UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II

CORSO DI LAUREA IN INGEGNERIA INFORMATICA
CORSO DI INGEGNERIA DEL SOFTWARE
PROF. R. PIETRANTUONO- A.A. 2023 – 24

Progetto

Vendita prodotti per il negozio di detersivi "Detergents Shopping"

Studenti del team ROMOMO:

Giuseppe Rocco	N46005684	giuseppe.rocco6@studenti.unina.it
Federica Mosca	N46005491	fed.mosca@studenti.unina.it
Vittorio Monfrecola	N46005942	vi.monfrecola@studenti.unina.it

Versione 1.5 del *4/06/2024*

A questo elaborato saranno allegati il codice sorgente e tutti i diagrammi in formato .PNG

Indice

1. Specifiche informali

Sistema di vendita (online) di prodotti del negozio “Detergents Shopping”

Si vuole realizzare una applicazione per la gestione del negozio online “Detergents Shopping” per la vendita di detersivi e altri prodotti confezionati per la cura della casa e della persona.

Il sistema deve consentire ad un impiegato di aggiungere, modificare o rimuovere i prodotti. Ogni prodotto ha un codice, un nome, una descrizione, un prezzo e la quantità disponibile (nr di pezzi del prodotto disponibili).

Per effettuare la spesa online, i clienti devono preventivamente registrarsi fornendo nome utente, password, nr telefono mobile e carta di credito. Il cliente visualizza i prodotti e può aggiungerli/rimuoverli al/dal proprio carrello della spesa, al termine della quale effettua l’acquisto – eventualmente richiedendo l’applicazione di uno sconto (vedere oltre) - procedendo al pagamento e richiedendo la consegna a domicilio. Al completamento del pagamento, il sistema: a) invia un messaggio sul cellulare del cliente, elencando gli articoli ordinati e il totale della spesa; b) invia una notifica al fattorino, che si occupa di consegnare la spesa al cliente; c) aggiorna la quantità disponibile dei singoli prodotti ordinati. La spesa è caratterizzata da un ID, una data, dal costo totale, da uno stato (IN_CORSO, ORDINATA, CONSEGNATA) e dai prodotti acquistati con le rispettive quantità. Il fattorino registra nel sistema l’avvenuta consegna inserendo l’ID della spesa.

I clienti che hanno effettuato più di N spese divengono clienti abituali e possono usufruire di sconti speciali. L’impiegato può inserire un nuovo sconto per i clienti abituali, che viene comunicato ai clienti con un messaggio sul loro telefono mobile. Gli sconti sono caratterizzati da un codice, una percentuale, una data di scadenza, e possono essere applicati una sola volta per una spesa.

Per finalità di marketing l’impiegato può richiedere al sistema di generare un report di tutti clienti che hanno effettuato almeno N spese e del numero di spese effettuate da ciascuno, con l’importo complessivo speso da ciascun cliente.

2. Analisi e specifica dei requisiti

2.1 Analisi nomi-verbi

Si vuole realizzare una applicazione per la gestione del negozio online “Detergents Shopping” per la vendita di detersivi e altri prodotti confezionati per la cura della casa e della persona.

Il sistema deve consentire ad un impiegato di aggiungere, modificare o rimuovere i prodotti. Ogni prodotto ha un codice, un nome, una descrizione, un prezzo e la quantità disponibile (nr di pezzi del prodotto disponibili).

Per effettuare la spesa online, i clienti devono preventivamente registrarsi fornendo nome utente, password, nr telefono mobile e carta di credito. Il cliente visualizza i prodotti e può aggiungerli/rimuoverli al/dal proprio carrello della spesa, al termine della quale effettua l’acquisto – eventualmente richiedendo l’applicazione di uno sconto (vedere oltre) - procedendo al pagamento e richiedendo la consegna a domicilio. Al completamento del pagamento, il sistema: a) invia un messaggio sul cellulare del cliente, elencando gli articoli ordinati e il totale della spesa; b) invia una notifica al fattorino, che si occupa di consegnare la spesa al cliente; c) aggiorna la quantità disponibile dei singoli prodotti ordinati. La spesa è caratterizzata da un ID, una data, dal costo totale, da uno stato (IN_CORSO, ORDINATA, CONSEGNATA) e dai prodotti acquistati con le rispettive quantità. Il fattorino registra nel sistema l’avvenuta consegna inserendo l’ID della spesa.

I clienti che hanno effettuato più di N spese divengono clienti abituali e possono usufruire di sconti speciali. L’impiegato può inserire un nuovo sconto per i clienti abituali, che viene comunicato ai clienti con un messaggio sul loro telefono mobile. Gli sconti sono caratterizzati da un codice, una percentuale, una data di scadenza, e possono essere applicati una sola volta per una spesa.

Per finalità di marketing l’impiegato può richiedere al sistema di generare un report di tutti clienti che hanno effettuato almeno N spese e del numero di spese effettuate da ciascuno, con l’importo complessivo speso da ciascun cliente.

LEGENDA
CLASSE FUNZIONALITA’
ATTRIBUTO ATTORE
CLASSE-ATTORE

2.2 Revisione dei requisiti

1. Il sistema deve offrire agli impiegati una funzionalità di aggiunta dei prodotti.
2. Il sistema deve offrire agli impiegati una funzionalità di modifica dei prodotti.
3. Il sistema deve offrire agli impiegati una funzionalità di rimozione dei prodotti.
4. Di ogni prodotto si vuole memorizzare il codice, il nome, la descrizione, il prezzo e un attributo che ne rappresenta la quantità disponibile (numero di pezzi disponibili).
5. Il sistema deve offrire ai clienti una funzionalità di registrazione.
6. Di ogni cliente registrato si vuole memorizzare nome utente, passoword, numero di cellulare, dati della carta di credito.
7. Il sistema deve offrire ad ogni cliente una funzionalità di carrello personale per la spesa.
8. Il sistema deve offrire al cliente una funzionalità per visualizzare tutti i prodotti.
9. Il sistema deve offrire al cliente una funzionalità per aggiungere prodotti al suo carrello.
10. Il sistema deve offrire al cliente una funzionalità per rimuovere prodotti dal suo carrello.
11. Il sistema deve offrire al cliente una funzionalità per finalizzare l'acquisto dei prodotti nel suo carrello.
12. Il sistema deve offrire al cliente una funzionalità per richiedere uno sconto sul totale dell'acquisto al momento della formalizzazione della transazione.
13. Il sistema deve offrire al cliente una funzionalità per richiedere la consegna a domicilio dell'ordine.
14. Il sistema deve offrire al cliente una funzionalità per inviare un messaggio di conferma dell'acquisto sul cellulare del cliente.
15. Il messaggio di conferma deve contenere la lista degli articoli acquistati e l'importo totale dell'ordine.
16. Il sistema deve offrire una funzionalità per l'invio automatico di notifiche al fattorino che si occupa di consegnare l'ordine.
17. Il sistema deve offrire una funzionalità per l'aggiornamento automatico delle quantità dei prodotti (numero di prodotti disponibili) dopo un acquisto.
18. Di ogni spesa si vuole memorizzare un identificativo (ID), una data (data di acquisto), costo totale (somma dei prezzi dei singoli articoli), uno stato (IN CORSO, CONSEGNATA, ORDINATA) e dagli articoli acquistati con le rispettive quantità.
19. Il sistema deve offrire al fattorino una funzionalità di registrazione dell'avvenuta consegna di un ordine.
20. Per registrare l'avvenuta consegna, il fattorino deve specificare l'identificativo della spesa.
21. Il sistema deve assegnare lo status di "cliente abituale" ad ogni cliente che effettua N spese.
22. Il sistema deve offrire ai clienti abituali degli sconti speciali da applicare sul totale dell'acquisto.
23. Il sistema deve offrire agli impiegati una funzionalità per inserire nuovi sconti speciali.
24. Il sistema deve offrire una funzionalità per l'invio automatico di notifiche al cellurare dei clienti per ogni nuovo sconto speciale.
25. Di ogni sconto si vuole memorizzare un codice identificativo, un valore percentuale, una data (data di scadenza).
26. Ogni sconto deve poter essere applicato una sola volta per una spesa.
27. Il sistema deve offrire agli impiegati una funzionalità per generare un report di tutti i clienti che hanno effettuato almeno N spese e del numero di spese effettuate da ciasciuno, con l'importo complessivo speso da ciascun cliente.

2.3 Glossario dei termini

Termine	Descrizione	Sinonimi
Prodotto	Singolo oggetto in vendita al negozio di detersivi	Articolo
Spesa	Acquisto effettuato dal cliente, comprende uno o più prodotti	Acquisto, Ordine
Carrello	Un contenitore digitale privato per ogni cliente, al quale possono essere aggiunti o rimossi dei prodotti in vendita	
Sconto	Detrazione di una parte dell'importo totale della spesa	

2.4 Classificazione dei requisiti

2.4.1 Requisiti funzionali

ID	Requisito	Origine (n. frase dei requisiti revisionati)
RF01	Il sistema deve offrire agli impiegati una funzionalità di aggiunta dei prodotti.	1
RF02	Il sistema deve offrire agli impiegati una funzionalità di modifica dei prodotti.	2
RF03	Il sistema deve offrire agli impiegati una funzionalità di rimozione dei prodotti.	3
RF04	Il sistema deve offrire ai clienti una funzionalità di registrazione.	5
RF05	Il sistema deve offrire ad ogni cliente una funzionalità di carrello personale per la spesa.	7
RF06	Il sistema deve offrire al cliente una funzionalità per aggiungere prodotti al suo carrello.	9
RF07	Il sistema deve offrire al cliente una funzionalità per rimuovere prodotti dal suo carrello.	10
RF08	Il sistema deve offrire al cliente una funzionalità per finalizzare l'acquisto dei prodotti nel suo carrello.	11
RF09	Il sistema deve offrire al cliente una funzionalità per richiedere uno sconto sul totale dell'acquisto al momento della formalizzazione della transazione.	12
RF10	Il sistema deve offrire al cliente una funzionalità per richiedere la consegna a domicilio dell'ordine.	13
RF11	Il sistema deve offrire al cliente una funzionalità per inviare un messaggio di conferma dell'acquisto sul cellulare del cliente.	14
RF12	Il messaggio di conferma deve contenere la lista degli articoli acquistati e l'importo totale dell'ordine.	15
RF13	Il sistema deve offrire una funzionalità per l'invio automatico di notifiche al fattorino che si occupa di consegnare l'ordine.	16
RF14	Il sistema deve offrire una funzionalità per l'aggiornamento automatico delle quantità dei prodotti (numero di prodotti disponibili) dopo un acquisto.	17
RF15	Il sistema deve offrire al fattorino una funzionalità di registrazione dell'avvenuta consegna di un ordine.	19
RF16	Il sistema deve assegnare lo status di "cliente abituale" ad ogni cliente che effettua N spese.	21
RF17	Il sistema deve offrire ai clienti abituali degli sconti speciali da applicare sul totale dell'acquisto.	22
RF18	Il sistema deve offrire agli impiegati una funzionalità per inserire nuovi sconti speciali.	23
RF19	Il sistema deve offrire una funzionalità per l'invio automatico di notifiche al cellurare dei clienti per ogni nuovo sconto speciale.	24
RF20	Il sistema deve offrire agli impiegati una funzionalità per generare un report di tutti i clienti che hanno effettuato almeno N spese e del numero di spese effettuate da ciasciuno, con l'importo complessivo speso da ciascun cliente.	27

2.4.2 Requisiti sui dati

ID	Requisito	Origine (n. frase dei requisiti revisionati)
RD01	Di ogni prodotto si vuole memorizzare il codice, il nome, la descrizione, il prezzo e un attributo che ne rappresenta la quantità disponibile (numero di pezzi disponibili).	4
RD02	Di ogni cliente registrato si vuole memorizzare nome utente, passoword, numero di cellulare, dati della carta di credito.	6
RD03	Di ogni spesa si vuole memorizzare un identificativo (ID), una data (data di acquisto), costo totale (somma dei prezzi dei singoli articoli), uno stato (IN CORSO, CONSEGNATA, ORDINATA) e dagli articoli acquistati con le rispettive quantità.	18
RD04	Per registrare l'avvenuta consegna, il fattorino deve specificare l'identificativo della spesa.	20
RD05	Di ogni sconto si vuole memorizzare un codice identificativo, un valore percentuale, una data (data di scadenza)	25

2.4.3 Vincoli / Altri requisiti

ID	Requisito/Vincolo	Origine (n. frase dei requisiti revisionati)
V01	Ogni sconto deve poter essere applicato una sola volta per una spesa	25
RNF01	Per l'invio delle notifiche al cellulare dei clienti e dei fattorini, deve essere disponibile un server per l'inoltro della messaggistica su numero di cellulare.	
RNF02	Il sistema deve poter gestire una moltitudine di accessi per permettere a più clienti di effettuare acquisti.	
RNF03	Il negozio deve disporre di addetti alle consegne (fattorini) per poter permettere al sistema di espletare la sua funzione.	

