

*UNIVERSITA' DEGLI STUDI DI PALERMO*



**Corso di Laurea in Ingegneria Informatica  
A.A. 2002-2003**

---

**Strumento web-based ad agenti  
per la produzione collaborativa di  
ipertesti multimediali**

---

*Tesina di Ingegneria del Software di:*

XXX

XXX

# INDICE ANALITICO

<b>Pianificazione del Progetto</b> .....	(incompleta)	4
1 Diagramma di Gantt.....		5
<b>Requisiti di Sistema</b> .....		8
Introduzione .....		9
1 Diagramma di GANTT .....		5
2 Overview del prodotto.....		9
3 Requisiti funzionali .....		11
4 Requisiti non funzionali .....		11
4.1 Interfaccia utente e fattore umano .....		11
4.2 Documentazione .....		12
4.3 Considerazioni hardware .....		12
4.4 Vincoli: pseudo-requisiti .....		13
5 Modelli del sistema .....		13
5.1 Scenari.....		13
5.1.1 Identificazione degli attori.....		13
5.1.2 Identificazione degli scenari.....		14
5.1.3 Descrizione degli scenari.....		14
5.2 Funzionalità del sistema .....		34
5.2.1 Domain Description.....		34
5.2.2 Descrizione dei casi d'uso .....		34
<b>Progetto del sistema</b> .....		45
1 Fase 1: Domain Description.....		46
2 Fase 2: Agent Identification.....		47
3 Fase 3: Role Identification .....		49
3.1 Ruoli identificati.....		64
4 Fase 4: Tasks Specification.....		67
4.1 Agente: Access Manager .....		67
4.1.1 Task: AccessApplication.....		67
4.1.2 Task: GiveVCard .....		68
4.1.3 Task: UsersManager .....		68
4.2 Agente: Author Gui .....		68
4.2.1 Task: ReceiveWorkingArguments .....		69
4.2.2 Task: ExportAssets .....		69
4.2.3 Task: RetrieveAsset .....		70
4.2.4 Task: UploadArguments .....		70
4.2.5 Task: SearchAssets .....		70
4.3 Agente: Common Gui.....		71
4.3.1 Task: ShowCommonWindows .....		72
4.3.2 Task: AccountManager .....		72
4.3.3 Task: LookWhoIsAround .....		72
4.3.4 Task: SendEmail.....		73
4.3.5 Task: SendMessage .....		73
4.3.6 Task: TakeMessage .....		73
4.3.7 Task: GetVCard .....		74
4.3.8 Task: MembersManager .....		74
4.3.9 Task: ReceiveArguments.....		74

4.3.10 Task: Contract Argument .....	75
4.3.11 Task: UpdateWorkingArguments .....	75
4.4 Agente: Ecs Gui .....	75
4.4.1 Task: AssignArguments .....	76
4.4.2 Task: ReceiveArgumentsNegotiation .....	77
4.4.3 Task: RequestDefineCourse .....	77
4.4.4 Task: RequestDeleteCourse .....	77
4.4.5 Task: SearchArgumentBehaviour .....	78
4.4.6 Task: RequestMetadataWrite .....	78
4.4.7 Task: RequestComposeCourse .....	78
4.4.8 Task: NewMember .....	79
4.4.9 Task: ShowEcsWindows .....	79
4.5 Agente: Editor .....	79
4.5.1 Task: ReceiveAssetsUpload .....	80
4.5.2 Task: CreateMetadataFile .....	80
4.5.3 Task: SendAsset .....	81
4.5.4 Task: ReceiveExportedArgument .....	81
4.5.5 Task: ComposeCourse .....	81
4.5.6 Task: ServeAssetSearch .....	82
4.6 Agente: Planner .....	82
4.6.1 Task: ReceiveCourseDefinition .....	83
4.6.2 Task: DeleteCourse .....	83
4.6.3 Task: ServeArgumentSearch .....	83
5 Fase 5: Ontology Description .....	83
5.1 Domain Ontology Description .....	84
5.1.1 Ontologie .....	90
5.2 Communication Ontology Description .....	98
5.2.1 Conoscenza degli agenti .....	98
5.2.2 Contenuto dei messaggi .....	99
6 Fase 6: Role Description .....	100
6.1 Comunicazione .....	102
6.2 Dipendenze .....	102
7 Fase 7: Protocol Description .....	105
8 Fase 8: Multi-Agent Structure Definition .....	105
9 Fase 9: Multi-Agent Behaviour Description .....	107
9.1 Fase Single Agent Structure Definition .....	107
9.1.1 Agente: Author Gui .....	107
9.1.1.1 Attributi .....	108
9.1.1.2 Metodi .....	109
9.1.2 Agente: Common Gui .....	113
9.1.2.1 Attributi .....	113
9.1.2.2 Metodi .....	115
9.1.3 Agente: Access Manager .....	121
9.1.3.1 Attributi .....	122
9.1.3.2 Metodi .....	123
9.1.4 Agente: Editor .....	125
9.1.4.1 Attributi .....	126
9.1.4.2 Metodi .....	127

9.1.5 Agente: Planner .....	132
9.1.5.1 Attributi.....	133
9.1.5.2 Metodi.....	134
9.1.6 Agente: Ecs Gui.....	137
9.1.6.1 Attributi.....	137
9.1.6.2 Metodi.....	139
10 Fase 10 Deployment Configuration .....	145
11 Progetto dell'archivio .....	147
11.1 Progetto Concettuale .....	147
11.1.1 Descrizione dell'archivio.....	147
11.1.2 Identificazione degli elementi dell'ERD .....	148
11.2 ERD .....	149
11.3 Progetto Logico .....	150
11.4 Tabelle .....	151
<b>Manuale dell'utente</b> .....	154
1 Utilizzo del tool.....	155
2 Funzionalità dell'ECS .....	157
2.1 New member .....	157
2.2 Modify member .....	158
2.3 Define course .....	159
2.4 Develop Corse .....	162
2.5 Send Message .....	170
2.6 View Message .....	171
2.7 Settings .....	171
3 Funzionalità dell'Autore .....	172
3.1 Contracting Arguments .....	172
3.2 New Object .....	174
3.3 Developing Arguments.....	175
3.4 Send message e View Message .....	181
3.5 Settings .....	182
<b>Appendice</b> .....	183
1 Materiale fornito.....	184
2.1 Istruzioni per l'installazione .....	184
2.2 Installazione dei driver del DB .....	185
3 Statistiche .....	190

## ***Pianificazione del Progetto***

## 1 Diagramma di GANTT

Il seguente diagramma mostra la vista complessiva della pianificazione del progetto:

Oltre al diagramma di Gantt qui andava messa una pianificazione completa come studiata durante il corso e come viene fatto nelle tesine ad oggetti

M. Cossentino







## ***Requisiti di sistema***

## 1. Introduzione

Il recente e rapido sviluppo delle tecnologie, la loro diffusione capillare nelle aziende e nelle strutture didattiche di ogni livello (scuole, università) ed il basso costo ormai raggiunto dagli strumenti di comunicazione sono stati negli ultimi anni i fattori determinanti per l'ideazione, la crescita e la standardizzazione di sistemi informativi per la formazione a distanza (FAD).

In particolare le industrie si sono rese conto dell'importanza del fattore umano la cui continua formazione e crescita professionale assicura all'azienda l'apporto di nuove idee e l'aumento di produttività.

Per le aziende e le strutture didattiche diventa dunque fondamentale la possibilità di fornire ai dipendenti o agli studenti gli strumenti più adatti per conseguire una formazione mirata e adeguata alle capacità dell'individuo.

L'*e-learning* viene definito da molti (John Chamber – Presidente e CEO di Cisco Systems, Piper Jaffrey – US Bancorp ed altri), come la prossima *killer application* di Internet ed è ormai una realtà consolidata. Esso ha acquisito una fisionomia propria solo negli ultimi tempi, con il presentarsi sul mercato di svariati modelli implementativi, ed è chiara oggi la differenziazione tra i diversi tool di cui deve essere dotata l'azienda o la scuola per poter garantire ai propri utenti la fruizione on-line di moduli didattici.

Una piattaforma *e-learning* è infatti costituita principalmente di un tool per la produzione dei percorsi didattici ed uno per la fruizione di questi, rispettivamente detti *Authoring Tool* e *Learning Management System* (LMS), che possono trovarsi anche in combinazioni più o meno avanzate.

Il seguente lavoro di Tesi presenta un *Authoring Tool* collaborativo, basato sulle specifiche SCORM di ADL (*Advanced Distributed Learning*) che si sono ormai imposte come *standard de facto* sul panorama mondiale. Il Tool presenta le caratteristiche principali comuni a qualunque altro *Authoring Tool* commerciale ed è stato prodotto basandosi sulla piattaforma ad agenti Jade.

## 2 Overview del prodotto

Si propone la realizzazione di un tool per la produzione collaborativa di corsi dal contenuto multimediale.

Il Tool è strutturato in modo tale che il lavoro degli autori delle risorse didattiche sia coordinato da uno staff. È possibile pensare che, in un contesto accademico, gli autori siano docenti, ricercatori e quanti altri partecipano all'insegnamento e lo staff sia composto da un gruppo scelto tra questi ultimi.

Lo staff definisce i corsi, descrivendone la struttura e suddividendoli in argomenti, ognuno con una propria descrizione; successivamente questi ultimi vengono assegnati agli autori, i quali possono decidere di non accettare l'incarico. Se l'incarico viene accettato, questi dovranno sviluppare l'argomento a proprio piacimento.

Il Tool consente agli autori lo sviluppo degli argomenti mediante l'uso di un editor html di tipo WYSIWYG (*What you see is what you get*) che permette a chiunque, non per forza in possesso di conoscenze informatiche, di produrre pagine web con contenuti multimediali come immagini, filmati o grafici.

In particolare sarà possibile la condivisione delle risorse elettroniche usate; queste potranno poi essere ricercate all'interno del database delle risorse condivise grazie all'uso dei metadati, informazioni sulle risorse.

Ogni autore può scambiare messaggi con qualunque altro (anche con l'ECS) scegliendo tra un *instant message* o una *e-mail* (il client smtp è integrato nel tool) e questo facilita la collaborazione.

Le caratteristiche principali dell'autoring tool sono di seguito riassunte:

- Gli autori collaborano nella produzione di un corso sviluppandone gli argomenti.
- I corsi vengono definiti da un Editorial Central Staff e suddivisi in argomenti o sottocorsi.
- Gli argomenti vengono assegnati agli autori che possono rifiutare l'assegnazione
- Lo sviluppo degli argomenti da parte degli autori avviene tramite l'inserimento di pagine *html* a contenuto multimediale
- Man mano che sviluppano gli argomenti, gli autori producono e condividono con gli altri autori le informazioni atomiche.
- Al termine dello sviluppo di un argomento, ogni autore lo "esporta" verso l'*editorial central staff*; la procedura di esportazione mira a due scopi: il completamento di un corso e la condivisione degli argomenti.

L'*editorial central staff* completa lo sviluppo di un corso quando tutti gli argomenti che lo compongono sono stati esportati dagli autori.

### 3 Requisiti funzionali

Il tool deve essere in grado di:

- Definire la struttura dei corsi da sviluppare contenente sottocorsi e argomenti;
- Suddividere lo sviluppo degli argomenti che compongono i corsi tra più autori che collaborano alla loro produzione;
- Permettere un meccanismo di contrattazione per gli argomenti da assegnare agli autori;
- Mantenere lo stato di avanzamento di produzione dei corsi;
- Permette l'invio di messaggi tra gli utenti del sistema tramite un servizio di *instant messaging* o tramite email;
- Sviluppare gli argomenti mediante la produzione di pagine web con contenuti multimediali;
- Assemblare i corsi sviluppati per renderli utilizzabili da strumenti di pubblicazione e fruizione delle risorse sul web;
- Mantenere un archivio di risorse prodotte;
- Permettere il riutilizzo delle risorse prodotte presenti nell'archivio per la produzione dei contenuti;
- Mantenere un archivio degli utenti del sistema.

### 4 Requisiti non funzionali

#### 4.1 Interfaccia utente e fattore umano

Il tool è indirizzato a docenti, ricercatori, formatori aziendali, collaboratori aziendali o accademici che svolgono il compito di fornire la formazione agli utenti. In questo caso si terrà in considerazione un utilizzo accademico del tool che, però, deve poter essere utilizzato con le stesse funzionalità in ambito aziendale.

Gli utenti del tool potrebbero avere conoscenze basilari delle tecnologie informatiche; si richiede dunque un'interfaccia utente molto semplice e di

immediato utilizzo, che guidi l'autore nella produzione degli argomenti secondo lo standard scelto.

Affinché tali requisiti siano soddisfatti si richiede che il tool permetta di:

- ricercare efficientemente ed in maniera semplice una risorsa riutilizzabile e permetterne l'inserimento in un corso
- visualizzare un'anteprima di una risorsa quando questo può essere utile alla produzione di contenuti didattici
- produrre una pagina web secondo lo stile *wysiwyg* (*what you see is what you get*) senza che l'utente intervenga direttamente nella manipolazione del codice *html* prodotto
- definire in maniera intuitiva la struttura dei corsi o degli argomenti prodotti

## 4.2 Documentazione

Durante la produzione del tool saranno prodotti e rilasciati i seguenti documenti:

Documento	Utenti del documento
Documento di analisi dei requisiti (RAD)	Sviluppatori, cliente
Descrizione del progetto secondo la metodologia PASSI	Sviluppatori
Manuale dell'utente	Cliente

## 4.3 Considerazioni hardware

Il tool è progettato per essere eseguito su Personal Computer basati su processori Intel o AMD.

Ogni Personal Computer su cui si desidera installare il tool deve essere fornito di accesso alla rete Internet e possedere i seguenti pacchetti:

- Jade 2.6 con codec RDF
- JRE 1.4
- Librerie Java:
  - o Javamail 1.3
  - o Jaf 1.0.2

Deve essere accessibile via Web un Server FTP di appoggio con tecnologia Bullet Proof utilizzato per la condivisione delle risorse in cui siano registrati gli account degli utenti del sistema.

#### **4.4 Vincoli: pseudo-requisiti**

Vincolo fondamentale per la progettazione e lo sviluppo del tool è la modellazione di un sistema multiagente (MAS, multi agent system) mediante la metodologia PASSI, usufruendo dell'apposito tool PTK (Passi ToolKit) disponibile come plugin per il Software di progettazione "Rational Rose".

Il toolkit per la progettazione di sistemi multi-agente PASSI prevede la possibilità di scegliere la piattaforma su cui implementare gli agenti tra FIPA-OS e JADE.

Per questo lavoro si è scelto di utilizzare la piattaforma JADE perché questa rispetto FIPA-OS offre prestazioni pressoché costanti all'aumentare del numero degli agenti presenti nella piattaforma; in un sistema in cui il numero di agenti presenti nella piattaforma è proporzionale al numero di autori che collaborano alla produzione di contenuti, visto che per ogni autore si aggiungono due agenti alla piattaforma, tale caratteristica risulta essere molto importante qualora venga coinvolto un elevato numero di autori.

Tuttavia la piattaforma FIPA-OS offre dei vantaggi per applicazioni di tipo real-time; tale vincolo è stato trascurato nella scelta di JADE dal momento che il tool che è stato progettato in questo lavoro non richiede particolari vincoli di tempo e reattività degli agenti.

### **5 Modelli del sistema**

#### **5.1 Scenari**

##### **5.1.1 Identificazione degli attori**

Author: uno o più autori che producono informazioni atomiche, argomenti e collaborano nella produzione di corsi;

Editorial Central Staff: entità che raccoglie una o più persone il cui compito è:

- Ideare i corsi e suddividerli in argomenti

- Assegnare ai vari autori gli argomenti necessari da sviluppare
- Assemblare gli argomenti consegnati dai singoli autori per produrre un corso
- Mantenere un DB contenente la lista degli autori e i relativi lavori in corso (anche stato di avanzamento ai fini della pubblicazione finale)
- Pubblicare gli ipertesti prodotti;

End User: l'utente finale che usufruisce attraverso il browser dei moduli didattici pubblicati.

### 5.1.2 Identificazione degli scenari

- Comunicazione tra membri (Authors ed Editorial Central Staff);
- Ideazione di un corso (da parte dell' Editorial Central Staff);
- Autenticazione (sia per Author che Editorial Central Staff);
- Produzione di un argomento (da parte dell' Author);
- Assemblamento degli argomenti ricevuti (da parte dell' Editorial Central Staff);
- Pubblicazione dei corsi (da parte dell' Editorial Central Staff);
- Negoziare gli argomenti (tra Authors ed Editorial Central Staff);
- Esportazione degli argomenti completati (dagli Authors all' Editorial Central Staff);
- Gestione degli utenti (da parte dell' Editorial Central Staff);
- Assegnazione degli argomenti (da parte dell' Editorial Central Staff);

### 5.1.3 Descrizione degli scenari

Comunicazione tra membri (Authors ed Editorial Central Staff)

---

<i>Scenario name</i>	sendMessage
<i>Participating actor</i>	bob, frank: Author
<i>instances</i>	
<i>Flow of Events</i>	1. Bob, un Author intento a sviluppare

l'argomento "Telecomunicazioni" assegnatogli, decide di consultarsi con Frank, esperto in materia ed anche lui Author.

2. Bob seleziona la funzione "Invia messaggio" dall'interfaccia utente che mostra l'elenco di utenti ed uno spazio su cui scrivere il messaggio.
3. Bob seleziona Frank dall'elenco di utenti e scrive sullo spazio destinato al messaggio: "Ciao Frank, mi daresti una mano?".
4. Bob inoltra il messaggio.
5. Sull'interfaccia utente di Frank viene notificata la ricezione di un nuovo messaggio da parte di Bob.
6. Frank sceglie di visualizzare il messaggio e questo viene mostrato.
7. Frank, una volta letto il messaggio, decide di cancellare il messaggio.

---

<i>Scenario name</i>	sendingAnEmail
<i>Participating actor</i>	bob, frank: Author
<i>instances</i>	
<i>Flow of Events</i>	<ol style="list-style-type: none"> <li>1. Bob, un Author intento a sviluppare l'argomento "Telecomunicazioni" assegnatogli, decide di consultarsi con Frank, esperto in materia ed anche lui Author, tramite una email.</li> <li>2. Bob seleziona la funzione "Invia messaggio" dall'interfaccia utente che mostra l'elenco di utenti ed uno spazio su cui scrivere il messaggio.</li> <li>3. Bob seleziona Frank dall'elenco di utenti e scrive sullo spazio destinato al messaggio:</li> </ol>



“Ciao Frank, ti scrivo questa email per chiedere una tua collaborazione. Fammi sapere. Saluti, Bob.”.

4. Bob inoltra l’email.
5. Frank scarica la posta con il suo programma di gestione email preferito. Tra i messaggi ricevuti figura una email il cui oggetto è “Email inviata da Bob”.
6. Frank sceglie di visualizzare l’email viene mostrato quanto scritto da Bob.

