ALMA MATER STUDIORUM
UNIVERSITÀ DEGLI STUDI DI BOLOGNA

Seconda Facoltà di Ingegneria
Corso di Laurea in Ingegneria Informatica

AGENT-BASED LEARNING ENVIRONMENT

Elaborata nel corso di: Sistemi Multi-Agente LM

Relazione di:
FRANCESCO BRUSCHI
ANDREA GIULIA CIALOTTI
FRANCESCA CIOFFI

Docente:
Prof. ANDREA OMICINI

ANNO ACCADEMICO 2010–2011

Indice

Introduzione

Il nostro progetto consiste nella creazione di un ambiente virtuale tipo E-learning in cui sia l'insegnante che i compagni di studio non sono persone ma sono rappresentati da agenti software. Ci siamo avvalsi anche di un LMS (Learning Management System) chiamato Eledge e prodotto dall'università dello Utah (`http://eledge.sourceforge.net`) : esso permette di fare entrare meglio lo studente nell'ottica del corso, e di gestire in modo coerente e ben organizzato il materiale relativo al corso stesso.

Questo tool servirà soltanto come interfacciamento verso l'utente, mentre per i co-learner ci siamo avvalsi di una suite di tools per il NLP (Natural Language Processing) dell'università dello Stanford (`http://nlp.stanford.edu/software/index.shtml`) che consentono di fare un'analisi logica a partire da un testo in inglese che serviranno per l'appendimento degli stessi.

Parole chiave: Multi-Agent System, E-learning, Learning Management System, Natural Language Processing

Capitolo 1

Requisiti

1.1 Premessa

Si noti prima di tutto che i passi che qui riportiamo sono una codifica dell'ipotetico processo di sviluppo che porterebbe alla realizzazione del nostro sistema; questo perché nella realtà, i nostri requisiti non erano ben determinati a priori e sono cambiati durante il processo di sviluppo.

1.2 Requisiti di dominio

Come in ogni sistema di e-learning anche in questo caso, ovviamente, lo scopo atteso è quello di far apprendere all'utente più informazioni possibili relative al corso corrente. Esso è organizzato in questo modo: dopo la spiegazione di ogni lezione viene sottoposto ad ogni studente, umano o agente che sia, un test che verifichi il suo grado di apprendimento relativamente all'argomento appena esposto. Il procedere con la lezione successiva avverrà soltanto quando lo studente avrà ottenuto una valutazione sufficiente alla lezione precedente. L'entità che ha il compito di occuparsi di questa gestione del corso è l'agente software incaricato del compito del teacher.

I casi d'uso sono principalmente due, ognuno dei quali avrà delle parti specifiche per l'utente e per i gli agenti co-learner.

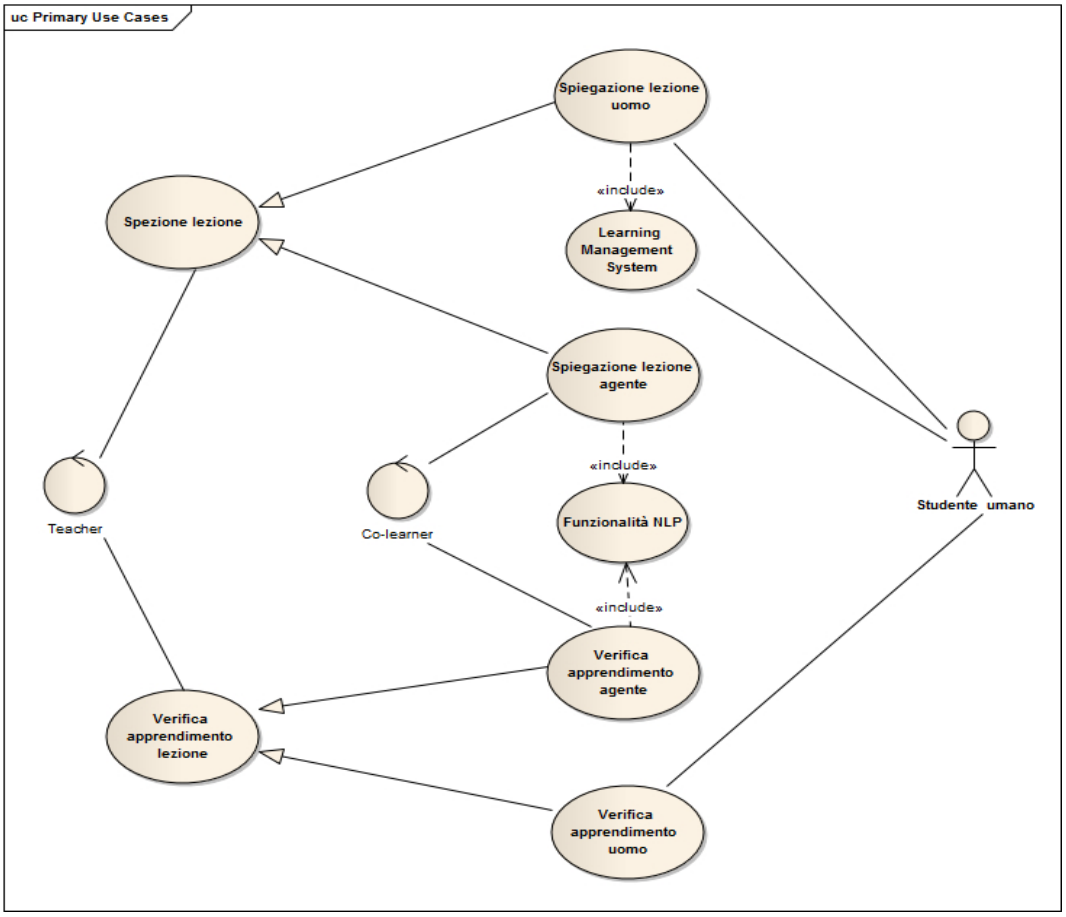

Figura 1.1: Casi d'uso

1.3 Requisiti funzionali

Nello specifico il nostro sistema, a differenza dei comuni sistemi di e-learning ha delle caratteristiche peculiari, date dalla presenza degli agenti software sopra menzionati. In particolare, le features specifiche che l'introduzione di tali agenti comporta possono essere sintetizzate come segue:

- gli errori commessi nei test dagli "studenti" saranno un utile spunto per il ripasso di tutti o se la fase di apprendimento non fosse risultata abbastanza efficace, ciò potrebbe essere utile per assimilare tali contenuti;

- l'interazione fra studente e agenti avviene mediante l'uso di due chat: una viene utilizzata dall'agente teacher per dare istruzioni allo studente, l'altra invece viene utilizzata dallo studente per fare domande al co-learner "vicino di banco" relativamente ad argomenti del corso che non gli sono chiari;

- i co-learner sono tutti ugualmente bravi ad apprendere la lezione, ma il loro comportamento durante le "sessioni d'esame" si diversifica da agente ad agente: alcuni saranno più propensi a rischiare una risposta quando non conoscono quale sia quella giusta, altri co-learner invece lo saranno meno, e quindi saranno più propensi a rispondere "non lo so".

1.3.1 Struttura dell'esame

Bisogna dire, a questo punto, che per quanto riguarda l'organizzazione degli esami si è scelta questa soluzione: ogni esame è composto da tre parti:

- una domanda vero/falso;

- una domanda con risposta da scegliersi, in modo mutuamente esclusivo, fra due possibilità;

- una domanda con tre possibili risposte; in questo caso, però, le risposte corrette possono essere nessuna, una, due, o tutte e tre.

Nella sua totalità, dunque, ogni esame è composto da 1 + 1 + 3 domande, e ad ognuna di queste, oltre che sì o no, si può rispondere "non lo so". I punteggi totalizzati per ogni domanda sono i seguenti:

- risposta corretta (vero o falso) →+1 punto
- risposta sbagliata (vero o falso) →-1 punto
- risposta "non lo so" →0 punti

Se l'utente vuole passare l'esame deve totalizzare almeno un punteggio di 3 su 5.

Capitolo 2

Analisi del problema

2.1 Struttura del sistema

Dal punto di vista strutturale il sistema è composto dalle seguenti entità:

- il LMS (Learning Management System), che è il sistema attraverso cui gestiremo gli artefatti relativi al supporto per l'apprendimento(immagini, video, testi, pagine Web, GUI) da parte dello studente;
- il teacher, ovvero l'entità attiva che gestisce l'avanzamento del corso e che decide quali artefatti usare per una determinata lezione; è una sorta di coordinatore del sistema stesso;
- i co-learners, cioè le entità attive che rappresentano i compagni di classe dello studente. Tra i co-learners, in particolare, viene scelto un "vicino di banco", ovvero colui con il quale lo studente può comunicare tramite un supporto che fa da ponte fra i due.

Il problema, per sua natura, porta con sé alcune questioni a cui bisognerà dare soluzione in fase di progetto:

- dovrà essere scelta una piattaforma ad agenti per quanto riguarda l'implementazione delle entità attive del sistema; questa scelta infatti sembra la più naturale ed ovvia, se pensiamo all'insegnante e ai co-learner, come è naturale che sia, come entità computazionalmente attive, autonome, e proattive;

- l'immersione dell'utente in un "ambiente di apprendimento" è una caratteristica importante di un sistema come questo; a questo fine sarà opportuno creare, dal nulla o a partire da ambienti già esistenti, un tale ambiente che possa supportare lo scopo educativo del corso.

Secondo quanto appena detto risulta naturale mappare le entità attive che ricoprono i ruoli di teacher e co-learners in altrettanti agenti software.

