LEZIONE 26
RDD
&&
ANALISI OBJECT ORIENTED

Ingegneria del Software e Progettazione Web

Università degli Studi di Tor Vergata - Roma

Guglielmo De Angelis
guglielmo.deangelis@isti.cnr.it

RDD

- un modo di pensare alla progettazione O.O.
  - **R**esponsibility **D**riven **D**esign
- le classi e/o le istanze in un sistema software sono associati ad una descrizione delle loro responsabilità
  - sono assegnate dal progettista
  - obblighi sulla struttura o sul comportamento in relazione al suo ruolo all'interno del sistema
    - obblighi sulla struttura o sul comportamento
    - responsabilità di fare
    - responsabilità di conoscere
    - responsabilità di collaborare

… uno scenario tipico:

- come individuare le classi di analisi?
- esistono tecniche a supporto di questa attività?
- le classe di analisi hanno tutte/sempre la stessa funzione?

… uno scenario tipico:

- come individuare le classi di analisi?
- esistono tecniche a supporto di questa attività? **SI!**
- le classe di analisi hanno tutte/sempre la stessa funzione? **Ovviamente NO!**

… uno scenario tipico:

- come individuare le classi di analisi?
- esistono tecniche a supporto di questa attività? **SI!**
- le classe di analisi hanno tutte/sempre la stessa funzione? **Ovviamente NO!**

RUP, ad esempio, suggerisce agli analisti di distinguere le classi di analisi in 3 macro categorie

- **boundary**
- **control**
- **entity**

PARENTESI : perché si ha bisogno di modellare aspetti non previsti dal linguaggio ?!?!

- principali motivazioni :
  - il dominio di applicazione o il committente hanno bisogno di esprimere concetti (sufficientemente) generali che non sono nativamente inclusi nel linguaggio di modellazione (i.e. UML)
  - il modellista ha bisogno di rappresentare aspetti legati allo specifico processo di sviluppo adottato
    - dando differenti interpretazioni ad uno stesso elemento del linguaggio in base alle fasi del processo
  - il modellista vuole raffinare i modelli esistenti in funzione del contesto
    - risolvendo eventuali ambiguità
    - per una generazione del codice più efficiente/efficacie

PARENTESI : stereotipi

- gli stereotipi sono il principale meccanismo per la personalizzazione di UML
- uno stereotipo estende il significato di un elemento del linguaggio (meta-classe)
  - il risultato è un nuovo elemento nel linguaggio
  - l'elemento del linguaggio dove è possibile applicare uno stereotipo è chiamato “base class”
- attraverso la definizione di uno stereotipo possono essere definiti
  - ulteriori interpretazioni per un elemento
  - ulteriori caratteristiche di un elemento
  - ulteriori relazioni con altri elementi

BCE : quali responsabilità?

- **boundary**: mediano l'interazione tra il biente
  - rappresentano elementi al conf
  - regolano l'interazione
- **control**: coordi uso del sistema
  - on il sistema
  - q dominio del problema
- **entit** chiave del sistema
  - sono te di esecuzione
  - dipen di dominio del problema
    - gloss se case, business model, etc.
  - modellano il comportamento di una entità di dominio incapsulando un insieme coeso di dati

