UNIVERSITÀ DELLA CALABRIA
il Campus per eccellenza

Deep Modeling

Mario Alviano

Google Forms and Reading

- Rispondete al modulo online su CIA

  https://forms.gle/ePz4eAsDgu8EQeqw9

- Capire il concetto di controllo statico del codice

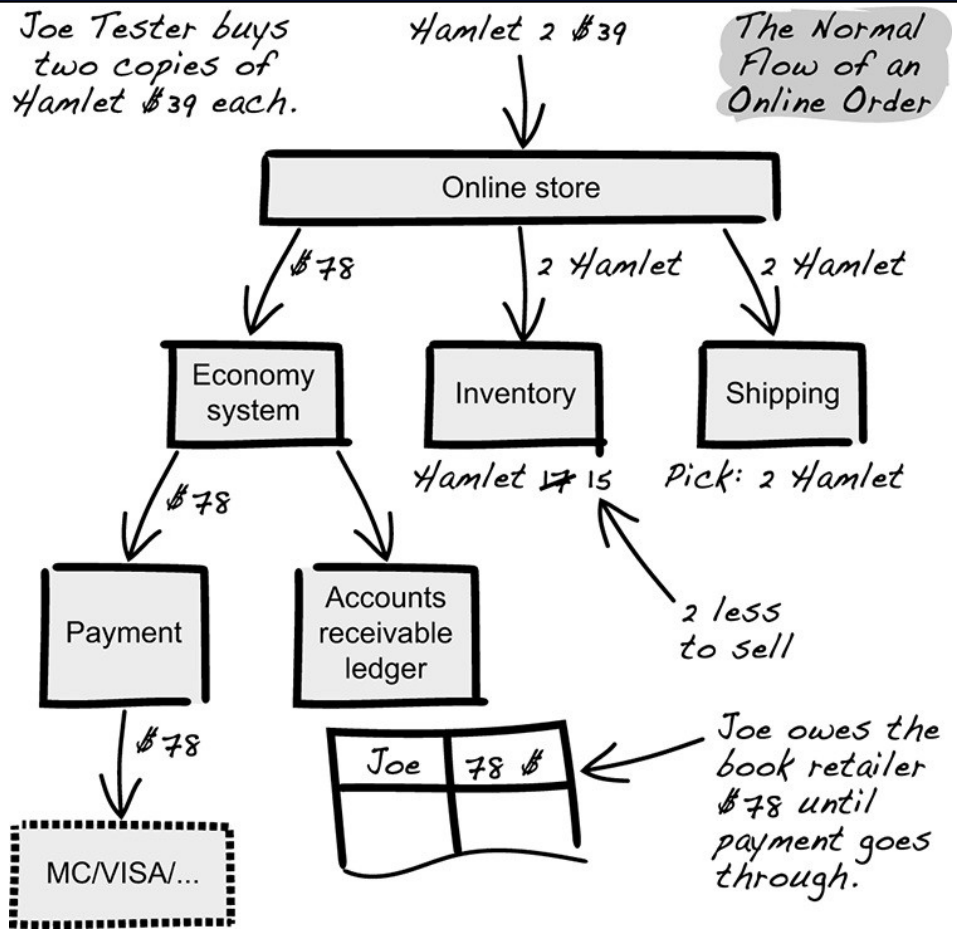
  http://web.mit.edu/6.031/www/fa18/classes/01-static-checking/

  (sezioni **Static Typing**, **Static Checking, Dynamic Checking, No Checking** e **Surprise: Primitive Types Are Not True Numbers**, including reading exercises)

