Testo Esercizio
Si consideri un sistema per la gestione di un magazzino di un negozio scelto a piacere dal candidato. Il sistema è in grado di gestire le seguenti operazioni:

- Arrivo di nuovi prodotti;
- Controllo del livello di scorta di un prodotto;
- Estrazione del prodotto per la vendita e fatturazione al cliente finale;
- Emissione automatica di un ordine al fornitore del prodotto qualora il livello di scorta scende al di sotto di una certa soglia

Si illustrino le gerarchie di classi più significative e si costruisca il relativo diagramma di classi.
Si costruisca un diagramma di sequenza per le operazioni di arrivo di un nuovo prodotto e di emissione automatica di un ordine al fornitore in seguito alla vendita di una certa quantità di tale prodotto al cliente finale.
Si modelli il sistema in esame mediante un DFD nel quale si descrivano i tre flussi principali relativi all'acquisto di un prodotto, al riordino automatico di un prodotto se la quantità disponibile è inferiore a quella desiderata e all'arrivo di un nuovo prodotto.
Si modelli mediante una rete di Petri il sistema in esame in cui il fornitore è visto come un produttore di prodotti, il cliente è visto come un consumatore di prodotti. Il magazzino inizialmente ha uno spazio libero per immagazzinare un nuovo prodotto. Il fornitore può produrre un nuovo prodotto solo se c'è spazio disponibile nel magazzino per il suo immagazzinamento; il cliente può acquistare (consumare) un prodotto qualora il magazzino contiene tale prodotto.

Sommario

- Parte 1 (modello UML, vista logica, aspetti strutturali) – rappresentazione delle gerarchie e delle classi principali del sistema in UML;
- Parte 2 (modello UML, vista logica, aspetti dinamici) – rappresentazione di uno scenario d'utilizzo del sistema mediante diagrammi di sequenza;
- Parte 3 (modello del flusso di dati all'interno del sistema) – data flow diagram;
- Parte 4 (modello delle transizioni all'interno del sistema) – rete di Petri.

Note relative alla modellazione UML.
La soluzione proposta è solo una possibile soluzione. In generale, non esiste la "soluzione corretta" o il "modello migliore". I modelli si costruiscono esattamente come avviene per qualsiasi artefatto umano. Molti dettagli possono essere soggettivi e dipendono fondamentalmente dalla capacità di astrazione, dallo spirito di osservazione e dalla creatività dell'autore. Questa proposta di soluzione rappresenta solo una linea guida: essa costituisce un invito a sperimentare, ad essere creativi, ma allo stesso tempo ad essere coerenti. Gli aspetti fondamentali di un modello UML sono i seguenti:

- Individuare i concetti essenziali del sistema e rappresentarli opportunamente;
- Caratterizzare ciascun concetto con il livello di dettaglio utile per risolvere il problema;
- Mantenere la coerenza tra le diverse viste (o prospettive) del modello. Se mostriamo un'interazione tra due oggetti in un diagramma di sequenza, per esempio, il metodo descritto nel diagramma deve essere stato prima definito nella classe, con gli opportuni argomenti (se ce ne sono).

Un modello è ragionevole quando contiene queste tre caratteristiche.

Note relative al testo dell'esercizio.
Solitamente i testi d'esame sono più brevi: viene richiesto un esercizio su UML e su una rete di Petri, ad esempio. Difficilmente in un unico problema vengono chiesti tutti i punti illustrati in questo esercizio. Tuttavia mi è sembrato interessante mostrare come sia possibile fornire diversi modelli per uno stesso problema. La comparazione dei vari modelli ci permette di capire come si possa affrontare l'analisi di uno stesso problema secondo prospettive diverse, ottenendo rappresentazioni altrettanto diverse.

Svolgimento

Parte 1

Il primo passo richiesto dall'esercizio consiste nell'identificare le principali gerarchie d'ereditarietà e costruire un modello strutturale del sistema basato sui diagrammi di classe. Costruiamo tale modello per piccoli incrementi. Le seguenti gerarchie si deducono facilmente dal testo dell'esercizio:

- *Prodotto* (sceglieremo in seguito il tipo di negozio e, quindi, esamineremo successivamente i diversi tipi di prodotto previsti);
- *Ordine* - esistono degli ordini d'acquisto di un prodotto fatti al cliente finale (*OrdineCliente*) e degli ordini d'acquisto fatti al fornitore di un prodotto(*OrdineFornitore*);

Una terza gerarchia potenzialmente utile (e sicuramente ragionevole) è costituita dal personale del negozio (*PersonaleNegozio*). Possiamo qui identificare diverse entità quali l'addetto al magazzino (*AddettoMagazzino*), l'addetto di un reparto del negozio (*AddettoReparto*) e il commesso alla cassa (*CommessoCassa*). La **figura 1** illustra queste entità, fornendo una descrizione delle caratteristiche interne più importanti (attributi e metodi).