definire e marcare le classi di analisi utilizzando gli stereotipi

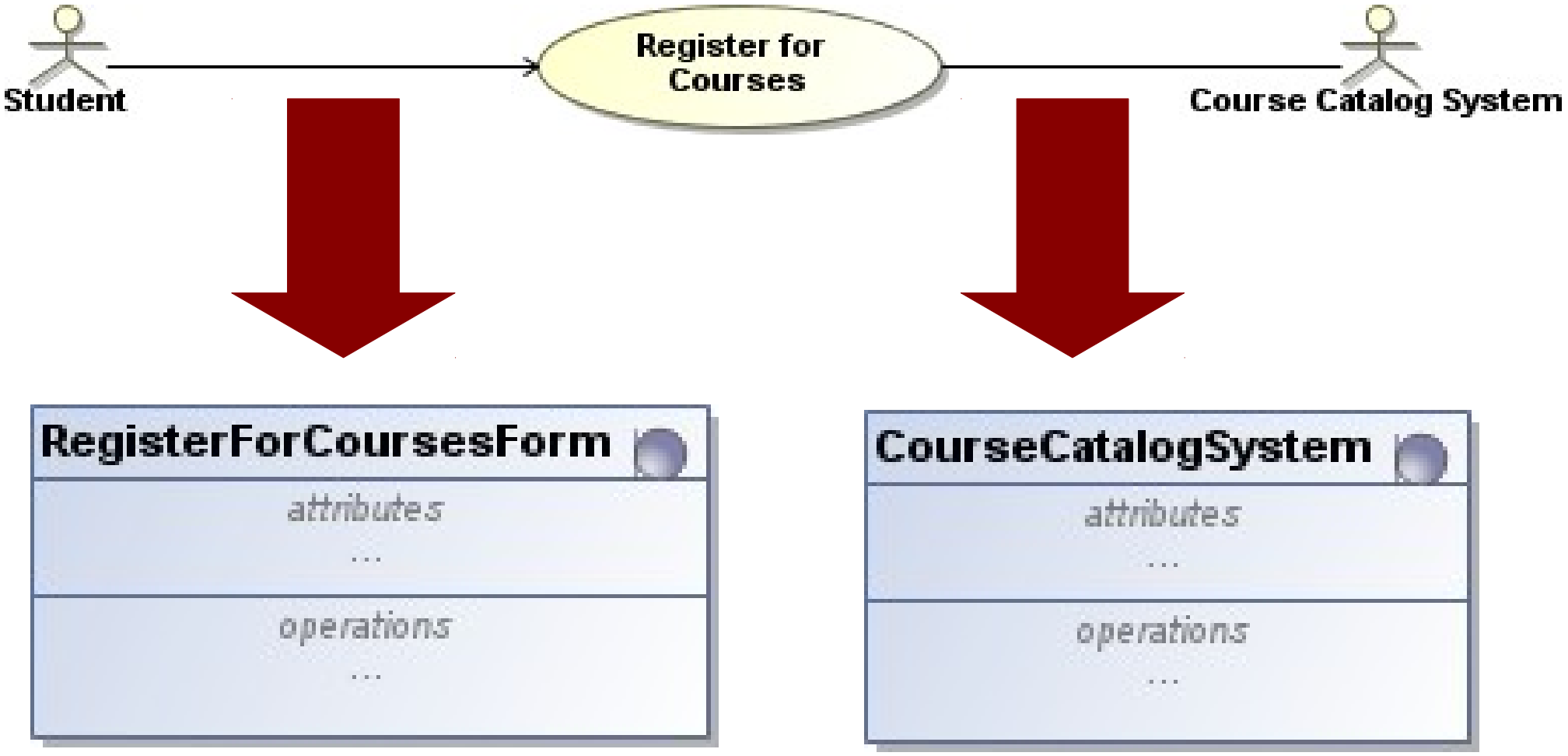
- <<boundary>>
- <<control>>
- <<entity>>

BCE : quali responsabilità?

- **boundary**: mediano l'interazione tra il sistema e l'ambiente
  - rappresentano elementi al confine del sistema
  - regolano l'interazione con gli attori
- **control**: coordinano il comportamento durante un caso d'uso del sistema
  - rappresentano comportamenti dipendenti dall'interazione attesa con il sistema (i.e. comportamento descritto dal caso d'uso)
  - sono indipendenti dal modo di attivazione dell'interazione
  - queste classi dovrebbero scaturire dall'analisi dominio del problema
- **entity**: rappresentano le astrazioni chiave del sistema
  - sono indipendenti dall'ambiente di esecuzione
  - dipendono dall'analisi dominio del problema
    - glossario, use case, business model, etc.
  - modellano il comportamento di una entità di dominio incapsulando un insieme coeso di dati