2.5 Modellazione dei casi d'uso

2.5.1 Attori e casi d'uso

Attori Primari:

- Impiegato
- Cliente
- Tempo
- ServizioSMS
- Cliente registrato

Attori Secondari:

- Fattorino

Casi d'uso:

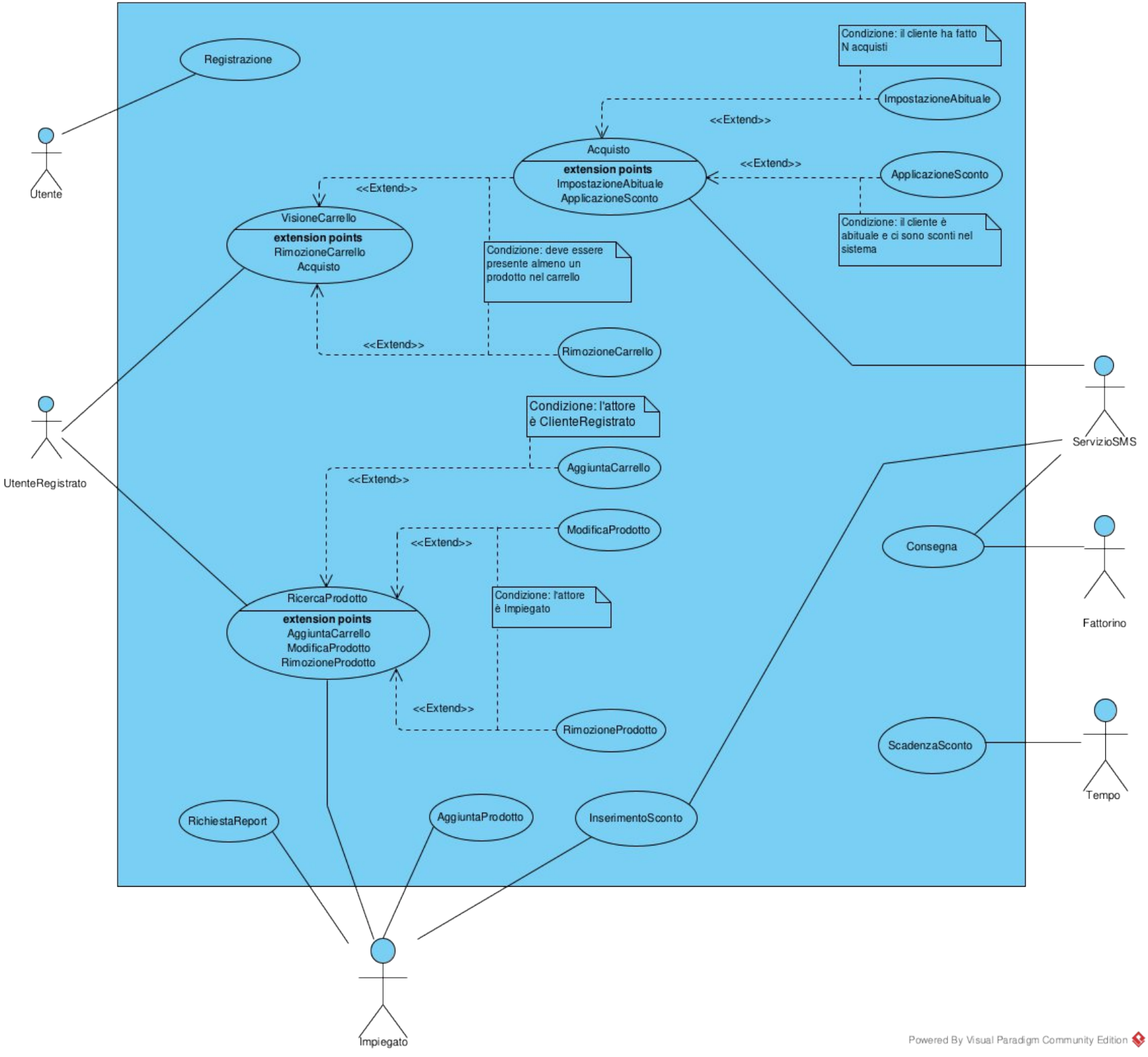
- Registrazione
- AggiuntaProdotto
- RicercaProdotto
- Consegna
- VisioneCarrello
- RichiestaReport
- InserimentoSconto
- ScadenzaSconto

Casi d' uso di estensione:

- ApplicazioneSconto
- RimozioneProdotto
- ModificaProdotto
- Acquisto
- AggiuntaCarrello
- RimozioneCarrello
- ImpostazioneAbituale

Caso d'uso	**Attori Primari**	**Attori Secondari**	**Incl. / Ext.**
Registrazione	Cliente		
AggiuntaProdotto	Impiegato		
RicercaProdotto	Impiegato, ClienteRegistrato		AggiuntaCarrello(EXT), ModificaProdotto(EXT), RimozioneProdotto(EXT)
Consegna	ServizioSMS	Fattorino	
VisioneCarrello	ClienteRegistrato		Acquisto(EXT), RimozioneCarrello(EXT)
RichiestaReport	Impiegato		
InserimentoSconto	Impiegato	ServizioSMS	
AggiuntaCarrello	ClienteRegistrato		
RimozioneCarrello	ClienteRegistrato		
RimozioneProdotto	Impiegato		
ModificaProdotto	Impiegato		
Acquisto	ClienteRegistrato	ServizioSMS	ApplicazioneSconto(EXT), ImpostazioneAbituale(EXT)
ApplicazioneSconto	ClienteRegistrato		
ImpostazioneAbituale	ClienteRegistrato		
ScadenzaSconto	Tempo		

2.5.2 Diagramma CU



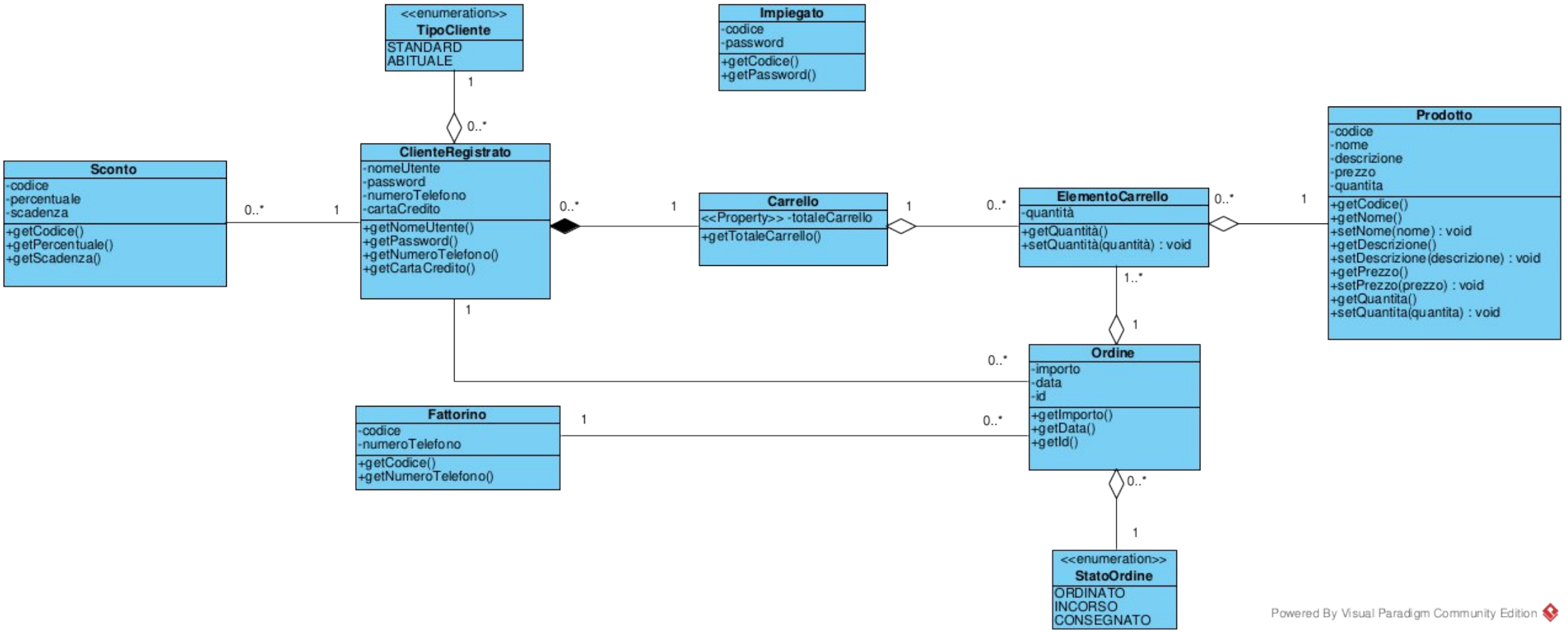
2.5.3 Scenari

Caso d'uso:	Registrazione
Attore primario	Cliente
Attore secondario	*nessuno*
Descrizione	Il Cliente si registra nel sistema per poter usufruire dei servizi
Pre-Condizioni	*nessuno*
Sequenza di eventi principale	1. Il caso d'uso inizia quando il cliente clicca sul pulsante "Registrazione". 2. Il Sistema prepara un form da compilare con NOME UTENTE, PASSWORD, NR TELEFONO, CARTA DI CREDITO. 3. Il Sistema presenta il form al Cliente. 4. Il Cliente inserisce NOME UTENTE, PASSWORD, NR TELEFONO, CARTA DI CREDITO. 5. Il Cliente clicca sul pulsate "Conferma registrazione". 6. Il Sistema verifica che le informazioni inserite dal Cliente non siano già presenti nella base dati. 7. Il Sistema salva le informazioni del cliente. 8. Il Sistema presenta una conferma di avvenuta registrazione.
Post-Condizioni	I Clienti diventano Clienti registrati e possono usufruire dei servizi.
Casi d'uso correlati	*nessuno*
Sequenza di eventi alternativi	7.a. Se NOME UTENTE o NR TELEFONO del Cliente sono già presenti nel sistema, viene presentato un messaggio di errore.

