

Corso di Laurea Magistrale in Ingegneria Informatica
A.A. 2023/2024
Ingegneria del Software



Di Bartolo Giorgio 1000001998
Garozzo Silvia 1000069223
Santanocito Matteo 1000001955

Indice

1. Ideazione e analisi dei requisiti	5
1.1 Introduzione.....	5
1.2 Requisiti.....	5
1.3 Obiettivo dei casi d'uso.....	6
1.4 Modello dei casi d'uso.....	7
UC1: Inserisci nuovo dipendente.....	7
UC2: Gestisci ruolo dipendente.....	8
UC3: Inserisci ferie o permessi.....	9
UC4: Gestisci congedi	10
UC5: Gestisci date assegnate	11
UC6: Visualizza congedi.....	12
UC7: Visualizza congedi personali.....	13
UC8: Assegna buoni pasto	14
UC9: Gestisci buono pasto	15
UC10: Visualizza buoni pasto personali.....	16
1.5 Regole di dominio.....	17
1.6 Glossario	17
2. Analisi Orientata agli Oggetti.....	18
2.1 Introduzione	18
2.2 Modello di Dominio.....	19
2.3 SSD e Contratti	20
• Iterazione 1.....	20
UC1: Inserisci nuovo dipendente	20
Contratto CO1: inserisciNuovoDipendente.....	21
Contratto CO2: confermaInserimento	21
UC2: Gestisci ruolo dipendente.....	22
Contratto CO1: confermaRuolo	22

• Iterazione 2	23
UC3: Inserisci ferie o permessi	23
Contratto CO1: inserisciFerie	23
Contratto CO2: confermaFerie	24
Contratto CO3: inserisciPermesso	24
Contratto CO4: confermaPermesso	25
UC4: Gestisci congedi	25
Contratto CO1: confermaCongedo	26
UC5: Gestisci date assegnate	26
Contratto CO1: richiediProlungamentoFerie	27
Contratto CO2: confermaProlungamentoFerie	27
Contratto CO3: richiediProlungamentoPermesso	28
Contratto CO4: confermaProlungamentoPermesso	28
UC6: Visualizza congedi	29
UC7: Visualizza congedi personali	29
• Iterazione 3	30
UC8: Assegna buoni pasto	30
Contratto CO1: creaBuonoPasto	30
Contratto CO2: confermaBuonoPasto	31
UC9: Gestisci buono pasto	31
Contratto CO1: confermaAttivazione	32
UC10: Visualizza buoni pasto personali	32
3. Progettazione	33
3.1 Diagrammi delle Classi	33
3.2 Diagrammi di Sequenza	34
• Caso d'Uso di Avviamento	34
• Iterazione 1	34
UC1: Inserisci nuovo dipendente	34

SD inserisciNuovoDipendente.....	34
SD confermaInserimento.....	35
UC2: Gestisci ruolo dipendente.....	35
SD mostraDipendenti	35
SD confermaRuolo.....	36
• Iterazione 2.....	36
UC3: Inserisci ferie o permessi.....	36
SD inserisciFerie	36
SD confermaFerie	37
SD inserisciPermesso	37
SD confermaPermesso.....	38
UC4: Gestisci congedi.....	38
SD mostraCongedi.....	38
SD sovrapposizioneCongedi	39
SD confermaCongedo	39
UC5: Gestisci date assegnate	40
SD mostraCongediApprovati.....	40
SD richiediProlungamentoFerie.....	40
SD confermaProlungamentoFerie	41
SD richiediProlungamentoPermesso	41
SD confermaProlungamentoPermesso	42
UC6: Visualizza congedi	42
SD visualizzaCongediComplessivi.....	42
UC7: Visualizza congedi personali	43
SD visualizzaCongedi.....	43
• Iterazione 3	43
UC8: Assegna buoni pasto.....	43
SD visualizzaDipendenti	43

SD creaBuonoPasto	44
SD confermaBuonoPasto.....	44
UC9: Gestisci buono pasto.....	45
SD visualizzaBuoniPastoValidi	45
SD confermaAttivazione.....	45
UC10: Visualizza buoni pasto personali.....	46
SD visualizzaBuoniPasto	46
4. Testing.....	47
4.1 Introduzione.....	47
4.2 Individuazione dei casi di test e testing unitario.....	47

1. Ideazione e analisi dei requisiti

1.1 Introduzione

BenefitFlow è un software aziendale progettato per semplificare e sostituire i tradizionali processi manuali nella gestione dei benefit del personale. La mancanza di un sistema automatizzato attualmente crea disagi sia per i dirigenti che desiderano gestire tali benefici, sia per i dipendenti che intendono monitorarli. BenefitFlow si presenta come una soluzione rivolta alle aziende che intendono automatizzare tale processo, migliorandone significativamente l'efficienza complessiva. L'obiettivo è offrire un software che faciliti la gestione e il monitoraggio di tali benefici sia da parte dei dirigenti che dei dipendenti, consentendo in particolare a questi ultimi di avere una visione chiara di ferie, permessi e buoni pasto.

1.2 Requisiti

Gestione dei dipendenti: implementazione di un sistema per l'inserimento di uno nuovo dipendente, l'assegnazione dei buoni pasto e la gestione di ferie e permessi.

Gestione dei benefit: implementazione di un sistema flessibile per generare richieste di ferie o permessi, gestirne un'eventuale prolungamento, attivare buoni pasto e visualizzare gli storici per ognuno di tali benefit.

1.3 Obiettivo dei casi d'uso

Analizzando i requisiti sopra riportati, sono stati individuati gli attori principali a cui è destinato il sistema e gli obiettivi che si intendono raggiungere.

Da queste informazioni sono stati ricavati i seguenti casi d'uso:

Attore	Obiettivo	Caso d'uso
Amministratore	Gestire la registrazione di uno specifico dipendente	UC1: Inserisci nuovo dipendente
Amministratore	Gestire l'assegnazione di uno specifico ruolo con annesso livello	UC2: Gestisci ruolo dipendente
Dipendente	Richiedere date specifiche relative a permessi e ferie.	UC3: Inserisci ferie o permessi
Amministratore	Gestire le date richieste dai dipendenti evitando sovrapposizioni	UC4: Gestisci congedi
Dipendente	Gestire le date approvate proponendo un prolungamento in caso di necessità	UC5: Gestisci le date assegnate
Amministratore	Visualizzare le date complessive per ferie e permessi di tutti i dipendenti	UC6: Visualizza congedi
Dipendente	Visualizzare la propria pianificazione di ferie e permessi	UC7: Visualizza congedi personali
Amministratore	Gestire l'assegnazione del buono pasto in base al livello del dipendente	UC8: Assegna buono pasto
Dipendente	Gestire l'utilizzo del buono pasto attivandolo	UC9: Gestisci buono pasto
Dipendente	Visualizzare la propria lista di buoni pasto	UC10: Visualizza buoni pasto personali

1.4 Modello dei casi d'uso

UC1: Inserisci nuovo dipendente

Nome del caso d'uso	UC1: Inserisci nuovo dipendente
Portata	Applicazione BenefitFlow
Livello	Obiettivo azienda
Attore primario	Amministratore
Parti interessate e interessi	- Amministratore: vuole gestire l'inserimento di un nuovo dipendente
Pre-condizioni	
Garanzia di successo	Le informazioni relative ai dipendenti sono inserite con successo nel Sistema
Scenario principale di successo	<ol style="list-style-type: none">1. L'Amministratore vuole inserire un nuovo dipendente nel Sistema aziendale.2. L'Amministratore sceglie l'attività "inserimento nuovo dipendente".3. L'Amministratore inserisce nome, cognome, anno di nascita e ruolo.4. L'Amministratore indica di aver finito. Il sistema registra le informazioni inserite.
Estensioni	* a. In un qualsiasi momento il Sistema fallisce e si arresta. <ol style="list-style-type: none">1. L'Amministratore riavvia il software e ripristina lo stato precedente del Sistema.2. Il Sistema ripristina lo stato.
Requisiti speciali	
Elenco varianti tecnologiche e dati	
Frequenza ripetizioni	Legata all'ingresso di un nuovo dipendente in azienda
Varie	

UC2: Gestisci ruolo dipendente

Nome del caso d'uso	UC2: Gestisci ruolo dipendente
Portata	Applicazione BenefitFlow
Livello	Obiettivo azienda
Attore primario	Amministratore
Parti interessate e interessi	- Amministratore: vuole gestire l'assegnazione o la modifica del ruolo di un dipendente
Pre-condizioni	
Garanzia di successo	Le informazioni relative al ruolo di un dipendente sono inserite o modificate con successo nel Sistema
Scenario principale di successo	<ol style="list-style-type: none"> 1. L'Amministratore vuole inserire o modificare il ruolo di un dipendente nel Sistema aziendale. 2. L'Amministratore sceglie l'attività "gestisci ruolo". 3. L'Amministratore richiede al Sistema la lista dei dipendenti per poterla visionare. 4. Il Sistema mostra la lista dei dipendenti. 5. L'Amministratore cerca nella lista fornita dal Sistema il dipendente, tramite la matricola, a cui vuole assegnare o modificare il ruolo. 6. L'Amministratore configura il ruolo del dipendente. 7. L'Amministratore indica di aver finito. Il Sistema conferma e registra il nuovo ruolo.
Estensioni	<p>*a. In un qualsiasi momento il Sistema fallisce e si arresta.</p> <ol style="list-style-type: none"> 1. L'Amministratore riavvia il software e ripristina lo stato precedente del Sistema. 2. Il Sistema ripristina lo stato. <p>5a. L'Amministratore prova a ottenere il ruolo di un dipendente che non esiste</p> <ol style="list-style-type: none"> 1. Il Sistema genera un messaggio di errore. 2. L'Amministratore ricerca nuovamente il dipendente ripetendo il passaggio 5
Requisiti speciali	
Elenco varianti tecnologiche e dati	
Frequenza ripetizioni	
Varie	

UC3: Inserisci ferie o permessi

Nome del caso d'uso	UC3: Inserisci ferie o permessi
Portata	Applicazione BenefitFlow
Livello	Obiettivo azienda
Attore primario	Dipendente
Parti interessate e interessi	- Dipendente: vuole gestire l'inserimento di ferie e permessi per effettuare una richiesta
Pre-condizioni	
Garanzia di successo	Le informazioni relative ai benefit sono inserite con successo nel Sistema
Scenario principale di successo	<ol style="list-style-type: none"> Il Dipendente vuole poter inserire nel Sistema le sue richieste relative a ferie o permessi. Il Dipendente sceglie l'attività "richiedi congedo". Il Dipendente sceglie il tipo di benefit da voler inserire. In base al tipo scelto, il Dipendente può inserire il benefit definendo motivazione, matricola, dataInizio, dataFine se si tratta di Ferie; oppure fornire motivazione, matricola, data, oraInizio, oraFine se si tratta di un Permesso. Il Dipendente indica di aver finito. Il Sistema registra le informazioni.
Estensioni	<p>*a. In un qualsiasi momento il Sistema fallisce e si arresta.</p> <ol style="list-style-type: none"> Il Dipendente riavvia il software e ripristina lo stato precedente del Sistema. Il Sistema ripristina lo stato.
Requisiti speciali	
Elenco varianti tecnologiche e dati	
Frequenza ripetizioni	Ogni volta che il dipendente vuole richiedere dei benefit
Varie	

UC4: Gestisci congedi

Nome del caso d'uso	UC4: Gestisci congedi
Portata	Applicazione BenefitFlow
Livello	Obiettivo azienda
Attore primario	Amministratore
Parti interessate e interessi	<ul style="list-style-type: none"> - Amministratore: vuole garantire una corretta schedulazione dei benefit che ogni dipendente ha richiesto. - Dipendente: vuole conoscere l'esito della richiesta precedentemente effettuata.
Pre-condizioni	Il dipendente conosce già le date che ha richiesto
Garanzia di successo	Le date e le ore relative alle ferie e ai permessi sono gestite con successo. Il sistema registra le informazioni.
Scenario principale di successo	<ol style="list-style-type: none"> 1. L'Amministratore vuole gestire i benefit richiesti dai dipendenti. 2. L'Amministratore sceglie l'attività "gestisci congedi". 3. Il Sistema mostra la lista delle richieste per ogni Dipendente a cui è associata la matricola. 4. L'Amministratore seleziona la richiesta tramite il suo codice univoco e controlla eventuali sovrapposizioni con altre richieste. 5. Il Sistema mostra, per quel benefit, se sono già state richieste date/ora simili da altri Dipendenti. 6. L'Amministratore gestisce la richiesta approvandola o rifiutandola. 7. L'Amministratore indica di aver finito. Il Sistema registra l'informazione.
Estensioni	<p>*a. In un qualsiasi momento il Sistema fallisce e si arresta.</p> <ol style="list-style-type: none"> 1. L'Amministratore riavvia il software e ripristina lo stato precedente del Sistema. 2. Il Sistema ripristina lo stato. <p>4a. L'Amministratore prova ad inserire un codice univoco per il Benefit diverso da quello possibile o non esistente.</p> <ol style="list-style-type: none"> 1. Il Sistema genera un messaggio di errore. 2. L'Amministratore inserisce nuovamente il codice ripetendo il passaggio 4.
Requisiti speciali	
Elenco varianti tecnologiche e dati	
Frequenza ripetizioni	Legata alle richieste eseguite dal dipendente
Varie	

UC5: Gestisci date assegnate

Nome del caso d'uso	UC5: Gestisci date assegnate
Portata	Applicazione BenefitFlow
Livello	Obiettivo azienda
Attore primario	Dipendente
Parti interessate e interessi	- Dipendente: vuole gestire le proprie ferie e i propri permessi richiedendone un prolungamento.
Pre-condizioni	Il Dipendente deve poter conoscere quale sono le date approvate
Garanzia di successo	Le nuove date o ore relative alle ferie o ai permessi sono modificate con successo. Il sistema registra le informazioni.
Scenario principale di successo	<ol style="list-style-type: none"> Il Dipendente vuole modificare la date o le ore precedentemente approvate. Il Dipendente sceglie l'attività "gestisci congedo". Il Dipendente sceglie il tipo di benefit da voler prolungare. Il Dipendente richiede, dalla lista completa dei benefit, quelli approvati. Il Sistema mostra la lista precedentemente richiesta. Il Dipendente, in base al benefit di cui vuole richiedere il prolungamento, fornisce il codice e la data di fine se si tratta di Ferie oppure l'ora di fine se si tratta di un Permesso. Il Sistema modifica lo stato e registra l'informazione.
Estensioni	<p>*a. In un qualsiasi momento il Sistema fallisce e si arresta.</p> <ol style="list-style-type: none"> Il Dipendente riavvia il software e ripristina lo stato precedente del Sistema. Il Sistema ripristina lo stato. <p>5a. Il Dipendente prova ad inserire un codice univoco per il Benefit diverso da quello possibile o non esistente.</p> <ol style="list-style-type: none"> Il Sistema genera un messaggio di errore. Il Dipendente inserisce nuovamente il codice ripetendo il passaggio 5.
Requisiti speciali	
Elenco varianti tecnologiche e dati	
Frequenza ripetizioni	Legata alle modifiche che il dipendente vuole effettuare.
Varie	

