

Ingegneria del Software I

Unified Modelling Language

Behavior Diagrams:
Interaction Diagram:
Sequence and Collaboration Diagrams
State Diagram
Activity Diagram

1

Modelli Dinamici

- ◆ I modelli dinamici descrivono il comportamento del sistema in funzione del tempo
 - I modelli dinamici sono un tipo di modello operativo (modello che descrive il comportamento desiderato)
 - Utili soprattutto per sistemi orientati al controllo (embedded systems, sistemi interattivi, traduttori, sistemi operativi, software di comunicazione)
- ◆ Modelli dinamici in UML
 - Diagrammi di interazione
 - » Diagrammi di sequenza
 - » Diagrammi di collaborazione
 - Diagrammi di stato
 - Diagrammi di attività

2

Interaction Diagrams

- Un Interaction Diagram modella le interazioni che sussistono tra un certo numero di oggetti che interagiscono per risolvere un problema
- Una **interazione** è un comportamento che comprende un insieme di messaggi scambiati tra un insieme di oggetti nell'ambito di un contesto per raggiungere uno scopo

Esistono due differenti tipi di Interaction Diagram:

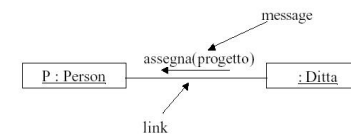
- Sequence diagrams
- Collaboration diagram

Entrambi descrivono dinamicamente le caratteristiche di un sistema software, evidenziando però le interazioni (scambio di messaggi) ed il loro ordinamento temporale anche se con enfasi diversa.

3

Interaction

- ◆ Una interaction, tipicamente, avviene tra oggetti tra cui esiste un link (esiste un'istanza di una association)
- ◆ Un *message* è una specificazione di una comunicazione tra oggetti che trasmette informazione con l'aspettativa che ne conseguirà una attività. La *ricezione di un message* può essere considerata una istanza di un evento.



4

Behavior Diagrams

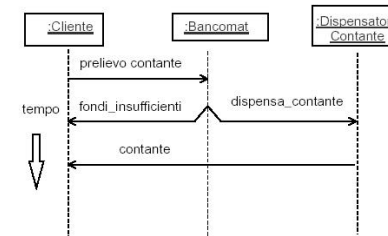
Ingegneria del Software I

Sequence Diagram

- ◆ Evidenziano la sequenza temporale delle azioni
- ◆ Non si vedono le associazioni tra oggetti
- ◆ Le attività svolte dagli oggetti sono mostrate su linee verticali
- ◆ La sequenza dei messaggi scambiati tra gli oggetti è mostrata su linee orizzontali
- ◆ Possono corrispondere a uno scenario specifico o a un intero caso d'uso (aggiungendo salti e iterazioni)
- ◆ Si possono annotare con vincoli temporali

5

Sequence Diagram: caso d'uso "prelievo contante"



6

Sequence Diagram

- Gli elementi tipici di un Sequence Diagram sono gli oggetti ed i messaggi attraverso cui essi interagiscono.
- Lo scambio di messaggi è rappresentato da frecce labellate.
- I messaggi possono esplicitamente far riferimento ai metodi (operazioni) effettivamente coinvolti; il livello di dettaglio può essere tale da specificare anche i parametri.
- Presenza di elementi di controllo quali ripetizioni cicliche e condizioni.

7

Messaggi ed azioni

- ◆ Ad un message possono corrispondere diversi tipi di azioni; in UML sono previste le seguenti azioni di base:
 - Call : invoca una operation di un object; un oggetto può inviare un messaggio a se stesso, invocando una propria operation
 - Return : restituisce un valore al chiamante
 - Send : invia un signal ad un oggetto
 - Create : crea un object
 - Destroy : distrugge un oggetto; un oggetto può distruggere se stesso
- ◆ I message scambiati tra due o più oggetti possono formare una sequenza; una sequenza di message deve avere un punto di inizio
- ◆ Per meglio visualizzare l'ordine sequenziale dello scambio dei message è possibile 'numerare' i message antepoendo al loro nome un numero che indica l'ordine nella sequenza.

