

La classe `CategoriaEntry` tiene traccia dell'associazione tra una categoria e la sua categoria padre; essa contiene pertanto i metodi necessari a recuperare le informazioni relative a tale relazione (percorso dalla categoria radice alla categoria corrente, categoria padre e categoria corrente) e dei metodi per trasformare una categoria da Foglia a Nodo e associarla alla propria categoria padre.

La classe `Controller` contiene metodi per consentire l'accesso a un profilo Configuratore, differenziando tra il caso di registrazione e quello di accesso standard. Ognuno di questi metodi richiama poi uno o più ulteriori metodi che permettono l'utilizzo delle funzionalità a cui l'utente è autorizzato ad accedere.

La classe `Main` contiene il solo metodo main che permette di interagire con l'applicazione selezionando l'azione da compiere tra accesso, registrazione o uscita dall'applicazione.

L'*enum* `Giorno` gestisce l'elenco dei giorni della settimana.

La classe `InfoScambio` gestisce le informazioni di scambio che devono essere impostate dall'utente Configuratore.

La classe `IntervalloOrario` contiene metodi per validare la correttezza di un intervallo orario introdotto dall'utente, assicurandosi che l'intervallo non sia ancora stato inserito tra le informazioni di scambio e che non si sovrapponga ad altri intervalli già presenti.

La classe `Orario` contiene metodi per validare la correttezza di un singolo orario introdotto dall'utente, assicurando che esso sia nel formato *[hh:mm]* e che i minuti possano essere o 00 oppure 30, in quanto gli scambi possono avvenire solo allo scoccare dell'ora o della mezz'ora.

2.2.3 Diagrammi UML comportamentali

In questa sezione sono riportati alcuni diagrammi comportamentali a corredo dei diagrammi dei casi d'uso e UML delle classi che illustrano il funzionamento della seconda versione dell'applicazione.

I diagrammi UML riportati di seguito sono altamente descrittivi, in modo da essere utilizzabili per mostrare all'utente in maniera intuitiva il funzionamento dell'applicazione, pur rispettando le direttive UML.