UC6: Visualizza congedi

Nome del caso d'uso	UC6: Visualizza congedi
Portata	Applicazione BenefitFlow
Livello	Obiettivo azienda
Attore primario	Amministratore
Parti interessate e interessi	- Amministratore vuole visualizzare i congedi di tutti i Dipendenti
Pre-condizioni	
Garanzia di successo	I congedi richiesti vengono mostrati all'Amministratore
Scenario principale di successo	<ol style="list-style-type: none"> 1. L'Amministratore vuole visualizzare i congedi di tutti i Dipendenti. 2. L'Amministratore sceglie l'attività "visualizza congedi". 3. L'Amministratore sceglie il tipo di congedo che vuole visualizzare. 4. Il Sistema mostra l'elenco di congedi richiesti in base al tipo fornito alla richiesta precedente.
Estensioni	<p>*a. In un qualsiasi momento il Sistema fallisce e si arresta.</p> <ol style="list-style-type: none"> 1. L'Amministratore riavvia il software e ripristina lo stato precedente del Sistema. 2. Il Sistema ripristina lo stato.
Requisiti speciali	
Elenco varianti tecnologiche e dati	
Frequenza ripetizioni	
Varie	

UC7: Visualizza congedi personali

Nome del caso d'uso	UC7: Visualizza congedi personali
Portata	Applicazione BenefitFlow
Livello	Obiettivo azienda
Attore primario	Dipendente
Parti interessate e interessi	- Dipendente vuole visualizzare tutti i propri congedi
Pre-condizioni	
Garanzia di successo	I congedi richiesti vengono mostrati al Dipendente
Scenario principale di successo	<ol style="list-style-type: none"> 1. Il Dipendente vuole visualizzare tutti i suoi congedi. 2. Il Dipendente sceglie l'attività "visualizza congedi". 3. Il Dipendente sceglie il tipo di congedo che vuole visualizzare in base alla propria matricola. 4. Il Sistema mostra l'elenco di congedi richiesti.
Estensioni	<p>*a. In un qualsiasi momento il Sistema fallisce e si arresta.</p> <ol style="list-style-type: none"> 1. Il Dipendente riavvia il software e ripristina lo stato precedente del Sistema. 2. Il Sistema ripristina lo stato.
Requisiti speciali	
Elenco varianti tecnologiche e dati	
Frequenza ripetizioni	
Varie	

UC8: Assegna buoni pasto

Nome del caso d'uso	UC8: Assegna buoni pasto
Portata	Applicazione BenefitFlow
Livello	Obiettivo azienda
Attore primario	Amministratore
Parti interessate e interessi	- Amministratore: vuole assegnare i buoni pasto per ogni dipendente
Pre-condizioni	
Garanzia di successo	Il buono pasto gestito dall'amministratore viene assegnato al dipendente. Il sistema registra le informazioni relative al buono pasto.
Scenario principale di successo	<ol style="list-style-type: none"> 1. L'Amministratore vuole assegnare un buono pasto al singolo Dipendente. 2. L'Amministratore scegli l'attività "assegna buoni pasto". 3. L'Amministratore richiede la lista relativa ai Dipendenti. 4. Il Sistema mostra ogni Dipendente, con relativi matricola e livello. 5. L'Amministratore crea un buono a favore di un dipendente fornendo la relativa matricola. 6. Il Sistema genera per quel buono pasto un codice univoco, assegna un valore monetario in base al livello ricoperto dal Dipendente e una data di scadenza. 7. L'Amministratore indica di aver finito. Il Sistema registra l'informazione.
Estensioni	<p>*a. In un qualsiasi momento il Sistema fallisce e si arresta.</p> <ol style="list-style-type: none"> 1. L'Amministratore riavvia il software e ripristina lo stato precedente del Sistema. 2. Il Sistema ripristina lo stato. <p>5a. L'Amministratore inserisce una matricola errata.</p> <ol style="list-style-type: none"> 1. Il sistema genera un codice di errore in quanto non è associata a nessun dipendente. 2. L'Amministratore ripete il passaggio 5 inserendo una matricola diversa.
Requisiti speciali	
Elenco varianti tecnologiche e dati	
Frequenza ripetizioni	Legata all'assegnazione dei buoni pasto
Varie	

UC9: Gestisci buono pasto

Nome del caso d'uso	UC9: Gestisci buono pasto
Portata	Applicazione BenefitFlow
Livello	Obiettivo azienda
Attore primario	Dipendente
Parti interessate e interessi	- Dipendente: vuole attivare i propri buoni pasto
Pre-condizioni	Il Dipendente deve poter conoscere quale sono i suoi buoni pasto validi.
Garanzia di successo	Lo stato del buono pasto viene modificato dal Sistema una volta che questo viene attivato.
Scenario principale di successo	<ol style="list-style-type: none"> Il Dipendente vuole attivare uno dei suoi buoni pasto validi. Il Dipendente sceglie l'attività "gestisci buono pasto". Il Dipendente visualizza, dall'elenco completo dei buoni pasto, la lista dei propri in base alla matricola. Il Sistema mostra la lista precedentemente richiesta. Il Dipendente, richiede l'attivazione del buono pasto fornendo il codice associato. Il Sistema conferma l'attivazione modificandone lo stato.
Estensioni	<p>*a. In un qualsiasi momento il Sistema fallisce e si arresta.</p> <ol style="list-style-type: none"> Il Dipendente riavvia il software e ripristina lo stato precedente del Sistema. Il Sistema ripristina lo stato. <p>5a. Il Dipendente fornisce il codice di un BuonoPasto diverso da quello possibile o non esistente.</p> <ol style="list-style-type: none"> Il sistema genera un codice di errore in quanto non è associato a nessun buono pasto. Il Dipendente ripete il passaggio 5 inserendo un codice diverso.
Requisiti speciali	
Elenco varianti tecnologiche e dati	
Frequenza ripetizioni	Legata alle attivazioni dei buoni pasto che il dipendente vuole effettuare.
Varie	

UC10: Visualizza buoni pasto personali

Nome del caso d'uso	UC10: Visualizza buoni pasto personali
Portata	Applicazione BenefitFlow
Livello	Obiettivo azienda
Attore primario	Dipendente
Parti interessate e interessi	- Dipendente vuole visualizzare tutti i propri buoni pasto
Pre-condizioni	
Garanzia di successo	I buoni pasto vengono mostrati al Dipendente
Scenario principale di successo	<ol style="list-style-type: none"> Il Dipendente vuole visualizzare tutti i suoi buoni pasto. Il Dipendente sceglie l'attività "visualizza buoni pasto". Il Dipendente visualizza la lista dei propri buoni pasto in base alla sua matricola. Il Sistema mostra l'elenco richiesto.
Estensioni	<p>*a. In un qualsiasi momento il Sistema fallisce e si arresta.</p> <ol style="list-style-type: none"> Il Dipendente riavvia il software e ripristina lo stato precedente del Sistema. Il Sistema ripristina lo stato.
Requisiti speciali	
Elenco varianti tecnologiche e dati	
Frequenza ripetizioni	
Varie	

1.5 Regole di dominio

Per il corretto funzionamento del software devono essere rispettate le seguenti regole di dominio:

ID	Regola	Modificabilità	Sorgente
R1	In base al livello del ruolo assegnato al dipendente viene assegnato un valore al buono pasto.	Bassa	Politica interna dell'azienda

1.6 Glossario

Vengono riportati qui i termini più significativi e le relative definizioni

Termine	Definizioni
Amministratore	Gestore dell'azienda, che si occupa dell'attribuzioni di tutti i benefit che spettano ad ogni dipendente
Dipendente	Personale dell'azienda, gestito dall'amministratore e che usufruisce dei servizi da lui assegnati
Ferie	Periodo di tempo, gestito dall'amministratore e richiesto dal dipendente per potersi assentare
Permessi	Rappresentano un periodo di tempo espresso in ore, richiesto dal dipendente al fine di potersi assentare per un motivo specifico previa approvazione dell'amministratore
Buoni Pasto	Ulteriore forma di benefit rilasciata dall'amministratore in base al livello e al ruolo assunto dal dipendente
Lista congedi	Rappresenta la lista di ferie e permessi attribuiti al dipendente durante un arco temporale
Ruolo	Incarico ricoperto del dipendente all'interno dell'azienda, assegnato dall'amministratore
Livello	Posizione gerarchica ricoperta dal dipendente legata al ruolo che svolge, assegnato dall'amministratore

2. Analisi Orientata agli Oggetti

2.1 Introduzione

L'implementazione dell'applicazione è stata guidata da un processo iterativo ed evolutivo basato sul modello UP (Unified Process), suddiviso in tre fasi. Grazie a questo metodo è stato possibile sviluppare gradualmente l'architettura del software BenefitFlow.

Ciascuna iterazione ha affrontato diverse problematiche gestendo:

- **Iterazione 1:**

- Implementazione degli scenari *UC1: Inserisci nuovo dipendente* e *UC2: Gestisci ruolo dipendente*.

- **Iterazione 2:**

- Implementazione degli scenari *UC3: Inserisci ferie o permessi*, *UC4: Gestisci Congedi*, *UC5: Gestisci date assegnate*, *UC6: Visualizza congedi* e *UC7: Visualizza congedi personali*.

L'analisi congiunta di questi scenari è finalizzata a ottenere una comprensione completa delle attività che amministratore e dipendente possono svolgere relativamente alla gestione dei congedi.

- **Iterazione 3:**

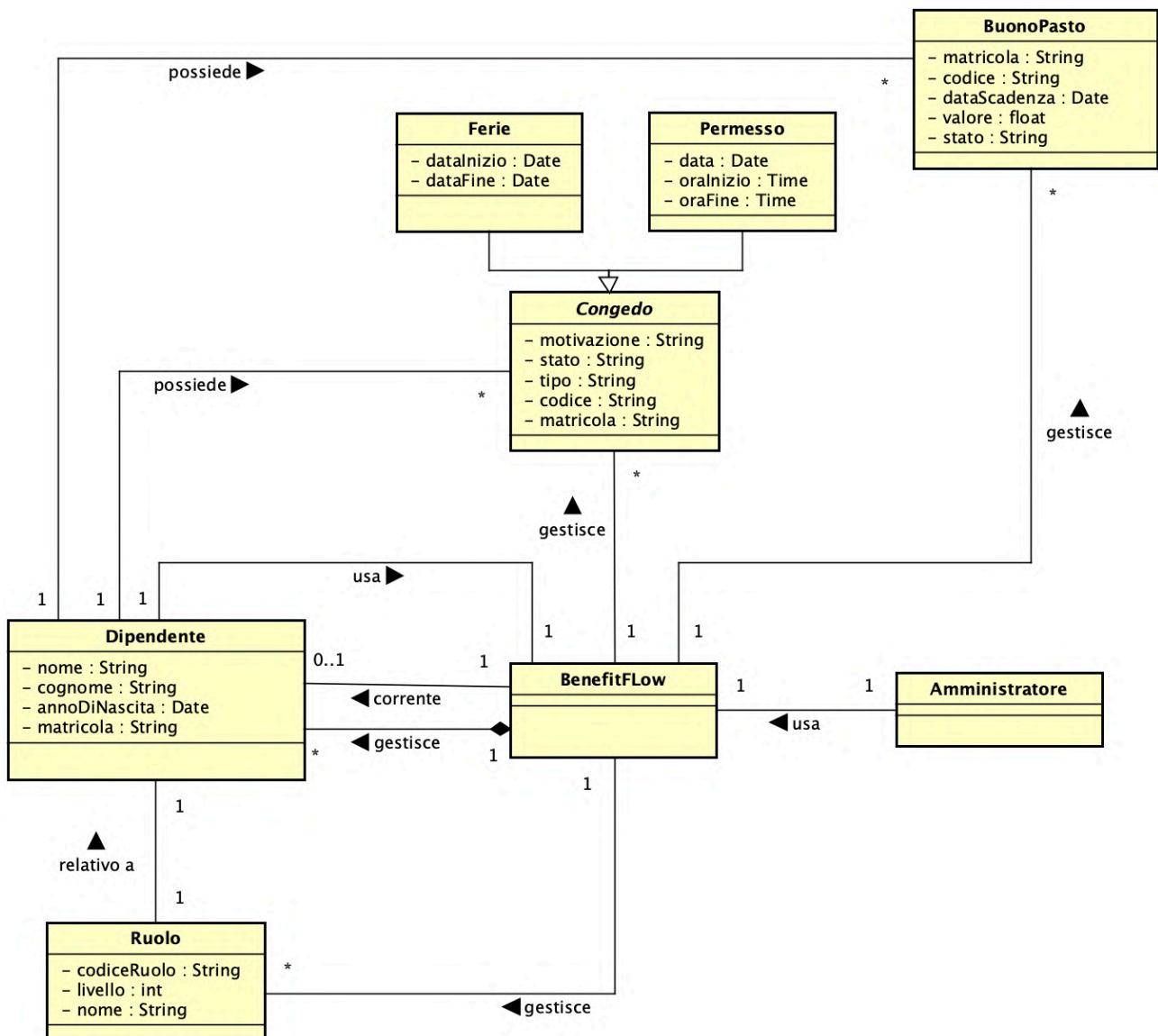
- Implementazione degli scenari *UC8: Assegna buoni pasto*, *UC9: Gestisci buono pasto* e *UC10: Visualizza buoni pasto personali*.

L'analisi congiunta di questi scenari è finalizzata a ottenere una comprensione completa delle attività che amministratore e dipendente possono svolgere relativamente alla gestione dei buoni pasto.

2.2 Modello di Dominio

La Modellazione del Business è la disciplina che si occupa di fornire una visione completa del dominio. In particolare, essa include la creazione del Modello di Dominio, un'elaborato grafico che individua concetti, attributi e associazioni rilevanti.

Il modello di dominio viene iterato tenendo conto di ciascuna Iterazione e si presenta come:



Sono state individuate le seguenti classi:

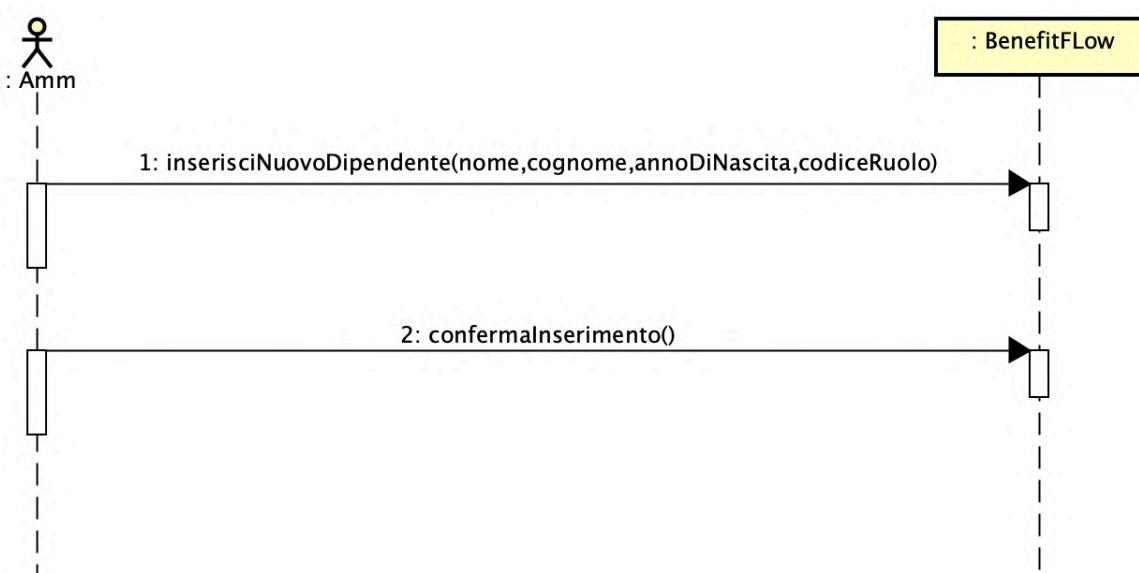
- **BenefitFlow**: rappresenta il sistema BenefitFlow
- **Amministratore**: attore primario che interagisce con il sistema
- **Dipendente**: attore secondario che interagisce con il sistema
- **Ruolo**: rappresenta la posizione ricoperta dal dipendente all'interno dell'azienda
- **Congedo**: rappresenta le diverse tipologie di congedi che il dipendente può richiedere.
- **Ferie**: indica il congedo specifico. Il dipendente può farne richiesta fornendo una data di inizio e una data di fine.
- **Permesso**: indica il congedo specifico. Il dipendente può farne richiesta fornendo una data, un'orario d'inizio e uno di fine.
- **BuonoPasto**: rappresenta un titolo di pagamento fornito dall'amministratore ai dipendenti

2.3 SSD e Contratti

Continuando con l'approccio Orientato agli Oggetti il passaggio successivo coinvolge la realizzazione dei Diagrammi di Sequenza di Sistema (SSD) per delineare il flusso degli eventi di input e output relativi ai diversi casi d'uso esaminati in ogni iterazione. Le principali funzionalità del sistema, identificate nei SSD, saranno espresse attraverso i Contratti. Pertanto avremo:

- **Iterazione 1**

UC1: Inserisci nuovo dipendente



Contratto CO1: inserisciNuovoDipendente

Operazione:

`inserisciNuovoDipendente(nome,cognome,annoDiNascita,codiceRuolo)`

Riferimenti:

Caso d'uso: Inserisci nuovo dipendente

Pre-Condizioni:

- Esiste una lista di Ruoli r;
- L'amministratore si autentica in BenefitFlow.

