UNIVERSITÀ DEGLI STUDI DI SALERNO

Ingegneria del Software: Manutenzione e Testing

IMPACT ANALYSIS DOCUMENT

ANNO ACCADEMICO 2015/2016

Versione 1.0

Coordinatore del Progetto:

Nome
Prof. De Lucia Andrea

Partecipanti:

Nome	Matricola
De Chiara Davide	0222500088
Longo Alessandro	0222500121

Revision History:

Data	Versione	Descrizione	Autore
28/10/2015	1.0	Stesura del documento	Davide De Chiara Alessandro Longo

Indice

1. Introduzione

1.1 Scopo del documento

Il progetto di manutenzione del sistema Simplex prevede l'aggiunta di diverse funzionalità inizialmente non supportate dal sistema e la ristrutturazione di alcune componenti fondamentali del sistema come il database.

1.2 Definizioni e Acronimi

Definizioni:

- **Deliverable:** qualsiasi artefatto, come documentazione o componente software, da consegnare al cliente dopo una o più fasi di progetto.

Acronimi:

- **TIR:** Test Incident Report
- **TCS:** Test Case Specification
- **TER:** Test Execution Report
- **TP:** Test Plan
- **IAD:** Impact Analysis Document
- **MR:** Management Report
- **QP:** Quality Plan
- **CIS:** Candidate Impact Set
- **AIS:** Actual Impact Set
- **CR:** Change request

2. Definizione delle funzionalità

2.1 Identificazione della modifica richiesta

L'intervento di manutenzione per il progetto Simplex si suddivide in 3 richieste di cambiamento:

- CR1: Migrazione dello Storage Layer da Talent Hunt a Simplex;
- CR2: Migrazione dell'Application Layer da Talent Hunt a Simplex;
- CR3: Re-ingegnerizzazione dell'architettura del sistema Simplex;

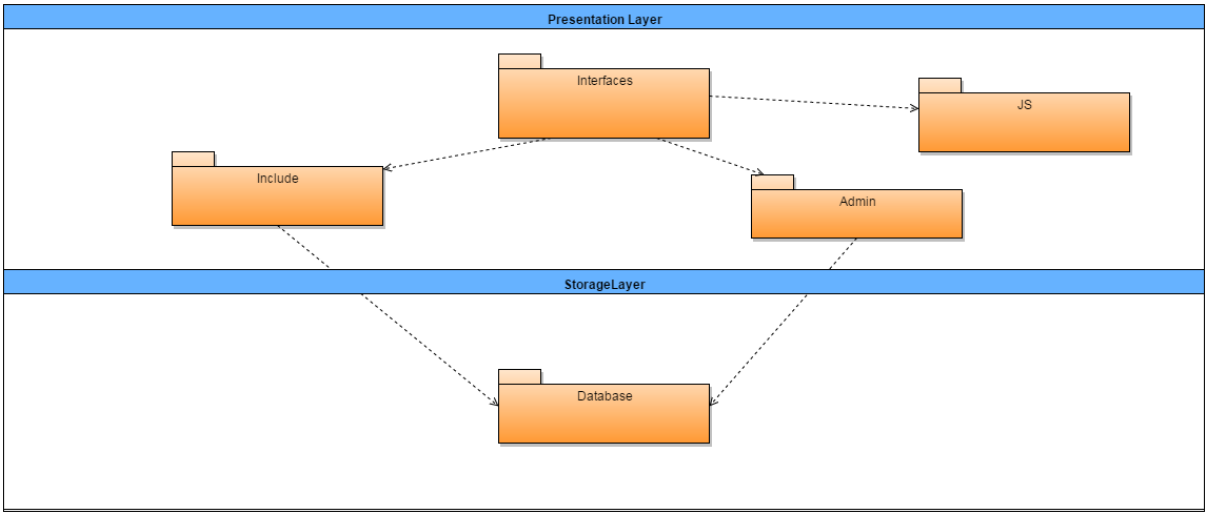
Per evitare un approccio Big Bang e consentire comunque al team di progetto di poter sviluppare il sistema Simplex si è pensato di adottare un approccio incrementale con una fase iniziale di migrazione delle componenti necessarie da Talent Hunt a Simplex.
La CR1 riguarda la migrazione da Talent Hunt a Simplex dello Storage Layer, in quanto tale Layer corrisponde, in Talent Hunt, ad un unica classe.
La CR2 riguarda la migrazione da Talent Hunt a Simplex dell'Application Layer.
La CR3 riguarda il miglioramento del nuovo sistema implementato "Simplex" in particolare è stato richiesto un cambiamento nell'architettura del sistema, spostando parte della logica applicativa dalla view al controller.

2.2 Classificazione del tipo di manutenzione

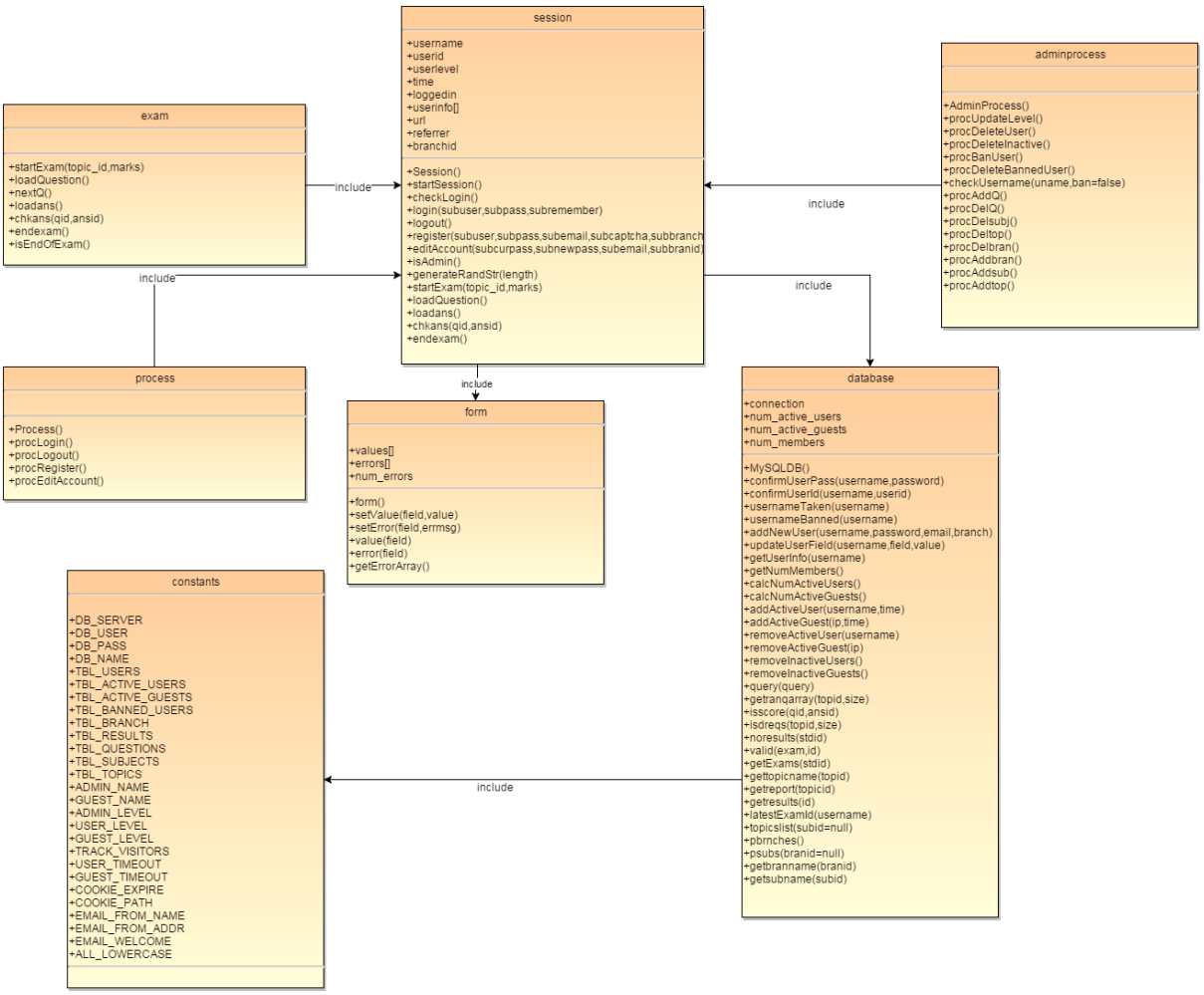
I lavori di manutenzione di questo progetto rientrano nella tipologia detta manutenzione evolutiva in quanto sarà migliorata l'architettura del sistema stesso e saranno create nuove componenti e aggiunte funzionalità al sistema.