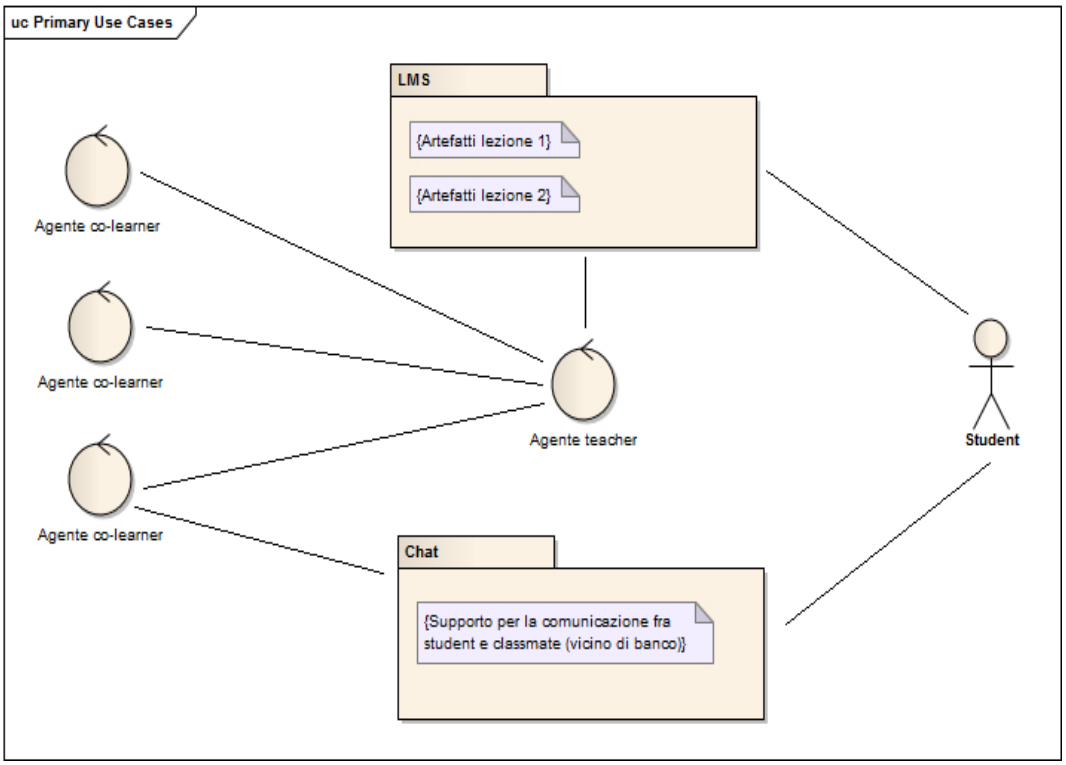

Figura 2.1: Rappresentazione della struttura generale del sistema

2.2 General Pedagocical Agent Pattern

Le entità che possono essere realizzate come agenti sono il teacher ed i co-learners. In letteratura (articolo [5]) per la realizzazione di un agente

pedagogico si ritrova spesso il pattern General Pedagogical Agent; in tale nome si noti, in particolare, l'aggettivo "general"; esso deriva dal fatto che tale pattern è stato astratto a partire da come erano stati realizzati vari agenti, in progetti diversi, nell'ambito di applicazioni con scopo educativo; i diversi ruoli ricoperti dagli agenti in tali circostanze, sono per esempio quello di studente, insegnante, compagno, e assistente. Tutti questi diversi tipi di agente rispecchiano, pur nella loro diversità, lo schema generale identificato nel pattern sopra citato.

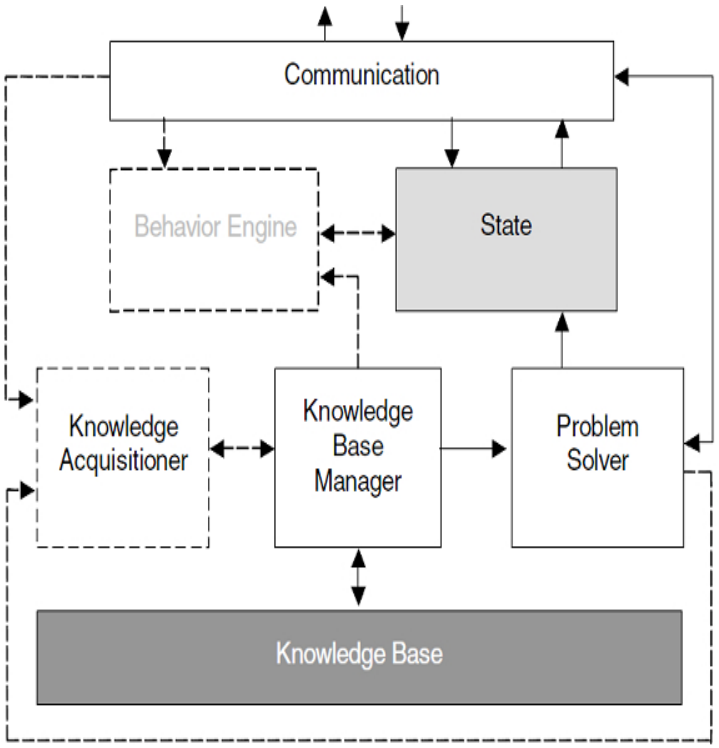

Figura 2.2: GPA Pattern

Veniamo ora a descrivere nel dettaglio le varie parti che costituiscono lo schema di "agente pedagogico" sopra riportato.

- Il Problem Solver è un modulo di controllo ed un decision maker che decide se un attore deve entrare in azione,cosa deve fare, a quale sti-

molo deve reagire. In particolare si occupa di stabilire quando l'agente passa da uno stato all'altro.

- L'interfaccia Communication ha il compito di percepire l'ambiente di learning dinamico e agire su di esso (le percezioni includono riconoscere situazioni nelle quali gli agenti pedagogici possono intervenire, come un'azione dello studente, un progresso di un co-learner e la disponibilità di informazioni desiderate). In particolare nel presente progetto si evincono due tipologie di comunicazione (agente-agente, agente-utente), le quali richiedono modi di comunicare e tecnologie completamente diverse, a causa delle diversità fra i due diversi fruitori della comunicazione.

- State è un'astrazione generale per molti tipi di stati interni degli agenti pedagogici; può contenere anche emozioni e caratteristiche riguardanti la personalità di un agente e anche diversi stati mentali dell'agente e comportamenti verbali. Inoltre può contenere dei valori di parametri che rappresentano le relazioni che è in grado di instaurare con altri agenti, in accordo con i suoi goal. Include la sua disposizione, l'attitudine all'insegnamento, determina se un agente impara facendo un ragionamento veloce o impiegandoci più tempo.

- Nel nostro caso una caratteristica di tale genere è presente in ogni agente co-learner: ognuno di essi, infatti, dovrà avere, da un punto di vista comportamentale, un diverso livello di predisposizione rispetto alla volontà di "tirare a caso" una risposta della quale non conosce la risposta.

- Per quanto riguarda i blocchi Knowledge Base e Knowledge Base Manager bisogna dire che essi, nel nostro caso, sono presenti sono all'interno dei coLearners, e non nel teacher. La base di conoscenza di ogni co-learner dovrà contenere i concetti appresi durante tutte le lezioni precedenti; la parte dell'agente che farà da manager di tale KB sarà quella che permetterà al co-learner stesso di memorizzare nuovi concetti e di interrogarsi relativamente al valore di verità di un'affermazione.

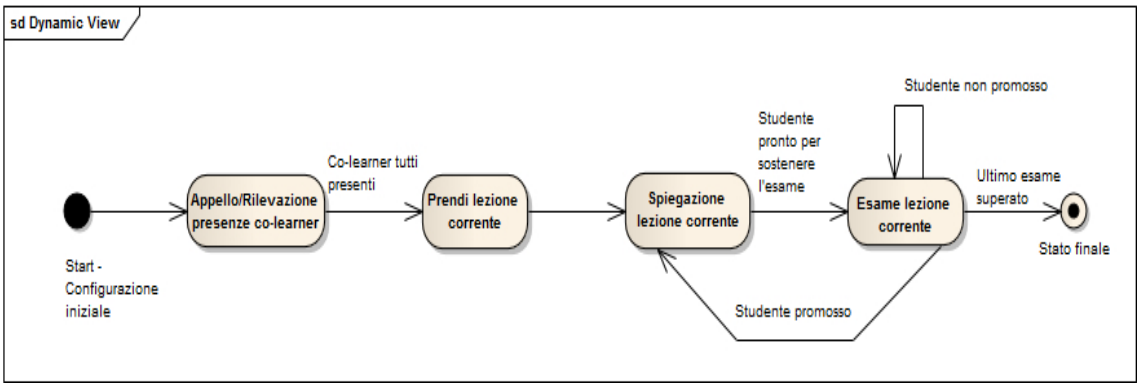

Figura 2.3: Macchina a stati rappresentante il comportamento interno dell'agente teacher

2.2.1 Comportamento del teacher

Nello stato iniziale vi è la configurazione iniziale del Teacher e l'appello (meccanismo di sincronizzazione iniziale tramite il quale il teacher aspetta che tutti gli agenti co-learner siano pronti a partire; in questo modo dovrebbe essere possibile far partire indifferentemente prima il teacher o i co-learners, senza problemi).
Dopo di questo si passa ad un comportamento ciclico uguale per ogni lezione, in cui ad ogni iterazione si rendono disponibili i contenuti relativi alla lezione corrente sia ai coLearners che all'utente; una volta che l'utente si ritiene preparato può passare a sostenere l'esame; anche i co-learners dovranno rispondere alle stesse domande. A questo punto il teacher valuta la prova, e se quella dello student risulta sufficiente prosegue spiegando la lezione successiva, altrimenti effettua un ripasso delle domande a cui la classe (sia gli agenti che lo studente) ha risposto in modo errato, e ripropone all'intera classe un altro esame da sostenere relativo alla stessa lezione. Fino a quando la valutazione dello studente non è sufficiente il sistema non può transitare alla lezione successiva. Alla fine del corso, quando lo student ha superato anche l'ultima prova, il teacher avverte gli altri agenti che il corso è terminato, e mostra allo student il voto finale del corso dato dalla media di tutti i voti ottenuti in ogni esame superato, infine chiude sé stesso.