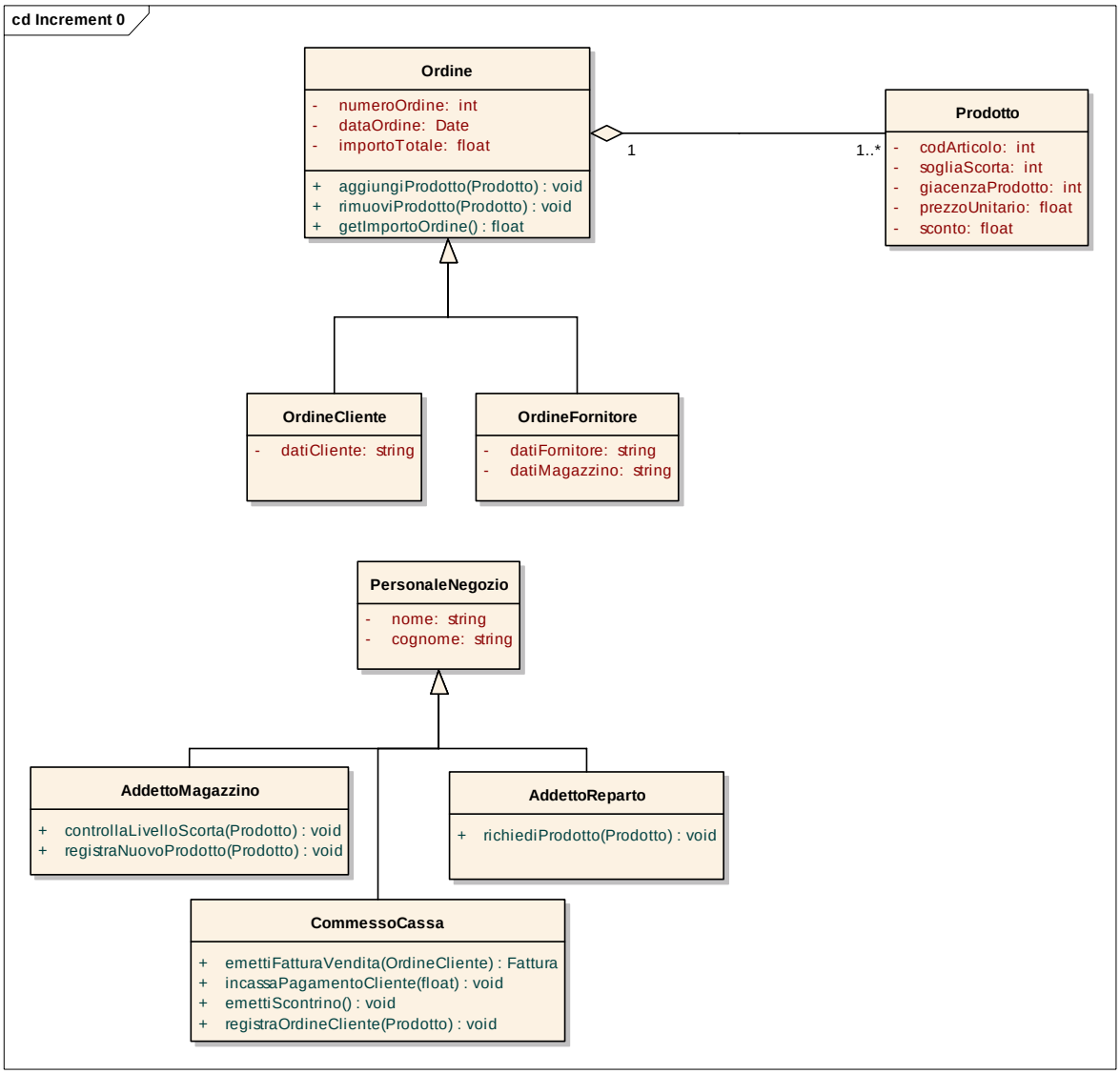

Figura 1 - Le prime gerarchie del sistema

In particolare osserviamo come un ordine aggreghi al suo interno uno o più prodotti. Per ogni prodotto possiamo conoscere il suo codice d'articolo, la soglia della scorta di magazzino al di sotto della quale è necessario fare un riordino al fornitore, l'attuale giacenza del prodotto in magazzino, il prezzo unitario del prodotto e l'eventuale sconto. I dati più importanti per un ordine sono il numero dell'ordine, la data dell'ordine e l'importo totale. Per quanto riguarda gli ordini dei clienti, aggiungiamo i dati del cliente (dovremmo fatturare…), mentre per quanto riguarda l'ordine al fornitore scegliamo di includere sia i dati del fornitore, sia quelli del magazzino. Per quanto riguarda la gerarchia sul personale, a parte i dati anagrafici, è interessante modellare alcune operazioni che ciascun membro può effettuare. L'addetto al magazzino, ad esempio, sarà responsabile del controllo del livello di scorta di un prodotto oppure della registrazione di un nuovo prodotto, l'addetto alla cassa dell'emissione della fattura, l'addetto di reparto di soddisfare richieste di informazioni (da parte di clienti) relativamente ad un prodotto (locazione del prodotto nel negozio, ad esempio).
Il secondo incremento ci porta ad analizzare entità collegate alle gerarchie proposte, quali ad esempio il magazzino e il fornitore, illustrate in **figura 2**. Il fornitore ha le responsabilità di inviare un nuovo prodotto al magazzino e di registrare un ordine di un prodotto (proveniente sempre dal magazzino). In generale ad un magazzino sono associati uno o più fornitori. Ipotizziamo che ci sia un fornitore per ogni prodotto e che ogni fornitore produca un solo tipo di prodotti.