---

<i>Scenario name</i>	checkingMessagesQueue
<i>Participating actor instances</i>	bob, frank: Author
<i>Flow of Events</i>	<ol style="list-style-type: none"> <li>1. Bob, un Author intento a sviluppare l’argomento “Telecomunicazioni” assegnatogli, decide di consultare l’elenco dei messaggi tra i quali due contengono consigli sull’argomento precedentemente inviati da Frank.</li> <li>2. Bob seleziona la funzione “Messaggi ricevuti” dall’interfaccia utente che mostra uno dei messaggi ricevuti.</li> <li>3. Bob scorre la lista di messaggi finchè trova il messaggio che cercava. Mentre Bob lo sta leggendo si accorge di aver ricevuto un nuovo messaggio tramite una notifica e il conseguente aggiornamento dell’elenco di messaggi.</li> </ol>

---

Ideazione di un corso (da parte dell’Editorial Central Staff)

---

<i>Scenario name</i>	simpleCourseDefinition
<i>Participating actor</i>	bob: Editorial Central Staff

*instances*

*Flow of Events*

1. I membri dello staff si sono riuniti per ed hanno deciso la creazione di un nuovo corso: “Linguaggio Java”. Si è deciso di suddividere il corso in tre argomenti: “Introduzione a Java”, “Programmazione Base”, “Programmazione Avanzata”.
2. Bob, uno dei membri dello staff, seleziona dall’interfaccia utente dell’ Editorial Central Staff, la funzionalità “Definisci Nuovo Corso”.
3. Bob inserisce in corrispondenza al titolo del corso il testo “Linguaggio Java” e nel campo dedicato alla descrizione scrive “Il corso si prefigge lo scopo di fornire una visione di insieme del linguaggio Java”.
4. Bob inserisce in corrispondenza al titolo dell’argomento il testo “Introduzione a Java” aggiungendovi una breve descrizione come fatto per il corso. Successivamente Bob conferma l’aggiunta al corso dell’argomento appena descritto.
5. Bob ripete l’operazione precedente per gli argomenti “Programmazione Base” e “Programmazione Avanzata”.
6. Bob inoltra le definizioni del corso.

---

<i>Scenario name</i>	ComplexCourseDefinition
----------------------	-------------------------

<i>Participating actor</i>	bob: Editorial Central Staff
----------------------------	------------------------------

*instances*

*Flow of Events*

1. I membri dello staff si sono riuniti per ed hanno deciso la creazione di un nuovo corso: “Linguaggi ad Oggetti”. Si è deciso di suddividere il corso in due sottocorsi ed

un argomento, i primi due aventi i titoli “Linguaggio Java” e “Linguaggio C++”, il terzo argomento avente titolo “Introduzione ai linguaggi ad Oggetti”. Il sottocorso “Linguaggio Java” è composto da tre argomenti: “Introduzione a Java”, “Programmazione Base in Java”, “Programmazione Avanzata in Java”. Il sottocorso “Linguaggio C++” comprende gli argomenti: “Introduzione a C++”, “Programmazione Base in C++”, “Programmazione Avanzata in C++”

2. Bob, uno dei membri dello staff, seleziona dall'interfaccia utente dell' Editorial Central Staff, la funzionalità “Definisci Nuovo Corso”.
3. Bob inserisce in corrispondenza al titolo del corso il testo “Linguaggi ad Oggetti” e nel campo dedicato alla descrizione scrive “Il corso si prefigge lo scopo di fornire una visione di insieme dei principali linguaggi ad Oggetti”.
4. Bob inserisce in corrispondenza al titolo del sottocorso il testo “Linguaggio Java” aggiungendovi una breve descrizione come fatto per il corso. Successivamente Bob conferma l'aggiunta al corso del sottocorso appena descritto.
5. Bob aggiunge al sottocorso “Linguaggio Java” i tre argomenti relativi.
6. Bob ripete le due operazioni precedenti per il sottocorso “Linguaggio C++”
7. Bob aggiunge infine l'argomento “Introduzione ai linguaggi ad Oggetti”

8. Bob inoltra le definizioni del corso.

---

<i>Scenario name</i>	courseDefinitionWithSharedArgument
<i>Participating actor</i>	bob: Editorial Central Staff
<i>instances</i>	
<i>Flow of Events</i>	<ol style="list-style-type: none"><li>1. I membri dello staff si sono riuniti per ed hanno deciso la creazione di un nuovo corso: "Linguaggio Java". Si è deciso di suddividere il corso in tre argomenti: "Introduzione a Java", "Programmazione Base", "Programmazione Avanzata".</li><li>2. Bob, uno dei membri dello staff, seleziona dall'interfaccia utente dell' Editorial Central Staff, la funzionalità "Definisci Nuovo Corso".</li><li>3. Bob inserisce in corrispondenza al titolo del corso il testo "Linguaggio Java" e nel campo dedicato alla descrizione scrive "Il corso si prefigge lo scopo di fornire una visione di insieme del linguaggio Java".</li><li>4. Prima di definire un nuovo argomento, Bob vuole verificare se l'argomento "Introduzione a Java" sia già stato precedentemente sviluppato.</li><li>5. Bob attiva la funzionalità di ricerca di argomenti completati. Viene mostrato un elenco di parecchi argomenti. Bob inserisce la parola chiave "Java" ed inoltra la richiesta di ricerca; l'elenco si riduce così a pochi argomenti, tra i quali "Primi passi in Java".</li><li>6. Bob, pensando che si tratti di un argomento utile al corso, decide di visualizzarne una anteprima.</li></ol>

7. Bob gradisce l'argomento "Primi passi in Java" e lo aggiunge al corso.
8. Bob inserisce i rimanenti argomenti e inoltra la conferma di definizione del corso.

---

Autenticazione (sia per Authors che Editorial Central Staff)

---

<i>Scenario name</i>	correctAuthentication
<i>Participating actor</i>	bob: Author
<i>instances</i>	
<i>Flow of Events</i>	<ol style="list-style-type: none"> <li>1. Bob, seduto di fronte la sua postazione, decide di avviare il programma per sviluppare degli argomenti che gli sono stati assegnati.</li> <li>2. Bob seleziona il comando per lanciare il programma e viene visualizzata un interfaccia di autenticazione.</li> <li>3. Bob inserisce il proprio nome utente "Bobby75" e la propria password "ILoveNY" per accedere al programma.</li> <li>4. Bob conferma l'invio dei dati di autenticazione inseriti. Il programma mostra l'interfaccia principale con le funzionalità a disposizione dell'Author.</li> </ol>

---



---

<i>Scenario name</i>	failureWhileAuthenticating
<i>Participating actor</i>	bob: Editorial Central Staff
<i>instances</i>	
<i>Flow of Events</i>	<ol style="list-style-type: none"> <li>1. Bob, seduto di fronte la sua postazione, decide di avviare il programma per definire un nuovo corso.</li> <li>2. Bob seleziona il comando per lanciare il programma e viene visualizzata un interfaccia di autenticazione.</li> </ol>

---

3. Bob inserisce il proprio nome utente “Ecs” e la propria password “chiefv” per accedere al programma.
  4. Bob conferma l’invio dei dati di autenticazione inseriti. Il programma mostra un messaggio di errore per inserimento errato dei dati di autenticazione .
  5. Bob inserisce nuovamente il suo nome utente e la password “chief”. Questa volta il programma riconosce i dati di autenticazione inseriti da Bob e mostra l’interfaccia principale con le funzionalità a disposizione dell’Editorial Central Staff.
- 

Produzione di un argomento (da parte dell’ Author)

<i>Scenario name</i>	addingAPage
<i>Participating actor instances</i>	bob: Author
<i>Flow of Events</i>	<ol style="list-style-type: none"> <li>1. Bob, seduto di fronte la sua postazione, seleziona la funzione per produrre gli argomenti che gli sono stati assegnati.</li> <li>2. Il programma mostra la lista di argomenti a cui Bob deve lavorare e per ogni argomento visualizza la lista di pagine fino ad ora aggiunte e il relativo corso di appartenenza dell’argomento.</li> <li>3. Bob sceglie dalla lista l’argomento “Introduzione a Java”.</li> <li>4. Al momento la lista delle pagine già prodotte per questo argomento è vuota e Bob decide di aggiungere una nuova pagina .</li> </ol>

5. Bob seleziona l'opzione per aggiungere una pagina e viene mostrata una scelta di scheletri di pagine che aiuteranno Bob per sviluppare la sua pagina.
6. Bob seleziona lo scheletro la cui anteprima riporta un immagine affiancata da una singola colonna di testo.
7. Bob al posto dell'immagine campione tramite l'opzione per aggiungere un immagine inserisce il logo di Java con la tazza di caffè fumante e al posto del testo campione comincia a scrivere: "questo argomento tratterà il linguaggio java partendo da una breve introduzione sui linguaggi ad oggetti..."
8. Bob decide di inserire tra il testo tramite l'opzione per aggiungere un link, un riferimento alla pagina "http://java.sun.com"
9. Bob soddisfatto della sua pagina seleziona l'opzione per salvare la pagina e le da il nome di "paginaIniziale.html".
10. Adesso Bob tra la lista delle pagine dell'argomento visualizza anche la pagina appena creata.

---

<i>Scenario name</i>	modifyingAPage
<i>Participating actor</i>	bob: Author
<i>instances</i>	
<i>Flow of Events</i>	<ol style="list-style-type: none"> <li>1. Bob, esperto di Java, ha un ripensamento sulla pagina che ha appena prodotto nello scenario precedente e dopo aver selezionato dalla lista delle pagine relative all'argomento "Introduzione a Java" la</li> </ol>

- pagina “paginaIniziale.html” seleziona l’opzione per modificare la pagina.
2. L’interfaccia mostra la pagina e Bob decide di aggiungere per chiarezza un esempio di codice Java; seleziona l’opzione “Aggiungi Nuova Informazione Atomica” e sceglie il file “SampleClass.java”.
  3. Bob conferma l’aggiunta di tale informazione atomica e viene visualizzata un interfaccia che permette l’inserimento dei metadati.
  4. Bob sceglie l’opzione per mostrare soltanto i campi obbligatori da compilare e comincia a inserire i metadati; tra gli altri campi, tra le parole chiave inserisce “codice” e “Java”,
  5. Bob seleziona l’opzione per confermare l’inserimento dei metadati; adesso la pagina contiene il link al file “SampleClass.java”.
  6. Bob soddisfatto della sua pagina seleziona l’opzione per salvare la pagina.

---

<i>Scenario name</i>	deletingAPage
<i>Participating actor</i>	bob: Author
<i>instances</i>	
<i>Flow of Events</i>	<ol style="list-style-type: none"> <li>1. Bob, avendo lavorato durante tutta la notte all’argomento “Introduzione a Java”, si accorge di avere prodotto la pagina “paginaIniziale.html” utilizzando uno scheletro poco adatto alle sue necessità; pertanto decide di cancellare tale pagina selezionandola dalla lista delle pagine prodotte e attivando la funzione “Cancella Pagina”.</li> <li>2. Viene mostrato un messaggio di conferma</li> </ol>



prima che la pagina venga irrimediabilmente cancellata; Bob conferma la cancellazione della pagina selezionata che adesso non è più tra la lista delle pagine prodotte.

---

<i>Scenario name</i>	addingASharedResourceToAPage
<i>Participating actor instances</i>	Frank, bob: Author
<i>Flow of Events</i>	<ol style="list-style-type: none"><li>1. Frank, sta lavorando alla produzione dell'argomento "Java vs. C++" e seleziona l'opzione per aggiungere una pagina.</li><li>2. Frank sceglie come layout una pagina vuota e comincia a scrivere del testo: "Adesso a titolo di esempio potrete osservare alcuni spezzoni di codice in Java e in C++...".</li><li>3. Frank seleziona il testo "esempio codice C++" e attiva l'opzione per aggiungere una nuova informazione atomica. Frank sceglie il file "SampleCode.cpp" e successivamente riempie i metadati per tale informazione atomica e ne conferma l'inserimento.</li><li>4. Frank adesso seleziona il testo "esempio codice Java" e attiva l'opzione per la ricerca di informazioni atomiche. Frank inserisce tra le chiavi per la ricerca la parola "codice" e conferma la ricerca. Viene restituita una lista di informazioni atomiche tra cui è presente l'informazione atomica "SampleClass.java" precedentemente prodotta in uno scenario da Bob.</li><li>5. Frank visualizza un anteprima e rimanendone soddisfatto conferma l'inserimento di tale informazione atomica.</li></ol>

6. Frank salva la pagina con il nome “EsempiDiCodice.html”.
7. Adesso la lista delle pagine relative all’argomento “Java vs. C++” contiene tra le altre anche la pagina “EsempiDiCodice.html”.

---

Assemblamento degli argomenti ricevuti (da parte dell’Editorial Central Staff)

---

<i>Scenario name</i>	assembleCompletedCourse
<i>Participating actor</i>	frank: Editorial Central Staff
<i>instances</i>	bob, james: Author
<i>Flow of Events</i>	<ol style="list-style-type: none"> <li>1. Frank, membro dell’Editorial Central Staff ha ricevuto dagli Authors Bob e James gli argomenti che questi hanno sviluppato: “Introduzione a Java”, “Programmazione Base” e “Programmazione Avanzata”. Adesso il corso “Linguaggio Java” già visto negli scenari precedenti è pronto per essere assemblato.</li> <li>2. Frank seleziona la funzione per assemblare il corso; viene visualizzata un interfaccia in cui sono presenti gli argomenti del corso “Linguaggio Java”; Frank dispone gli argomenti scegliendo l’ordine sequenziale con cui gli argomenti verranno esposti all’utente finale che fruirà il corso .</li> <li>3. Frank ordina gli argomenti secondo la sequenza: “Introduzione a Java”, “Programmazione Base”, “Programmazione Avanzata”.</li> <li>4. Frank adesso seleziona l’opzione per</li> </ol>

inserire i metadati relativi al corso “Linguaggio Java”. Viene visualizzata l’interfaccia per la compilazione dei metadati.

5. Frank dopo aver riempito tutti i campi dei metadati ne conferma l’inserimento.
6. Frank seleziona adesso la conferma di assemblamento del corso.
7. Il programma completa l’operazione di assemblamento al termine della quale il corso è pronto per essere pubblicato e fruito.

Pubblicazione dei corsi (da parte dell’Editorial Central Staff, *LMS Sample*)

<i>Scenario name</i>	publishingACourse
<i>Participating actor</i>	frank: Editorial Central Staff
<i>instances</i>	john: End User
<i>Flow of Events</i>	<ol style="list-style-type: none"> <li>1. Frank, membro dell’Editorial Central Staff, dopo aver assemblato il corso “Linguaggio Java” vuole renderlo pubblico agli utenti. Per fare ciò Frank ha a disposizione il programma di fruizione dei corsi <i>ADL LMS Sample</i>.</li> <li>2. Frank avvia il programma, si autentica come amministratore e seleziona la funzione di pubblicazione di un nuovo corso.</li> <li>3. Frank seleziona il corso “Linguaggio Java” che ha assemblato in uno scenario precedente e conferma la scelta.</li> <li>4. John, studente registrato al programma per la fruizione di corsi, desidera fruire il corso</li> </ol>

“Linguaggio Java”; si collega tramite il suo browser al programma di fruizione e si autentica con il proprio nome utente e password.

5. John seleziona l’opzione per vedere i corsi disponibili tra i quali è elencato “Linguaggio Java”. John pertanto lo seleziona e conferma la registrazione al corso.

---

Negoziazione degli argomenti (tra Authors ed Editorial Central Staff)

<i>Scenario name</i>	acceptingAnArgument
<i>Participating actor</i>	frank: Editorial Central Staff
<i>instances</i>	john: Author
<i>Flow of Events</i>	<ol style="list-style-type: none"> <li>1. John, durante una sessione di lavoro, riceve una notifica di assegnazione di un nuovo argomento.</li> <li>2. John sceglie la funzionalità che gli permette di visualizzare gli argomenti assegnatigli e le loro descrizioni.. Tra questi compare il nuovo argomento “Programmazione Base”, insieme al commento inserito da Frank al momento dell’assegnazione “Ciao, occupatene tu, sei la persona più adatta; non serve che sia molto approfondito”.</li> <li>3. John crede, date le sue capacità ed esperienza a riguardo, di essere in grado di sviluppare l’argomento e accetta l’argomento aggiungendo il commento “Ok Frank, me ne occupo io”.</li> <li>4. Frank, durante la sua sessione di lavoro, riceve una notifica di accettazione da parte</li> </ol>

di John dell'argomento "Programmazione Base".

5. Frank decide di visualizzare l'interfaccia che mostra lo stato di avanzamento dei corsi; viene mostrato il corso "Linguaggio Java" ed in corrispondenza all'argomento "Programmazione Base" viene specificato che questo è stato accettato da John con il commento "Ok Frank, me ne occupo io".

---

<i>Scenario name</i>	contractingAnArgument
<i>Participating actor instances</i>	frank: Editorial Central Staff john: Author
<i>Flow of Events</i>	<ol style="list-style-type: none"> <li>1. John, durante una sessione di lavoro, riceve una notifica di assegnazione di un nuovo argomento.</li> <li>2. John sceglie la funzionalità che gli permette di visualizzare gli argomenti assegnatigli e le loro descrizioni.. Tra questi compare il nuovo argomento "Programmazione Base", insieme al commento inserito da Frank al momento dell'assegnazione "Ciao, occupatene tu, sei la persona più adatta; non serve che sia molto approfondito".</li> <li>3. John crede, date le sue capacità ed esperienza a riguardo, di non essere in grado di sviluppare l'argomento e lo rifiuta aggiungendo il commento "Frank, non penso di esserne in grado".</li> <li>4. Frank, durante la sua sessione di lavoro, riceve una notifica di rifiuto da parte di John dell'argomento "Programmazione Base".</li> <li>5. Frank decide di visualizzare l'interfaccia</li> </ol>

che mostra lo stato di avanzamento dei corsi; viene mostrato il corso “Linguaggio Java” ed in corrispondenza all’argomento “Programmazione Base” viene specificato che questo è stato rifiutato da John con il commento “Frank, non penso di esserne in grado”.

6. Frank è convinto che John sia la persona adatta per sviluppare l’argomento, e glielo riassegna inserendo il commento “Ma che dici, sono sicuro che nessuno potrebbe fare meglio di te”.
7. John, riceve nuovamente la notifica di ricezione dell’argomento. Visualizza nuovamente il commento e, presosi d’animo, si lascia convincere dalle parole di Frank.
8. John accetta l’argomento inserendo il commento “Ok Frank; ve diamo cosa posso fare”
9. Frank riceve la notifica di accettazione dell’argomento da parte di John.

<i>Scenario name</i>	refusingAnArgument
<i>Participating actor instances</i>	frank: Editorial Central Staff john, jack: Author
<i>Flow of Events</i>	<ol style="list-style-type: none"> <li>1. John, durante una sessione di lavoro, riceve una notifica di assegnazione di un nuovo argomento.</li> <li>2. John sceglie la funzionalità che gli permette di visualizzare gli argomenti assegnatigli e le loro descrizioni.. Tra questi compare il nuovo argomento “Programmazione Base”,</li> </ol>

insieme al commento inserito da Frank al momento dell'assegnazione "Ciao, occupatene tu, sei la persona più adatta; non serve che sia molto approfondito".

3. John crede, date le sue capacità ed esperienza a riguardo, di non essere in grado di sviluppare l'argomento e lo rifiuta aggiungendo il commento "Frank, non penso di esserne in grado".
4. Frank, durante la sua sessione di lavoro, riceve una notifica di rifiuto da parte di John dell'argomento "Programmazione Base".
5. Frank decide di visualizzare l'interfaccia che mostra lo stato di avanzamento dei corsi; viene mostrato il corso "Linguaggio Java" ed in corrispondenza all'argomento "Programmazione Base" viene specificato che questo è stato rifiutato da John con il commento "Frank, non penso di esserne in grado".
6. Frank è convinto che John sia la persona adatta per sviluppare l'argomento, e glielo riassegna inserendo il commento "Ma che dici, sono sicuro che nessuno potrebbe fare meglio di te".
7. John, riceve nuovamente la notifica di ricezione dell'argomento. Visualizza nuovamente il commento e rifiuta definitivamente l'argomento scrivendo a Frank "No Frank; proprio non me la sento".
8. Frank riceve la notifica di rifiuto dell'argomento da parte di John e decide di riassegnarlo a Jack.