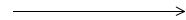
8

Behavior Diagrams

Ingegneria del Software I

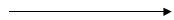
Tipi di messaggi

Semplice: rappresenta un flat flow of control; il controllo è passato dal mittente al ricevente

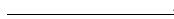


Sincrono: rappresenta un nested flow of control; il controllo è passato dal mittente al ricevente ed il mittente aspetta che il ricevente gli restituisca il controllo

- possono generarsi sequenze innestate di messaggi: il ricevente invia un altro messaggio ad un altro oggetto

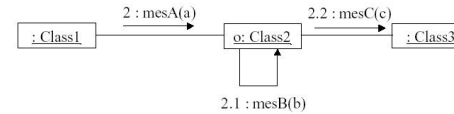


Asincrono: rappresenta un non-nested flow of control tramite un signal; il mittente 'segnala' il ricevente tramite il message e continua senza aspettare il ricevente, che può o meno ritornare informazioni

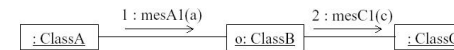


9

Tipi di messaggi



Nested flow of control - Sequenze innestate di messaggi synchronous

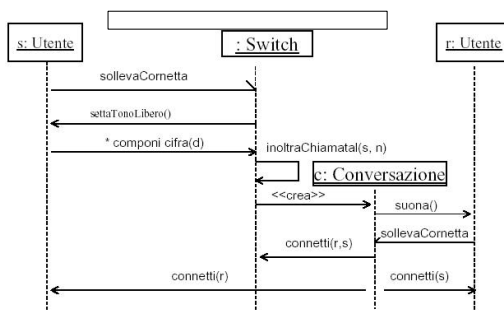


Flat flow of control - sequenze di messaggi semplici non innestati

10

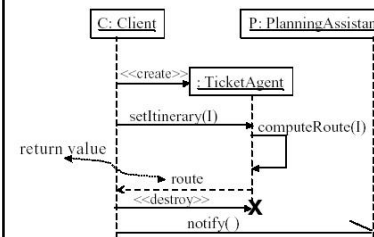
Sequence Diagram:

scenario "chiamata effettuata con successo" del caso d'uso "effettua chiamata interna"



11

Creazione e distruzione di oggetti



Lifeline di un oggetto: linea tratteggiata rappresentante l'esistenza (vita) di un oggetto in un periodo di tempo

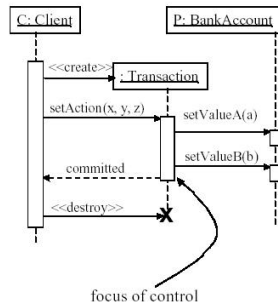
in cima vanno gli oggetti che vi partecipano per tutta la durata, gli altri che sono creati durante la interaction vanno posti all'altezza della action di creazione

12

Behavior Diagrams

Ingegneria del Software I

Focus of control



- ◆ Indica il periodo di tempo durante il quale un oggetto sta eseguendo una action, direttamente o indirettamente
- ◆ sottili rettangoli (posti sulle lifeline) indicanti il periodo di tempo durante il quale un oggetto sta eseguendo una action, direttamente o indirettamente
- ◆ la cima del rettangolo è allineata con lo start della action (ricezione del message); il fondo è allineato con il completamento della action, ed eventualmente con un message di ritorno.

13

Coordinatore e controllori

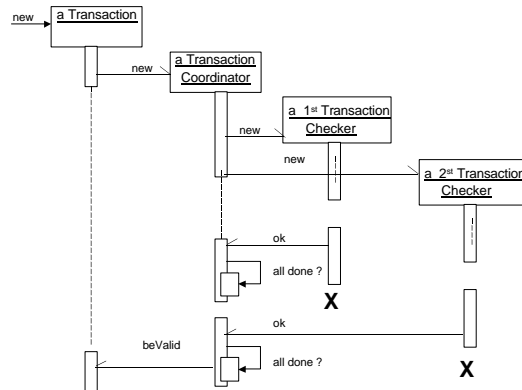
I Sequence Diagram si rivelano piuttosto efficaci anche per descrivere processi concorrenti.

... una Transazione per essere portata a termine deve far riferimento ad un Coordinatore, che crea due controllori che effettuano una serie di controlli relativi ai dati coinvolti dalla Transazione.