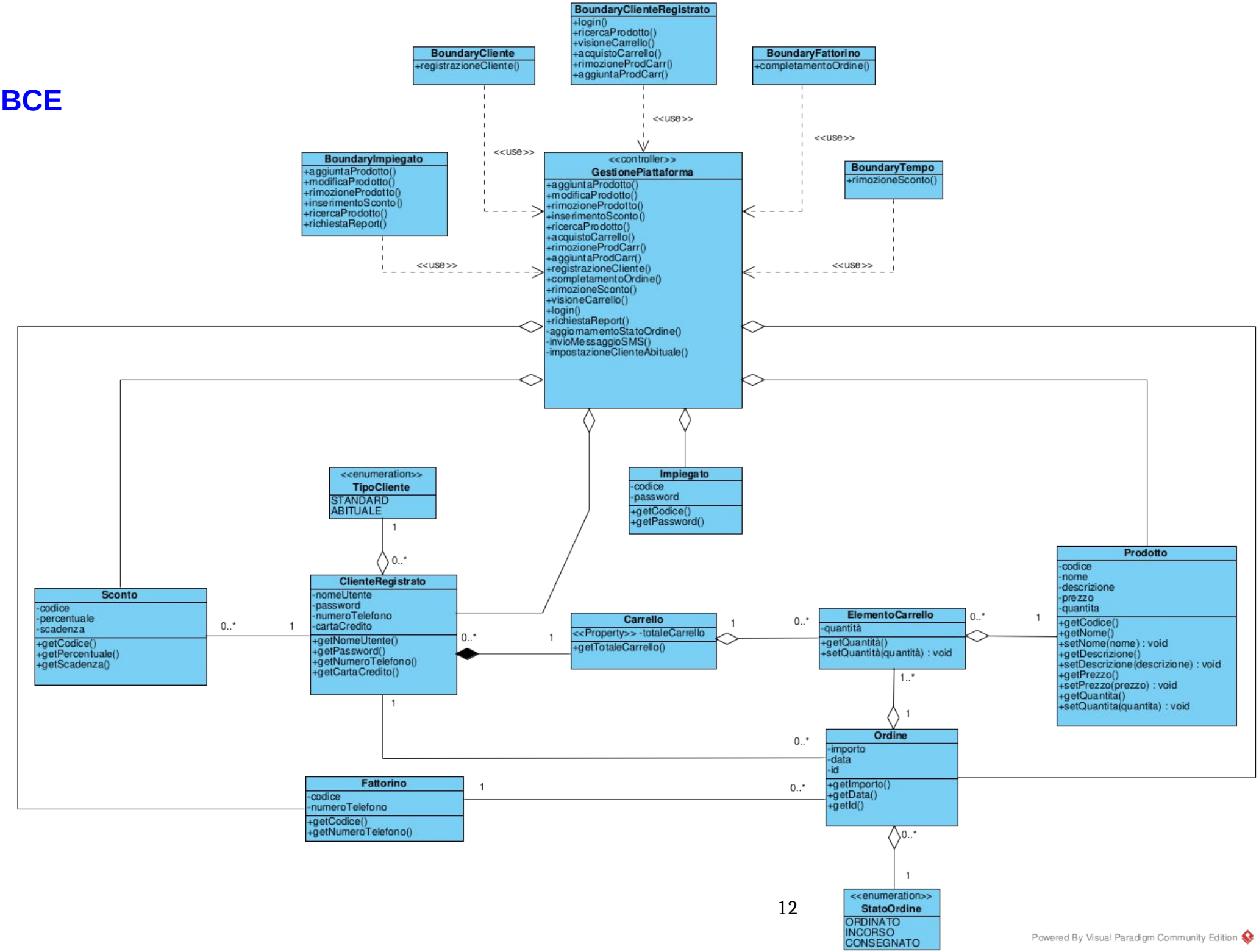
Caso d'uso:	AggiuntaProdotto
Attore primario	Impiegato
Attore secondario	*nessuno*
Descrizione	L'impiegato aggiunge un prodotto al catalogo del negozio
Pre-Condizioni	*nessuno*
Sequenza di eventi principale	1. Il caso d'uso inizia quando l'impiegato clicca sul pulsante "Aggiungi Prodotto". 2. Il Sistema prepara un form da compilare con CODICE, NOME, DESCRIZIONE, PREZZO, QUANTITA' DISPONIBILE. 3. Il Sistema presenta il form all'Impiegato. 4. L'Impiegato inserisce CODICE, NOME, DESCRIZIONE, PREZZO, QUANTITA' DISPONIBILE. 5. L'Impiegato clicca sul pulsante "Conferma". 6. Il Sistema verifica che le informazioni inserite dall'Impiegato non siano già presenti nella base dati. 7. Il Sistema salva le informazioni del prodotto. 8. Il Sistema presenta una conferma di avvenuta aggiunta.
Post-Condizioni	Un prodotto viene aggiunto al catalogo
Casi d'uso correlati	*nessuno*
Sequenza di eventi alternativi	6.a Se QUANTITA' DISPONIBILE e' minore di zero, viene presentato un messaggio di errore 7.a. Se CODICE è già presente nel sistema, viene presentato un messaggio di errore.

Caso d'uso:	RichiestaReport
Attore primario	Impiegato
Attore secondario	*nessuno*
Descrizione	L'impiegato richiede un resoconto sui clienti che hanno effettuato N spese
Pre-Condizioni	*nessuno*
Sequenza di eventi principale	1. Il caso d'uso inizia quando l'impiegato clicca sul pulsante "Genera resoconto". 2. Il Sistema chiede all'Impiegato il numero di spese "Soglia" per la generazione del resoconto 3. L'Impiegato inserisce il valore richiesto (N). 4. L'impiegato clicca sul pulsante "Conferma". 5. Il Sistema recupera dalla base dati tutti i ClientiRegistrati che hanno effettuato più di N spese 6. Il Sistema recupera il numero di spese totali effettuate dai ClientiRegistrati recuperati in precedenza 7. Il Sistema determina il totale speso da ciascuno dei ClientiRegistrati recuperati in precedenza. 8. Il Sistema prepara una tabella per la presentazione ordinata di tutti i ClientiRegistrati con associato N di spese e TOT speso. 9. Il Sistema presenta all'Impiegato tutti i dati.
Post-Condizioni	L'Impiegato riceve un resoconto con tutti i ClientiRegistrati che hanno effettuato almeno N spese, con il numero totale di spese effettuate da questi ultimi e l'importo complessivo speso da ciascuno
Casi d'uso correlati	*nessuno*
Sequenza di eventi alternativi	5.a. Se N è minore o uguale a zero, viene presentato un messaggio di errore. 5.b Se non è presente alcun ClienteRegistrato che superi la soglia di N spese, viene mostrato un messaggio di avviso.