---

Esportazione degli argomenti completati (dagli Authors all'Editorial Central Staff)

---

<i>Scenario name</i>	exportingAnArgument
<i>Participating actor</i>	Frank: Editorial Central Staff
<i>instances</i>	john: Author
<i>Flow of Events</i>	<ol style="list-style-type: none"><li>1. John, di fronte alla sua postazione di lavoro, ha appena completato l'ultima pagina dell'argomento "Programmazione Java" ed è pronto per esportarlo.</li><li>2. John sceglie la funzionalità che gli permette di esportare l'argomento e viene visualizzata un'interfaccia che gli permette di ordinare le pagine che ha prodotto all'interno dell'argomento; John, dunque, seleziona una ad una le pagine nell'ordine in cui queste verranno poi fruite e, una volta che le ha ordinate tutte, conferma quanto fatto.</li><li>3. John inserisce dunque i metadati relativi all'argomento prodotto e conferma infine tale inserimento.</li><li>4. Frank riceve la notifica che John ha esportato l'argomento "Programmazione Java" e sull'interfaccia di avanzamento dei corsi viene mostrato che l'argomento è stato completato.</li><li>5. Frank visualizza un'anteprima dell'argomento, scorrendo tutte le pagine prodotte da John.</li></ol>

---

Gestione degli utenti (da parte dell'Editorial Central Staff)



<i>Scenario name</i>	addingAMember
<i>Participating actor</i>	frank: Editorial Central Staff
<i>instances</i>	john: Author
<i>Flow of Events</i>	<ol style="list-style-type: none"> <li>1. Frank desidera registrare al sistema un nuovo autore, John.</li> <li>2. Frank seleziona l'opzione che gli permette l'inserimento di nuovi autori e viene visualizzata un'interfaccia che richiede il riempimento di una serie di campi, tra cui "Username", "Password", "NickName", "Email".</li> <li>3. Frank riempie tutti i campi visualizzati e conferma l'inserimento.</li> </ol>

<i>Scenario name</i>	modifingAMember
<i>Participating actor</i>	frank: Editorial Central Staff
<i>instances</i>	john: Author
<i>Flow of Events</i>	<ol style="list-style-type: none"> <li>1. John, un Author , ha di recente cambiato il proprio indirizzo email e desidera comunicarlo a Frank. Pertanto John invia a Frank il messaggio "Frank, volevo chiederti di aggiornare i miei dati; la mia nuova email è <a href="mailto:john@myemail.com">john@myemail.com</a>".</li> <li>2. Frank riceve il messaggio e lo visualizza. Frank attiva dunque l'opzione di modifica dei dati degli utenti.</li> <li>3. Viene mostrato un elenco di tutti gli utenti registrati sul sistema e Frank seleziona il nome di John. Frank sceglie dunque di modificarne i dati.</li> </ol>

<i>Scenario name</i>	deletingAMember
<i>Participating actor</i>	frank: Editorial Central Staff

<i>instances</i>	john: Author
<i>Flow of Events</i>	<ol style="list-style-type: none"> <li>1. John, docente di Programmazione, è andato in pensione ed i suoi dati devono essere eliminati dal sistema.</li> <li>2. Frank, attiva l'opzione di modifica dei dati degli utenti e seleziona dall'elenco il nome di John.</li> <li>3. Frank seleziona l'opzione di cancellazione e viene visualizzata una richiesta di conferma.</li> <li>4. Frank conferma la richiesta ed il nome di John scompare dall'elenco.</li> </ol>

---

Assegnazione degli argomenti (da parte dell'Editorial Central Staff)

---

<i>Scenario name</i>	assigningArguments
<i>Participating actor</i>	frank: Editorial Central Staff
<i>instances</i>	john, jack: Author
<i>Flow of Events</i>	<ol style="list-style-type: none"> <li>1. Frank ha appena definito il corso "Linguaggio Java", composto dai due argomenti "Programmazione Base" e "Programmazione Avanzata"; Frank desidera assegnare i due argomenti.</li> <li>2. Frank seleziona l'opzione di assegnazione argomenti e viene mostrata un'interfaccia contenente tra l'altro l'elenco di autori, tra i quali compaiono John e Jack.</li> <li>3. Frank controlla i dati personali di ognuno dei due utilizzando l'apposita funzione, e vede che John è un esperto programmatore. Frank dunque decide di assegnare l'argomento "Programmazione Base" a Jack e l'argomento "Programmazione Avanzata" a John.</li> </ol>

4. Frank aggiunge alle due assegnazioni i relativi commenti e conferma l'assegnazione degli argomenti ai rispettivi autori.

## 5.2 Funzionalità del sistema

### 5.2.1 Domain Description

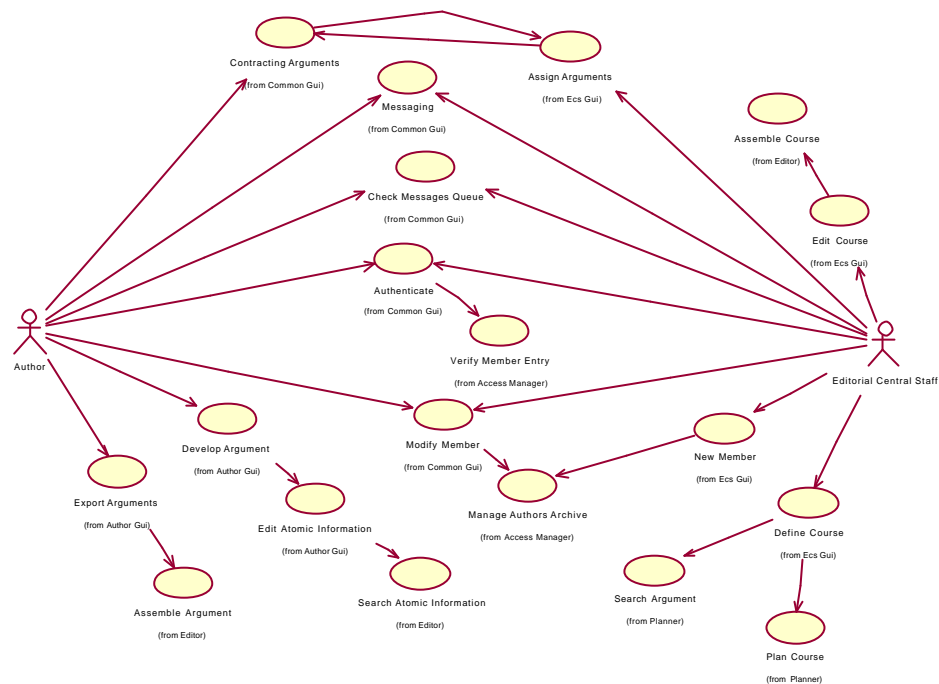


Figura 1: Diagramma dei casi d'uso della fase Domain Description.

### 5.2.2 Descrizione dei casi d'uso

<i>Use case name</i>	Authenticate
<i>Participating actor</i>	Iniziato da Author o Editorial Central Staff
<i>Entry condition</i>	1. L'Author o l'Editorial Central Staff avvia il programma.
<i>Flow of events</i>	2. Il programma risponde presentando il form di autenticazione. 3. L'Author o l'Editorial Central

	Staff riempie i campi relativi allo username e alla password e ne conferma l'invio.
<i>Exit condition</i>	4. Se i dati inseriti sono corretti il programma visualizza le funzionalità a disposizione dell'Author o dell'Editorial Central Staff.
<i>Special requirements</i>	L'Author o l'Editorial Central Staff riceve un messaggio d'errore se i dati inseriti non sono corretti.

---

<i>Use case name</i>	Messaging
<i>Participating actor</i>	Iniziato da Author o Editorial Central Staff
<i>Entry condition</i>	1. L'Author o l'Editorial Central Staff attiva la funzionalità per l'invio di messaggi.
<i>Flow of events</i>	2. Il programma risponde presentando la lista degli utenti e un campo in cui inserire il messaggio. 3. L'Author o l'Editorial Central Staff riempie il campo del messaggio, sceglie l'utente a cui inviare il messaggio e conferma l'invio del messaggio o tramite instant messaging o tramite e-mail.
<i>Exit condition</i>	4. Il messaggio o l'e-mail vengono inviati correttamente all'utente selezionato.
<i>Special requirements</i>	Nessuno

---

<i>Use case name</i>	CheckMessagesQueue
<i>Participating actor</i>	Iniziato da Author o Editorial Central Staff
<i>Entry condition</i>	1. L'Author o l'Editorial Central

	Staff attiva la funzionalità per visualizzare la coda dei messaggi ricevuti.
<i>Flow of events</i>	<ol style="list-style-type: none"> <li>Il programma risponde presentando la lista dei messaggi ricevuti e gli utenti da cui sono stati inviati i messaggi.</li> <li>L'Author o l'Editorial Central Staff scorre in entrambe le direzioni la lista dei messaggi che vengono visualizzati in un apposito campo.</li> <li>L'Author o l'Editorial Central Staff elimina dalla lista dei messaggi quelli che ha letto o che non gli interessano.</li> </ol>
<i>Exit condition</i>	<ol style="list-style-type: none"> <li>L'Author o l'Editorial Central Staff chiude l'interfaccia per la visualizzazione della coda dei messaggi.</li> </ol>
<i>Special requirements</i>	Ogni volta che arriva un nuovo messaggio l'utente deve ricevere una notifica.
<hr/>	
<i>Use case name</i>	NewMember
<i>Participating actor</i>	Iniziato da Editorial Central Staff
<i>Entry condition</i>	<ol style="list-style-type: none"> <li>L'Editorial Central Staff attiva la funzionalità per inserire un nuovo utente.</li> </ol>
<i>Flow of events</i>	<ol style="list-style-type: none"> <li>Il programma risponde presentando un form in cui inserire i dati dell'utente.</li> <li>L'Editorial Central Staff riempie i campi con i dati del nuovo utente.</li> </ol>
<i>Exit condition</i>	<ol style="list-style-type: none"> <li>L'Editorial Central Staff conferma l'immissione dei dati per il nuovo utente che viene registrato.</li> </ol>
<i>Special requirements</i>	Il campo username deve identificare univocamente un utente.
<hr/>	
<hr/>	

<i>Use case name</i>	ModifyMember
<i>Participating actor</i>	Iniziato da Author o Editorial Central Staff
<i>Entry condition</i>	1. L'Editorial Central Staff attiva la funzionalità per modificare i dati di un utente già esistente.
<i>Flow of events</i>	2. Il programma risponde presentando la lista degli utenti e due possibili opzioni rispettivamente per cancellare definitivamente un utente già esistente o per modificarne i suoi dati. 3. L'Editorial Central Staff seleziona dalla lista uno degli utenti e sceglie di cancellarlo o modificarlo. 4. Se l'Editorial Central Staff sceglie di modificare un utente viene visualizzato un form con i dati esistenti dell'utente. 5. l'Editorial Central Staff interviene modificando i campi di interesse all'interno del form.
<i>Exit condition</i>	6. L'Editorial Central Staff conferma la modifica dei dati per l'utente già esistente o ne conferma la sua cancellazione.
<i>Special requirements</i>	Il campo username non può essere modificato.

---

<i>Use case name</i>	DefineCourse
<i>Participating actor</i>	Iniziato da Editorial Central Staff
<i>Entry condition</i>	1. L'Editorial Central Staff attiva la funzionalità per definire un nuovo corso.
<i>Flow of events</i>	2. Il sistema risponde presentando un interfaccia dove è possibile inserire titolo e

	descrizione sia per il corso che per gli argomenti e i sottocorsi che si vogliono aggiungere al corso.
	3. L'Editorial Central Staff inserisce titolo e descrizione per ogni sottocorso che vuole aggiungere al corso e ne conferma l'inserimento.
	4. L'Editorial Central Staff inserisce titolo e descrizione per ogni argomento che vuole aggiungere all'interno del corso o di un sottocorso e ne conferma l'inserimento.
	5. L'Editorial Central Staff tramite un meccanismo di ricerca messo a disposizione dall'interfaccia può aggiungere al corso o a un sottocorso uno o più argomenti che sono stati già sviluppati precedentemente in altri corsi.
<i>Exit condition</i>	6. L'Editorial Central Staff conferma l'invio per il corso appena definito.
<i>Special requirements</i>	Un corso o un sottocorso deve contenere al suo interno almeno un argomento sia esso già sviluppato o appena definito.
<hr/>	
<i>Use case name</i>	AssignArguments
<i>Participating actor</i>	Iniziato da Editorial Central Staff
<i>Entry condition</i>	1. L'Editorial Central Staff attiva la funzionalità per assegnare agli autori gli argomenti di corsi precedentemente definiti.
<i>Flow of events</i>	2. Il sistema risponde presentando un interfaccia dove è possibile visualizzare la lista dei corsi definiti, una struttura che rappresenta un corso selezionato dalla lista

	dei corsi e una lista degli autori.
	3. L'Editorial Central Staff seleziona un argomento appartenente al corso mostrato e lo assegna ad un autore della lista degli autori; l'Editorial Central Staff ripete l'operazione per tutti gli argomenti che vuole assegnare agli autori della lista.
	4. L'Editorial Central Staff può inserire un commento in ogni argomento che sta assegnando ad un autore.
	5. Tra la lista degli argomenti assegnati agli autori l'Editorial Central Staff può scartare quelli che sono stati rifiutati dagli autori per rassegnarli ad un altro autore o può inserire un ulteriore commento per ritentare ad assegnare l'argomento all'autore
<i>Exit condition</i>	6. L'Editorial Central Staff conferma l'assegnazione per gli argomenti che ha appena associato agli autori .
<i>Special requirements</i>	Il sistema notifica ad ogni autore la ricezione di uno o più argomenti inviati dall' Editorial Central Staff
<hr/>	
<i>Use case name</i>	ContractingArguments
<i>Participating actor</i>	Iniziato da Author
<i>Entry condition</i>	1. L'Author attiva la funzionalità per contrattare gli argomenti assegnatigli dall'Editorial Central Staff.
<i>Flow of events</i>	2. Il sistema risponde presentando un interfaccia dove è possibile visualizzare la lista degli argomenti assegnatigli con la loro



	descrizione e il corso di appartenenza di ogni argomento.
	3. L'Author seleziona uno degli argomenti della lista e decide se accettarlo o meno.
	4. Se l'Author accetta un argomento che gli è stato assegnato questo viene inserito in una lista di argomenti che l'Author dovrà sviluppare; in caso contrario l'Author può inserire un commento per giustificare la sua rinuncia o per chiedere chiarimenti sull'argomento e conferma la sua rinuncia. In entrambi i casi l'argomento può essere accompagnato da un commento e viene eliminato dalla lista degli argomenti assegnati.
	5. L'Author ripete le operazioni precedenti per gli altri argomenti della lista che gli sono stati assegnati.
<i>Exit condition</i>	6. L'Author ha accettato o rifiutato tutti gli argomenti della lista degli argomenti assegnati.
<i>Special requirements</i>	Il sistema notifica all'Editorial Central Staff il rifiuto o l'accettazione di un argomento assegnato all'Author.
<hr/>	
<i>Use case name</i>	DevelopArgument
<i>Participating actor</i>	Iniziato da Author
<i>Entry condition</i>	1. L'Author attiva la funzionalità per sviluppare gli argomenti che gli sono stati inviati e che ha accettato.
<i>Flow of events</i>	2. Il sistema risponde presentando un interfaccia dove è possibile visualizzare la lista degli argomenti da sviluppare, lo stato

di avanzamento degli stessi in termini di pagine già prodotte e il relativo corso di appartenenza dell'argomento.

3. L'Author seleziona un argomento e può scegliere di aggiungere una nuova pagina, modificarne una già esistente o cancellarla.
4. Se l'Author seleziona l'opzione per aggiungere una nuova pagina il sistema mostra un editor di pagine con la possibilità di costruire la pagina a partire da uno scheletro predefinito altrimenti se viene selezionata l'opzione per modificare una pagina selezionata il sistema mostra un editor che contiene la pagina che si vuole modificare.
5. L'Author sviluppa la pagina inserendo del testo, delle immagini, dei link e delle informazioni atomiche (vedi caso d'uso `EditAtomicInformation`).
6. L'Author salva la pagina appena creata e questa viene aggiunta alla lista di pagine che mostrano lo stato di avanzamento dell'argomento.
7. L'Author ripete le operazioni di creazione, modifica e cancellazione di una pagina tutte le volte necessarie.
8. L'Author termina lo sviluppo delle pagine dell'argomento o lo sospende per riprenderlo in seguito.

*Exit condition*

*Special requirements*

I link presenti nelle pagine non possono fare riferimento a risorse locali (devono essere delle URL del tipo `http:// ftp://`) a meno che non si tratti di informazioni atomiche .

<i>Use case name</i>	EditAtomicInformation
<i>Participating actor</i>	Iniziato da Author
<i>Entry condition</i>	1. L'Author mentre sta sviluppando una pagina seleziona del testo sceglie l'opzione per inserire un informazione atomica.
<i>Flow of events</i>	2. Il sistema risponde presentando un interfaccia dove è possibile scegliere se aggiungere una nuova informazione atomica o se ricercarne una già pronta e sviluppata precedentemente da un Author. 3. Se l'Author seleziona l'opzione per inserire un informazione atomica già pronta viene visualizzata una lista di informazioni atomiche disponibili tra cui l'Author puo effettuare una ricerca. 4. L'Author sceglie se mostrare un'anteprima di tale informazione atomica ed eventualmente ne conferma l'inserimento. 5. Se l'Author seleziona l'opzione per inserire una nuova informazione atomica il sistema mostra un interfaccia su cui l'Author ricerca in locale il file da utilizzare per l'informazione atomica; l'Author conferma la scelta del file e il sistema mostra un form per riempire i metadati relativi all'informazione atomica. 6. L'Author riempie i campi dei metadati e conferma l'invio di questi ultimi.
<i>Exit condition</i>	7. L'Author torna a sviluppare la pagina dopo aver inserito l'informazione atomica.
<i>Special requirements</i>	Nessuno.

---

<i>Use case name</i>	ExportArgument
<i>Participating actor</i>	Iniziato da Author
<i>Entry condition</i>	1. L'Author ha terminato la produzione delle pagine per un dato argomento e seleziona la funzionalità per esportarlo.
<i>Flow of events</i>	2. Il sistema risponde presentando un interfaccia dove è possibile ordinare in maniera sequenziale le pagine prodotte per l'argomento. 3. L'Author ordina le pagine e conferma l'operazione; il sistema adesso mostra un form per l'inserimento dei metadati relativi all'argomento. 4. L'Author riempie i campi dei metadati e ne conferma l'invio.
<i>Exit condition</i>	5. Il sistema aggiorna la lista degli argomenti in fase di sviluppo cancellando l'argomento appena esportato.
<i>Special requirements</i>	L'Editorial Central Staff riceve una notifica quando l'autore esporta un argomento.

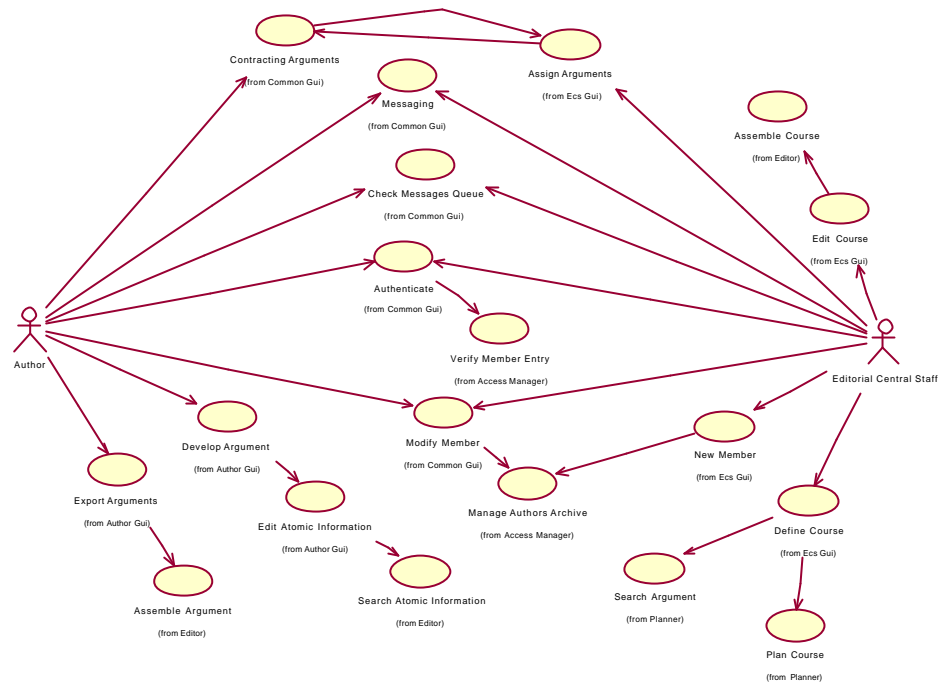
<i>Use case name</i>	EditCourse
<i>Participating actor</i>	Iniziato da Editorial Central Staff
<i>Entry condition</i>	1. L'Editorial Central Staff ha ricevuto tutti gli argomenti sviluppati relativi ad un corso e adesso lo vuole assemblare.
<i>Flow of events</i>	2. L'Editorial Central Staff sceglie la funzionalità per assemblare un corso. 3. Il sistema mostra un interfaccia in cui viene visualizzata una struttura del corso con i suoi argomenti.