- Rispondete al modulo online sul problema del teatro

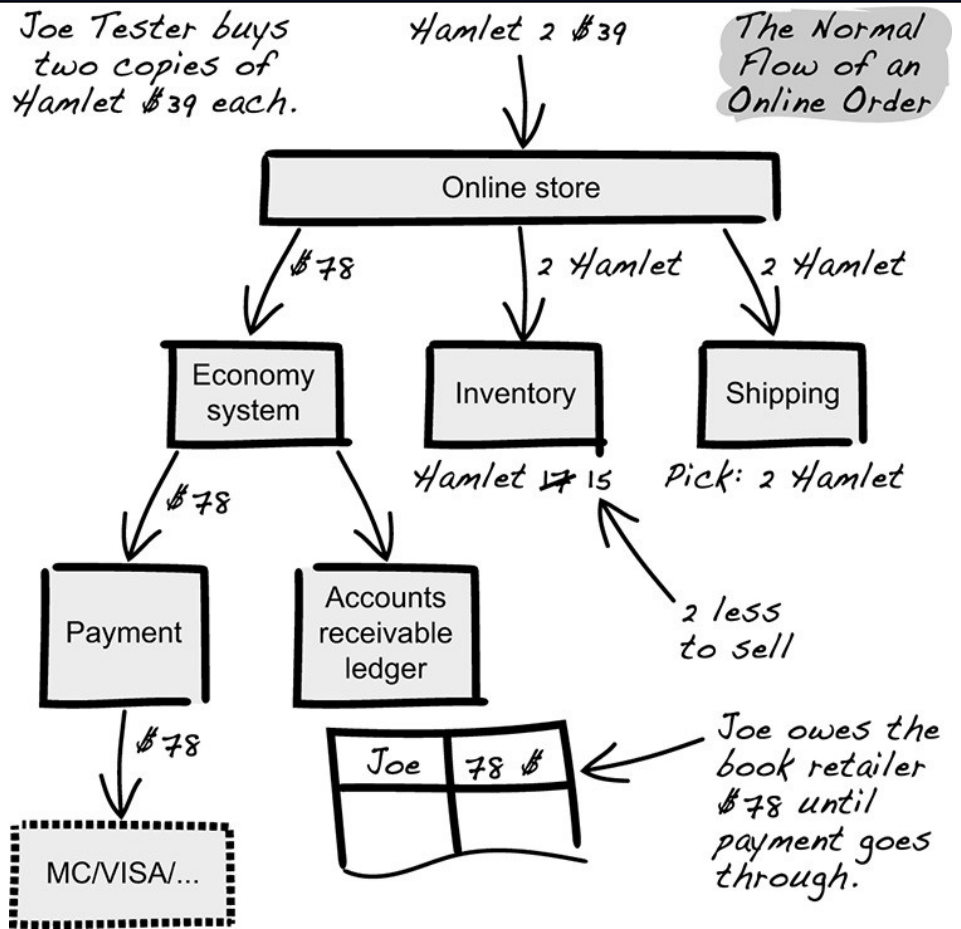
  https://forms.gle/JYBYJsSEpusV1X5h7

Caso di studio. Comprerò -1 libro

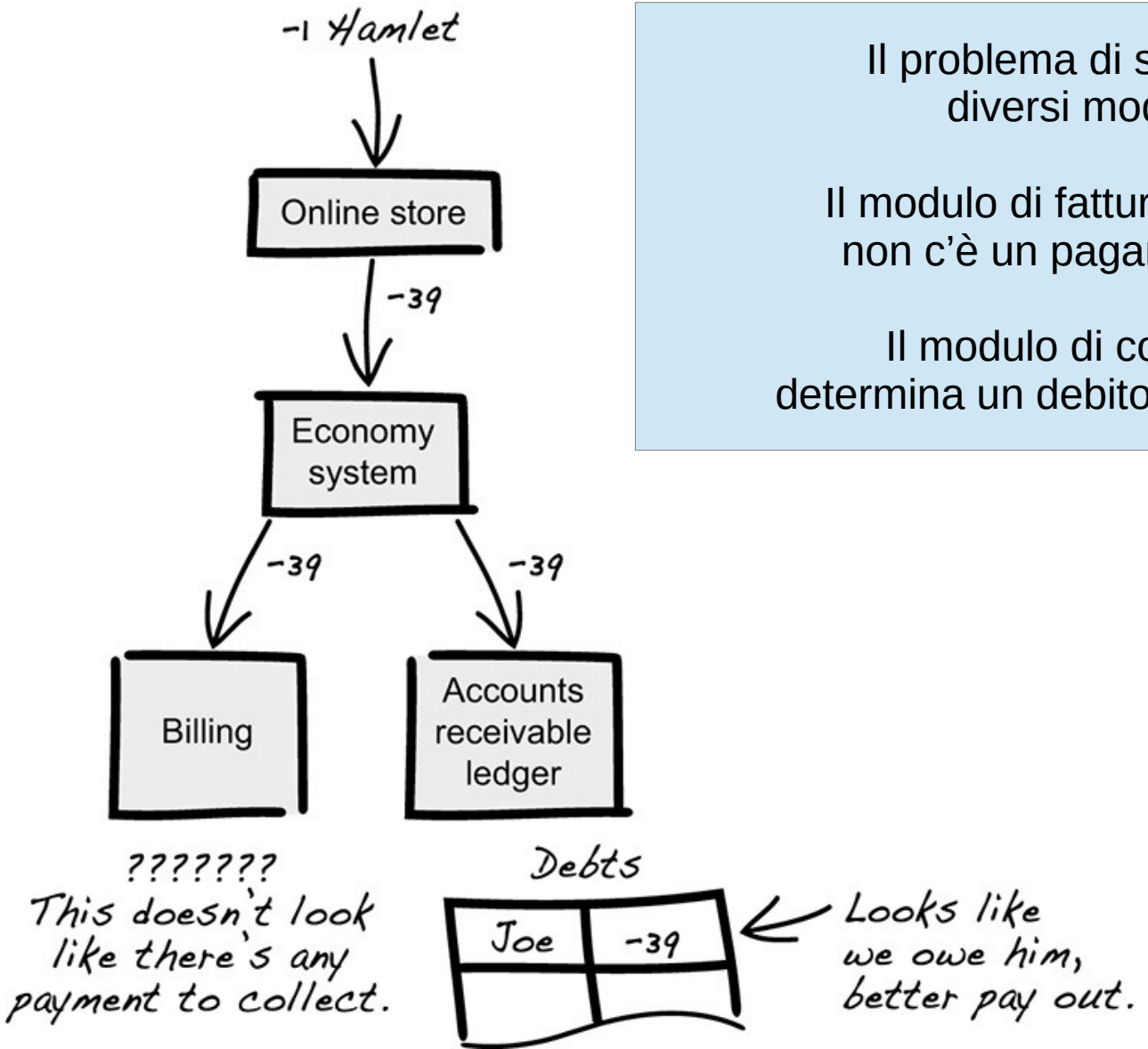


- Il capitolo 2 del libro Secure by Design descrive un problema di sicurezza di un negozio online di libri
- Problema individuato durante un controllo di sicurezza
- Diversi check superati: pacchetti malevoli verso il web server bloccati, nessuna porta anomala aperta, session id protetti, configurazioni corrette