2.2.2 Comportamento dei co-learners

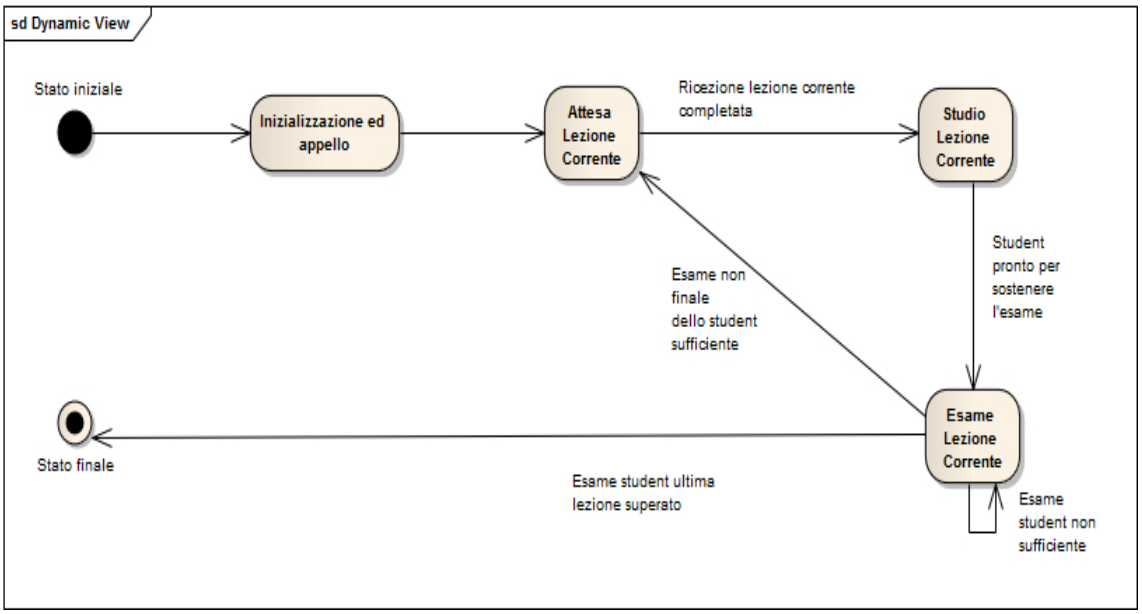

Figura 2.4: Macchina a stati rappresentante il comportamento interno degli agenti co-learner

All'avvio dell'agente esso entra nella fase di inizializzazione e di appello; finita questa fase anch'esso comincia ad eseguire un comportamento ciclico. Ad ogni iterazione esso studierà la lezione corrente e poi, quando l'utente sarà pronto a sostenere l'esame, anch'esso lo farà. Inoltre il co-learner eletto come "vicino di banco" dello studente deve anche rispondere alle eventuali domande da parte dell'utente stesso; questo è possibile sempre, tranne quando ci si trova in sede d'esame.

Capitolo 3

Progetto

3.1 Architettura

La figura 3.1 mostra l'idea che sta alla base dell'architettura del nostro sistema di e-learning.

Gli agenti sono in grado di comunicare tramite il servizio di messaggistica presente nell'ambiente Jade [3]; in particolare tali scambi di messaggi avverranno fra il teacher ed ognuno dei co-learner e viceversa. Per quanto riguarda invece le comunicazioni fra l'utente e gli agenti, il mezzo di comunicazione diretto è quello della chat (una verso l'insegnante ed una verso il "vicino di banco"), mentre, per quanto riguarda la pubblicazione, da parte del teacher, delle lezioni, man mano che il corso procede, essa si avvarrà della pagina content dell'ambiente Eledge (si veda l'appendice B.1). Infine possiamo notare l'esistenza di un'ulteriore GUI, che appare soltanto in corrispondenza di un esame: essa conterrà il compito con le domande a cui l'utente dovrà rispondere.

3.2 Interazione via Eledge

Come già accennato, per realizzare l'ambiente di apprendimento è stato scelto di utilizzare Eldge, un LMS sviluppato dall'università dello Utah. Eledge è stato sviluppato come framework per l'E-learning, in modo da rendere accessibile via Internet il materiale dei vari corsi agli studenti. Nel nostro progetto esso è lo strumento attraverso il quale l'agente teacher riesce

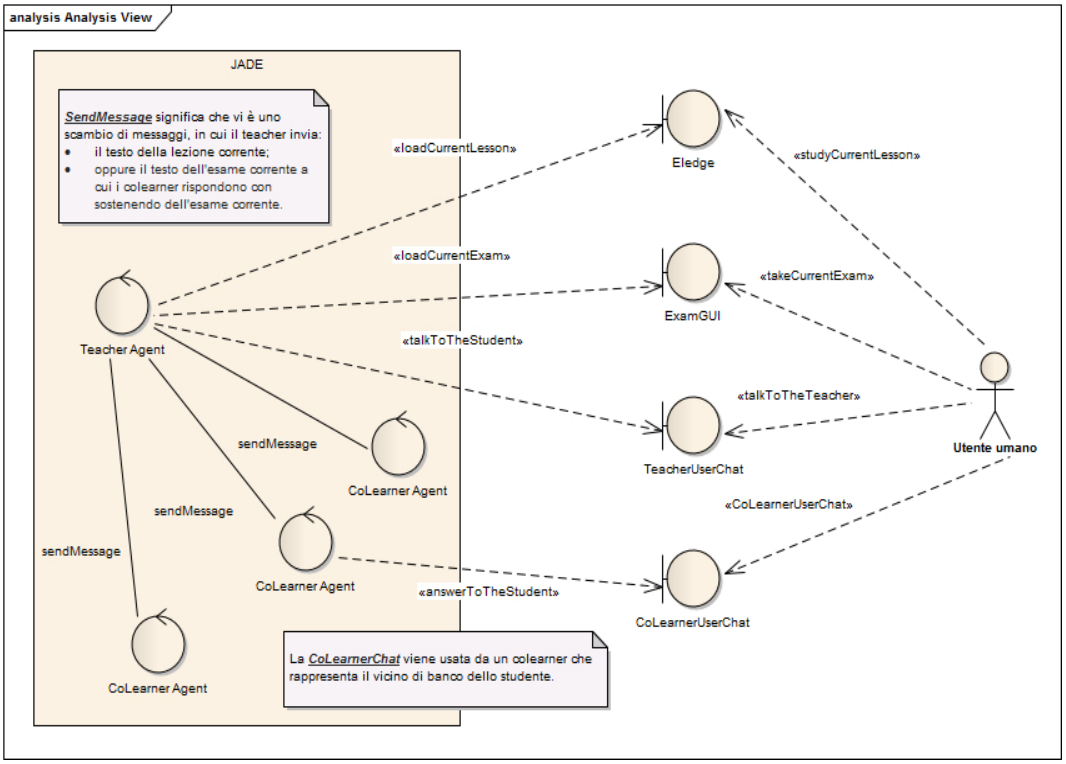

Figura 3.1: Architettura

ad interagire con lo student. Esso rappresenta un contenitore all'interno del quale il teacher inserisce gli artefatti utili per l'apprendimento. Gli artefatti, nel nostro caso, sono le varie lezioni, realizzate attraverso pagine html.

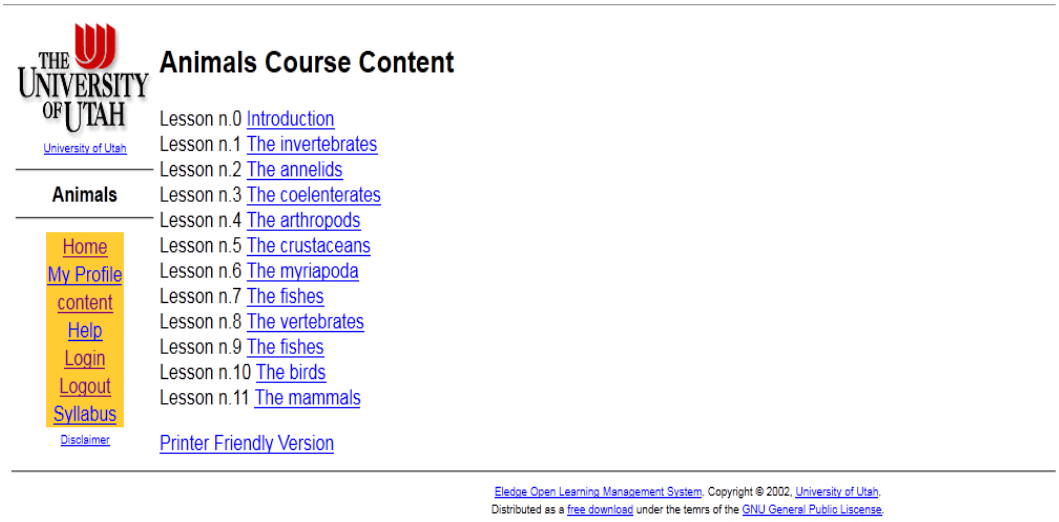

Figura 3.2: Come si presenta la pagina content di Eledge alla fine del corso

Il funzionamento di Eledge si basa sul meccanismo delle servlet java; di conseguenza è ovvio che il modo più naturale per interagire con esse è proprio quello di comunicare tramite messaggi Http. L'agente teacher, quindi, pubblica le lezioni su Eledge inviando messaggi http con metodo Post, indicando i valori dei campi che normalmente sarebbero riempiti in modo grafico attraverso le rispettive form html. Per le comunicazioni Http nel presente progetto abbiamo scelto di utilizzare la libreria httpclient di Apache.