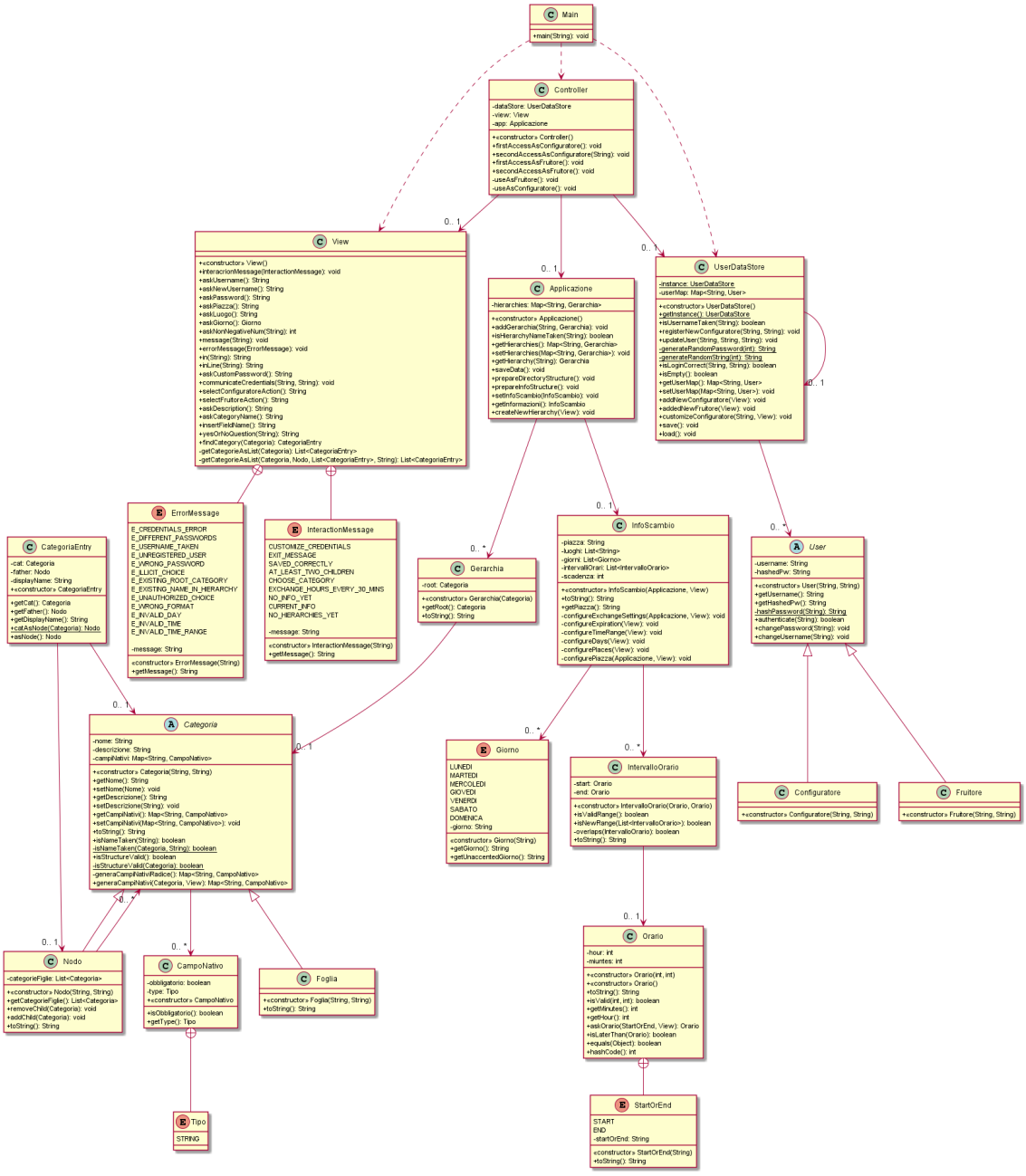

Figura 2.9: Diagramma UML delle classi semplificato - Versione 2

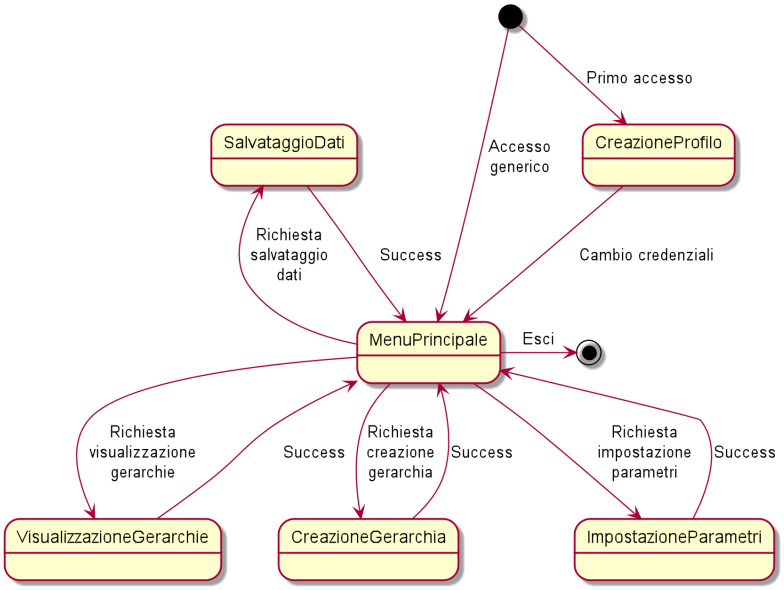

Figura 2.10: Ciclo di vita del Configuratore - Versione 2

Diagramma degli Stati: Ciclo di vita dell'utente Configuratore

In Figura 2.10 è rappresentato il ciclo di vita dell'utente Configuratore: a seconda dell'azione che esso intraprende nei confronti dell'applicazione il Configuratore si trova in un determinato stato.
L'accesso all'applicazione può avvenire o in qualità di primo accesso oppure come accesso ordinario: nel caso si tratti di un primo accesso il configuratore si riconduce allo stato `CreazioneProfilo` in cui avviene la creazione di un nuovo profilo Configuratore a cui segue un necessario cambio delle credenziali, mentre nel caso si tratti di un login ordinario l'utente accede direttamente al menu principale.
In tutti i passaggi di stato si considera la situazione ideale in cui non vi siano errori nell'esecuzione delle operazioni. Nel caso in cui si verifichino errori di esecuzione allora è lasciato sottinteso il passaggio attraverso uno stato di Errore prima di tornare al menu principale.

Diagramma degli Stati: Ciclo di vita dell'utente Fruitore

In Figura 2.11 è rappresentato il ciclo di vita dell'utente Fruitore: a seconda dell'azione che esso intraprende nei confronti dell'applicazione il Fruitore si trova in un determinato stato.
L'accesso all'applicazione può avvenire o in qualità di primo accesso oppure come accesso ordinario: nel caso si tratti di un primo accesso il Fruitore si riconduce allo stato `CreazioneProfilo` in cui avviene la creazione di un nuovo profilo Fruitore, mentre nel caso si tratti di un login ordinario l'utente accede direttamente al menu principale.
In tutti i passaggi di stato si considera la situazione ideale in cui non vi siano errori nell'esecuzione delle operazioni. Nel caso in cui si verifichino errori di esecuzione allora è lasciato sottinteso il passaggio attraverso uno stato di Errore prima di tornare al menu principale.

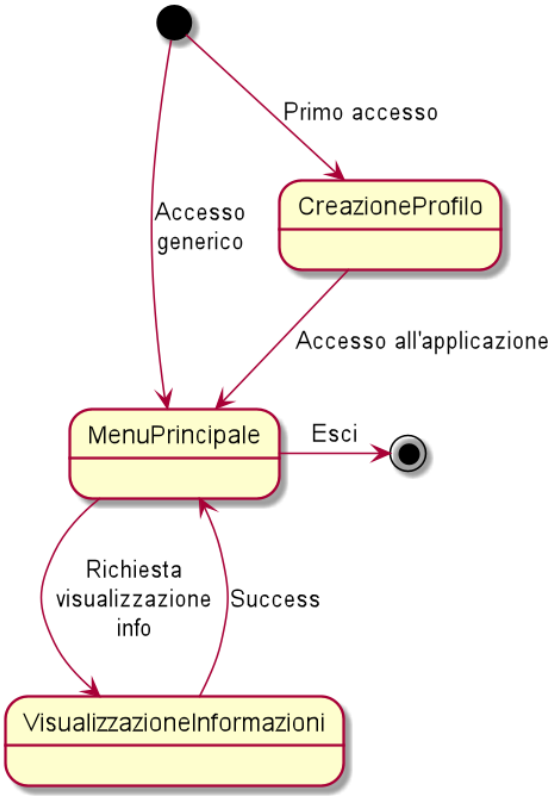

Figura 2.11: Ciclo di vita del Fruitore - Versione 2