BCE – boundary

BCE – boundary



BCE – boundary

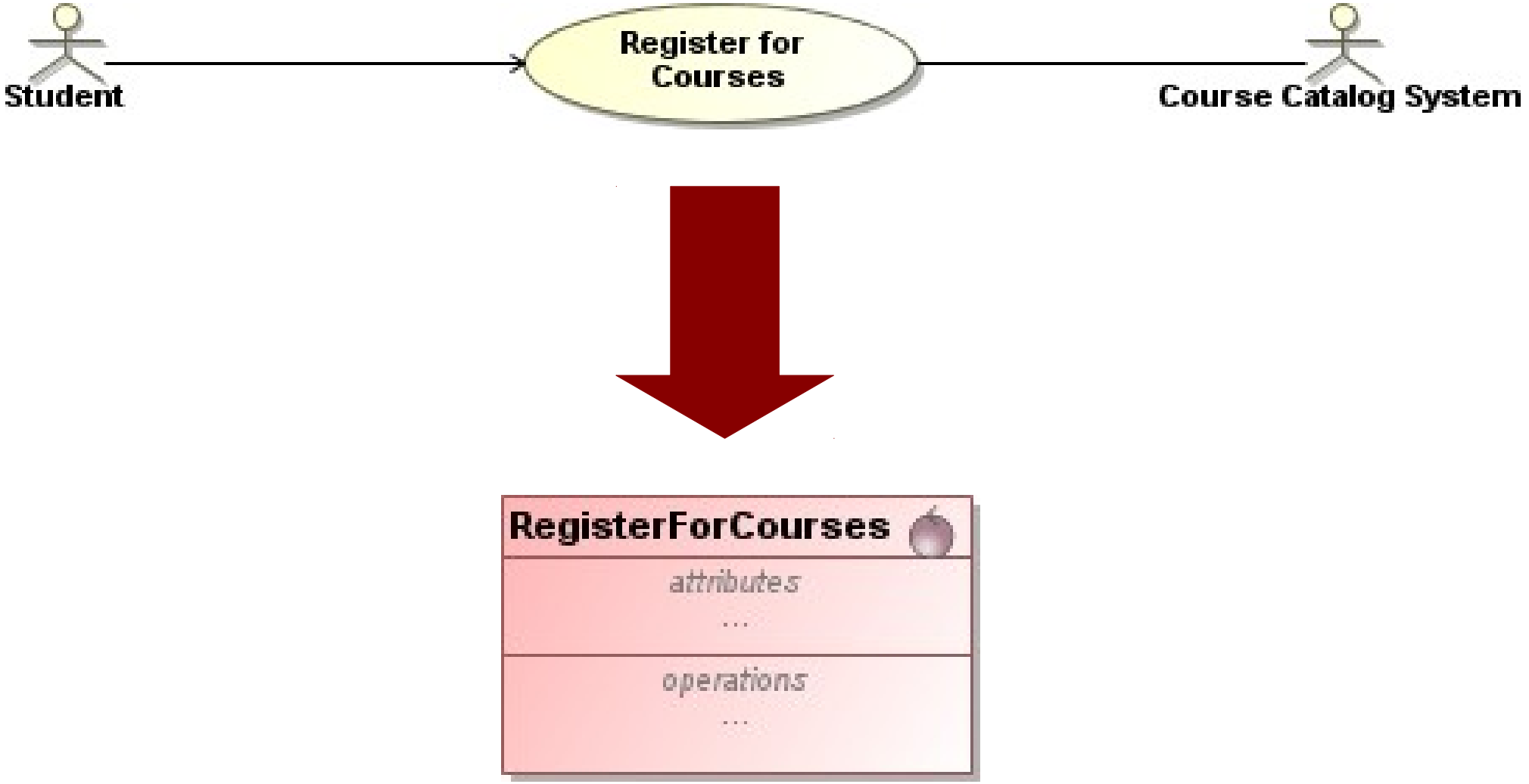
- enfatizzare le informazioni e le operazioni necessarie per l'interazione con l'attore
- NON concentrarsi su dettagli implementativi della GUI

BCE : quali responsabilità?

- **boundary**: mediano l'interazione tra il sistema e l'ambiente
  - rappresentano elementi al confine del sistema
  - regolano l'interazione con gli attori
- **control**: coordinano il comportamento durante un caso d'uso del sistema
  - rappresentano comportamenti dipendenti dall'interazione attesa con il sistema (i.e. comportamento descritto dal caso d'uso)
  - sono indipendenti dal modo di attivazione dell'interazione
  - queste classi dovrebbero scaturire dall'analisi dominio del problema
- **entity**: rappresentano le astrazioni chiave del sistema
  - sono indipendenti dall'ambiente di esecuzione
  - dipendono dall'analisi dominio del problema
    - glossario, use case, business model, etc.
  - modellano il comportamento di una entità di dominio incapsulando un insieme coeso di dati