In fase di progetto si è deciso che i contenuti del corso fossero costituiti da pagine html: ogni volta che l'insegnante deve spiegare una lezione pubblica il link alla pagina corrispondente nella pagina content di Eledge. Ogni lezione è propedeutica alla successiva ed allo studente per poter accedere al corso non è richiesta nessuna conoscenza pregressa. Lo studente, quindi, inizialmente vedrà solamente il contenuto della lezione 0; in seguito, se verrà promosso nell'esame di questa lezione, il teacher gli caricherà anche il link alla lezione 1, e così via.

3.3 Interazione via Chat

Lo studente interagisce sia con il teacher che con il co-learner (vicino di banco) attraverso l'uso di due chat:

- una in cui lo studente può fare delle domande al co-learner, le quali devono essere poste in lingua inglese e non in forma interrogativa, bensì sempre come frasi complete in forma affermativa alle quali ci si aspetta una risposta di tipo "Vero", "Falso", "Non lo so";

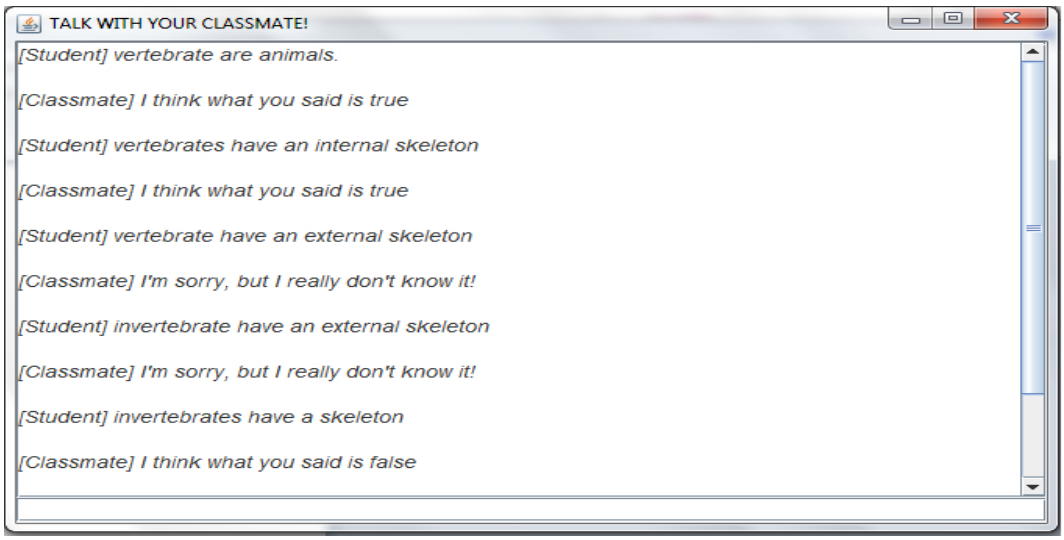

Figura 3.3: Chat fra il classmate ("vicino di banco") e lo student

- una attraverso la quale si scambiano delle informazioni teacher e student e in particolare lo student fa presente al teacher che è pronto a sostenere l'esame.

Il vero insegnante del corso, ovvero colui che si preoccupa di preparare il materiale per le lezioni, per specificare gli esami delle lezioni dovrà scrivere un file di testo in cui sono presenti le domande e le risposte di ogni esame secondo una specifica sintassi. Al fine di rendere semplice e gradevole il sostenimento degli esami da parte dello student abbiamo realizzato un'interfaccia grafica ad hoc con un layout simile a quello che normalmente hanno i test a risposta chiusa; esso, nel nostro caso, riporta le tre tipologie di domande già menzionate:

- una domanda vero/falso;
- una domanda con due risposte mutuamente esclusive;
- una domanda con risposte a scelta multipla.

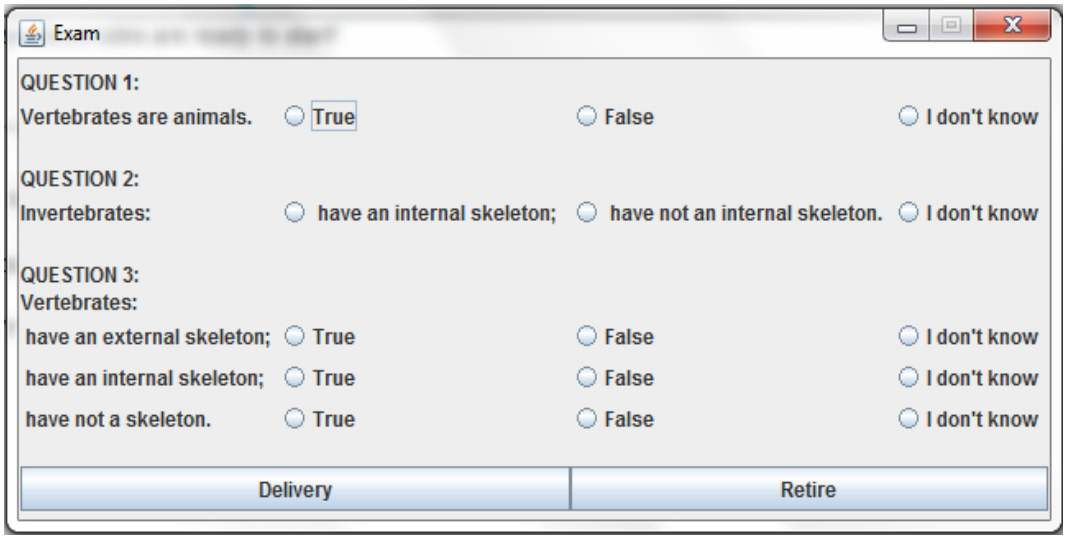

Figura 3.4: Interfaccia utilizzata per il sostenimento degli esami

Alla fine del corso verrà visualizzato il voto finale, ottenuto come media aritmetica dei voti di promozione ottenuti in ogni singolo esame. Si noti che i casi di ritiro e di bocciatura non verranno ad influire sul voto di tale esame, né sul voto finale; soltanto il voto con cui si supera il test verrà preso in considerazione; non è possibile nemmeno risostenere un esame già superato al fine di migliorare il voto.

L'agente teacher, una volta proposto l'esame corrente sia ai co-learners che allo student, si metterà in attesa che essi consegnino le risposte; ogni volta che uno di essi consegna il proprio compito, all'interno del teacher viene creato un thread ExamEvaluator che ha il compito di correggerlo e di inserire il risultato all'interno di un monitor. Dopo che il teacher ha ricevuto tutti i compiti si rivolge anch'esso allo stesso monitor, ponendosi in attesa che tutti i compiti siano stati corretti; in pratica, attraverso il meccanismo del monitor, è stata realizzata una barriera; essa permette di

gestire il parallelismo di più threads che operano concorrentemente, in maniera indipendente dal thread principale del teacher.

Se lo studente umano viene promosso all'esame il corso procede con la lezione successiva.

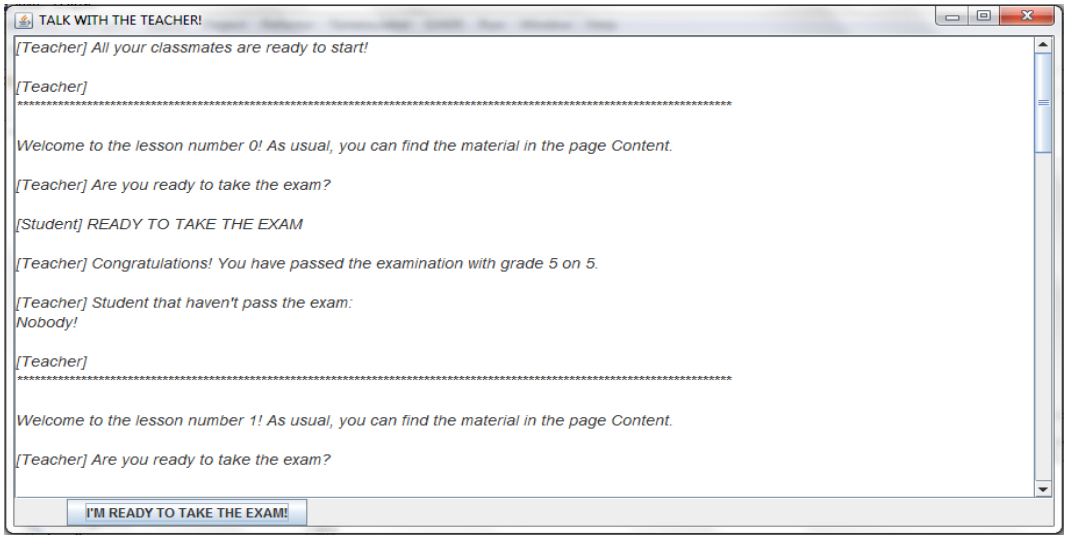

Figura 3.5: Esempio di chat fra il teacher e lo student in caso di esito positivo dell'esame

Se lo studente umano, al contrario, non dovesse essere promosso, allora il teacher organizzerà un ripasso relativamente a tale lezione, ed in seguito tutti ripeteranno tale esame.

E' dunque importante notare che l'avanzamento del corso, in ultima analisi, dipende solamente dalla qualità delle risposte dello student, e non di quelle dei co-learners; questo perché l'obiettivo dell'applicazione è quello di far imparare le lezioni all'utente, e non agli agenti, che sono creati solo perché utili a tale fine.

