

Sommario

1 Ideazione	3
1.1 Requisiti.....	3
1.2 Obiettivi e casi d'uso.....	3
1.3 Modello dei casi d'uso in formato dettagliato	4
UC1: Registrazione amministratore	4
UC2: Gestisci professori.....	5
UC3: Gestisci classi	5
UC4: Inserimento nuovo alunno	6
UC5: Creazione lezione	7
UC6: Invio avviso	8
UC7: Gestione registro assenze.....	8
UC8: Gestione registro voti	9
UC9: Visualizzazione info classe	10
1.4 Diagramma UML dei casi d'uso	11
1.5 Regole di dominio	11
1.6 Specifiche Supplementari	12
1.7 Glossario.....	12
2 Documento di visione	13
2.1 Introduzione.....	13
2.2 Scopo	13
2.3 Portata.....	13
2.4 Definizioni, acronimi e abbreviazioni.....	13
2.5 Posizionamento	13
2.6 Parti interessate e descrizioni utente	14
3 Analisi iterativa	15
3.1 Riepilogo iterazioni	15
3.2 Modello di dominio	16
3.3 SSD e contratti	17
❖ Iterazione 1:	17
❖ Iterazione 2:	20
❖ Iterazione 3:	23

❖ Iterazione 4:	26
❖ Iterazione 5:	29
4 Progettazione	31
5.1 Diagramma delle classi	31
5.2 Diagrammi di sequenza	31
❖ Iterazione 1	32
❖ Iterazione 2	34
❖ Iterazione 3	38
❖ Iterazione 4	42
❖ Iterazione 5	45
5 Testing	47
6 Pattern	49

1 Ideazione

1.1 Requisiti

Il gestore di un istituto scolastico richiede un software di facile utilizzo che permetta l'informatizzazione dei servizi scolastici. Il software deve permettere al gestore dell'istituto di registrarsi al sistema compilando un form e quindi accedere come amministratore. All'amministratore del sistema deve essere permesso gestire i professori nell'organico, gestire le classi, aggiungere alunni ad una classe, e creare una lezione di una materia scolastica selezionando il professore che la tiene, la classe, e l'orario. Il sistema deve garantire, inoltre, la possibilità di inviare un avviso ad una o più classi che verrà memorizzato in uno storico. Il gestore dell'istituto gradisce che il sistema possa tenere in considerazione la capienza massima di ogni classe, e che quindi non sia possibile aggiungere alunni ad una classe oltre tale limite. Un ulteriore requisito richiesto è che il sistema possa facilitare la stesura del calendario delle lezioni, evitando sovrapposizioni tra gli orari dei professori e delle classi.

Ogni professore potrà accedere al sistema tramite un identificativo ed una password forniti dall'istituto. Il professore deve poter gestire il registro delle assenze e il registro voti, abbinando un voto ad un alunno in una lezione da lui tenuta.

Anche gli alunni potranno accedere al sistema con un identificativo ed una password loro forniti, dopo l'accesso sarà possibile visionare le informazioni relative alla loro classe quali: elenco studenti, calendario delle lezioni, voti personali, avvisi e la media voti calcolata secondo le regole d'istituto correnti.

1.2 Obiettivi e casi d'uso

Di seguito un riepilogo casi d'uso:

Attore	Obiettivo	Caso d'uso
Amministratore	Registrare un amministratore del sistema	UC1: Registrazione Amministratore
Amministratore	Aggiungere o rimuovere professori all'organico	UC2: Gestione professori
Amministratore	Aggiungere o rimuovere classi all'istituto	UC3: Gestione classi
Amministratore	Aggiungere un nuovo alunno ad una classe	UC4: Inserimento nuovo alunno
Amministratore	Creare lezione di una materia, tenuta da un professore in una classe	UC5: Creazione lezione
Amministratore	Inviare un avviso ad una o più classi e salvarlo nello storico	UC6: Invio avviso

Professore	Registrare le assenze giornaliere degli alunni	UC7: Gestione registro assenze
Professore	Registrare il voto di un alunno per una lezione	UC8: Gestione registro voti
Alunno	Visualizzare informazioni personali e della classe	UC9: Visualizzazione info classe

1.3 Modello dei casi d'uso in formato dettagliato

Durante la fase di ideazione solo i casi d'uso 4,5,7 sono stati descritti in formato dettagliato, poi gradualmente sono stati analizzati dettagliatamente anche tutti gli altri casi d'uso ed è stato aggiornato il presente capitolo.

UC1: Registrazione amministratore

Nome del caso d'uso	UC1: Registrazione amministratore
Portata	Applicazione ScuolaGO
Livello	Obiettivo amministratore
Attore primario	Amministratore e gestore dell'istituto
Parti interessate e interessi	Amministratore: vuole registrarsi al sistema velocemente
Pre-condizioni	L'amministratore non deve essere già presente sul sistema
Garanzia di successo	L'amministratore viene registrato dal sistema
Scenario principale di successo	<ol style="list-style-type: none"> 1. L'amministratore apre l'applicazione ScuolaGO 2. L'amministratore clicca su "Registrati" 3. Il sistema richiede dei dati tramite form 4. L'amministratore compila il form e conferma 5. Il sistema controlla se non sono presenti altri amministratore con la stessa email 6. Il sistema registra l'amministratore e notifica la conferma
Estensioni	<p>5a) Nel sistema è già presente un amministratore con la stessa email inserita nel form:</p> <ol style="list-style-type: none"> 1. Il sistema notifica un errore 2. L'amministratore deve inserire un nuovo indirizzo email 3. Il sistema controlla la nuova email inserita
Requisiti Speciali	Nessuno
Tecnologie	Nessuna
Frequenza di ripetizione	Ogni volta che un amministratore vuole registrarsi al sistema

UC2: Gestisci professori

Nome del caso d'uso	UC2: Gestione professori
Portata	Applicazione ScuolaGO
Livello	Obiettivo amministratore
Attore primario	Amministratore e gestore dell'istituto
Parti interessate e interessi	-Amministratore: vuole aggiungere o rimuovere un professore dall'organico dell'istituto -Professore: vuole che venga effettivamente aggiunto all'organico, e che gli sia, quindi, garantito l'accesso al sistema
Pre-condizioni	L'amministratore deve aver effettuato il login nel sistema
Garanzia di successo	Viene aggiunto all'organico un nuovo professore
Scenario principale di successo	<ol style="list-style-type: none"> 1. L'amministratore clicca sul tab "Professori" 2. Il sistema mostra i professori già presenti nell'organico 3. Il sistema richiede i dati del nuovo professore tramite form 4. L'amministratore compila il form e conferma 5. Il sistema abbina un ID al professore e lo aggiunge all'organico
Estensioni	<p>a) L'amministratore vuole rimuovere un professore dall'organico</p> <ol style="list-style-type: none"> 1. L'amministratore seleziona un professore dall'elenco e clicca su "Rimuovi professore" 2. Il sistema elimina il professore dall'organico e tutte le sue lezioni
Requisiti Speciali	Nessuno
Tecnologie	Nessuna
Frequenza di ripetizione	Ogni volta che un amministratore vuole aggiungere o rimuovere un professore dall'organico.

UC3: Gestisci classi

Nome del caso d'uso	UC3: Gestione classi
Portata	Applicazione ScuolaGO
Livello	Obiettivo amministratore
Attore primario	Amministratore e gestore dell'istituto
Parti interessate e interessi	-Amministratore: vuole aggiungere o rimuovere una classe dall'istituto
Pre-condizioni	L'amministratore deve aver effettuato il login nel sistema
Garanzia di successo	Viene aggiunto all'istituto una nuova classe
Scenario principale di successo	<ol style="list-style-type: none"> 1. L'amministratore clicca sul tab "Classi" 2. Il sistema mostra le classi già presenti nell'istituto 3. Il sistema richiede i dati della nuova classe tramite form

	4. L'amministratore compila il form e conferma 5. Il sistema controlla che non siano presenti classi con lo stesso nome inserito nel form 6. Il sistema aggiunge la classe all'istituto
Estensioni	5a) Il sistema rileva una classe con lo stesso nome digitato nel form: 1. Il sistema notifica che la classe è già presente 2. L'amministratore cambia nome della classe 3. Il sistema controlla nuovamente il nome della classe a) L'amministratore vuole rimuovere una classe dall'istituto 1. L'amministratore seleziona una classe dall'elenco e clicca su "Rimuovi classe" 2. Il sistema elimina la classe dall'istituto, le sue lezioni e gli alunni che conteneva
Requisiti Speciali	Nessuno
Tecnologie	Nessuna
Frequenza di ripetizione	Ogni volta che un amministratore vuole aggiungere o rimuovere una classe dall'istituto

UC4: Inserimento nuovo alunno

Nome del caso d'uso	UC4: Inserimento nuovo alunno
Portata	Applicazione ScuolaGO
Livello	Obiettivo amministratore
Attore primario	Amministratore e gestore dell'istituto scolastico
Parti interessate e interessi	-Amministratore: vuole aggiungere un nuovo alunno ad una classe -Alunno: vuole che venga effettivamente aggiunto ad una classe, e che gli sia, quindi, garantito l'accesso al sistema
Pre-condizioni	L'amministratore deve aver effettuato il login nel sistema
Garanzia di successo	Viene aggiunto ad una classe un nuovo alunno
Scenario principale di successo	1. L'amministratore clicca sul tab "Alunni" 2. Il sistema mostra l'elenco delle classi e una tabella con tutti gli alunni già presenti nell'istituto 3. Il sistema richiede i dati del nuovo alunno tramite form 4. L'amministratore completa il form, seleziona una classe dall'elenco, e conferma l'aggiunta di un nuovo alunno alla classe 5. Il sistema controlla se per la classe selezionata non sia già stato raggiunto il limite massimo di alunni 6. Il sistema abbina un ID all'alunno e lo aggiunge alla classe selezionata

Estensioni	<p>5a) Il sistema rileva che per la classe selezionata è già stato raggiunto il limite massimo di alunni:</p> <ol style="list-style-type: none"> 1. Il sistema notifica il superamento della capienza per la classe 2. L'amministratore deve selezionare una nuova classe 3. Il sistema controlla nuovamente la capienza della classe
Requisiti Speciali	Nessuno
Tecnologie	Nessuna
Frequenza di ripetizione	Ogni volta che l'amministratore vuole aggiungere un alunno ad una classe.

UC5: Creazione lezione

Nome del caso d'uso	UC5: Creazione lezione
Portata	Applicazione ScuolaGO
Livello	Obiettivo amministratore
Attore primario	Amministratore e gestore dell'istituto scolastico
Parti interessate e interessi	-Amministratore: vuole che il calendario delle lezioni sia chiaro sia agli alunni che ai professori -Professore: vuole che non ci siano sue lezioni i cui orari si sovrappongano
Pre-condizioni	L'amministratore deve aver effettuato il login nel sistema
Garanzia di successo	Viene inserita nel calendario una nuova lezione
Scenario principale di successo	<ol style="list-style-type: none"> 1. L'amministratore clicca sul tab "Lezioni" 2. Il sistema mostra l'elenco dei professori nell'organico e delle classi nell'istituto 3. Il sistema richiede i dati della nuova lezione tramite form 4. L'amministratore seleziona un professore e una classe dagli elenchi, gli orari e la materia e conferma la nuova lezione 5. Il sistema verifica che non ci siano sovrapposizioni di altre lezioni col professore o con la classe selezionati 6. Il sistema aggiunge la lezione al calendario e notifica la conferma
Estensioni	<p>4a) Il sistema rileva una sovrapposizione tra le lezioni:</p> <ol style="list-style-type: none"> 1. Il sistema notifica la sovrapposizione 2. L'amministratore seleziona un nuovo orario 3. Il sistema controlla nuovamente se la lezione può essere aggiunta al calendario
Requisiti Speciali	Nessuno
Tecnologie	Nessuna
Frequenza di ripetizione	Ogni volta che un amministratore vuole creare una lezione.

UC6: Invio avviso

Nome del caso d'uso	UC6: Invio avviso
Portata	Applicazione ScuolaGO
Livello	Obiettivo amministratore
Attore primario	Amministratore e gestore dell'istituto scolastico
Parti interessate e interessi	-Amministratore: vuole inviare un avviso a delle classi in maniera affidabile -Alunno: vuole ricevere l'avviso tempestivamente
Pre-condizioni	L'amministratore deve aver effettuato il login nel sistema
Garanzia di successo	Viene aggiunto allo storico un nuovo avviso
Scenario principale di successo	<ol style="list-style-type: none"> 1. L'amministratore clicca sul tab "Avvisi" 2. Il sistema mostra un elenco delle classi dell'istituto 3. L'amministratore seleziona una o più classi dall'elenco, digita il corpo del messaggio e lo invia 4. Il sistema abbina all'avviso la data corrente e lo aggiunge allo storico avvisi
Estensioni	Estensioni non previste
Requisiti Speciali	Nessuno
Tecnologie	Nessuna
Frequenza di ripetizione	Ogni volta che un amministratore vuole inviare un avviso.

UC7: Gestione registro assenze

Nome del caso d'uso	UC7: Gestione registro assenze
Portata	Applicazione ScuolaGO
Livello	Obiettivo professore
Attore primario	Professore dell'organico dell'istituto
Parti interessate e interessi	-Professore: vuole facilmente e velocemente registrare l'assenza degli alunni
Pre-condizioni	Il professore deve aver eseguito il login nel sistema
Garanzia di successo	L'assenza degli alunni viene memorizzata nel registro assenze

Scenario principale di successo	<ol style="list-style-type: none"> 1. Il professore clicca sul tab “Registro assenze” 2. Il sistema mostra l’elenco delle classi nell’istituto 3. Il professore seleziona una classe 4. Il sistema mostra gli alunni della classe selezionata 5. Il professore seleziona uno o più alunni, e conferma le assenze 6. Il sistema abbina alla lista degli alunni assenti la data corrente 7. Il sistema controlla che non siano già state salvate assenze relative alla classe selezionata nella data corrente 8. Il sistema aggiunge al registro assenze le assenze confermate
Estensioni	<p>7a) Il sistema rileva che nella data corrente sono già state salvate assenze relative alla classe selezionata:</p> <ol style="list-style-type: none"> 1. Il sistema notifica che il registro è già stato compilato per quella classe 2. Il professore non può registrare le assenze
Requisiti Speciali	Nessuno
Tecnologie	Nessuna
Frequenza di ripetizione	Una volta al giorno per classe.

UC8: Gestione registro voti

Nome del caso d’uso	UC6: Gestione registro voti
Portata	Applicazione ScuolaGO
Livello	Obiettivo professore
Attore primario	Professore dell’organico dell’istituto
Parti interessate e interessi	-Professore: vuole facilmente e in maniera affidabile registrare i voti degli alunni -Alunno: vuole che i suoi voti vengano registrati correttamente
Pre-condizioni	Il professore deve aver eseguito il login nel sistema
Garanzia di successo	Il voto dell’alunno viene memorizzata nel registro voti
Scenario principale di successo	<ol style="list-style-type: none"> 1. Il professore clicca sul tab “Registro voti” 2. Il sistema mostra l’elenco delle lezioni del professore che ha effettuato il login 3. Il professore seleziona una lezione 4. Il sistema mostra gli alunni della classe in cui si tiene la lezione selezionata 5. Il professore seleziona un alunno, inserisce la valutazione e conferma 6. Il sistema aggiunge al registro voti il voto dell’alunno per la lezione selezionata

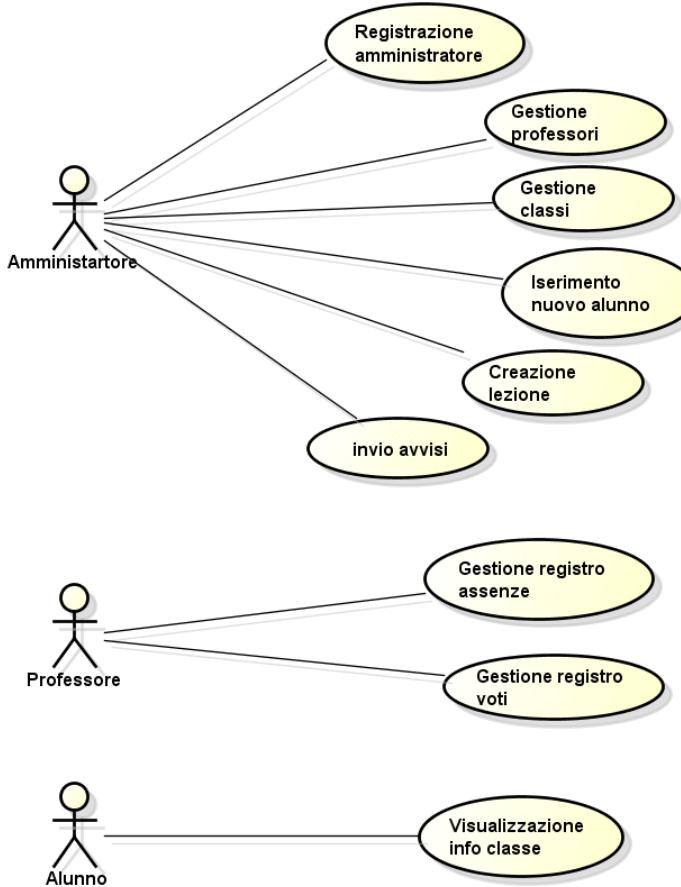
Estensioni	Nessuna estensione prevista.
Requisiti Speciali	Nessuno
Tecnologie	Nessuna
Frequenza di ripetizione	Ogni volta che il professore vuole registrare un voto per un alunno.

UC9: Visualizzazione info classe

Nome del caso d'uso	UC9: Visualizzazione info classe
Portata	Applicazione ScuolaGO
Livello	Obiettivo alunno
Attore primario	Alunno di una classe dell'istituto
Parti interessate e interessi	-Alunno: vuole che l'informazione della sua classe siano fruibili facilmente anche per un utente poco esperto
Pre-condizioni	L'alunno deve aver eseguito il login nel sistema
Garanzia di successo	L'alunno riesce a visionare correttamente le informazioni
Scenario principale di successo	<ol style="list-style-type: none"> 1. Il sistema mostra l'elenco degli studenti della classe dell'alunno che ha effettuato il login 2. Il sistema mostra gli avvisi relativi alla classe dell'alunno 3. Il sistema mostra il calendario delle lezioni della classe dell'alunno 4. Il sistema mostra i voti e la media dell'alunno
Estensioni	Nessuna estensione prevista.
Requisiti Speciali	Nessuno
Tecnologie	Nessuna
Frequenza di ripetizione	Ogni volta che l'alunno effettua il login al sistema.

1.4 Diagramma UML dei casi d'uso

Di seguito il diagramma UML dei casi d'uso:



1.5 Regole di dominio

- Al momento dell'aggiunta di un nuovo utente, alunno o professore, l'amministratore fornirà una password arbitraria. L'ID, invece, verrà assegnato dal sistema in maniera univoca. Questi due dati (ID, password) saranno utili agli utenti per effettuare il login.
- Possono essere presenti nel sistema alunni o professori con lo stesso nome, cognome e data di nascita, ma avranno un ID differente.
- Non è possibile avere due classi con lo stesso nome.
- I professori o le classi non possono avere più lezioni nello stesso orario.
- Non è possibile registrare le assenze di una classe più di una volta al giorno.
- Un professore può registrare le assenze di una classe che non riguarda le sue lezioni, ma non i voti.
- Ogni classe ha un numero massimo di alunni che può contenere. Per la legge che evita il sovraffollamento delle aule questo numero non può mai essere maggiore di 30.
- Ogni voto è un numero intero da 0 a 10.
- Tra le regole dell'istituto vi sono anche quelle che determinano il metodo con cui calcolare la media voti degli alunni, utile per la valutazione finale. Queste regole possono variare annualmente a discrezione del preside. Fino ad ora sono stati proposti due metodi: uno, più severo, che arrotonda la media per difetto e non tiene conto di voti riguardanti materie come

Religione e Educazione fisica, l'altro, più indulgente, che invece tiene conto di tutte le materie ed arrotonda per eccesso. È necessaria flessibilità a cambiamenti o ad aggiunte di nuovi metodi.

1.6 Specifiche Supplementari

-Usabilità: L'interfaccia grafica deve essere intuitiva e semplice, specialmente il lato sistema dedicato agli alunni

-Affidabilità: Il software deve essere affidabile e deve garantire la conservazione dei dati.

-Vicoli hardware e software: L'unico requisito di sistema per l'esecuzione del software è la presenza di Java Virtual Machine.

-Vicoli di sviluppo del software: Il linguaggio di programmazione utilizzato per il software è Java. Per l'interfaccia grafica si utilizza la libreria Java Swing.

-Aspetti legali: Le tecnologie utilizzate per la progettazione e realizzazione del sistema sono di tipo open source o freeware.