I controlli effettuati dai controllori sono affidati a processi che operano in maniera indipendente e concorrente. Se tali controlli hanno esito positivo la Transazione è valida

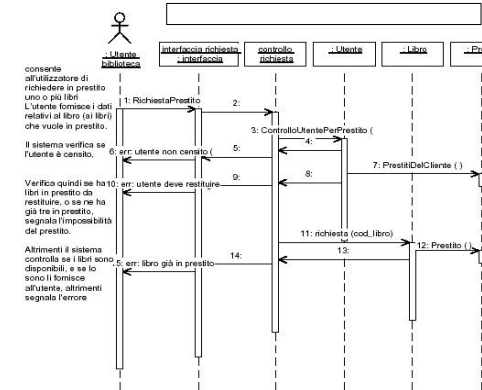
14

Coordinatore e controllori



15

Sequence Diagram: Esempio



È possibile accompagnare con una descrizione testuale per migliorare la leggibilità

16

Behavior Diagrams

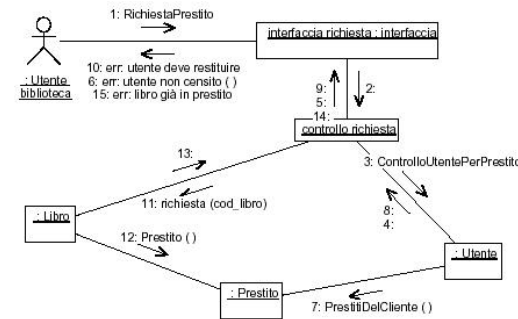
Ingegneria del Software I

Collaboration Diagram

- ◆ Enfatizza l'organizzazione strutturale degli oggetti che spediscono e ricevono messaggi
- ◆ Specifica gli oggetti che collaborano tra loro in un dato scenario, ed i messaggi che si indirizzano
- ◆ La sequenza dei messaggi è meno evidente che nel diagramma di sequenza, mentre sono più evidenti i legami tra gli oggetti
 - Per meglio visualizzare l'ordine sequenziale dello scambio dei messaggi è possibile 'numerare' i message antepoendo al loro nome un numero che indica l'ordine nella sequenza
- ◆ Può essere utilizzato in fasi diverse (analisi, disegno di dettaglio), e rappresentare diverse tipologie di oggetti
- ◆ Diagrammi di sequenza e diagrammi di collaborazione esprimono informazioni simili, ma le evidenziano in modo diverso
- ◆ Diagrammi di sequenza e diagrammi di collaborazione sono isomorfi, è possibile cioè trasformare uno nell'altro

17

Esempio di Collaboration Diagram



18

State Diagram

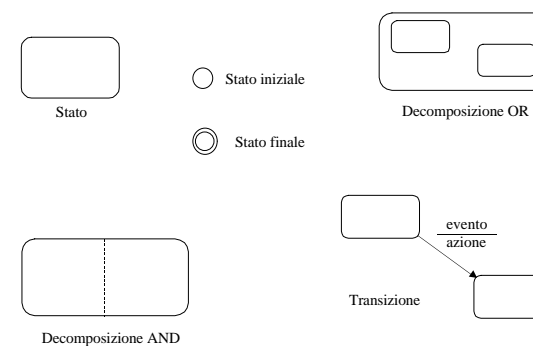
Un sistema software orientato agli oggetti è costituito da una molteplicità di oggetti le cui azioni ed interazioni sono funzionali all'assolvimento delle responsabilità del sistema.

Ognuno degli oggetti è caratterizzato da uno stato che evolve nel tempo, uno State Diagram consente di rappresentare

- gli *stati* di un oggetto, specifica il ciclo di vita di un oggetto
- il comportamento dei singoli oggetti in termini di
 - Eventi a cui gli oggetti (la classe) sono sensibili
 - Azioni prodotte
 - Transizioni di stato
 - » Identificazione degli stati interni degli oggetti le *transizioni di stato* dell'oggetto e gli eventi che le hanno provocate.