Il ripasso consiste nel ripetere, sia allo student che ai co-learners, tutte le domande che sono state da qualcuno sbagliate, correlate con la relativa risposta corretta.

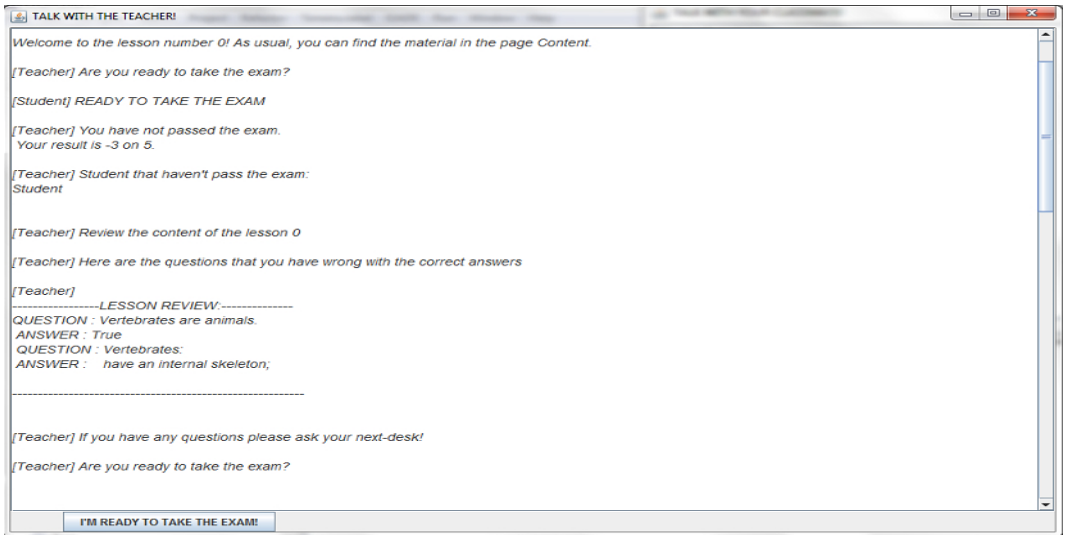

Figura 3.6: Esempio di chat fra il teacher e lo student in caso di esito negativo dell'esame

3.4 Modalità di apprendimento dei co-learners

Un altro punto importante del progetto è la questione dell'apprendimento da parte degli agenti co-learner. Per realizzarlo ci siamo avvalsi di una suite di tools per il NLP (Natural Language Processing) realizzata dall'università dello Stanford. Per il funzionamento di Stanford NLP si fa riferimento al relativo appendice C.1.

L'apprendimento delle lezioni da parte del singolo co-learner si sviluppa attraverso questi passi:

- il teacher invia al co-learner un messaggio ACL contenente il testo di tutta la lezione; la comunicazione fra i colearner ed il teacher avviene sempre, infatti, tramite il servizio di messaggistica presente nell'ambiente Jade [3];
- le singole frasi della lezione vengono elaborate dallo Stanford CoreNLP, che in uscita produce, per ogni frase della lezione, una struttura dati di tipo SemanticGraph, che contiene l'analisi logica di tale frase (C.1);

- ogni SemanticGraph viene analizzato, estrapolando da esso i contenuti fondamentali della frase, che verranno poi a loro volta mappati in un insieme di triple rdf all'interno della base di conoscenza dell'agente.

Ogni co-learner ha infatti una base di conoscenza realizzata come grafo RDF. Per la manipolazione di grafi RDF abbiamo utilizzato la libreria JRDF (`http://jrdf.sourceforge.net/`). Ogni tripla rappresenta un concetto che l'agente ha appreso durante una spiegazione del teacher. Come abbiamo detto anche in appendice, la struttura dati SemanticGraph non contiene alcuna informazione di tipo semantico relativamente al significato reale della frase, ma si limita a contenere le relazioni che determinano l'analisi logica della frase stessa. E' stato quindi nostro compito quello di mappare alcuni dei costrutti linguistici più comuni in una forma standard all'interno della nostra KB. Il co-learner non la ha capacità di dimostrare verità a partire dalle informazioni contenute nella sua KB, perché non sono state implementate in esso regole come la deduzione matematica; non sarà in grado, ad esempio, partendo dal fatto che un cane è un quadrupede e che i quadrupedi hanno quattro zampe, a dire che i cani hanno quattro zampe. L'utilizzo dei tool di Stanford, e la memorizzazione dei concetti fondamentali in una forma standard (uguale sia nella memorizzazione delle lezioni che nella comprensione delle domande d'esame) permettono tuttavia di rendere più flessibile il matching fra il modo in cui viene espressa la lezione ed il modo con cui viene formulata la domanda. Ad esempio, se una lezione contenesse la frase "Dogs are beautiful animals", ed una domanda d'esame dicesse "My dog is an animal", la risposta dell'agente sarebbe "Vero". E' dunque chiaro che il co-learner lavora ad un livello di astrazione più alto, rispetto a quello che si avrebbe dal semplice confronto sintattico fra due frasi.

L'agente ha una capacità di apprendimento limitata solo a quei costrutti fondamentali del linguaggio naturale che abbiamo implementato. Questi sono mappati in una o due triple per frase. La prima tripla, che a meno di casi particolari dovrebbe essere sempre presente, assume uno di questi significati:

- soggetto + verbo essere + aggettivo;
- soggetto + verbo essere + sostantivo;
- soggetto + verbo avere + complemento oggetto;

- soggetto + verbo generico + complemento oggetto;
- soggetto + verbo generico + complemento indiretto.

Sono state inoltre gestite, per ognuna di queste tipologie, anche le affermazioni in forma negativa. La seconda tripla, che può essere presente o meno, può invece assumere uno di questi significati:

- complemento oggetto/indiretto + (relazione di attributo) + aggettivo (non numerico)
- complemento oggetto/indiretto + (relazione di attributo) + aggettivo numerico

Una volta eseguita la configurazione viene eseguito un appello durante il quale l'agente teacher si registra in Jade [3] come servizio e si mette in attesa di ricevere un messaggio di presenza per ogni co-learner, quando li avrà ricevuti tutti potrà partire con la spiegazione della prima lezione. I co-learners inviano a polling un messaggio di presenza fino a quando non avranno ricevuto la prima lezione da studiare. Si noti che questo meccanismo funziona in qualsiasi ordine con cui vengono avviati gli agenti.

I co-learners e lo student non utilizzano lo stesso tipo di artefatto per studiare le lezioni. Per gli studenti umani i contenuti didattici vengono visualizzati tramite pagine html; per gli agenti sono invece inseriti in un file di testo semplice. La differenza non è solo nel formato del file, ma è anche nella struttura delle frasi che compongono le lezioni. Si è scelto infatti di facilitare l'apprendimento dell'agente inserendo i contenuti informativi sotto forma di frasi che rispecchiano meglio la sua modalità di elaborazione del linguaggio naturale (in pratica sono frasi semplici).

Nella nostra KB è stato esplicitamente gestito il concetto di forma negativa del predicato di una frase, allo scopo di poter differenziare il caso in cui la risposta è "Falso" dal caso in cui la risposta, semplicemente, è sconosciuta all'agente. Ciò non sarebbe stato possibile tramite il cosiddetto approccio "a mondo chiuso", per il quale se un dato non è presente significa che rappresenta un concetto falso (e non esiste l'idea di poter rispondere "Non lo so"). Una volta che il co-learner ha finito di apprendere la lezione, esso è pronto a sostenere l'esame. Per rispondere alle domande che gli vengono poste, esso interroga la propria base di conoscenza. Tale interrogazione si