1.7 Glossario

- **Amministratore/Admin/Gestore istituto:** è il gestore dell'istituto scolastico o chiunque abbia il ruolo di amministratore del sistema (preside, vicepreside, addetto informatico etc.).
- **Professore:** è il professore che lavora nell'istituto ed è, dunque, presente nell'organico. Ha accesso al lato professore del sistema.
- **Organico:** insieme di professori che lavorano nell'istituto.
- **Orario:** orario scolastico, definito dal giorno settimanale e da un numero che identifica l'ora (*es. Lunedì 1: lunedì prima ora 8:00-9:00; Giovedì 5: giovedì quinta ora 12:00-13:00*).
- **Lezione:** corso di una materia tenuto da un professore in una classe in orari scolastici.
- **Calendario:** insieme di tutte le lezioni.
- **Alunno/Studente:** alunno di una classe dell'istituto. Ha accesso al lato alunno del sistema.
- **Classe:** insieme di alunni che condividono un'aula. Il nome della classe viene indicato con anno e sezione (*es. 1A,3C..*), ogni classe ha un differente numero massimo di alunni consentito.
- **Istituto:** struttura scolastica che raccoglie di tutte le classi.
- **Registro assenze:** memorizza una lista di alunni assenti giornalmente.
- **Registro voti:** memorizza i voti degli alunni relativi ad una lezione.

2 Documento di visione

2.1 Introduzione

Il progetto mira allo sviluppo di un software che permetta la gestione di un istituto scolastico e dei servizi scolastici tramite sistema informatico, quindi, il facile controllo dell'organico professori, delle classi, degli alunni e delle lezioni. I professori potranno accedere al sistema e consultare velocemente registro dei voti e delle assenze, così come gli alunni che potranno visionare i loro dati.

2.2 Scopo

Lo scopo di questo documento di visione è quello di evidenziare le caratteristiche e le funzionalità del software “ScuolaGO”, releggendo i dettagli al Modello dei casi d’uso.

2.3 Portata

Questo documento di visione è relativo al software “ScuolaGO”, sviluppato dallo studente D. Calabretta. L’obiettivo è quello di rispettare i requisiti richiesti dal committente.

2.4 Definizioni, acronimi e abbreviazioni

Per definizioni, acronimi e abbreviazioni vedere il Glossario (1.8)

2.5 Posizionamento

Opportunità di business: Il software consentirà una gestione informatizzata dei servizi scolastici, per questo motivo faciliterà il lavoro dell’amministratore e dei professori. Un obiettivo fondamentale del software è quello di garantire semplicità di utilizzo in maniera da servire più utenti possibili. Proprio per la sua comodità e per l’interfaccia intuitiva, ScuolaGO si differenzia da software concorrenti.

Formulazione del problema:

Descrizione del problema	Per controllare tutti i servizi scolastici possono essere necessari più software. Per un amministratore è difficoltoso gestire il calendario delle lezioni dell’intero istituto manualmente. I professori sono costretti a utilizzare registri cartacei. Avvisare gli alunni con le tradizionali circolari risulta un metodo inefficiente e inaffidabile. Inoltre, gli alunni e i loro genitori non possono visionare i voti da casa o consultare il calendario delle lezioni.
Attori coinvolti	Gestore istituto, professori, alunni
Impatto	Processo di gestione dei servizi scolastici soggetto a errori: possibili incomprensioni tra alunni e professori riguardo a voti o lezioni, errori di trascrizione per i registri cartacei, problemi tra amministratore e professori per la stesura del calendario, genitori dell’alunno insoddisfatti per l’impossibilità di tenere traccia dell’andamento scolastico del figlio o di eventuali comunicazioni.
Benefici in caso di successo	Gestione servizi scolastici facile e veloce, eliminazione totale di problemi dovuti ad incomprensioni con l’alunno, vantaggi di registri e calendario elettronici consultabili da casa e robusti ad errori umani.

Formulazione della posizione del prodotto:

Destinatari	Il software è destinato ad un istituto scolastico.
Obiettivi	Il committente necessita di un sistema capace di informatizzare la gestione dei servizi scolastici utilizzabile sia dal gestore che da alunni e professori.
Tipologia	Software personalizzato.
Funzione	Sistema di gestione istituto scolastico.
Soluzioni alternative attuali	Soluzioni simili sono disponibili sul mercato, ma non con le stesse caratteristiche di "ScuolaGO".
Caratteristiche prodotto	Il prodotto, grazie alla semplice interfaccia, è di facile utilizzo e permette di rispettare i requisiti del committente e tutte le esigenze di professori meno avvezzi alla tecnologia, o di utenti in età scolare.

2.6 Parti interessate e descrizioni utente

Riepilogo delle parti interessate:

Committente: chi ha commissionato lo sviluppo del software e che fornisce agli sviluppatori i requisiti di progettazione.

Sviluppatore software: si occupa della creazione del software, cerca di rispettare i requisiti richiesti dal committente.

Riepilogo dell'utente:

Amministratore: gestore istituto che utilizza il software per il controllo dei servizi scolastici.

Professore: utilizza il software per aggiornare registro assenze e registro voti.

Alunno: utilizza il software per visionare avvisi, calendario delle lezioni e voti.

Ambiente dell'utente:

- Il sistema sarà utilizzato dal gestore dell'istituto, dal professore o dall'alunno.
- L'interfaccia deve essere semplice ed intuitiva.

3 Analisi iterativa

3.1 Riepilogo iterazioni

La realizzazione del software è stata affrontata con approccio iterativo, ciò ha permesso la graduale modifica del progetto ed implementazione delle funzionalità, garantendo, così, una più facile gestione delle problematiche di progettazione e di eventuali correzioni del software.

Di seguito il dettaglio delle iterazioni:

- **Iterazione 1:**

Implementazione delle funzionalità che garantiscono l'accesso al sistema all'amministratore e l'inserimento di un nuovo alunno, per lo sviluppo di quest'ultimo caso d'uso si è considerato necessario occuparsi anche della gestione delle classi, in quanto, il nuovo alunno deve essere inserito in una classe già creata.

-Vengono create le classi: *Admin, Istituto, Classe, Alunno*.

-*UC1: Registrazione admin, UC3 Gestione classi, UC4 Inserimento nuovo alunno*

- **Iterazione 2:**

In questa iterazione si è scelto di implementare le funzionalità di gestione professore e creazione lezione. La presenza di queste funzionalità è stata ritenuta indispensabile per lo sviluppo degli ulteriori casi d'uso.

-Vengono aggiunte le classi: *OrganicoProfessori, Professore, Calendario, Lezione-*

-*UC2: Gestione professore, UC5 Creazione lezione.*

- **Iterazione 3:**

Implementazione dei casi d'uso riguardanti le attività permesse al professore.

-Vengono aggiunte le classi: *RegistroAssenze, Assenze, RegistroVoti, Voto.*

-*UC7: Gestione registro assenze, UC8 Gestione registro voti.*

- **Iterazione 4:**

Questa iterazione è stata riservata ai restanti casi d'uso che erano stati ritenuti meno importanti all'interno del sistema: la possibilità, per l'amministratore, di inviare avvisi a classi e il caso d'uso riservato all'alunno.

-Vengono aggiunte le classi: *StoricoAvvisi, Avviso*

-*UC6: Invio avviso, UC9 Visualizzazione info classe.*

- **Iterazione 5:**

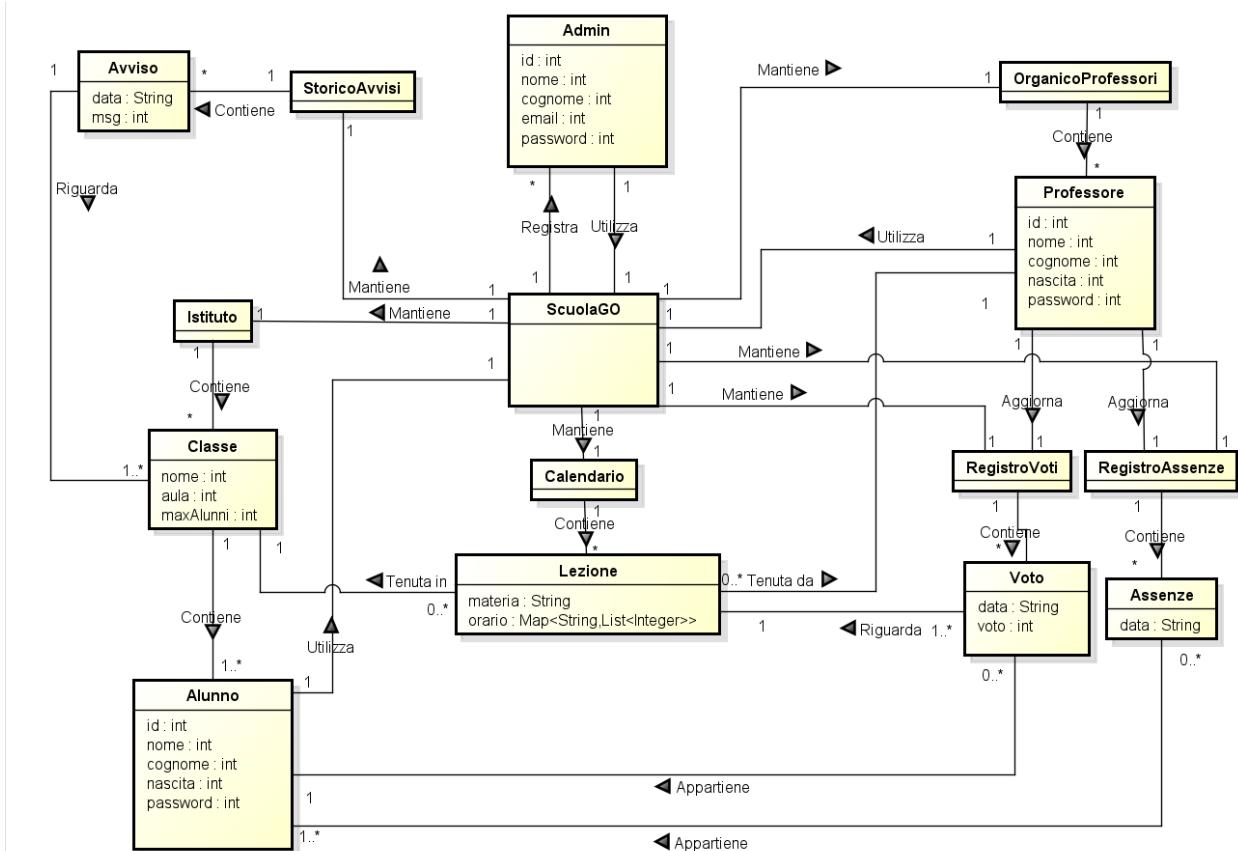
L'ultima iterazione ha consentito di procedere allo sviluppo degli scenari alternativi dei casi d'uso 2 e 3, in particolare, alla funzionalità di rimozione professori e classi. Inoltre, si è notato come le classi *Admin, Professore* e *Alunno* fossero molto simili, questa ridondanza viene risolta aggiungendo una classe astratta *Utente* e sfruttando l'ereditarietà.

-*UC2: Gestione professore (alternativo), UC3 Gestione classi (alternativo).*

3.2 Modello di dominio

In questo capitolo viene illustrato il modello di dominio, un grafico che evidenzia le classi concettuali del progetto, i loro attributi e le relazioni principali tra di esse.

Di seguito viene mostrato il modello di dominio risultante alla fine dell'ultima iterazione. Per visionare i modelli nelle altre iterazioni consultare la cartella "Documentazione".



Classi concettuali:

- **ScuolaGO**: rappresenta il sistema ScuolaGO
- **Admin**: amministratore del sistema
- **Istituto**: contiene le classi scolastiche
- **Classe**: rappresenta una classe dell'istituto e contiene gli alunni
- **Alunno**: alunno di una classe dell'Istituto
- **Calendario**: contiene le lezioni
- **Lezione**: lezione di una materia tenuta da un professore in una classe
- **OrganicoProfessori**: contiene i professori che lavorano all'istituto
- **Professore**: professore dell'organico
- **RegistroVoti**: contiene i voti degli alunni
- **Voto**: valutazione di un alunno in una lezione
- **RegistroAssenze**: contiene le assenze
- **Assenze**: contiene lista di alunni assenti giornalmente
- **StoricoAvvisi**: contiene gli avvisi
- **Avviso**: avviso inviato dall'amministratore ad una lista di classi

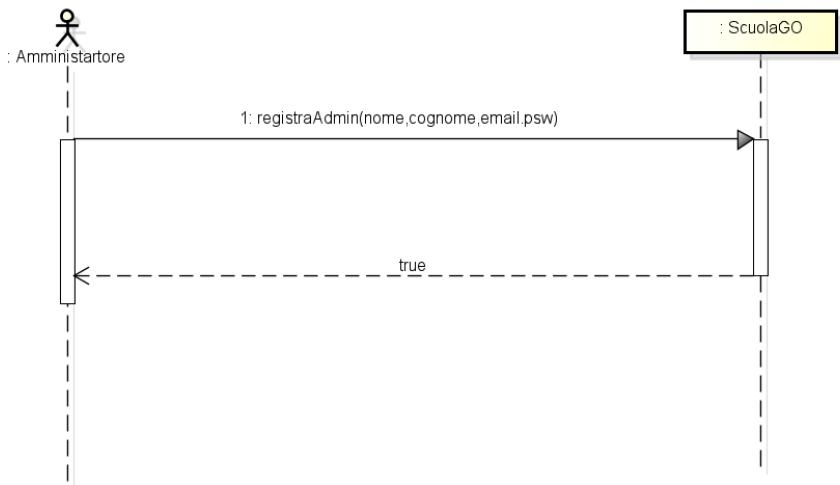
3.3 SSD e contratti

Durante le iterazioni sono stati creati i diagrammi di sequenza, che mostrano l'interazione tra l'utente e il sistema per ogni caso d'uso, e i contratti che descrivono le operazioni contenute nei diagrammi.

❖ Iterazione 1:

UC1: Registrazione Amministratore

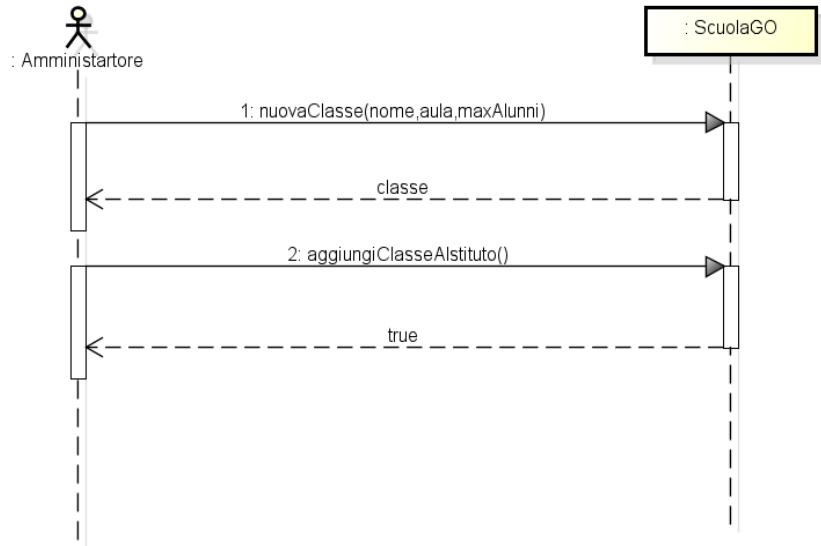
SSD UC1:



Contratti UC1:	
Nome	CO1: registraAdmin
Operazione	registraAdmin(nome:String ,cognome:String ,email:String, password:String)
Riferimenti	Caso d'uso 1: Registrazione amministratore
Pre-condizioni	
Post-condizioni	Viene creata un'istanza <i>admin</i> di Admin con tutti gli attributi forniti dall'utente.

UC3: Gestione classi

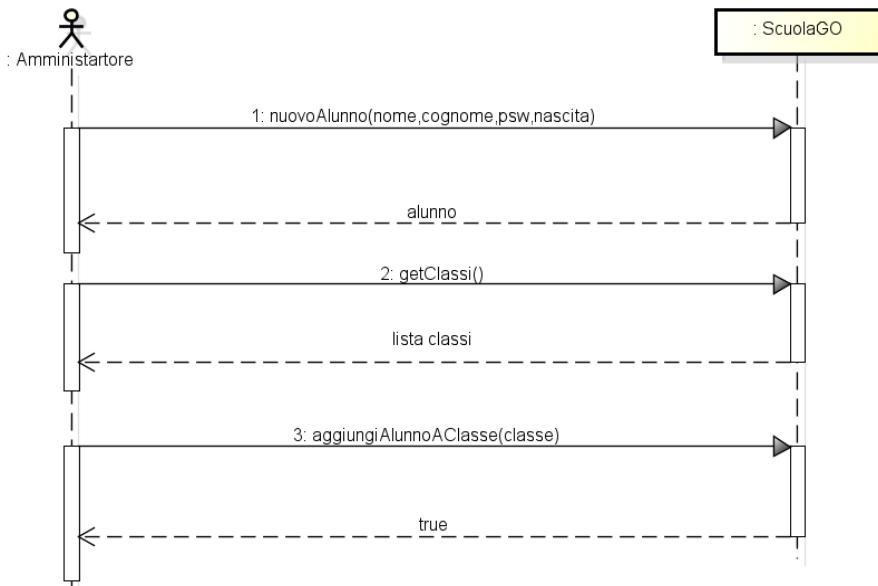
SSD UC3:



Contratti UC3:	
Nome	CO1: nuovaClasse
Operazione	<code>nuovaClasse(nome:String, aula:String, maxAlunni:int)</code>
Riferimenti	Caso d'uso 3: Gestione classe
Pre-condizioni	L'amministratore ha effettuato il login
Post-condizioni	Viene creata un'istanza <i>classeCorrente</i> di Classe tramite Istituto con gli attributi previsti.
Nome	CO2: aggiungiClasseAlstituto
Operazione	<code>aggiungiClasseIstituto()</code>
Riferimenti	Caso d'uso 3: Gestione classe
Pre-condizioni	È presente l'istanza <i>classeCorrente</i> di Classe
Post-condizioni	<i>classeCorrente</i> viene salvato nella lista classi di Istituto ed associato ad esso tramite associazione "contiene".

UC4: Inserimento nuovo alunno

SSD UC4:

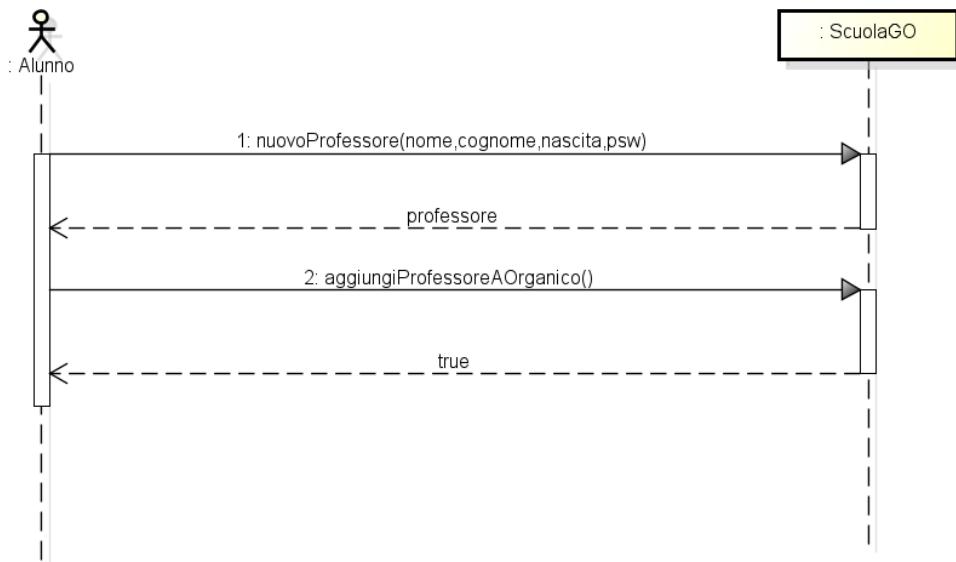


Contratti UC4:	
Nome	CO1: nuovoAlunno
Operazione	nuovaAlunno(nome:String,aula:String,password:String, nascita:String)
Riferimenti	Caso d'uso 4: Inserimento nuovo alunno
Pre-condizioni	L'amministratore ha effettuato il login
Post-condizioni	Viene creata un'istanza <i>alunnoCorrente</i> di Alunno tramite Classe con parte degli attributi
Nome	CO2: getClassi
Operazione	getClassi()
Riferimenti	Caso d'uso 4: Inserimento nuovo alunno
Pre-condizioni	
Post-condizioni	Il sistema recupera la lista di classi presenti nell'istituto tramite Istituto
Nome	CO3: aggiungiAlunnoAClasse
Operazione	aggiungiAlunnoAClasse(classe: Classe)
Riferimenti	Caso d'uso 4: Inserimento nuovo alunno
Pre-condizioni	-È presente l'istanza <i>alunnoCorrente</i> di Alunno. -L'utente ha selezionato un'istanza <i>classe</i> di Classe
Post-condizioni	<i>alunnoCorrente</i> viene aggiunto alla lista alunni di <i>classe</i> ed associato ad essa tramite associazione "contiene".