BCE – control

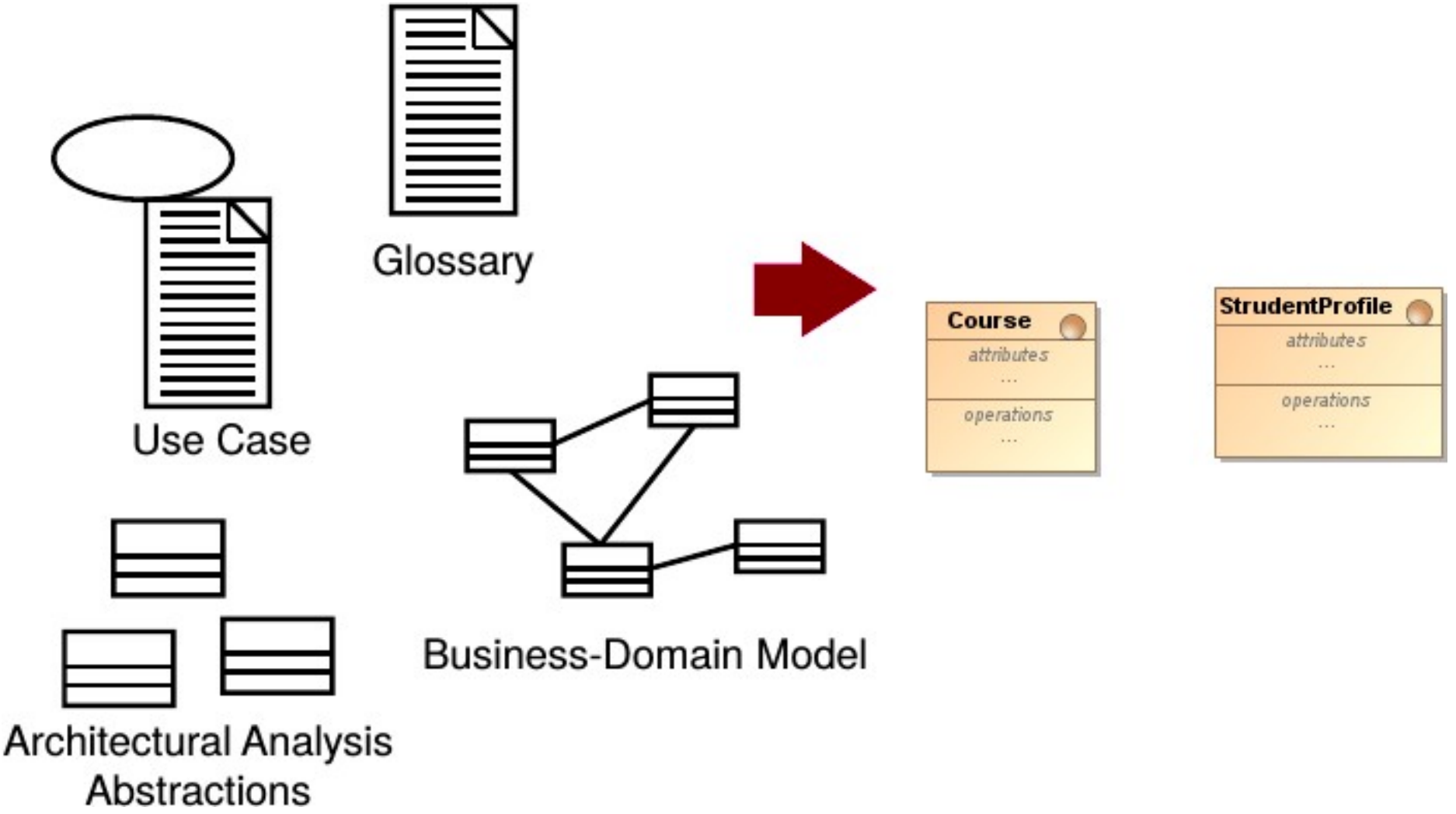
BCE – control


Student
Register for Courses
Course Catalog System
RegisterForCourses
attributes
...
operations
...

BCE : quali responsabilità?

- **boundary**: mediano l'interazione tra il sistema e l'ambiente
  - rappresentano elementi al confine del sistema
  - regolano l'interazione con gli attori
- **control**: coordinano il comportamento durante un caso d'uso del sistema
  - rappresentano comportamenti dipendenti dall'interazione attesa con il sistema (i.e. comportamento descritto dal caso d'uso)
  - sono indipendenti dal modo di attivazione dell'interazione
  - queste classi dovrebbero scaturire dall'analisi dominio del problema
- **entity**: rappresentano le astrazioni chiave del sistema
  - sono indipendenti dall'ambiente di esecuzione
  - dipendono dall'analisi dominio del problema
    - glossario, use case, business model, etc.
  - modellano il comportamento di una entità di dominio incapsulando un insieme coeso di dati