Post-Condizioni:

- È stata creata un'istanza d di Dipendente;
- Gli attributi di d sono stati inizializzati;
- È stata recuperata l'istanza r di Ruolo sulla base di codiceRuolo;
- L'istanza r è stata associata a d tramite l'associazione "relativo a";
- L'istanza d è stata associata a BenefitFlow tramite l'associazione "corrente".

Contratto CO2: confermaInserimento

Operazione:

`confermaInserimento()`

Riferimenti:

Caso d'uso: Inserisci nuovo dipendente

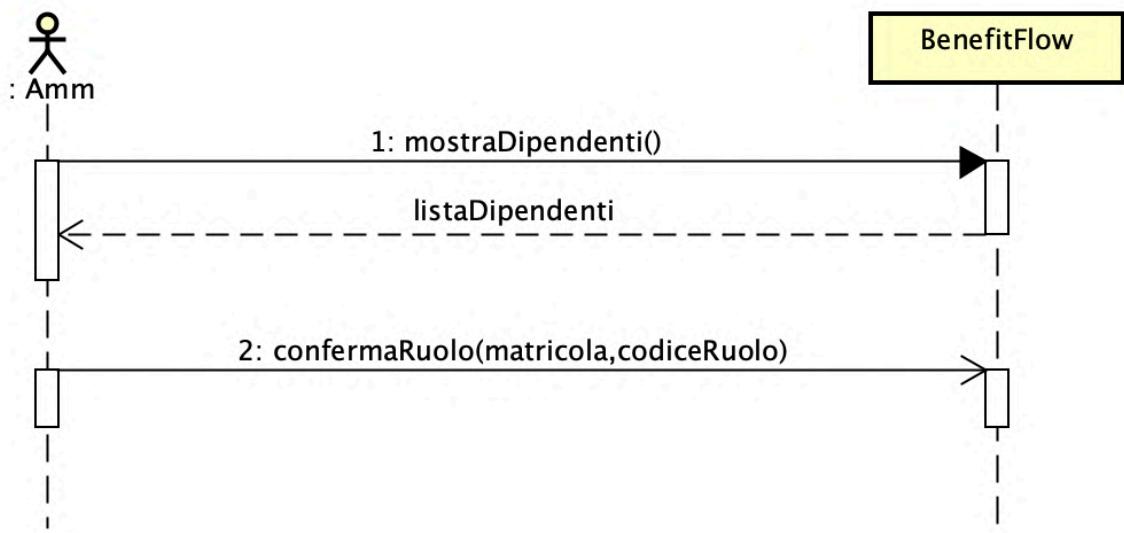
Pre-Condizioni:

- È in corso l'inserimento del Dipendente d.

Post-Condizioni:

- È stata associata l'istanza d di Dipendente corrente a BenefitFlow tramite l'associazione "gestisce".

UC2: Gestisci ruolo dipendente



Contratto CO1: confermaRuolo

Operazione:

confermaRuolo(matricola,codiceRuolo)

Riferimenti:

Caso d'uso: Gestisci ruolo dipendente

Pre-Condizioni:

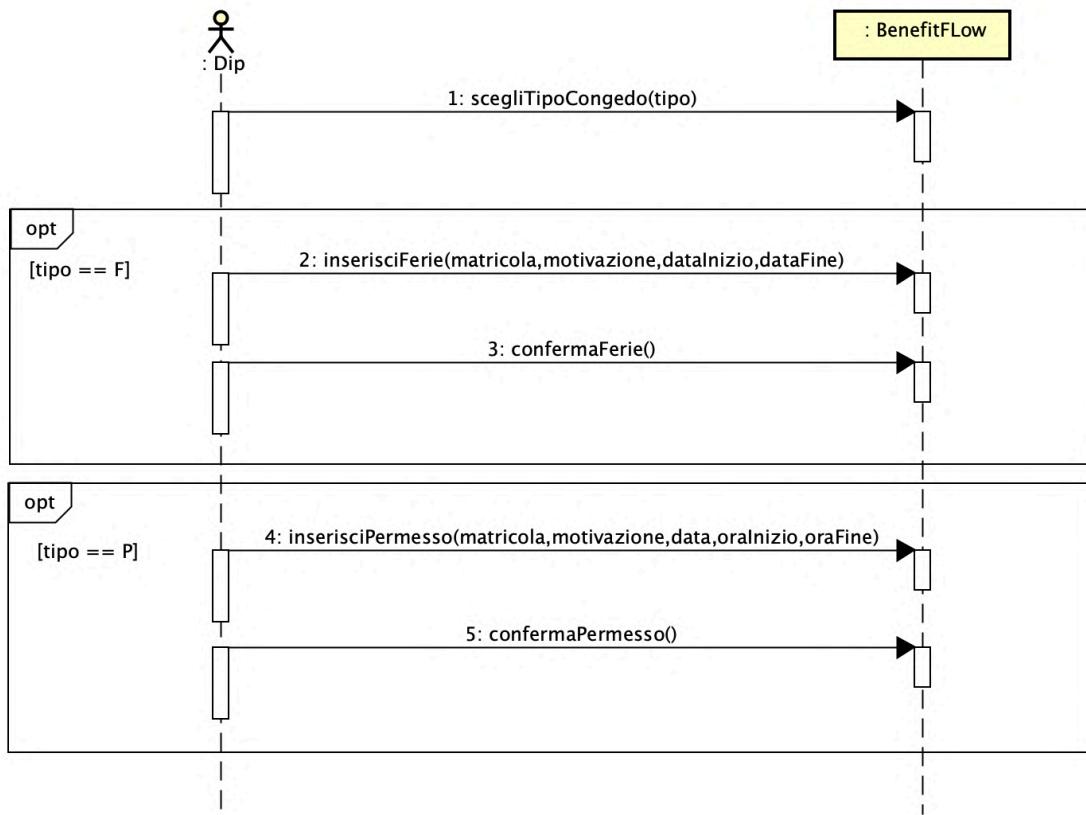
- È in corso l'assegnazione del Ruolo r.

Post-Condizioni:

- È stata recuperata l'istanza di d sulla base di matricola;
- È stata recuperata l'istanza di r da d;
- È stato inizializzato l'attributo codiceRuolo del Ruolo r su d.

• Iterazione 2

UC3: Inserisci ferie o permessi



Contratto CO1: inserisciFerie

Operazione:

`inserisciFerie(matricola, motivazione, dataInizio, dataFine)`

Riferimenti:

Caso d'uso: Inserisci ferie o permessi

Pre-Condizioni:

- Il dipendente si autentica in BenefitFlow.

Post-Condizioni:

- È stata creata un'istanza f di Ferie;
- Gli attributi di f sono stati inizializzati;
- È stata recuperata l'istanza d di Dipendente sulla base della matricola;
- L'istanza f è associata a d tramite l'associazione "possiede".

Contratto CO2: confermaFerie

Operazione:

confermaFerie()

Riferimenti:

Caso d'uso: Inserisci ferie o permessi

Pre-Condizioni:

- È in corso l'inserimento della Ferie f.

Post-Condizioni:

- È associata l'istanza f di Ferie a BenefitFlow tramite l'associazione "gestisce".

Contratto CO3: inserisciPermesso

Operazione:

inserisciPermesso(matricola,motivazione,data,oraInizio,oraFine)

Riferimenti:

Caso d'uso: Inserisci ferie o permessi

Pre-Condizioni:

- Il dipendente si autentica in BenefitFlow.

Post-Condizioni:

- È stata creata un'istanza p di Permesso;
- Gli attributi di p sono stati inizializzati;
- È stata recuperata l'istanza d di Dipendente sulla base della matricola;
- L'istanza p è associata a d tramite l'associazione "possiede".

Contratto CO4: confermaPermesso

Operazione:

confermaPermesso()

Riferimenti:

Caso d'uso: Inserisci ferie o permessi

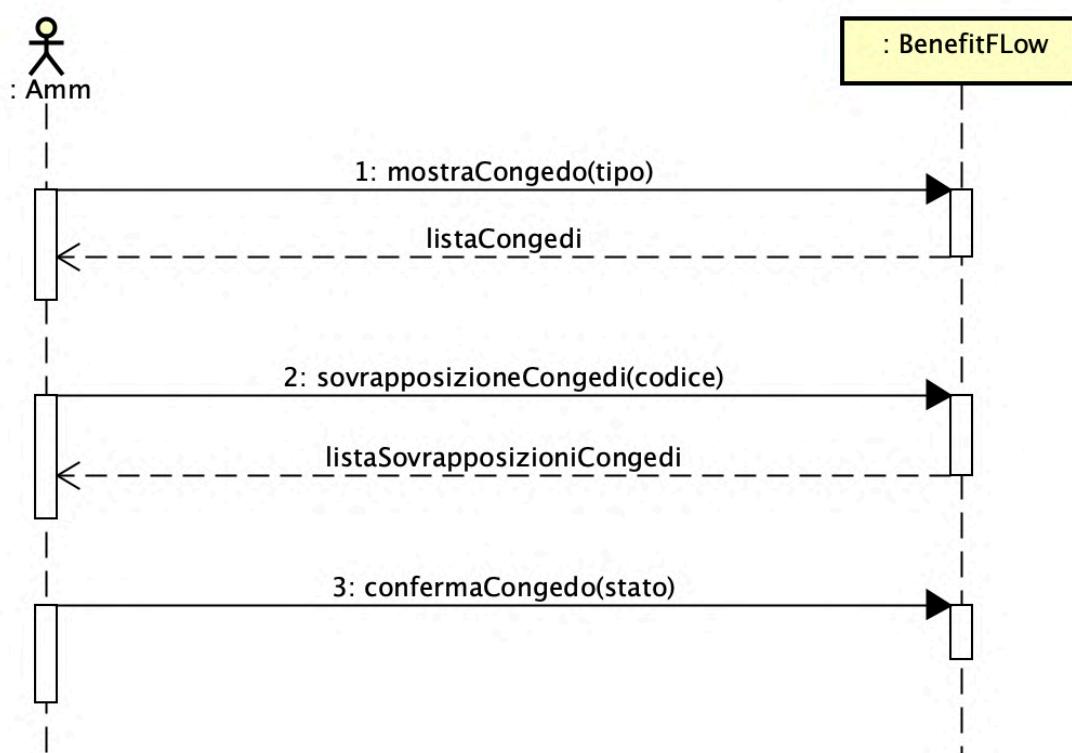
Pre-Condizioni:

- È in corso l'inserimento del Permesso p.

Post-Condizioni:

- È associata l'istanza p di Permesso a BenefitFlow tramite l'associazione "gestisce".

UC4: Gestisci congedi



Contratto CO1: confermaCongedo

Operazione:

confermaCongedo(stato)

Riferimenti:

Caso d'uso: Gestisci congedi

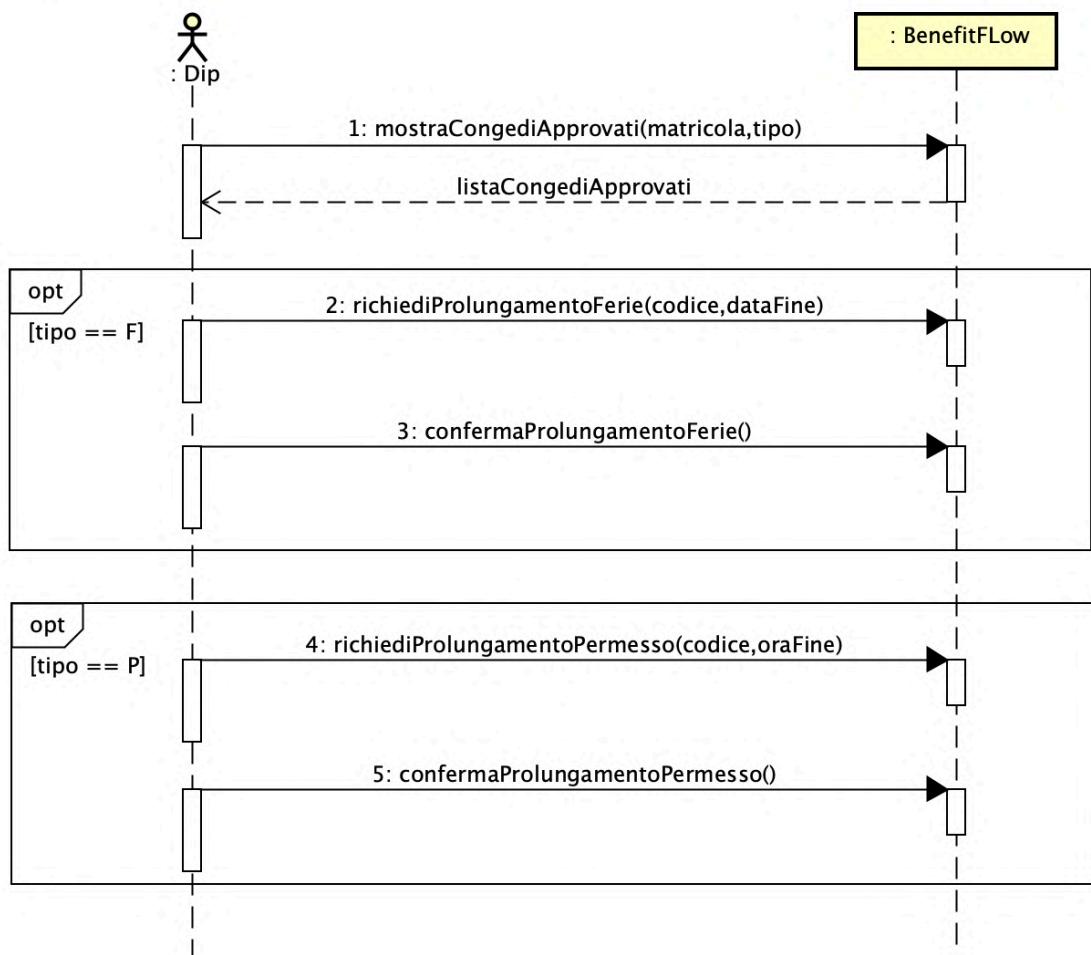
Pre-Condizioni:

- È in corso la modifica dello stato del Congedo c.

Post-Condizioni:

- È stato aggiornato l'attributo stato del Congedo c.

UC5: Gestisci date assegnate



Contratto CO1: richiediProlungamentoFerie

Operazione:

richiediProlungamentoFerie(codice,dataFine)

Riferimenti:

Caso d'uso: Gestisci date assegnate

Pre-Condizioni:

- È in corso la creazione di una nuova istanza di Ferie f.