2.6 Diagramma delle classi (analisi)

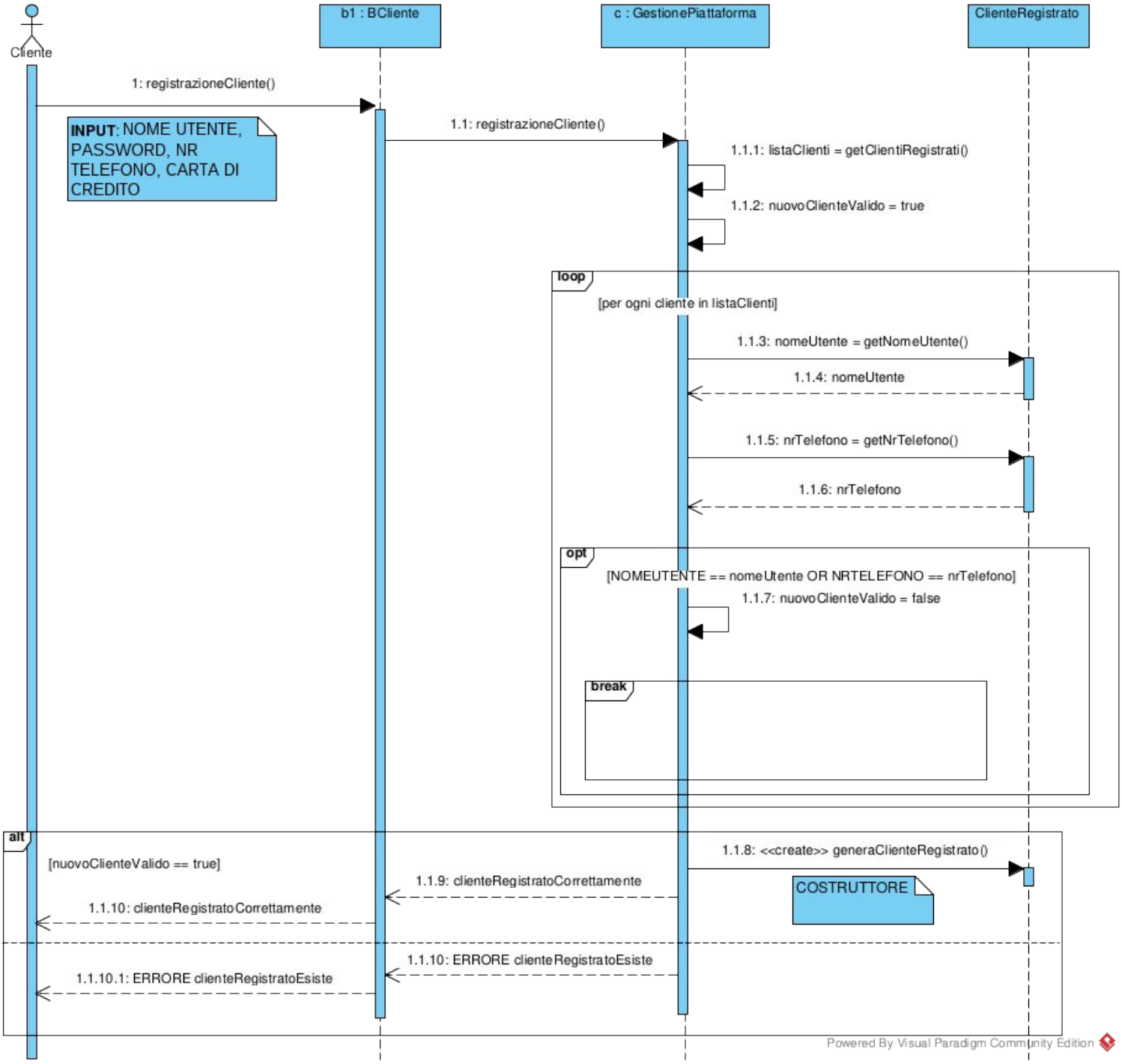


BCE

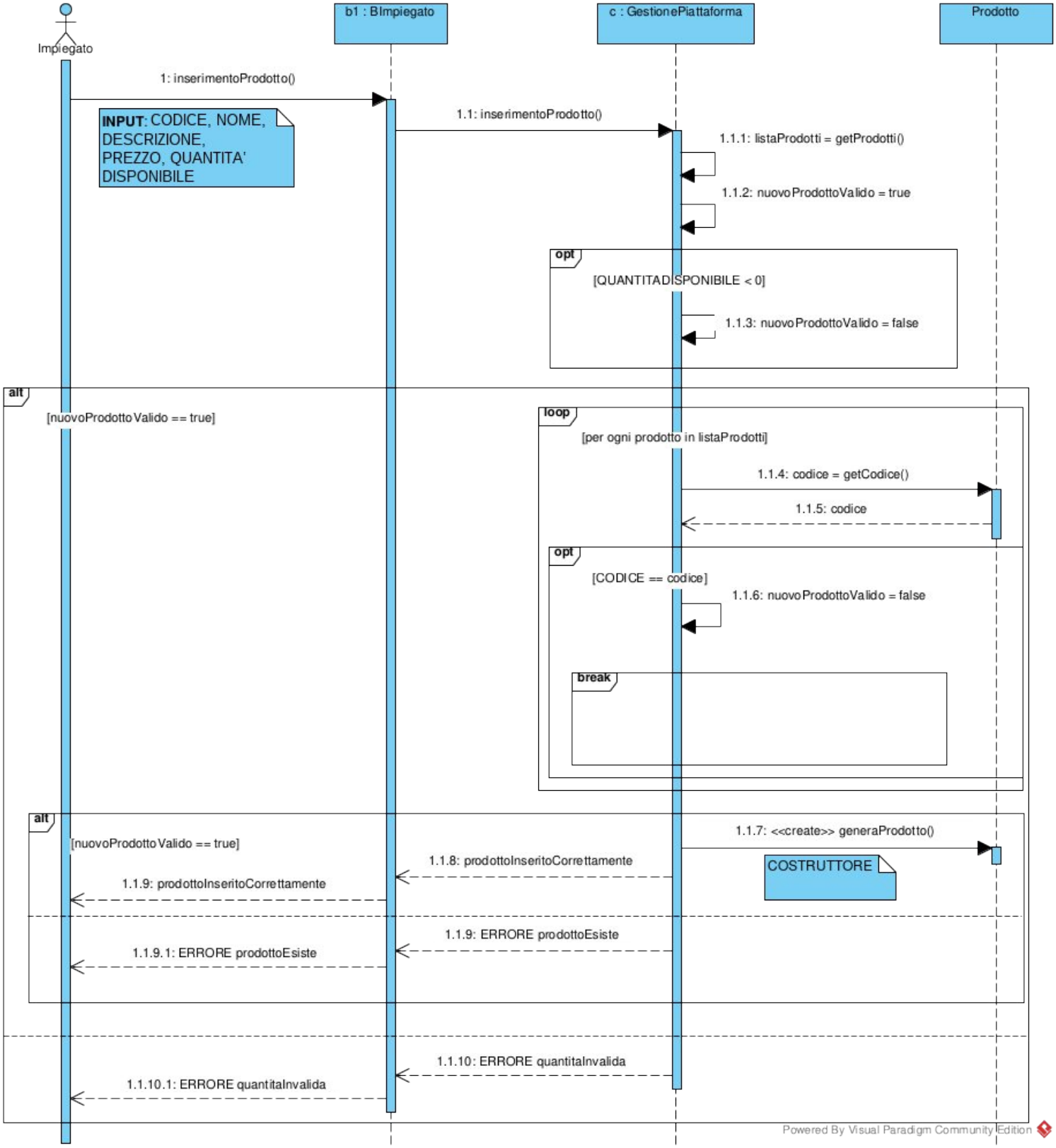


2.7 Diagrammi di sequenza

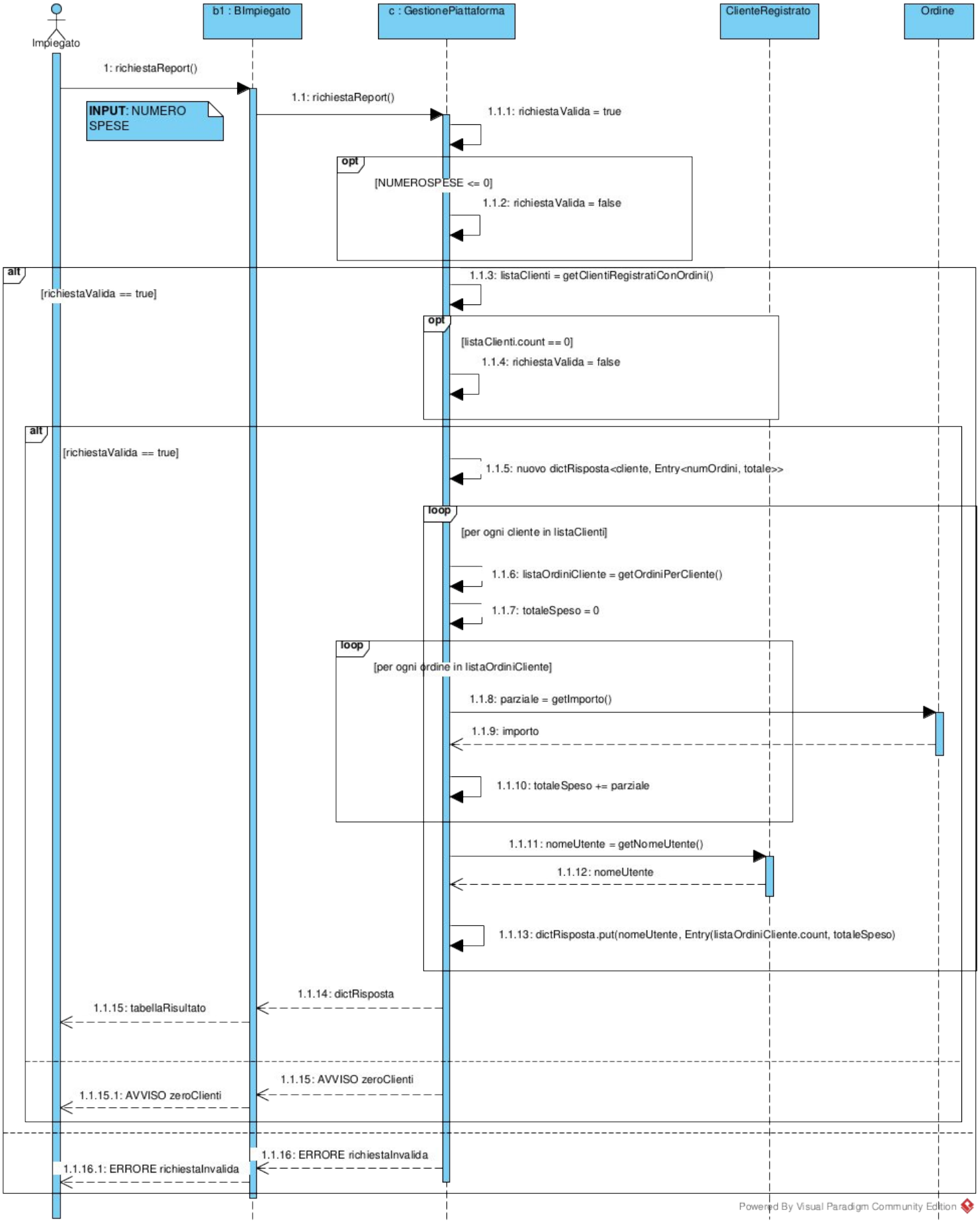
2.7.1 Registrazione



2.7.2 AggiuntaProdotto



2.7.3 RichiestaReport



3. Stima dei costi

- Tabella di riferimento per le complessità di dati e transazioni

	SEMPLICE	MEDIO	COMPLESSO
NILF	**3**	**4**	**6**
NEIF	**4**	**5**	**7**
NEI	**3**	**4**	**6**
NEO	**7**	**10**	**15**
NEQ	**5**	**7**	**10**

- Tabella elenco dei fattori correttivi (il cui valore è compreso tra 0 e 5)

FATTORI CORRETTIVI	
COMUNICAZIONE DATI	1
DISTRIBUZIONE ELABORAZIONE	0
PRESTAZIONI	3
UTILIZZO INTENSIVO CONFIGURAZIONE	2
FREQUENZA DELLE TRANSAZIONI	3
INSERIMENTO DATI INTERATTIVO	3
EFFICIENZA PER L'UTENTE FINALE	4
AGGIORNAMENTO INTERATTIVO	3
COMPLESSITA' ELABORATIVA	0
RIUSABILITA'	2
FACILITA' INSTALLAZIONE	1
FACILITA' GESTIONE OPERATIVA	0
MOLTEPLICITA' DI SITI	3
FACILITA' DI MODIFICA	2
	27

COMPLESSITÀ ILF e EIF

	VALORE	SEMPLICE	MEDIO	COMPLESSO	TOT
NILF	**1**			**6**	**6**
NEIF	**0**				**0**

CU: Registrazione

	VALORE	SEMPLICE	MEDIO	COMPLESSO	TOT
NEI	**1**	**3**			**3**
NEO	**1**	**7**			**7**
NEQ	**0**				**0**

CU: AggiuntaProdotto

	VALORE	SEMPLICE	MEDIO	COMPLESSO	TOT
NEI	**1**	**3**			**3**
NEO	**1**	**7**			**7**
NEQ	**0**				**0**

CU: RichiestaReport

	VALORE	SEMPLICE	MEDIO	COMPLESSO	TOT
NEI	**1**	**3**			**3**
NEO	**1**			**15**	**15**
NEQ	**1**			**10**	**10**

UFP = 44

AFP = 0,92

FP = 40,48 ~= 40

LLOC/FP = 53 (JAVA)

LLOC/FP = 53 (JAVA)

LLOC = 2.332

LLOC = 2.120

4. Piano di test funzionale

PIANO DI TEST CON *CATEGORY-PARTITION TESTING* PER LA FUNZIONALITÀ “REGISTRAZIONE”.

NOMEUTENTE	NRTELEFONO	CARTACREDITO	PASSWORD	ELEMENTO DI SISTEMA DATABASE
• Stringa di lunghezza <= 255 caratteri • Stringa che contiene simboli [ERROR] • Stringa di lunghezza > 255 [ERROR]	• Stringa di numeri interi di lunghezza = 10 • Stringa che contiene caratteri o simboli [ERROR] • Stringa di numeri interi di lunghezza != 10 [ERROR]	• Stringa di numeri interi di lunghezza = 16 • Stringa che contiene caratteri o simboli [ERROR] • Stringa di numeri interi di lunghezza != 16 [ERROR]	• Stringa di lunghezza <= 255 caratteri • Stringa di lunghezza > 255 [ERROR]	• Utente non presente all’interno del database • Utente già presente all’interno del database [ERROR]

Il numero di test da effettuarsi senza particolari vincoli è: 3 * 3 * 3 * 2 * 2 = 108

Introducendo i vincoli [ERROR].

Il numero di Test da eseguire per testare singolarmente i vincoli è 8 (2 per NOMEUTENTE, 2 per NRTELEFONO, 2 per CARTACREDITO, 1 per PASSWORD, 1 per DATABASE).

Il numero di test risultante è: (1*1*1*1*1) + 8 = 9

TEST SUITE

ID	Descrizione	Classi di equivalenza coperte	Pre-condizioni	Input	Output Attesi	Post-condizioni Attese
1	Tutti gli input validi	Nomeutente valido, Nrtelefono valido, Cartacredito valido, Password valido, Elemento database valido	Il database è inizializzato correttamente e esiste nessun'altro utente con questa email o cellulare	{nomeUtente: paolobrosio22, nrTelefono: 3553925732, cartaCredito: 4444444444444444, password: hulk#buster27}	Registrazione avvenuta correttamente	Il cliente è correttamente registrato all'interno della base dati.
2	Stringa NOMEUTENTE contiene simboli	Nomeutente con simboli [ERROR], Nrtelefono valido, Cartacredito valido, Password valido, Elemento database valido	Il sistema richiede al cliente di inserire un nome utente al momento della registrazione	{nomeUtente: paolo@@brosio22, nrTelefono: 3553925732, cartaCredito: 4444444444444444, password: hulk#buster27}	ERRORE: Il nome utente inserito non è valido, riprovare.	Il sistema chiede di reinserire il nome utente.
3	Stringa NOMEUTENTE > 255	Nomeutente > 255 [ERROR], Nrtelefono valido, Cartacredito valido, Password valido, Elemento database valido	Il sistema richiede al cliente di inserire un nome utente al momento della registrazione	{nomeUtente: forzanapoliiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiii ... iiiii, nrTelefono: 3553925732, cartaCredito: 4444444444444444, password: hulk#buster27}	ERRORE: Il nome utente inserito non è valido, riprovare.	Il sistema chiede di reinserire il nome utente.
4	Stringa NRTELEFONO contiene caratteri o	Nomeutente valido, Nrtelefono con caratteri o simboli [ERROR],	Il sistema richiede al cliente di inserire un numero di telefono al	{nomeUtente: paolobrosio22, nrTelefono:	ERRORE: Il numero di telefono inserito non è valido, riprovare.	Il sistema chiede di reinserire il numero di telefono

	simboli	Cartacredito valido, Password valido, Elemento database valido	momento della registrazione	5ASDA2#32@, cartaCredito: 4444444444444444, password: hulk#buster27}		
5	Stringa NRTELEFONO con lunghezza != 10	Nomeutente valido, Nrtelefono con lunghezza != 10 [ERROR], Cartacredito valido, Password valido, Elemento database valido	Il sistema richiede al cliente di inserire un numero di telefono al momento della registrazione	{nomeUtente: paolobrosio22, nrTelefono: 3553925732333, cartaCredito: 4444444444444444, password: hulk#buster27}	ERRORE: Il numero di telefono inserito non è valido, riprovare.	Il sistema chiede di reinserire il numero di telefono
6	Stringa CARTACREDITO contiene caratteri o simboli	Nomeutente valido, Nrtelefono valido, Cartacredito con caratteri o simboli [ERROR], Password valido, Elemento database valido	Il sistema richiede al cliente di inserire un numero di carta di credito al momento della registrazione	{nomeUtente: paolobrosio22, nrTelefono: 3553925732, cartaCredito: 4S444#4444G44444, password: hulk#buster27}	ERRORE: Il numero di carta inserito non è valido, riprovare.	Il sistema chiede di reinserire il numero di carta di credito
7	Stringa CARTACREDITO con lunghezza != 16	Nomeutente valido, Nrtelefono valido, Cartacredito con lunghezza != 16 [ERROR], Password valido, Elemento database valido	Il sistema richiede al cliente di inserire un numero di carta di credito al momento della registrazione	{nomeUtente: paolobrosio22, nrTelefono: 3553925732, cartaCredito: 777, password: hulk#buster27}	ERRORE: Il numero di carta inserito non è valido, riprovare.	Il sistema chiede di reinserire il numero di carta di credito
8	Stringa PASSWORD > 255	Nomeutente valido, Nrtelefono valido, Cartacredito valido, Password con lunghezza > 255	Il sistema richiede al cliente di inserire una password al momento della	{nomeUtente: paolobrosio22, nrTelefono: 3553925732,	ERRORE: La passowrd inserita non è valida, riprovare.	Il sistema chiede di reinserire la password