2.3 Overview del sistema attuale

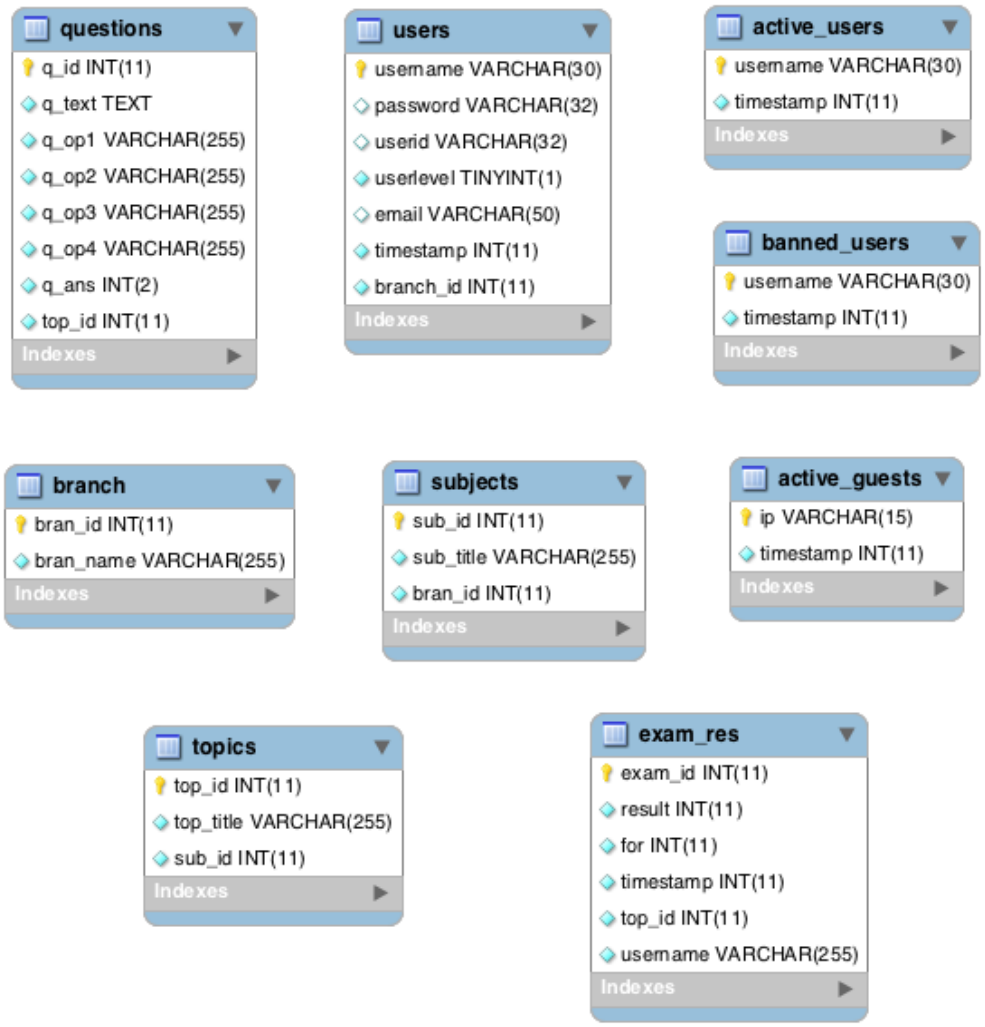
Il sistema Talent Hunt non segue un architettura specifica, cercando di definirla possiamo dire che si basa su due livelli riportati di seguito.



Il class diagram relativo al sistema Talent Hunt comprende le seguenti classi:



Il seguente schema rappresenta il DB del sistema Talent Hunt:



2.4 Obiettivi del lavoro di manutenzione

Il sistema Talent Hunt, allo stato attuale, è totalmente sprovvisto di documentazione e il codice è sprovvisto di commenti.

Il lavoro di manutenzione di questo progetto si propone di aggiungere le nuove funzionalità indicate nelle richieste di modifica e, inoltre, di effettuare test più approfonditi al fine di trovare e risolvere bug sconosciuti.

3. Impact Analysis CR1

L'obiettivo della CR1 è migrare lo Storage Layer del sistema Talent Hunt al sistema Simplex. Obiettivo di tale analisi è valutare l'impatto che tale modifica potrebbe comportare in termini di lavoro, tenendo quindi conto anche dell'esperienza del team di progetto.
Ad ogni modifica apportata verrà indicato l'impatto della modifica sul progetto; in particolare:

- **Impatto Basso:** Dal momento che il sistema è in fase d'implementazione e non di manutenzione, l'aggiunta di una nuova classe viene considerata di basso impatto. Questo in virtù anche della bassa esperienza del team di sviluppo.
- **Impatto Medio:** Modifiche rilevanti ma non eccessive (ad. es. migrazione di alcuni metodi di componenti).
- **Impatto Alto:** Migrazione di più componenti dal vecchio sistema al nuovo, con riutilizzo di diversi metodi di una singola componente ricollocati in più componenti del nuovo sistema.

3.1 Candidate Impact Set

L'insieme previsto delle componenti che saranno impattate dalla migrazione dello Storage Layer sono:

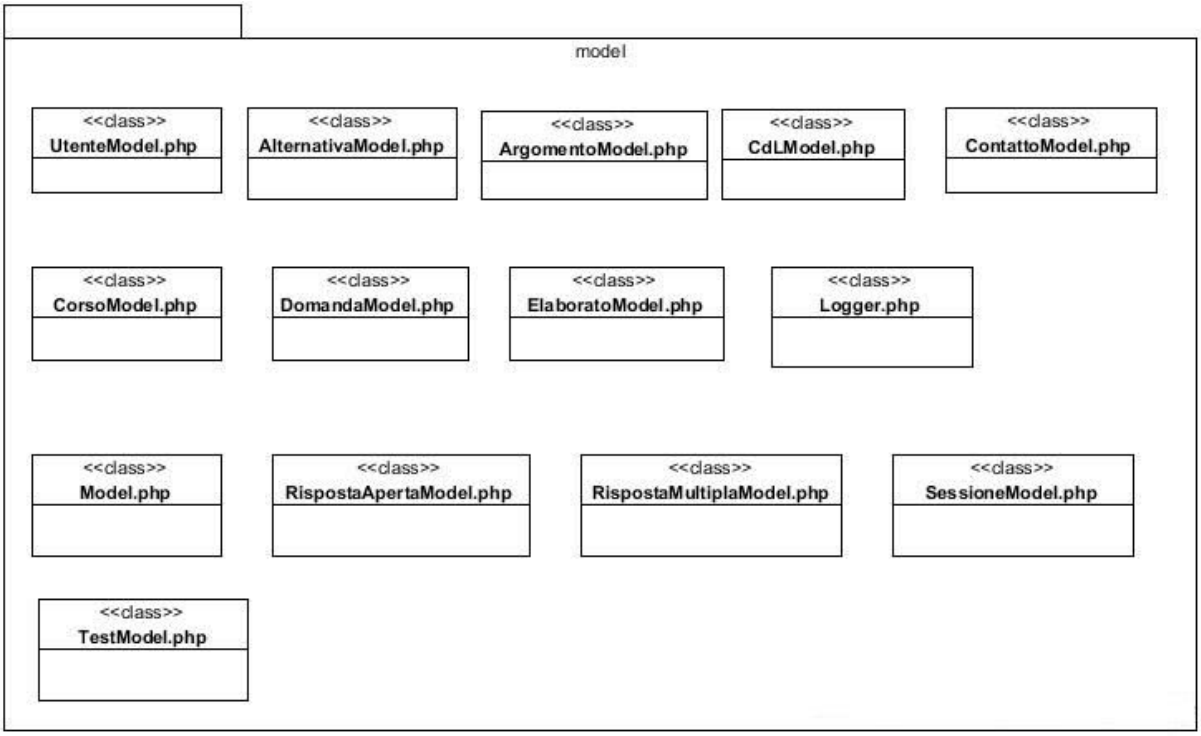
- La **classe database**: Tale classe verrà divisa in diversi manager, tenendo conto delle funzionalità che il sistema Simplex dovrà avere. In particolare le classi aggiunte e create a partire dalla classe database saranno:
    - Utente
    - Contatto
    - Test
    - Argomento (considerando topic come argomento)
    - Domanda
    - RispostaAperta
    - RispostaMultipla
    - Alternativa
    - Corso (considerando un subject come corso)
    - CdL (considerando un branch come CdL)
    - Sessione
    - Elaborato
    - Logger
    - Model