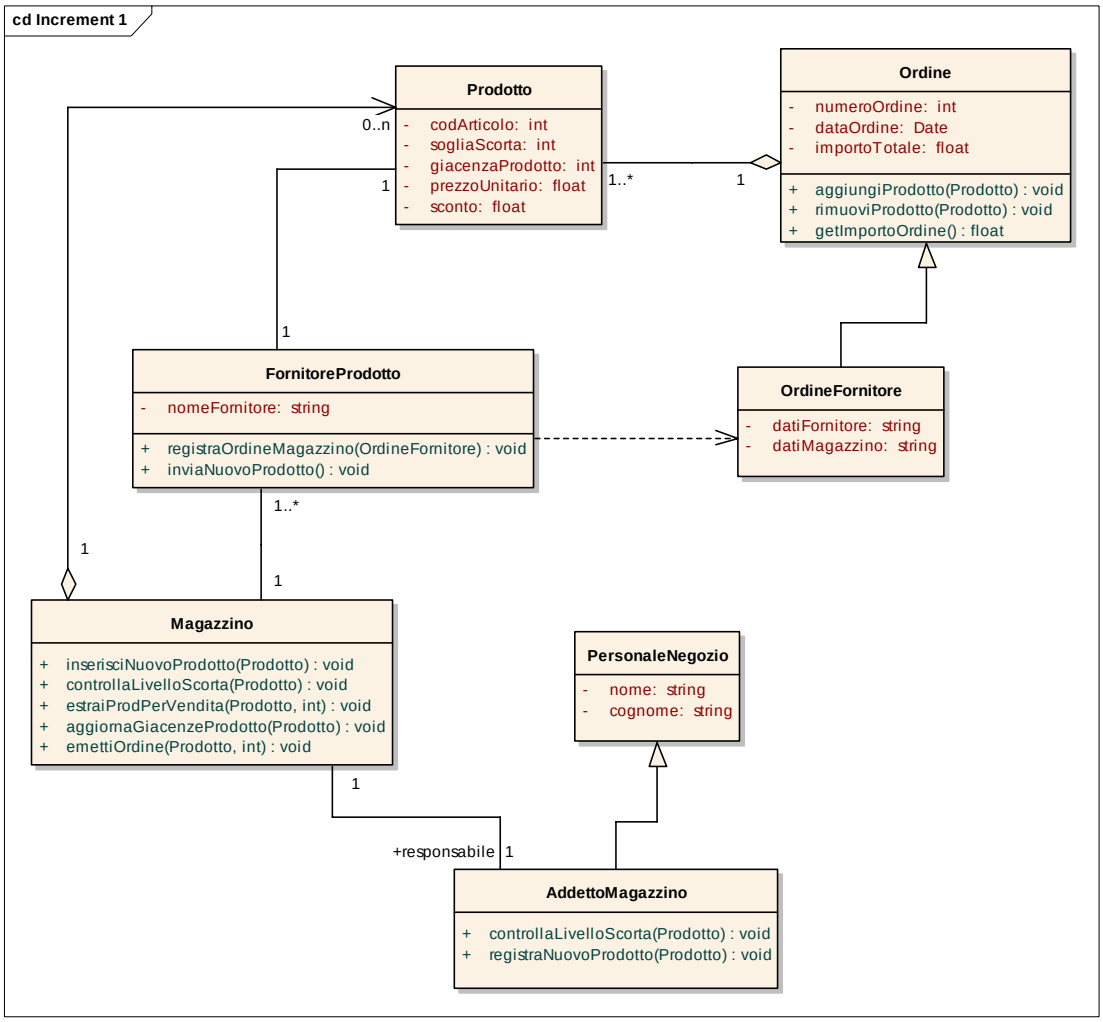

Figura 2 – Aggiungiamo alcune entità necessarie per completare l'esercizio

Il magazzino, per contro, aggrega da zero (il magazzino può essere vuoto) a n prodotti (il magazzino ha una sua capienza massima, quindi non può contenere un numero arbitrario * di prodotti). L'addetto al magazzino è associato al magazzino stesso e svolge il ruolo di responsabile.

Il fornitore prodotto riceve (come parametro del metodo *registraOrdineMagazzino*) un'istanza di un OrdineFornitore, quindi dipende da essa.
Fino a questo momento il modello proposto è ragionevole per qualsiasi negozio. Scegliamo ora di modellare un particolare tipo di negozio che vende prodotti musicali e affini. Alcune classi derivate della gerarchia *Prodotto* posso essere i cd musicali (*CDMusicale*), i *DVD*, la prevendita di biglietti di concerti (*BigliettoConcerto*) e i supporti vergini (*SupportoVergine*), non ulteriormente specificati (ma è ragionevole poter estendere la gerarchia dei supporti con entità del tipo *Nastro*, *CD*, *CR-RW*, *Musicassetta*, ecc.). La **figura 3** illustra la gerarchia prodotto con gli attributi più importanti. Notiamo anche l'inserimento della classe *Fattura* relativamente ad un ordine di un cliente, come richiesto dall'esercizio.

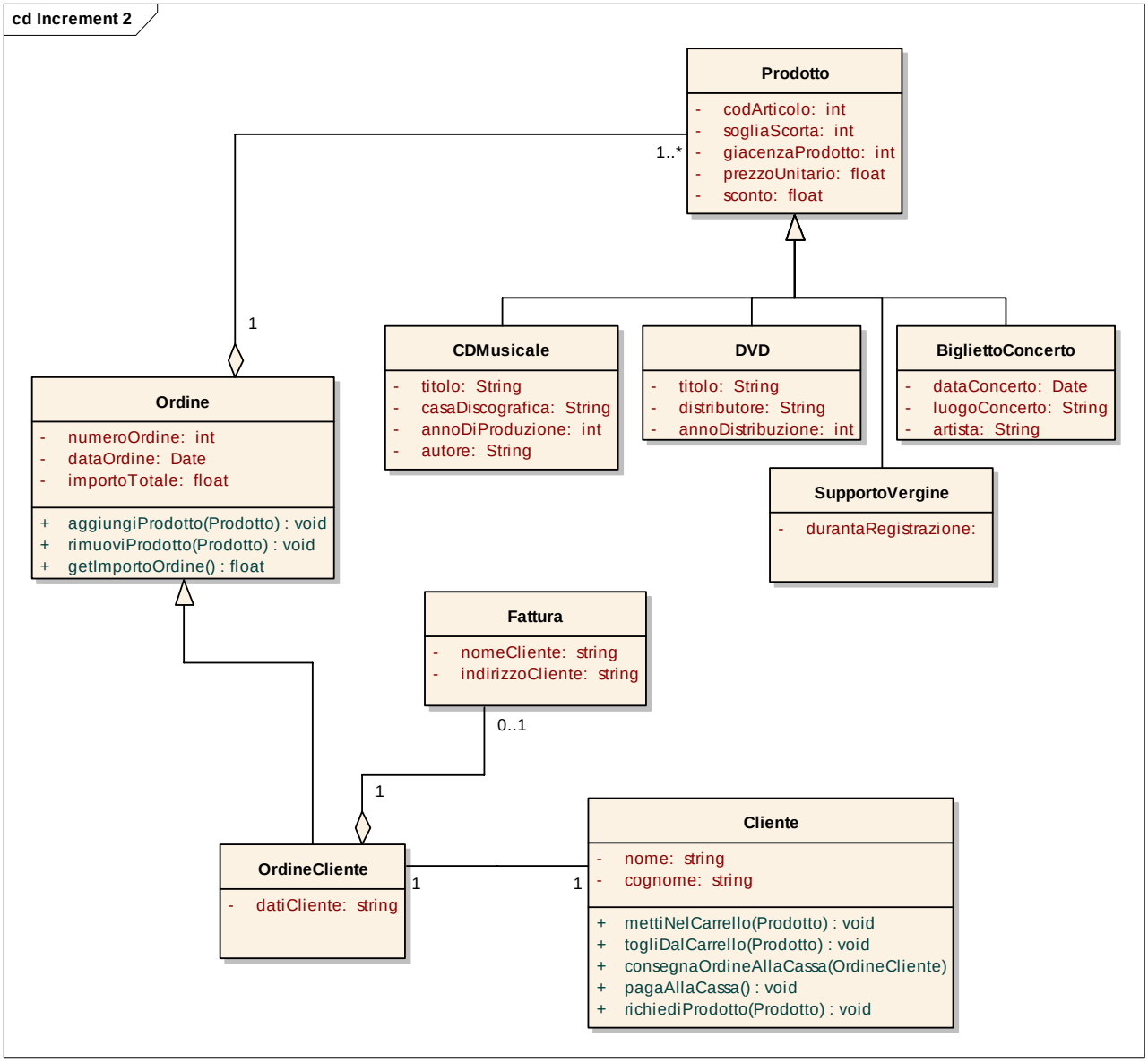

Figura 3 - La gerarchia prodotto

L'ultimo incremento (tipicamente mai elaborato all'esame, a causa del limitato tempo a disposizione), consiste nel mettere insieme tutti i pezzi della soluzione fin qui descritta. Aggiungiamo inoltre qualche dipendenza (una ovvia, probabilmente, tra Cliente e CommessoCassa) e otteniamo il diagramma illustrato in **figura 4**.