		[ERROR], Elemento database valido	registrazione	cartaCredito: 4444444444444444, password: hulk#busteeeeeeeeeeee ... eeeeeeeer27}		
9	Cliente già presente nel sistema	Nomeutente valido, Nrtelefono valido, Cartacredito valido, Password valido, Elemento database [ERROR]	Il database contiene già un utente registrato con lo stesso nome utente o lo stesso cellulare	{nomeUtente: paolobrosio22, nrTelefono: 3553925732, cartaCredito: 4444444444444444, password: hulk#buster27}	ERRORE: Cliente già presente nel sistema.	Il sistema restituisce un messaggio di errore e chiude la funzionalità

PIANO DI TEST CON *CATEGORY-PARTITION TESTING* PER LA FUNZIONALITÀ “AGGIUNTAPRODOTTO”.

CODICE	DESCRIZIONE	NOME	PREZZO	NDISPONIBILE	ELEMENTO DI SISTEMA DATABASE
• Stringa di lunghezza <= 255 caratteri • Stringa che contiene simboli [ERROR] • Stringa di lunghezza > 255 [ERROR]	• Stringa di lunghezza <= 255 caratteri • Stringa di lunghezza > 255 [ERROR]	• Stringa di lunghezza <= 255 caratteri • Stringa che contiene simboli [ERROR] • Stringa di lunghezza > 255 [ERROR]	• Numero floating point compreso tra 0 e 1000 • Numero floating point NON compreso tra 0 e 1000 [ERROR]	• Numero intero compreso tra 0 e 1000 • Numero intero NON compreso tra 0 e 1000 [ERROR]	• Prodotto non presente all’interno del database • Prodotto già presente all’interno del database [ERROR]

Il numero di test da effettuarsi senza particolari vincoli è: 3 * 2 * 3 * 2 * 2 * 2 = 144

Introducendo i vincoli [ERROR].

Il numero di Test da eseguire per testare singolarmente i vincoli è 8 (2 per CODICE, 1 per DESCRIZIONE, 2 per NOME, 1 per PREZZO, 1 per NDISPONIBILE, 1 per DATABASE).

Il numero di test risultante è: (1*1*1*1*1*1) + 8 = 9

TEST SUITE

ID	Descrizione	Classi di equivalenza coperte	Pre-condizioni	Input	Output Attesi	Post-condizioni Attese
1	Tutti gli input validi	Codice valido, Descrizione valido, Nome valido, Prezzo valido, Ndisponibile valido Elemento database valido	Il database è inizializzato correttamente e esiste nessun'altro prodotto con questo codice	{codice: AEERX923, descrizione: "Bagnodoccia 9 in 1 da uomo, per la cura della tua auto da uomo", nome: PuliMax, prezzo: 0.67, Ndisponibile: 90}	Inserimento avvenuto correttamente	Il prodotto è correttamente registrato all'interno della base dati.
2	Stringa CODICE contiene simboli	Codice con simboli [ERROR], Descrizione valido, Nome valido, Prezzo valido, Ndisponibile valido Elemento database valido	Il sistema richiede all'impiegato di digitare un codice al momento dell' inserimento di un prodotto	{codice: AEER@@@X92, descrizione: "Bagnodoccia 9 in 1 da uomo, per la cura della tua auto da uomo", nome: PuliMax, prezzo: 0.67, Ndisponibile: 90}	ERRORE: Il codice inserito non è valido, riprovare.	Il sistema chiede di reinserire il codice.
3	Stringa CODICE con lunghezza > 255	Codice con lunghezza > 255 [ERROR], Descrizione valido, Nome valido, Prezzo valido, Ndisponibile valido Elemento database valido	Il sistema richiede all'impiegato di digitare un codice al momento dell' inserimento di un prodotto	{codice: E EE EEEE EEEEE ... EEEEEE, descrizione: "Bagnodoccia 9 in 1 da uomo, per la cura della tua auto da uomo", nome: PuliMax, prezzo: 0.67, Ndisponibile: 90}	ERRORE: Il codice inserito non è valido, riprovare.	Il sistema chiede di reinserire il codice.

4	Stringa DESCRIZIONE con lunghezza > 255	Codice valido, Descrizione con lunghezza > 255 [ERROR], Nome valido, Prezzo valido, Ndisponibile valido Elemento database valido	Il sistema richiede all'impiegato di digitare una descrizione al momento dell' inserimento di un prodotto	{codice: AEERX923, descrizione: "Bagnodocciaaaaaaaaaaaa .. . aa", nome: PuliMax, prezzo: 0.67, Ndisponibile: 90}	ERRORE: la descrizione inserita non è valida, riprovare.	Il sistema chiede di reinserire la descrizione.
5	Stringa NOME contiene simboli	Codice valido, Descrizione valido, Nome con simboli [ERROR], Prezzo valido, Ndisponibile valido Elemento database valido	Il sistema richiede all'impiegato di digitare un nome al momento dell' inserimento di un prodotto	{codice: AEEX92, descrizione: "Bagnodoccia 9 in 1 da uomo, per la cura della tua auto da uomo", nome: Puli#@Max, prezzo: 0.67, Ndisponibile: 90}	ERRORE: Il nome inserito non è valido, riprovare.	Il sistema chiede di reinserire il nome.
6	Stringa NOME con lunghezza > 255	Codice valido, Descrizione valido, Nome con lunghezza > 255 [ERROR], Prezzo valido, Ndisponibile valido Elemento database valido	Il sistema richiede all'impiegato di digitare un nome al momento dell' inserimento di un prodotto	{codice: AEEX92, descrizione: "Bagnodoccia 9 in 1 da uomo, per la cura della tua auto da uomo", nome: PuliMaaaaaaaaaaaaaaaa .., prezzo: 0.67, Ndisponibile: 90}	ERRORE: Il nome inserito non è valido, riprovare.	Il sistema chiede di reinserire il nome.

7	Numero PREZZO NON compreso tra 0 e 1000	Codice valido, Descrizione valido, Nome valido, Prezzo non compreso tra 0 e 1000 [ERROR], Ndisponibile valido Elemento database valido	Il sistema richiede all'Impiegato di digitare un prezzo al momento dell' inserimento di un prodotto	{codice: AEERX923, descrizione: "Bagnodoccia 9 in 1 da uomo, per la cura della tua auto da uomo", nome: PuliMax, prezzo: -0.67, Ndisponibile: 90}	ERRORE: Il prezzo inserito non è valido, riprovare.	Il sistema chiede di reinserire il prezzo
8	Numero NDISPONIBILE NON compreso tra 0 e 1000	Codice valido, Descrizione valido, Nome valido, Prezzo valido, Ndisponibile non compreso tra 0 e 1000 [ERROR], Elemento database valido	Il sistema richiede all'Impiegato di digitare una quantità disponibile al momento dell' inserimento di un prodotto	{codice: AEERX923, descrizione: "Bagnodoccia 9 in 1 da uomo, per la cura della tua auto da uomo", nome: PuliMax, prezzo: 0.67, Ndisponibile: -90}	ERRORE: La quantità disponibileinserita non è valida, riprovare.	Il sistema chiede di reinserire la quantità disponibile
9	Prodotto già presente nel sistema	Codice valido, Descrizione valido, Nome valido, Prezzo valido, Ndisponibile valido Elemento database [ERROR]	Il database contiene già un prodotto registrato con lo stesso codice	{codice: AEERX923, descrizione: "Bagnodoccia 9 in 1 da uomo, per la cura della tua auto da uomo", nome: PuliMax, prezzo: 0.67, Ndisponibile: 90}	ERRORE: Prodotto già presente nel sistema.	Il sistema restituisce un messaggio di errore e chiude la funzionalità

PIANO DI TEST CON *CATEGORY-PARTITION TESTING* PER LA FUNZIONALITÀ “RICHIESTAREPORT”.

NORDINI	ELEMENTO DI SISTEMA DATABASE
• Numero intero compreso tra 0 e 1000 • Numero intero NON compreso tra 0 e 1000 [ERROR]	• Sono presenti utenti con almeno NORIDNI nel database • NON sono presenti utenti con almeno NORDINI nel database [ERROR]

Il numero di test da effettuarsi senza particolari vincoli è: 2 * 2 = 4

Introducendo i vincoli [ERROR].

Il numero di Test da eseguire per testare singolarmente i vincoli è 2 (1 per NORDINI, 1 per DATABASE).

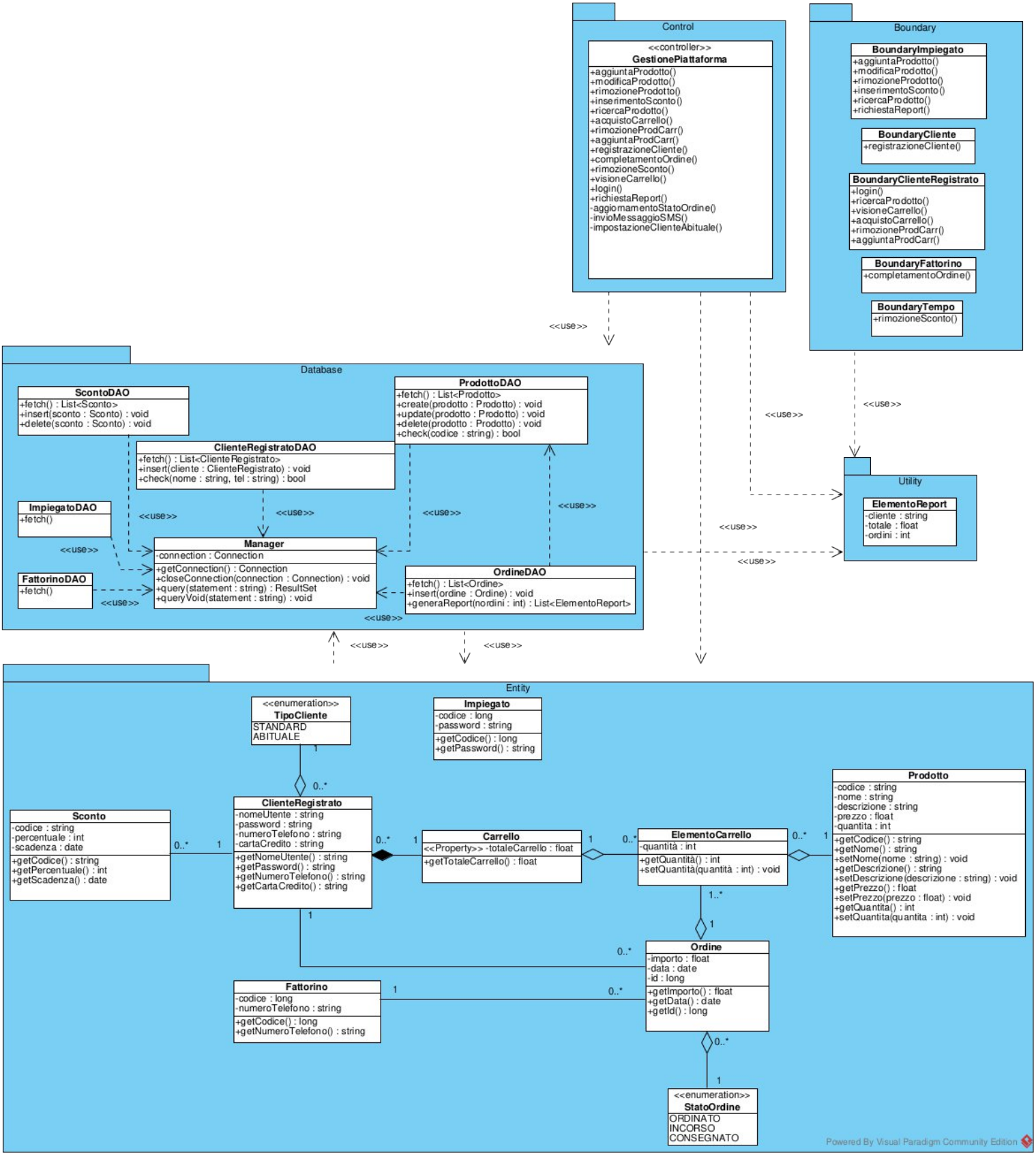
Il numero di test risultante è: (1*1) + 2 = 3