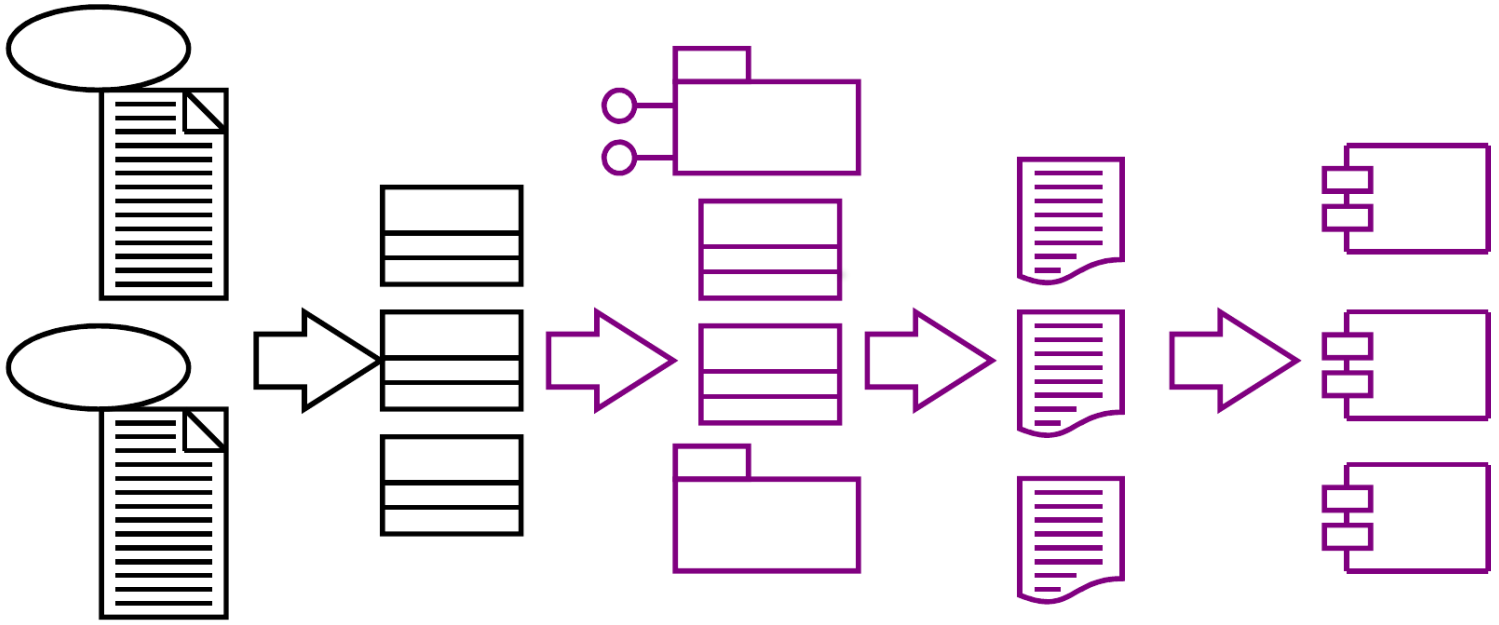
BCE – entity


Glossary
Use Case
Business-Domain Model
Architectural Analysis
Abstractions
Course