❖ Iterazione 2:

UC2: Gestione classi

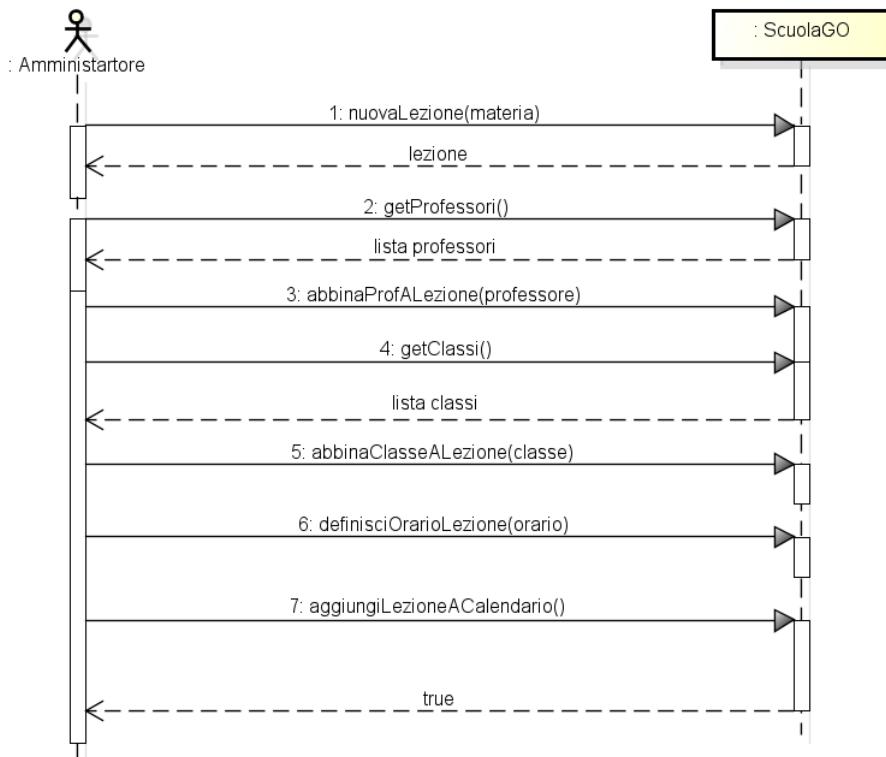
SSD UC2:



Contratti UC2:	
Nome	CO1: nuovoProfessore
Operazione	nuovoProfessore (nome,cognome,data di nascita,password)
Riferimenti	Caso d'uso 2: Gestione professore
Pre-condizioni	L'amministratore ha effettuato il login
Post-condizioni	Viene creata un'istanza <i>professoreCorrente</i> di Professore tramite OrganicoProfessori con gli attributi previsti
Nome	CO2: aggiungiProfessoreAOrganico
Operazione	aggiungiProfessoreAOrganico()
Riferimenti	Caso d'uso 2: Gestione professore
Pre-condizioni	È presente un'istanza <i>professoreCorrente</i> di Professore
Post-condizioni	<i>professoreCorrente</i> viene salvato nella lista professori di OrganicoProfessori ed associato ad esso tramite l'associazione "contiene".

UC5: Creazione lezione

SSD UC5:



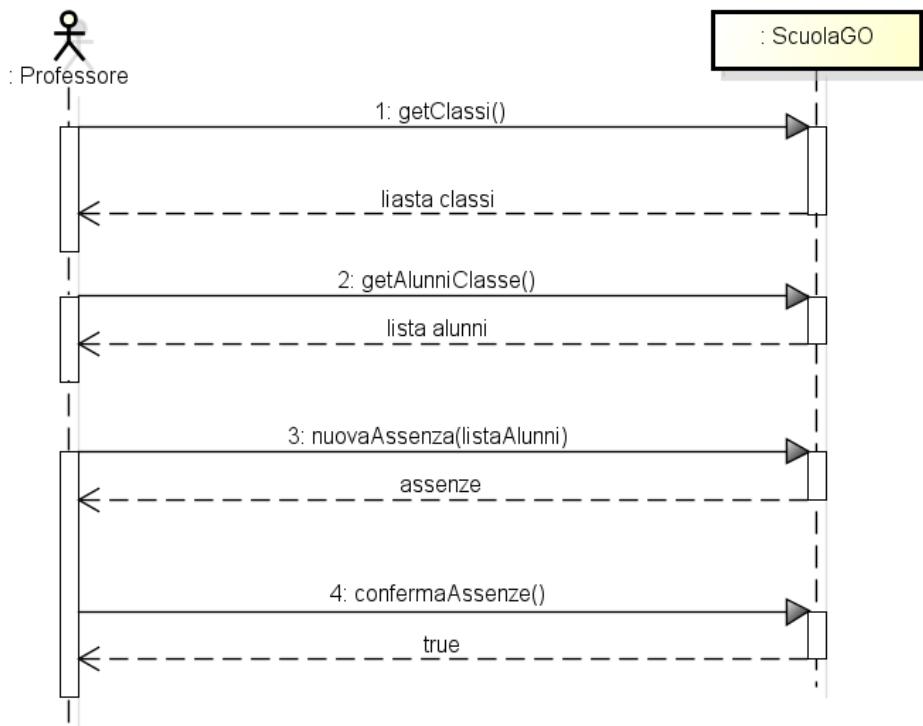
Contratti UC5:	
Nome	CO1: nuovaLezione
Operazione	nuovaLezione (materia:String)
Riferimenti	Caso d'uso 5: Creazione lezione
Pre-condizioni	L'amministratore ha effettuato il login
Post-condizioni	Viene creata un'istanza <i>lezioneCorrente</i> di Lezione tramite Calendario con l'attributo materia
Nome	CO2: getProfessori
Operazione	getProfessori()
Riferimenti	Caso d'uso 5: Creazione lezione
Pre-condizioni	
Post-condizioni	Il sistema recupera l'elenco di professori tramite OrganicoProfessori
Nome	CO3: abbinaProfALezione
Operazione	abbinaProfALezione(professore:Professore)
Riferimenti	Caso d'uso 5: Creazione lezione
Pre-condizioni	-È presente l'istanza <i>lezioneCorrente</i> di Lezione -L'utente ha selezionato un'istanza di Professore
Post-condizioni	Viene associata <i>lezioneCorrente</i> a l'istanza <i>professore</i> di Professore selezionata dall'utente tramite associazione "tenuta da".

Nome	CO4: getClassi
Operazione	getClassi()
Riferimenti	Caso d'uso 5: Creazione lezione
Pre-condizioni	Il professore ha effettuato il login
Post-condizioni	Il sistema recupera la lista di classi tramite Istituto
Nome	CO5: abbinaCasseALezione
Operazione	abbinaClasseALezione(classe: Classe)
Riferimenti	Caso d'uso 5: Creazione lezione
Pre-condizioni	-È presente l'istanza <i>lezioneCorrente</i> di Lezione . L'utente ha selezionato un'istanza <i>classe</i> di Classe
Post-condizioni	Viene associata <i>lezioneCorrente</i> a l'istanza <i>classe</i> di Classe selezionata dall'utente. tramite associazione "tenuta in"
Nome	CO6: definisciOrarioLezione
Operazione	definisciOrarioLezione (orario:Map<String,Integer>)
Riferimenti	Caso d'uso 5: Creazione lezione
Pre-condizioni	È presente l'istanza <i>lezioneCorrente</i> di Lezione
Post-condizioni	Viene aggiornato l'attributo orario di Lezione con l'orario selezionato dall'utente
Nome	CO7: aggiungiLezioneACalendario
Operazione	aggiungiLezioneACalendario()
Riferimenti	Caso d'uso 5: Creazione lezione
Pre-condizioni	È presente l'istanza <i>lezioneCorrente</i> di Lezione
Post-condizioni	<i>lezioneCorrente</i> viene salvato nella lista lezioni di Calendario ed associato ad esso tramite l'associazione "contiene".

❖ Iterazione 3:

UC7: Gestione registro assenze

SSD UC7:

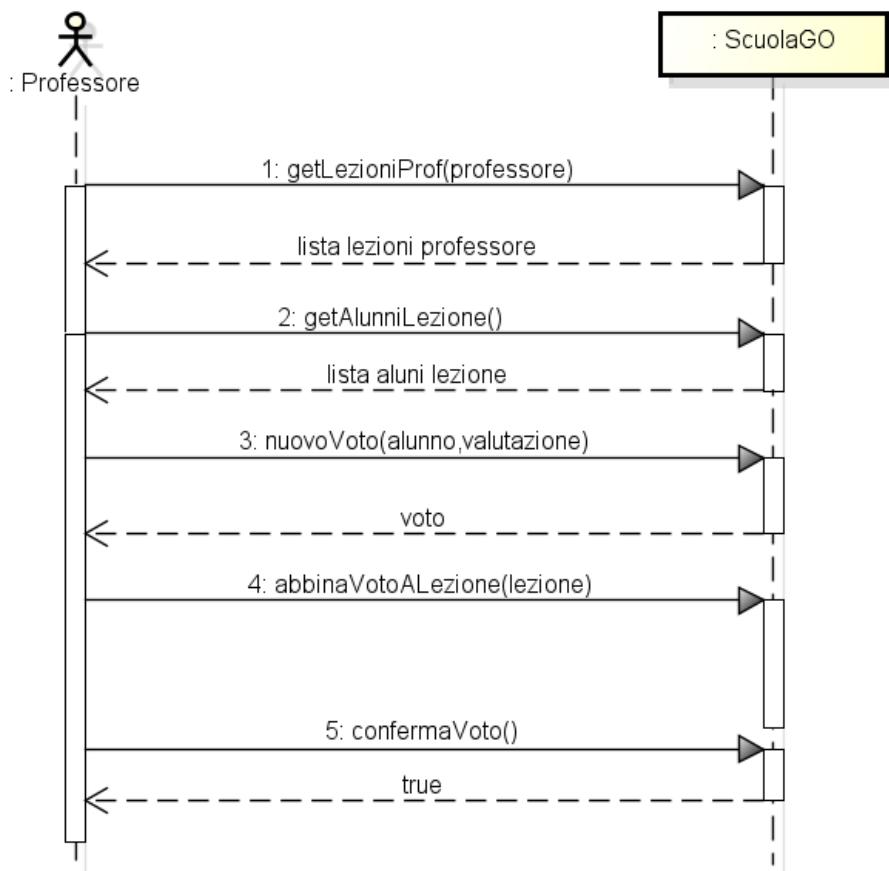


Contratti UC7:	
Nome	CO1: <code>getClassi</code>
Operazione	<code>getClassi()</code>
Riferimenti	Caso d'uso 7: Gestione registro assenze
Pre-condizioni	Il professore ha effettuato il login
Post-condizioni	Il sistema recupera la lista di classi tramite Istituto
Nome	CO2: <code>getAlunniClasse</code>
Operazione	<code>getAlunniClasse ()</code>
Riferimenti	Caso d'uso 7: Gestione registro assenze
Pre-condizioni	L'utente ha selezionato un'istanza <i>classe</i> di Classe
Post-condizioni	Il sistema recupera la lista di alunni tramite l'istanza <i>classe</i> di Classe selezionata dall'utente
Nome	CO3: <code>nuovaAssenza</code>
Operazione	<code>nuovaAssenza(listaAlunni: List<Alunno>)</code>
Riferimenti	Caso d'uso 7: Gestione registro assenze
Pre-condizioni	L'utente ha selezionato la <i>listaAlunni</i> , una lista di istanze di Alunno

Post-condizioni	<ul style="list-style-type: none"> -Viene creata l'istanza assenzeCorrente di Assenze, tramite RegistroAssenze, con l'attributo listaAlunni. - assenzeCorrente viene associata ad ogni istanza di Alunno in listaAlunni tramite associazione “appartiene”. -Viene aggiornato l'attributo data di Assenze con la data corrente -Viene aggiornato l'attributo classe di Assenze con la classe degli alunni selezionati
Nome	CO4: confermaAssenze
Operazione	confermaAssenze()
Riferimenti	Caso d'uso 7: Gestione registro assenze
Pre-condizioni	Esiste un'istanza assenzeCorrente di Assenze
Post-condizioni	assenzeCorrente viene salvato nella lista assenze di RegistroAssenze ed associato ad esso tramite l'associazione “contiene”.

UC8: Gestione registro voti

SSD UC8:



Contratti UC8:

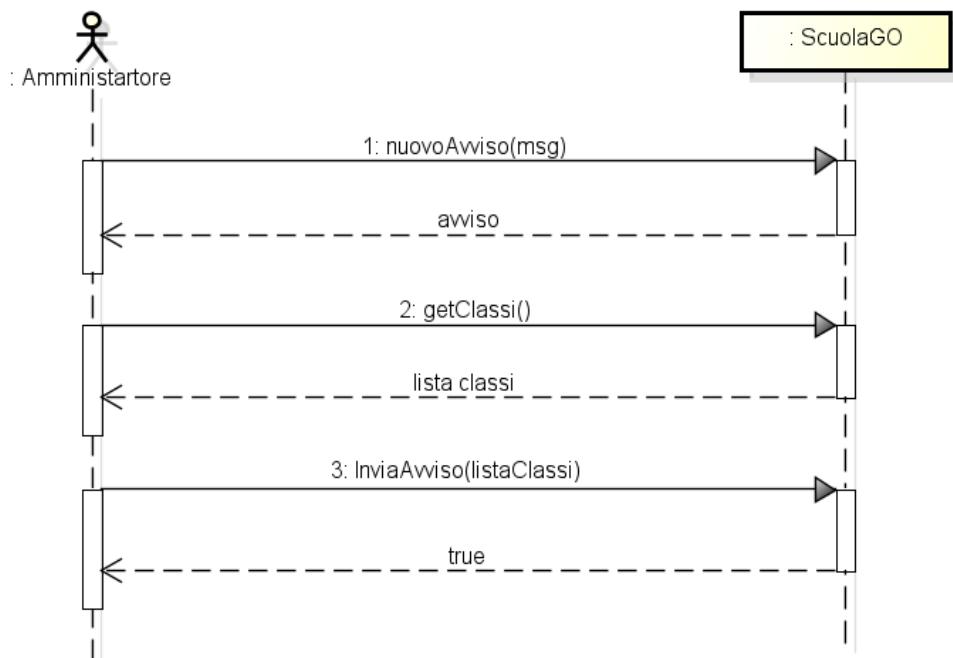
Nome	CO1: getLezioniProf
Operazione	getLezioniProf (professore:Professore)

Riferimenti	Caso d'uso 8: Gestione registro voti
Pre-condizioni	-Il professore ha effettuato il login -Viene recuperata l'istanza <i>prof_login</i> di Professore tramite login
Post-condizioni	Il sistema recupera la lista di lezioni tramite l'istanza <i>prof_login</i> di Professore
Nome	CO2: getAlunniLezione
Operazione	getAlunniLezione()
Riferimenti	Caso d'uso 8: Gestione registro voti
Pre-condizioni	L'utente ha selezionato un'istanza <i>lezione</i> di Lezione
Post-condizioni	Il sistema recupera la lista di alunni tramite l'istanza <i>lezione</i> di Lezione selezionata dall'utente
Nome	CO3: nuovoVoto
Operazione	nuovoVoto(alunno:Alunno, valutazione:int)
Riferimenti	Caso d'uso 8: Gestione registro voti
Pre-condizioni	L'utente ha selezionato un'istanza <i>alunno</i> di Alunno dalla lista
Post-condizioni	- Viene creata l'istanza <i>votoCorrente</i> di Voto tramite RegistroVoto con l'attributo <i>alunno</i> selezionato dall'utente e l'attributo <i>valutazione</i> . - <i>votoCorrente</i> viene associato ad <i>alunno</i> tramite associazione "appartiene" - Viene aggiornato l'attributo <i>data</i> di Voto con la data corrente
Nome	CO4: abbinaVotoALezione
Operazione	abbinaVotoALezione(lezione:Lezione)
Riferimenti	Caso d'uso 8: Gestione registro voti
Pre-condizioni	-Esiste un'istanza <i>votoCorrente</i> di Voto
Post-condizioni	<i>votoCorrente</i> viene associato all'istanza <i>lezione</i> di Lezione selezionata dall'utente tramite associazione "riguarda"
Nome	CO5: confermaVoto
Operazione	confermaVoto()
Riferimenti	Caso d'uso 5
Pre-condizioni	Esiste un'istanza <i>votoCorrente</i> di Voto
Post-condizioni	<i>votoCorrente</i> viene salvato nella lista voti di RegistroVoti ed associato ad esso tramite l'associazione "contiene".

❖ Iterazione 4:

UC6: Invio avviso

SSD UC6:

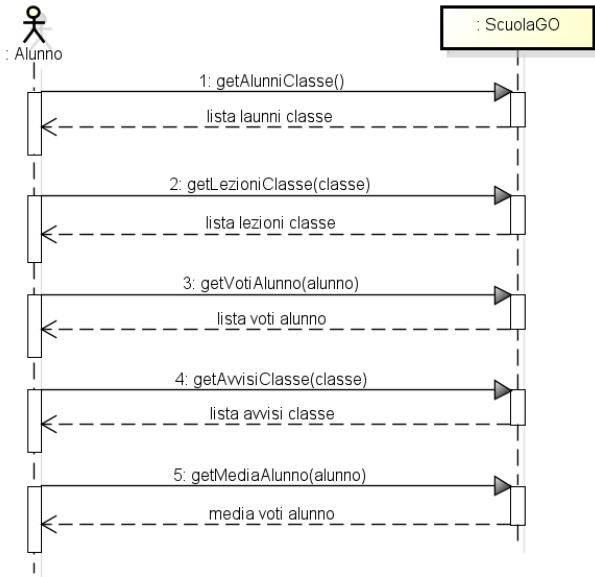


Contratti UC6:	
Nome	CO1: nuovoAvviso
Operazione	nuovoAvviso(msg:String)
Riferimenti	Caso d'uso 6: Invia avviso
Pre-condizioni	Il professore ha effettuato il login
Post-condizioni	Viene creata un'istanza <i>avvisoCorrente</i> di Avviso StoricoAvvisi con l'attributo msg
Nome	CO2: getClassi
Operazione	getClassi()
Riferimenti	Caso d'uso 6: Invia avviso
Pre-condizioni	
Post-condizioni	Il sistema recupera la lista di classi tramite Istituto
Nome	CO3: InviaAvviso
Operazione	inviAvviso(listaClassi: List<Classe>)
Riferimenti	Caso d'uso 6: Invia avviso
Pre-condizioni	L'utente ha selezionato la <i>listaClassi</i> , una lista di istanze di Classe

Post-condizioni	<ul style="list-style-type: none"> -Ogni istanza di Classe di listaClassi viene associata ad avvisoCorrente tramite associazione “riguarda” -Viene aggiornato l'attributo data di <i>avvisoCorrente</i> con la data corrente
------------------------	--

UC9: Visualizzazione info classe

SSD UC9:



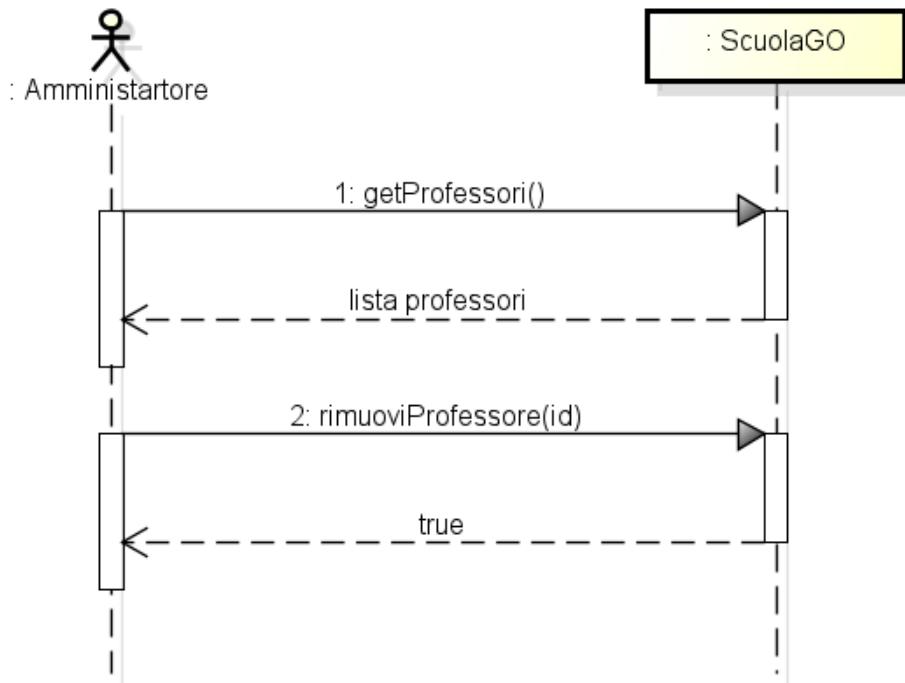
Contratti UC9:	
Nome	CO1: <i>getAlunniClasse</i>
Operazione	<i>getAlunniClasse()</i>
Riferimenti	Caso d'uso 9: Visualizzazione info classe
Pre-condizioni	<ul style="list-style-type: none"> -L'alunno ha effettuato il login -Viene recuperata l'istanza <i>alunno_login</i> di Alunno tramite login -Viene recuperata l'istanza <i>miaClasse</i> di Classe tramite <i>alunno_login</i>
Post-condizioni	Il sistema recupera la lista di alunni della classe tramite l'istanza <i>miaClasse</i> di Classe
Nome	CO2: <i>getLezioniClasse</i>
Operazione	<i>getLezioniClasse(miaClasse: Classe)</i>
Riferimenti	Caso d'uso 9: Visualizzazione info classe
Pre-condizioni	Sono disponibili le istanze <i>alunno_login</i> e <i>miaClasse</i>
Post-condizioni	Il sistema recupera la lista di lezioni della classe tramite Calendario, utilizzando l'istanza <i>miaClasse</i> di Classe
Nome	CO3: <i>getVotiAlunno</i>

Operazione	getVotiAlunno(alunno: Alunno)
Riferimenti	Caso d'uso 9: Visualizzazione info classe
Pre-condizioni	Sono disponibili le istanze <i>alunno_login</i> e <i>miaClasse</i>
Post-condizioni	Il sistema recupera la lista di voti dell'alunno tramite RegistroVoti, utilizzando l'istanza <i>alunno_login</i>
Nome	CO4: getAvvisiClasse
Operazione	getAvvisiClasse(classe: Classe)
Riferimenti	Caso d'uso 6
Pre-condizioni	Sono disponibili le istanze <i>alunno_login</i> e <i>miaClasse</i>
Post-condizioni	Il sistema recupera la lista di avvisi per la classe tramite StoricoAvvisi, utilizzando l'istanza <i>miaClasse</i>
Nome	CO5: getMediaAlunno
Operazione	getMediaAlunno(alunno:Alunno)
Riferimenti	Caso d'uso 6
Pre-condizioni	Sono disponibili le istanze <i>alunno_login</i> e <i>miaClasse</i>
Post-condizioni	Il sistema restituisce la media voti dell'alunno tramite RegistroVoti, utilizzando l'istanza <i>alunno_login</i> . RegistroVoti utilizza un'apposita strategia per il calcolo.

❖ Iterazione 5:

UC2 (alternativo): Gestione professori

SSD UC2 (alternativo):

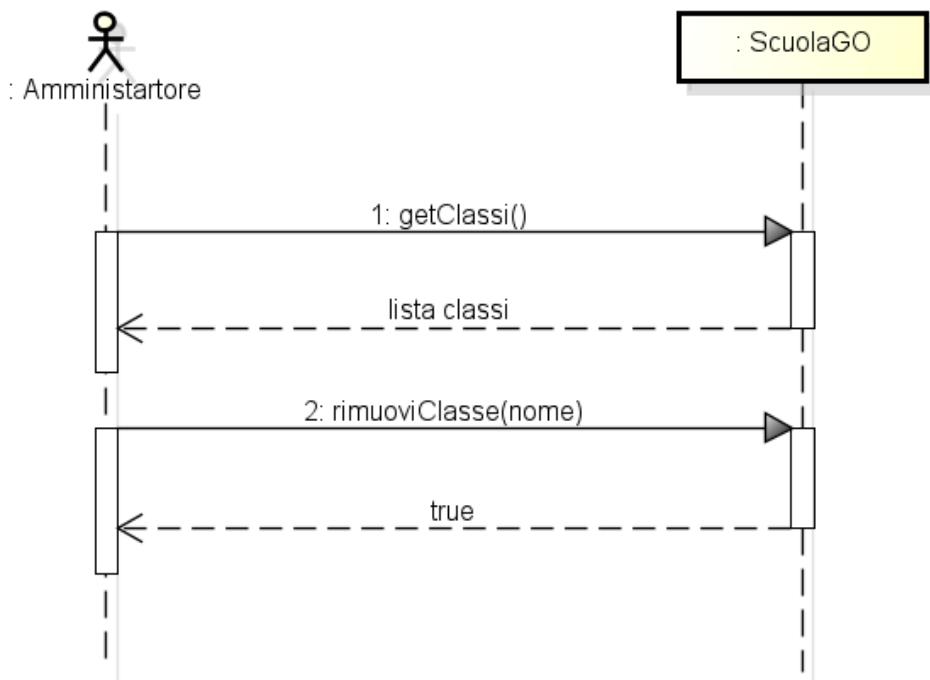


Contratti UC2 (alternativo):

Nome	CO1: getProfessori
Operazione	getProfessori()
Riferimenti	Caso d'uso 2 alternativo: Gestione professori (rimozione)
Pre-condizioni	L'amministratore ha effettuato il login
Post-condizioni	Il sistema recupera la lista di professori tramite OrganicoProfessori
Nome	CO2: rimuoviProfessore
Operazione	rimuoviProfessore(id: int)
Riferimenti	Caso d'uso 2 alternativo: Gestione professori (rimozione)
Pre-condizioni	L'utente ha selezionata un'id professore dalla lista professori
Post-condizioni	-Il sistema rimuove dal Calendario le lezioni del professore con l'id selezionato -Il sistema rimuove dall' OrganicoProfessori il professore con l'id selezionato

UC3 (alternativo): Gestione classi

SSD UC3 (alternativo):



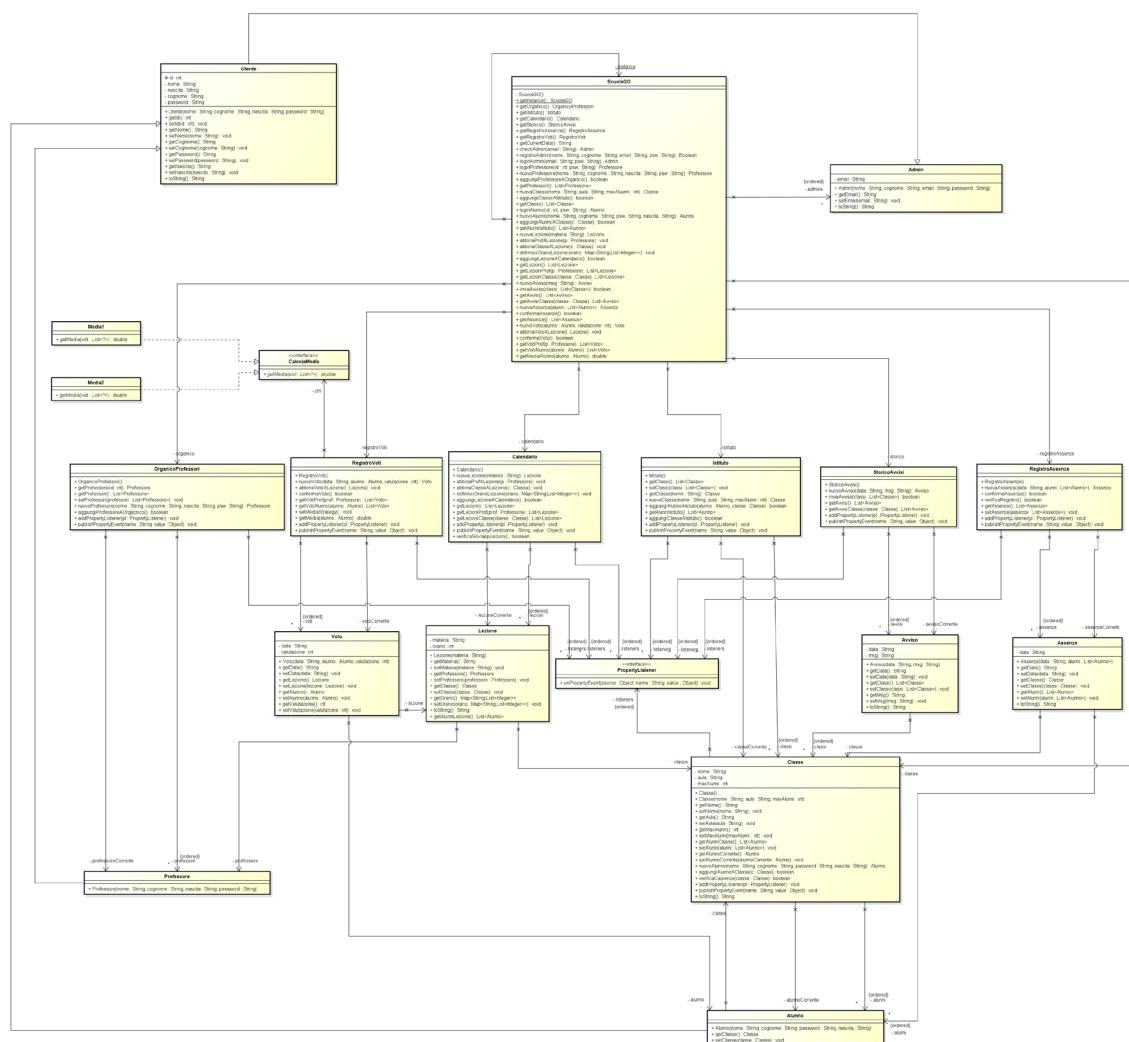
Contratti UC3 (alternativo):

Nome	CO1: getClassi
Operazione	getProfessori()
Riferimenti	Caso d'uso 3 alternativo: Gestione professori (rimozione)
Pre-condizioni	L'amministratore ha effettuato il login
Post-condizioni	Il sistema recupera la lista di classi tramite Istituto
Nome	CO2: rimuoviClasse
Operazione	rimuoviClasse(nome: String)
Riferimenti	Caso d'uso 3 alternativo: Gestione professori (rimozione)
Pre-condizioni	L'utente ha selezionata un nome classe dalla lista classi
Post-condizioni	-Il sistema rimuove dal Calendario le lezioni della classe con il nome selezionato -Il sistema rimuove dall' Istituto la classe con il nome selezionato

4 Progettazione

4.1 Diagramma delle classi

Un elemento fondamentale nella fase di progettazione è la definizione degli oggetti software. Il diagramma delle classi seguente rappresenta il punto di vista statico del progetto risultante dopo l'ultima iterazione, vengono mostrate dettagliatamente le classi e le relazioni. Per visionare il diagramma delle altre iterazioni consultare la cartella “Documentazione”.

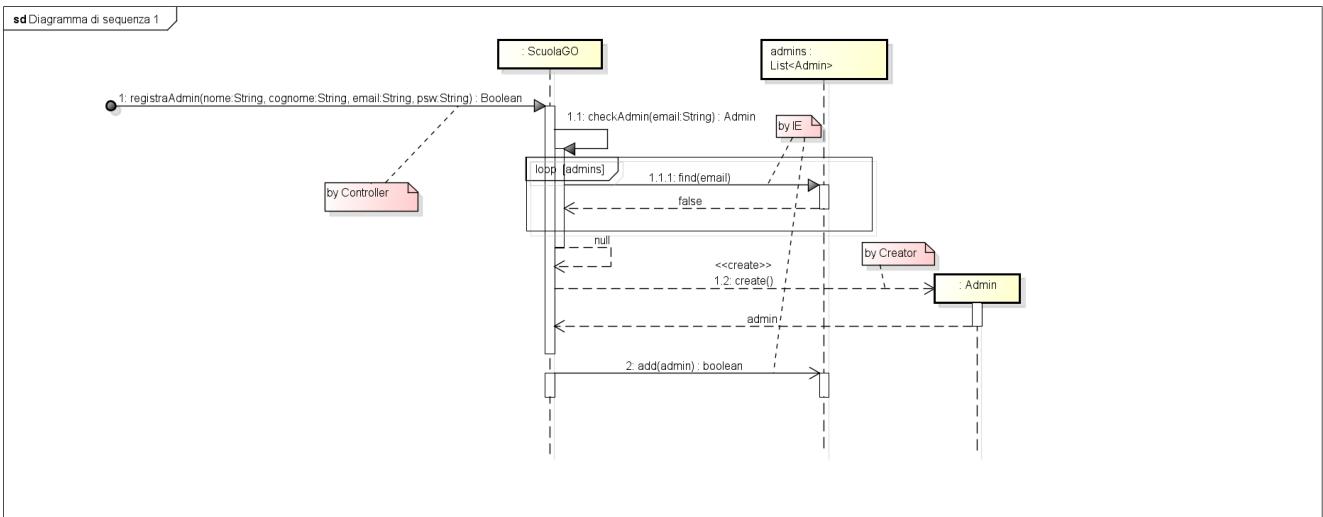


4.2 Diagrammi di sequenza

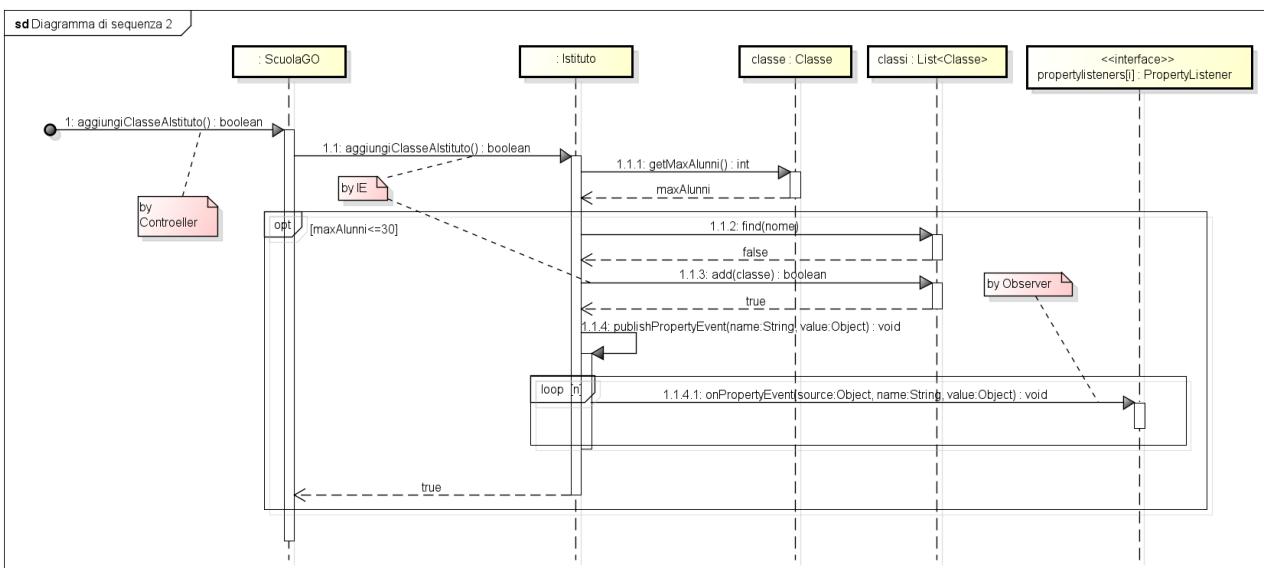
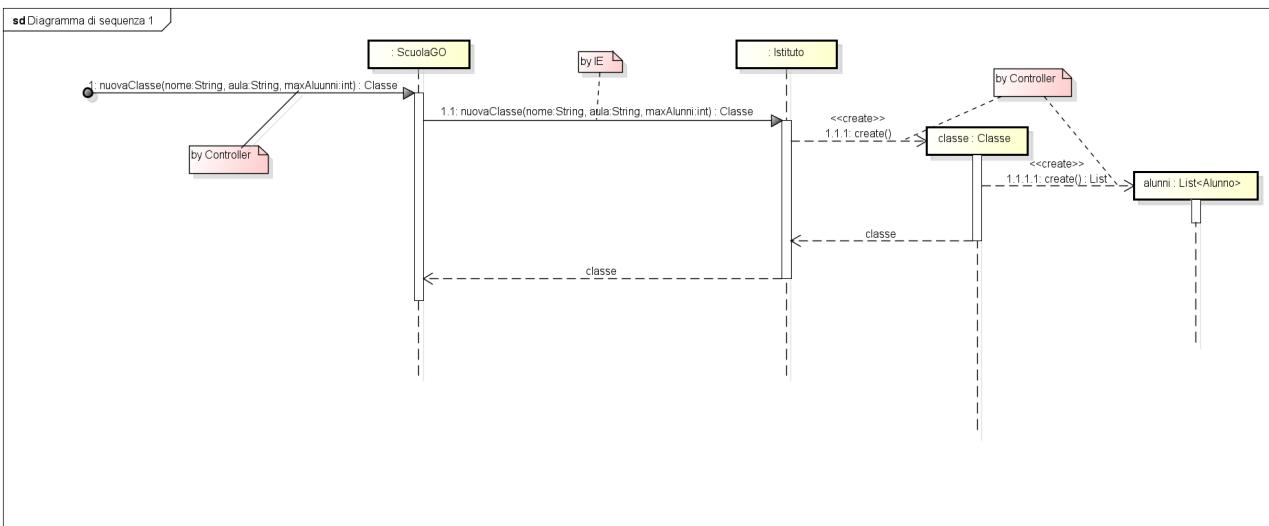
In questo capitolo viene analizzato il punto di vista dinamico del progetto, mostrando l'interazione tra gli oggetti del sistema per ogni caso d'uso. Di seguito sono presenti i diagrammi di sequenza secondo l'ordine di iterazione.