TEST SUITE

ID	Descrizione	Classi di equivalenza coperte	Pre-condizioni	Input	Output Attesi	Post-condizioni Attese
1	Tutti gli input validi	Nordini valido, Elemento database valido	Il database è inizializzato correttamente ed esiste almeno un utente, con almeno N ordini ad esso associati	{Nordini: 5}	Sarà resituita una lista di clienti con associati totale speso e numero di ordini	L'impiegato potrà visionare il report che ha richiesto
2	Numero NORDINI non compreso tra 0 e 1000	Nordini non compreso tra 0 e 1000 [ERROR], Elemento database valido	Il sistema richiede All'impiegato di specificare un numero "soglia" di oridni al momento della richiesta	{Nordini: -1838}	ERRORE: Il numero di ordini inserito non è valido, riprovate.	Il sistema chiede di reinserire il numero di ordini
3	Non sono presenti Clienti con almeno N ordini nel sistema	Nordini valido, Elemento database [ERROR]	Il database non contiene clienti con associati almeno N ordini	{Nordini: 5}	ERRORE: Non esistono clienti con questo numero di ordini	Il sistema restituisce un messaggio di errore e chiude la funzionalità

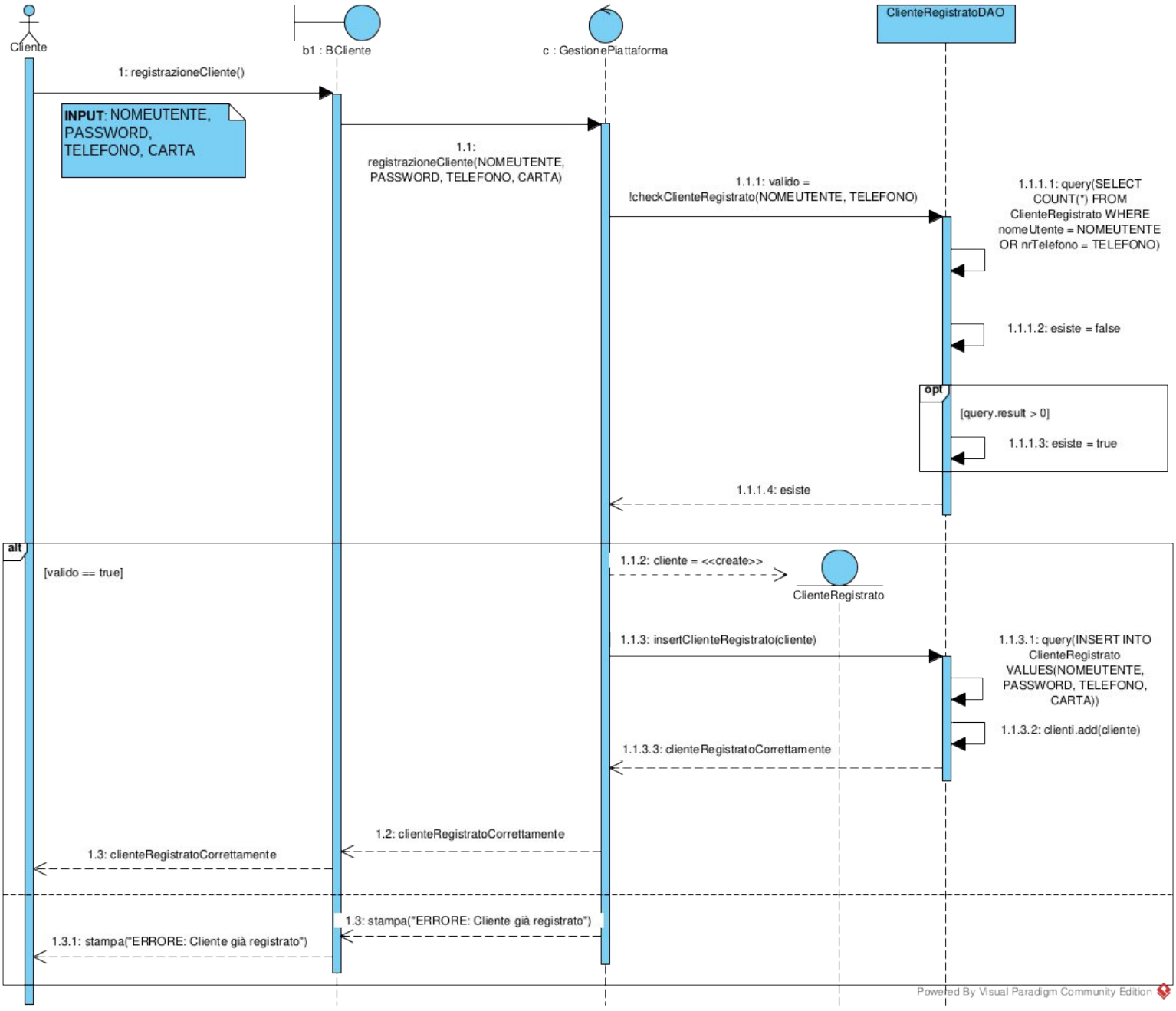
5. Progettazione

5.1 Diagramma delle classi

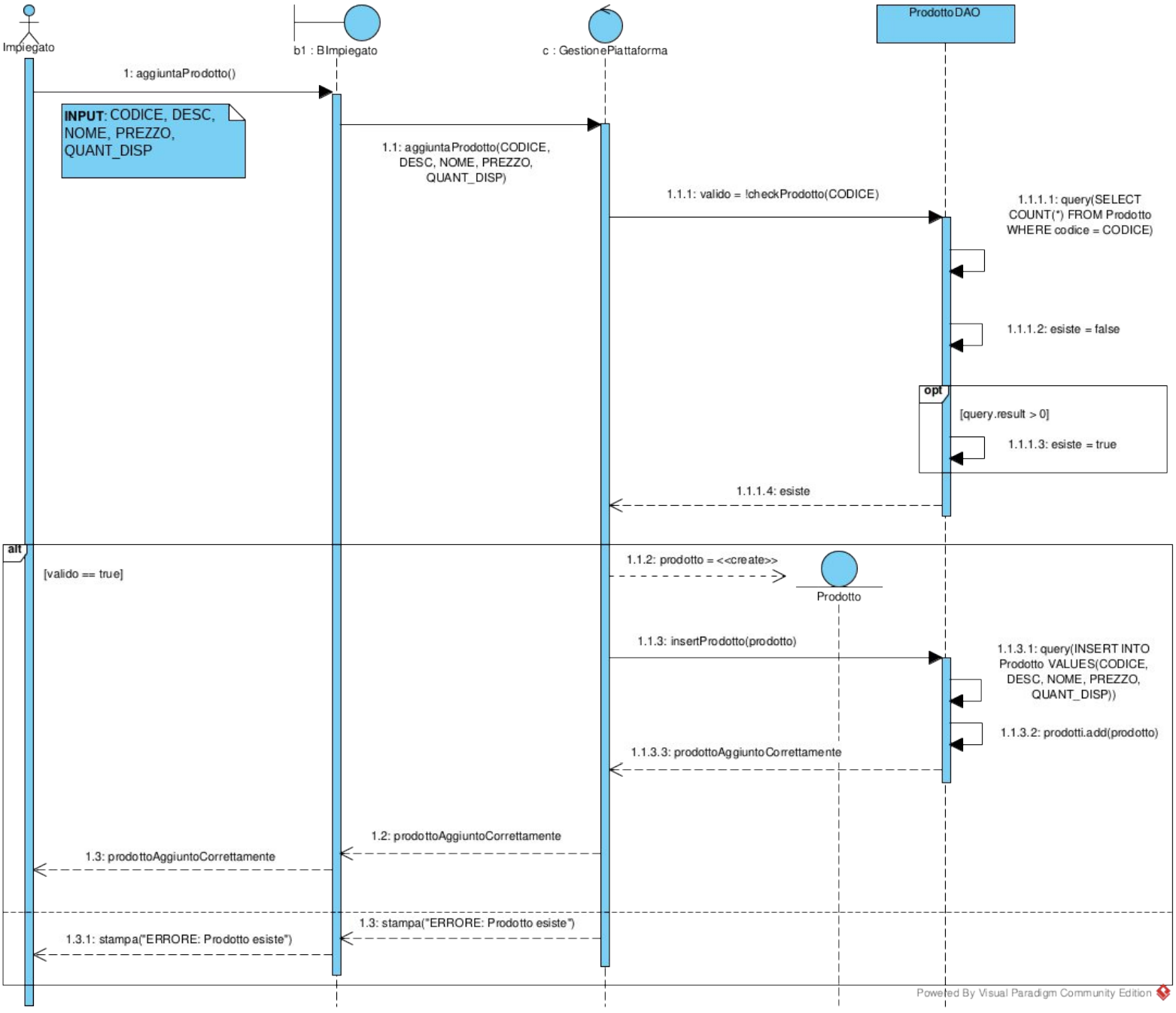


5.2 Diagrammi di sequenza

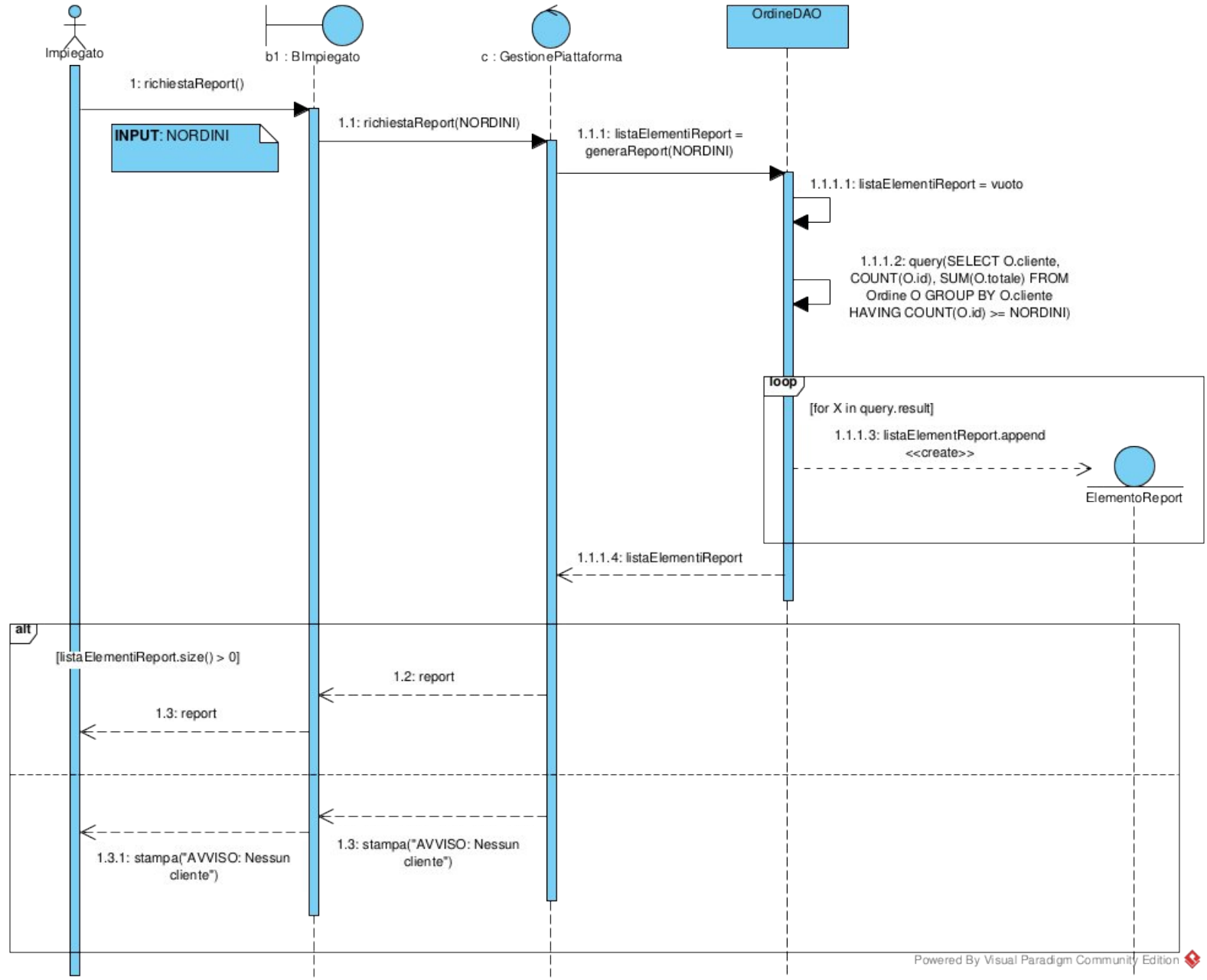
Registrazione



AggiuntaProdotto



RichiestaReport



6. Implementazione

6.1 Scelte di sviluppo

Si è scelto di procedere senza un build system automatizzato (come Maven o Gradle) perchè nel complesso si fa poco uso di framework di terze parti. La compilazione, il testing e l'esecuzione del software sono gestiti tramite **Makefile**, che risulta essere particolarmente efficace in queste circostanze. Il motivo alla base di questa scelta è la voluta possibilità di poter compilare, eseguire ed eventualmente anche sviluppare il software su un **sistema minimale**.

Il software scelto per la base dati è **MariaDB**, per l'estrema flessibilità che offre il linguaggio **MySQL** e le ottime performance che questo software ha dimostrato di poter ottenere anche su larga scala. Inoltre MariaDB estende le capacità del linguaggio di query con utility accessorie che permettono una migliore manipolazione dei dati in ingresso e in uscita dalla base dati. La connessione è possibile grazie al pacchetto .jar **MariaDB-Java-Client**, posto insieme a JUnit e JSON-Simple nella cartella **/deps** del progetto.

Le scelte di cui sopra, hanno consentito ai membri del team di poter sviluppare con ambienti e sistemi operativi di preferenza personale, grazie alle runtime indipendenti dalla piattaforma.

6.2 Requisiti per il deployment

Di seguito tutto il necessario per procedere al deployment:

- Java Development Kit, versione 17 (**minimo consigliato**)
- Docker e Docker Compose (le ultime versioni **includono il comando "compose"**)
- Make

6.3 Deployment

Una volta clonato il repository del progetto, recarsi nella sua directory radice e:

- Lanciare il comando "**make**" per compilare il codice sorgente.
- Lanciare il comando "**docker compose up -d**" per avviare il contenitore MariaDB
- Lanciare il comando "**make run**" per avviare l'applicativo.

Eventualmente, è possibile lanciare i test con JUnit utilizzando il comando "make test".

6.4 Commenti sullo sviluppo

Nel processo di implementazione sono state effettuate un insieme di migliorie, non pensate o comunque in un primo momento non considerate durante la fase di analisi e progettazione del software. Quindi seguito sarà riportato tutto ciò che è stato aggiunto o migliorato. Gli unici casi d'uso implementati sono Registrazione, AggiuntaProdotto e RichiestaReport, ovvero quelli scelti da ogni singolo studente del progetto. Oltre questi,

ovviamente sono stati implementati un insieme di metodi necessari per il corretto funzionamento dei casi d'uso.

In Generale:

Sono stati creati i seguenti package: **boundary**, **entity**, **control**, **dao**, **exception**, **utility.**
Per facilitare le successive implementazioni e favorire una maggiore pulizia generale del codice sorgente, il team ha provveduto a fornire ulteriori classi:

Per il package utility sono state create le classi **Logger, InputScanner e JSONCoder**.
Logger è un rimpiazzo della stampa classica di Java, utile per compattare sezioni con output elaborati nel codice sorgente. Fornisce la possibilità di colorare il testo scegliendo tra vari livelli di severità del messaggio. Provvede a stampare nel corretto flusso di output, utilizzando stderr per i messaggi di errore e stdout per tutti gli altri.
InputScanner è un rimpiazzo, o meglio un wrapper, per lo scanner classico di Java, utile in quei casi ove si voglia ottenere un input sanitizzato e certamente privo di errori. InputScanner è particolarmente utile per navigare un'interfaccia testuale con menù a selezione numerica intera, mantenendo pulito e leggibile il codice chiamante.