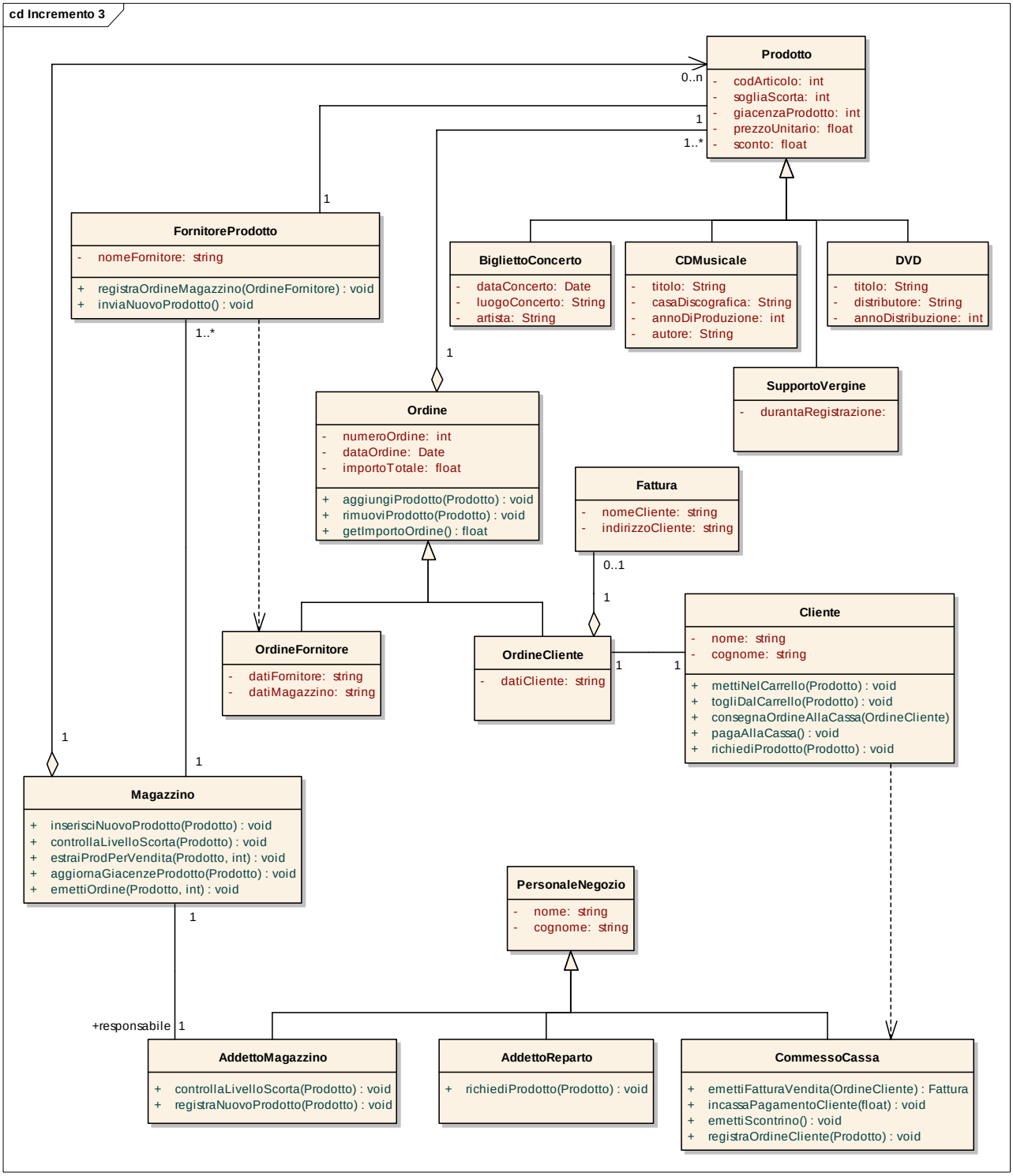

Figura 4 - Il modello strutturale completo

Tutti gli elementi della parte 1 dell'esercizio sono stati identificati e descritti.

Parte 2

Realizziamo ora i due scenari richiesti dall'esercizio. Dobbiamo descrivere mediante un diagramma di sequenza l'arrivo di un nuovo prodotto nel magazzino (facile) e il riordino automatico di un prodotto al rispettivo fornitore a seguito della vendita di tale prodotto al cliente. Nel primo caso, dobbiamo descrivere le interazioni tra le classi *FornitoreProdotto*, *AddettoMagazzino* e *Magazzino*. Immaginiamo che ad iniziare lo scenario sia l'entità *GestoreProdottiFornitore*, esterna al nostro sistema, e per questo rappresentata mediante un attore. Il gestore attiva il metodo *inviaNuovoProdotto* della classe *FornitoreProdotto*. Questo metodo chiama poi *registraNuovoProdotto* di *AddettoMagazzino* il quale, a sua volta, invoca il metodo *inserisciNuovoProdotto* di *Magazzino*. Il magazzino, per completare lo scenario, deve aggiornare la giacenza del nuovo prodotto, immaginando che gli acquisti siano fatti in stock sempre costanti e noti alla classe *Magazzino*. In alternativa, si può modificare il metodo *inviaNuovoProdotto* (e conseguentemente anche tutti gli altri) in modo da passare come argomento sia il prodotto, sia la quantità. La **figura 5** illustra lo scenario di arrivo del nuovo prodotto.