	4. L'Editorial Central Staff ordina i sottocorsi e gli argomenti all'interno del corso per decidere l'ordine in cui questo verrà fruito.
	5. Per il corso e per ogni sottocorso l'Editorial Central Staff aggiunge i metadati relativi in un'apposita form.
<i>Exit condition</i>	6. L'Editorial Central Staff conferma l'operazione di assemblamento del corso.
<i>Special requirements</i>	Nessuno.

---

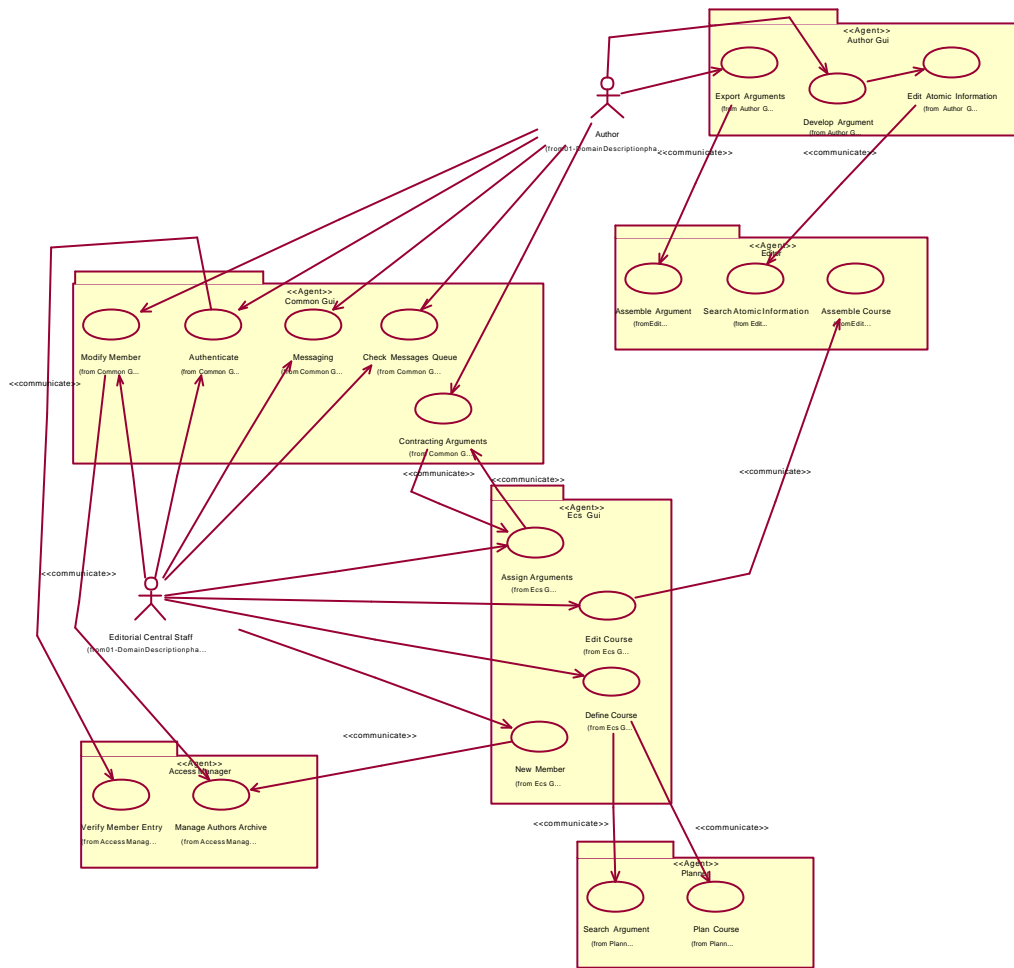
## ***Progetto del sistema***

## 1 Fase 1: Domain Description



**Figura 2: Diagramma dei casi d'uso della fase Domain Description.**

## 2 Fase 2: Agent Identification



**Figura 3: diagramma del sistema multi-agente nella fase Agent Identification.**

I package presenti nel diagramma identificano i singoli agenti; i casi d'uso all'interno di ogni package sono le funzionalità dell'agente. In questo diagramma si nota che le relazioni tra casi d'uso appartenenti a package differenti hanno lo stereotipo "communicate"; tali relazioni individuano le interazioni sociali tra i diversi agenti all'interno del sistema multi-agente. Come è possibile notare gli attori Editorial Central Staff e Author interagiscono soltanto con gli agenti GUI che sono: AuthorGui, EcsGui e CommonGui. In una prima versione del progetto gli attori interagivano direttamente con l'intera società di agenti; questa soluzione è stata scartata poiché nel caso in cui si fosse deciso di cambiare le interfacce che gli agenti mettono a disposizione degli attori, si sarebbe dovuto rivoluzionare



l'intero sistema multiagente. Al contrario la soluzione proposta, permette di intervenire soltanto su un numero limitato di agenti qualora si vogliano apportare delle modifiche all'interfaccia e in generale alle interazioni tra attori e agenti. Un altro aspetto che è stato oggetto di diverse modifiche è dato dagli agenti GUI; in una prima versione del progetto gli attori potevano comunicare con un unico agent GUI che offriva le funzionalità necessarie sia per gli autori che per l'editorial central staff. Tale soluzione ha portato allo sviluppo di un agente notevolmente complesso che è stato dunque abbandonato sia per la difficoltà implementative e progettuali che questo avrebbe sollevato, sia per il fatto che l'agente GUI in soluzione unica avrebbe rallentato notevolmente la piattaforma ad agenti JADE e quindi tutto il sistema di agenti. Una soluzione più efficace, che è quella proposta, ha affrontato il problema dividendo l'agente GUI in tre agenti più semplici: AuthorGui, EcsGui e CommonGui. Descriviamo adesso gli agenti coinvolti all'interno del MAS:

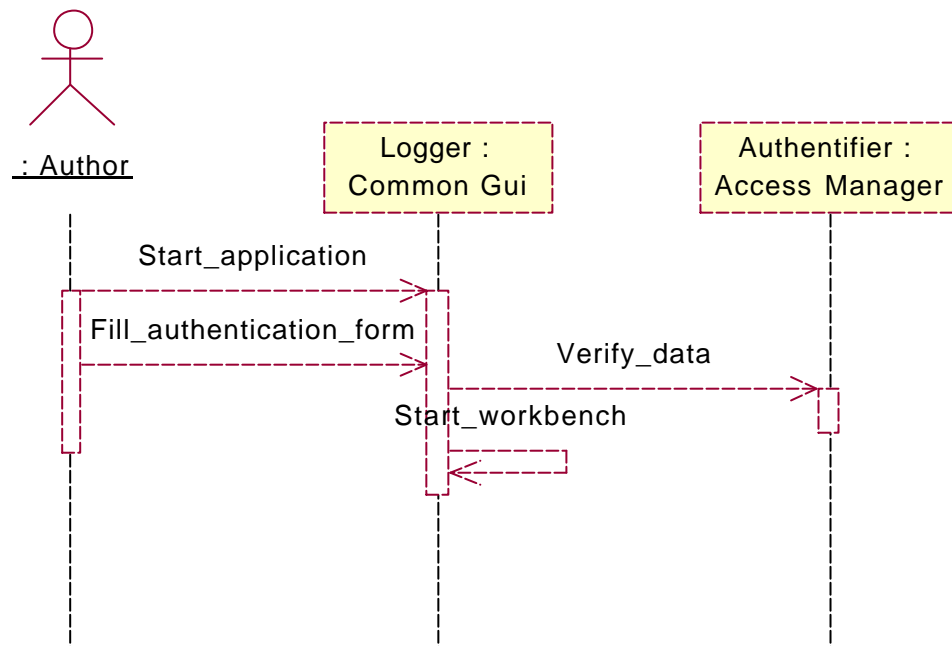
- **Agente Access Manager:** questo agente si occupa delle operazioni necessarie per l'autenticazione degli utenti nel sistema siano essi gli autori o l'editorial central staff. Un'altra responsabilità di questo agente è quella di mantenere un archivio degli utenti registrati nel sistema e gestirne i loro profili;
- **Agente Author Gui:** questo agente gestisce le interfacce messe a disposizione dell'autore per la gestione e la produzione dei contenuti. Quando un argomento viene terminato l'agente si preoccupa di esportare l'argomento verso l'editorial central staff;
- **Agente Common Gui:** questo agente gestisce le interfacce e le funzionalità comuni messe a disposizione dell'autore e dell'editorial central staff; tali funzionalità includono invio di messaggi, invio di e-mail, interfaccia di autenticazione, contrattazione degli argomenti e gestione del profilo utente;
- **Agente Ecs Gui:** questo agente gestisce le interfacce messe a disposizione dell'editorial central staff; tra le funzionalità di questo agente abbiamo la definizione e la gestione dei corsi, l'assegnazione degli argomenti agli autori e la gestione dei profili degli autori;
- **Agente Editor:** questo agente assembla gli argomenti che vengono esportati dagli autori, assembla i corsi i cui argomenti sono stati

tutti completati e gestisce un meccanismo di ricerca per riutilizzare le informazioni atomiche nella produzione di argomenti;

- **Agente Planner:** questo agente gestisce i corsi definiti dall'editorial central staff; inoltre fornisce un meccanismo di ricerca per riutilizzare argomenti già sviluppati da inserire nei corsi che l'editorial central staff definisce.

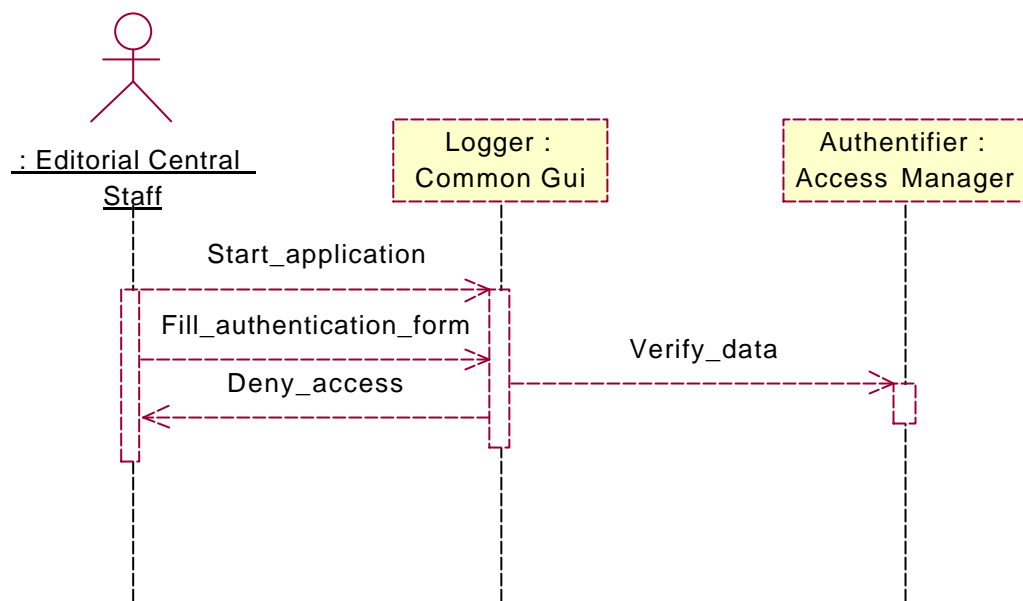
### 3 Fase 3: Role Identification

In questa fase procediamo all'identificazione dei ruoli dei singoli agenti a partire dagli scenari che si ottengono dalla fase Agent Identification.



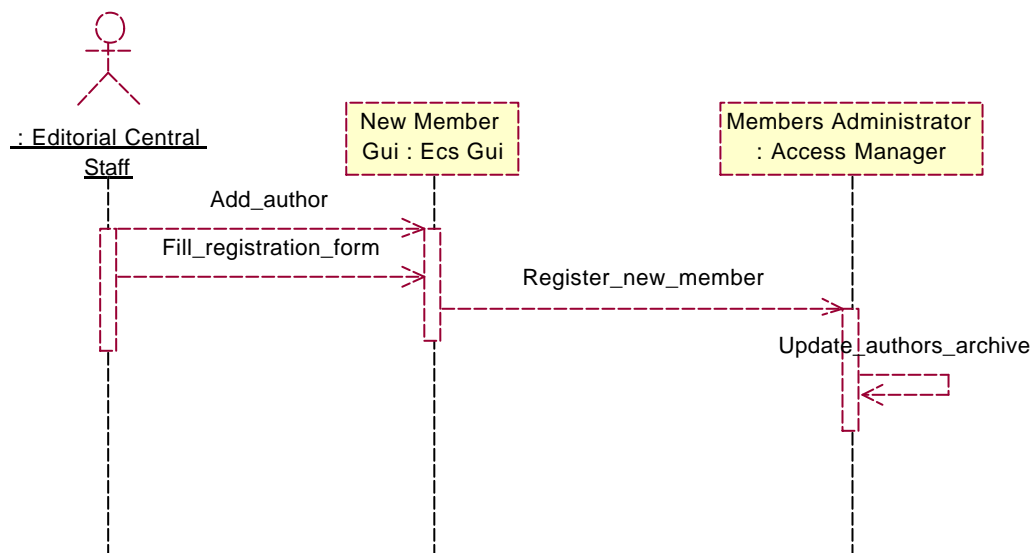
**Figura 4: Diagramma di sequenza "Author Authentication Ok".**

In questo diagramma di sequenza un autore lancia il programma di authoring e per accedere alle funzionalità del sistema esegue una procedura di autenticazione che va a buon fine e mostra all'autore il suo banco di lavoro.



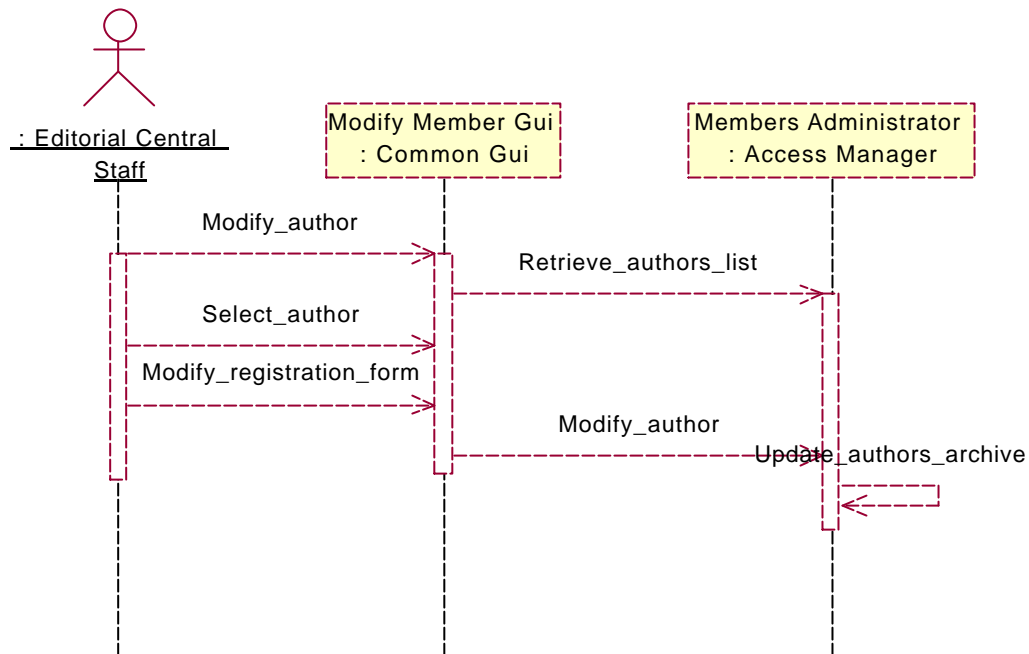
**Figura 5: Diagramma di sequenza " Ecs Authentication Ko".**

In questo diagramma di sequenza l'editorial central staff avvia il programma ed esegue la procedura di autenticazione; tuttavia i dati forniti al sistema non sono corretti e quindi l'accesso viene annullato.



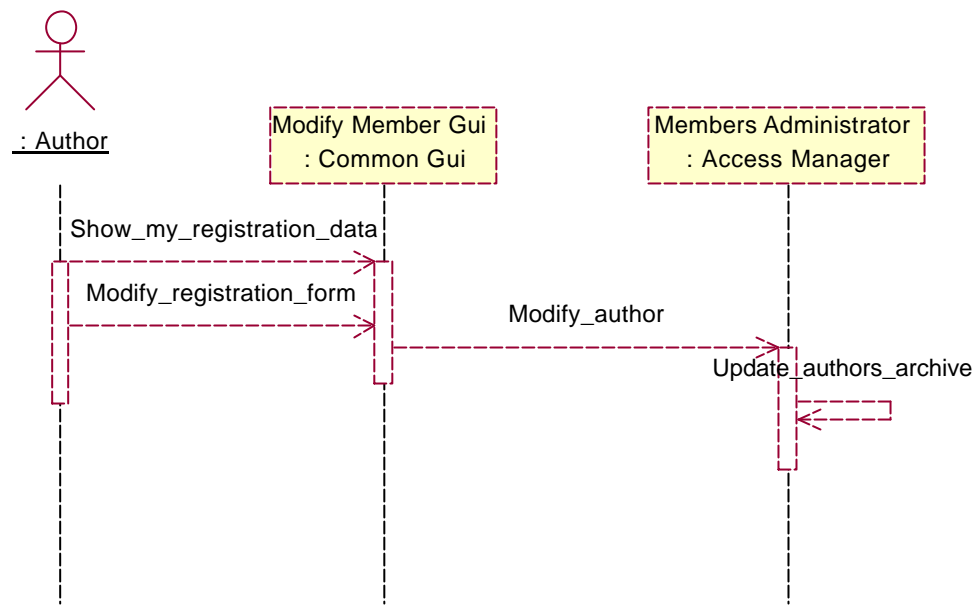
**Figura 6: Diagramma di sequenza "New Member Entry".**

In questo diagramma l'editorial central staff sceglie la funzionalità per aggiungere un nuovo autore, riempie il profilo del nuovo autore e alla fine ne conferma l'inserimento che viene memorizzato nell'archivio.



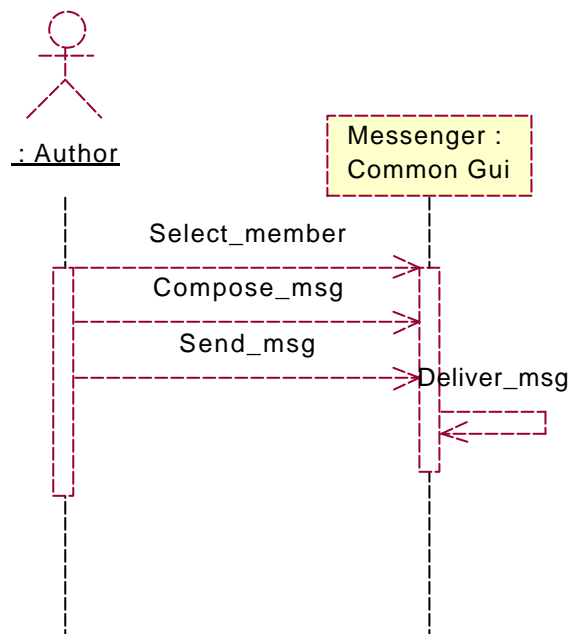
**Figura 7: Diagramma di sequenza "Modify Member Entry".**

L'editorial central staff attiva la funzionalità per modificare il profilo di un utente; dopo aver scelto l'utente di cui desidera modificarne il profilo interviene sui campi relativi e conferma la modifica del profilo effettuata. Tale modifica viene memorizzata all'interno dell'archivio.