JSONCoder è una estensione di JSONParser, una classe del pacchetto json-simple, che ha lo scopo di fornire le necessarie utility per la codifica degli oggetti da importare nella base dati sotto forma di longtext.

Per il package dao è stata creata l'interfaccia generica denominata "**Interface**", che tutte le classi Dao implementano. L'interfaccia supporta i quattro metodi CRUD (denominati nel nostro caso "fetch", "insert", "update", "delete") e prevede una specifica generalizzata per consentire implementazioni personalizzate sulla base del tipo di dato che la classe Dao andrà a gestire. E' importante sottolineare che **non tutti i metodi sono implementati**, dal momento che il team ha scelto di dare precedenza allo stretto necessario per il corretto funzionamento dei casi d'uso.

Nel package exception sono state definite le seguenti eccezioni personalizzate: **ParametroInvalido, ProdottoEsistente, UtenteEsistente, ReportVuoto**. Vale la pena di spendere due parole su ParametroInvalido, un'eccezione creata su misura per gestire valori inesatti o errati all'interno di cicli, che fa uso dei due parametri "index" e "label" per chiedere all'utilizzatore di reinserire gli input da tastiera in maniera sistematica. Si sfrutta a pieno questa eccezione nelle classi Boundary.

Caso d'uso Registrazione:

L'implementazione di questo UC segue correttamente la specifica di progettazione, facendo uso dei seguenti metodi:

-***registrazione***(): metodo pubblico della classe control "GestionePiattaforma", verifica che i parametri forniti da tastiera siano corretti e che il Cliente non sia già registrato; dopodichè avvia la procedura di registrazione sottostante con la classe DAO, eventualmente lanciando le dovute eccezioni.

-**registrazione**(): metodo privato della classe boundary "BoundaryCliente", il cui scopo è fornire indicazioni sul procedimento e ottenere dal Cliente un input da tastiera, che verrà poi inviato al metodo di cui sopra, della classe control. Gestisce le eccezioni e presenta gli eventuali messaggi di errore.

Caso d’uso AggiuntaProdotto:

-**aggiuntaProdotto**(): metodo pubblico della classe control “GestionePiattaforma“, valida e aggiunge un nuovo prodotto al database. Verifica che i parametri forniti (codice, nome, descrizione, prezzo, quantita) rispettino i criteri di lunghezza e formato, lancia eccezioni ParametroInvalido per parametri non validi, e controlla se il prodotto esiste già con prodottoDAO.check(codice), lanciando ProdottoEsistente in caso affermativo. Se tutti i controlli sono superati, inserisce il nuovo prodotto nel database con prodottoDAO.insert().

-**aggiuntaProdotto**(): metodo privato della classe boundary “BoundaryImpiegato”, guida l'utente nell'inserimento di un nuovo prodotto, richiedendo e validando i parametri "CODICE", "NOME", "DESCRIZIONE", "PREZZO" e "QUANTITA'". I parametri vengono passati al controller per l'aggiunta del prodotto, con un messaggio di conferma visualizzato al termine dell'operazione. Gestisce le eccezioni e presenta gli eventuali messaggi di errore.

Caso d’uso RichiestaReport:

-**richiestaReport**(): metodo pubblico della classe control “GestionePiattaforma”, esegue controlli sul parametro in ingresso e chiama il metodo “generaReport()” della classe OrdineDAO. Se il report ricevuto è vuoto, lancia un’eccezione.

-**richiestaReport**(): metodo privato della classe boundary “BoundaryImpiegato”, chiede all’utilizzatore di inserire “NUMERO ORDINI” da tastiera, provvedendo a contattare il control di cui sopra e a mostrare il risultato a schermo. Gestisce le eccezioni presentando eventualmente i messaggi di errore e/o chiedendo di reinserire il parametro.

7. Testing funzionale

REGISTRAZIONE

ID	Descrizione	Classi di equivalenza coperte	Pre-condizioni	Input	Output Attesi	Post-condizioni Attese	Output Ottenuti	Post-condizioni Ottenute	Esito
1	Tutti gli input validi	Nomeutente valido, Nrtelefono valido, Cartacredito valido, Password valido, Elemento database valido	Il database è inizializzato correttamente e esiste nessun'altro utente con questa email o cellulare	{nomeUtente: paolobrosio22, nrTelefono: 3553925732, cartaCredito: 4444444444444444, password: hulk#buster27}	Registrazione avvenuta correttamente	Il cliente è correttamente registrato all'interno della base dati.	Registrazione avvenuta correttamente	Il cliente è correttamente registrato all'interno della base dati.	PASS
2	Stringa NOMEUTENTE contiene simboli	Nomeutente con simboli [ERROR], Nrtelefono valido, Cartacredito valido, Password valido, Elemento database valido	Il sistema richiede al cliente di inserire un nome utente al momento della registrazione	{nomeUtente: paolo@@brosio22, nrTelefono: 3553925732, cartaCredito: 4444444444444444, password: hulk#buster27}	ERRORE: Il nome utente inserito non è valido, riprovare.	Il sistema chiede di reinserire il nome utente.	ERRORE: Il nome utente inserito non è valido, riprovare.	Il sistema chiede di reinserire il nome utente.	PASS