attributes
...
operations
...
StrudentProfile
attributes
...
operations
...

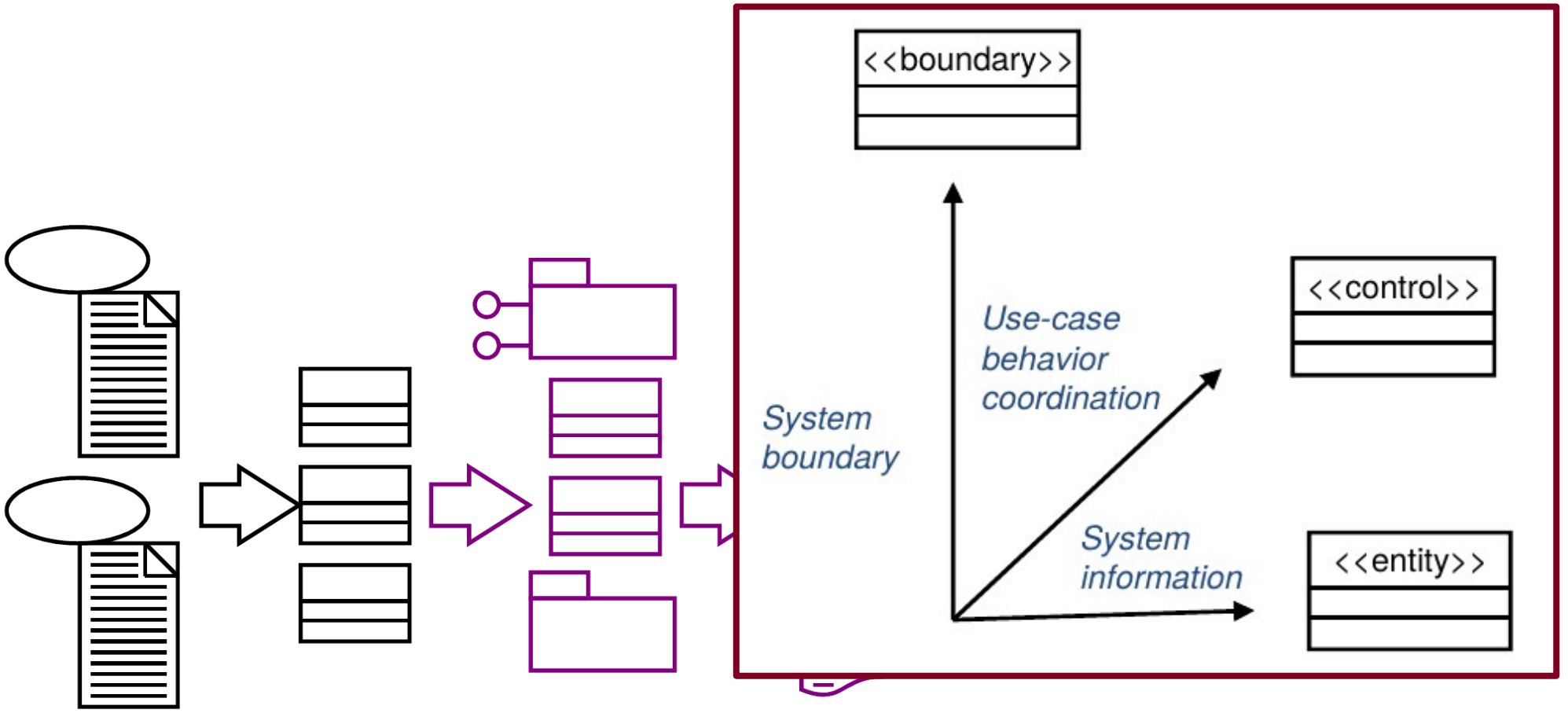
BCE : quali responsabilità?