❖ Iterazione 1

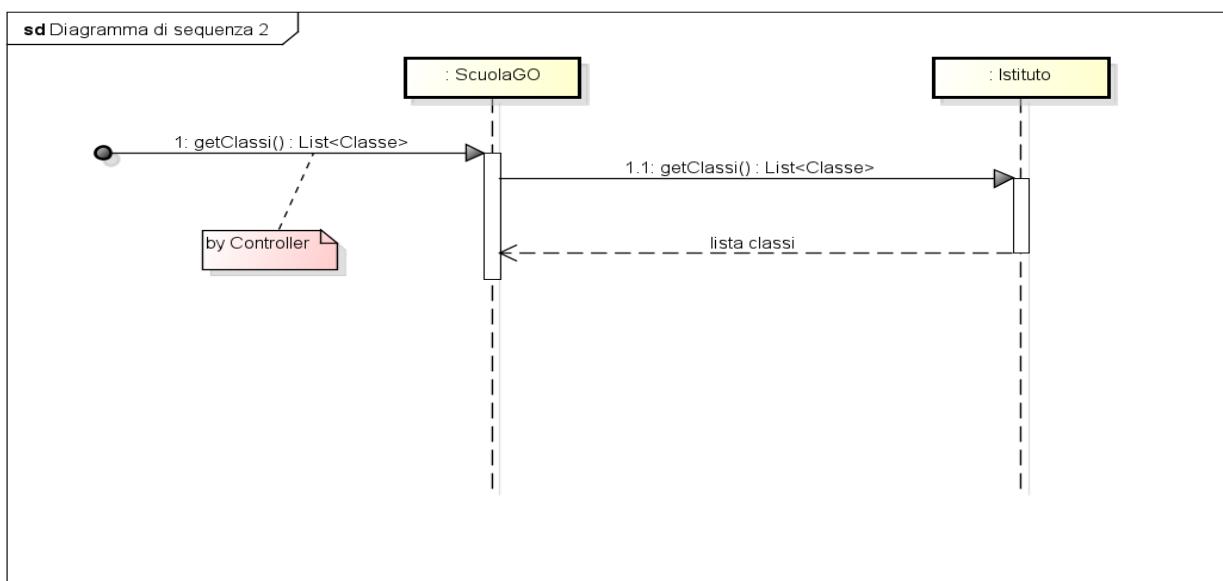
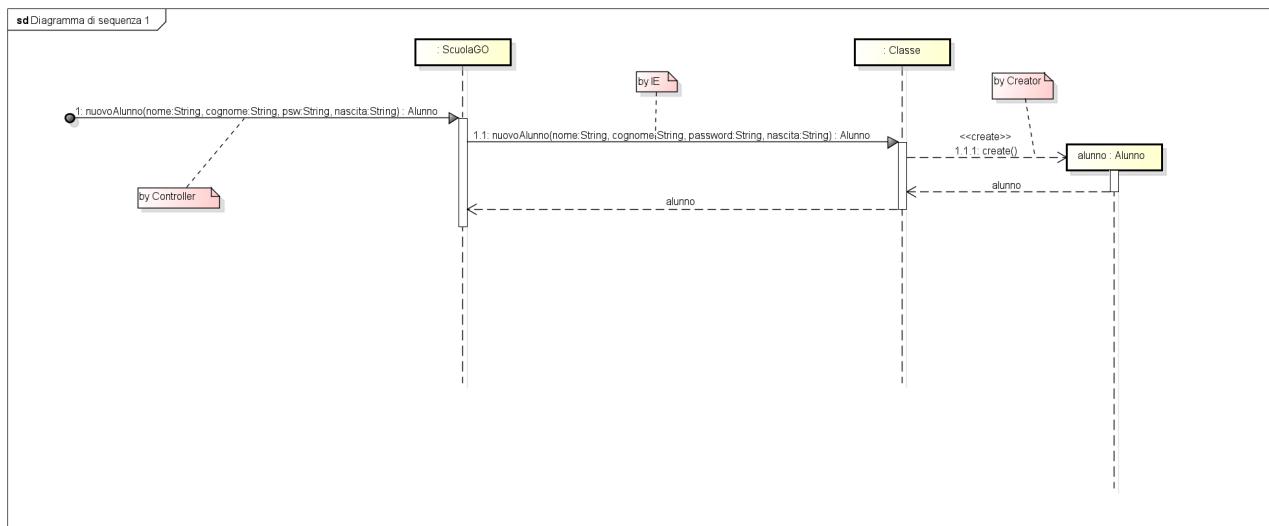
UC1: Registrazione amministratore

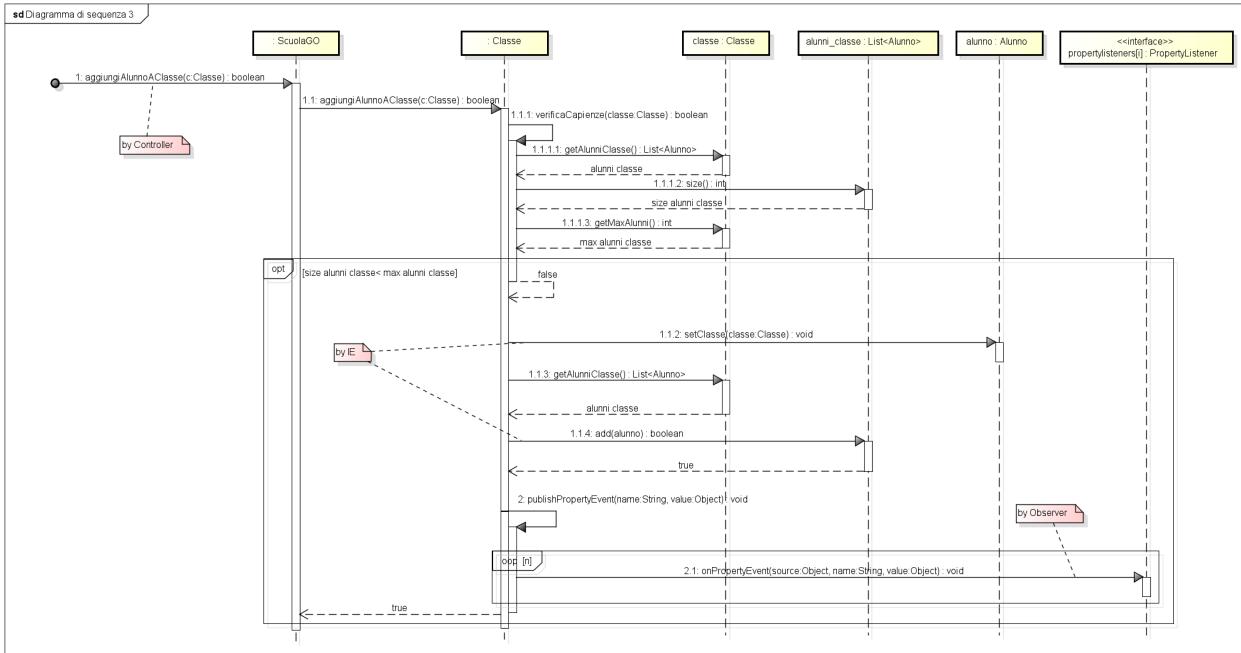


UC3: Gestione classi



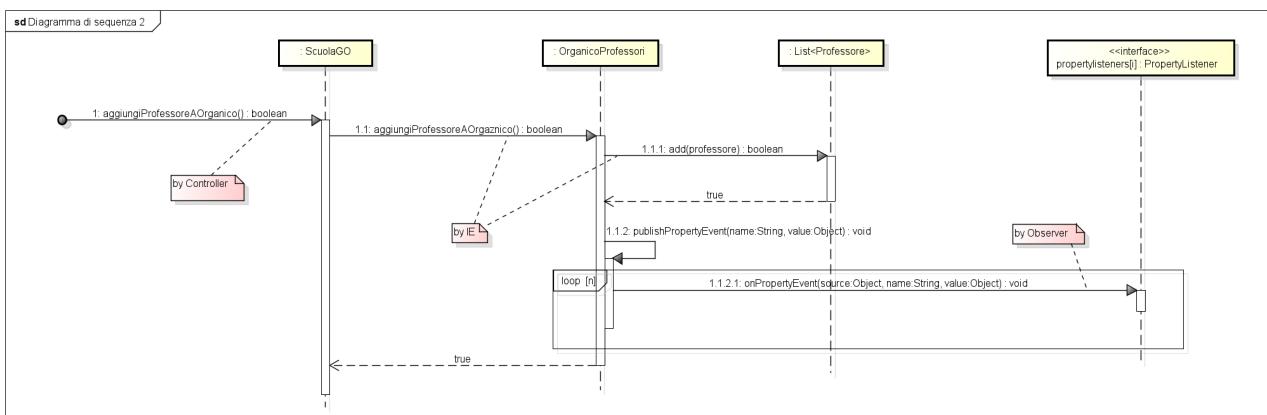
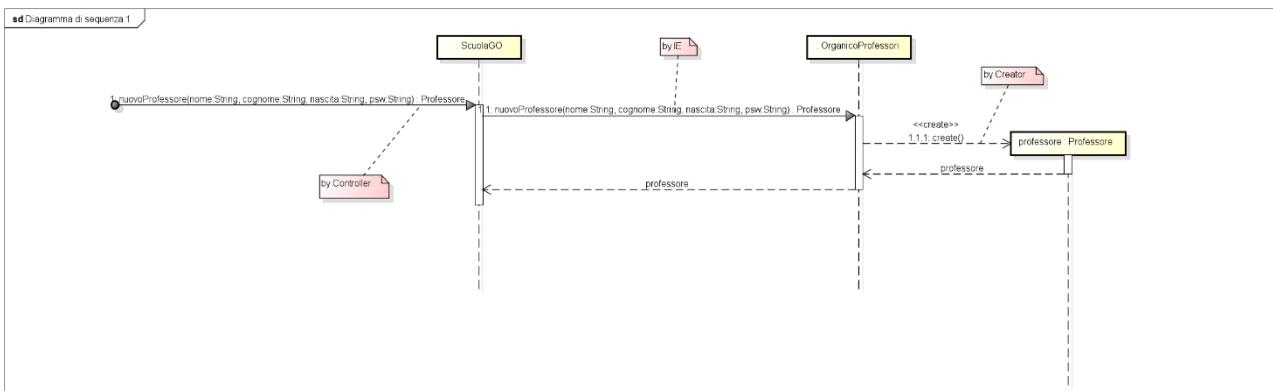
UC4: Inserimento nuovo alunno



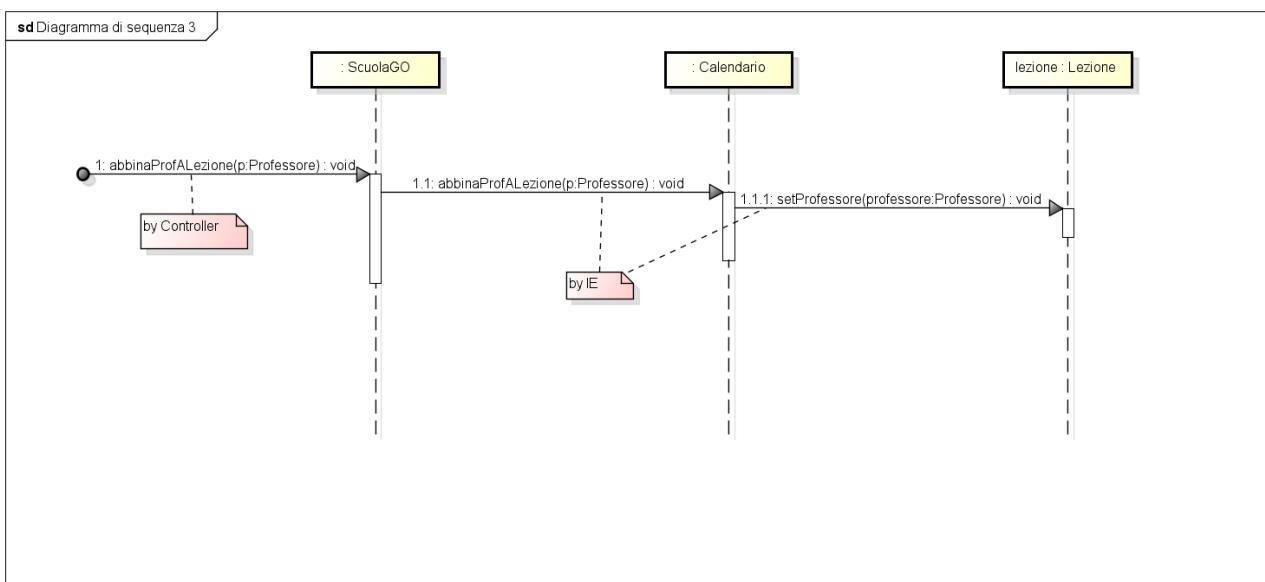
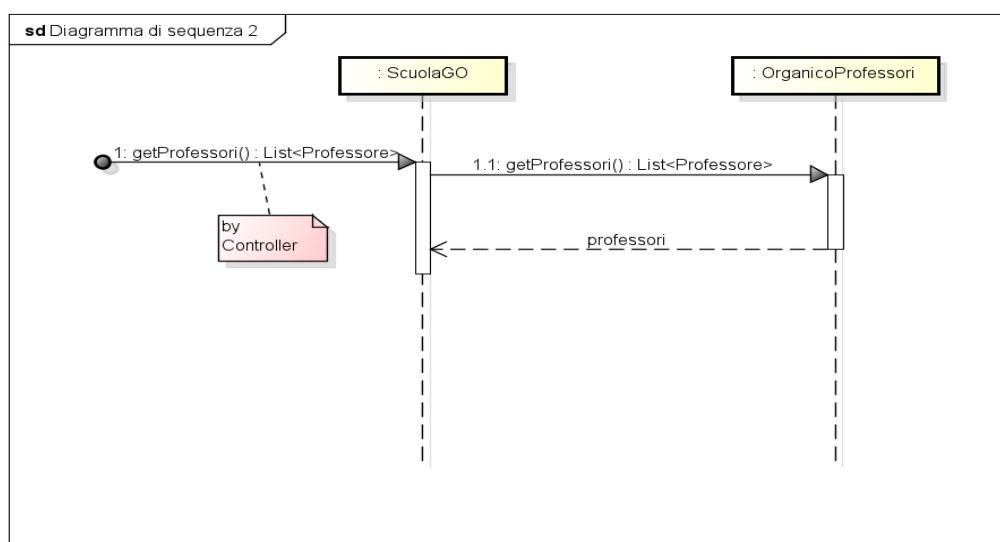
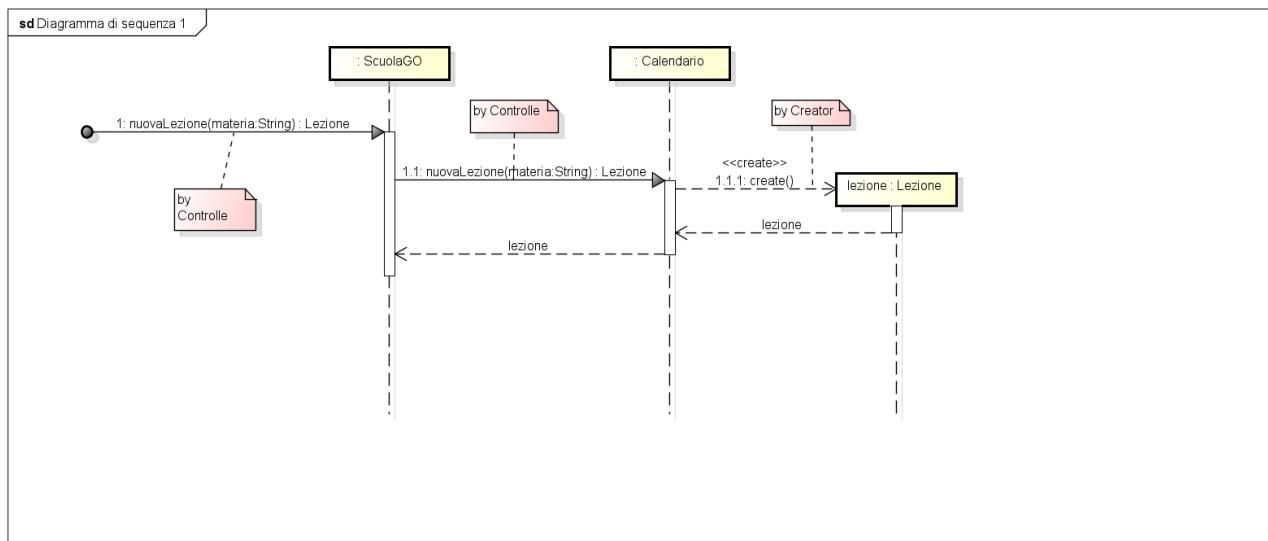


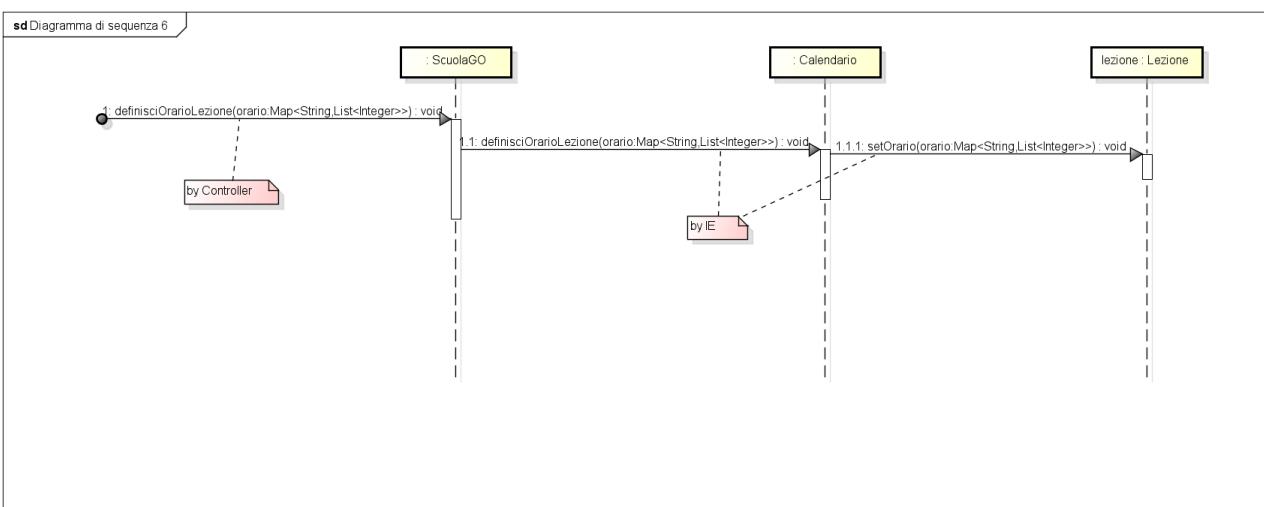
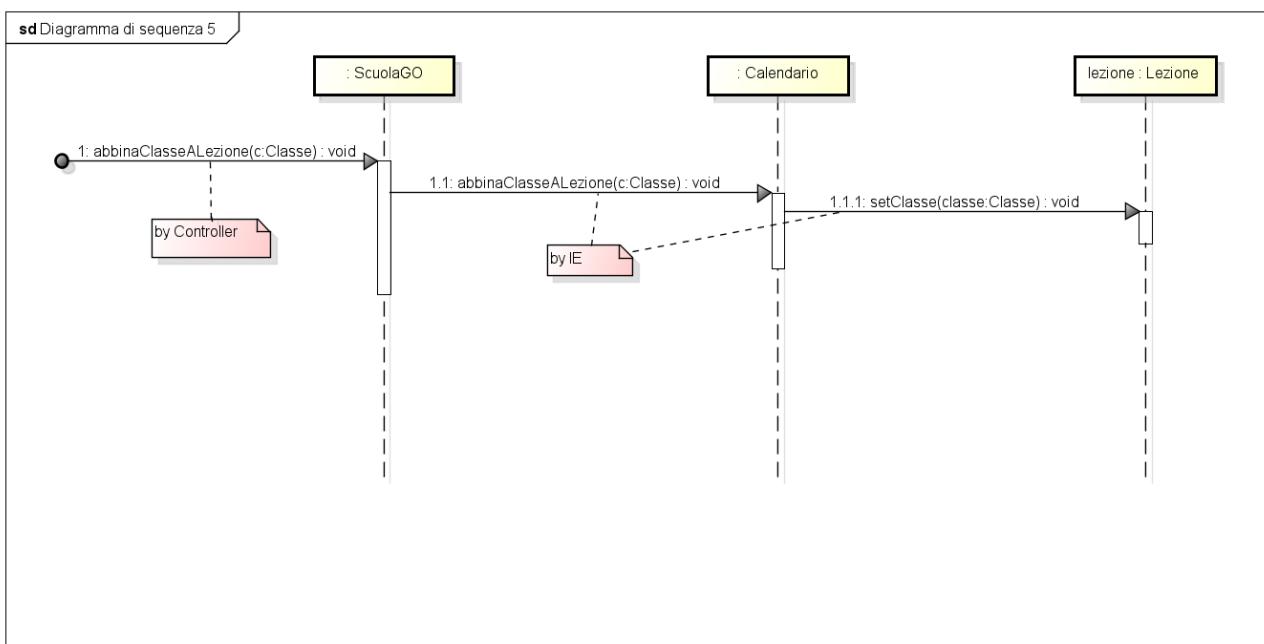
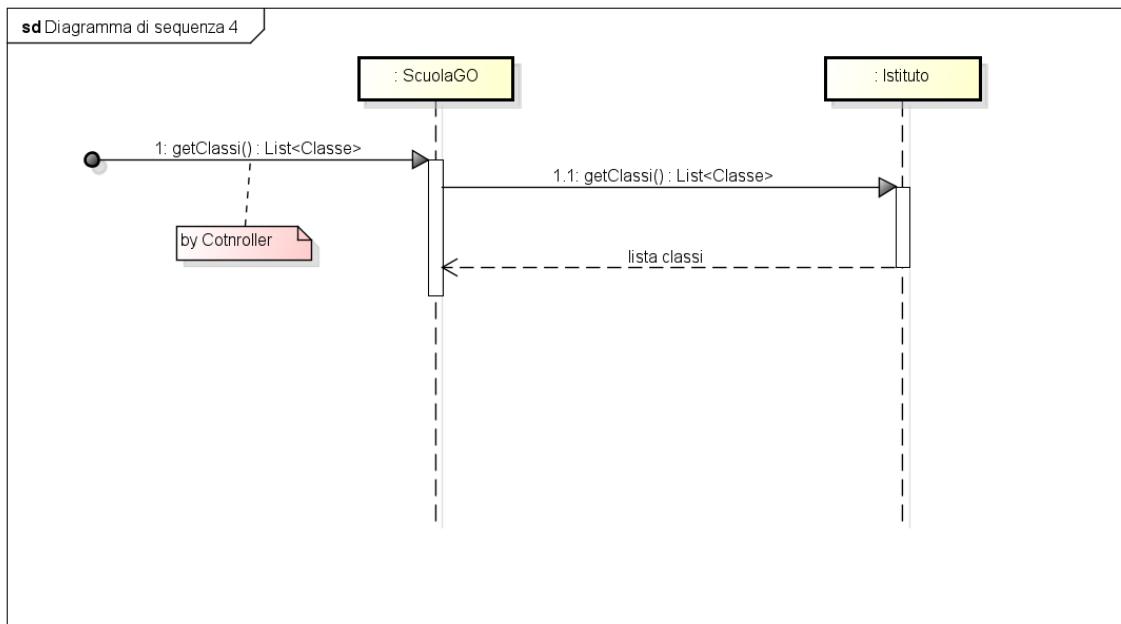
❖ Iterazione 2