Caso di studio. Comprerò -1 libro



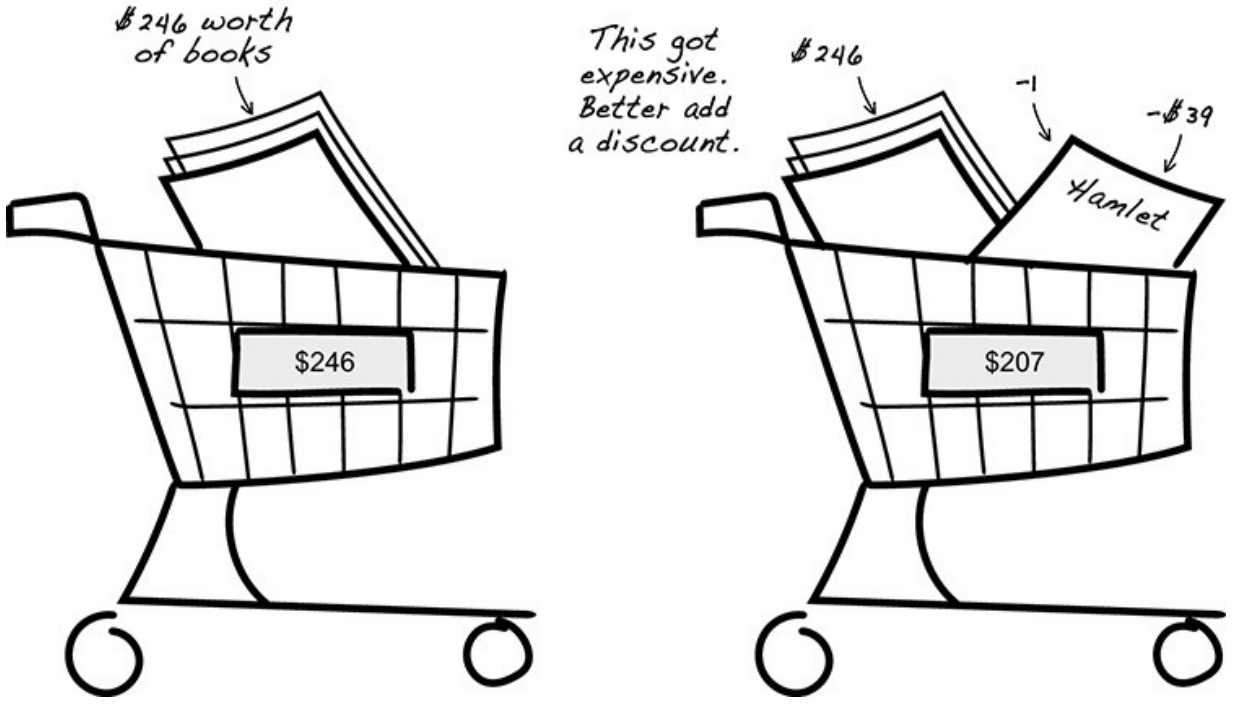
- Analisi del campo **quantità**
  - Provo codice JavaScript, non eseguito (no XSS)
  - Provo SQL Injection, niente
  - Provo -1, ordine accettato
- Il giorno dopo qualcuno dalla contabilità chiede chiarimenti
  - Il sistema ha emesso una nota di credito di $39
  - **Niente di strano per noi**, ma è indirizzato verso un membro del team di sicurezza