Post-Condizioni:

- È stata recuperata la vecchia istanza f sulla base di codice;
- È stata creata una nuova istanza f di Ferie;
- Gli attributi di f sono stati inizializzati;
- L'istanza f è stata associata a d tramite l'associazione "possiede".

Contratto CO2: confermaProlungamentoFerie

Operazione:

confermaProlungamentoFerie()

Riferimenti:

Caso d'uso: Gestisci date assegnate

Pre-Condizioni:

- Esiste un'istanza f di Ferie.

Post-Condizioni:

- È associata l'istanza f di Ferie a BenefitFlow tramite l'associazione "gestisce".

Contratto CO3: richiediProlungamentoPermesso

Operazione:

richiediProlungamentoPermesso(codice,oraFine)

Riferimenti:

Caso d'uso: Gestisci date assegnate

Pre-Condizioni:

- È in corso la creazione di una nuova istanza di Permesso p.

Post-Condizioni:

- È stata recuperata la vecchia istanza p sulla base di codice;
- È stata creata una nuova istanza p di Permesso;
- Gli attributi di p sono stati inizializzati;
- L'istanza p è stata associata a d tramite l'associazione "possiede".

Contratto CO4: confermaProlungamentoPermesso

Operazione:

confermaProlungamentoPermesso()

Riferimenti:

Caso d'uso: Gestisci date assegnate

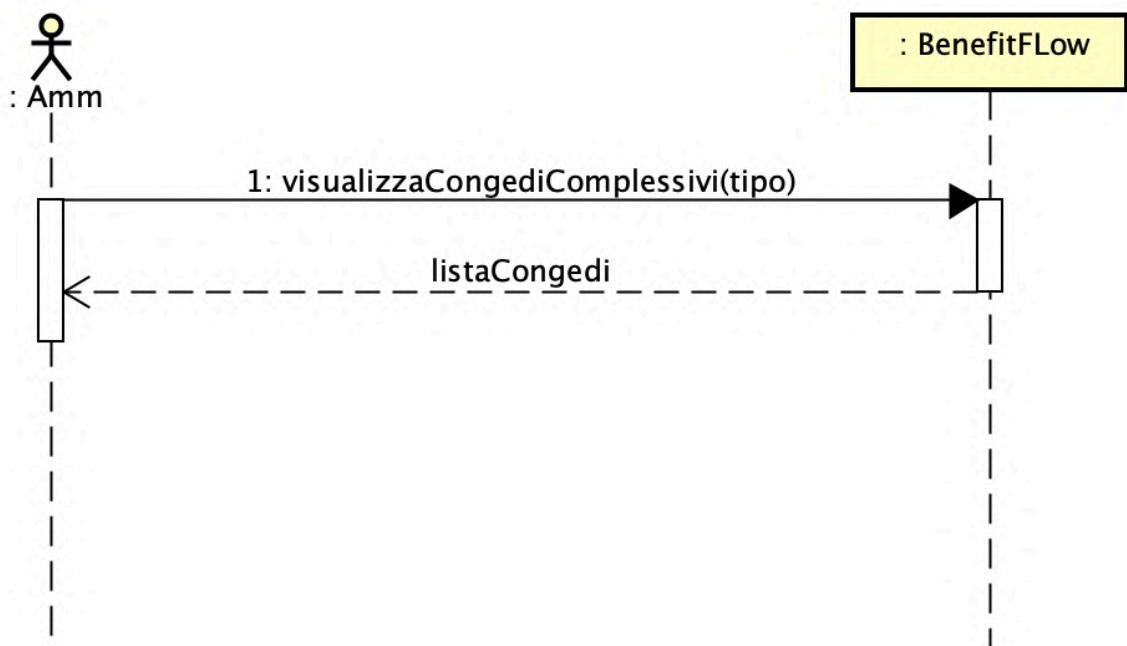
Pre-Condizioni:

- Esiste un'istanza p di Permesso.

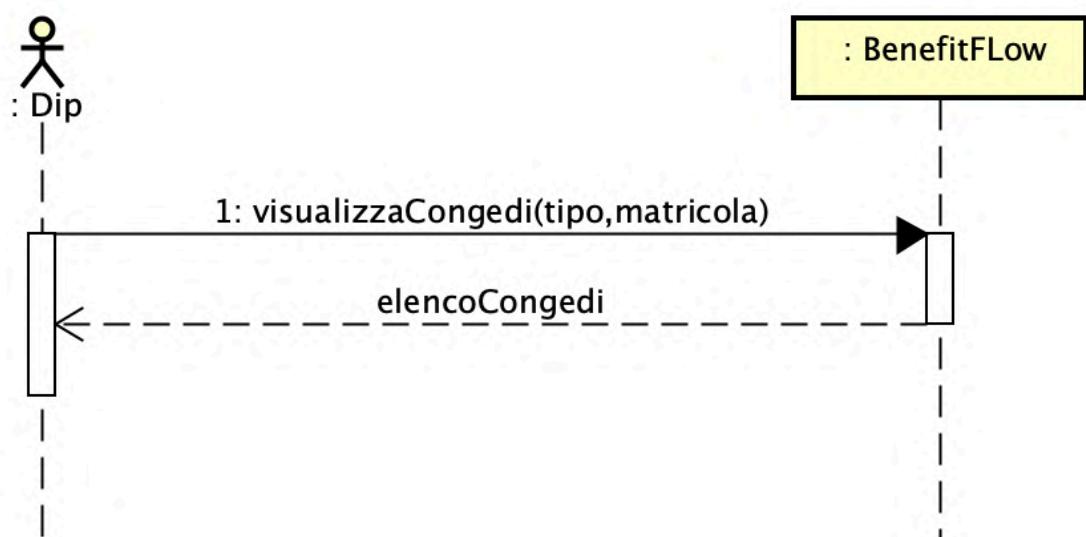
Post-Condizioni:

- È associata l'istanza p di Permesso a BenefitFlow tramite l'associazione "gestisce".

UC6: Visualizza congedi

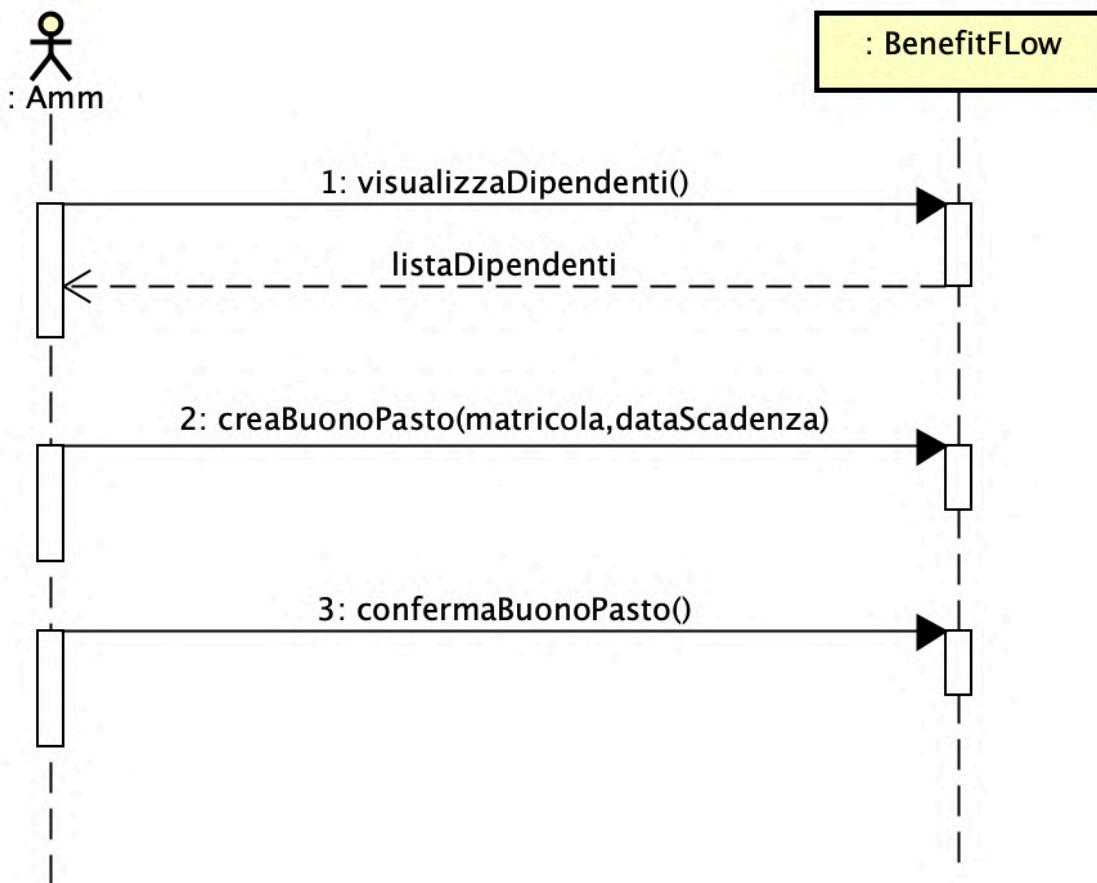


UC7: Visualizza congedi personali



- Iterazione 3

UC8: Assegna buoni pasto



Contratto CO1: creaBuonoPasto

Operazione:

creaBuonoPasto(matricola,dataScadenza)

Riferimenti:

Caso d'uso: Assegna buoni pasto

Pre-Condizioni:

- L'amministratore si autentica in BenefitFlow.

Post-Condizioni:

- È stata creata un'istanza bp di BuonoPasto;
- Gli attributi di bp sono stati inizializzati;
- È stato recuperata l'istanza d di Dipendente sulla base della matricola;
- È stato recuperato l'attributo livello dall'istanza d di Dipendente;
- È stato inizializzato l'attributo valore, sulla base del livello, mediante una strategia;
- L'istanza bp è associata a d tramite l'associazione "possiede".

Contratto CO2: confermaBuonoPasto

Operazione:

confermaBuonoPasto()

Riferimenti:

Caso d'uso: Assegna buoni pasto

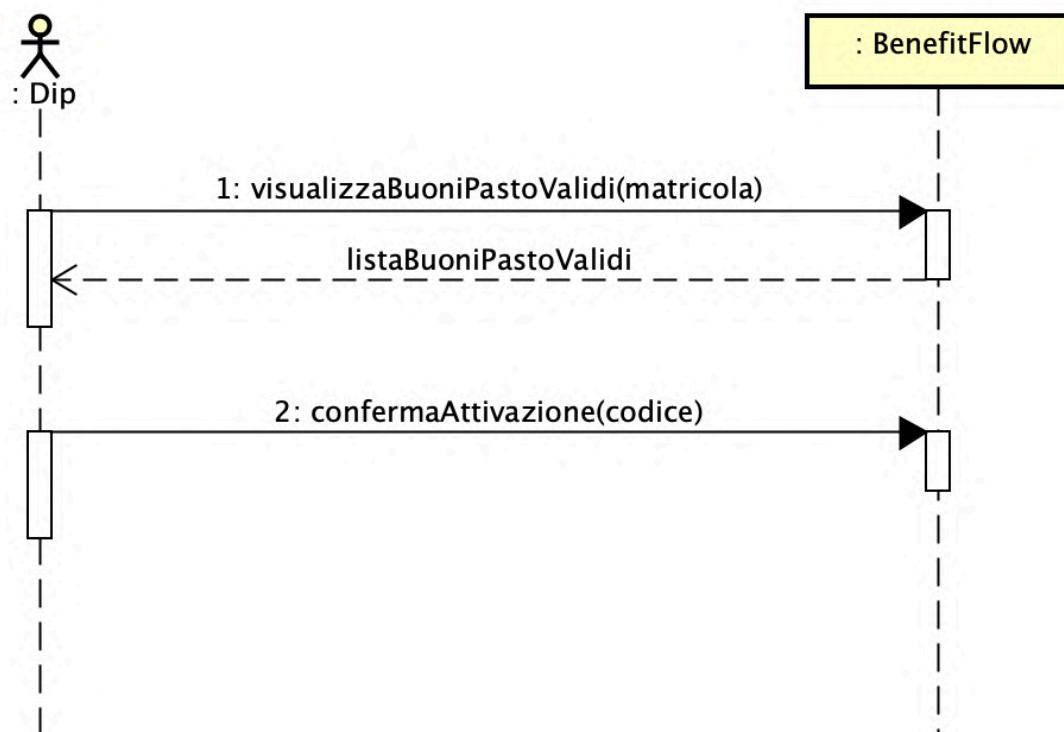
Pre-Condizioni:

- È in corso l'inserimento di un BuonoPasto bp.

Post-Condizioni:

- È associata l'istanza bp di BuonoPasto a BenefitFlow tramite l'associazione "gestisce".

UC9: Gestisci buono pasto



Contratto CO1: confermaAttivazione

Operazione:

confermaAttivazione(codice)

Riferimenti:

Caso d'uso: Gestisci buono pasto

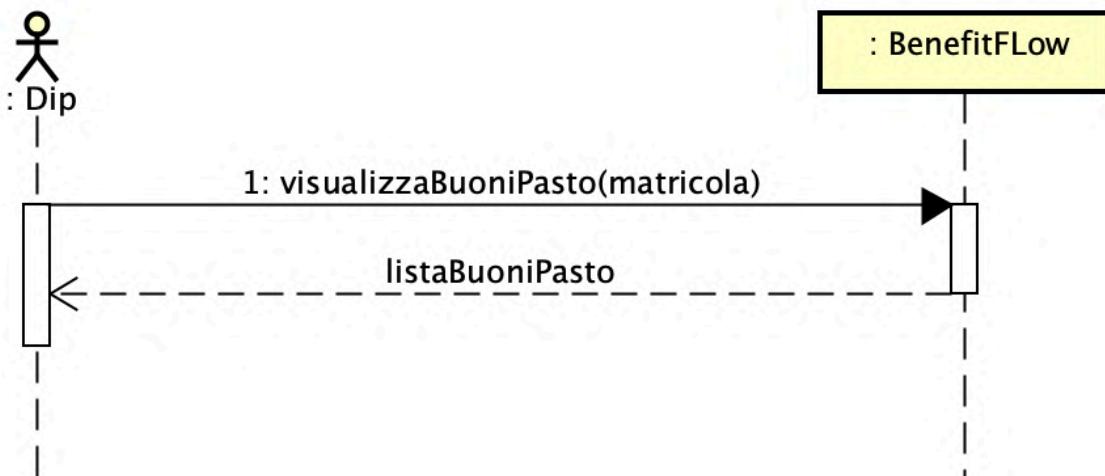
Pre-Condizioni:

- È in corso la modifica dello stato di bp.

Post-Condizioni:

- È stata recuperata l'istanza bp di BuonoPasto tramite il codice;
- È stato aggiornato l'attributo stato di bp.