- **boundary**: mediano l'interazione tra il sistema e l'ambiente
  - rappresentano elementi al confine del sistema
  - regolano l'interazione con gli attori
- **control**: coordinano il comportamento durante un caso d'uso del sistema
  - rappresentano comportamenti dipendenti dall'interazione attesa con il sistema (i.e. comportamento descritto dal caso d'uso)
  - sono indipendenti dal modo di attivazione dell'interazione
  - queste classi dovrebbero scaturire dall'analisi dominio del problema
- **entity**: rappresentano le astrazioni chiave del sistema
  - sono indipendenti dall'ambiente di esecuzione
  - dipendono dall'analisi dominio del problema
    - glossario, use case, business model, etc.
  - modellano il comportamento di una entità di dominio incapsulando un insieme coeso di dati

artefatti SW e BCE



artefatti SW e BCE



… in conclusione

- il pattern di analisi BCE consente di affrontare il seguente schema di problema
  - Quali classi partecipano ad un caso d'uso?
  - Quante sono le classi di analisi per un caso d'uso?

- lo schema di soluzione proposto viene identificato con il termine **VOPC**: View of Participating Classes
  - vista della classi partecipanti ad ogni caso d'uso

… uno scenario di esempio

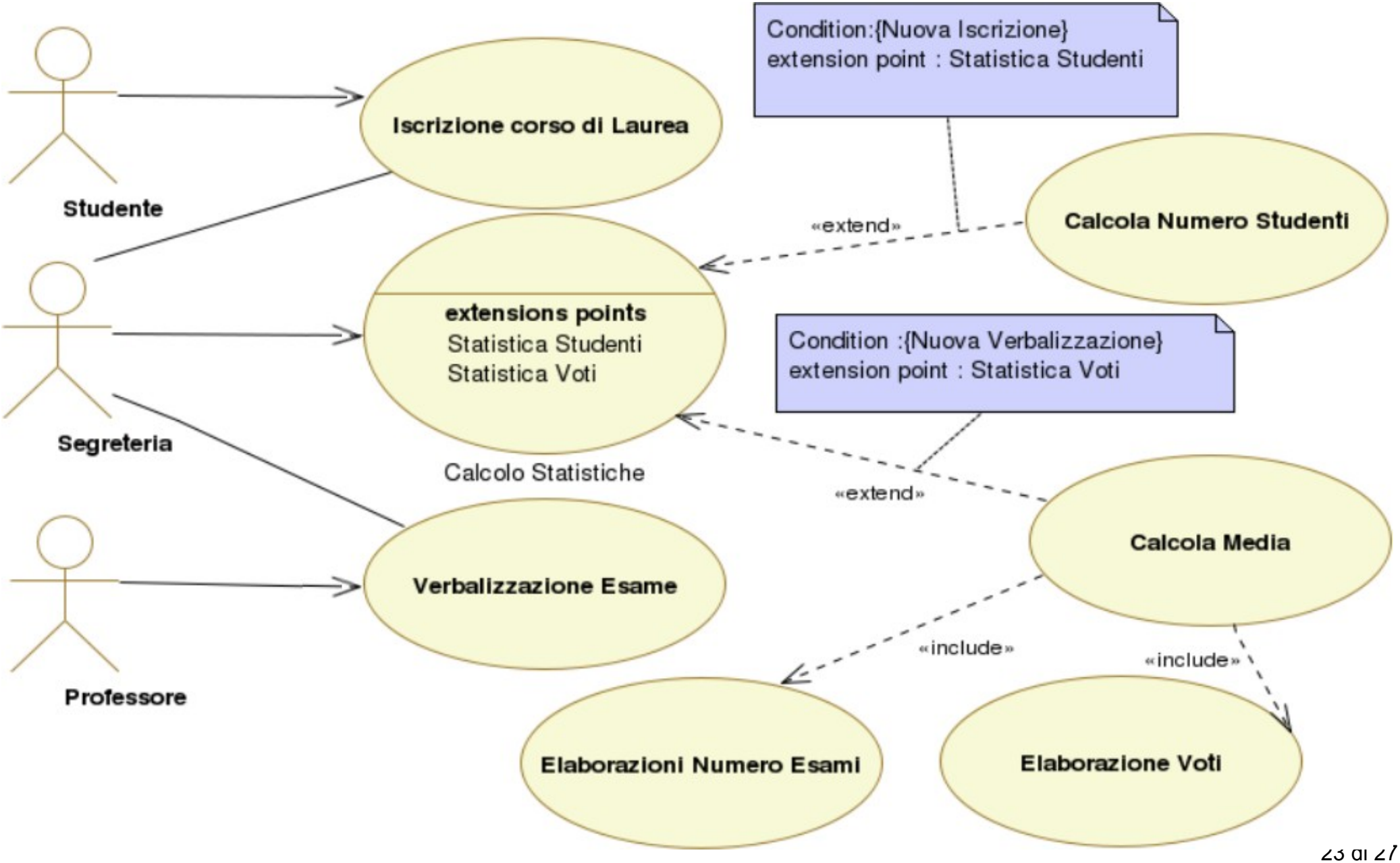
Si vuole modellare un sistema software per la gestione degli studenti e del loro percorso formativo di una Università

In particolare si vogliono modellare gli studenti (con nome, cognome, numero di matricola, età), il corso di laurea in cui sono iscritti, ed i corsi di cui hanno sostenuto l'esame, con il professore che ha verbalizzato l'esame, ed il voto conseguito. Di ogni corso di laurea interessa il codice e il nome. Di ogni corso interessa il nome e la disciplina a cui appartiene (ad esempio: matematica, fisica, informatica, ecc.). Di ogni professore interessa codice ed età.

Al momento dell'iscrizione, lo studente specifica il corso di laurea a cui si iscrive. Dopo che uno studente ha sostenuto un esame, il professore comunica l'avvenuta verbalizzazione con i dati relativi (studente, corso, voto). Una volta che il professore ha completato e chiuso le operazioni di registrazione esami, la segreteria vidima le registrazioni effettuate.

La segreteria vuole periodicamente fare delle statistiche. In particolare, ogni volta che uno studente verbalizza un nuovo esame, vuole calcolare la media dei voti e del numero di esami sostenuti dallo studente. Inoltre, non appena nel sistema una nuova iscrizione viene effettuata, la segreteria vuole aggiornare il numero di studenti di un corso di laurea.