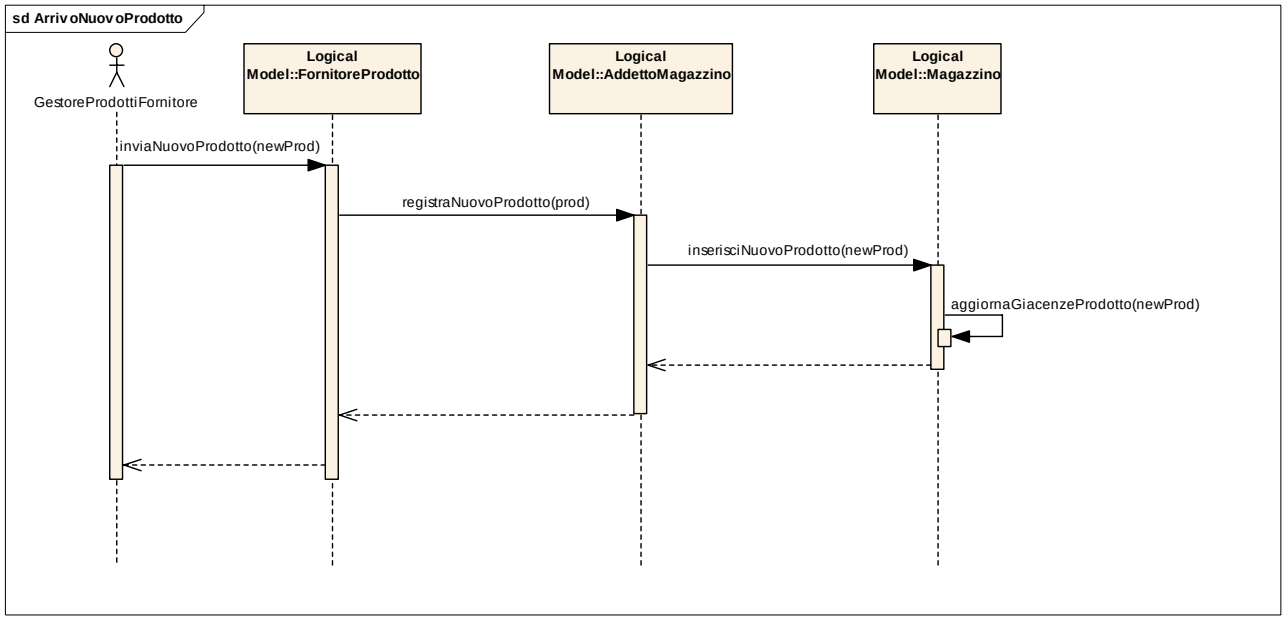

Figura 5 – Arrivo di un nuovo prodotto.

L'ultimo scenario richiesto prevede l'acquisto di un prodotto da parte del cliente e il conseguente riordino di tale prodotto al fornitore se la giacenza in magazzino scende al di sotto della soglia per effetto della vendita. Il cliente inizia lo scenario andando alla cassa per registrare l'ordine. Il commesso della cassa crea quindi un nuovo ordine cliente e aggiunge il prodotto ordinato. Quindi estrae il prodotto per la vendita dal magazzino, passando sia il prodotto, sia la quantità venduta. A questo punto il magazzino controlla il livello di scorta del prodotto e *se* la giacenza è inferiore alla soglia crea un nuovo ordine (questa volta al fornitore!), aggiunge il prodotto e chiama il metodo *registraOrdineMagazzino* della classe *Fornitore*. Ancora una volta si suppone che gli ordini al fornitore avvengano in stock fissi e noti sia al fornitore stesso, sia al magazzino (potrebbe essere reso esplicito aggiungendo l'attributo stockRiordino: int alla classe Magazzino). In alternativa anche qui potrebbe essere passata come argomento anche la quantità di prodotto da riordinare.

Viene infine mostrato uno scenario di possibile pagamento al solo scopo illustrare il life-time dell'oggetto OrdineCliente che, una volta fornito l'importo totale dell'ordine, può essere distrutto. Lo scenario di pagamento non è ulteriormente dettagliato perché non richiesto. Esercizi precedentemente svolti mostrano come effettuare tale pagamento con mezzi diversi quali il Bancomat, la carta di credito, gli assegni, ecc. La **figura 6** illustra lo scenario discusso.

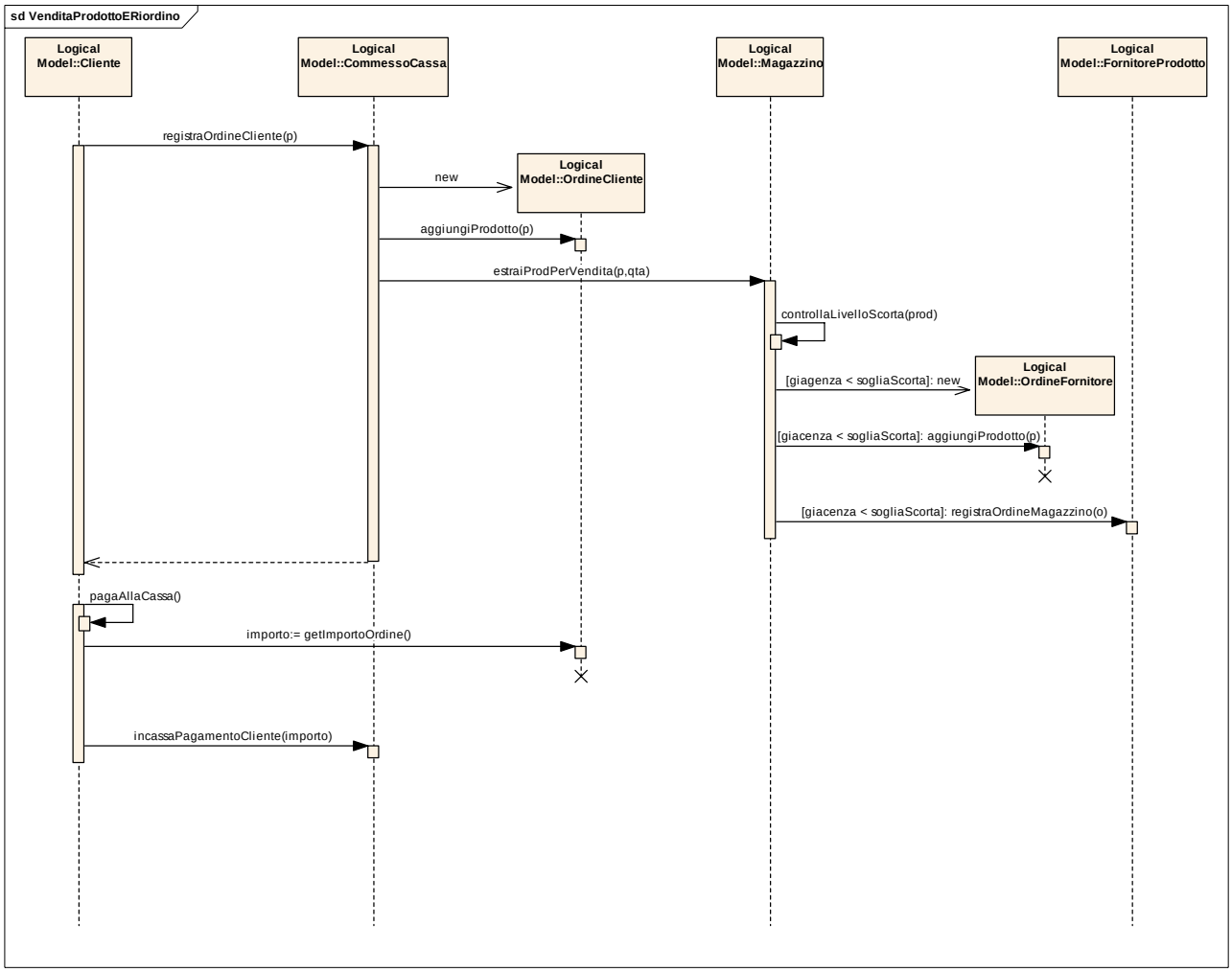

Figura 6 - Scenario di vendita e riordino di un prodotto

Parte 3

Nella terza parte dell'esercizio ci viene richiesto un data-flow diagram (DFD) per il sistema di gestione del magazzino. Un DFD è composto da attività (o processi), repository di dati (data store) e frecce che descrivono il flusso di dati nel sistema. Con questi elementi costruiamo il diagramma illustrato in **figura 7**. Identifichiamo due repository: il database del magazzino e il database del fornitore. Le attività principali sono già evidenti nel testo dell'esercizio e definiscono tre flussi di dati principali:

1. l'acquisto di un prodotto;
2. il riordino automatico del prodotto (se la quantità disponibile è inferiore alla quantità desiderata);
3. l'arrivo di un nuovo prodotto.