  Ad ogni manager corrisponde quindi un bean omonimo che verrà creato per il sistema Simplex.

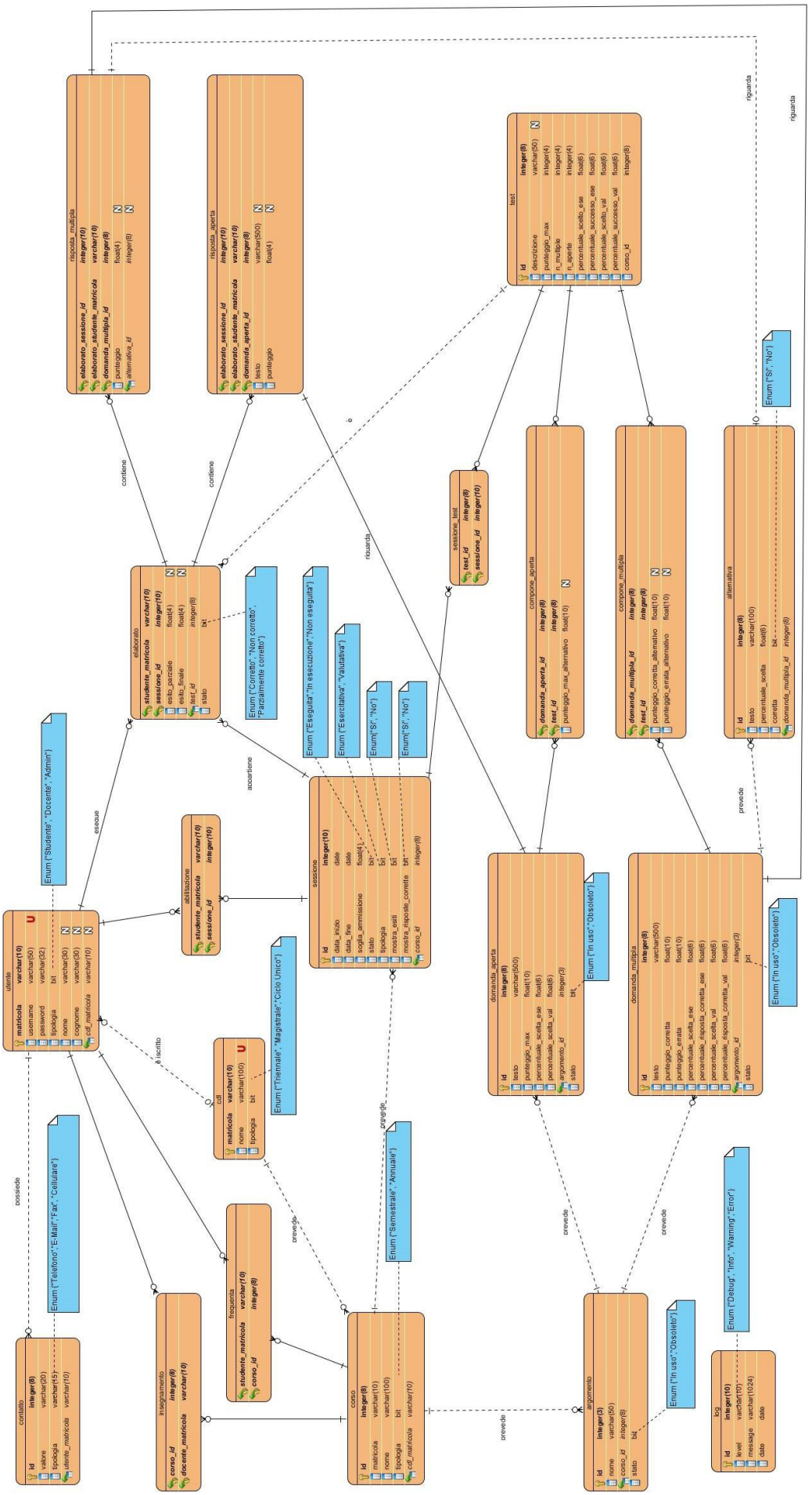
- Il **Database**: Il database viene riutilizzato in parte, aggiungendo le entità necessarie al sistema Simplex, in particolare:
    - Cdl (considerando l'entità già esistente Branch)
    - Corso (considerando l'entità Subject)
    - Argomento (considerando l'entità Topic)
    - Domanda (considerando l'entità Questions)
    - Utente (considerando l'entità Users)
    - Elaborato (considerando l'entità Exam_res)
    - Contatto
    - Sessione
    - Risposta_multipla
    - Risposta_aperta
    - Alternativa
    - Log

3.2 Actual Impact Set & Impact Evaluation

Dopo la migrazione dello Storage Layer di Talent hunt al sistema Simplex questo risulta essere il diagramma relativo al package model:



Inoltre dopo la migrazione il Database risultante è il seguente:


risposta_multipla
elaborato_sessione_id integer(10)
elaborato_studente_matricola varchar(10)
domanda_multipla_id integer(8)
punteggio float(4) N
alternativa_id integer(8) N
risposta_aperta
elaborato_sessione_id integer(10)
elaborato_studente_matricola varchar(10)
domanda_aperta_id integer(8)
testo varchar(500) N
punteggio float(4) N
test
id integer(8)
descrizione varchar(50) N
punteggio_max integer(4)
n_multiple integer(4)
n_aperte integer(4)
percentuale_scelto_ese float(6)
percentuale_successo_ese float(6)
percentuale_scelto_val float(6)
percentuale_successo_val float(6)
corso_id integer(8)
elaborato
studente_matricola varchar(10)
sessione_id integer(10) N
esito_parziale float(4) N
esito_finale float(4)
test_id integer(8)
stato bit
sessione_test
test_id integer(8)
sessione_id integer(10)
compone_aperta
domanda_aperta_id integer(8)
test_id integer(8)
punteggio_max_alternativo float(10) N
compone_multipla
domanda_multipla_id integer(8)
test_id integer(8)
punteggio_corretta_alternativo float(10) N
punteggio_errata_alternativo float(10) N
alternativa
id integer(8)
testo varchar(100)
percentuale_scelta float(6)
corretta bit
domanda_multipla_id integer(8)
utente
matricola varchar(10) U
username varchar(50) U
password varchar(32)
tipologia bit
nome varchar(30) N
cognome varchar(30) N
cdl_matricola varchar(10) N
abilitazione
studente_matricola varchar(10)
sessione_id integer(10)
sessione
id integer(10)
data_inizio date
data_fine date
soglia_ammissione float(4)
stato bit
tipologia bit
mostra_esiti bit
mostra_risposte_corrette bit
corso_id integer(8)
domanda_aperta
id integer(8)
testo varchar(500)
punteggio_max float(10)
percentuale_scelta_ese float(6)
percentuale_scelta_val float(6)
argomento_id integer(3)
stato bit
domanda_multipla
id integer(8)
testo varchar(500)
punteggio_corretta float(10)
punteggio_errata float(10)
percentuale_scelta_ese float(6)
percentuale_risposta_corretta_ese float(6)
percentuale_scelta_val float(6)
percentuale_risposta_corretta_val float(6)
argomento_id integer(3)
stato bit
cdl
matricola varchar(10) U
nome varchar(100)
tipologia bit
contatto
id integer(8)
valore varchar(20)
tipologia varchar(15)
utente_matricola varchar(10)
insegnamento
corso_id integer(8)
docente_matricola varchar(10)
frequenta
studente_matricola varchar(10)
corso_id integer(8)
corso
id integer(8)
matricola varchar(10)
nome varchar(100)
tipologia bit
cdl_matricola varchar(10)
argomento
id integer(3)
nome varchar(50)
corso_id integer(8)
stato bit
log
id integer(10)
level varchar(10)
message varchar(1024)
date date
Enum ("Studente", "Docente", "Admin")
Enum ("Telefono","E-Mail","Fax", "Cellulare")
Enum ("Triennale", "Magistrale","Ciclo Unico")
Enum ("Semestrale", "Annuale")
Enum ("Eseguita","In esecuzione","Non eseguita")
Enum ("Esercitativa", "Valutativa")
Enum ("Si","No")
Enum ("Si","No")
Enum ("Corretto", "Non corretto", "Parzialmente corretto")
Enum ("Si","No")
Enum ("In uso","Obsoleto")
Enum ("In uso","Obsoleto")
Enum ("In uso","Obsoleto")
Enum ("Debug", "Info", "Warning","Error")
contiene
contiene
è
riguarda
riguarda
esegue
associa
è iscritto
possiede
prevede
prevede
prevede
prevede
prevede

Componenti impattate dalla migrazione della **classe Database:**

Componente	Modifica apportata	Impatto
Utente	Creazione della classe utente con metodi della classe Database riutilizzabili (addNewUser, updateUserField)	Medio
Contatto	Creazione della classe Contatto	Basso
Test	Creazione della classe Test	Basso
Argomento	Creazione della classe argomento con metodi della classe Database riutilizzabili (getTopicName, TopicsList)	Medio
Domanda	Creazione della classe Domanda	Basso
RispostaAperta	Creazione della classe della RispostaAperta	Basso
RispostaMultipla	Creazione della classe RispostaMultipla	Basso
Alternativa	Creazione della classe Alternativa	Basso
Corso	Creazione della classe corso con i metodi della classe Database riutilizzabili (getSubname)	Medio
Cdl	Creazione della classe cdl con i metodi della classe Database riutilizzabili (getBranName)	Medio
Sessione	Creazione della classe Sessione	Basso
Elaborato	Creazione della classe elaborato con metodi della classe Database riutilizzabili (getResults)	Medio
Logger	Creazione della classe Logger	Basso
Model	Creazione della classe Model	Basso

Entità impattate dalla migrazione del **Database**:

Componente	Modifica apportata	Impatto
cdl	L’entità branch è stata convertita nell’entità cdl aggiungendo altri attributi	Medio
corso	L’entità subject è stata convertita nell’entità corso aggiungendo altri attributi	Medio
argomento	L’entità topic è stata convertita nell’entità argomento aggiungendo altri attributi	Medio
domanda_aperta	L’entità questions è stata convertita nell’entità domanda_aperta e domanda_chiusa aggiungendo altri attributi	Alto
domanda_chiusa	L’entità questions è stata convertita nell’entità domanda_aperta e domanda_chiusa aggiungendo altri attributi	Alto
utente	L’entità users è stata convertita nell’entità utente aggiungendo altri attributi	Medio
elaborato	L’entità exam_res è stata convertita nell’entità elaborato aggiungendo altri attributi	Medio
contatto	Aggiunta entità contatto	Basso
sessione	Aggiunta entità sessione	Basso
risposta_multipla	Aggiunta entità risposta_multiple	Basso
risposta_aperta	Aggiunta entità risposta_aperta	Basso
alternativa	Aggiunta entità alternativa	Basso
log	Aggiunta entità log	Basso

3.3 Recall & Precision

Per quanto riguarda le componenti impattate dalla migrazione della classe **Database** (Come detto in precedenza a ogni Model corrisponde un Bean quindi CIS e AIS non corrispondono a 14 ma a 28):

Recall = $| CIS \cap AIS | \div |AIS|$ = 28/28 = 100%

Precision = $| CIS \cap AIS | \div |CIS|$ = 28/28 = 100%

Per quanto riguarda le entità impattate dalla migrazione del **Database**:

Recall =$| CIS \cap AIS | \div |AIS|$ = 12/13 = 92%

Precision = $| CIS \cap AIS | \div |CIS|$= 12/12 = 100%

3.4 Cost/Benefit Analysis

3.4.1 Fattori umani

Il team di progetto non è in possesso di tutte le competenze necessarie ad effettuare l'operazione di migrazione dello Storage Layer. Per questo sarà necessario il supporto di programmatori più esperti.

3.4.2 Costi a breve e lungo termine

Tale operazione richiede dei tempi non molto lunghi, in quanto l'architettura del sistema Simplex è già stata definita nell'SDD per cui le operazioni di integrazione ed implementazione non risultano eccessive.

3.4.3 Vantaggi dell'operazione

La migrazione descritta consentirà al team di progetto di porre le basi per lo sviluppo del sistema Simplex.

3.5 Risorse utilizzate

Di seguito sono riportate le attività svolte per la migrazione dello Storage layer:

Task	Ore impiegate
Analisi dello Storage Layer di Talent Hunt	6
Implementazione dei manager e del Database di Simplex	38
Esecuzione Test e risoluzione bug	17
Totale task	61

4. Impact Analysis CR2

L'obiettivo della CR2 è migrare l'Application Layer del sistema Talent Hunt nel nuovo sistema Simplex. Obiettivo di tale analisi è valutare l'impatto che tale modifica potrebbe comportare in termini di lavoro, tenendo quindi conto anche dell'esperienza del team di progetto.
Ad ogni modifica apportata verrà indicato l'impatto della modifica sul progetto; in particolare:

- **Impatto Basso:** Dal momento che il sistema è in fase d'implementazione e non di manutenzione, l'aggiunta di una nuova classe viene considerata di basso impatto. Questo in virtù anche della bassa esperienza del team di sviluppo.
- **Impatto Medio:** Modifiche rilevanti ma non eccessive (ad. es. migrazione di alcuni metodi di componenti).
- **Impatto Alto:** Migrazione di più componenti dal vecchio sistema al nuovo, con riutilizzo di diversi metodi di una singola componente ricollocati in più componenti del nuovo sistema.