UC: soluzione possibile



… uno scenario di esempio

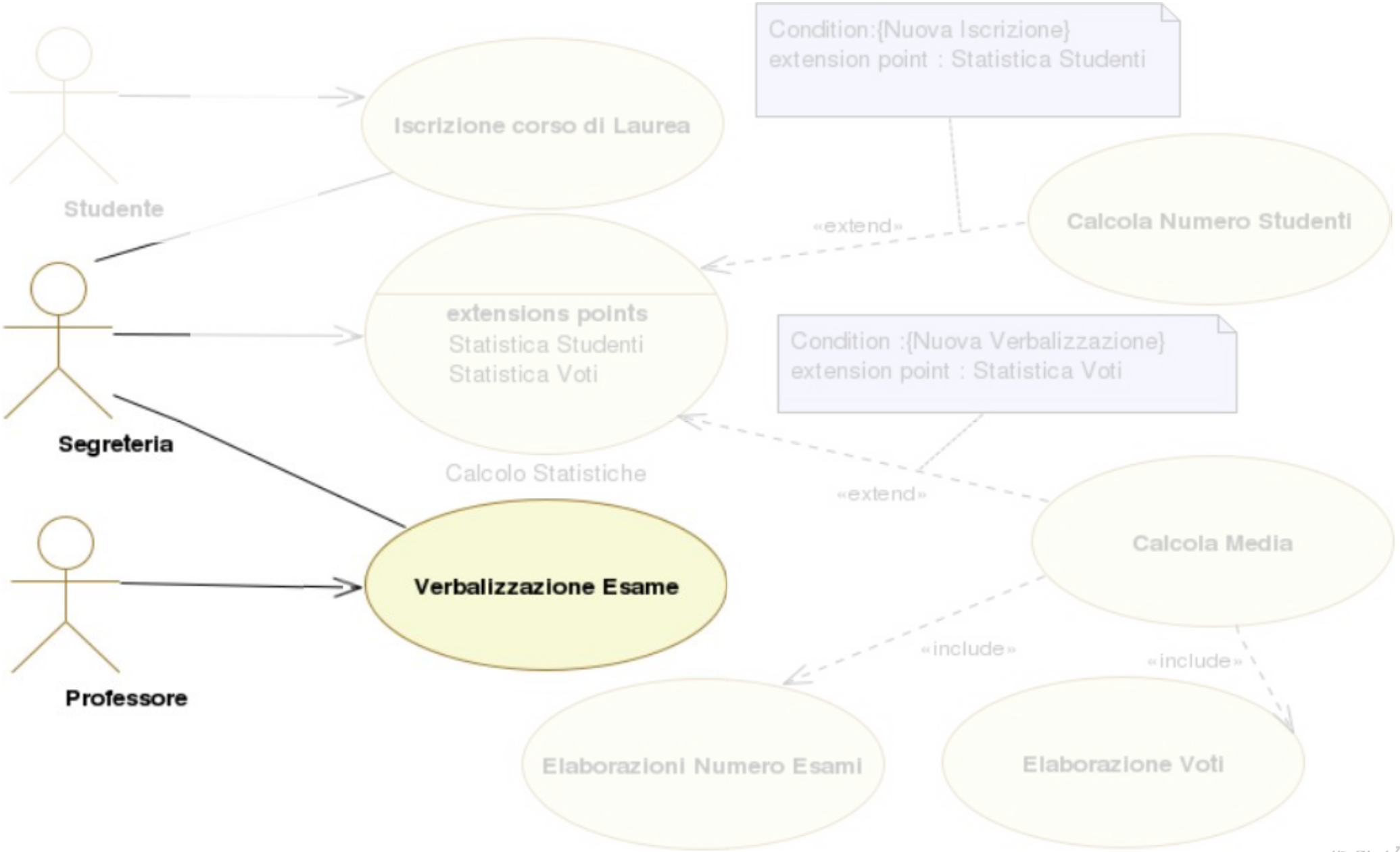
Si vuole modellare un sistema software per la gestione degli studenti e del loro percorso formativo di una Università

In particolare si vogliono modellare gli studenti (con nome, cognome, numero di matricola, età), il corso di laurea in cui sono iscritti, ed i corsi di cui hanno sostenuto l'esame, con il professore che ha verbalizzato l'esame, ed il voto conseguito. Di ogni corso di laurea interessa il codice e il nome. Di ogni corso interessa il nome e la disciplina a cui appartiene (ad esempio: matematica, fisica, informatica, ecc.). Di ogni professore interessa codice ed età.

Al momento dell'iscrizione, lo studente specifica il corso di laurea a cui si iscrive. **Dopo che uno studente ha sostenuto un esame, il professore comunica l'avvenuta verbalizzazione con i dati relativi (studente, corso, voto). Una volta che il professore ha completato e chiuso le operazioni di registrazione esami, la segreteria vidima le registrazioni effettuate.**

La segreteria vuole periodicamente fare delle statistiche. In particolare, ogni volta che uno studente verbalizza un nuovo esame, vuole calcolare la media dei voti e del numero di esami sostenuti dallo studente. Inoltre, non appena nel sistema una nuova iscrizione viene effettuata, la segreteria vuole aggiornare il numero di studenti di un corso di laurea.

UC: soluzione possibile



… proviamo ad applicare BCE

NOTA: per semplicità, nello svolgimento in aula verranno considerati lo UC diagram ed la descrizione del dominio come artefatti in input per il VOPC.

Nel vostro progetto, ed in generale, è buona norma considerare tutti gli artefatti a disposizione (e.g. Documento di Vision, Documenti di Dominio, Descrizione degli Scenari, etc.)

bibliografia di riferimento

- “Design Patterns – Elementi per il riuso di software a oggetti”, Gamma, Helm, Johnson, Vlissides (GoF). Addison-Wesley (1995).
  - Design Patterns: Elements of Reusable Object-Oriented Software
- “Applicare UML e i Pattern – Analisi e progettazione orientata agli oggetti”, Larman. 3za Edizione. Pearson (2005).
  - “Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development”