Identifichiamo ora le seguenti attività.

Acquisto di un prodotto necessita di:

- Verifica disponibilità prodotto
- Compilazione ordine del cliente
- Emissione della fattura
- Vendita del prodotto
- Aggiornamento del magazzino

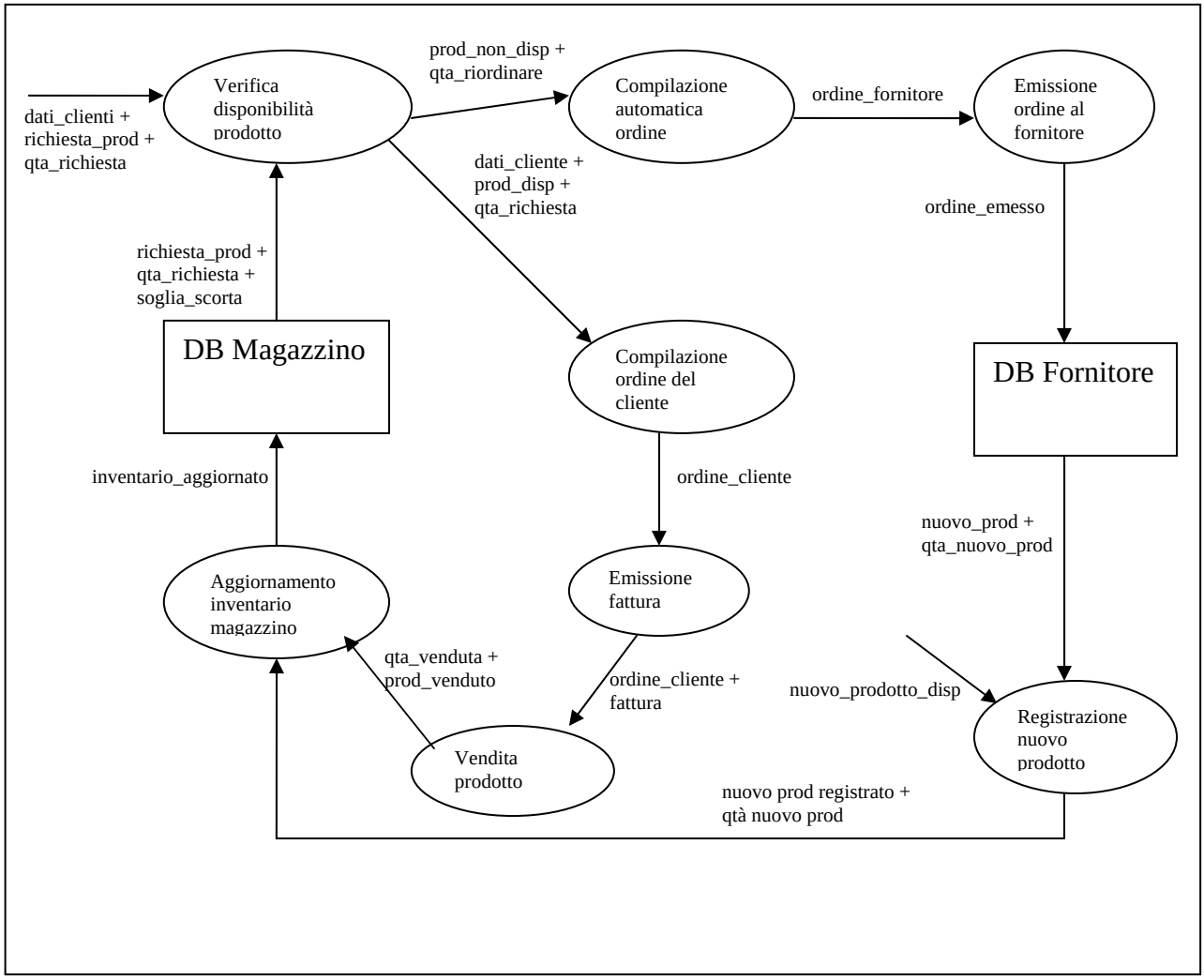

Figura 7 – Data Flow Diagram per il sistema di gestione del magazzino

Il riordino automatico di un prodotto può essere caratterizzato dalle seguenti attività:

- Verifica disponibilità prodotto

- Compilazione automatica ordine
- Emissione ordine al fornitore

Infine l'arrivo di un nuovo prodotto prevede:

- Registrazione del nuovo prodotto
- Aggiornamento inventario magazzino

L'attività di verifica della disponibilità di un prodotto prende in ingresso i dati del cliente, il prodotto richiesto e la quantità desiderata. Se il prodotto non è disponibile viene effettuata una compilazione automatica di un ordine al fornitore (attività che riceve in ingresso il nome del prodotto non disponibile e la quantità da riordinare e produce in uscita l'ordine al fornitore). Se invece il prodotto è disponibile viene effettuata la compilazione dell'ordine del cliente (attività che riceve in ingresso il nome del prodotto disponibile e la quantità richiesta e produce in output l'ordine del cliente). Dopo aver compilato un ordine del cliente, viene emessa la fattura a partire da tale ordine ed infine viene effettuata la vendita del prodotto. La vendita prende in input l'ordine del cliente e la fattura e produce in output la quantità venduta e il prodotto venduto, necessari all'attività di aggiornamento del magazzino per mantenere aggiornato l'inventario. Il resto del diagramma dovrebbe essere ormai piuttosto semplice da seguire.

Alcune considerazioni finali. Come possiamo intuire dall'esempio, per descrivere due flussi di dati alternativi è importante attribuire ai dati associati un nome espressivo che faccia intuire la dinamica (prodotto non disponibile vs. prodotto disponibile). Non abbiamo a disposizione altri costrutti: i flussi alternativi vanno sempre espressi in termini di flusso di dati!

Il diagramma, inoltre, non mostra come ogni singola attività elabori i dati di input per trasformarli in dati di output.

Una volta terminato il diagramma, infine, è importante controllare che ogni attività sia collegata ad almeno un flusso di dati in input e che produca almeno un flusso di dati in output.