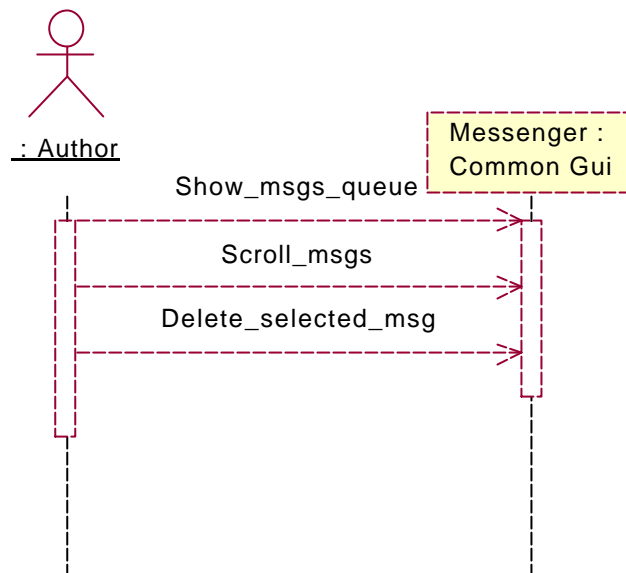
**Figura 8: Diagramma di sequenza "Modify Own Member Entry".**

Come per il diagramma precedente soltanto che gli utenti possono modificare solo il proprio profilo.



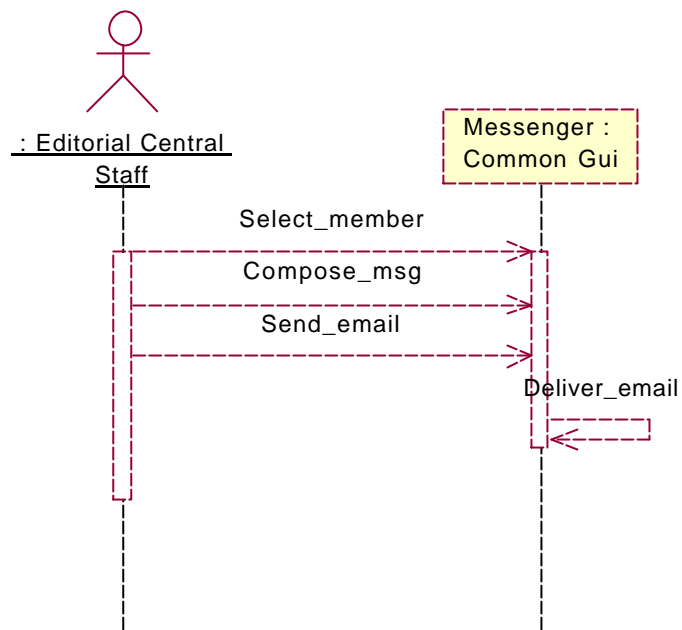
**Figura 9: Diagramma di sequenza "Messaging From Author".**

Un autore seleziona un utente del sistema per inviargli un messaggio, compone il messaggio da inviare e lo inoltra. Il sistema consegna il messaggio all'utente desiderato.



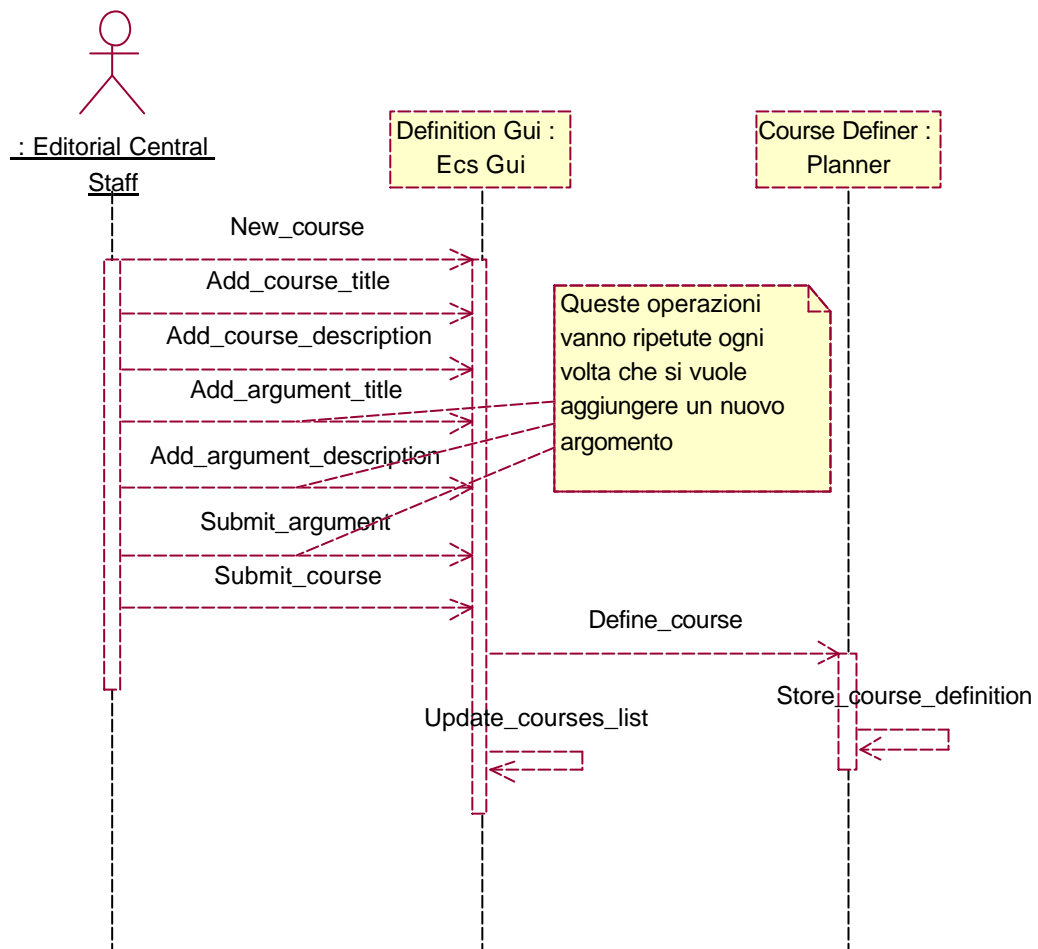
**Figura 10: Diagramma di sequenza "Message Queue Checking".**

Un autore verifica la coda dei messaggi ricevuti; può scorrere la lista dei messaggi e osservarne il contenuto. L'autore può decidere di cancellare o meno i messaggi che ha già letto.



**Figura 11: Diagramma di sequenza "Emailing From Ecs".**

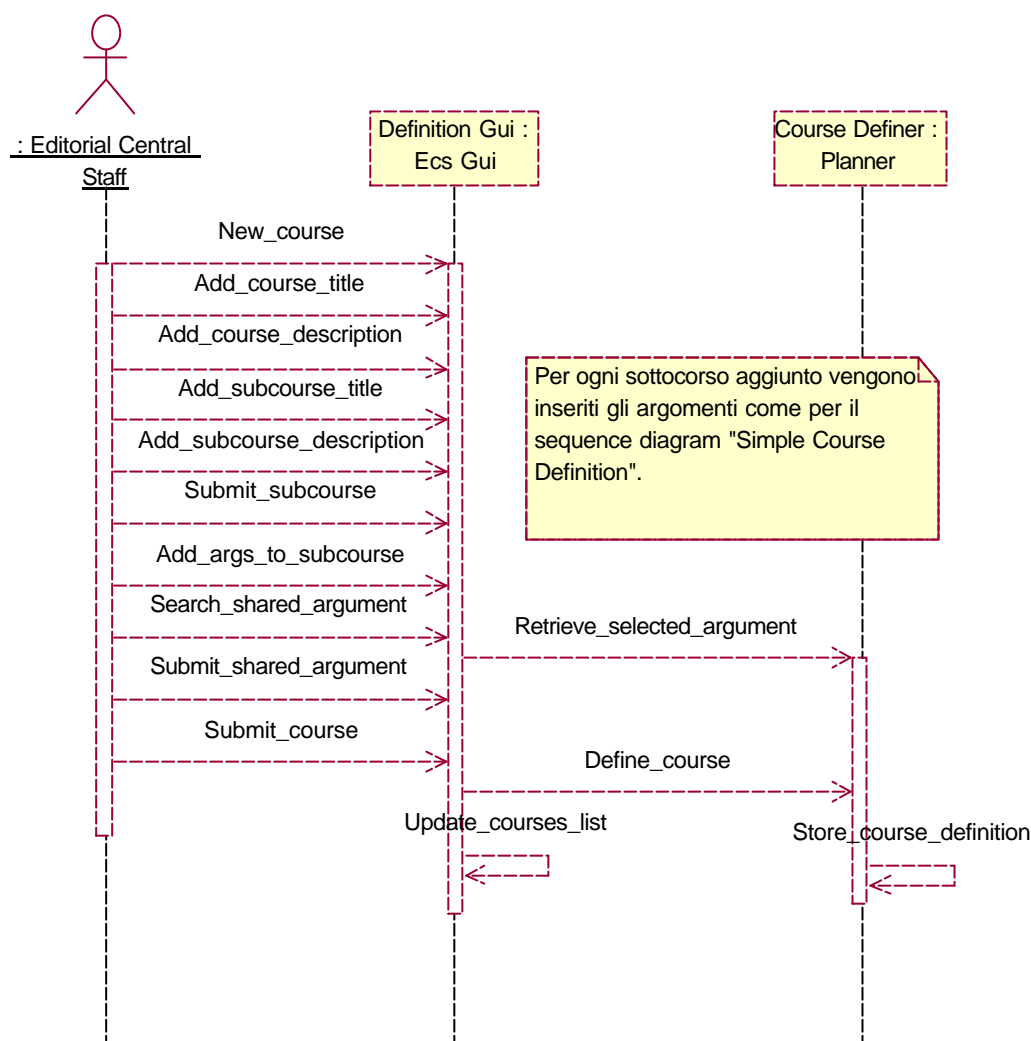
L'editorial central staff seleziona un utente per inviargli un'email, riempie il corpo dell'email e ne conferma l'invio. Il sistema provvede allo smistamento della posta all'indirizzo desiderato.



**Figura 12: Diagramma di sequenza "Simple Course Definition".**

L'editorial central staff seleziona la funzionalità per definire un nuovo corso, da un titolo e una breve descrizione al corso e comincia ad inserire all'interno del corso gli argomenti che desidera, ad ognuno dei quali da un titolo ed una breve descrizione. L'editorial central staff conferma il corso definito e il sistema si preoccupa di organizzare in archivio il corso appena definito e aggiorna di conseguenza le interfacce relative.





**Figura 13: Diagramma di sequenza "Complex Course Definition".**

L'editorial central staff seleziona la funzionalità per definire un nuovo corso, cui da un titolo e una breve descrizione. Al corso vengono aggiunti dei sottocorsi anch'essi caratterizzati da un titolo e da una descrizione; all'interno dei sottocorsi l'editorial central staff aggiunge gli argomenti che desidera come per il diagramma precedente. L'editorial central staff ricerca all'interno dell'archivio degli eventuali argomenti già sviluppati che può inserire all'interno del corso. L'editorial central staff conferma il corso appena definito e il sistema si preoccupa di organizzare il corso all'interno dell'archivio e di aggiornare le interfacce relative.

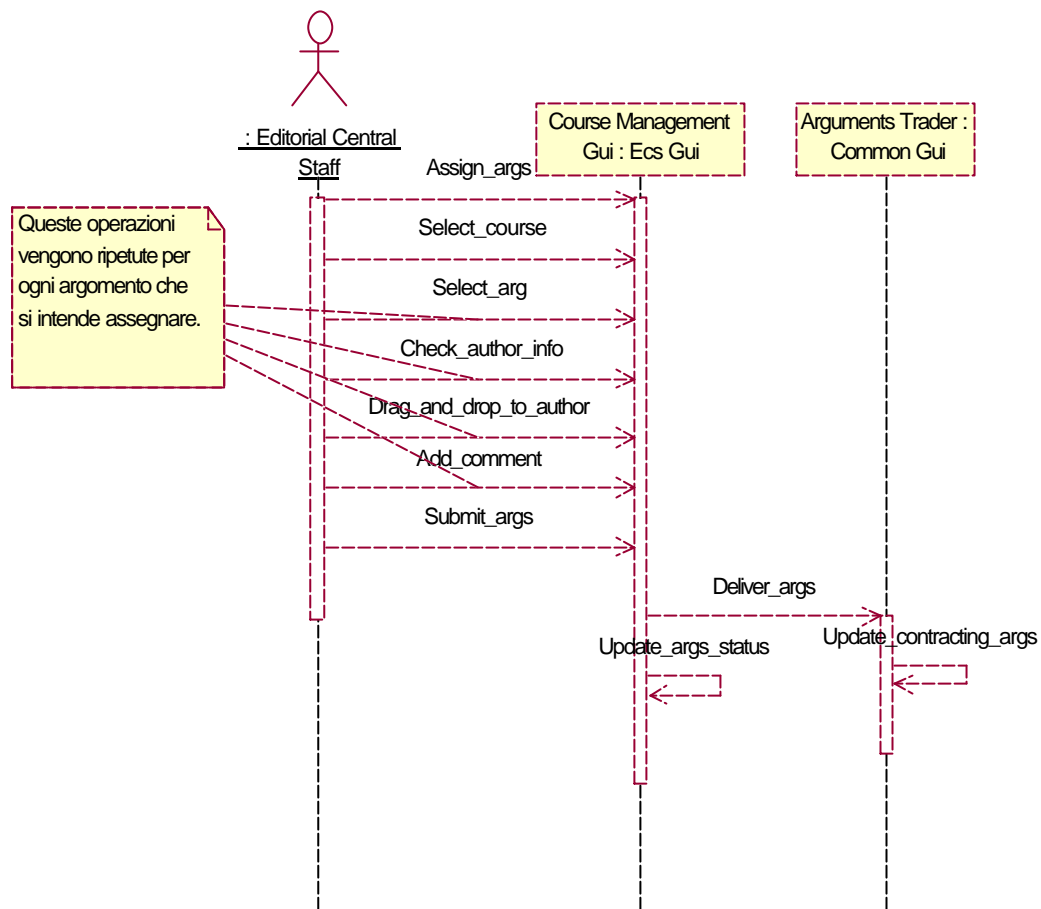
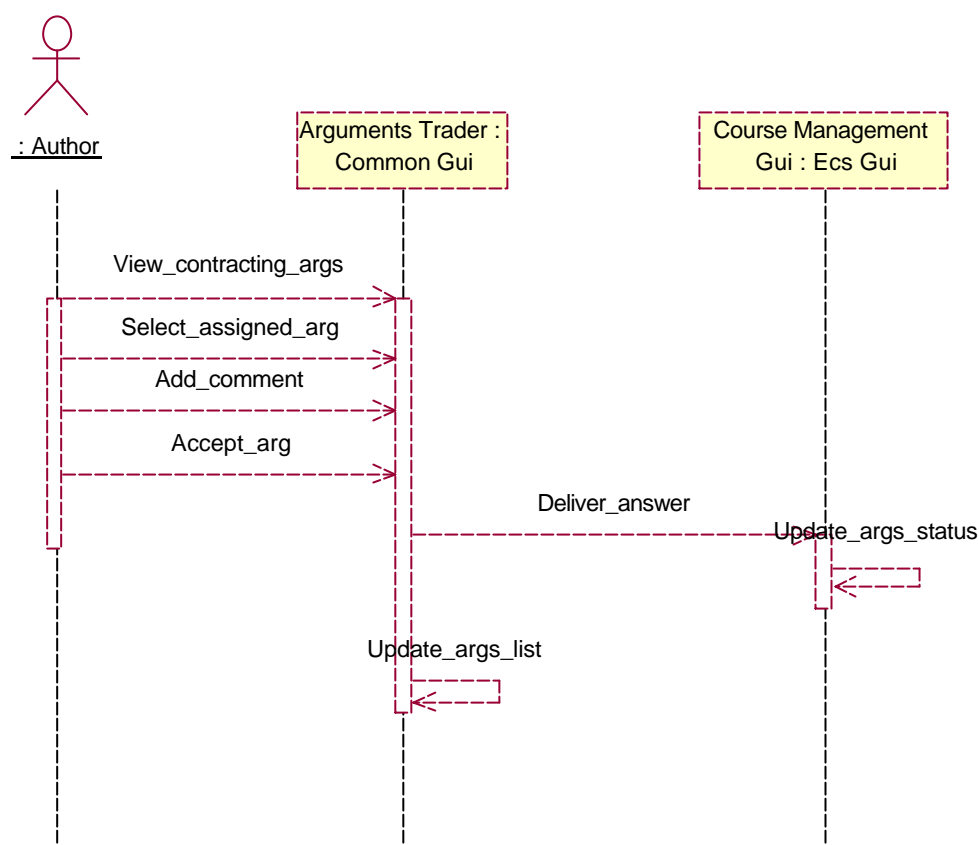


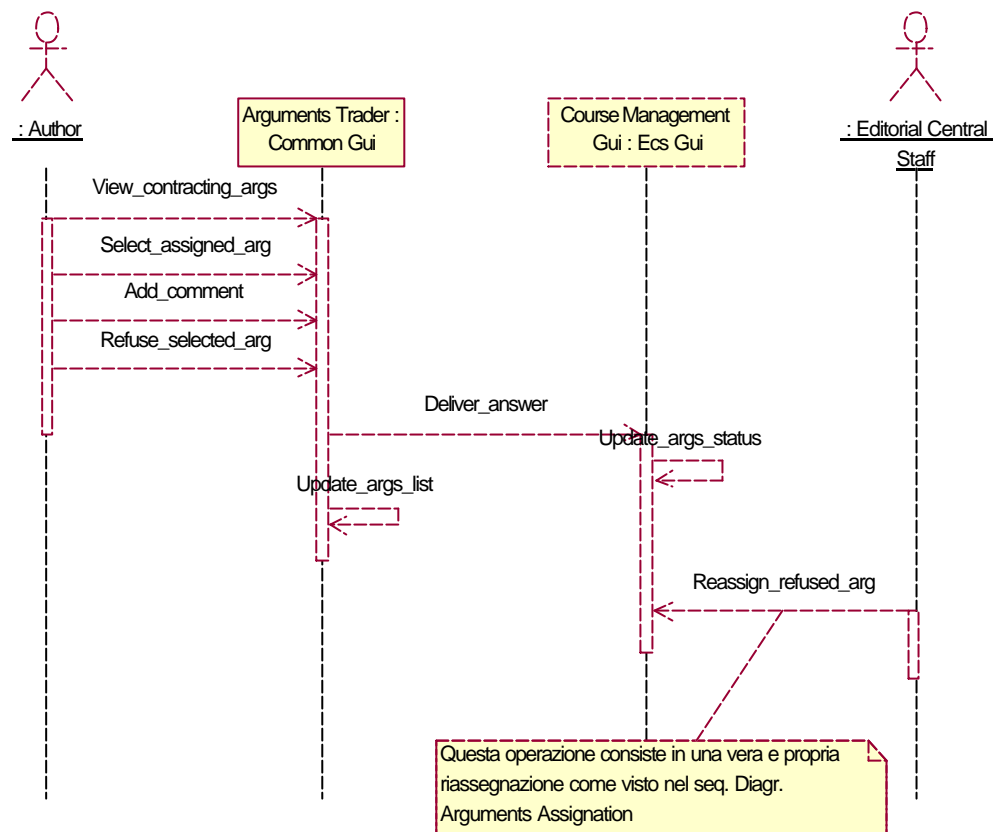
Figura 14: Diagramma di sequenza "Arguments Assignment".