Il problema di sicurezza attraversa diversi moduli del sistema

Il modulo di fatturazione stabilisce che non c'è un pagamento da pretendere

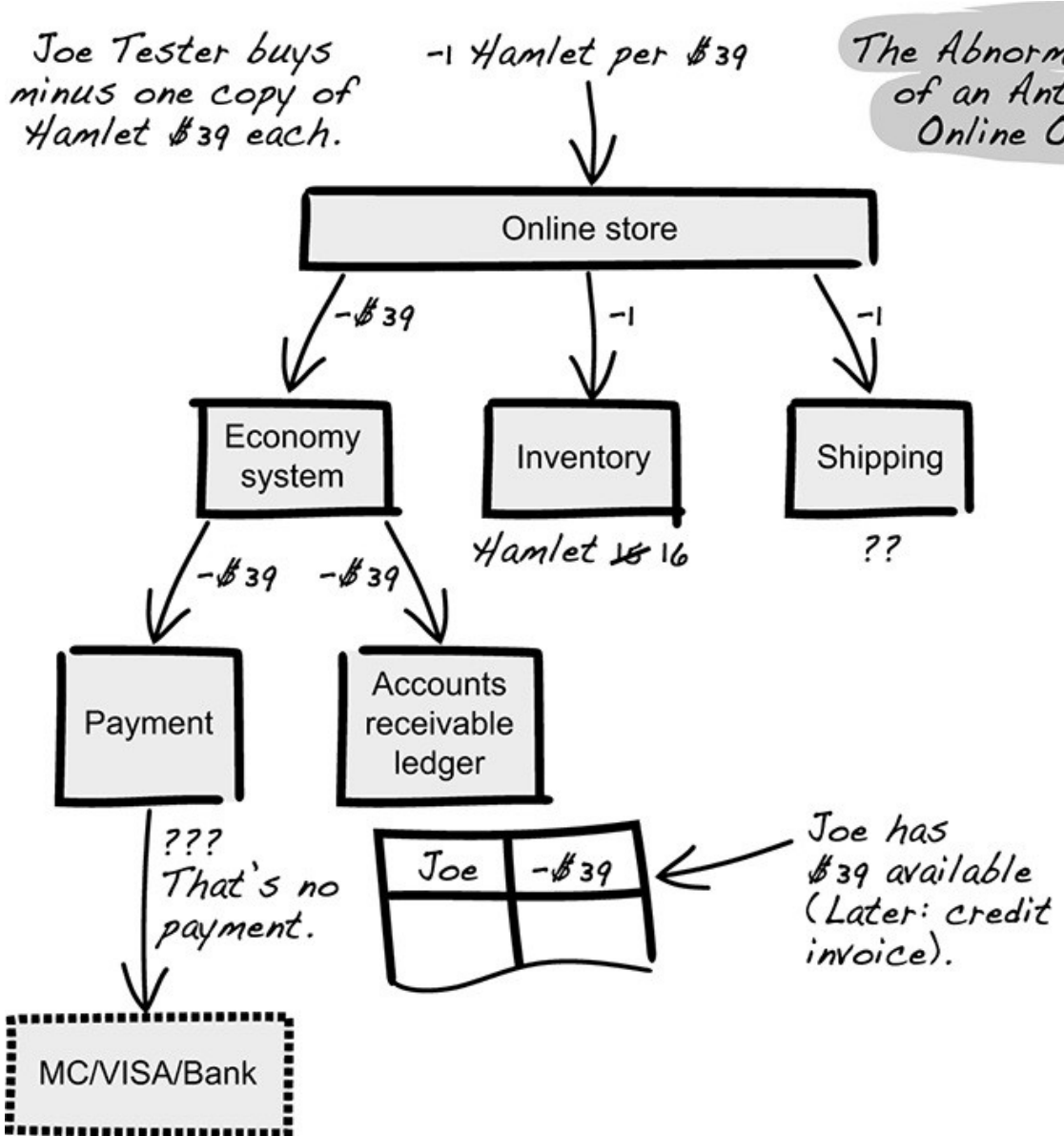
Il modulo di contabilità dei clienti determina un debito da saldare al più presto



Il bug di sicurezza può essere difficile da individuare

Se compro libri per $246 e aggiungo -1 libro da $39, pagherò $207

- Nel caso di studio questo sembra avvenisse di frequente
- Ulteriore investigazione rivela che il problema coinvolge altri sistemi dell’azienda