Parte 4
In quest'ultima parte dell'esercizio dobbiamo costruire una rete di Petri per il sistema in esame in cui il fornitore è visto come "produttore" (di un prodotto P), il cliente è visto come "consumatore" (del medesimo prodotto) e il magazzino è il repository condiviso. Più in particolare, assumiamo che il magazzino inizialmente abbia uno spazio libero per immagazzinare un nuovo prodotto P che non contiene già. Il fornitore può quindi produrre un prodotto P, ma il cliente non può ancora acquistarlo in quanto non disponibile nel magazzino.

La rete di seguito riportata in **figura 8** può essere un modello adeguato per descrivere questo sistema.

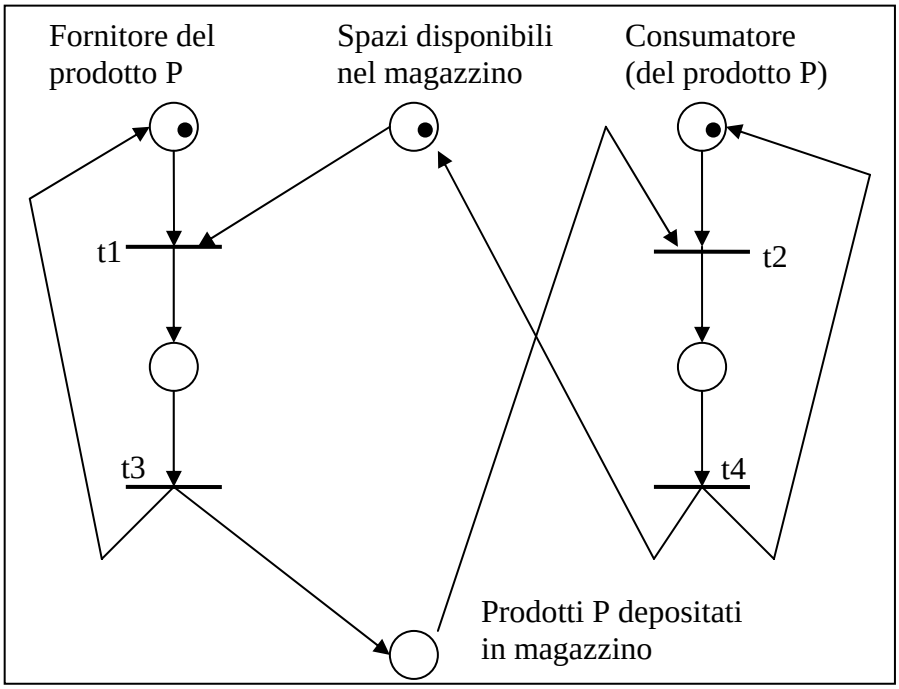

Figura 8 - Rete di Petri per il sistema in esame

Come possiamo osservare dalla rete di figura, la marcatura iniziale riflette esattamente lo stato iniziale del sistema. Assumiamo che il produttore produca al momento un solo prodotto P (abbiamo messo un solo token nella marcatura associata al produttore). L'unica transizione abilitata è t1, che designa la disponibilità di produrre. La transizione t2, che designa la disponibilità a consumare, rimane bloccata dall'attesa di almeno un prodotto P depositato in magazzino (il corrispondente place non ha alcun token al momento).
Ricordiamo che una transizione si attiva solo se *tutti* i place in ingresso hanno almeno un token. Per effetto dell'attivazione della transizione, un token viene prelevato da *tutti* i place in input e viene spostato in (tutti) i suoi place di output. Dopo l'attivazione di t1 la rete è illustrata in **figura 9**.

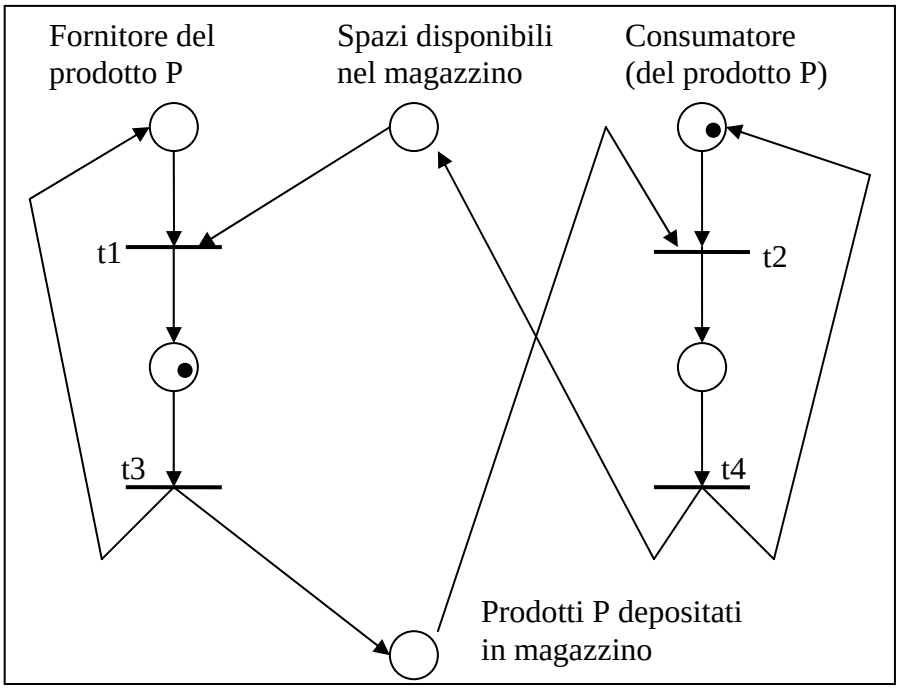

Figura 9 - Rete dopo l'attivazione della transizione t1: il produttore sta producendo

Dopo l'attivazione di t1, per le stesse ragioni esaminate nel caso precedente, l'unica transizione abilitata è t3, la cui attivazione designa la completata produzione e l'immagazzinamento di un prodotto P nel magazzino (**figura 10**). Il fornitore ora è di nuovo pronto a produrre un altro prodotto, ma la corrispondente transizione t1 è ora bloccata dalla mancanza di spazi disponibili del magazzino.