UC10: Visualizza buoni pasto personali



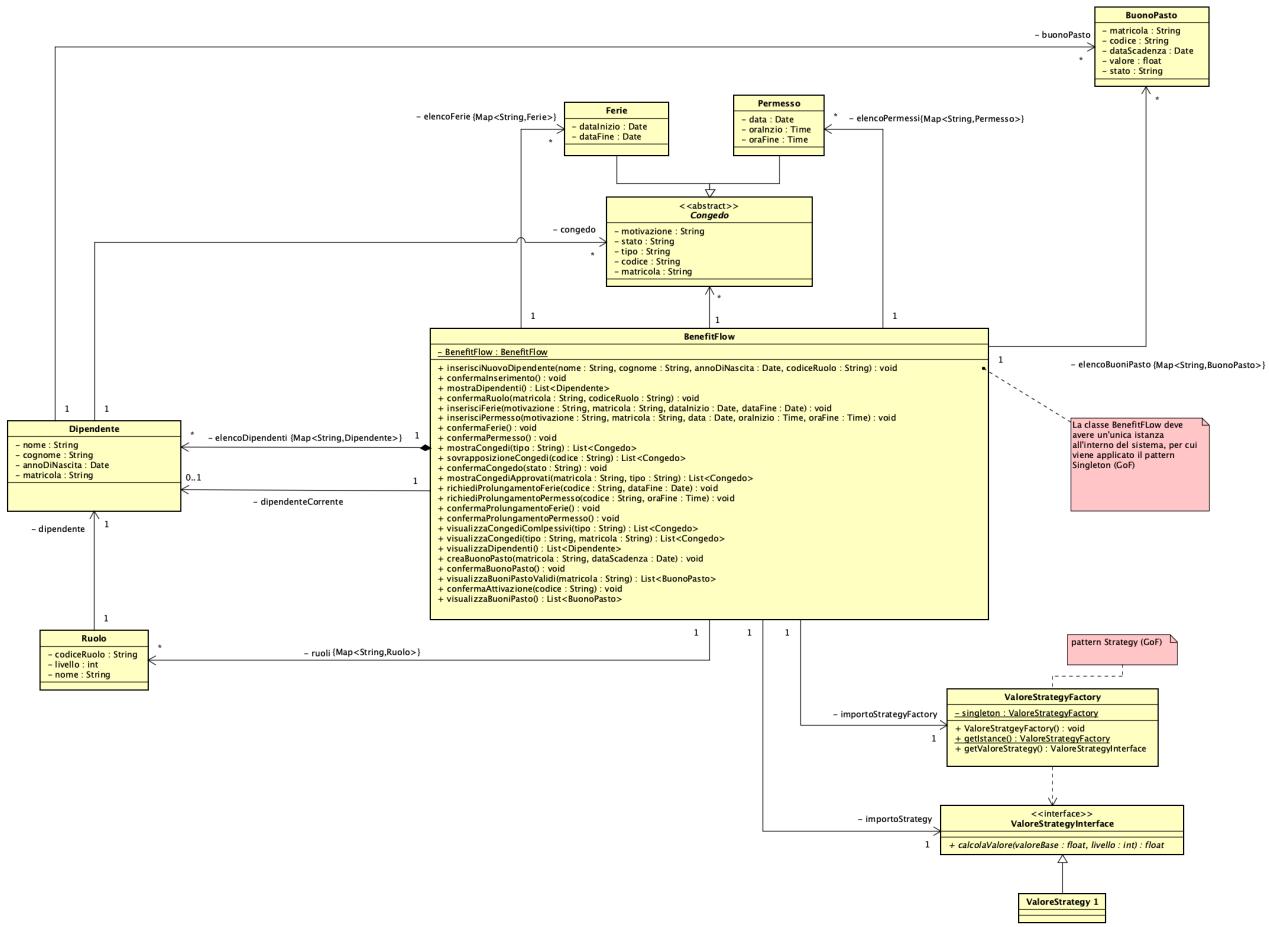
3. Progettazione

3.1 Diagrammi delle Classi

Durante la fase di progettazione, il focus è sul Modello di Progetto, che rappresenta una serie di diagrammi che descrivono la progettazione logica:

- Dinamicamente, attraverso i Diagrammi di Iterazione
- Staticamente, attraverso il Diagramma delle Classi

Il Diagramma delle Classi è:

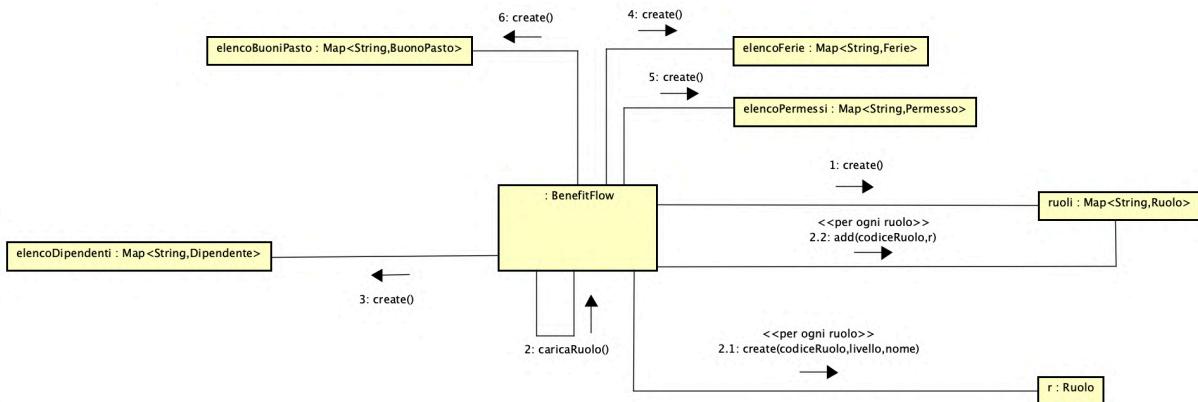


Si è deciso di utilizzare il pattern GoF **Strategy** per l'assegnazione del valore monetario dei singoli buoni pasto. Questo pattern, infatti, permette di modificare gli algoritmi o di aggiungerne di nuovi senza dover eseguire particolari modifiche sul codice, favorendone la flessibilità e la manutenibilità. In futuro sarà dunque possibile integrare nuove strategie a seconda delle sopravvenute esigenze.

3.2 Diagrammi di Sequenza

- **Caso d'Uso di Avviamento**

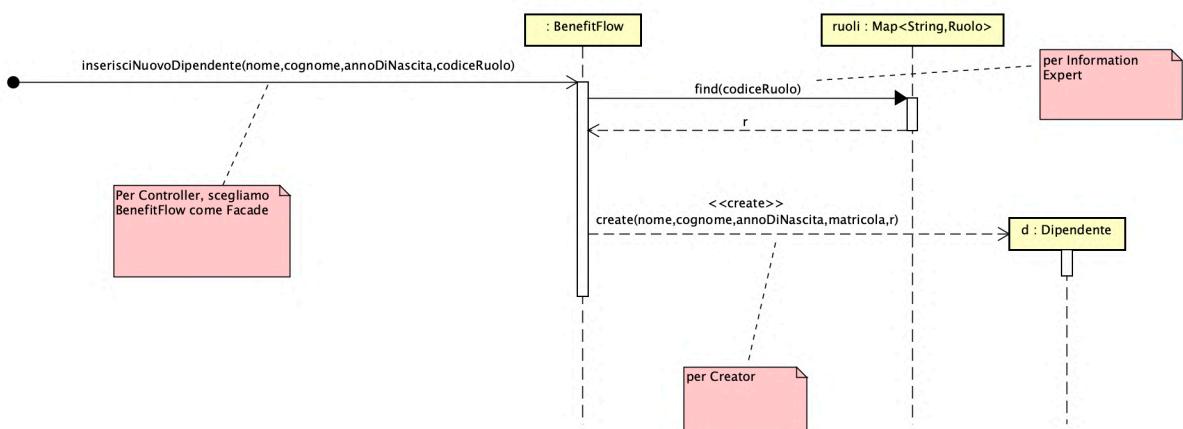
Per garantire che il sistema sia operativo fin dalla prima fase di sviluppo, è stato progettato un caso d'uso di avviamento che consentisse di creare una lista di dipendenti, ruoli e benefit.