19

Elementi grafici



20

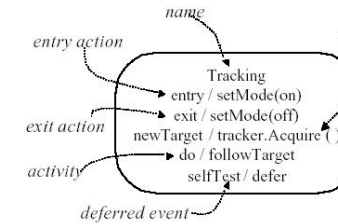
Behavior Diagrams

Ingegneria del Software I

Stato

- ◆ Situazione in cui l'oggetto soddisfa qualche condizione, esegue qualche attività o aspetta qualche condizione
- ◆ Attributi (opzionali):
 - **Nome**: una stringa; uno stato può essere anonimo
 - **Entry / Exit actions**: eseguite all'ingresso / uscita dallo stato (non interrompibili, durata istantanea)
 - **Transizioni interne**: non causano un cambiamento di stato
 - **Attività dello stato** (interrompibile, durata significativa)
 - **Eventi differiti**: non sono gestiti in quello stato ma posposti ed accordati per essere gestiti da altri oggetti in un altro stato
 - **Sottostati**: struttura innestata di stati; disgiunti (sequenzialmente attivi) o concorrenti (concorrentemente attivi)

Stato completo



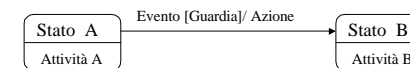
22

Transizioni

- ◆ Cambiamento da uno stato iniziale a uno finale
 - transizione esterna (stato finale diverso da stato iniziale)
 - interna (stato finale uguale a stato iniziale)
- ◆ Attributi (opzionali): evento [guardia] / azione
 - **Evento**
 - » segnale, messaggio da altri oggetti, passaggio del tempo, cambiamento
 - **Condizione di guardia**
 - » La transizione occorre se l'evento accade e la condizione di guardia è vera
 - **Azione**
 - » E' eseguita durante la transizione e non è interrompibile (durata istantanea)
 - **Transizioni senza eventi (triggerless) scattano**
 - » con guardia: se la condizione di guardia diventa vera
 - » senza guardia: se l'attività interna allo stato iniziale è completata

23

Come si rappresenta



Attività: il processo (o i processi) che l'oggetto svolge quando è in un certo stato.

Un'attività può prendere un tempo più o meno lungo e può essere interrotta da un evento (Event).

Evento: ciò che potenzialmente scatena una transizione.

Guardia: una condizione logica che può essere vera o falsa.

Azione: il processo (o i processi) che accompagna una transizione.

Evento, Guardia ed Azione sono comunque elementi opzionali

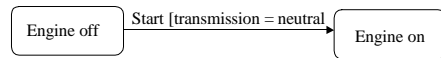
24

Behavior Diagrams

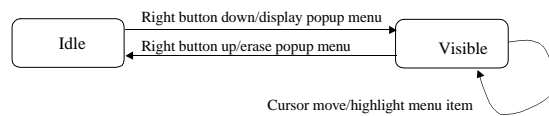
Ingegneria del Software I

Esempi

Engine



Menu



25

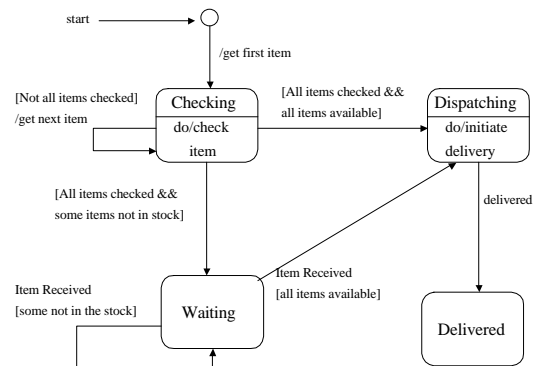
Descrizione di un esempio

Supponiamo di dover modellare il comportamento di un oggetto ORDINE che deve:

- 1) controllare la disponibilità di tutte le voci presenti in un certo ordine di acquisto;
- 2) nel caso in cui tutte le voci sono disponibili deve inviarle al cliente che le ha ordinate.
- 3) nel caso in cui alcune voci non sono presenti in magazzino in quantità sufficiente deve ritardare l'invio fino a quando anche tali voci non siano reperite.