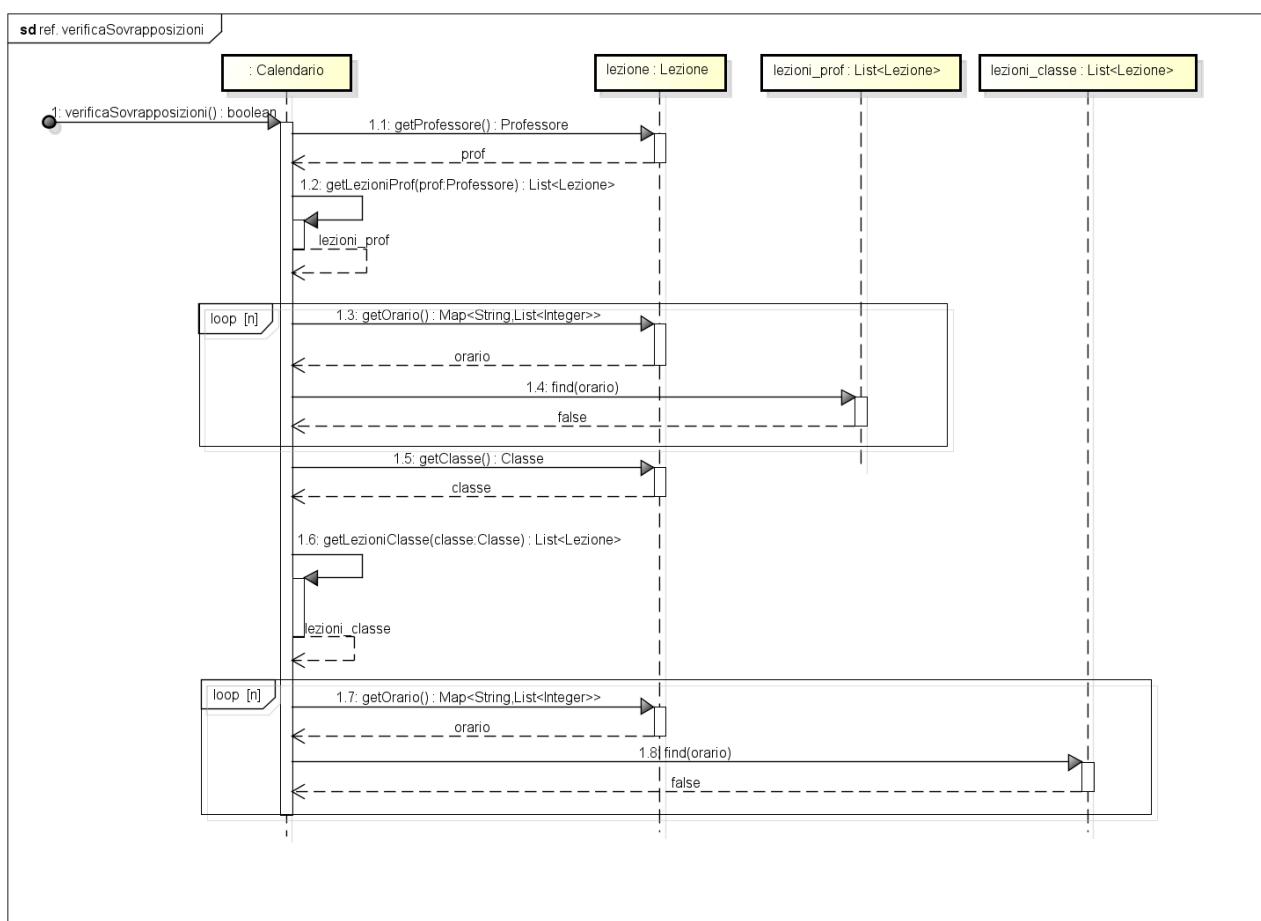
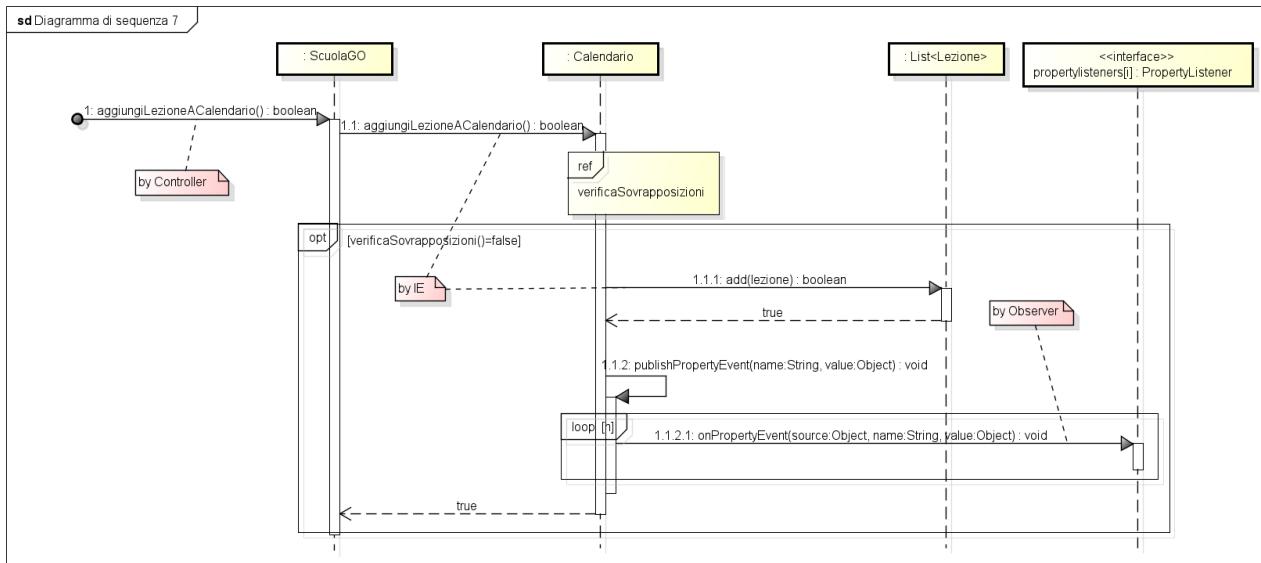
UC2: Gestione professori



UC5: Creazione lezione

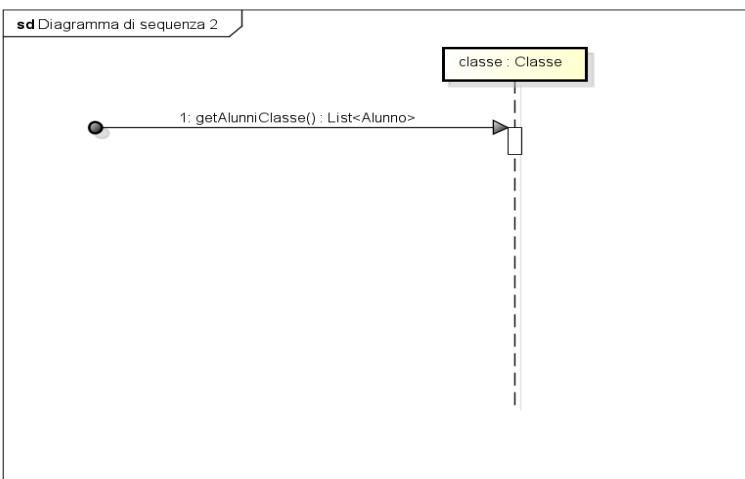
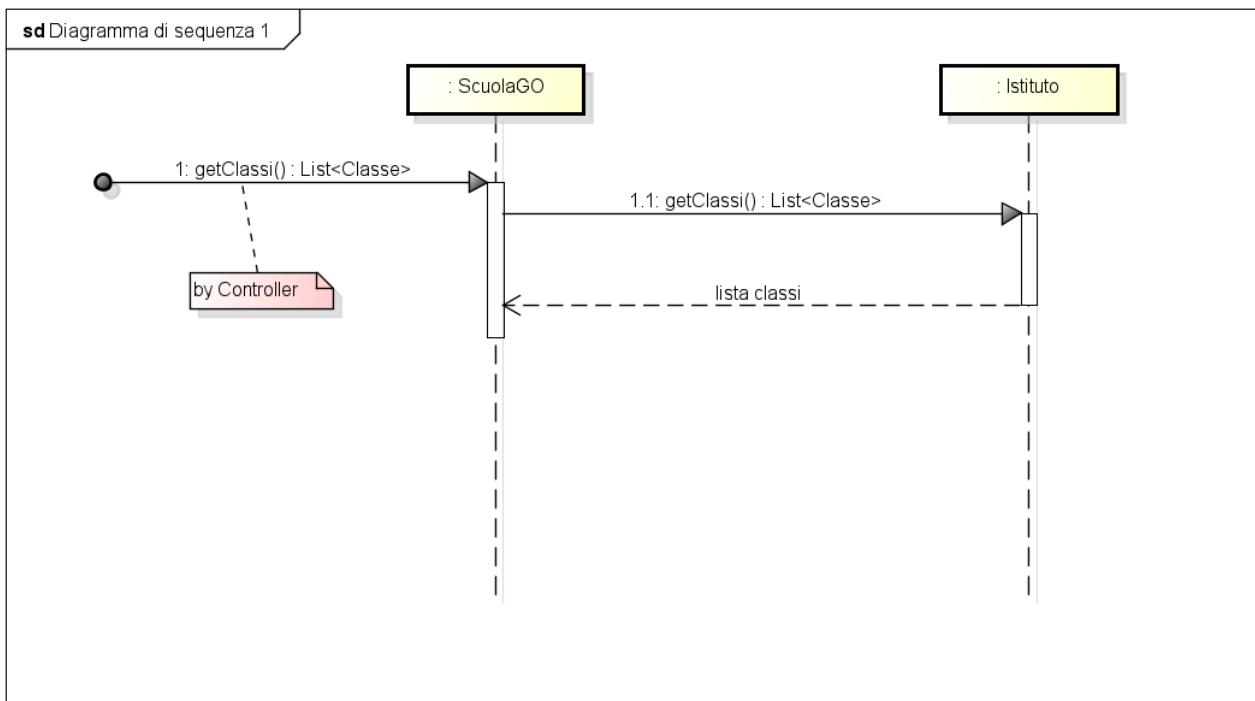


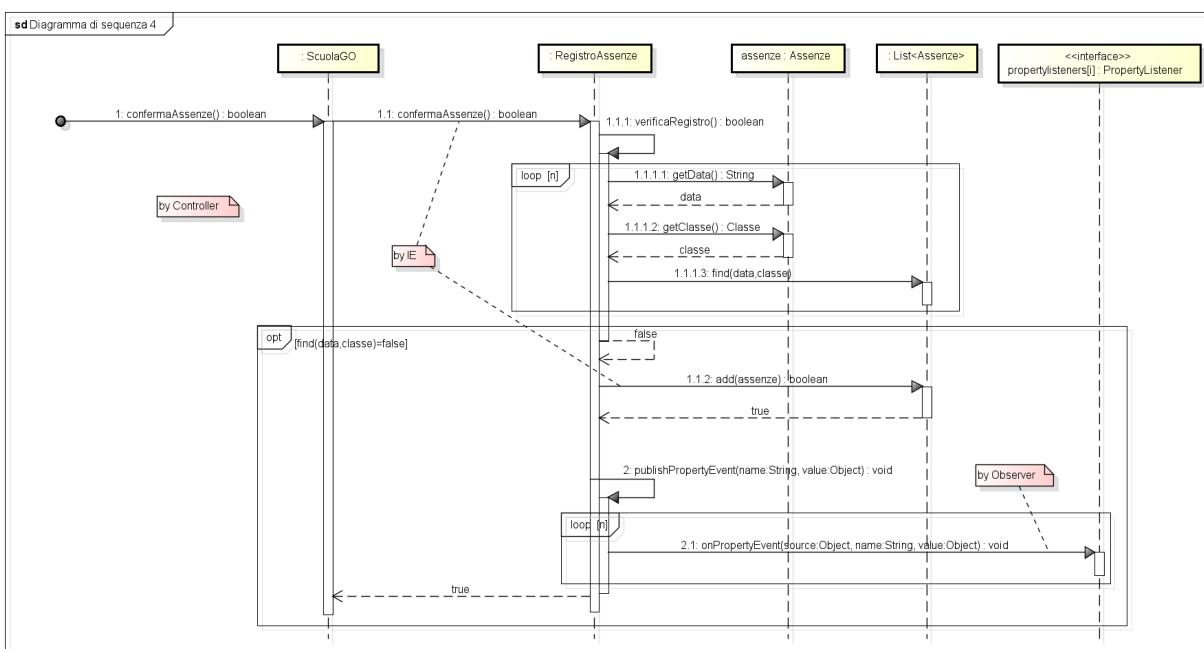
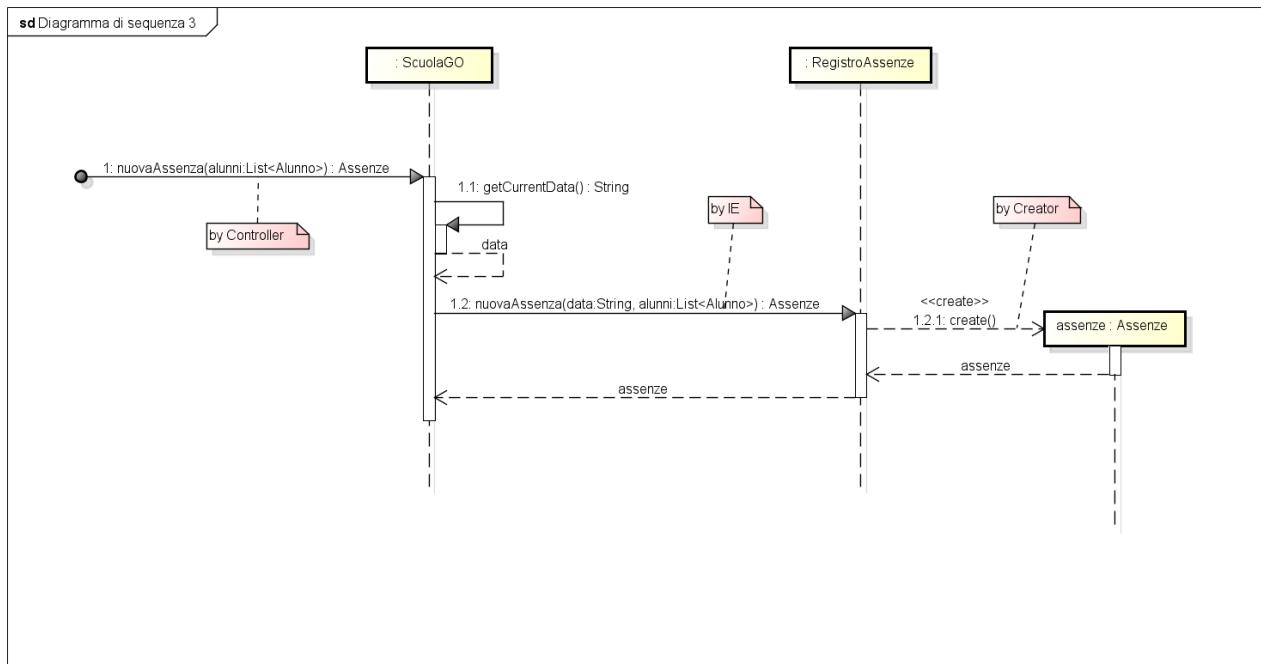




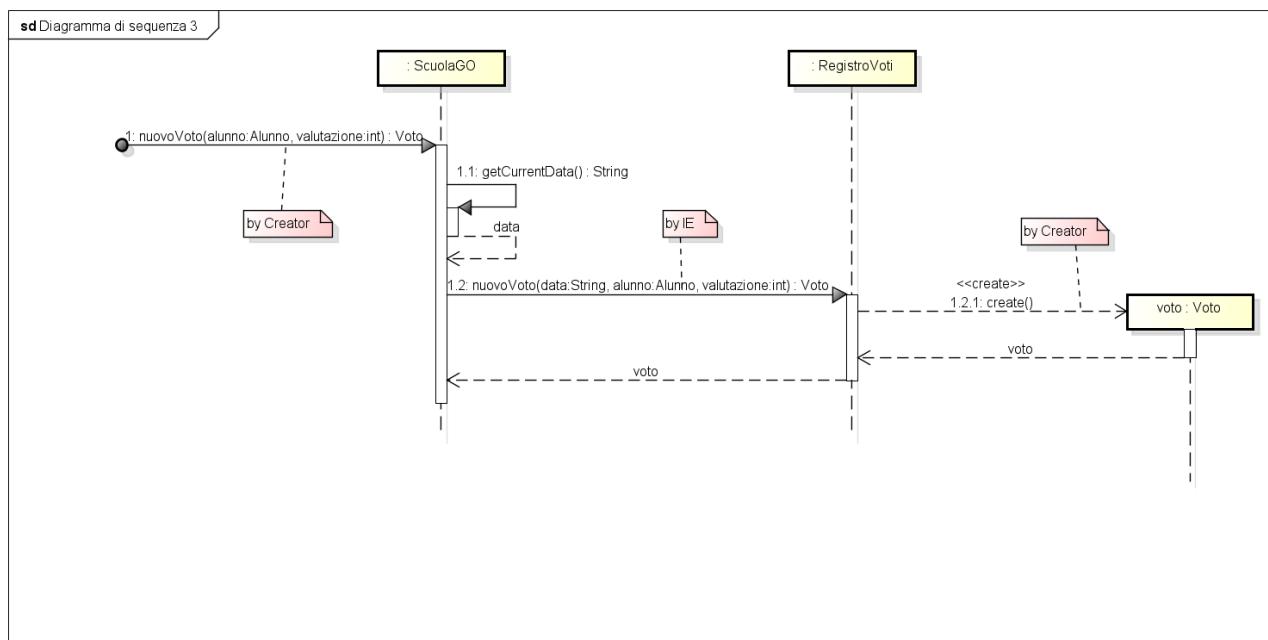
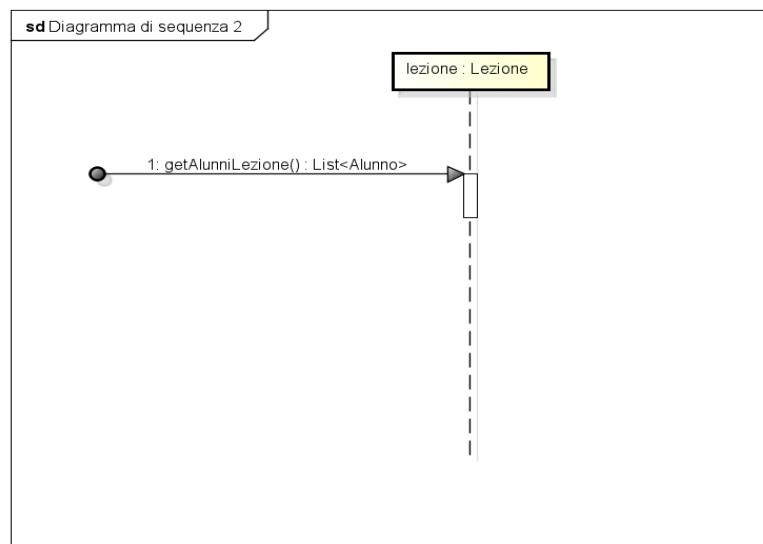
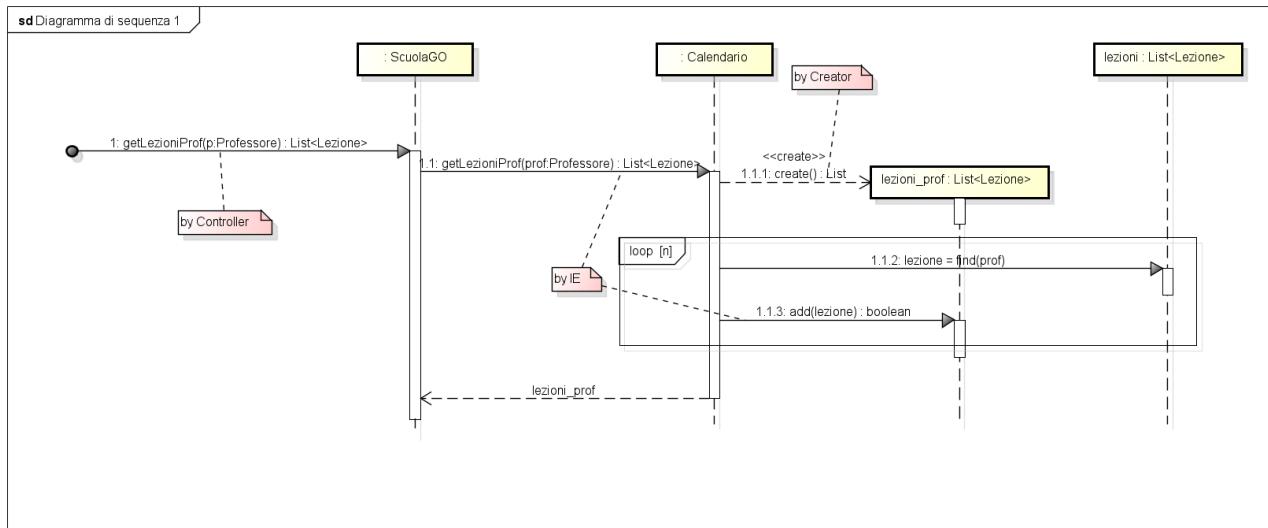
❖ Iterazione 3