3	Stringa NOMEUTENTE > 255	Nomeutente > 255 [ERROR], Nrtelefono valido, Cartacredito valido, Password valido, Elemento database valido	Il sistema richiede al cliente di inserire un nome utente al momento della registrazione	{nomeUtente: forzanapoliiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiii ... iiiii, nrTelefono: 3553925732, cartaCredito: 4444444444444444, password: hulk#buster27}	ERRORE: Il nome utente inserito non è valido, riprovare.	Il sistema chiede di reinserire il nome utente.	ERRORE: Il nome utente inserito non è valido, riprovare.	Il sistema chiede di reinserire il nome utente.	**PASS**
4	Stringa NRTELEFONO contiene caratteri o simboli	Nomeutente valido, Nrtelefono con caratteri o simboli [ERROR], Cartacredito valido, Password valido, Elemento database valido	Il sistema richiede al cliente di inserire un numero di telefono al momento della registrazione	{nomeUtente: paolobrosio22, nrTelefono: 5ASDA2#32@, cartaCredito: 4444444444444444, password: hulk#buster27}	ERRORE: Il numero di telefono inserito non è valido, riprovare.	Il sistema chiede di reinserire il numero di telefono	ERRORE: Il numero di telefono inserito non è valido, riprovare.	Il sistema chiede di reinserire il numero di telefono	**PASS**
5	Stringa NRTELEFONO con lunghezza != 10	Nomeutente valido, Nrtelefono con lunghezza != 10 [ERROR], Cartacredito valido, Password valido, Elemento database valido	Il sistema richiede al cliente di inserire un numero di telefono al momento della registrazione	{nomeUtente: paolobrosio22, nrTelefono: 3553925732333, cartaCredito: 4444444444444444, password: hulk#buster27}	ERRORE: Il numero di telefono inserito non è valido, riprovare.	Il sistema chiede di reinserire il numero di telefono	ERRORE: Il numero di telefono inserito non è valido, riprovare.	Il sistema chiede di reinserire il numero di telefono	**PASS**

6	Stringa CARTACREDITO contiene caratteri o simboli	Nomeutente valido, Nrtelefono valido, Cartacredito con caratteri o simboli [ERROR], Password valido, Elemento database valido	Il sistema richiede al cliente di inserire un numero di carta di credito al momento della registrazione	{nomeUtente: paolobrosio22, nrTelefono: 3553925732, cartaCredito: 4S444#4444G44444, password: hulk#buster27}	ERRORE: Il numero di carta inserito non è valido, riprovare.	Il sistema chiede di reinserire il numero di carta di credito	ERRORE: Il numero di carta inserito non è valido, riprovare.	Il sistema chiede di reinserire il numero di carta di credito	**PASS**
7	Stringa CARTACREDITO con lunghezza != 16	Nomeutente valido, Nrtelefono valido, Cartacredito con lunghezza != 16 [ERROR], Password valido, Elemento database valido	Il sistema richiede al cliente di inserire un numero di carta di credito al momento della registrazione	{nomeUtente: paolobrosio22, nrTelefono: 3553925732, cartaCredito: 777, password: hulk#buster27}	ERRORE: Il numero di carta inserito non è valido, riprovare.	Il sistema chiede di reinserire il numero di carta di credito	ERRORE: Il numero di carta inserito non è valido, riprovare.	Il sistema chiede di reinserire il numero di carta di credito	**PASS**
8	Stringa PASSWORD > 255	Nomeutente valido, Nrtelefono valido, Cartacredito valido, Password con lunghezza > 255 [ERROR], Elemento database valido	Il sistema richiede al cliente di inserire una password al momento della registrazione	{nomeUtente: paolobrosio22, nrTelefono: 3553925732, cartaCredito: 4444444444444444, password: hulk#busteeeeeeeeeee ... eeeeeeeer27}	ERRORE: La passowrd inserita non è valida, riprovare.	Il sistema chiede di reinserire la password	ERRORE: La passowrd inserita non è valida, riprovare.	Il sistema chiede di reinserire la password	**PASS**

9	Cliente già presente nel sistema	Nomeutente valido, Nrtelefono valido, Cartacredito valido, Password valido, Elemento database [ERROR]	Il database contiene già un utente registrato con lo stesso nome utente o lo stesso cellulare	{nomeUtente: paolobrosio22, nrTelefono: 3553925732, cartaCredito: 4444444444444444, password: hulk#buster27}	ERRORE: Cliente già presente nel sistema.	Il sistema restituisce un messaggio di errore e chiude la funzionalità	ERRORE: Cliente già presente nel sistema.	Il sistema restituisce un messaggio di errore e chiude la funzionalità	**PASS**

AggiuntaProdotto

ID	Descrizione	Classi di equivalenza coperte	Pre-condizioni	Input	Output Attesi	Post-condizioni Attese	Output Ottenuti	Post-condizioni Ottenute	Esito
1	Tutti gli input validi	Codice valido, Descrizione valido, Nome valido, Prezzo valido, Ndisponibile valido Elemento database valido	Il database è inizializzato correttamente e esiste nessun'altro prodotto con questo codice	{codice: AEERX923, descrizione: "Bagnodoccia 9 in 1 da uomo, per la cura della tua auto da uomo", nome: PuliMax, prezzo: 0.67, Ndisponibile: 90}	Inserimento avvenuto correttamente	Il prodotto è correttamente registrato all'interno della base dati.	Inserimento avvenuto correttamente	Il prodotto è correttamente registrato all'interno della base dati.	**PASS**
2	Stringa CODICE contiene simboli	Codice con simboli [ERROR], Descrizione valido, Nome valido, Prezzo valido, Ndisponibile valido Elemento database valido	Il sistema richiede all'impiegato di digitare un codice al momento dell' inserimento di un prodotto	{codice: AEER@@@X92, descrizione: "Bagnodoccia 9 in 1 da uomo, per la cura della tua auto da uomo", nome: PuliMax, prezzo: 0.67, Ndisponibile: 90}	ERRORE: Il codice inserito non è valido, riprovare.	Il sistema chiede di reinserire il codice.	ERRORE: Il codice inserito non è valido, riprovare.	Il sistema chiede di reinserire il codice.	**PASS**
3	Stringa CODICE con lunghezza > 255	Codice con lunghezza > 255 [ERROR], Descrizione valido, Nome valido, Prezzo valido, Ndisponibile valido Elemento database valido	Il sistema richiede all'impiegato di digitare un codice al momento dell' inserimento di un prodotto	{codice: E EE EEEE EEEEE ... EEEEEE, descrizione: "Bagnodoccia 9 in 1 da uomo, per la cura della tua auto da uomo", nome: PuliMax, prezzo: 0.67,	ERRORE: Il codice inserito non è valido, riprovare.	Il sistema chiede di reinserire il codice.	ERRORE: Il codice inserito non è valido, riprovare.	Il sistema chiede di reinserire il codice.	**PASS**

				Ndisponibile: 90}					
4	Stringa DESCRIZIONE con lunghezza > 255	Codice valido, Descrizione con lunghezza > 255 [ERROR], Nome valido, Prezzo valido, Ndisponibile valido Elemento database valido	Il sistema richiede all'impiegato di digitare una descrizione al momento dell' inserimento di un prodotto	{codice: AEERX923, descrizione: "Bagnodocciaaaaaaaaaaaa .. . aa", nome: PuliMax, prezzo: 0.67, Ndisponibile: 90}	ERRORE: la descrizione inserita non è valida, riprovare.	Il sistema chiede di reinserire la descrizione.	ERRORE: la descrizione inserita non è valida, riprovare.	Il sistema chiede di reinserire la descrizione.	**PASS**
5	Stringa NOME contiene simboli	Codice valido, Descrizione valido, Nome con simboli [ERROR], Prezzo valido, Ndisponibile valido Elemento database valido	Il sistema richiede all'impiegato di digitare un nome al momento dell' inserimento di un prodotto	{codice: AEEX92, descrizione: "Bagnodoccia 9 in 1 da uomo, per la cura della tua auto da uomo", nome: Puli#@Max, prezzo: 0.67, Ndisponibile: 90}	ERRORE: Il nome inserito non è valido, riprovare.	Il sistema chiede di reinserire il nome.	ERRORE: Il nome inserito non è valido, riprovare.	Il sistema chiede di reinserire il nome.	**PASS**
6	Stringa NOME con lunghezza > 255	Codice valido, Descrizione valido, Nome con lunghezza > 255 [ERROR], Prezzo valido, Ndisponibile valido Elemento database valido	Il sistema richiede all'impiegato di digitare un nome al momento dell' inserimento di un prodotto	{codice: AEEX92, descrizione: "Bagnodoccia 9 in 1 da uomo, per la cura della tua auto da uomo", nome: PuliMaaaaaaaaaaaaaaaa .., prezzo: 0.67, Ndisponibile: 90}	ERRORE: Il nome inserito non è valido, riprovare.	Il sistema chiede di reinserire il nome.	ERRORE: Il nome inserito non è valido, riprovare.	Il sistema chiede di reinserire il nome.	**PASS**

7	Numero PREZZO NON compreso tra 0 e 1000	Codice valido, Descrizione valido, Nome valido, Prezzo non compreso tra 0 e 1000 [ERROR], Ndisponibile valido Elemento database valido	Il sistema richiede all'Impiegato di digitare un prezzo al momento dell' inserimento di un prodotto	{codice: AEERX923, descrizione: "Bagnodoccia 9 in 1 da uomo, per la cura della tua auto da uomo", nome: PuliMax, prezzo: -0.67, Ndisponibile: 90}	ERRORE: Il prezzo inserito non è valido, riprovare.	Il sistema chiede di reinserire il prezzo	ERRORE: Il prezzo inserito non è valido, riprovare.	Il sistema chiede di reinserire il prezzo	**PASS**
8	Numero NDISPONIBILE NON compreso tra 0 e 1000	Codice valido, Descrizione valido, Nome valido, Prezzo valido, Ndisponibile non compreso tra 0 e 1000 [ERROR], Elemento database valido	Il sistema richiede all'Impiegato di digitare una quantità disponibile al momento dell' inserimento di un prodotto	{codice: AEERX923, descrizione: "Bagnodoccia 9 in 1 da uomo, per la cura della tua auto da uomo", nome: PuliMax, prezzo: 0.67, Ndisponibile: -90}	ERRORE: La quantità disponibileinserita non è valida, riprovare.	Il sistema chiede di reinserire la quantità disponibile	ERRORE: La quantità disponibileinserita non è valida, riprovare.	Il sistema chiede di reinserire la quantità disponibile	**PASS**
9	Prodotto già presente nel sistema	Codice valido, Descrizione valido, Nome valido, Prezzo valido, Ndisponibile valido Elemento database [ERROR]	Il database contiene già un prodotto registrato con lo stesso codice	{codice: AEERX923, descrizione: "Bagnodoccia 9 in 1 da uomo, per la cura della tua auto da uomo", nome: PuliMax, prezzo: 0.67, Ndisponibile: 90}	ERRORE: Prodotto già presente nel sistema.	Il sistema restituisce un messaggio di errore e chiude la funzionalità	ERRORE: Prodotto già presente nel sistema.	Il sistema restituisce un messaggio di errore e chiude la funzionalità	**PASS**

RichiestaReport

ID	Descrizione	Classi di equivalenza coperte	Pre-condizioni	Input	Output Attesi	Post-condizioni Attese	Output Ottenuti	Post-condizioni Ottenute	Esito
1	Tutti gli input validi	Nordini valido, Elemento database valido	Il database è inizializzato correttamente ed esiste almeno un utente, con almeno N ordini ad esso associati	{Nordini: 5}	Sarà resituita una lista di clienti con associati totale speso e numero di ordini	L'impiegato potrà visionare il report che ha richiesto	Resituita una lista di clienti con associati totale speso e numero di ordini	L'impiegato potrà visionare il report che ha richiesto	**PASS**
2	Numero NORDINI non compreso tra 0 e 1000	Nordini non compreso tra 0 e 1000 [ERROR], Elemento database valido	Il sistema richiede All'impiegato di specificare un numero "soglia" di oridni al momento della richiesta	{Nordini: -1838}	ERRORE: Il numero di ordini inserito non è valido, riprovate.	Il sistema chiede di reinserire il numero di ordini	ERRORE: Il numero di ordini inserito non è valido, riprovate.	Il sistema chiede di reinserire il numero di ordini	**PASS**
3	Non sono presenti Clienti con almeno N ordini nel sistema	Nordini valido, Elemento database [ERROR]	Il database non contiene clienti con associati almeno N ordini	{Nordini: 5}	ERRORE: Non esistono clienti con questo numero di ordini	Il sistema restituisce un messaggio di errore e chiude la funzionalità	ERRORE: Non esistono clienti con questo numero di ordini	Il sistema restituisce un messaggio di errore e chiude la funzionalità	**PASS**