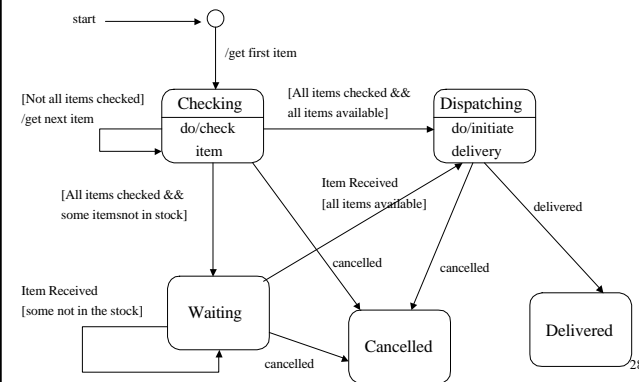
26

Esempio ORDINE



27

Esempio ORDINE



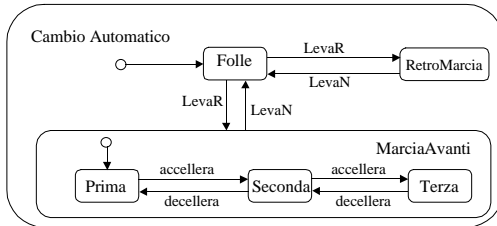
28

Behavior Diagrams

Ingegneria del Software I

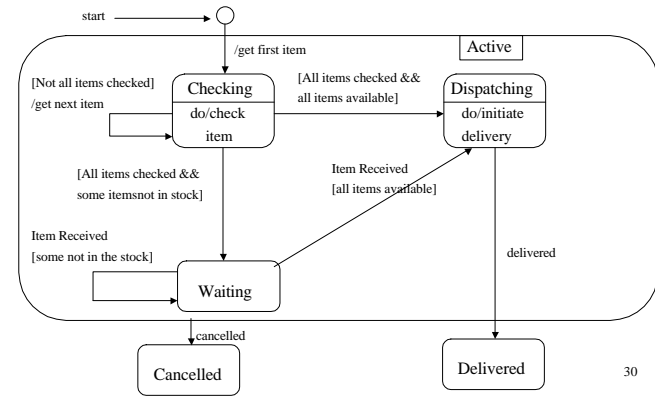
Sottostati sequenziali (decomposizione OR)

- ◆ Un macro stato equivale ad una scomposizione OR degli stati
 - I sottostati ereditano le transizioni dei loro superstati



29

Esempio ORDINE



30

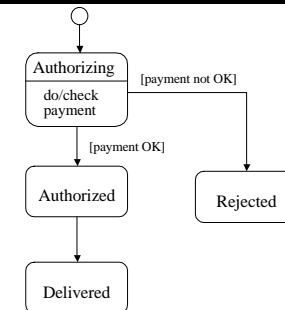
Concurrent substates (decomposizione AND)

- ◆ Più state machine sono eseguite in parallelo entro lo state che li racchiude
- ◆ Se un substate machine raggiunge lo stato finale prima dell'altro, il controllo aspetta lo stato finale dell'altro
- ◆ Quando avviene una transizione in uno stato con concurrent substate, il flusso di controllo subisce un fork per ciascun concurrent substate; alla fine esso si ricompone in un unico flusso con uno join

31

Esempio ORDINE

Oltre a dover tener conto della disponibilità della merce l'oggetto ORDINE deve anche verificare che il cliente assolva il pagamento. Se così non fosse l'ordine va rigettato. Operazioni in parallelo.