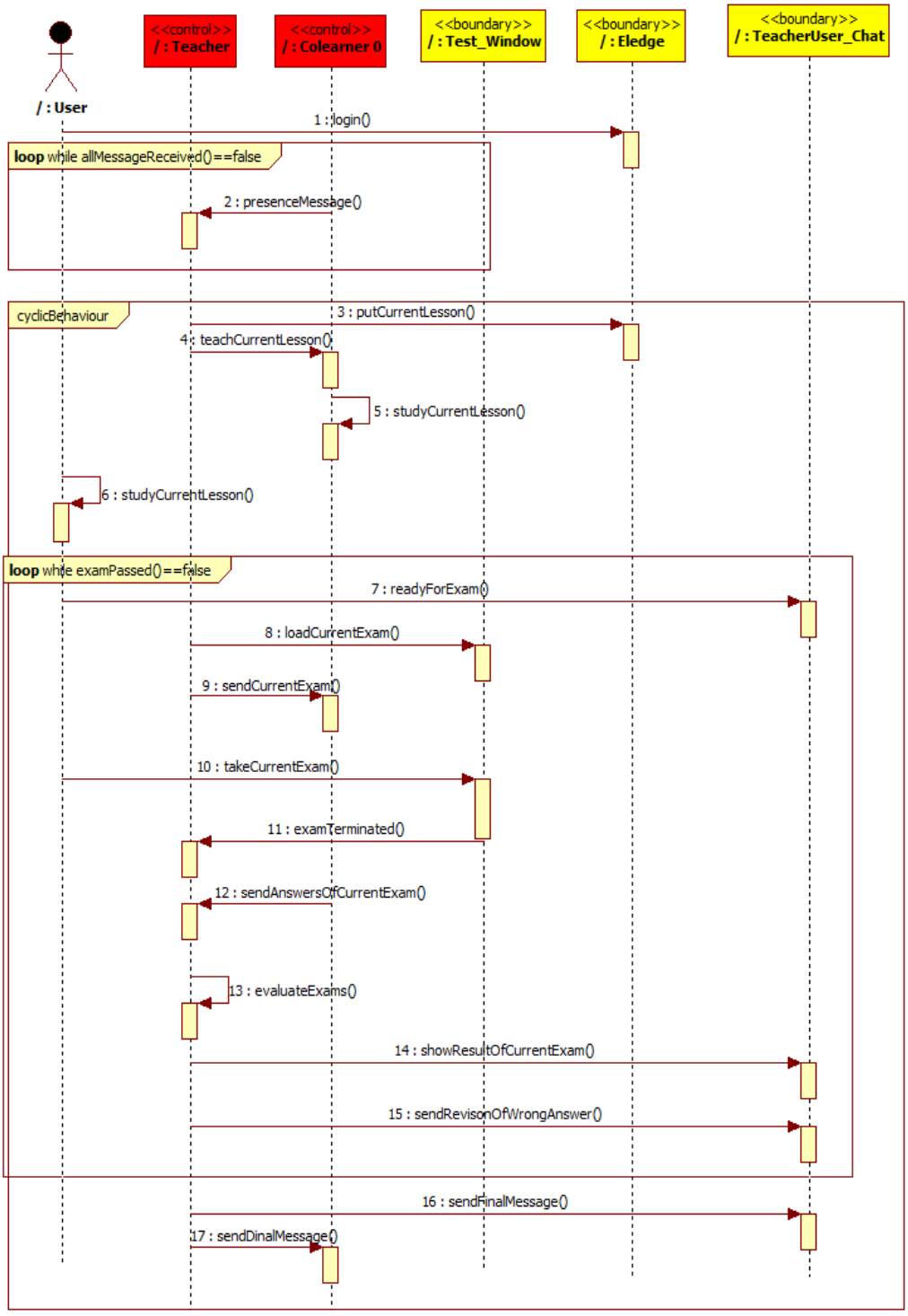

Figura 3.7: Sequence diagram

sviluppa in due query: in una si chiede se l'affermazione è vera, nell'altra si chiede se è vera la sua forma negata. Questo perché, sia nel caso in cui nella KB entrambe le forme non siano presenti (normale caso di non conoscenza), sia nel caso in cui sono presenti entrambe (dovuto ad un insegnamento contraddittorio), la risposta dovrà essere "Non lo so". Altrimenti, se la domanda è presente solo nella sua forma naturale la risposta sarà "Vero", se invece è presente solo nella sua forma negata la risposta sarà Falso.

A questo momento entra in gioco il parametro, presente in ogni co-learner, che rappresenta il grado di azzardo dell'agente, ovvero quanto esso vuole rispondere a domande di cui non conosce la risposta. In generale, più il grado di azzardo dell'agente è elevato più esso è tentato a rispondere in modo casuale a tali domande, viceversa più è basso più è propenso a rispondere "Non so". Si è deciso che gli agenti non possano ritirarsi all'esame, questo perché in ogni caso il loro voto non risulta significativo per il procedere del corso. Più che il voto, degli agenti è importante conoscere le risposte sbagliate, perché queste vengono aggiunte fra i contenuti che il teacher deve far ripassare a tutti gli studenti, in caso di fallimento del test da parte dello Student. In sede di ripasso il teacher ripropone a tutti il testo delle domande che sono state sbagliate da qualche studente (agente o utente); allo Student mostrerà dunque le coppie domanda-risposta corretta nella chat, mentre ai co-learners invierà tramite messaggi ACL direttamente le frasi che contengono i concetti corretti.

3.5 Interpretazione dal punto di vista del metamodello A&A

3.5.1 Agenti

Gli agenti presenti nel nostro MAS appartengono, come abbiamo più volte ripetuto, a due tipologie: c'è un teacher e ci sono diversi co-learners.

Per svolgere il goal di correzione dei compiti d'esame il teacher si avvale di un monitor, attraverso il quale riesce a capire quando i compiti sono stati tutti corretti. In base al risultato dell'esame dello student, il teacher modifica l'ambiente Eledge in modo differente. Nel caso in cui l'utente

venga promosso, esso esegue le operazioni di inserimento dei nuovi contenuti in Eledge e di invio delle lezioni ai co-learners; in caso contrario elabora i contenuti per il ripasso e predispone l'environment per tale scopo.

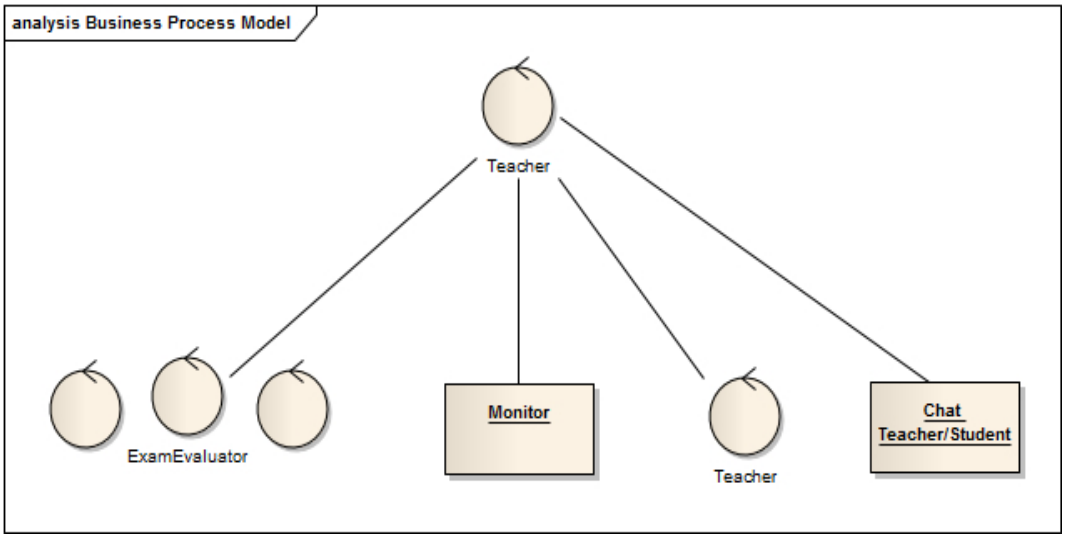

Figura 3.8: Livelli di astrazione del teacher

I principali goals degli agenti co-learner sono invece quelli di apprendere le lezioni e di sostenere gli esami. L'apprendimento dei concetti contenuti nelle lezioni è una azione interna all'agente, in quanto non modifica l'ambiente, bensì modifica la sua base di conoscenza. Per il sostenimento dell'esame, invece, oltre a ricevere il testo come messaggio, deve occuparsi anche di notificare al teacher le proprie risposte; la correttezza di queste ultime potrà influenzare l'ambiente, al momento del ripasso.

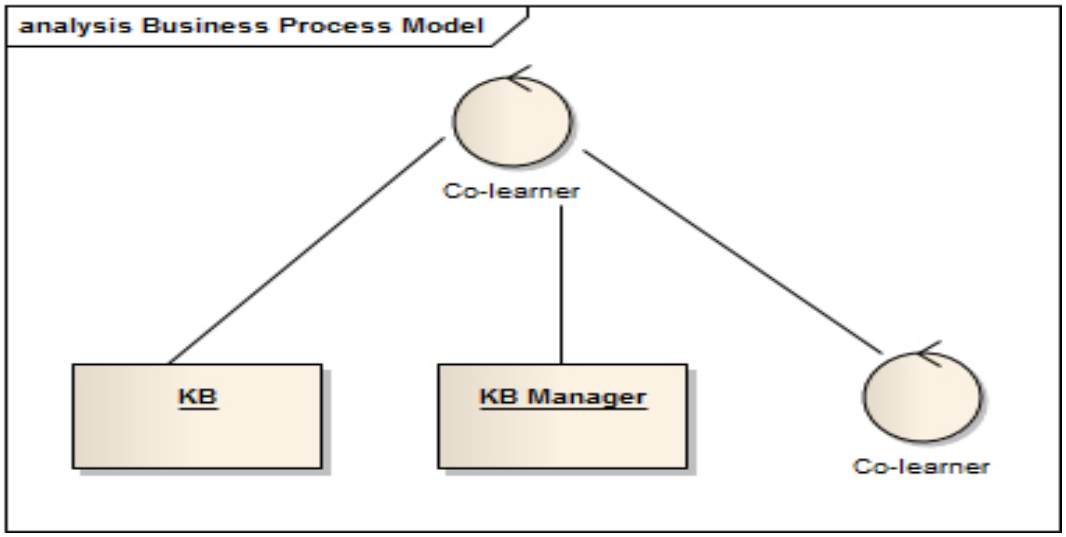

Figura 3.9: Livelli di astrazione dei co-learners

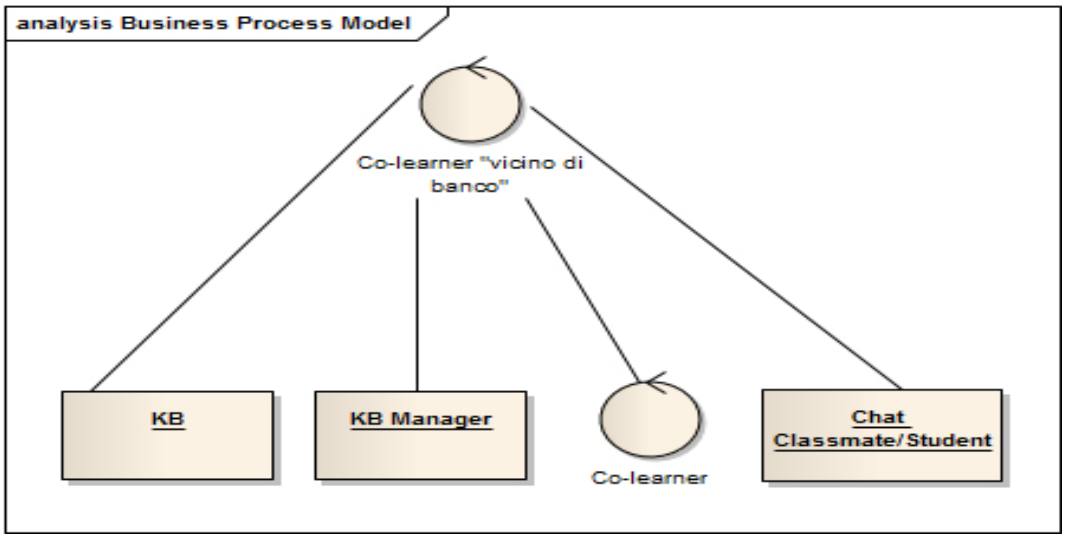

Figura 3.10: Livelli di astrazione del co-learner “vicino di banco”

3.5.2 Environment e artefatti

In un MAS l'environment è un'astrazione di prima classe. Nel nostro sistema esso è costituito da un insieme di artefatti di diversa natura che supportano gli agenti nello sviluppo del corso e nell'apprendimento.