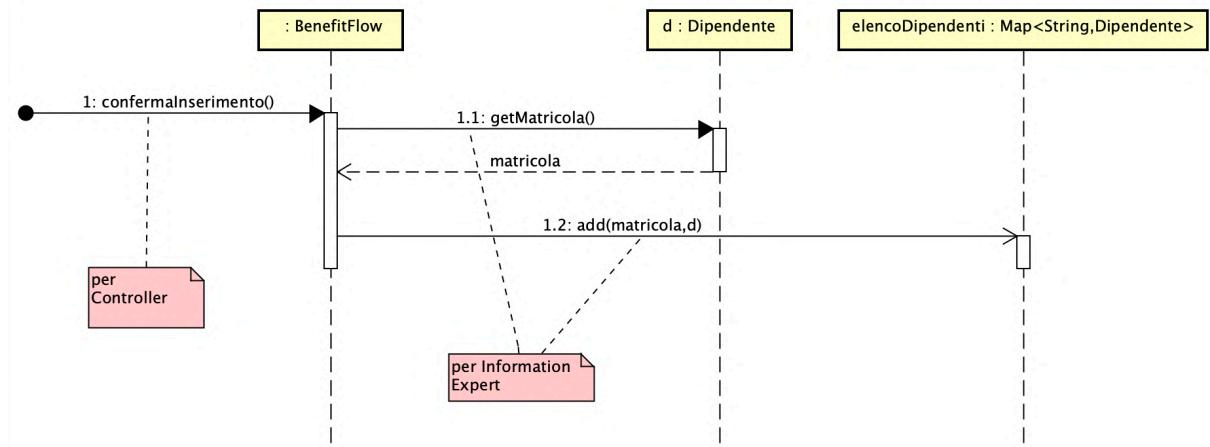
- **Iterazione 1**

UC1: Inserisci nuovo dipendente

SD inserisciNuovoDipendente

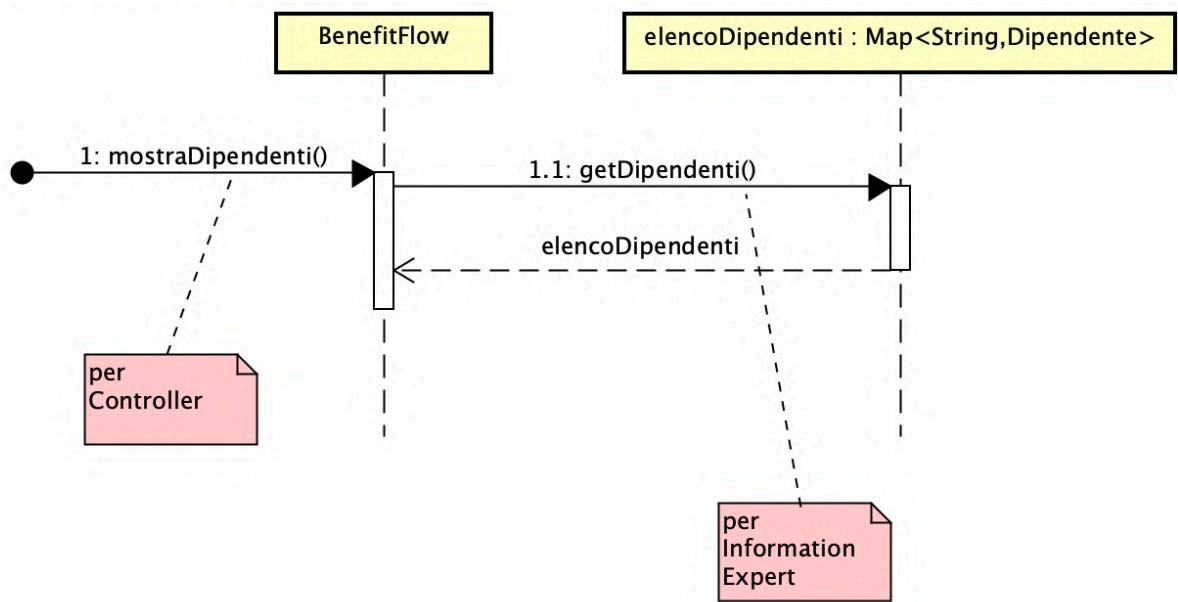


SD confermaInserimento

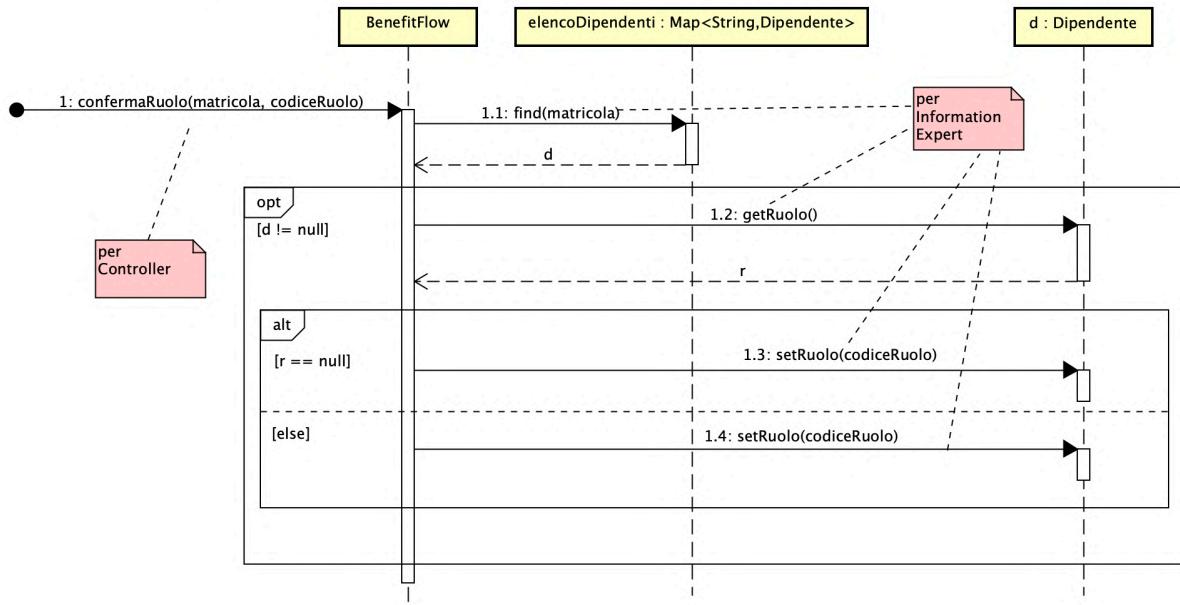


UC2: Gestisci ruolo dipendente

SD mostraDipendenti



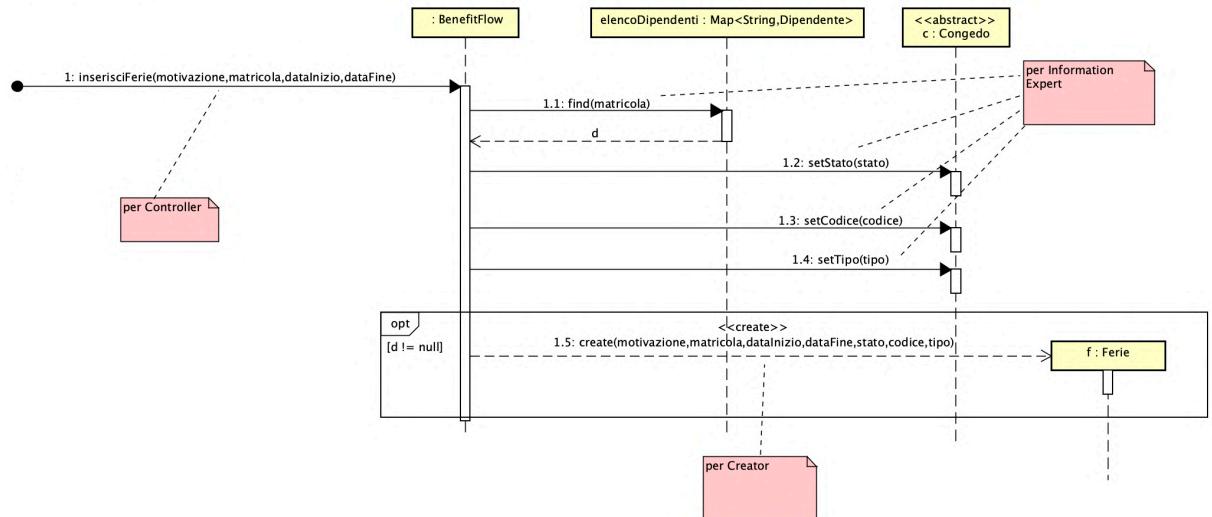
SD confermaRuolo



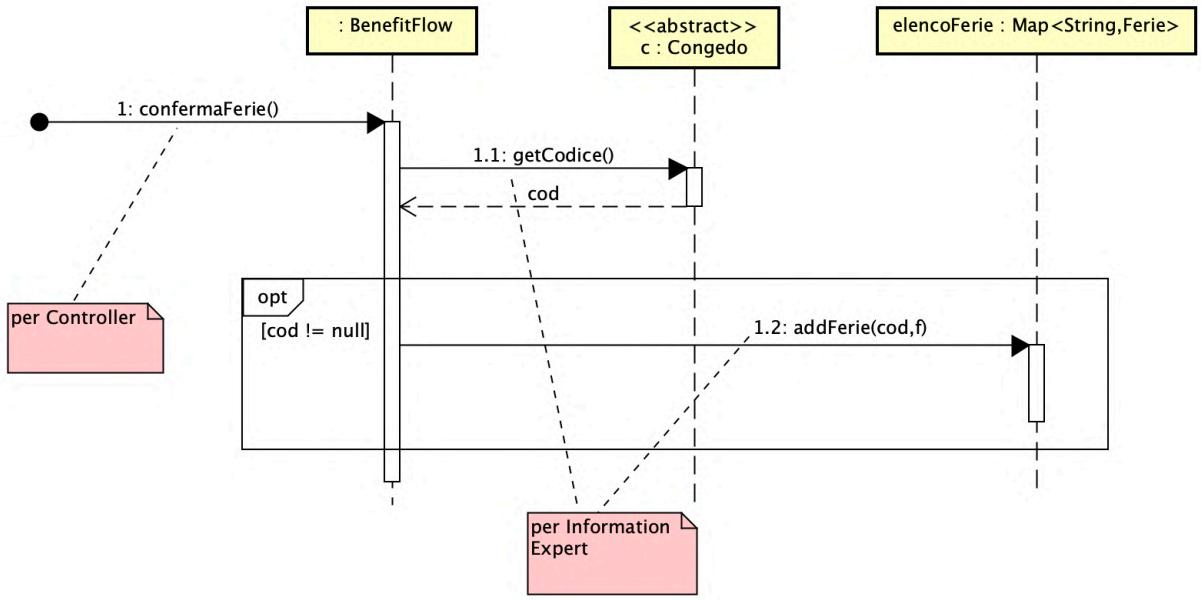
• Iterazione 2

UC3: Inserisci ferie o permessi

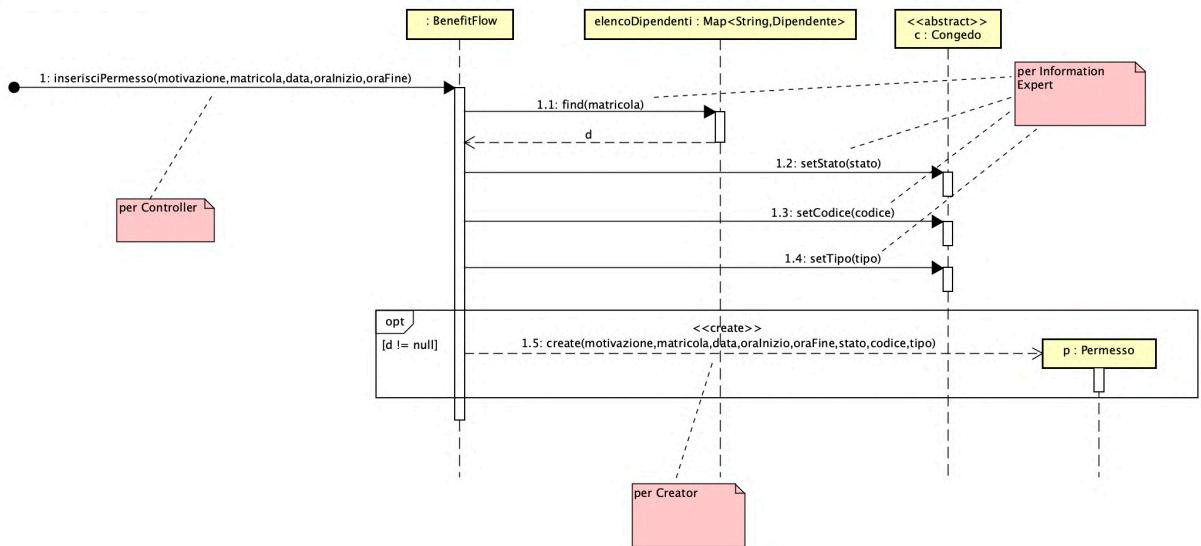
SD inserisciFerie



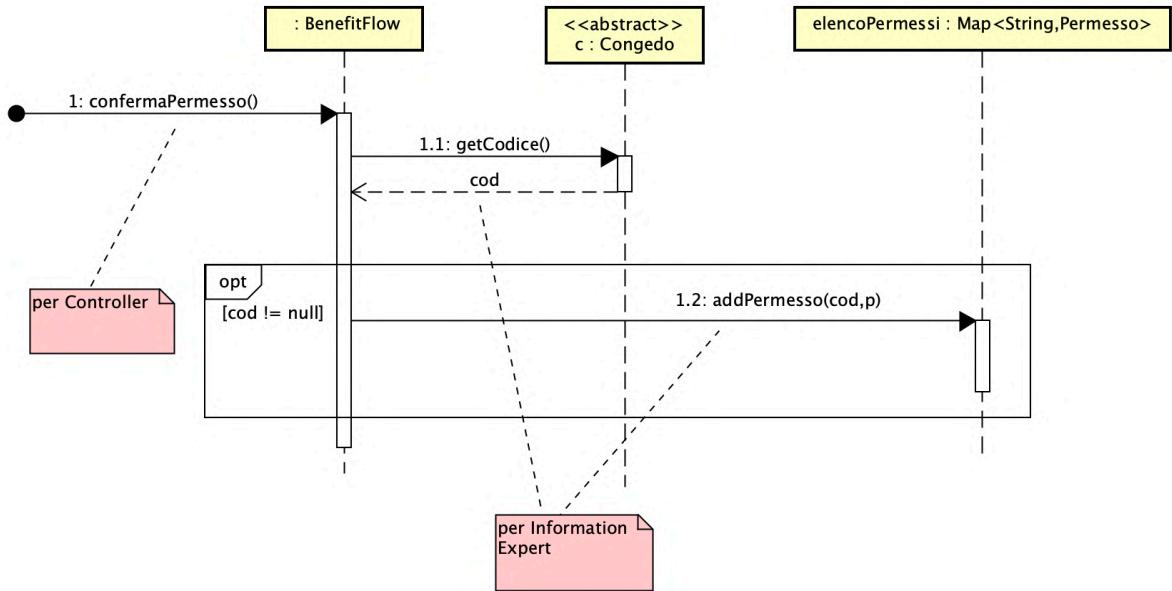
SD confermaFerie



SD inserisciPermesso

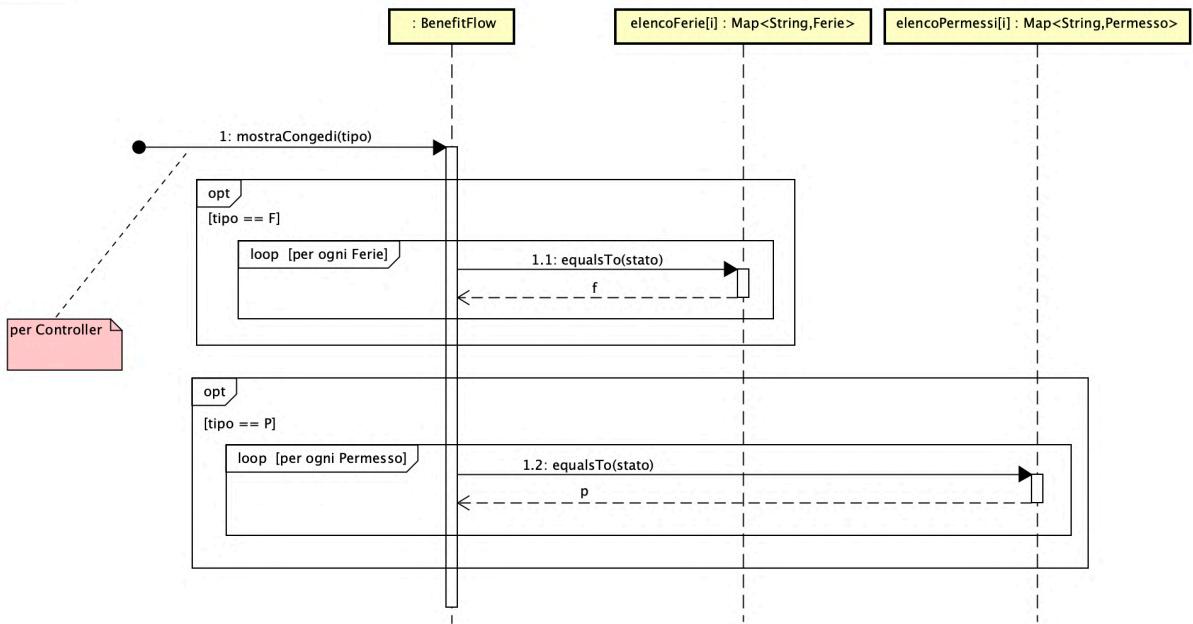


SD confermaPermesso

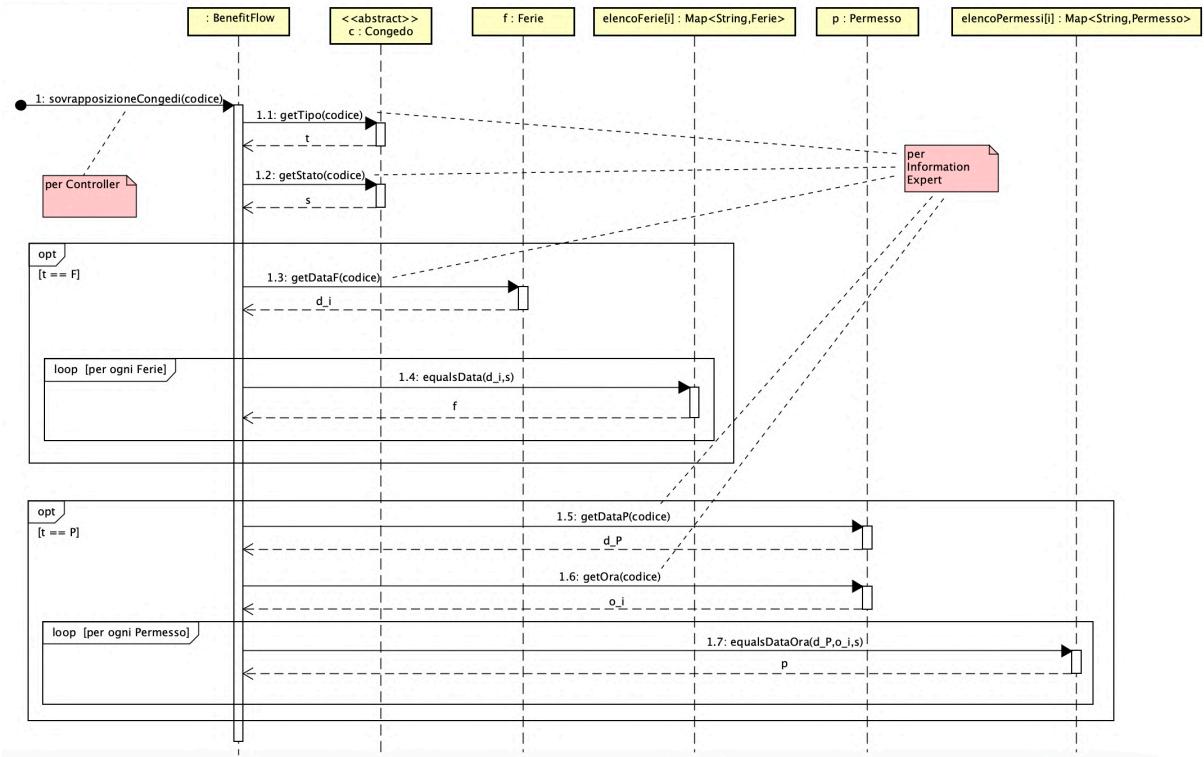


UC4: Gestisci congedi

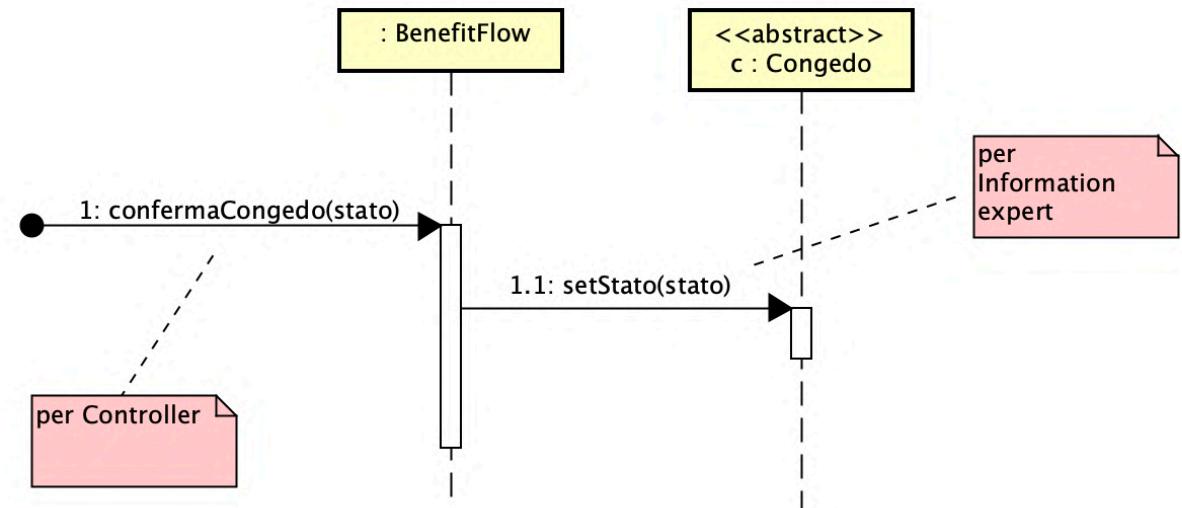
SD mostraCongedi



SD sovrapposizioneCongedi

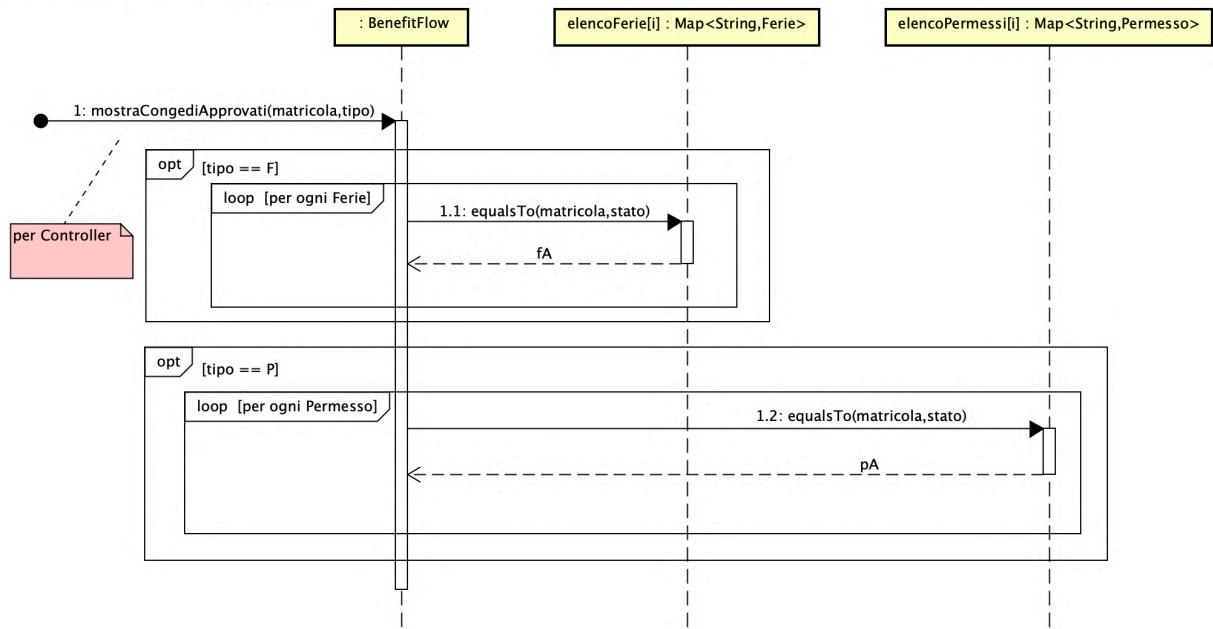


SD confermaCongedo

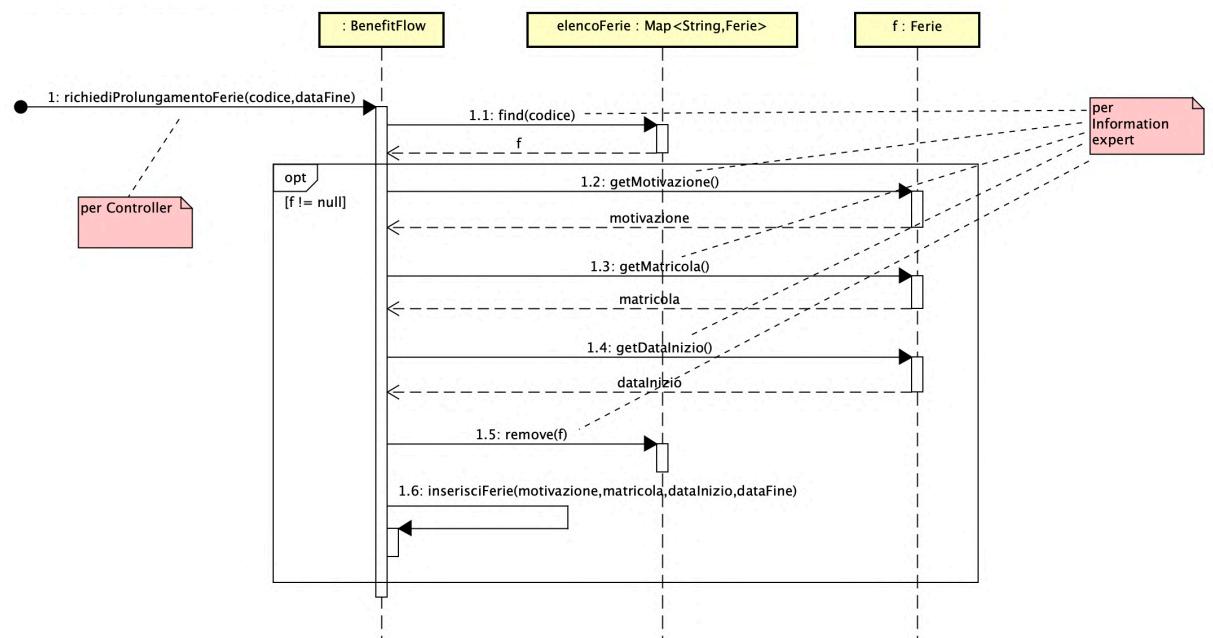


UC5: Gestisci date assegnate

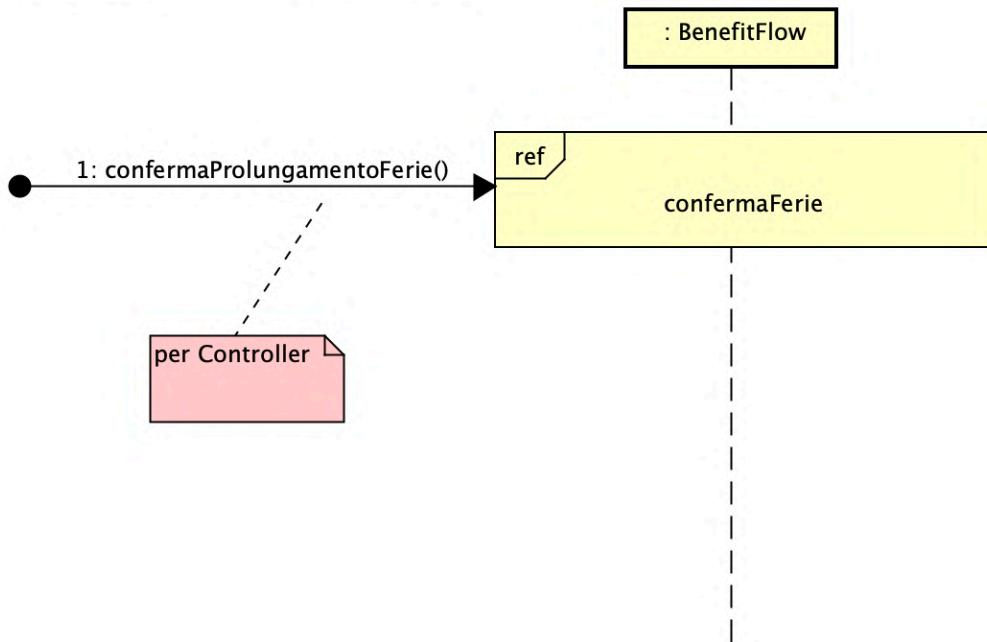
SD mostraCongediApprovati



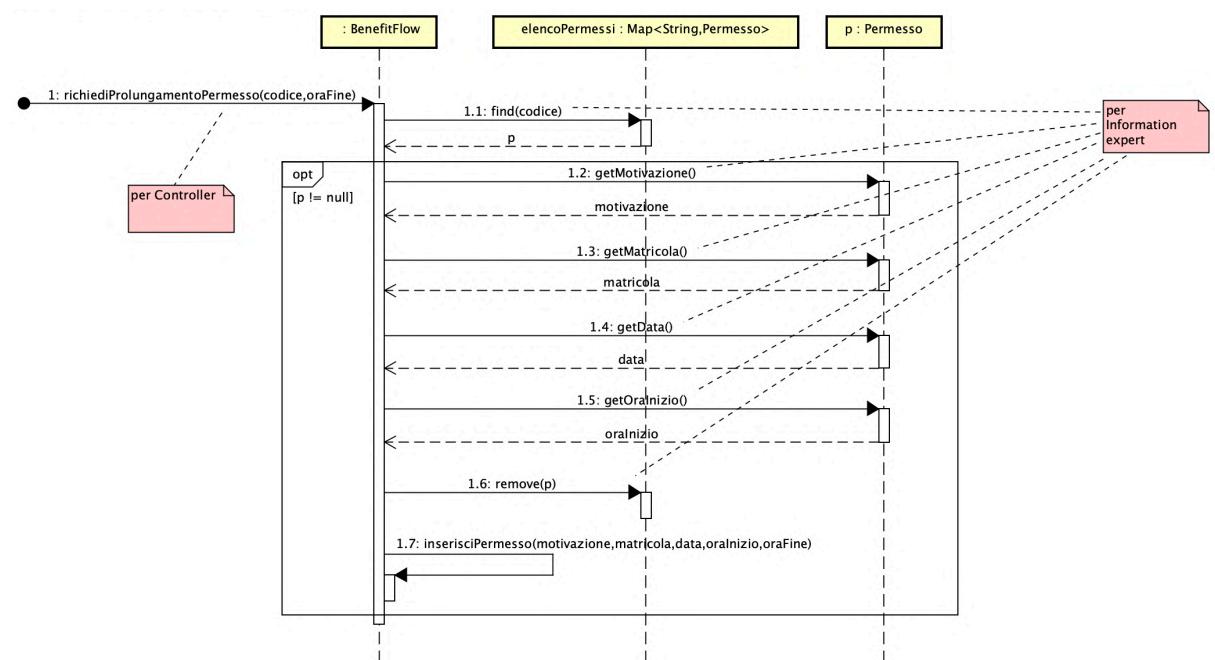
SD richiediProlungamentoFerie



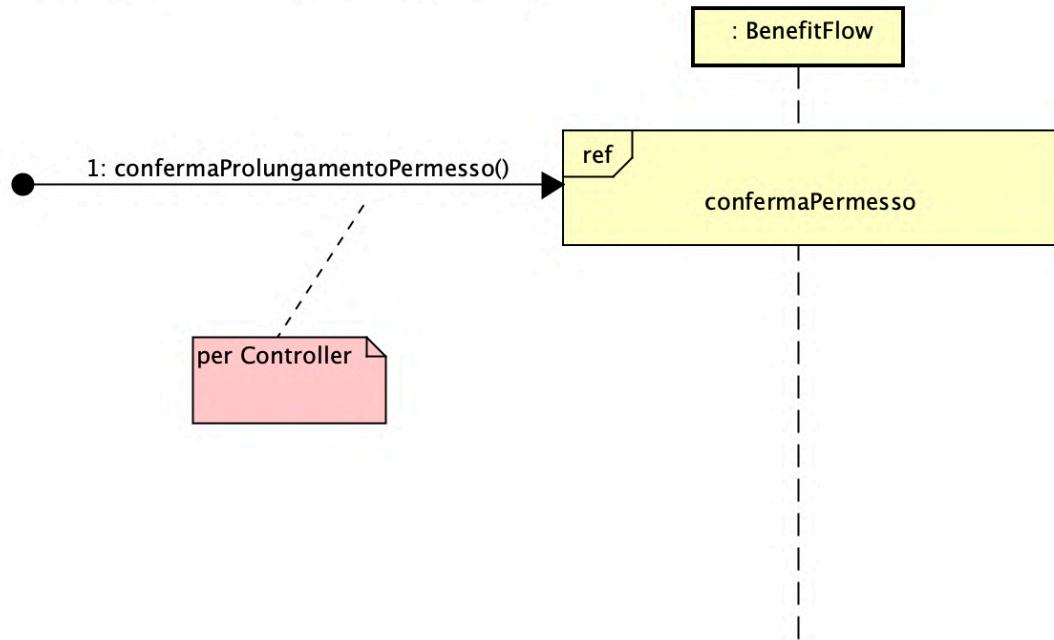
SD confermaProlungamentoFerie



SD richiediProlungamentoPermesso

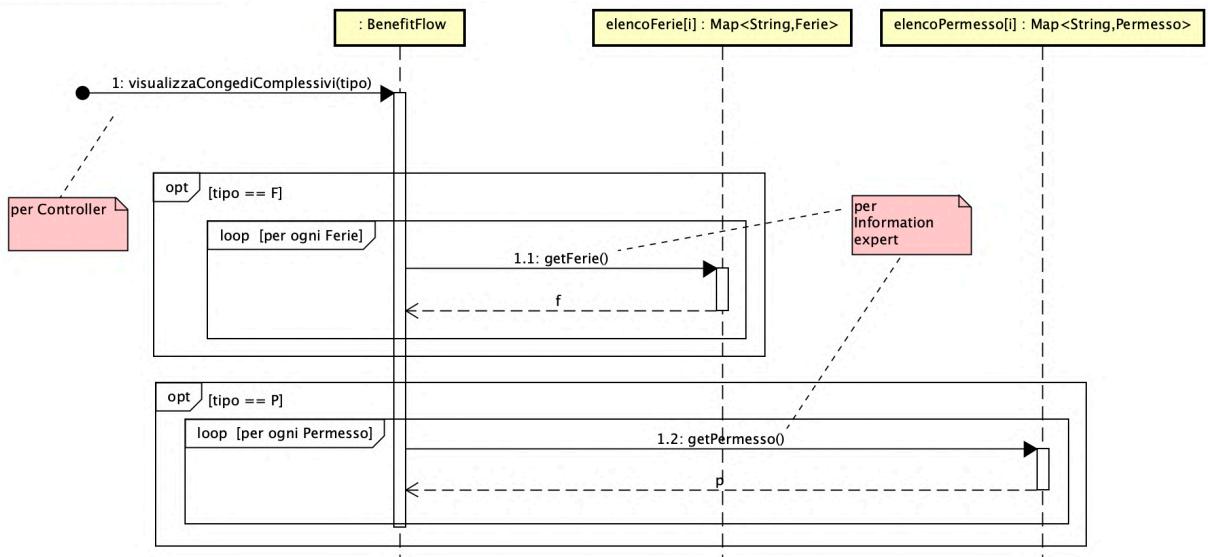


SD confermaProlungamentoPermesso



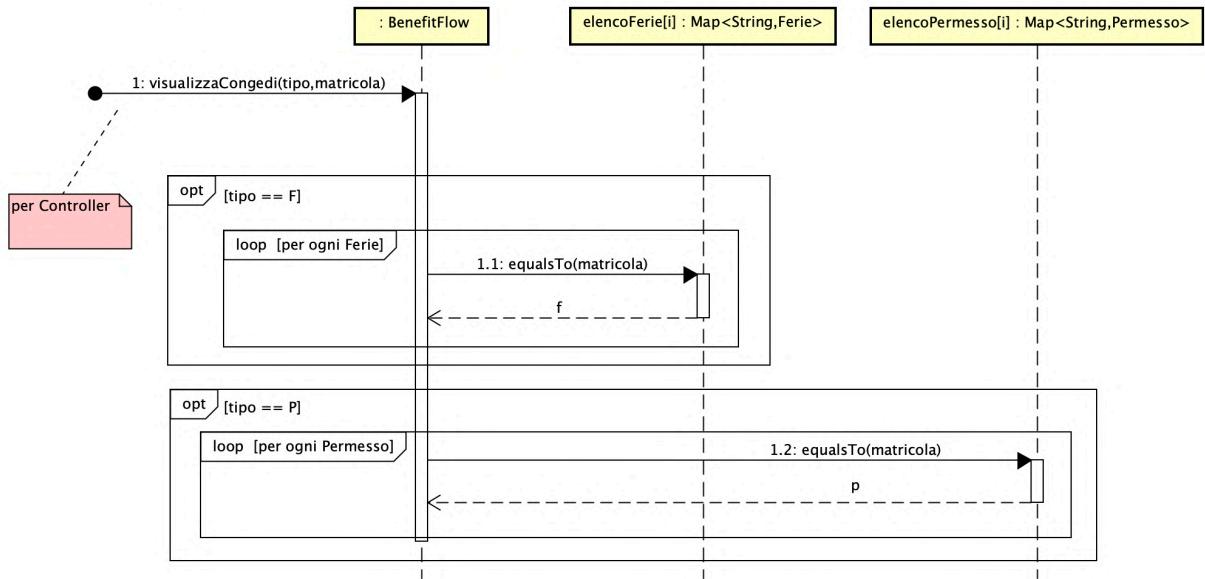
UC6: Visualizza congedi

SD visualizzaCongediComplessivi



UC7: Visualizza congedi personali

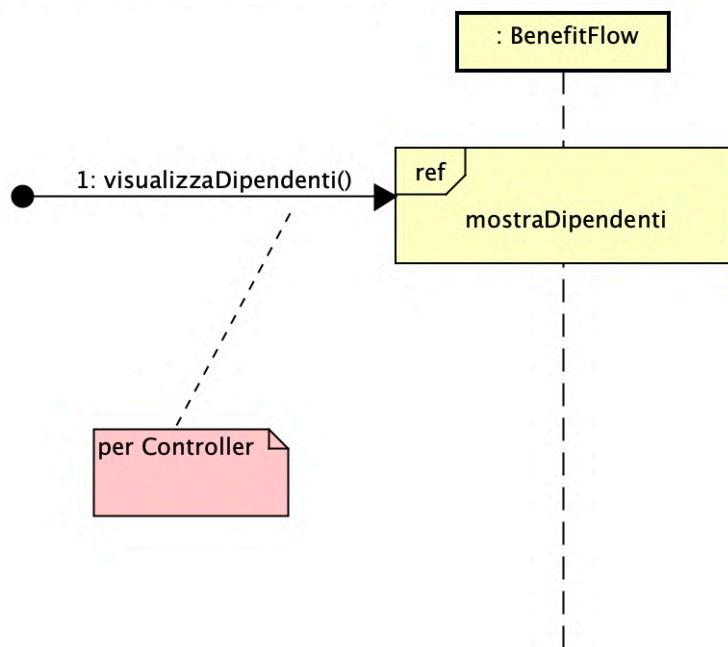
SD visualizzaCongedi



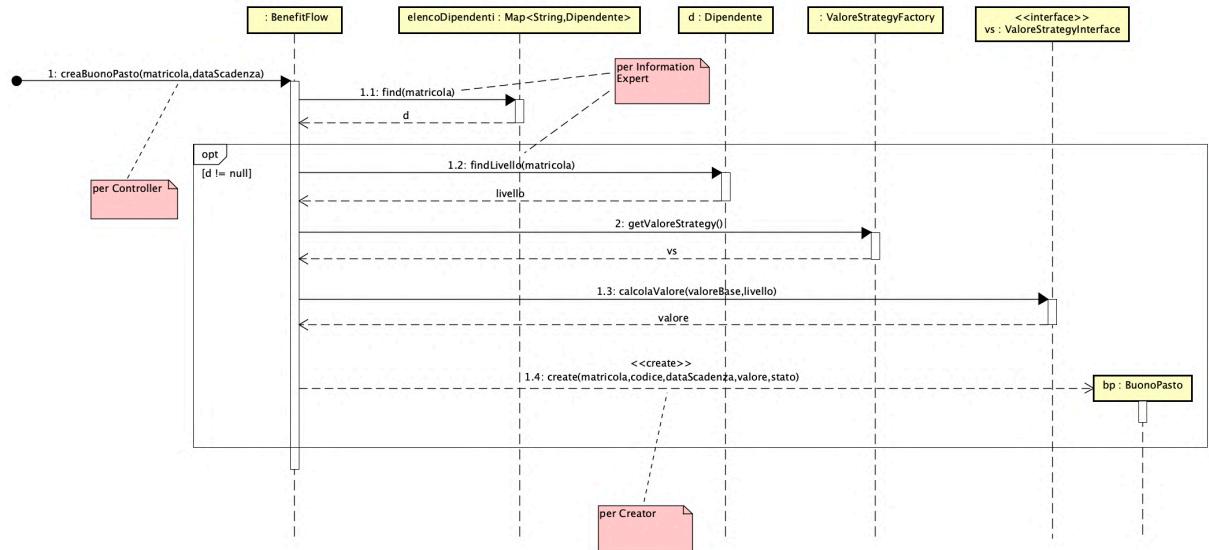
• Iterazione 3

UC8: Assegna buoni pasto

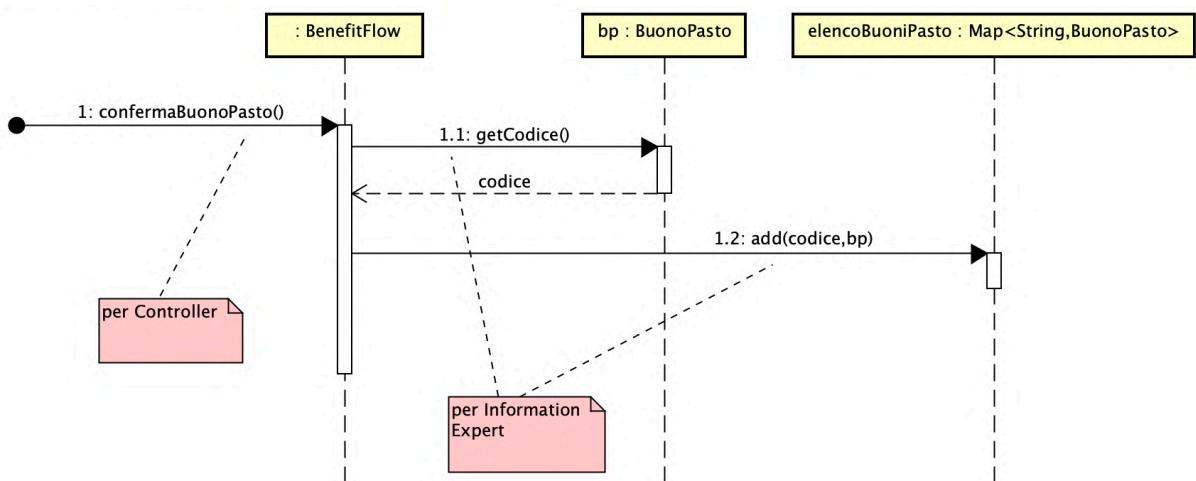
SD visualizzaDipendenti



SD creaBuonoPasto

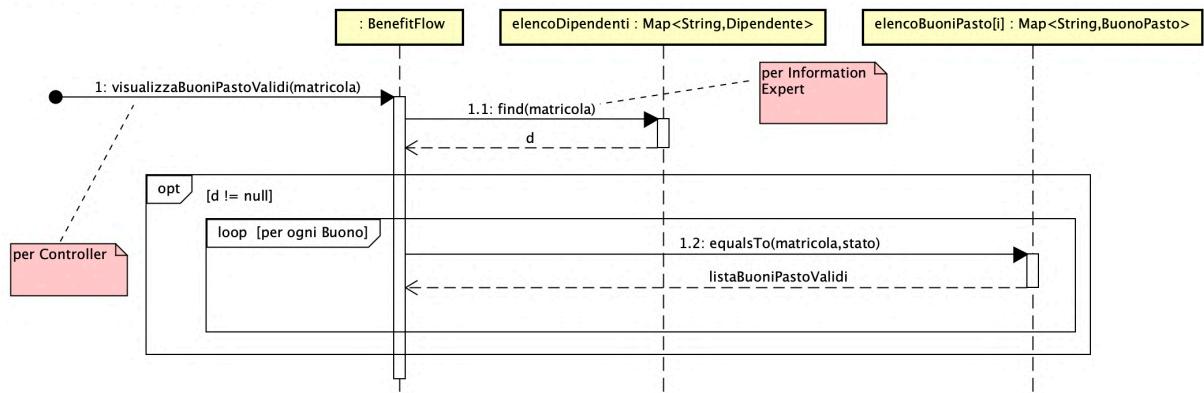


SD confermaBuonoPasto

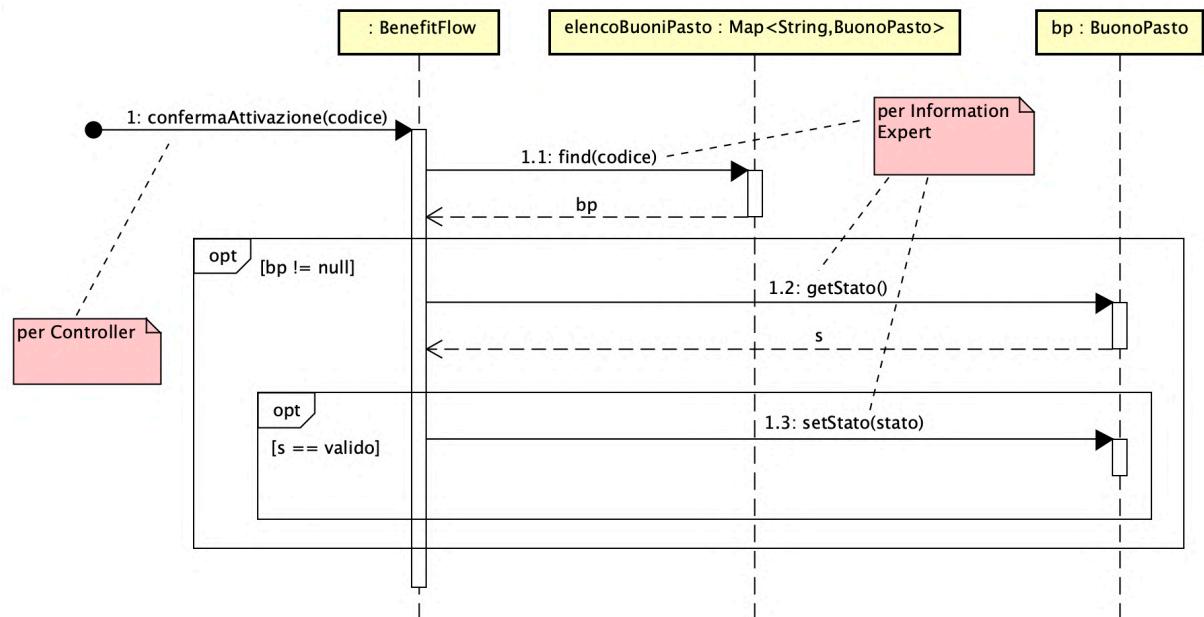


UC9: Gestisci buono pasto

SD visualizzaBuoniPastoValidi

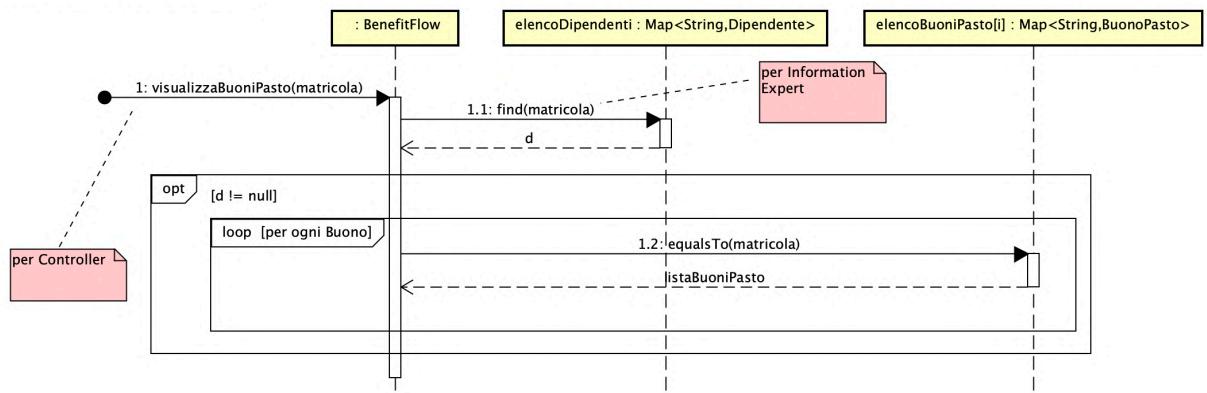


SD confermaAttivazione



UC10: Visualizza buoni pasto personali

SD visualizzaBuoniPasto



4. Testing

4.1 Introduzione

Il processo di sviluppo di un programma robusto ed efficiente dipende in gran parte dalla fase di testing, la quale aumenta la probabilità di individuare eventuali difetti. Un testing ben progettato può individuare comportamenti anomali, assicurando che il software soddisfi le esigenze del cliente.