L'editorial central staff seleziona la funzionalità per l'assegnazione degli argomenti agli autori; seleziona il corso di cui vuole assegnare gli argomenti e va trascinando via via gli argomenti dal corso verso l'autore cui lo vuole assegnare. Prima di assegnare un argomento ad un autore l'editorial central staff può visualizzare il profilo dell'autore; in questo modo può determinare se l'autore cui sta assegnando l'argomento abbia le competenze adatte per svilupparlo o meno. Per ogni argomento assegnato l'editorial central staff può inserire un commento. L'editorial central staff conferma l'assegnazione degli argomenti agli autori; il sistema provvede a smistare gli argomenti ai relativi autori e ad aggiornare le interfacce relative.



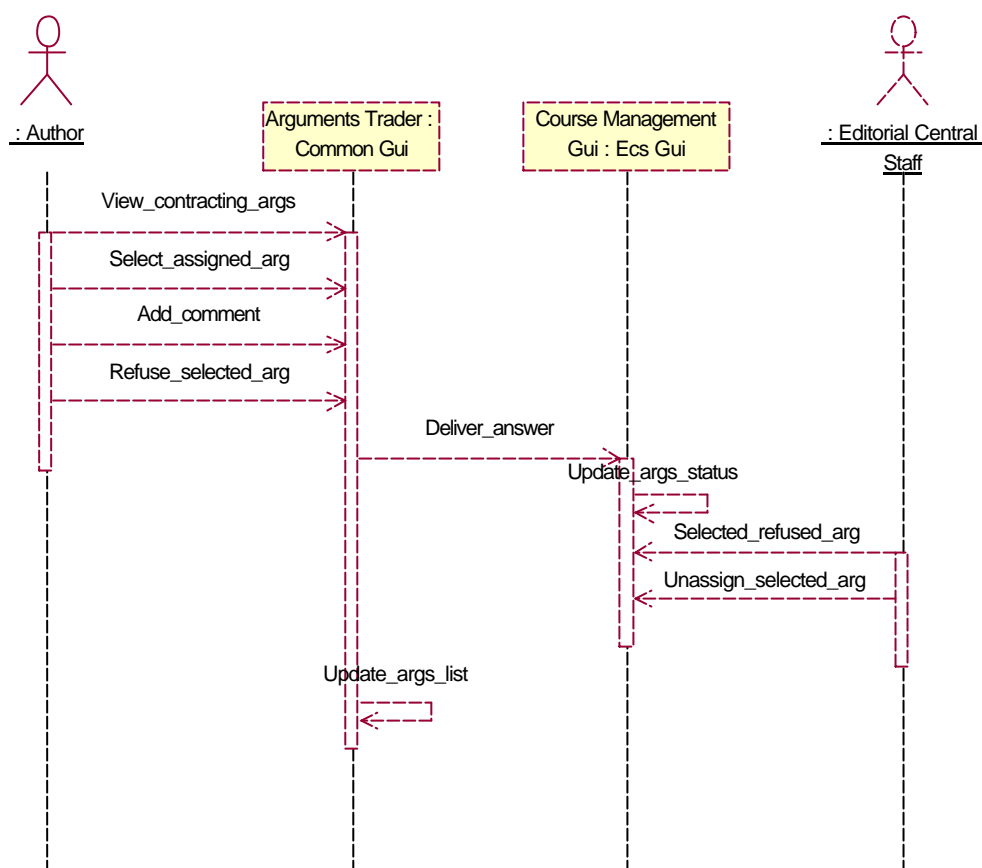
**Figura 15: Diagramma di sequenza "Accept Argument".**

L'autore visualizza gli argomenti assegnatigli, ne seleziona uno e lo accetta. L'autore può aggiungere un commento all'argomento che sta accettando. Il sistema consegna la risposta all'editorial central staff e provvede a aggiornare le interfacce relative.



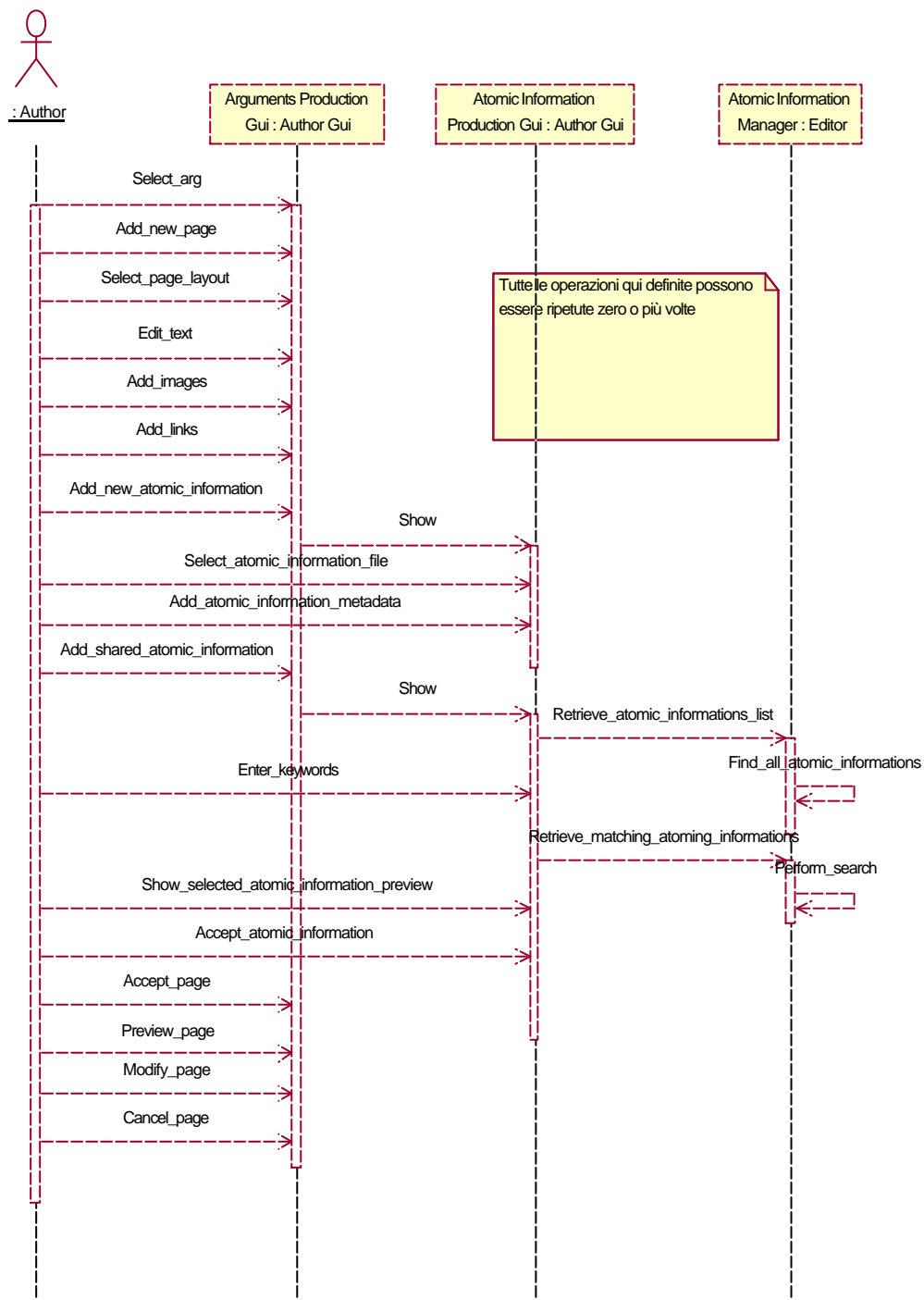
**Figura 16: Diagramma di sequenza "Refuse and Reassign Argument".**

L'autore visualizza gli argomenti assegnatigli, ne seleziona uno e lo rifiuta; l'autore può aggiungere un commento all'argomento che sta rifiutando; il sistema consegna la risposta all'editorial central staff che riprova a riassegnare tale argomento allo stesso autore aggiungendo nel commento chiarimenti riguardo l'argomento.



**Figura 17: Diagramma di sequenza "Refuse and Unassign Argument".**

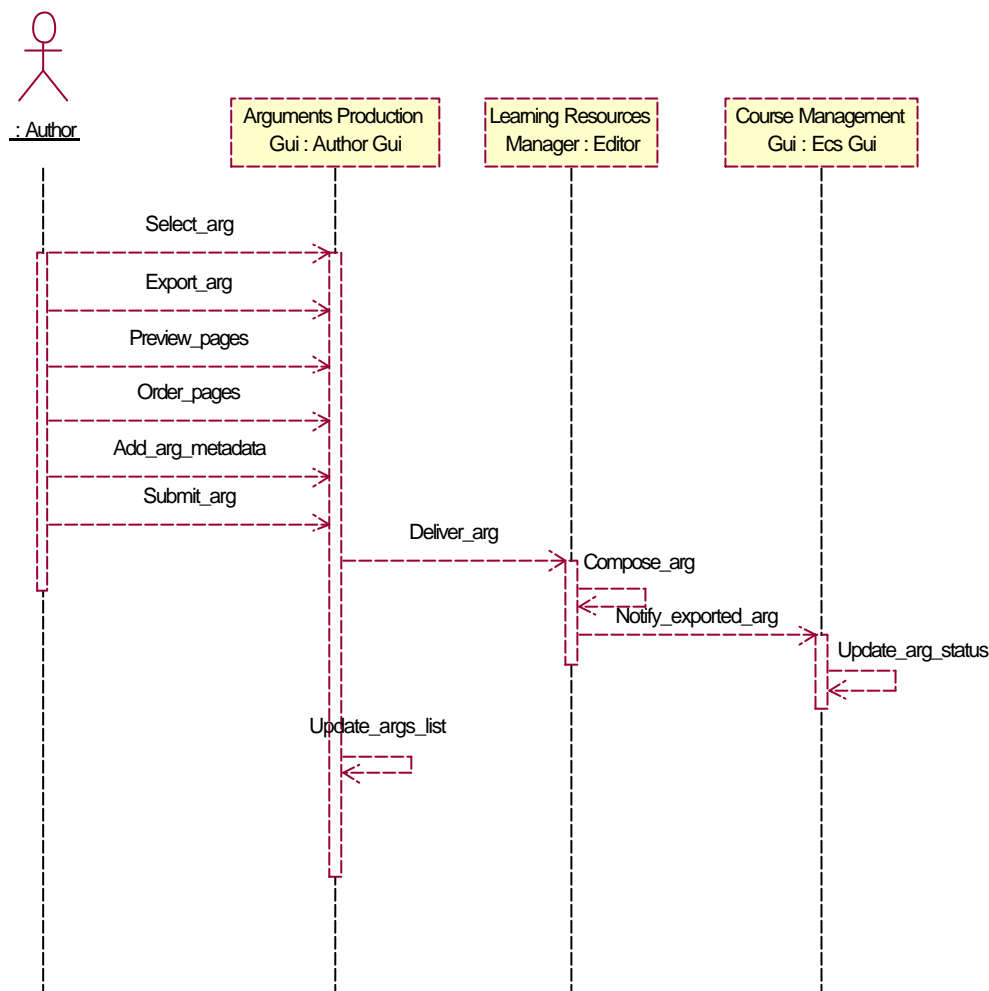
L'autore visualizza gli argomenti assegnatigli, ne seleziona uno e lo rifiuta come per il diagramma precedente; l'editorial central staff decide di cancellare l'assegnazione di tale argomento all'autore per assegnarlo così ad un altro autore.



**Figura 18: Diagramma di sequenza "Develop Argument".**

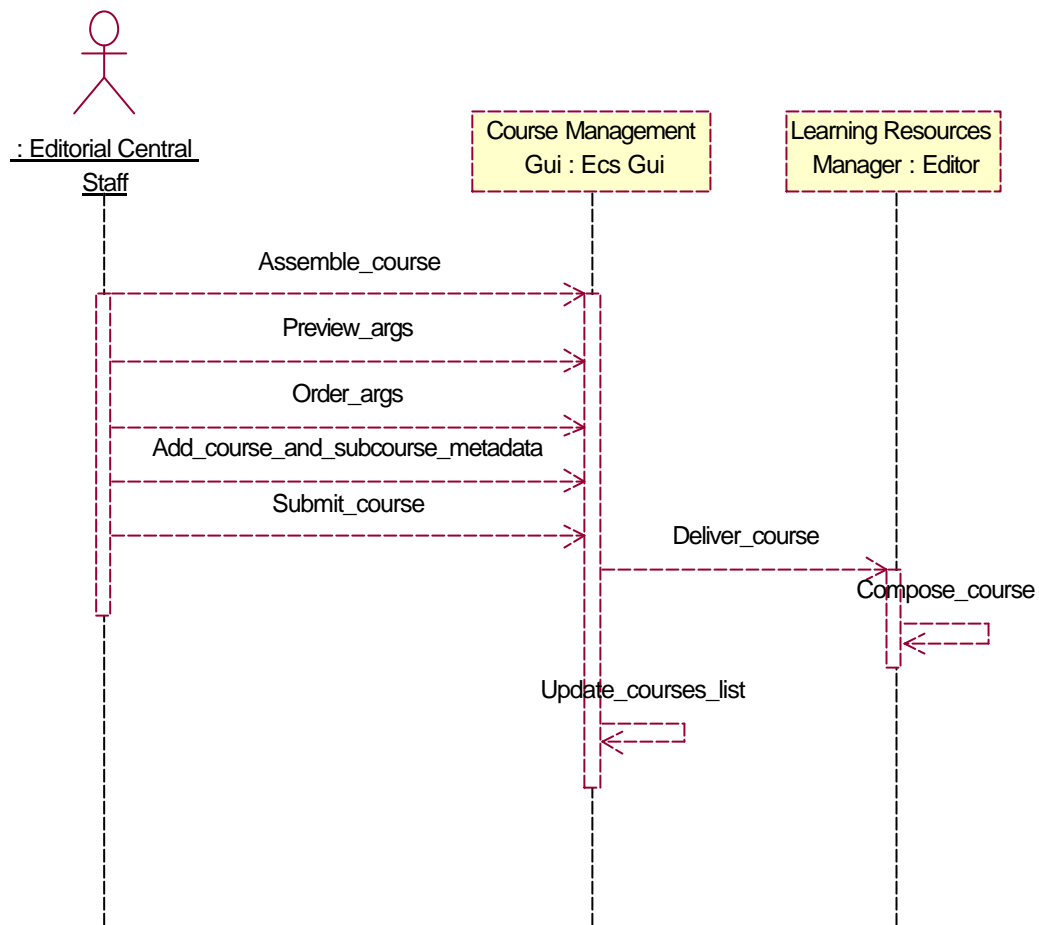
L'autore seleziona un argomento da sviluppare, aggiunge una nuova pagina che può essere creata con l'ausilio di layout predefiniti. All'interno della pagina l'autore può inserire testo, immagini, link ed informazioni atomiche che

comportano l'inserimento dei relativi metadati; il sistema mette a disposizione dell'autore uno strumento per ricercare informazioni atomiche già sviluppate e per inserirle nella pagina alla quale sta lavorando. Tutte queste operazioni vengono ripetute per tutte le pagine che si vogliono inserire. L'autore può decidere di modificare o di cancellare pagine che ha già sviluppato.



**Figura 19: Diagramma di sequenza "Export Argument".**

L'autore seleziona un argomento che contiene almeno una pagina e seleziona la funzionalità per esportarlo; l'autore ordina le pagine secondo l'ordine che preferisce aiutandosi con la visualizzazione delle anteprime delle pagine. Successivamente inserisce i metadati relativi all'argomento che si sta esportando e ne conferma l'esportazione. Il sistema provvede ad esportare e a memorizzare nell'archivio l'argomento sviluppato dall'autore.



**Figura 20: Diagramma di sequenza "Assemble Course".**

L'editorial central staff seleziona la funzionalità per assemblare un corso i cui argomenti sono stati tutti completati; ordina gli argomenti all'interno del corso aiutandosi con la visualizzazione di un anteprima degli argomenti, aggiunge i metadati al corso e agli eventuali sottocorsi e conferma l'operazione. Il sistema procede con le operazioni necessarie per predisporre il corso alla pubblicazione da parte di un LMS.

La tabella che segue mostra il risultato della fase *Role Identification*, elenca tutti i ruoli degli agenti che sono stati identificati e ne fornisce una loro descrizione.



### 3.1 Ruoli identificati

Name Role	Agent which plays it	Description	Responsibilities
Messenger	Common Gui	Interagisce con l'utente per la composizione del messaggio ed il suo destinatario ed invia il messaggio o l'email. Gestisce inoltre la coda di messaggi ricevuti.	Consegnare il messaggio all'agente CommonGui. Smistare i messaggi di posta elettronica. Gestire la coda dei messaggi. Gestire l'interfaccia dei messaggi.
Logger	Common Gui	In questo ruolo il Common Gui riceve i dati di autenticazione dall'attore e li invia per verifica all'Agente Access Manager. A seconda della risposta dell'Access Manager il ruolo permette o impedisce l'accesso alle funzionalità del sistema.	Ricevere i dati di autenticazione dagli attori e passarli all'agente Access Manager. Permettere o negare l'accesso al sistema. Gestire l'interfaccia di autenticazione.
Authentifier	Access Manager	Verifica i dati di login inviati dal Common Gui e risponde di conseguenza.	Verificare I dati di autenticazione degli utenti. Informare l'agente common gui sulla correttezza dei dati di autenticazione.
Modify Member Gui	Common Gui	L'utente interagisce con l'agente riempiendo una serie di campi con i dati dell'utente che vuole modificare o che vuole cancellare.	Ricevere i dati dei profili degli attori. Gestire l'interfaccia per la visualizzazione dei profili utente. Richiedere all'agente Access Manager la lista degli utenti. Inviare i dati dei profili utente all'agente Access Manager.
New Member Gui	Ecs Gui	L'utente interagisce con l'agente riempiendo una serie di campi con i dati del nuovo utente da inserire.	Ricevere dall'editorial central staff i dati di un nuovo utente. Gestire l'interfaccia del profilo utente. Inviare i dati del nuovo utente all'agente Access

			Manager.
Members Administrator	Access Manager	Questo ruolo di occupa di ricevere dagli agenti le informazioni per l'inserimento, la modifica o la cancellazione dei membri del sistema e le memorizza.	Ricevere i dati dei profili degli utenti. Aggiungere, modificare o cancellare un utente dall'archivio. Consegnare la lista degli utenti e i relativi profili all'agente Common Gui
Definition Gui	Ecs Gui	In questo ruolo l'utente interagisce con l'agente per definire un nuovo corso inserendo e modificando sottocorsi e argomenti, e inserendo argomenti già sviluppati.	Ricevere dall'editorial central staff i dati relativi alla definizione di un nuovo corso. Gestire l'interfaccia per la definizione dei corsi. Richiedere al planner argomenti già sviluppati. Inviare al Planner i corsi definiti.
Atomic Information Manager	Editor	In questo ruolo l'Editor gestisce le informazioni atomiche condivise, fornendo servizi di ricerca e mantenimento.	Riceve dall'agente AuthorGui la richiesta per mostrare le informazioni atomiche. Invia la lista delle informazioni atomiche. Effettua la ricerca nell'archivio per una data informazione atomica.
Atomic Information Production Gui	Author Gui	In questo ruolo l'autore interagisce con l'agente per la produzione o la ricerca di informazioni atomiche da inserire all'interno delle pagine.	Gestisce l'interfaccia per la produzione e la ricerca di informazioni atomiche. Riceve dall'autore le richieste per effettuare ricerche di informazioni atomiche. Richiede all'agente Editor di effettuare la ricerca di informazioni atomiche. Immagazzina i metadati relativi ad un informazione atomica.
Course Definer	Planner	Riceve le informazioni sul nuovo corso dall'agente Ecs Gui e le organizza creando un'apposita struttura dati.	Riceve dall'agente Ecs Gui i dati della definizione dei corsi. Organizza nell'archivio i corsi definiti. Effettua la ricerca di argomenti già sviluppati. Restituisce informazioni sulla ricerca

			di argomenti all'agente Ecs Gui.
Arguments Production Gui	Author Gui	In questo ruolo l'utente interagisce con l'Author Gui per la vera e propria produzione di un argomento, dove questa comprende: creazione e modifica di pagine, organizzazione dell'argomento ed esportazione dello stesso.	Gestisce l'interfaccia di produzione dell'argomento. Immagazzina i metadati delle informazioni atomiche e degli argomenti. Esporta gli argomenti prodotti che vengono memorizzati in archivio.
Learning Resources Manager	Editor	In questo ruolo l'editor si preoccupa di assemblare e organizzare argomenti e corsi interamente sviluppati.	Riceve gli argomenti sviluppati dagli autori. Organizza gli argomenti all'interno dell'archivio. Crea i file XML dei metadati delle risorse. Assembla i corsi i cui argomenti sono stati completati.
Course Management Gui	Ecs Gui	In questo ruolo l'Ecs interagisce con l'agente per la assegnazione, contrattazione e completamento degli argomenti di un corso.	Gestisce l'interfaccia di sviluppo dei corsi. Riceve la contrattazione degli argomenti dall'agente Common Gui. Riceve dall'editorial central staff l'ordine degli argomenti all'interno del corso. Riceve i metadati dei corsi inseriti dall'editorial central staff. Richiede all'agente Editor di assemblare i corsi.
Arguments Trader	Common Gui	In questo ruolo il Common Gui si occupa delle operazioni di assegnazione e contrattazione delle assegnazioni tra autori ed Ecs.	Gestisce l'interfaccia di contrattazione degli argomenti. Riceve gli argomenti assegnati dall'editorial central staff. Consegna all'agente Ecs Gui la contrattazione degli argomenti.