Principalmente esso consiste di tre artefatti:

- il Learning Management System Eledge;
- la chat di comunicazione fra il teacher e lo student;
- la chat di comunicazione fra lo student ed il suo vicino di banco.

Eledge, in qualità di artefatto del MAS, dispone delle seguenti interfacce:

- login() : il teacher (e l'utente) effettuano il login;
- putContent() : il teacher inserisce il contenuto delle lezioni;
- (getContent() : lo student recupera i contenuti delle lezioni tramite richieste http).

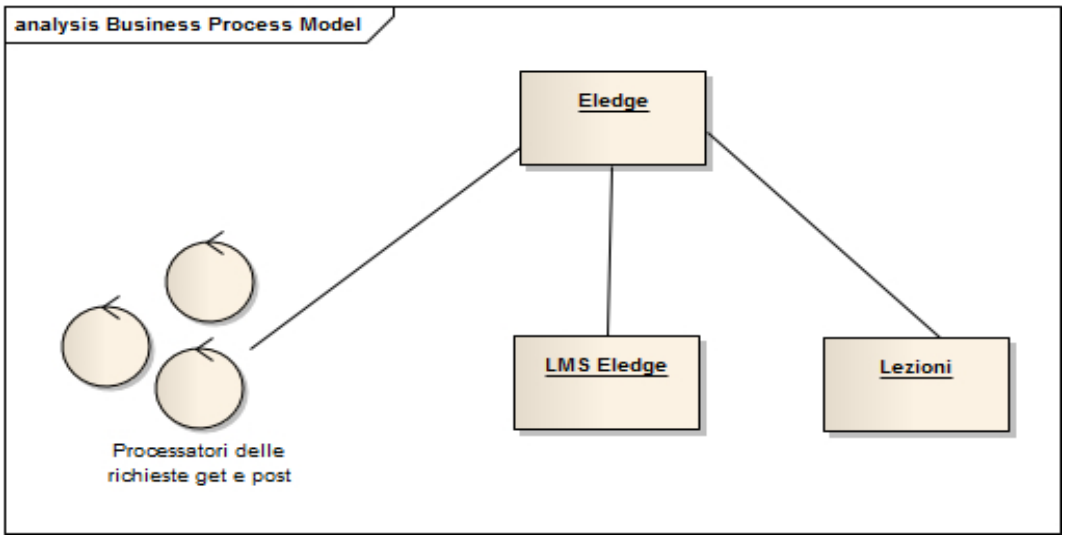

Figura 3.11: Livelli di astrazione del LMS Eledge

Si noti che entrambe le prime due interfacce sono rivolte al teacher; questo perché i co-learner non utilizzano mai questo framework (Eledge è l'ambiente che permette lo scambio dei contenuti informativi fra il teacher e lo student). Ad un livello di astrazione inferiore si vede, infatti, che Eledge, oltre a presentare all'utente l'idea di un ambiente virtuale per l'apprendimento, deve (nel nostro sistema) occuparsi di questi tre compiti fondamentali:

- gestire gli account (in particolare per poter distinguere lo studente dall'insegnante);
- raccogliere i contenuti del corso, ovvero le lezioni;
- processare le richieste get e post dello studente e del teacher, al fine di rendere fruibili tali contenuti agli utenti del sistema.

Oltre a Eledge, gli altri due grandi artefatti presenti nell'environment sono le due chat di comunicazione. L'interfaccia che entrambe espongono verso i rispettivi agenti è costituita dalla sola operazione di appendere il messaggio corrente all'interno dell'area di testo. Dal punto di vista dello student, invece, la chat con il vicino di banco permette di porgli un domanda in linguaggio naturale e di leggere la risposta, mentre la finestra verso il teacher gli permette di leggere le comunicazioni inserite da tale agente e di avvertirlo del fatto che egli è pronto a sostenere l'esame.

Capitolo 4

Conclusioni

Tramite il sistema realizzato pensiamo di aver messo in luce che l'ambito applicativo dell'E-learning può fare proficuamente uso dell'astrazione di Sistema Multi-Agente, soprattutto dato che i concetti di insegnante e di compagni di classe sono mappabili in modo naturale, all'interno di un sistema software, su quello di agente. Ciò è vero non solo perché tali concetti portano a pensare ad entità software sia computazionalmente attive che proattive, ma anche perché le features di apprendimento e di risposta intelligente alle domande porta nell'ottica di agenti "strong".

Il sistema presentato si può prestare, in ogni caso, all'aggiunta di nuove features:

- negli agenti co-learner, all'interno del KB Manager, si potrebbe implementare un meccanismo di memorizzazione e di utilizzo di regole di inferenza, di tipo deduttivo, induttivo, o abduttivo che siano; ciò porterebbe ad una maggior capacità nel saper rispondere alle domande poste. Anche nella versione attuale, comunque, la capacità mostrata dai co-learners di rispondere alle domande è sufficiente, rispetto agli scopi dell'applicazione.

- attualmente se l'utente del sistema fosse interessato ad una specifica lezione potrebbe solamente seguire l'intero corso, al limite chiudendo l'applicazione una volta sostenuto l'esame relativo alla lezione in questione; questo è normale, in quanto il nostro sistema si mette nell'ottica di proporre un intero "corso virtuale"; una possibile estensione, co-

munque, potrebbe consistere nella possibilità di seguire direttamente, all'interno del corso, singole lezioni a cui si è interessati. Ciò porterebbe il sistema nell'ottica di un corso composto da "Learning Objects", concetto, questo, ben noto in letteratura (si veda l'appendice A.2); ogni lezione avrebbe un descrittore, e bisognerebbe anche gestire le propedeuticità fra le diverse lezioni (nel sistema attuale le propedeuticità sono date in modo implicito da come il gestore del corso ha ordinato le lezioni).

Appendice A

E-learning

A.1 Definizione

Una definizione condivisa nella comunità scientifica è quella proposta dall'ASFOR (Associazione Italiana per la Formazione Manageriale):

"Metodologia didattica che offre la possibilità di erogare contenuti formativi elettronicamente (e-learning) attraverso Internet o reti Intranet. Per l'utente rappresenta una soluzione di apprendimento flessibile, in quanto fortemente personalizzabile e facilmente accessibile. Il termine e-learning copre un'ampia serie di applicazioni e processi formativi, quali computer based learning, Web-based learning e aule virtuali.

In effetti, sviluppare un sistema di e-learning significa sviluppare un ambiente integrato di formazione utilizzando le tecnologie di rete per progettare, distribuire, scegliere, gestire e ampliare le risorse per l'apprendimento. Le modalità più utilizzate per realizzare tale integrazione sono:

- l'autoapprendimento asincrono attraverso la fruizione di contenuti preconfezionati disponibili sulla piattaforma di erogazione;
- l' apprendimento in sincrono attraverso l' utilizzo della video-conferenza e delle aule virtuali;
- l'apprendimento collaborativo attraverso le attività delle comunità virtuali di apprendimento."

A.2 I Learning Objects

Nel nostro progetto ci siamo concentrati sulla prima, fra le precedenti tre modalità. In letteratura si è da anni affermato il concetto di *Learning Objects* (LO). L'idea di base è che i contenuti siano strutturati sotto forma di unità didattiche che nel loro complesso costituiscono un argomento completo. Un Learning Object è la più piccola entità che compone il contenuto di un corso dotata di senso compiuto dal punto di vista della formazione. Dall'aggregazione dei LO nascono le unità didattiche, che compongono i moduli, che a loro volta formano i corsi.

Nel presente progetto comunque non è stato applicato il concetto di Learning Object; anche se ogni pagina html contenente una lezione potrebbe essere vista come tale, in realtà un LO deve avere con sé un descrittore, il cui formato è stato oggetto della redazione di più di uno standard; ciò non avviene nel nostro caso, ma, come spiegato nelle conclusioni, potrebbe essere oggetto di una futura espansione del nostro progetto.

Appendice B

Eledge Open Learning Management System

B.1 Caratteristiche

Una piattaforma per l'E-learning è un software che permette di creare un ambiente virtuale di apprendimento all'interno del quale è possibile erogare corsi di formazione, gestire e monitorare i percorsi formativi degli utenti e accedere a una serie di strumenti di comunicazione e di servizi collegati. Le piattaforme orientate alla gestione di tutto il processo formativo sono denominate Learning Management System (LMS). Il LMS utilizzato nel nostro progetto è Eledge.

