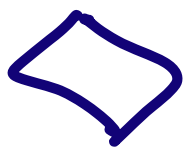


△ --- - überbrue

△ ————— credito


△ --- - DIPENDENZA

WQS-01



AGGREGAZIONE

qui evita insieme per
creare oggetto complesso

DA AGGREGATO A INSERIRE CON 
VICINO AD AGGREGATO

CAR  - evita



ENGINE

part-8



composition
& cancellata
& cancella

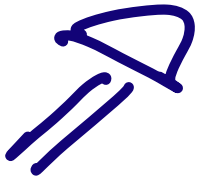
Person  - core



cerchia



TCP IO LIB MyReader MsgConsumer
MySender



QUESTA NON SI PREOCCUPA PIÙ

Reader Protocol Manager → ^{la deve sempre} riscuotere
questa classe "consume" msg in
arrivo e riconosce i comandi
consume message per interfaccia

↳ qui uso IMessageConsumer
che ha un metodo per ogni comando
da consumare

App è CommandCenter
ha metodi per elaborare msg
usa SenderProtocolHandler per
scrivere il msg di risposta
(secondo protocollo) e
lento con MySender

SERVER → crea istanza di
CLIENT
tutto a partire
da socket

In Java 7

Introduce interfaccia `CommandConsumer`
con metodi per comuni protocolli
Introduce `RPM` e metodo `consumeResponse`
che processa msg con uno `attachCar`
e richiama i metodi `execute` da
`consumer`

Il nuovo App che implementa i metodi
di `CommandConsumer` e elabora
le msg di risposta. L'idea con
SPM

Il server / client può essere per
classi con un'unica politica

→ si cambia politica per attività

Header è thread

while (!~~exp~~) {

log e → con RPH

}