**Figura 21: Identificazione dei ruoli degli agenti.**

## 4 Fase 4: Tasks Specification

In questa fase vengono descritti i task che svolge ogni singolo agente.

### 4.1 Agente: Access Manager

Agente che si occupa della gestione dei membri del sistema.

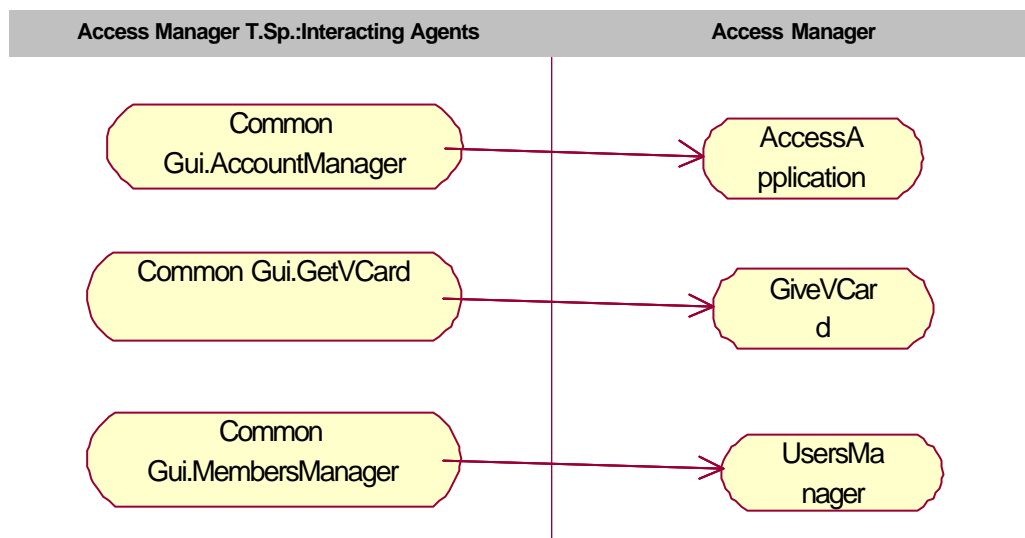


Figura 22: Diagramma dei task per l'agente Access Manager.

#### 4.1.1 Task: AccessApplication

Description	Verifica i dati di autenticazione inviati dal Common Gui.
Action	Effettua Query nel database per verificare la correttezza dei dati di autenticazione
Data	request_conv agent_to_request
Behavior	Il task attende per l'arrivo di dati di autenticazione da parte del task <i>CommonGui.AccountManager</i> , effettua una query sul database degli utenti e verifica l'esistenza dei dati; il task ritorna a <i>CommonGui.AccountManager</i> l'esito di tale ricerca.

#### 4.1.2 Task: GiveVCard

Description	Ricerca la VCard richiesta dal Common Gui e gliela restituisce.
Action	Ricerca all'interno del database degli utenti il profilo corrispondente ad un utente specifico e lo restituisce al task richiedente.
Data	request_conv
Behavior	Il task riceve dal task <i>CommonGui.GetVCard</i> la richiesta per l'invio di un profilo utente; il task ricerca nel database il profilo utente desiderato e lo restituisce a <i>CommonGui.GetVCard</i> .

#### 4.1.3 Task: UsersManager

Description	Riceve ed esegue una richiesta di modifica, cancellazione o creazione di una nuova VCard.
Action	Crea, modifica o cancella i dati di un profilo utente dal database degli utenti.
Data	request_conv agent_to_request
Behavior	Il task riceve dal task <i>CommonGui.MembersManager</i> la richiesta per creare, modificare o cancellare un profilo utente; di conseguenza aggiorna il database con i dati ricevuti.

#### 4.2 Agente: Author Gui

L'agente che si interfaccia all'autore e si occupa di smistare tutte le sue richieste verso gli altri agenti.

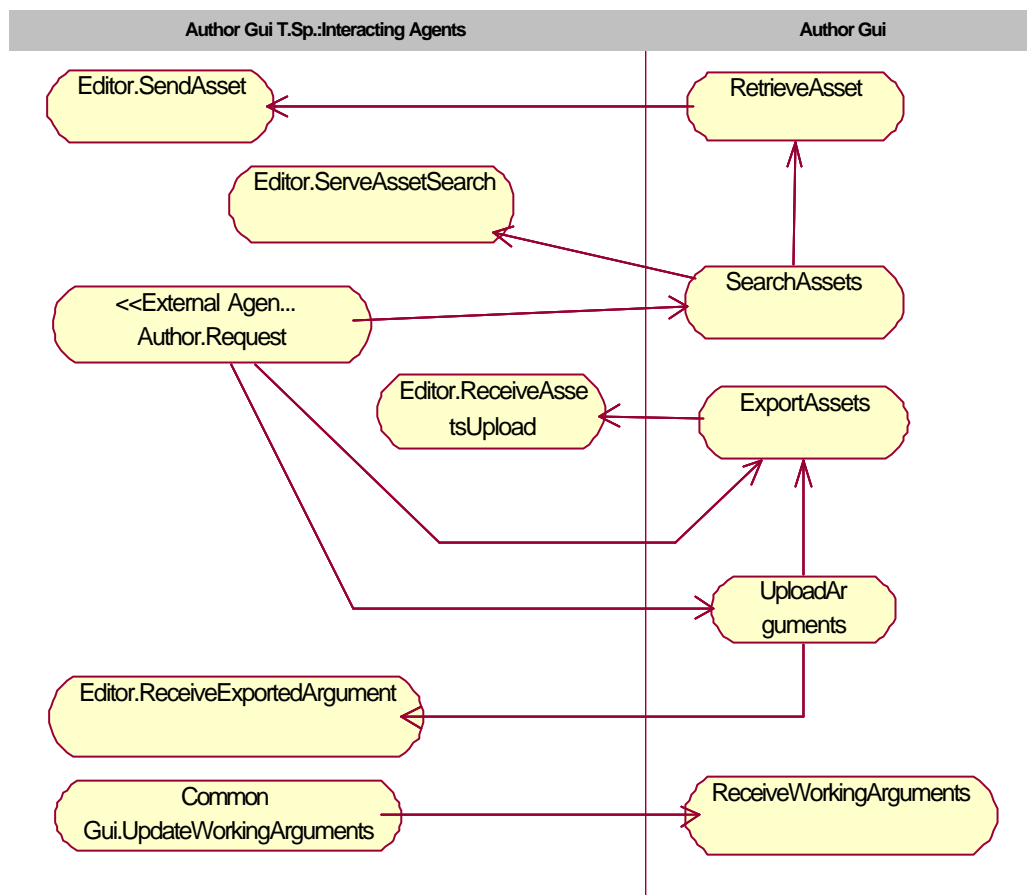


Figura 23: Diagramma dei task per l'agente Author Gui.

#### 4.2.1 Task: ReceiveWorkingArguments

Description	Riceve, memorizza e aggiorna l'interfaccia con gli argomenti che sono stati accettati dall'autore.
Action	Prende gli argomenti da sviluppare che sono stati accettati e i corsi relativi a tali argomenti.
Data	request_conv
Behavior	Il task riceve da <i>CommonGui.UpdateWorkingArguments</i> gli argomenti e i corsi relativi che sono stati accettati dall'autore, li memorizza e aggiorna l'interfaccia di sviluppo degli argomenti.

#### 4.2.2 Task: ExportAssets

Description	Esporta le informazioni atomiche relative ad un argomento.
Action	Invia i file e i metadati delle informazioni atomiche all'agente Editor.
Data	agent_to_request
Behavior	Il task deve esportare le informazioni atomiche relative ad un argomento completato dall'autore. Per far ciò invia i file delle informazioni atomiche e i metadati relativi all'agente Editor.

#### 4.2.3 Task: RetrieveAsset

Description	Richiede ed effettua il download di un'informazione atomica tra quelle condivise.
Action	Richiede all'agente Editor l'invio di un'informazione atomica e la inserisce nell'argomento che si sta sviluppando
Data	agent_to_request
Behavior	Il task richiede all'agente Editor l'invio di un'informazione atomica precedentemente ricercata all'interno di un archivio.

#### 4.2.4 Task: UploadArguments

Description	Esporta l'argomento completato con tutte le sue pagine e le relative informazioni sui metadati.
Action	Consegna l'argomento completato con le sue pagine web all'agente Editor.
Data	agent_to_request
Behavior	Il task esporta un argomento con le sue pagine web completato dall'autore e lo consegna all'agente Editor.

#### 4.2.5 Task: SearchAssets

Description	Richiede all'agente Editor di effettuare la ricerca di informazioni atomiche condivise.
-------------	---

Action	Effettua la ricerca di informazioni atomiche sulla base delle parole chiave fornite dall'utente.
Data	agent_to_request
Behavior	I task riceve dall'interfaccia utente le parole chiave necessarie per la ricerca di informazioni atomiche, le invia all'agente Editor che gli restituisce il risultato della ricerca.

### 4.3 Agente: Common Gui

Agente che riunisce funzionalità comuni offerte ad Author ed Editorial Central Staff interfacciandosi con essi e smistando le richieste agli altri agenti.

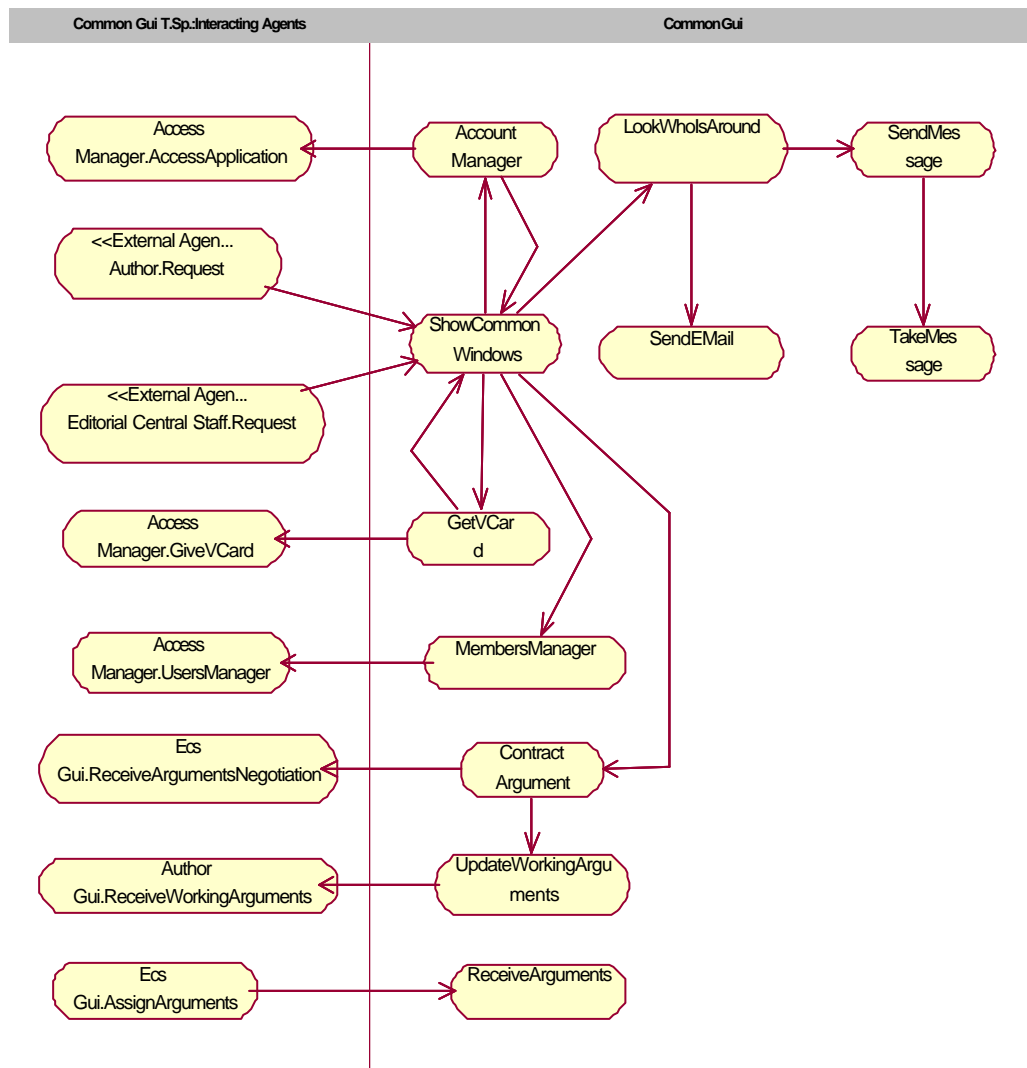


Figura 24: Diagramma dei task per l'agente Common Gui



### 4.3.1 Task: ShowCommonWindows

Description	Gestisce le finestre per la funzionalità comuni richieste dall'author o dall'ecs.
Action	Mostra le interfacce delle funzionalità richieste dall'utente.
Data	
Behavior	Il task gestisce le interfacce per le funzionalità comuni tra autore ed editorial central staff.

### 4.3.2 Task: AccountManager

Description	Riceve i dati di autenticazione immessi dall'utente e li invia all'Access Manager per verificarne la correttezza.
Action	Invia all'agente Access Manager i dati di autenticazione presenti nell'interfaccia di login dell'utente; permette o nega l'accesso alle funzionalità del sistema a seconda del risultato ricevuto.
Data	agent_to_request
Behavior	Il task prende dall'interfaccia di autenticazione i dati dell'utente necessari per accedere al sistema; invia tali dati all'agente Access Manager; riceve la risposta sulla correttezza dei dati di autenticazione e permette o nega l'accesso al sistema.

### 4.3.3 Task: LookWholsAround

Description	Comunica con l'AMS richiedendo quali autori siano online utilizzando il servizio di "White pages".
Action	Invia all'AMS una richiesta di servizio di pagine bianche; riceve informazioni sugli agenti della piattaforma.
Data	agent_to_request
Behavior	Il task comunica con l'AMS per sapere quali agenti sono presenti al

	momento nella piattaforma, in questo modo individua gli autori che al momento sono presenti all'interno del sistema, per potergli inviare successivamente un messaggio.
--	---

#### 4.3.4 Task: SendEmail

Description	Prende il testo del messaggio composto dall'utente e lo invia al server mediante il protocollo smtp.
Action	Prende il testo dell'email dall'interfaccia relativa, si connette al server di posta e tramite protocollo smtp invia l'email.
Data	
Behavior	Il task invia il messaggio di posta presente nell'interfaccia tramite protocollo smtp.

#### 4.3.5 Task: SendMessage

Description	Raccoglie il testo del messaggio dell'utente e lo invia all'istanza dell'agente Common Gui relativa al destinatario.
Action	Invia il messaggio all'istanza dell'agente Common Gui relativa all'autore desiderato.
Data	Message
Behavior	Il task prende dall'interfaccia il messaggio che deve essere inviato e lo manda all'agente Common Gui che appartiene all'autore cui deve essere inviato il messaggio.

#### 4.3.6 Task: TakeMessage

Description	Riceve i messaggi inviati e gestisce la relativa coda.
Action	Riceve i messaggi in arrivo che provengono da istanze diverse di agenti Common Gui.
Data	Message
Behavior	Il task attende per l'arrivo di messaggi da parte di altri utenti, li

	memorizza all'interno di una coda di messaggi e gestisce l'interfaccia dei messaggi aggiornandola con i messaggi ricevuti.
--	--

#### 4.3.7 Task: GetVCard

Description	Questo task riceve la richiesta di un utente di raccogliere i suoi dati personali e la inoltra all'Access Manager.
Action	Richiede all'Access Manager l'invio di un determinato profilo utente.
Data	agent_to_request
Behavior	Il task riceve dei dati di un utente e sulla base di tali dati richiede all'Access Manager l'invio del profilo utente relativo. Richiede l'aggiornamento dell'interfaccia del profilo utente.

#### 4.3.8 Task: MembersManager

Description	Invia all'Access Manager una richiesta di modifica o registrazione di una VCard.
Action	Invia una richiesta per la creazione, modifica o cancellazione di un profilo utente.
Data	agent_to_request
Behavior	Il task prende i dati relativi ad un utente e richiede all'Access Manager di creare, modificare o cancellare il profilo utente relativo.

#### 4.3.9 Task: ReceiveArguments

Description	Riceve gli argomenti e i relativi corsi di appartenenza assegnati dall'ecs ad un author.
Action	Prende gli argomenti in arrivo; aggiorna l'interfaccia relativa.
Data	request_conv agent_to_request
Behavior	Il task attende per la ricezione di argomenti assegnati dall'editorial

	central staff, li memorizza e aggiorna l'interfaccia per la contrattazione degli argomenti.
--	---

#### 4.3.10 Task: Contract Argument

Description	Invia all'agente Ecs Gui l'accettazione o il rifiuto espressi dall'autore per un argomento assegnatogli.
Action	Invia un messaggio con la risposta della contrattazione e il relativo argomento.
Data	agent_to_request
Behavior	Il task prende dall'interfaccia il commento della contrattazione relativa all'argomento e lo invia all'agente Ecs Gui con informazioni relative all'argomento accettato o rifiutato.

#### 4.3.11 Task: UpdateWorkingArguments

Description	Invia all'Author Gui gli argomenti accettati con tutte le informazioni che li accompagnano affinché questi possano essere sviluppati dall'autore.
Action	Invia all'agente Author Gui gli argomenti da sviluppare e i relativi corsi di appartenenza.
Data	agent_to_request
Behavior	Il task manda all'agente Author Gui sia gli argomenti che l'autore ha accettato sia i corsi di appartenenza per ogni argomento accettato.