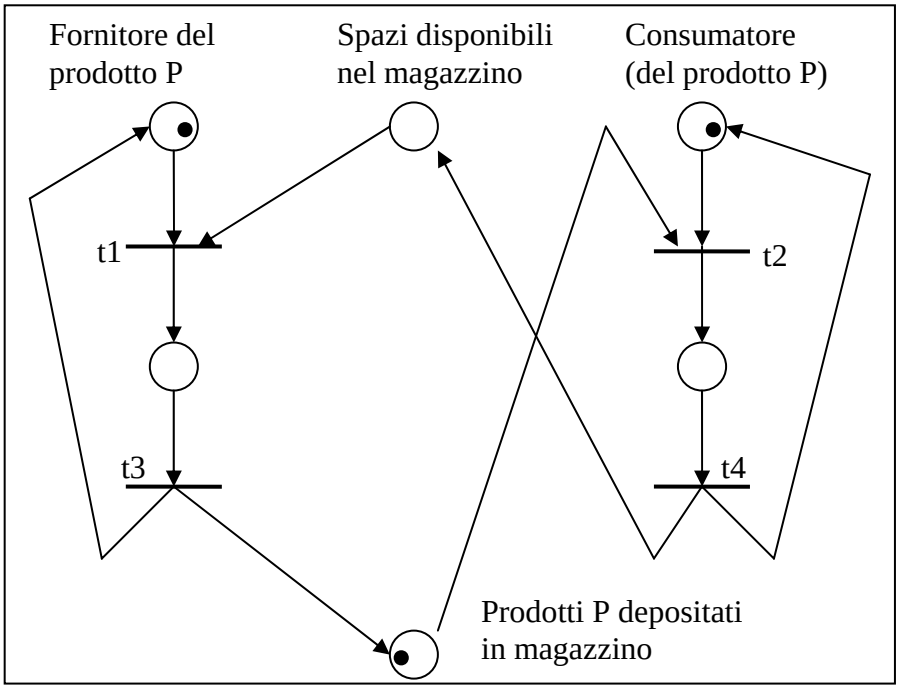

Figura 10 - Rete dopo l'attivazione della transizione t3: il prodotto P è disponibile nel magazzino

Dopo l'attivazione di t3 è pronta a scattare la transizione t2: il consumatore può prelevare dal magazzino il prodotto P che è ora disponibile, come illustrato in **figura 11**. In altre parole, il consumatore sta consumando…

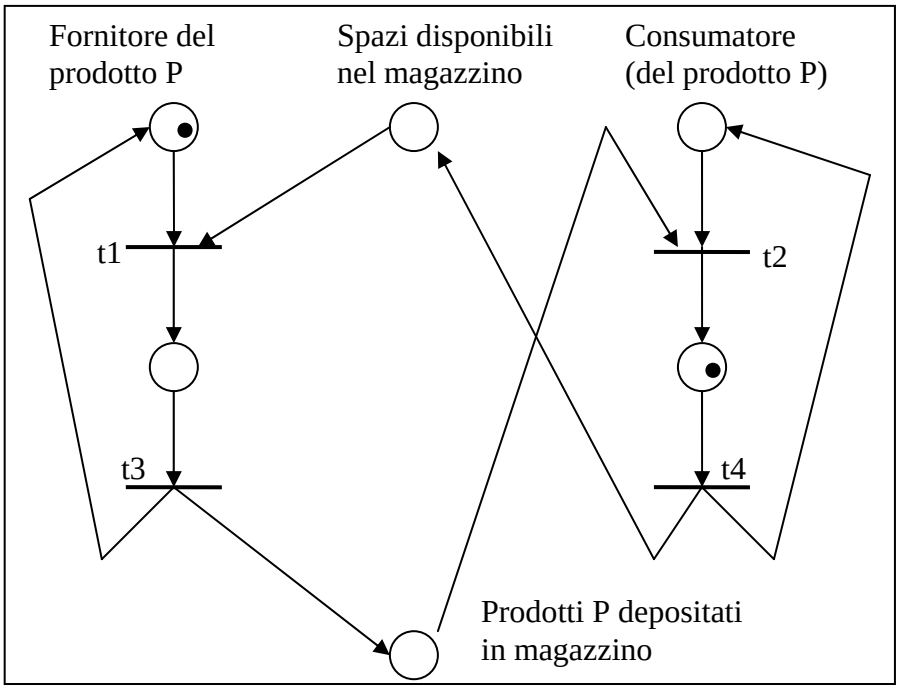

Figura 11 - Rete dopo l'attivazione della transizione t2: il consumatore sta consumando

L'ultima transizione ad essere eseguita è ora t4. Il consumatore ha consumato (ha acquistato) il prodotto P, rendendo libero uno spazio nel magazzino e si pone in attesa di consumare ancora (**figura 12**). La rete è adesso tornata allo stato iniziale ed il ciclo di transizioni si ripete.

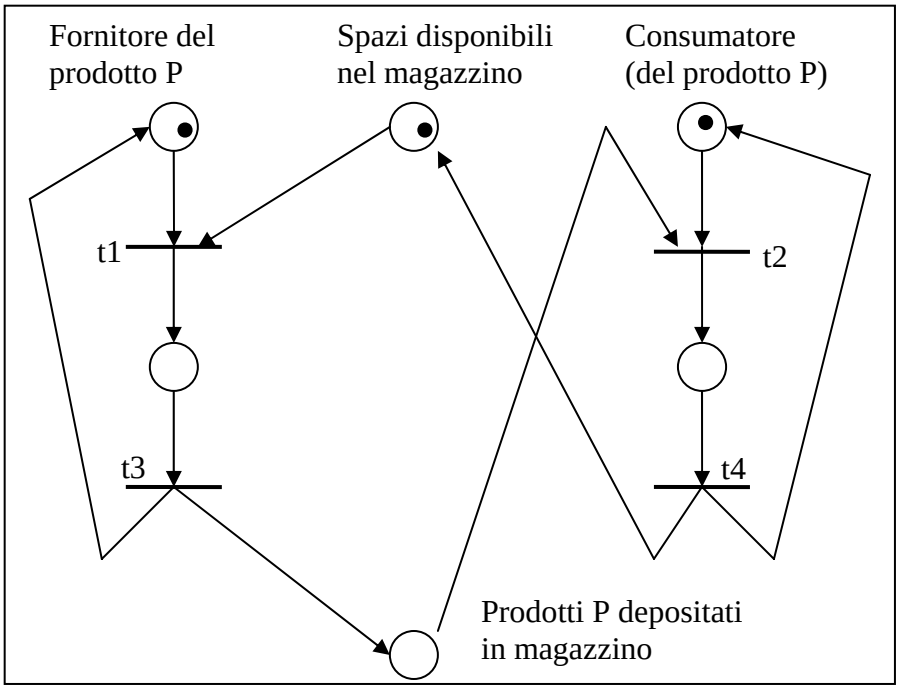

Figura 12 - Rete dopo l'attivazione della transizione t4: si ritorna allo stato iniziale

Nell'esercizio proposto abbiamo assunto che il fornitore produca un solo prodotto P alla volta ed analogamente per il consumatore. Con piccoli accorgimenti sulle marcature è possibile generalizzare la rete in modo da descrivere produzioni di partite di *n* prodotti P e consumi (richieste di acquisto) di *k* prodotti P.