I test possono essere suddivisi principalmente in due categorie:

- **Test unitari:** questi test valutano la correttezza del codice a livello di singole componenti, spesso verificando il comportamento di ogni metodo rispetto ai valori attesi. JUnit è uno dei framework più utilizzati per l'automazione di questi test, specialmente nel contesto di programmazione Java.
- **Test funzionali:** Questi test esaminano il corretto funzionamento del sistema software nel suo complesso, trattandolo come una "scatola nera" e verificando l'output in base agli input forniti.

Nel caso specifico dell'applicazione in questione, si è deciso di concentrarsi principalmente sui test unitari utilizzando JUnit.

4.2 Individuazione dei casi di test e testing unitario

Durante la fase preliminare del processo di testing, si è condotta un'analisi e una revisione del codice al fine di selezionare le classi e i metodi da sottoporre ai test. Il criterio utilizzato per individuare tali metodi consiste nel dare maggiore importanza a quelli che sono responsabili dell'introduzione, dell'aggiornamento e della modifica di nuove istanze. In particolare:

Classe BenefitFlow

`testInserisciNuovoDipendente():`

- Scopo: Verificare che il metodo crei correttamente un nuovo dipendente nel sistema.
- Realizzazione: Creiamo tre dipendenti, il primo con codice ruolo presente nella lista dei ruoli, il secondo con un codice non presente e il terzo senza codice. Viene verificata la corretta creazione del dipendente nei tre distinti casi.

testGeneratoreMatricola():

- Scopo: Verificare che il sistema generi correttamente la matricola.
- Realizzazione: Dati un nome e un cognome verifichiamo che la matricola generata sia quella aspettata.

testConfermaInserimento():

- Scopo: Verificare il corretto inserimento di un nuovo dipendente nella lista dipendenti
- Realizzazione: Creato e inserito un nuovo dipendente verifichiamo che la dimensione della lista sia pari a quella attesa.

testGestisciRuolo():

- Scopo: Verificare la corretta assegnazione di un ruolo ad un dipendente
- Realizzazione: Creato un nuovo dipendente e inserito nella relativa lista, si verifica il corretto aggiornamento del ruolo assegnato attraverso il codice ruolo.