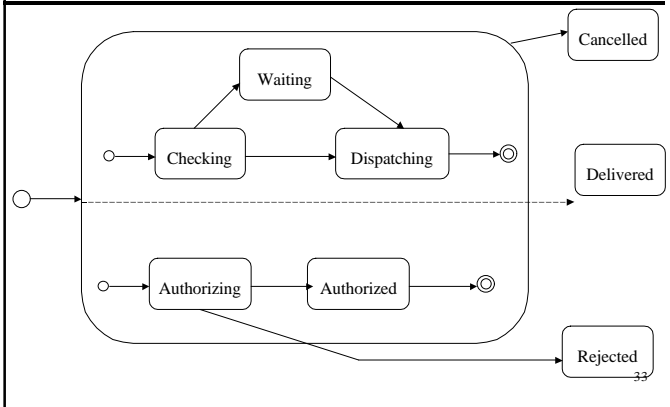


32

Behavior Diagrams

Ingegneria del Software I

Esempio ORDINE



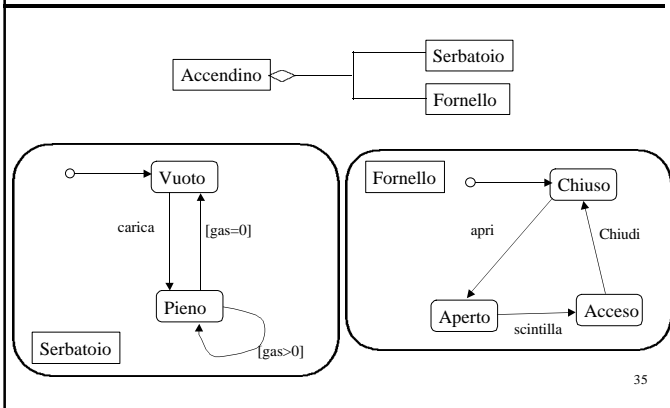
Concurrent substates

Attenzione: All'interno della sezione concorrente l'oggetto ordine in un dato istante è "in due stati" (situazione merce e situazione pagamento), all'esterno è sempre e solo in un unico stato (cancelled, delivered o rejected) .

Gli State Diagrams consentono di descrivere il comportamento di un oggetto attraverso differenti Use Case, pertanto sono da considerarsi complementari agli Interaction Diagram.

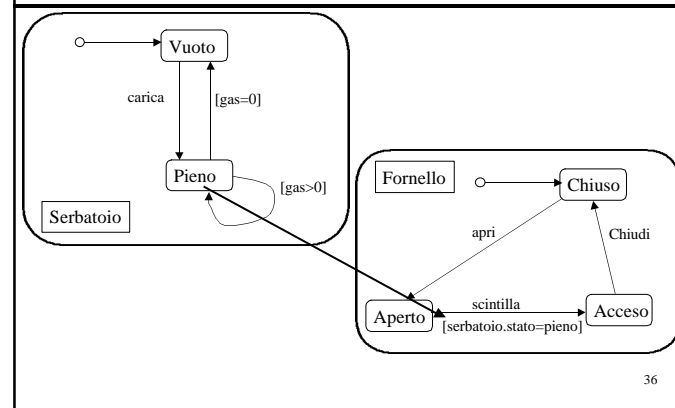
34

Oggetti composti



35

Interazione tra componenti



36

Behavior Diagrams

Ingegneria del Software I

Activity Diagrams

- Consentono di descrivere le attività (ed il flusso loro relativo) caratteristiche dell'applicazione che si sta sviluppando
- Forniscono la sequenza di operazioni che definiscono un'attività più complessa
- Permettono di rappresentare processi paralleli e la loro sincronizzazione
- Possono essere considerati State Diagram particolari in cui ogni stato è un'attività
- Un Activity Diagram può essere associato
 - A una classe
 - All'implementazione di un'operazione
 - Ad uno Use Case
- Retaggio della scomposizione "funzionale"

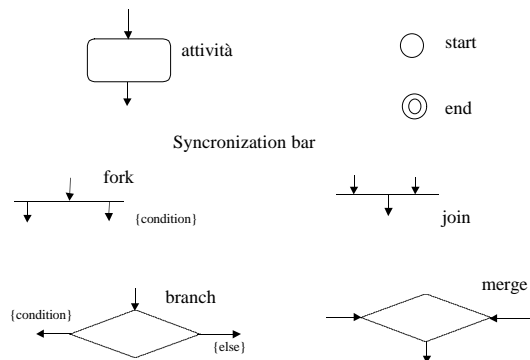
37

Activity Diagrams

- ◆ Derivano da event diagrams, SDL e reti di Petri
- ◆ Servono a rappresentare sistemi di workflow, oppure la logica interna di un processo di qualunque livello
- ◆ Utili per modellare
 - comportamenti sequenziali
 - non determinismo
 - concorrenza
 - sistemi distribuiti
 - business workflow
 - operazioni
- ◆ Sono ammessi stereotipi per rappresentare le azioni con i costrutti SDL