4.1 Candidate Impact Set

L'insieme delle componenti che si prevede saranno impattate dalla migrazione sono:

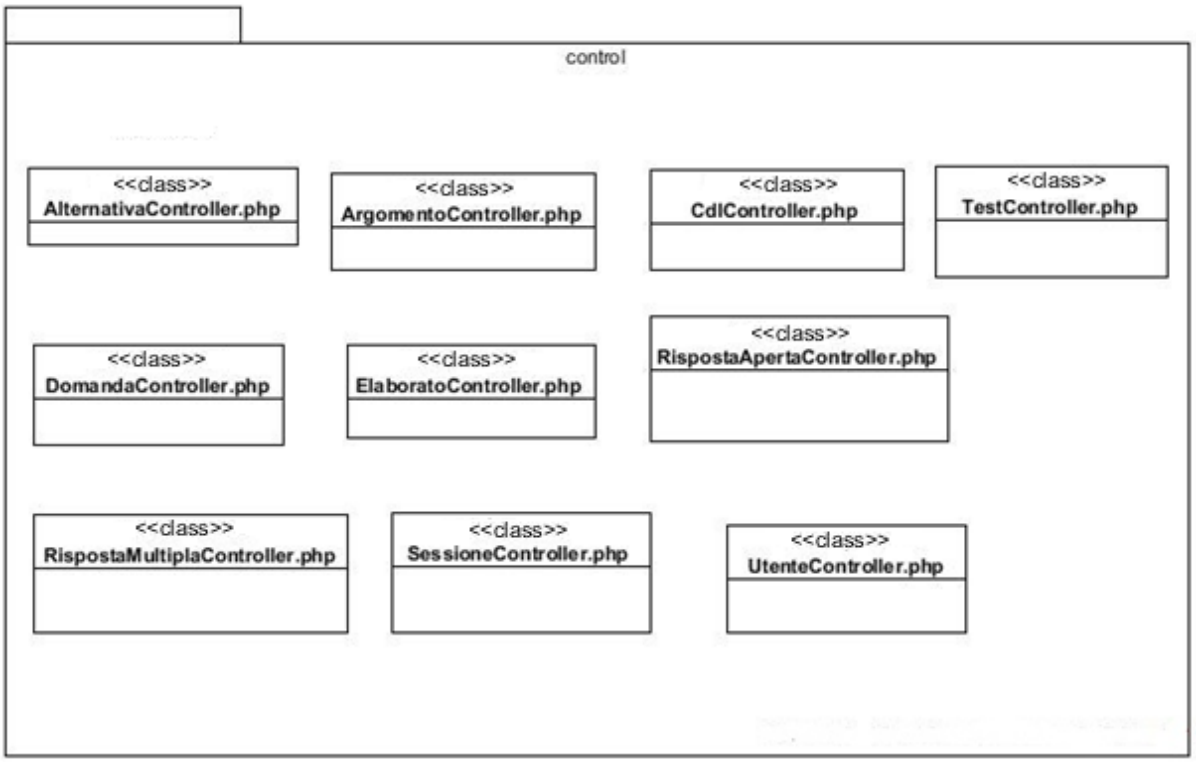
- La classe Session
- La classe Exam

In particolare le classi aggiunte e create a partire dalle suddette classi sono:

- UtenteController (considerando la classe Session)
- SessioneController
- CdLController
- ElaboratoController
- TestController (considerando la classe Exam)
- DomandaController
- RispostaApertaController
- RispostaMultiplaController
- AlternativaController
- ArgomentoController

4.2 Actual Impact Set & Impact Evaluation

Dopo la migrazione dell'Application Layer di Talent hunt al sistema Simplex questo risulta essere il diagramma relativo al package control:



Componenti impattate dalla migrazione delle classi Session e Exam**:**

Componente	Modifica apportata	Impatto
UtenteController	Creazione della classe UtenteController con i metodi della classe Session riutilizzabili (checklogin, login, logout, register)	Medio
SessioneController	Creazione della classe SessioneController	Basso
CdlController	Creazione della classe CdlController	Basso
ElaboratoController	Creazione della classe ElaboratoController	Basso
TestController	Creazione della classe TestController	Basso
DomandaController	Creazione della classe DomandaController riutilizzando i metodi di Exam(loadQuestion)	Medio
RispostaApertaController	Creazione della classe RispostaApertaController	Basso

RispostaMultiplaController	Creazione della classe RispostaMultiplaController	Basso
AlternativaController	Creazione della classe AlternativaController	Basso
ArgomentoController	Creazione della classe ArgomentoController	Basso

4.3 Recall & Precision

Per quanto riguarda le componenti impattate dalla migrazione delle classi **Session e Exam**:

Recall = $| CIS \cap AIS | \div |AIS|$= 10/10 = 100%

Precision = $| CIS \cap AIS | \div |CIS|$ = 10/10 = 100%

Tuttavia era stata prevista l'integrazione all'interno della classe TestController, invece tale integrazione è stata effettuata nella classe DomandaController.

4.4 Cost/Benefit Analysis

4.4.1 Fattori umani

Il team di progetto non è in possesso di tutte le competenze necessarie ad effettuare l'operazione di migrazione dell'Application Layer. E' stato quindi necessario effettuare un training di base sulle skill più critiche: PHP.

4.4.2 Costi a breve e lungo termine

Tale operazione richiede dei tempi non molto lunghi, in quanto l'architettura del sistema Simplex è già stata definita nell'SDD per cui le operazioni di integrazione ed implementazione non risultano eccessive.

4.4.3 Vantaggi dell'operazione

La migrazione descritta consentirà al team di progetto di porre le basi per lo sviluppo del sistema Simplex.

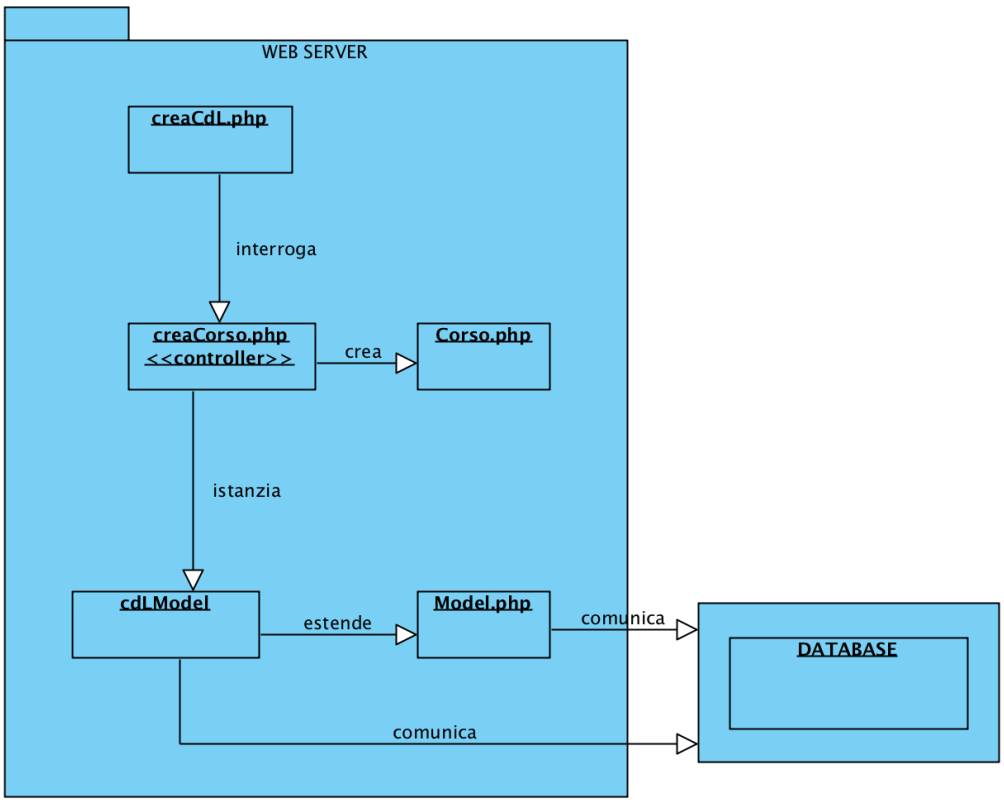
4.5 Risorse utilizzate

Di seguito sono riportate le attività svolte per la migrazione dell'Application layer:

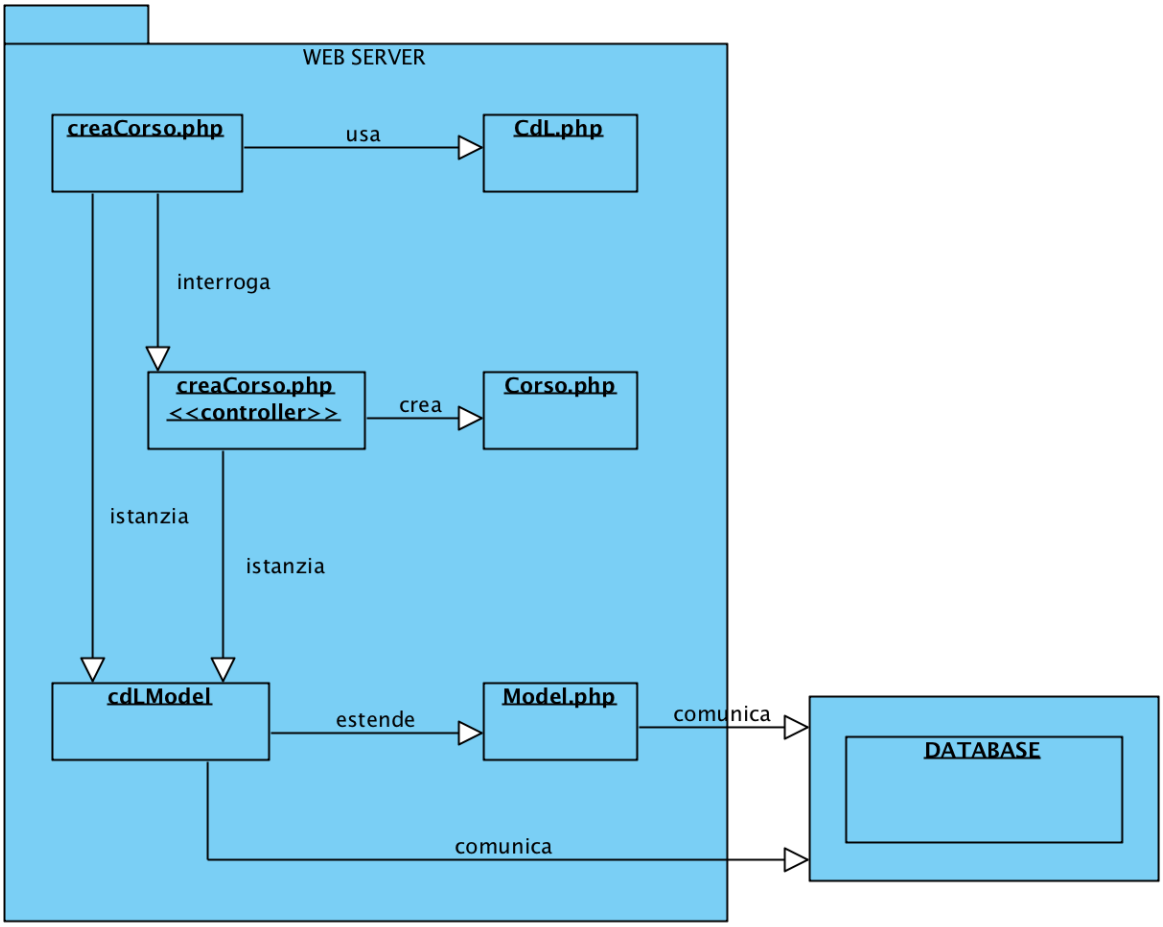
Task	Ore impiegate
Analisi dell'Application Layer di Talent Hunt	4
Implementazione dei Controller di Simplex	62
Esecuzione Test e risoluzione bug	18
Totale task	84

5. Impact Analysis CR3

L'obiettivo della CR3 è l'evoluzione del sistema Simplex. Tale modifica è necessaria per rendere il sistema riutilizzabile in futuro anche per applicazioni mobile. In particolare il funzionamento di Simplex è descritto da questo esempio:



La modifica riguarda quindi la comunicazione tra la view ed i Model, come descritto nell'esempio:

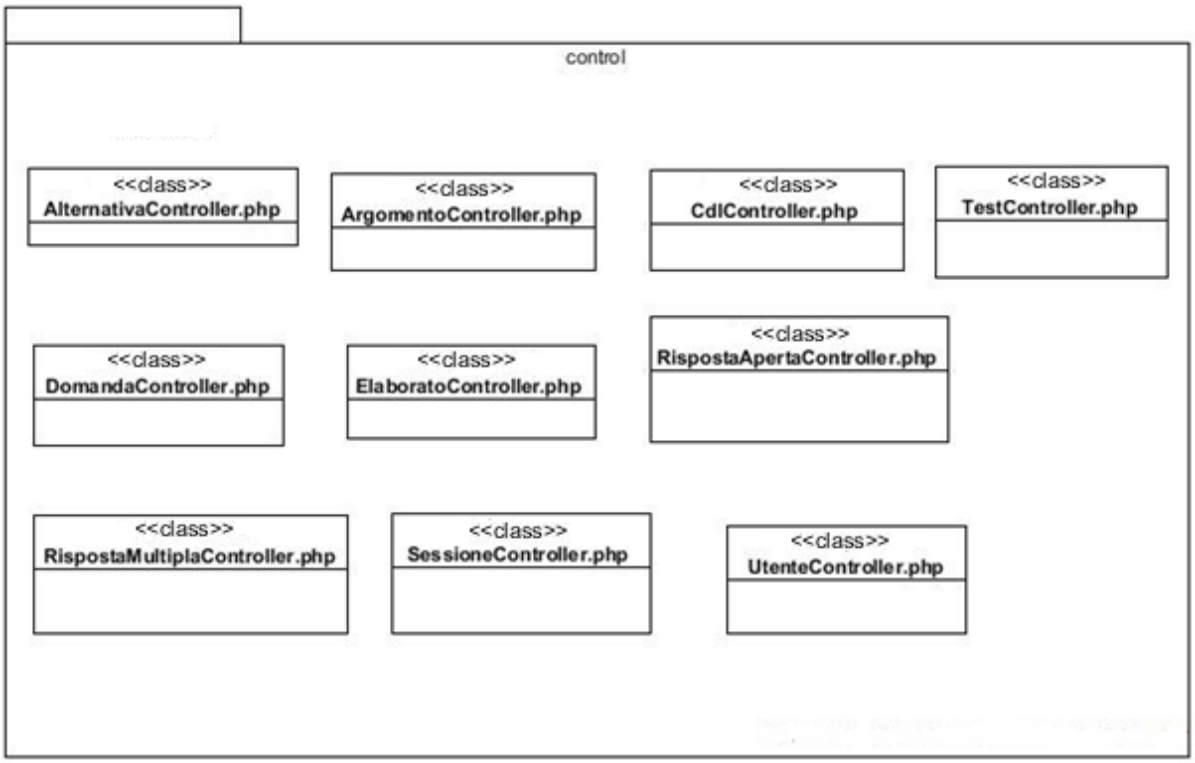


Obiettivo di tale analisi è valutare l'impatto che tale modifica potrebbe comportare in termini di lavoro, tenendo quindi conto anche dell'esperienza del team di progetto.
Ad ogni modifica apportata verrà indicato l'impatto della modifica sul progetto; in particolare:

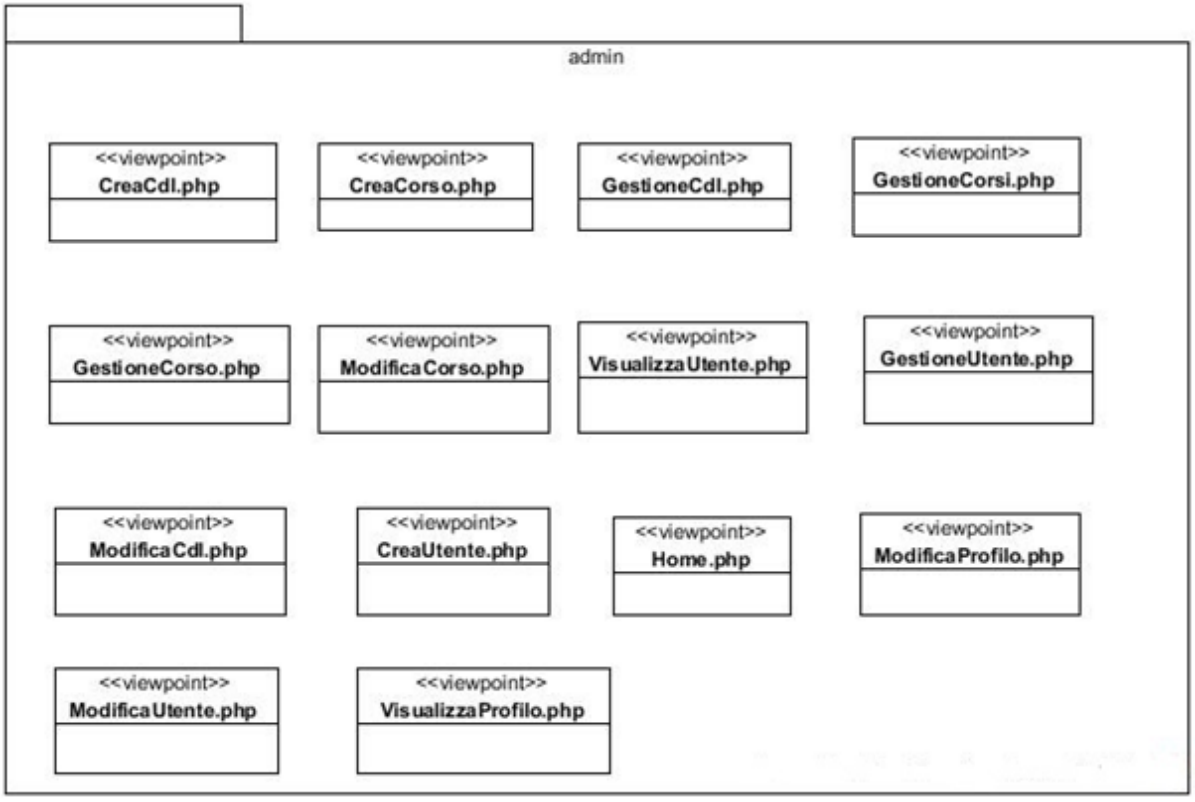
- **Impatto Basso:** Poche modifiche che non incidono pesantemente sul sistema.
- **Impatto Medio:** Modifiche rilevanti ma non eccessive.
- **Impatto Alto:** notevoli modifiche che stravolgono la componente, o creazione della componente prima non esistente.