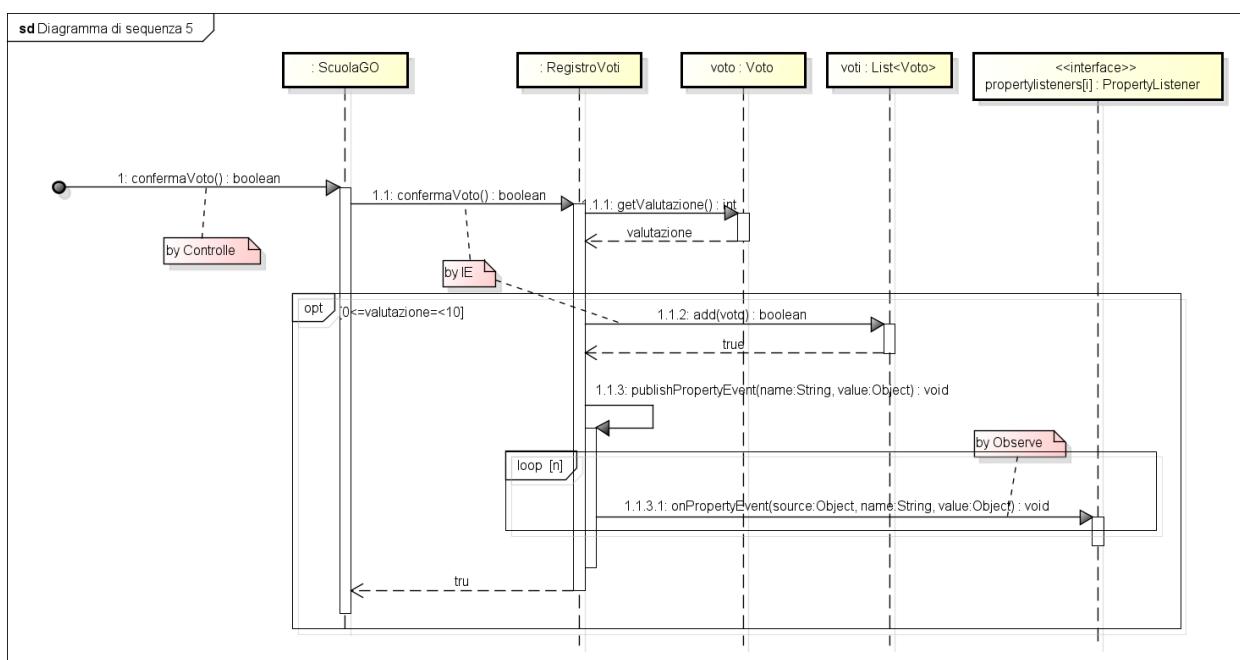
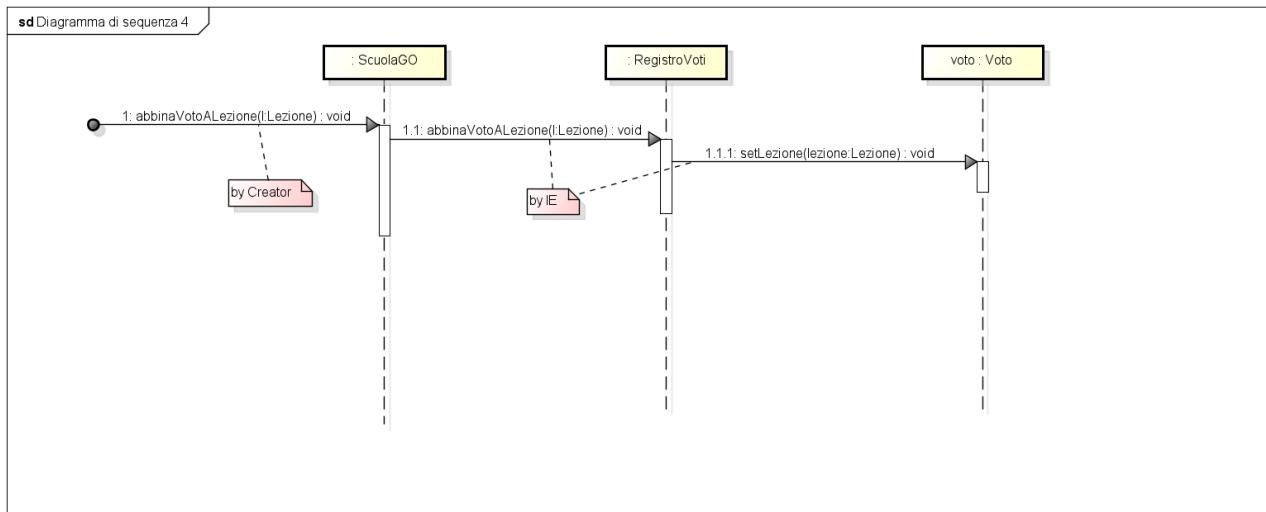
UC7: Gestione registro assenze





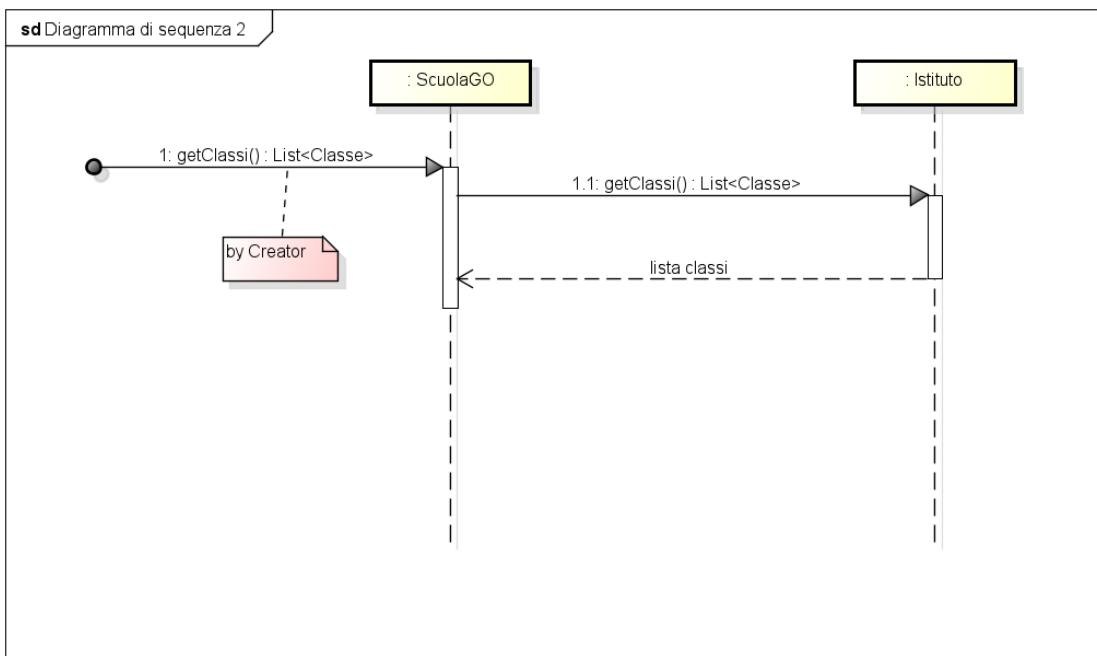
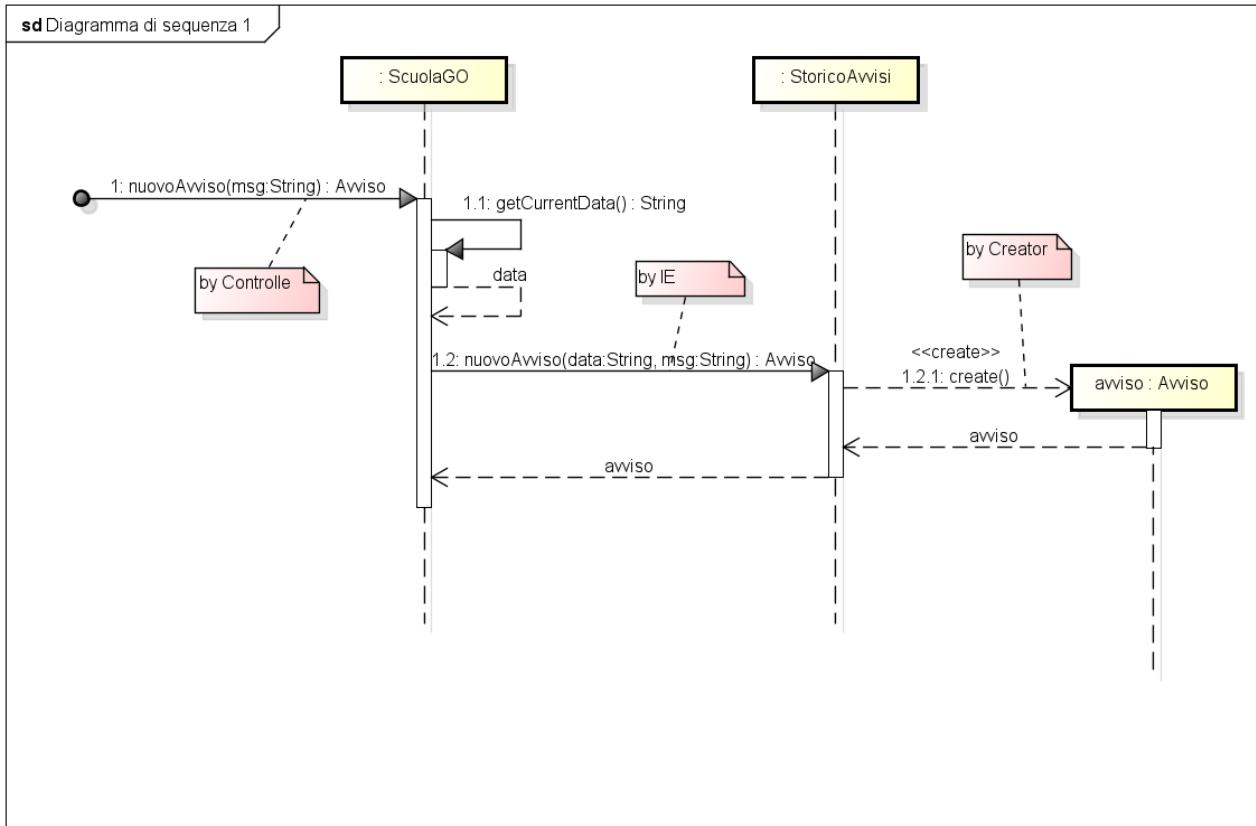
UC8: Gestione registro voti

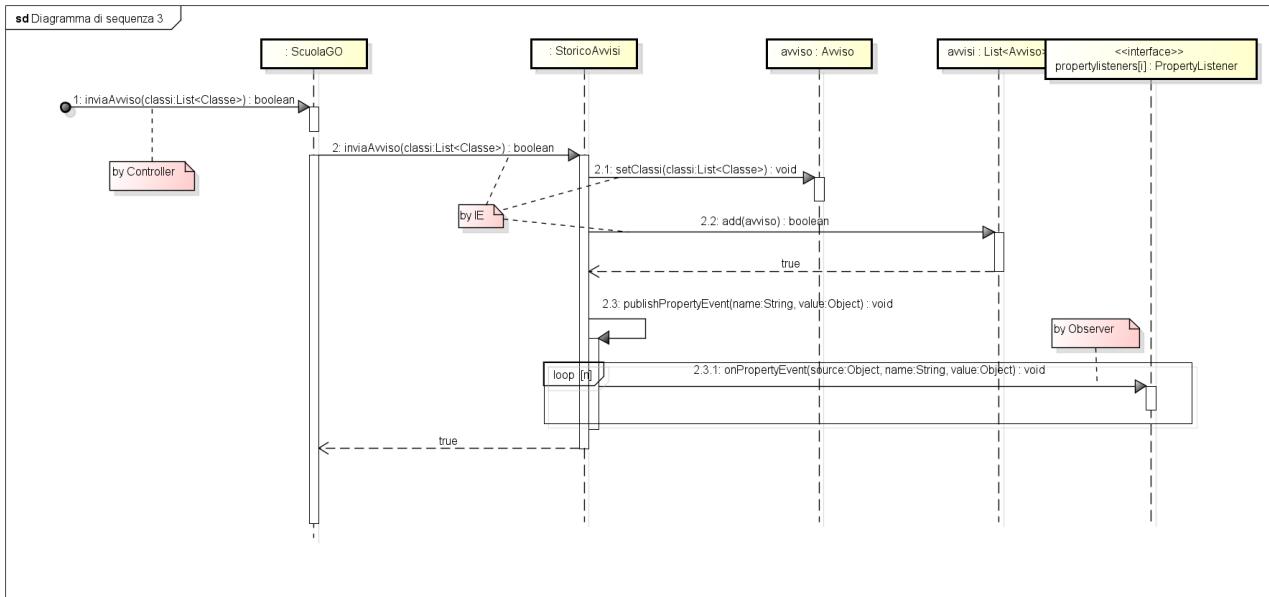




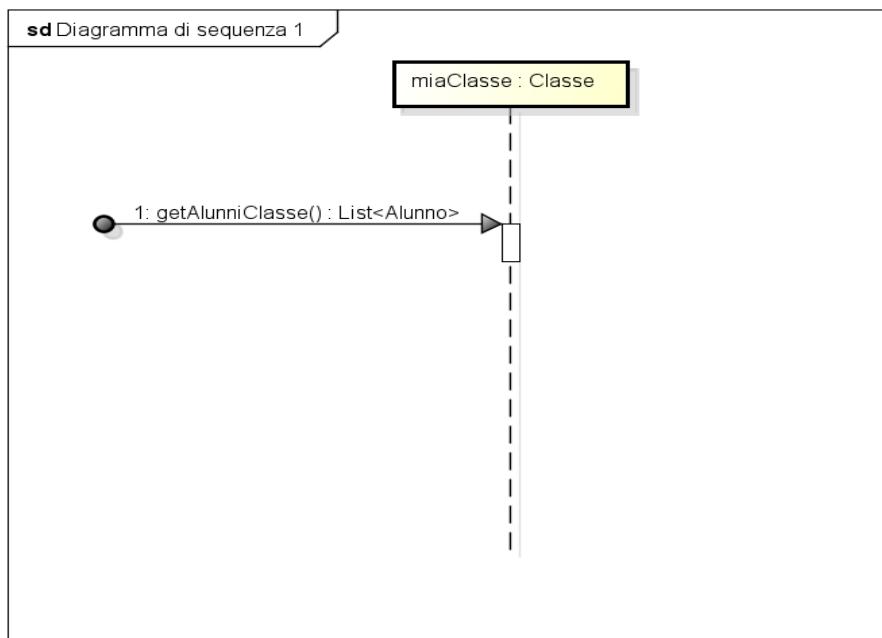
❖ Iterazione 4

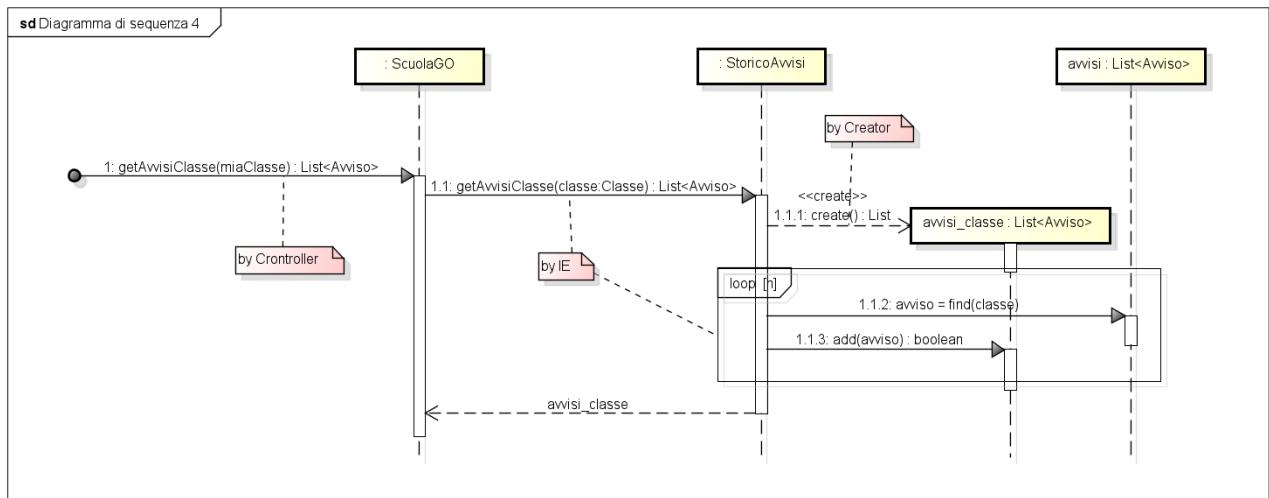
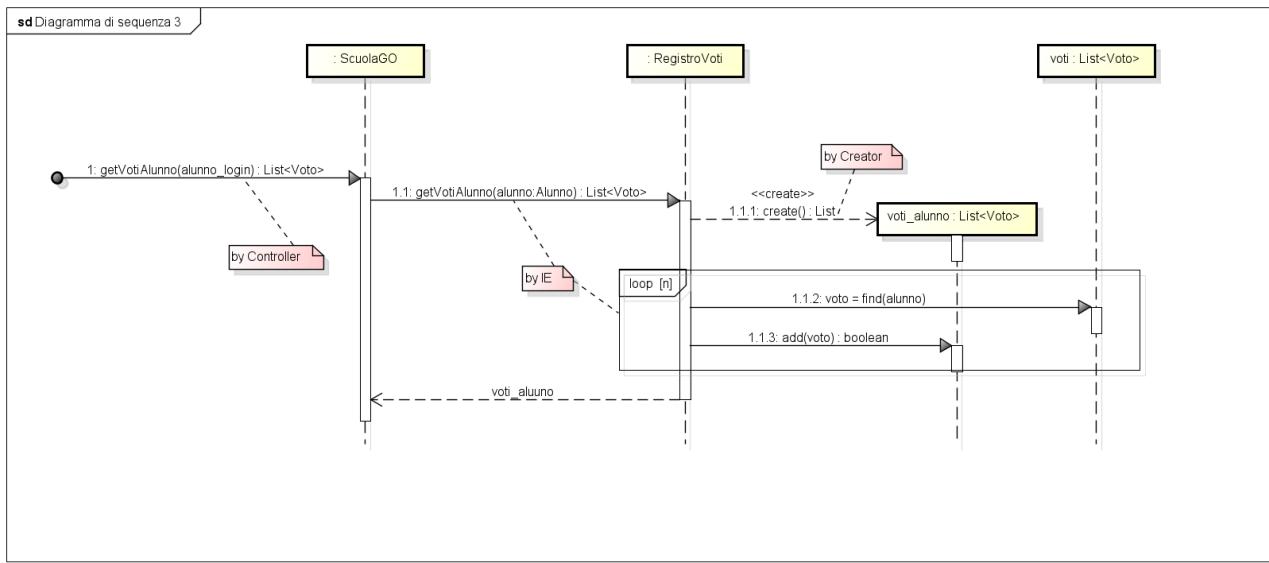
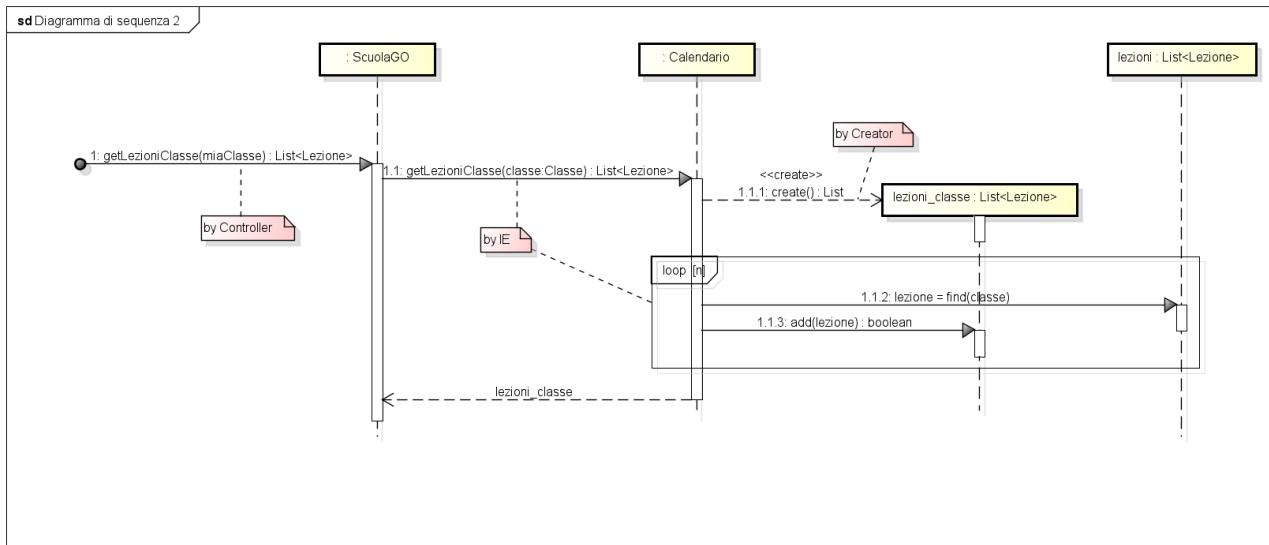
UC6: Invio avviso