testLoadRuolo():

- Scopo: Verificare il corretto inserimento dei ruoli nella relativa lista.
- Realizzazione: Creati i medesimi ruoli presenti nel metodo loadRuoli e inseriti all'interno di una lista, verifichiamo che le lunghezze delle liste siano uguali.

testInserisciFerie():

- Scopo: Verificare la corretta creazione di una ferie.
- Realizzazione: Creato un nuovo dipendente e inserito nella relativa lista, si verifica la corretta creazione della ferie.

testConfermaInserimentoFerie():

- Scopo: Verificare il corretto inserimento di una ferie.
- Realizzazione: Creato e inserito un nuovo dipendente, creata e inserita una ferie verifichiamo che la dimensione della lista sia pari a quella attesa.

testSovrapposizioneCongedi():

- Scopo: Verificare che due o più congedi siano sovrapposti in termini di data e ora.
- Realizzazione: Creati e inseriti due nuovi dipendenti, create e inserite più ferie sovrapposte, verifichiamo che la dimensione della lista dei congedi sovrapposti sia pari a quella attesa.

testCreazioneBuonoPasto():

- Scopo: Verificare la corretta creazione di un buono pasto.
- Realizzazione: Creato e inserito un nuovo dipendente, verifichiamo la corretta creazione di un buono pasto e il suo stato.

testConfermaBuonoPasto():

- Scopo: Verificare il corretto inserimento di un buono pasto.
- Realizzazione: Creato e inserito un nuovo dipendente, creato e inserito un buono pasto, verifichiamo che la dimensione della lista sia pari a quella attesa.

testConfermaAttivazione():

- Scopo: Verificare la corretta attivazione di un buono pasto.
- Realizzazione: Creato e inserito un nuovo dipendente e creato, inserito e attivato un buono pasto, verifichiamo che lo stato sia quello atteso.

testControlloScadenzaBP():

- Scopo: Verificare il corretto funzionamento del metodo controlloScadenzaBP
- Realizzazione: Creato e inserito un nuovo dipendente, creato e inserito un buono pasto con data precedente a quella corrente, verifichiamo che lo stato sia quello atteso.

Classe ValoreStrategy1

testCalcolaValore():

- Scopo: Verificare la corretta assegnazione del valore monetario di un buono pasto in base al livello del dipendente.
- Realizzazione: Verifichiamo che i valori dei buoni pasto siano quelli attesi.