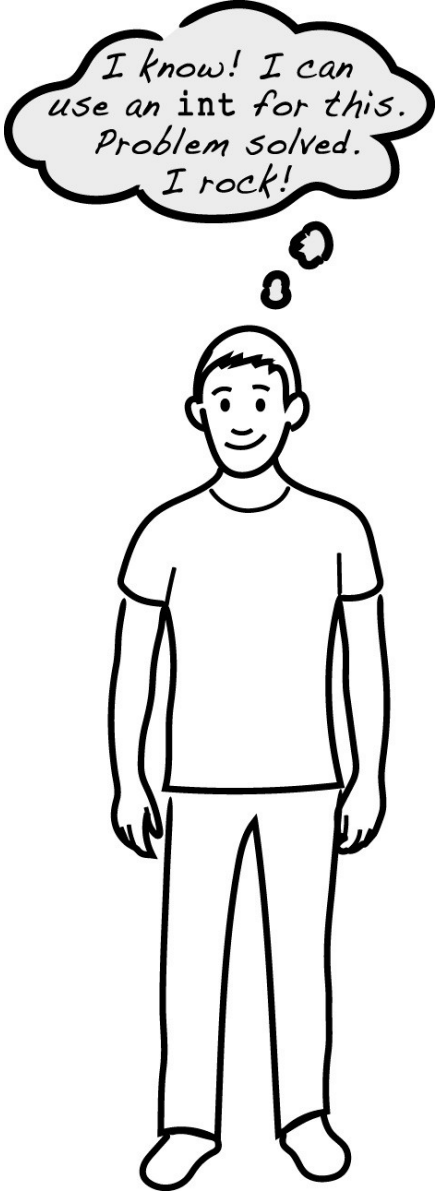
Comprare -1 libro causa
un incremento nell'inventario:
la rappresentazione informatica
dell'inventario diventa inconsistente

Comprare -1 libro causa
l'invio di -1 libro:
il sistema ignora la richiesta,
nessuno controlla il log

Le inconsistenze si compensano
e non vengono notate,
comportando perdite consistenti
per l'azienda

Progettazione superficiale

- Come può essere stato introdotto il bug?
- Comprare una quantità negativa?!?
- Eppure questo è stato reso possibile
- Lo sviluppatore non ha prestato attenzione
- Ha usato una soluzione semplice
- che però non modella bene il dominio!



Il modello superficiale (1)

- Discussione fra il venditore Sal e lo sviluppatore Deve

"And then you can add books to the order," says Sal.

"So, how do we describe a book?" questions Deve.

"We show a title and a price," answers Sal.

"What can the price be? Is it always a whole number?" asks Deve.

"Well no, a book can be priced $19.50, tax excluded," clarifies Sal.

Deve thinks, "So a book has a title and a price as attributes. The title is a string. The price is not an int, it's a float."

ATTENZIONE!

Mai usare float o double per rappresentare valute o altre quantità precise!

Il modello superficiale (2)

And Deve asks, “Is that all there is to a book?”

“Nah,” answers Sal, “it’s also important that it has an ISBN, so we can keep hardbacks and paperbacks separate.”

“OK. And then we add books to the order,” says Deve. “For example, *Moby Dick*, *Pride and Prejudice*, *Hamlet*, *Moby Dick* again, *1984*, and *Moby Dick* again?”

“Well, almost. We would say three *Moby Dick* books, as we don’t care about what order you buy them in.”

“OK,” Deve thinks. “It isn’t a float, it’s an integer.”

ATTENZIONE!

Mai fermare
la modellazione a
“È un intero”!

Il modello superficiale (3)

```
class Book {
    String title;
    String isbn;
    double price;
    ...
}

class Order {
    void addOrderLine(Book book, int quantity) {
    ...
    }
}
```

ATTENZIONE!

Deve non ha approfondito il concetto di titolo e ISBN, usa String

Ha fatto una domanda interessante sul prezzo: “È sempre un numero intero?”
Non ha però capito che è un concetto complesso: “… tasse escluse.”

Posso rappresentarlo in codice?
vs
Ho capito come funziona?

Gli errori di Deve

- Il tipo **int** può rappresentare numeri interi fra -2 miliardi e 2 miliardi. È una buona rappresentazione di *quantità*? Di altro che abbia senso?
- Ha senso che il titolo possa essere qualsiasi stringa?
- Ha senso che l'ISBN possa essere qualsiasi stringa?
- Deve ha perso anche l'opportunità di un controllo statico del codice: **title** e **isbn** hanno lo stesso tipo, **String**

```
void addCust(String name, String phone, String fax, int
creditStatus,
   int vipLevel, String contact, String contactPhone,
boolean partner)
```

Un modello più profondo (1)

"And then you can add books to the order," says Sal.