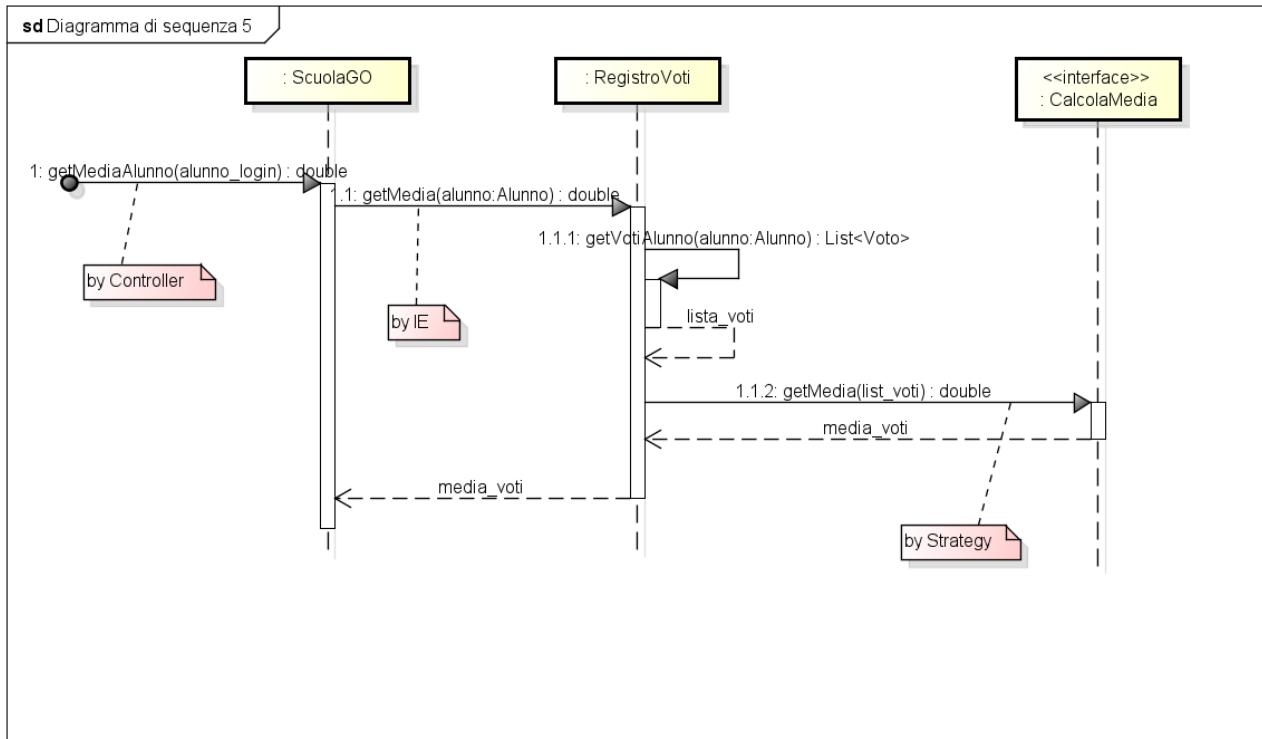




UC9: Visualizzazione info classe

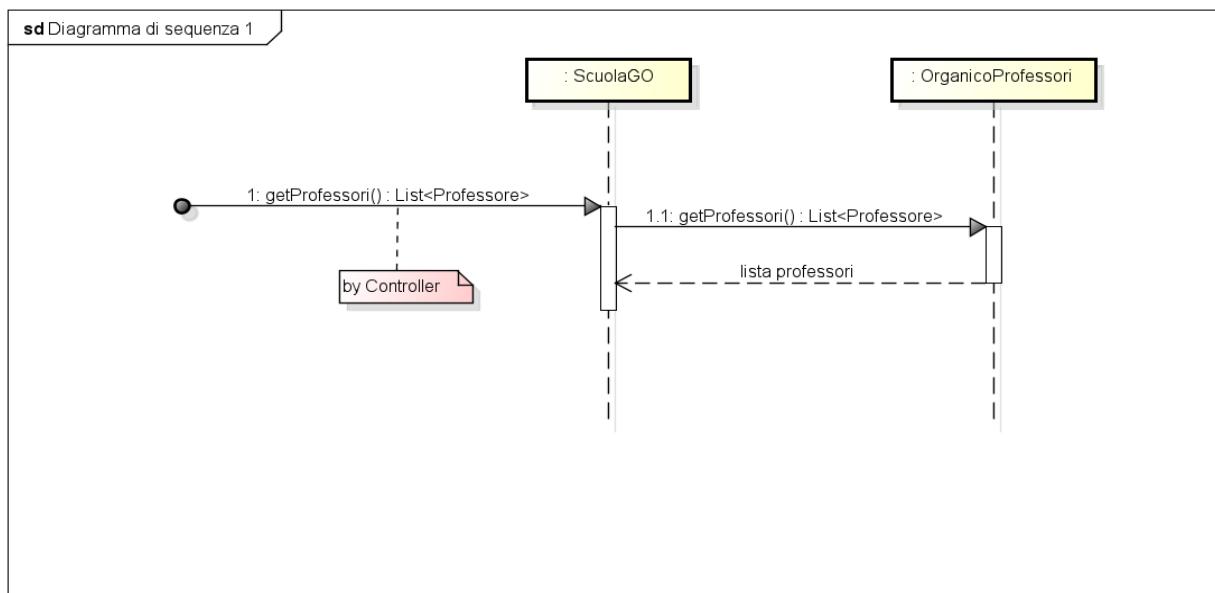


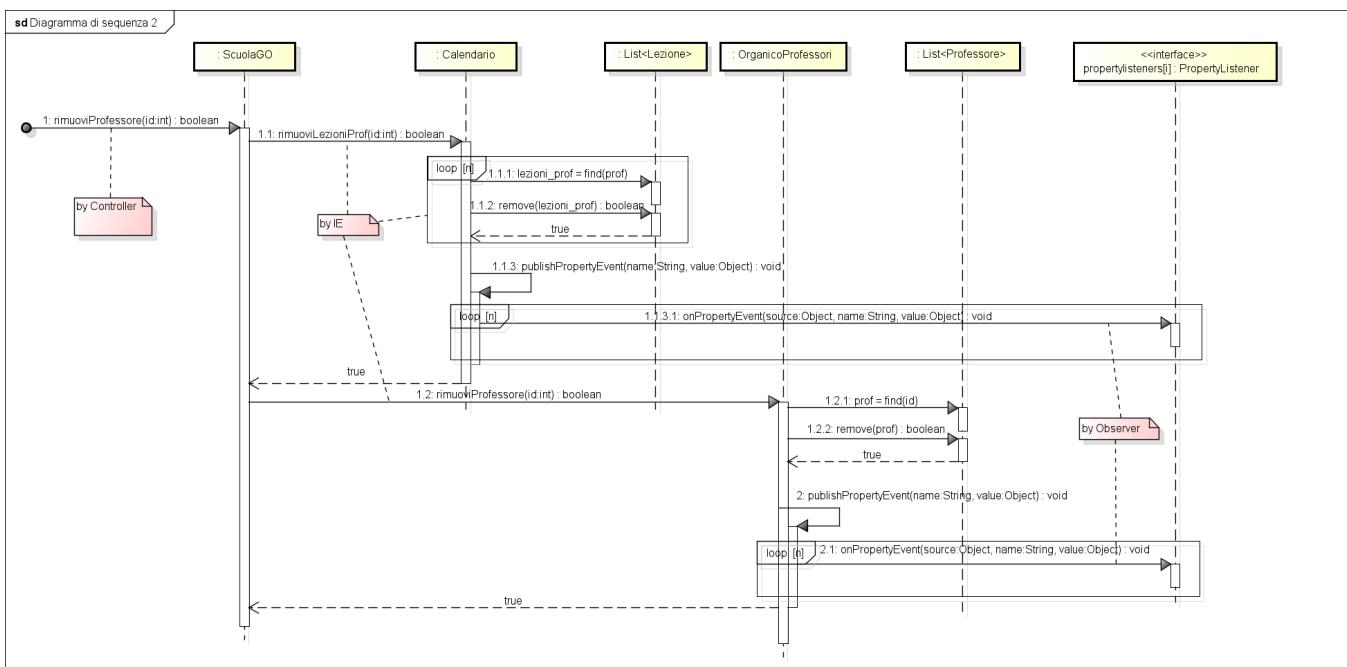




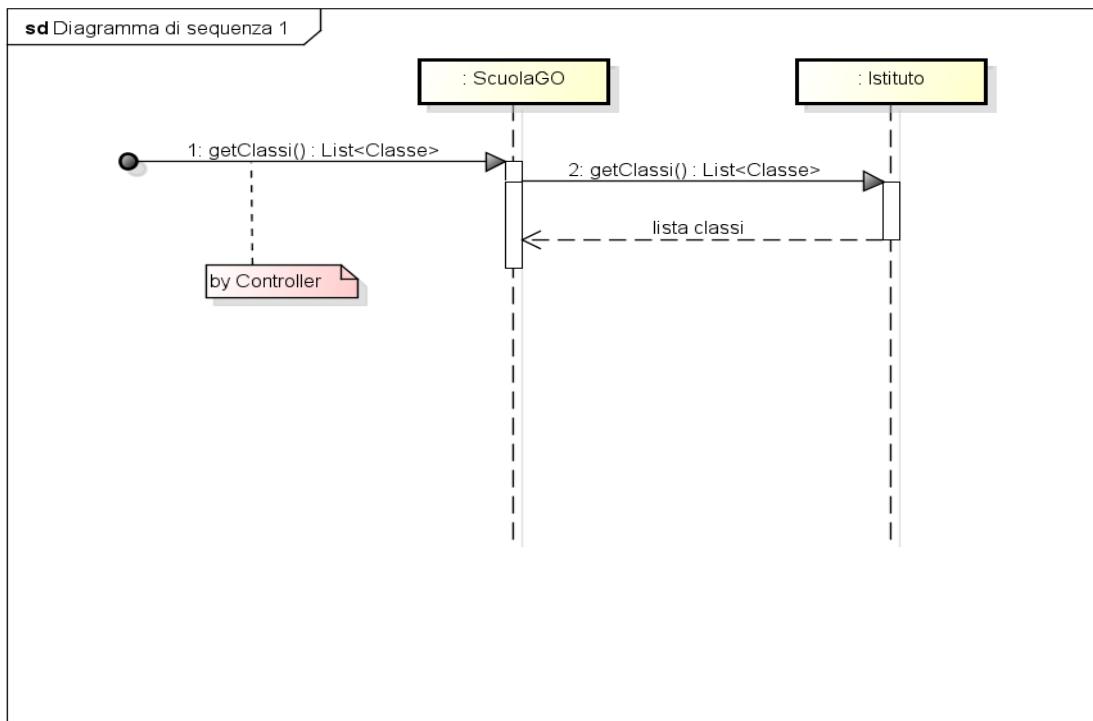
❖ Iterazione 5

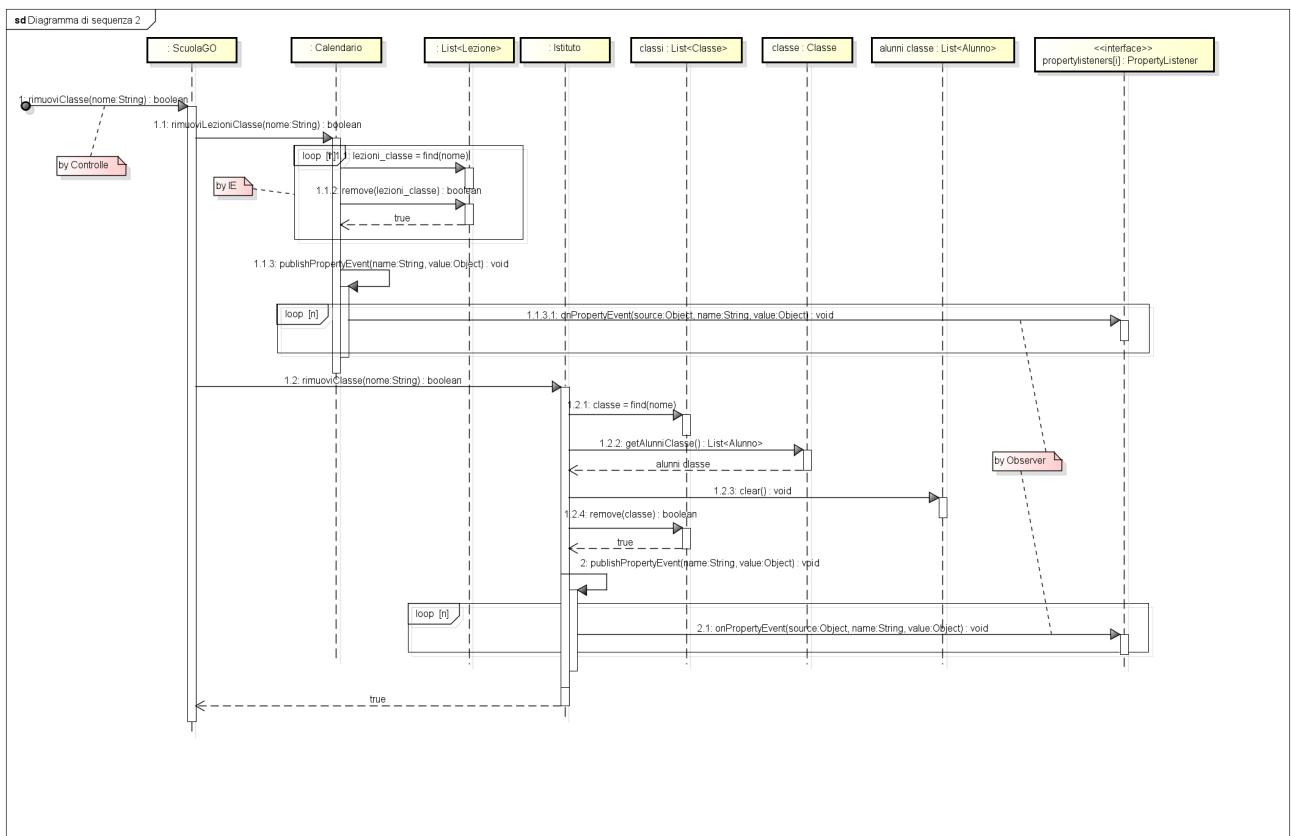
UC2 (alternativo): Gestione professori





UC3 (alternativo): Gestione classi





5 Testing

Il testing è una fase che ha permesso di rilevare errori e malfunzionamenti del sistema e, pertanto, di migliorare l'efficienza del software. Per affrontare questo processo sono stati scelti i test unitari, test che mirano a verificare la correttezza di unità di codice confrontando il risultato atteso con il risultato ottenuto. Il framework utilizzato per questo scopo è JUnit5, esso consente di eseguire test unitari sul codice Java.

Per ScuolaGO sono state individuate 8 classi di test, che coprono tutti i casi d'uso che modificano lo stato del programma (UC1-UC8). È stato possibile garantire una buona copertura, testando gran parte del codice delle funzioni presenti nelle varie classi di dominio. Di seguito vengono esaminate in dettaglio le classi di test:

1. testAdmin:

- `testRegistraAdmin()`: verifica l'avvenuta registrazione di un amministratore
- `testRegistraAdmin2()`: verifica la non avvenuta registrazione per un amministratore con un email già utilizzata
- `testLoginAdmin()`: verifica il login di un amministratore

2. testProfessore:

- *tesAggiungiProfessore()*: verifica l'aggiunta di un nuovo professore all'organico
- *tesLoginProfessore()*: verifica il login di un professore in organico
- *testRimuoviProfessore()*: verifica la rimozione di un professore e delle sue lezioni. Il calendario non dovrà contenere lezioni del prof rimosso

3. testClasse:

- *testAggiungiClasse()*: verifica l'aggiunta di una nuova classe all'istituto
- *testAggiungiClasse2()*: verifica la non avvenuta aggiunta di una classe con nome già utilizzato
- *testAggiungiClasse3()*: verifica l'impossibilità di aggiungere una classe con capienza massima maggiore di 30
- *testRimuoviClass()*: verifica la rimozione della classe, dei suoi alunni e delle sue lezioni. Gli alunni della classe devono essere rimossi e il calendario non dovrà contenere lezioni previste nella classe rimossa.

4. testAlunno:

- *testAggiungiAlunno()*: verifica l'aggiunta di un nuovo alunno in una classe dell'istituto
- *testAggiungiAlunno2()*: verifica la non avvenuta aggiunta di un alunno in una classe che ha già raggiunto la capienza massima
- *testLoginAlunno()*: verifica il login dell'alunno

5. testLezione:

- *tesCreaLezione()*: verifica la creazione di una nuova lezione
- *testCreaLezione2()*: verifica l'impossibilità di creare una lezione i quali orari si sovrappongono con altre lezioni del professore scelto
- *testCreaLezione2()*: verifica l'impossibilità di creare una lezione i quali orari si sovrappongono con altre lezioni della classe scelta

6. testAvviso:

- *testInviaAvviso()*: verifica l'invio di un nuovo avviso per una lista di classi

7. testAssenze:

- *testRegistraAssenze ()*: verifica la registrazione di nuove assenze nel registro assenze
- *testRegistraAssenze2()*: verifica l'impossibilità di registrare assenze più di una volta al giorno per classe

8. testVoto:

- *testRegistraVoto()*: verifica la registrazione di un voto nel registro voti
- *testRegistraVoto2()*: verifica l'impossibilità di registrare voti che non siano compresi tra 0 e 10

6 Pattern

Di seguito sono elencati alcuni dei pattern più significativi utilizzati per la progettazione del software.

- **Singleton:** utilizzato per creare una solo istanza del sistema ScuolaGO.
- **Controller:** utilizzato per indicare che il sistema ScuolaGo funge da controller, pertanto, su di esso verranno convogliati i comandi provenienti dall'interfaccia grafica. Sarà la classe ScuolaGo a gestire le funzionalità delegando compiti alle classi di dominio.
- **Creator:** pattern utilizzato più volte per specificare la responsabilità di creare un'istanza di classe.
- **Information Expert:** pattern necessario in tutti i casi d'uso per indicare chi ha le informazioni necessarie per effettuare una determinata azione.
- **Observer:** pattern utilizzato per aggiornare le tabelle e le liste dell'interfaccia grafica ad ogni modifica. Il pattern si serve della classe interfaccia "PropertyListener", essa viene implementata dalle finestre della GUI, esse si registrano per osservare gli oggetti di interesse.
- **Strategy:** utilizzato per definire le diverse strategie adottate dall'istituto per il calcolo della media dei voti di ogni alunno. Le classi "Media1" e "Media2" implementano l'interfaccia "CalcolaMedia" e rappresentano i 2 diversi metodi di calcolo finora proposti dall'istituto. "RegistroVoti" funge da contesto. Questo è un approccio flessibile, utile poiché l'istituto ha intenzione di aggiungere nuove strategie e di variarne l'utilizzo nel prossimo futuro.