38

Elementi Grafici



39

Activity Diagram: Elementi

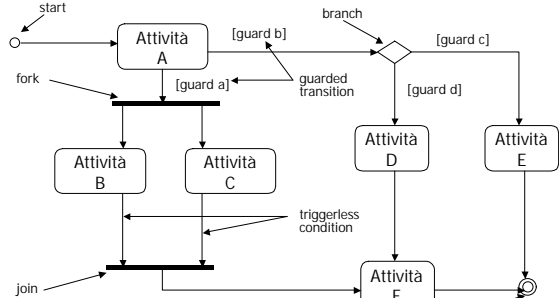
- ◆ **Activity:** un'esecuzione non atomica entro uno state machine
 - Una activity è composta da action, elaborazioni atomiche comportanti un cambiamento di stato del sistema o il ritorno di un valore
 - ◆ **Transition:** flusso di controllo tra due action successive
 - ◆ **Guard expression:** espressione booleana (condition) che deve essere verificata per attivare una transition
 - ◆ **Branch:** specifica percorsi alternativi in base a espressioni booleane; un branch ha una unica transition in ingresso e due o più transition in uscita
 - ◆ **Synchronization bar:** usata per sincronizzare flussi concorrenti
 - *fork:* per splittare un flusso su più transition verso action state concorrenti
 - *join:* per unificare più transition da più action state concorrenti in una sola
- » il numero di fork e di join dovrebbero essere bilanciati

40

Behavior Diagrams

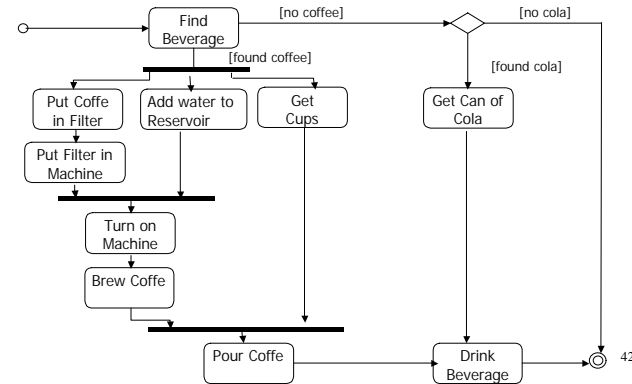
Ingegneria del Software I

Activity diagram



41

Activity Diagram: un esempio



42

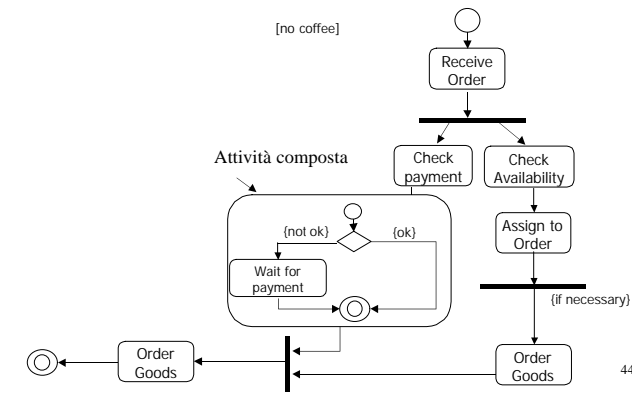
Action e Activity State

- *Action state*: azioni eseguibili atomiche; gli action states non possono essere decomposti né interrotti; l'esecuzione è intesa essere effettuata in in tempo insignificante
 - una action state può essere considerato come un caso particolare di activity state;
- *Activity state*: stati non atomici, e quindi decomponibili ed interrompibili
 - può essere a sua volta rappresentato con un activity diagram
 - stessa rappresentazione grafica di action state, con eventuali parti aggiuntive (es. entry ed exit action)

Do Building
entry/ setLock ()