"So, how do we describe a book?" questions Deve.

"We show a title and a price," answers Sal.

"What can the price be? Is it always a whole number?" asks Deve.

"Well no, a book can be priced $19.50, tax excluded," clarifies Sal.

Deve thinks, "So, a book has title and price as attributes. And price seems to be a complicated issue in itself because you mentioned tax. I'll need to dive into those later," and asks, "Is that all there is to a book?"

Identifica concetti complessi

Prendi nota e approfondisci prima di codificare

Un modello più profondo (2)

asks, “Is that all there is to a book?”

“Nah,” answers Sal, “it’s also important that it has an ISBN so we keep hardbacks and paperbacks separate.”

“OK. And then we add books to the order,” says Deve. “For example, *Moby Dick*, *Pride and Prejudice*, *Hamlet*, *Moby Dick* again, *1984*, and *Moby Dick* again?”

“Well, almost. We would say you have a quantity of three *Moby Dick* books, as we don’t care about what order you buy them in,” says Sal.

“Can you buy half a *Moby Dick*?” asks Deve.

“Of course not, silly.”

I libri sono in numero intero

Il venditore usa il termine **quantità**

Identifica i concetti del **dominio** e usali per approfondire l’analisi

Un modello più profondo (3)

"You used the word quantity," Deve says. "I want to understand that better. What happens if you have a quantity of three *Moby Dick* books and then they're removed? Do you then have a quantity of zero *Moby Dick* books?"

"Eehhh, not really. I mean, a quantity of zero isn't really a quantity at all. We'd say no quantity," Sal clarifies.

"This quantity seems to have some rules around it," Deve says. "So, how big can a quantity be? Two billion books?"

"Haha. Well, certainly not. Seriously, I think we're limited by the through-store logistics flow, and it can't handle orders bigger than a total quantity of 240."

Abbiamo identificato un estremo per il tipo quantità: 1

Chiediamo dell'estremo superiore

Se il venditore usa nuovi termini, chiediamo chiarimenti. Potrebbe essere un altro concetto del dominio da modellare

Un modello più profondo (4)

"The through-store flow?" asks Deve.

"Yep, that's what they call it. It's how the orders from the online store are handled at the warehouse; it's about box sizes, packing stations, and stuff. Orders bigger than that must go to the warehouse bulk flow. But we can't use that from the online store," Sal explains.

"What is the total quantity of an order? Can you give me an example?"

"That's simply adding the quantity of all books. If you have three *Hamlet*s, four *Pride and Prejudice*s, and one *Moby Dick*, then you have a total quantity of eight," Sal says.

La singola quantità non può eccedere 240

Simile per la quantità totale

```
class Book {    ①
    BookTitle title;

    ISBN isbn;
    Money price;
    ...
}

class Quantity {    ②

  ...
  Quantity(int quantityOfBooks) {
      isTrue(0 < quantityOfBooks, "Quantity must be
positive");
      isTrue(quantityOfBooks <= 240,
        "Quantity must fit in through-store flow, which is
limited to 240");
      ...
  }
}

class Order {
    void addOrderLine(Book book,
                      Quantity quantity) {    ③

        ...
    }

    Quantity totalQuantity() {
        ...
    }
}
```

Definisci tipi specifici

BookTitle, ISBN, Money

Ogni tipo viene validato durante la creazione (e in caso di modifica, se consentita)

Il compilatore garantisce che i dati sono validi

Non è possibile ordinare -1 libro

Troppe classi?

- Ogni concetto del dominio dovrebbe essere rappresentato con una classe
- La classe è responsabile della validazione dei dati
- Codifica la conoscenza associata al concetto
- Non usare una classe implica ripetere lo stesso codice di validazione in più punti del software
  - Bugs everywhere!

Reading exercise

- Alcuni principi generali di buona programmazione

  http://web.mit.edu/6.031/www/fa18/classes/04-code-review/

Questions