Eledge è un LMS sviluppato dall'Università dello Utah distribuito con licenza Open GPL GNU. Altre caratteristiche sono che è compatibile con tutti i sistemi operativi, consiste in una raccolta di servlet Java che utilizzano un database MySQL back-end per memorizzare le informazioni e contenuti dei corsi.Il sistema viene mandato in esecuzione dal web server Tomcat. L'interfaccia grafica è ridotta ed essenziale ma veloce e semplice da usare. Essa è divisa principalmente in due colonne. A sinistra è posto il menù con il link alle operazioni possibili, mentre a destra viene presentato il contenuto, spesso in semplice formato testuale. Una volta effettuato il login, il sistema riconosce se è in corso la sessione dell'insegnante o di un alunno ed in base a questa informazione provvederà ad abilitare operazioni diverse. Il ruolo di insegnante viene attribuito automaticamente all'utente che si registra per primo al corso, tutti gli altri sono studenti. Il compito

dell'insegnante è quello di caricare le unità didattiche sul sito, stabilire delle date in cui svolgere l'esame relativo all'unità didattica in corso e monitorare l'andamento della classe.

B.1.1 Home page

La home page del corso che si trova all'indirizzo `http://localhost:8080/ART1010.Home` si presenta in questo modo:

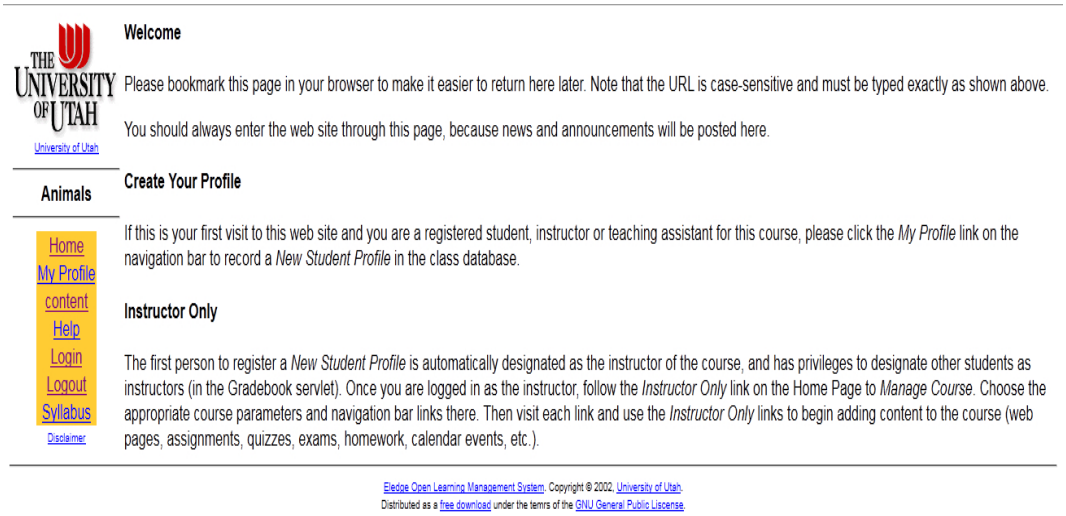

Figura B.1: Eledge Home page

Nel menù principale che si trova a sinistra ci sono le seguenti voci:

- *Home*, in questa pagina vi è qualche informazione di carattere pratico su Eledge e vi è la possibilità per l'utente teacher di modificare l'aspetto generale del corso (il nome, il logo,come si presenteranno le pagine Web e quali voci mostrare o nascondere nel menù principale);
- *My Profile*, qui si possono creare i vari profili e modificare quelli esistenti. Si ricordi che l'utente con i permessi di teacher è il primo utente che viene registrato al corso;
- *Content*, qui si possono gestire tutti i contenuti del corso (solo per il teacher) e visualizzarli (per tutti gli utenti);

- *Help*, qui ci sono delle FAQ su Eledge;
- *Login*;
- *Logout*;
- *Syllabus*, qui ci sono dei riferimenti del corso, sul teacher, materiale di riferimento.

B.2 Interazione con Eledge nel progetto

Come abbiamo detto, il funzionamento di Eledge si basa sul meccanismo delle servlet Java; di conseguenza è ovvio che il modo più naturale per interagire con esse è proprio quello di interagire con esse tramite messaggi Http. L'agente teacher, quindi, pubblica le lezioni su Eledge inviando messaggi http con metodo Post, indicando i valori dei campi che normalmente sarebbero riempiti in modo grafico attraverso le rispettive form html. Per le comunicazioni Http nel presente progetto abbiamo scelto di utilizzare la libreria httpclient di Apache. In fase di progetto si è deciso che i contenuti del corso fossero costituiti da pagine html: ogni volta che l'insegnante deve spiegare una lezione pubblica il link alla pagina corrispondente nella pagina content di Eledge.

Appendice C

Stanford CoreNLP

C.1 Caratteristiche

Stanford CoreNLP è un ambiente di programmazione che fornisce, anche tramite API, un insieme di strumenti di analisi del linguaggio naturale. Ciò avviene prendendo in input delle frasi in lingua Inglese e restituendo l'analisi grammaticale e logica di tali frasi; riconosce anche se vi sono nomi di aziende, persone, ecc; normalizza date, orari e quantità numeriche; evidenzia la struttura delle frasi, in termini di frasi e dipendenze tra le parole; è infine in grado di indicare quali sintagmi nominali (noun phrases) si riferiscono alla stessa entità. Esso fornisce dunque le basi fondamentali necessarie alle applicazioni che devono essere in grado di comprendere testo scritto in linguaggio naturale.

C.1.1 Annotazioni

L'obiettivo di questo tool è di permettere di ottenere in modo rapido e semplice una completa serie di annotazioni di testi in linguaggio naturale. Tali annotazioni vengono applicate in sequenza, secondo uno schema architetturale di tipo "pipeline". Alcuni tipi di annotazione del testo presenti sono:

- tagging delle parti del discorso ("part-of-speech tagging" / POS-tagging);
- riconoscimento di entità con nome proprio (NER);

- parsing;
- sistema di coreferenziazione fra parole.

Tramite tali annotazioni si ottiene, in uscita al parser, una struttura dati che rappresenta, al livello di astrazione da noi usato, l'analisi logica della frase. Per esempio NSub(Paolo,tall) significa che Paolo è il soggetto della frase, e che egli è "tall".

C.1.2 SemanticGraph

La struttura dati in questione si chiama SemanticGraph : a dispetto del nome, però, essa non contiene alcuna informazione relativa alla semantica della frase, ma, come già detto, contiene solamente le relazioni logiche che sussistono fra le varie parti della frase. È stato progettato per essere altamente flessibile ed estensibile, cioè con una sola opzione è possibile selezionare i tipi di annotazione da abilitare o meno.

C.2 Open Source Software

Il codice di Stanford CoreNLP è sotto la licenza GPL completa, che permette il suo utilizzo per scopi di ricerca, progetti di software libero, servizi di software, ecc, ma non in distribuzione software proprietario. Il download è di 442 MB e richiede Java 1.5 +.

Appendice D

Installazione del sistema

Riportiamo di seguito la completa procedura necessaria per far funzionare il sistema con il corso d'esempio riportato. Si noti che l'installazione non è veloce, e che, comunque, se l'intenzione è quella di installarlo autonomamente nel proprio sistema, sarà necessario, prima di seguire i seguenti passaggi, scaricare le librerie al seguente indirizzo: `http://nlp.stanford.edu/software/corenlp.shtml` , e posizionarle nella cartella lib del progetto.

D.1 Installazione dei servizi di base e configurazione iniziale del sistema

1. Se non lo fosse, provvedere ad installare Java nel vostro sistema
2. Installare il DBMS MySql, e creare un database di nome art1010
3. Installare Apache Tomcat, e sostituire, in webapps, la cartella ROOT con quella fornita (nel caso si voglia creare un nuovo corso, sarà necessario cambiare il materiale educativo ivi contenuto)
4. Lanciare il programma java updateDB contenuto nelle classi della web application di cui sopra; ciò creerà, nel db art1010, le tabelle necessarie a Eledge per funzionare
5. Registrarsi in Eledge con le seguenti credenziali:
    - user: francy

- password: frafra86

6. A questo punto è necessario che il gestore del corso (il vero insegnante) inserisca, cliccando sul bottone "Manage Content Template" all'interno della pagina Content, i link relativi alle pagine html contenenti le lezioni del corso. Questi verranno salvati dal LMS, e l'agente Teacher, al momento opportuno, semplicemente li attiverà" rendendoli visibili all'interno della pagina Content come link html. Per creare ognuno di tali elementi (cioè per ogni lezione del corso) si segua il seguente procedimento:

   - cliccare su "Template Wizard"
   - scrivere "lezioneXXX", dove XXX è il numero della lezione, cominciando da 0
   - type: input
   - page location: bottom of page
   - cliccare " →"
   - section position: end (per inserirlo sempre sotto il precedente)
   - HTML preText: a piacere, si può scegliere di scrivere, ad esempio, "Lesson n.XXX"
   - HTML postText: <a href="ART1010/content/lezioneXXX.html" >Nome della lezione
   - default value: false
   - cliccare su " →"

D.2 Lancio dell'applicazione

Se si lavora sotto Linux o Mac basterà lanciare il file run.sh", mentre se si lavora sotto Windows bisognerà lanciare il file run.bat"; si noti che entrambi questi script si aspettano in ingresso un intero (consigliabilmente basso) che rappresenta il numero di agenti di tipo co-learner che saranno presenti nel sistema di E-learning.

Bibliografia

[1] Università dello Utah. http://eledge.sourceforge.net.

[2] Università dello Stanford. http://nlp.stanford. edu/software/index.shtml.

[3] Java Agent DEvelopment Framework (JADE). http://jade.tilab.com/.

[4] Libreria JRDF. http://jrdf.sourceforge.net/.

[5] Vladan Devedzic e Andreas Harrer. Architectural Patterns in Pedagogical Agents, Lecture notes in Computer Science, 2002, SpingerLink. http://www.springerlink.com/content/l420051205qt9628/.