43

Esempio: Gestione Ordini



44

Behavior Diagrams

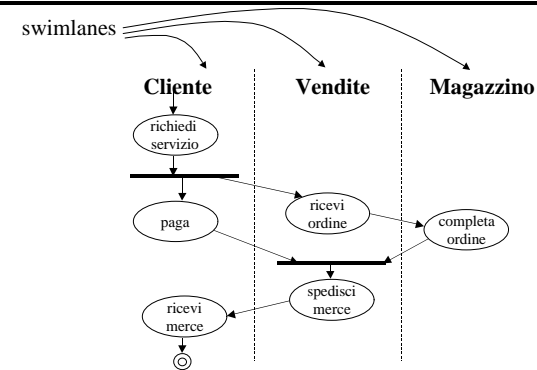
Ingegneria del Software I

Swimlanes

- costruito grafico rappresentante un insieme partizionato insieme di actions;
 - sono usati per modellare le responsabilità di uno o più oggetti rispetto all'attività
- per ogni oggetto responsabile di action/activity nel diagramma è definito uno swimlane, identificato da un nome univoco nel diagramma
 - le action/activity state sono divise in gruppi e ciascun gruppo è assegnato allo swimlane dell'oggetto responsabile per esse
 - l'ordine con cui gli swimlane si succedono non ha alcuna importanza
 - le transition possono attraversare swimlane per raggiungere uno state in uno swimlane non adiacente a quello di start della transition

45

Swimlanes: esempio

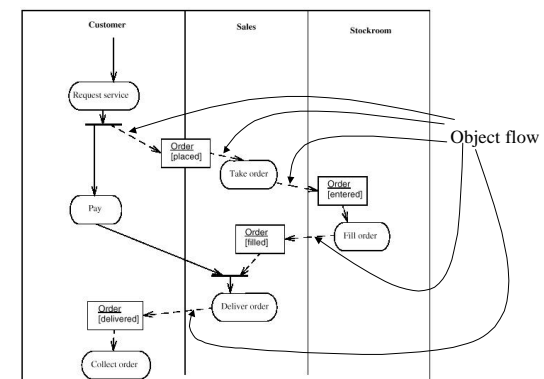


46

Attività e flussi di oggetti (Object flow)

- Gli object flow sono coinvolti nel flusso di controllo delle action/activity state
- gli object flow sono associations tra action/activity state e object; modellano l'utilizzo di object da parte di action/activity state e l'influenza di queste su essi
- gli objects possono essere
 - output di una action: la action crea lo object, la freccia della relationship punta lo object
 - input di una action: questa usa lo object, la freccia della relationship punta la action
 - manipolati da qualsiasi numero di action: lo output di una action può essere lo input di un'altra
 - presenti più volte nello stesso diagramma: ogni presenza indica un differente punto della vita dello object
- può essere rappresentato lo stato di un object indicandolo tra [] al di sotto del nome dello object

Object Flow: Esempio



48

Behavior Diagrams

Ingegneria del Software I

Activity Diagram e Use Case

L'applicazione più significativa degli Activity Diagram è relativa agli Use Case. In particolare è possibile evidenziare sia le azioni che devono essere intraprese nell'ambito di un singolo Use Case che tra differenti Use Case.

49

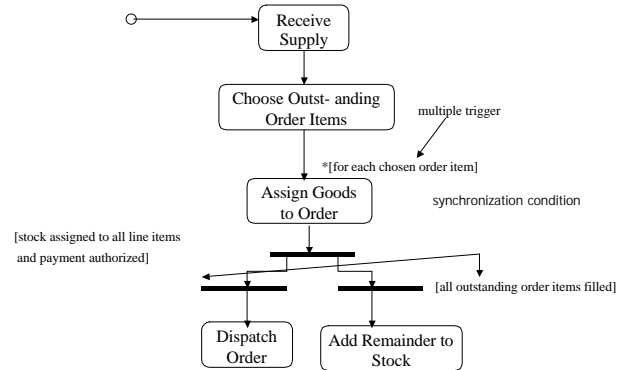
Activity Diagram e Use Case: Esempio

Un possibile brano di use case

... quando un fornitore consegna un quantitativo di merce è necessario verificare gli ordini bloccati e decidere quali di questi possono essere sbloccati dal nuovo quantitativo di merce. Quindi effettuiamo le assegnazioni della nuova merce agli ordini bloccati e la merce rimanente viene sistemata in magazzino ...

50

Activity Diagram e Use Case: Esempio



51

Behavior Diagrams