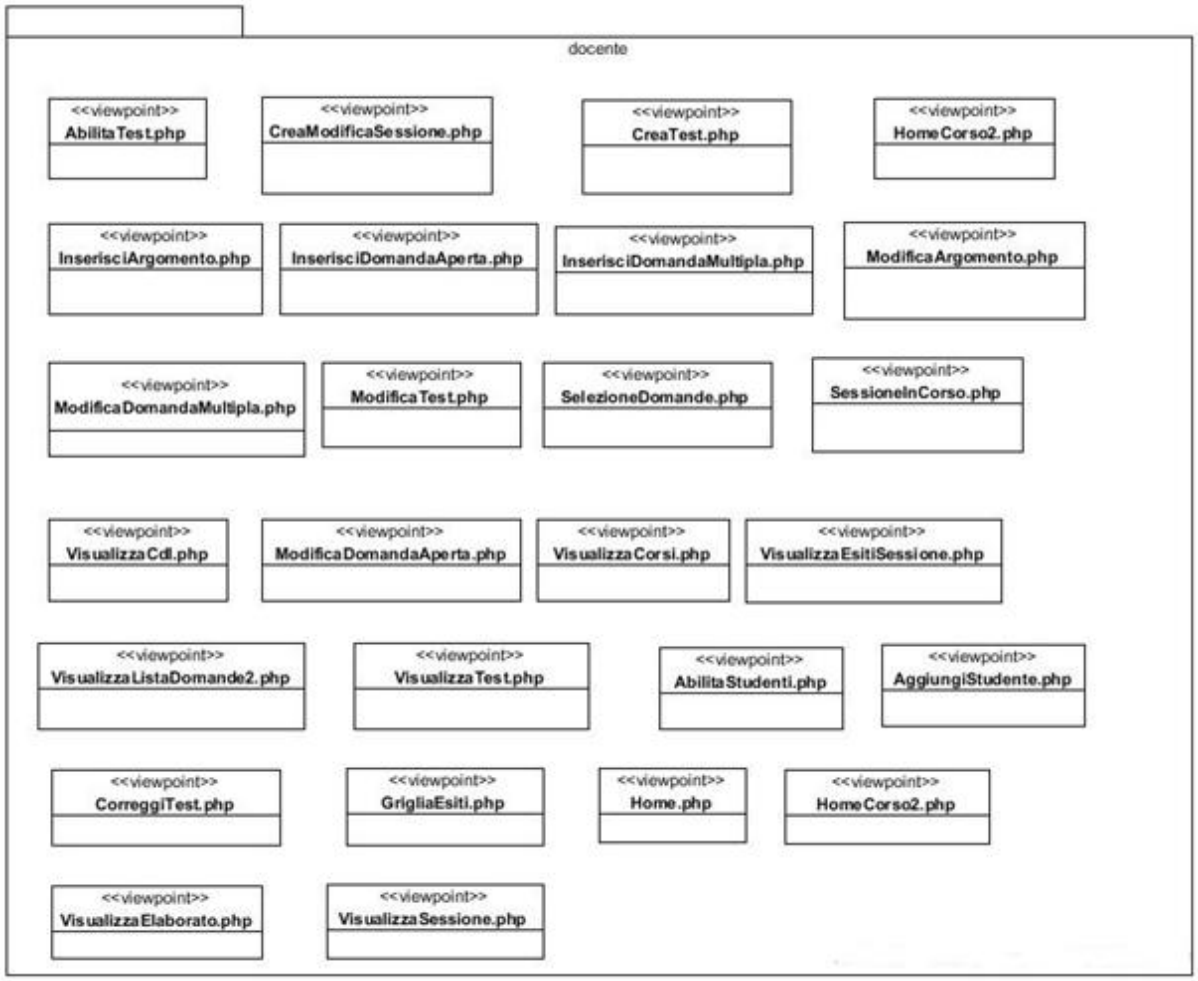
5.1 Candidate Impact Set

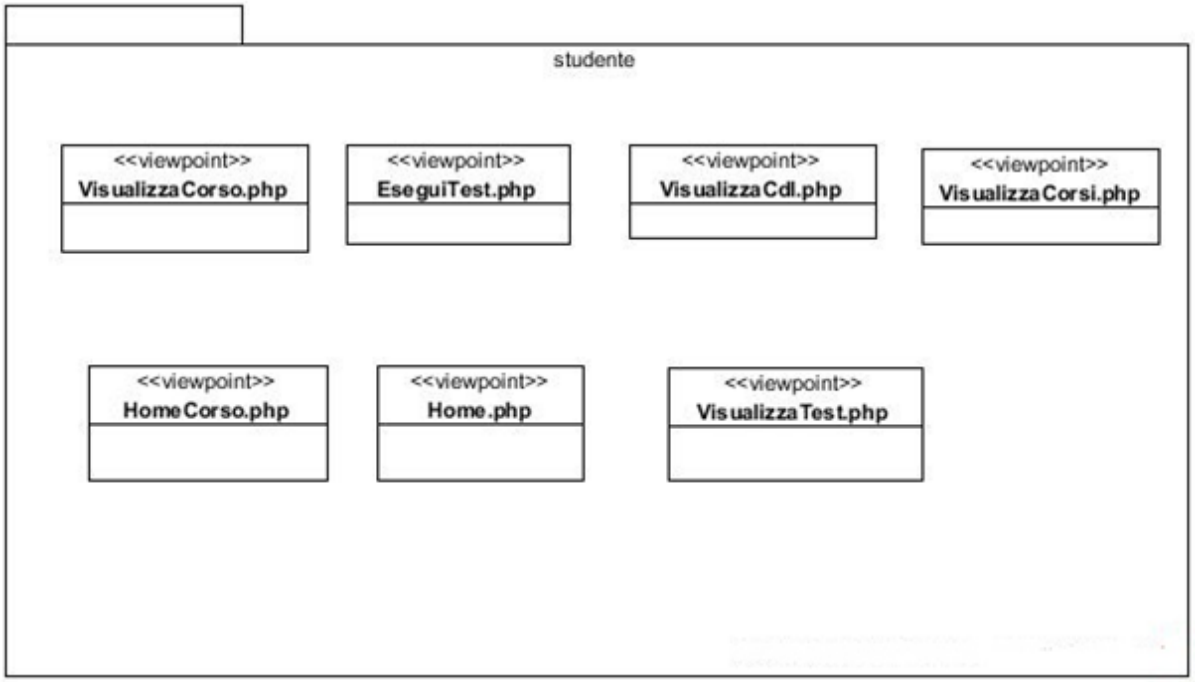
Il package seguente descrive il candidate Impact Set è il seguente:



Considerando anche le seguenti view:




docente
<<viewpoint>>
AbilitaTest.php
<<viewpoint>>
CreaModificaSessione.php
<<viewpoint>>
CreaTest.php
<<viewpoint>>
HomeCorso2.php
<<viewpoint>>
InserisciArgomento.php
<<viewpoint>>
InserisciDomandaAperta.php
<<viewpoint>>
InserisciDomandaMultipla.php
<<viewpoint>>
ModificaArgomento.php
<<viewpoint>>
ModificaDomandaMultipla.php
<<viewpoint>>
ModificaTest.php
<<viewpoint>>
SelezioneDomande.php
<<viewpoint>>
SessioneInCorso.php
<<viewpoint>>
VisualizzaCdl.php
<<viewpoint>>
ModificaDomandaAperta.php
<<viewpoint>>
VisualizzaCorsi.php
<<viewpoint>>
VisualizzaEsitiSessione.php
<<viewpoint>>
VisualizzaListaDomande2.php
<<viewpoint>>
VisualizzaTest.php
<<viewpoint>>
AbilitaStudenti.php
<<viewpoint>>
AggiungiStudente.php
<<viewpoint>>
CorreggiTest.php
<<viewpoint>>
GrigliaEsiti.php
<<viewpoint>>
Home.php
<<viewpoint>>
HomeCorso2.php
<<viewpoint>>
VisualizzaElaborato.php
<<viewpoint>>
VisualizzaSessione.php


studente
<<viewpoint>>
VisualizzaCorso.php
<<viewpoint>>
EseguiTest.php
<<viewpoint>>
VisualizzaCdl.php
<<viewpoint>>
VisualizzaCorsi.php
<<viewpoint>>
HomeCorso.php
<<viewpoint>>
Home.php
<<viewpoint>>
VisualizzaTest.php

5.2 Actual Impact Set & Impact Evaluation

Dopo la re-ingegnerizzazione di Simplex le seguenti classi risultano impattate:

Componente	Modifica apportata	Impatto
UtenteController	La classe è stata eliminata ed il codice è stato suddiviso in diverse pagine: modificaProfilo.php login.php logout.php register.php creaUtente.php salvaUtente.php eliminaUtente.php	Alto
SessioneController	La classe è stata eliminata ed il codice è stato suddiviso in diverse pagine: abilitaStudente.php aggiornaSessioneInCorso.php aggiungiStudente.php annullaEsame.php avviaSessione.php cambiaSoglia.php checkDataSessione.php correggiTest.php creaSessione.php filtroTests.php gestoreDataServer.php indexSessioneInCorso.php indexVisualizza.php indexVisualizzaEsiti.php modificaDataFineInCorso.php modificaSessione.php rimuoviSessione.php terminaSessione.php	Alto
CdlController	La classe è stata eliminata ed il codice è stato suddiviso in diverse pagine: creaCdL.php modificaCdL.php eliminaCdL.php creaCorso.php modificaCorso.php eliminaCorso.php iscriviDisiscriviStudente.php	Alto
ElaboratoController	La classe è stata eliminata	Alto

	ed il codice è stato suddiviso in diverse pagine: consegna.php abbandona.php controllaAbilitazione.php creaElaborato.php gestioneCountdown.php	
TestController	La classe è stata eliminata ed il codice è stato suddiviso in diverse pagine: creaTest.php modificaTest.php eliminaTest.php	Alto
DomandaController	La classe è stata eliminata ed il codice è stato suddiviso in diverse pagine: creaDomandaAperta.php modificaDomandaAperta.php rimuoviDomandaAperta.php creaDomandaMultipla.php modificaDomandaMultipla.php rimuoviDomandaMultipla.php	Alto
RispostaApertaController	La classe è stata eliminata ed il codice è stato suddiviso in diverse pagine: creaRispostaApertaMultipla.php updateAperta.php	Alto
RispostaMultiplaController	La classe è stata eliminata ed il codice è stato suddiviso in diverse pagine: creaRispostaApertaMultipla.php updateMultipla.php	Alto
AlternativaController	La classe è stata eliminata ed il codice è stato suddiviso in diverse pagine: creaDomandaMultipla.php modificaDomandaMultipla.php rimuoviDomandaMultipla.php	Alto
ArgomentoController	La classe è stata eliminata ed il codice è stato suddiviso in diverse pagine:	Alto

	creaArgomento.php modificaArgomento.php rimuoviArgomento.php	

Componente	Modifica apportata	Impatto
Tutte le view del CIA	Rimozione dell'include dei Controller. Implementazione della comunicazione con le pagine control elencate nella tabella precedente.	Medio

Per il Package Diagram si fa riferimento all'ODD 2.1.

5.3 Recall & Precision

Per quanto riguarda le componenti impattate dalla re-ingegnerizzazione abbiamo:

Recall = $| CIS \cap AIS | \div |AIS|$ = 57/57 = 100%

Precision = $| CIS \cap AIS | \div |CIS|$= 57/57 = 100%

5.4 Cost/Benefit Analysis

5.4.1 Fattori umani

Il team di progetto è già in possesso di tutte le competenze necessarie ad effettuare l'operazione di re-ingegnerizzazione del sistema. Questo è dovuto al fatto che il sistema era già stato interamente implementato per cui l'esperienza del team e la conoscenza del sistema erano maggiori.

5.4.2 Costi a breve e lungo termine

L'architettura del sistema Simplex è già stata ridefinita nell'SDD tuttavia le operazioni di re-ingegnerizzazione hanno comportato un dispendio di tempo e sforzi notevoli dovuti all'elevato numero di componenti implementate.

5.4.3 Vantaggi dell’operazione

La re-ingegnerizzazione del sistema permetterà di avere una struttura autonoma e sicura che garantisce una maggiore manutenibilità e evoluzione del sistema stesso.

5.5 Risorse utilizzate

Di seguito sono riportate le attività svolte per la migrazione dell’Application layer:

Task	Ore impiegate
Analisi del sistema Simplex	5
Implementazione della re-ingegnerizzazione di Simplex	30
Esecuzione Test e risoluzione bug	13
Totale task	48