### 4.4 Agente: Ecs Gui

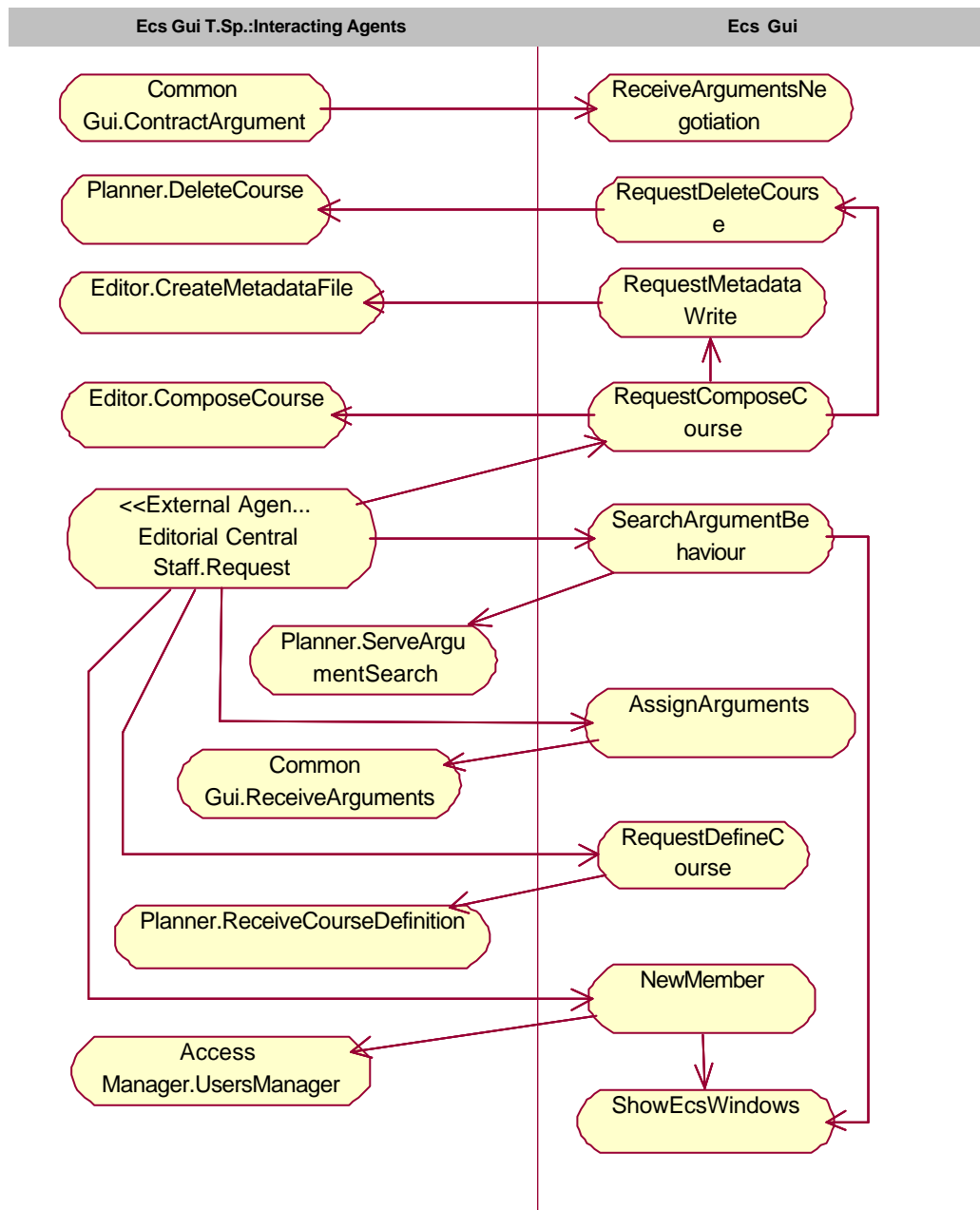


Figura 25: Diagramma dei task per l'agente Ecs Gui.

#### 4.4.1 Task: AssignArguments

Description	Questo task invia le definizioni degli argomenti all'istanza del common gui dell'autore cui l'Ecs li ha assegnati.
Action	Invia le definizioni degli argomenti e i corsi di appartenenza ai relativi autori.

Data	agent_to_request
Behavior	Il task consegna gli argomenti e i corsi relativi a tutti gli autori cui l'editorial central staff abbia assegnato almeno un argomento.

#### 4.4.2 Task: ReceiveArgumentsNegotiation

Description	Riceve l'accettazione o il rifiuto di un argomento da parte di un autore aggiornando le relative interfacce e memorizzando la risposta.
Action	Riceve la contrattazione degli argomenti e aggiorna l'interfaccia di contrattazione.
Data	request_conv agent_to_request
Behavior	Il task è in ascolto per la ricezione di contrattazione di argomenti; quando arrivano delle risposte dagli autori il task aggiorna lo stato degli argomenti in fase di contrattazione e aggiorna l'interfaccia.

#### 4.4.3 Task: RequestDefineCourse

Description	Riceve dall'interfaccia tutte le informazioni relative ad un corso definito dall'Ecs e le comunica al Planner.
Action	Invia all'agente Planner la definizione del corso presente nell'interfaccia di definizione dei corsi
Data	agent_to_request
Behavior	Il task prende dall'interfaccia la definizione del corso con tutti i suoi argomenti e manda tali informazioni all'agente Planner.

#### 4.4.4 Task: RequestDeleteCourse

Description	Questo task dopo l'assemblamento di un corso richiede al planner di eliminare le informazioni nell'archivio relative a questo corso.
Action	Richiede al Planner la cancellazione

	del corso selezionato.
Data	agent_to_request
Behavior	Il task una volta che il corso è stato assemblato, invia al Planner una richiesta per cancellare le informazioni relative al corso selezionato, dal momento che non sono più necessarie.

#### 4.4.5 Task: SearchArgumentBehaviour

Description	Invia al planner la richiesta di ricerca di argomenti condivisi affinché questi possano essere aggiunti ad un corso in fase di definizione.
Action	Richiede l'invio di informazioni su argomenti già sviluppati tramite l'utilizzo di parole chiave.
Data	agent_to_request
Behavior	Il task prende dall'interfaccia le parole chiave e richiede all'agente Planner di effettuare una ricerca di argomenti sulla base di queste parole chiave.

#### 4.4.6 Task: RequestMetadataWrite

Description	Richiede la trasformazione dei metadati inseriti dall'utente in un apposito formato di memorizzazione.
Action	Invia all'agente Editor i metadati del corso inseriti dall'utente per memorizzarli in un apposito formato.
Data	agent_to_request
Behavior	Il task prende dall'interfaccia i metadati del corso e li invia all'agente Editor.

#### 4.4.7 Task: RequestComposeCourse

Description	Richiede all'editor l'assemblamento di un corso i cui argomenti sono stati già sviluppati
Action	Invia all'Editor le informazioni

	necessarie del corso che deve essere assemblato insieme con l'ordinamento degli argomenti all'interno del corso.
Data	agent_to_request
Behavior	Il task riceve dall'utente la richiesta per assemblare un corso e inoltra la richiesta all'agente Editor.

#### 4.4.8 Task: NewMember

Description	Richiede l'inserimento dei dati per un nuovo utente e invia all'agente Access Manager i dati del profilo utente da creare.
Action	Invia i dati del profilo utente da inserire.
Data	agent_to_request
Behavior	Il task prende dall'interfaccia i dati dell'utente da inserire e li invia all'agente Access Manager richiedendo di inserire un nuovo utente.

#### 4.4.9 Task: ShowEcsWindows

Description	Gestisce le finestre dell'Ecs.
Action	Mostra e gestisce le interfacce per le funzionalità offerte all'editorial central staff.
Data	
Behavior	Il task riceve delle richieste per la visualizzazione delle interfacce dell'editorial central staff e di conseguenza gestisce la loro visualizzazione.

### 4.5 Agente: Editor

Agente che si occupa della produzione di contenuti didattici, della creazione dei metadati e dell'assemblamento dei corsi.



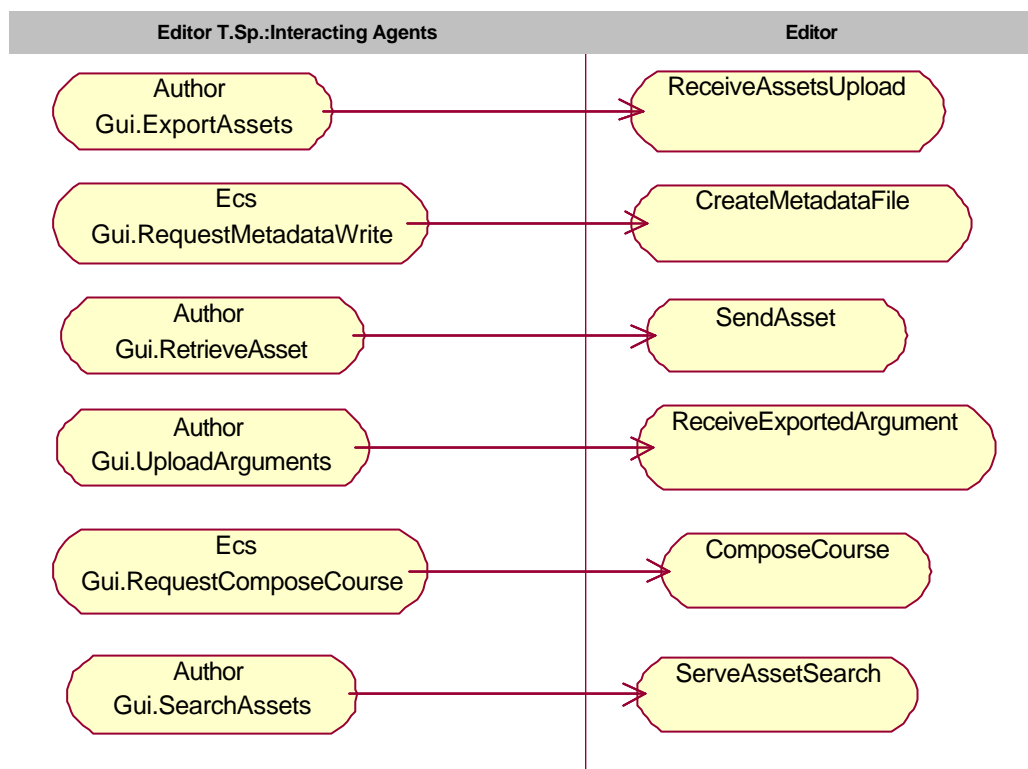


Figura 26: Diagramma dei task per l'agente Editor.

#### 4.5.1 Task: ReceiveAssetsUpload

Description	Riceve, completa e organizza le informazioni atomiche ricevute.
Action	Riceve le informazioni atomiche esportate, memorizza i metadati delle informazioni atomiche nel database.
Data	request_conv
Behavior	Il task attende per l'arrivo di informazioni atomiche; quando queste vengono esportate, vengono memorizzate in un archivio e vengono organizzati i metadati in un apposito formato.

#### 4.5.2 Task: CreateMetadataFile

Description	Il task riceve i metadati relativi a tutte le risorse (informazioni atomiche, argomenti e corsi), li traduce in un opportuno formato e li memorizza.
-------------	--

Action	Scrive i metadati in un apposito formato.
Data	request_conv
Behavior	Il task riceve i metadati descritti testualmente dall'utente e li memorizza in un apposito formato.

### 4.5.3 Task: SendAsset

Description	Una volta effettuata la ricerca o quando si vuole fare un anteprima di un asset questo task invia l'informazione atomica richiesta.
Action	Invia l'asset richiesto.
Data	request_conv
Behavior	Il task a seguito del risultato di una ricerca invia l'asset richiesto dal task <i>AuthorGui.RetrieveAsset</i> .

### 4.5.4 Task: ReceiveExportedArgument

Description	Riceve l'argomento esportato, lo completa e lo organizza in memoria.
Action	Riceve l'argomento esportato, ne memorizza i contenuti e i metadati.
Data	request_conv agent_to_request
Behavior	Il task attende per l'arrivo di argomenti che vengono esportati dagli autori, quando arriva un argomento memorizza i contenuti catalogando all'interno di un archivio e memorizza i metadati relativi in un apposito formato.

### 4.5.5 Task: ComposeCourse

Description	Riceve la richiesta di composizione di un corso i cui argomenti sono stati già sviluppati, lo completa, ne crea i metadati e lo organizza in memoria
Action	Assembla le risorse del corso e lo prepara per la pubblicazione.
Data	request_conv agent_to_request

Behavior	Il task riceve la richiesta di assemblamento di un corso i cui argomenti sono stati completati, memorizza i metadati del corso in un apposito formato, organizza i contenuti del corso secondo l'ordinamento deciso dall'utente e prepara il corso per la pubblicazione.
----------	--

#### 4.5.6 Task: ServeAssetSearch

Description	Esegue la ricerca di informazioni atomiche condivise e restituisce i risultati trovati.
Action	Ricerca sul database informazioni atomiche sulla base di parole chiave
Data	request_conv agent_to_request
Behavior	Il task riceve le richieste per ricercare informazioni atomiche ed effettua la ricerca nel database restituendo i risultati ottenuti.

### 4.6 Agente: Planner

Agente che si occupa della organizzazione dei corsi definiti dall'Ecs all'interno di un archivio.

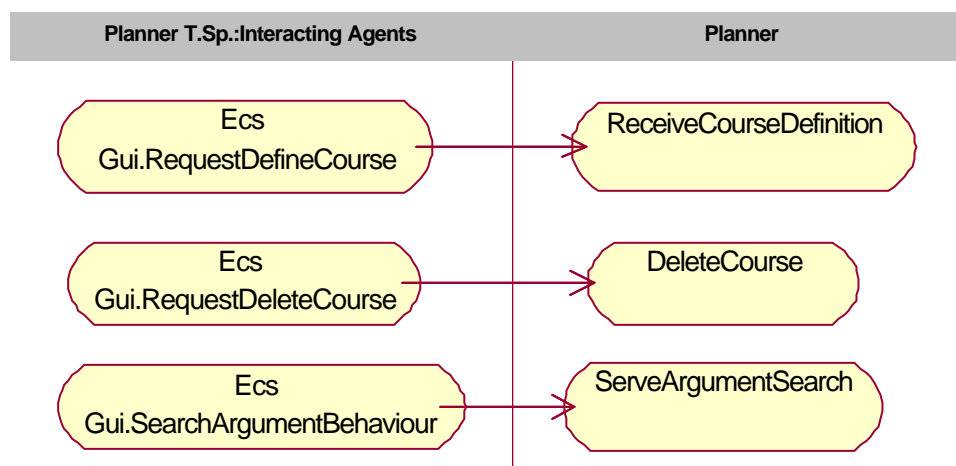


Figura 27: Diagramma dei task per l'agente Planner.

#### 4.6.1 Task: ReceiveCourseDefinition

Description	Riceve dall'agente Ecs Gui la definizione di corsi e la memorizza in un'apposita struttura dati.
Action	Memorizza all'interno di un archivio le definizioni del corso e degli argomenti.
Data	request_conv agent_to_request
Behavior	Il task riceve dall'agente Ecs Gui i dati relativi alla definizione di un corso e dei suoi argomenti, li memorizza in un archivio secondo un'opportuna gerarchia.

#### 4.6.2 Task: DeleteCourse

Description	Elimina i dati e le informazioni relative ad un corso che è stato appena completato.
Action	Cancella dall'archivio le informazioni e la struttura dati di un corso assemblato.
Data	request_conv
Behavior	Il task cancella le informazioni di definizione relative al corso assemblato.

#### 4.6.3 Task: ServeArgumentSearch

Description	Effettua la ricerca di argomenti condivisi quando devono essere aggiunti alla definizione di un nuovo corso.
Action	Ricerca argomenti secondo parole chiave e restituisce il risultato della ricerca.
Data	request_conv agent_to_request
Behavior	Il task riceve la richiesta per ricercare argomenti già sviluppati, effettua la ricerca nell'archivio e restituisce i risultati ottenuti.

### 5 Fase 5: Ontology Description

In questa fase sono presenti due diagrammi che descrivono rispettivamente il dominio dell'ontologia degli agenti e l'utilizzo delle ontologie per la comunicazione tra agenti. Per permettere una migliore visualizzazione i diagrammi sono stati divisi in più parti; va ricordato comunque che questa fase prevede un solo diagramma per il dominio dell'ontologia *Domain Ontology Description* (D.O.D.) e un diagramma per l'utilizzo delle ontologie nella comunicazione tra agenti *Communication Ontology Description* (C.O.D.). Per tutti i diagrammi che non risultano leggibili si consulti la documentazione allegata al presente lavoro.

## 5.1 Domain Ontology Description

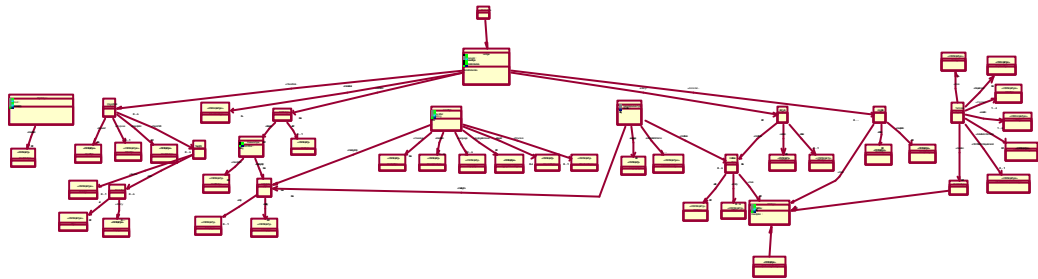
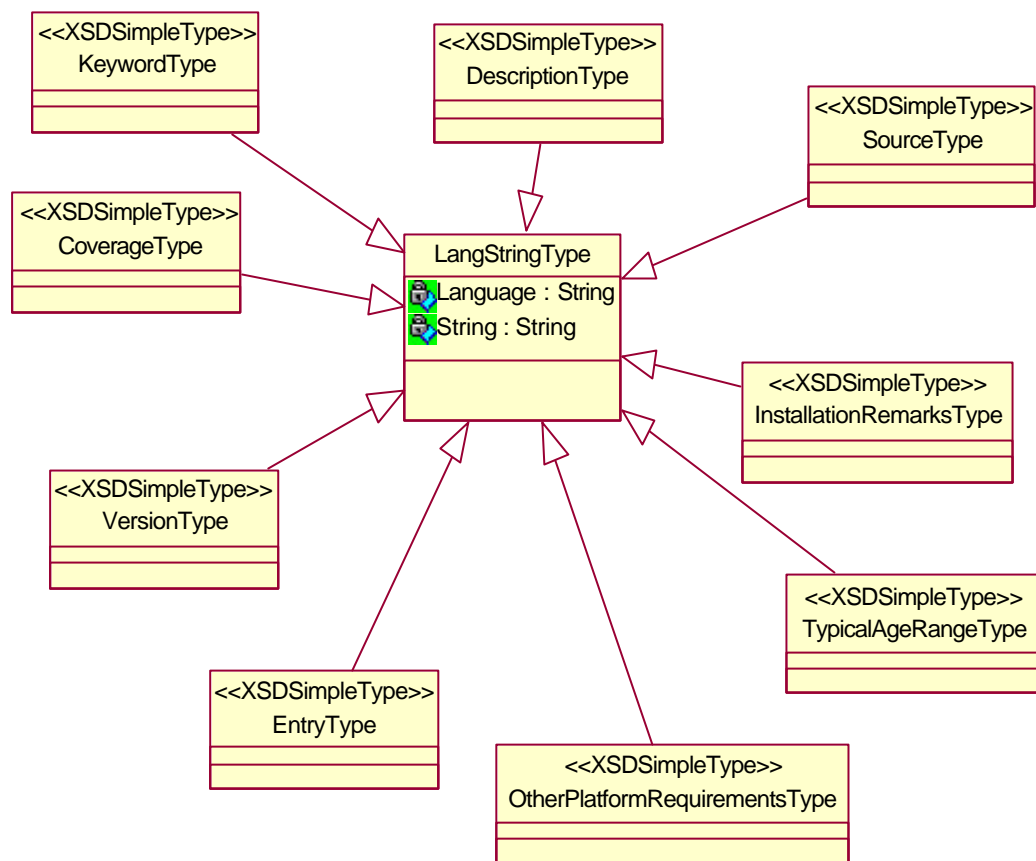
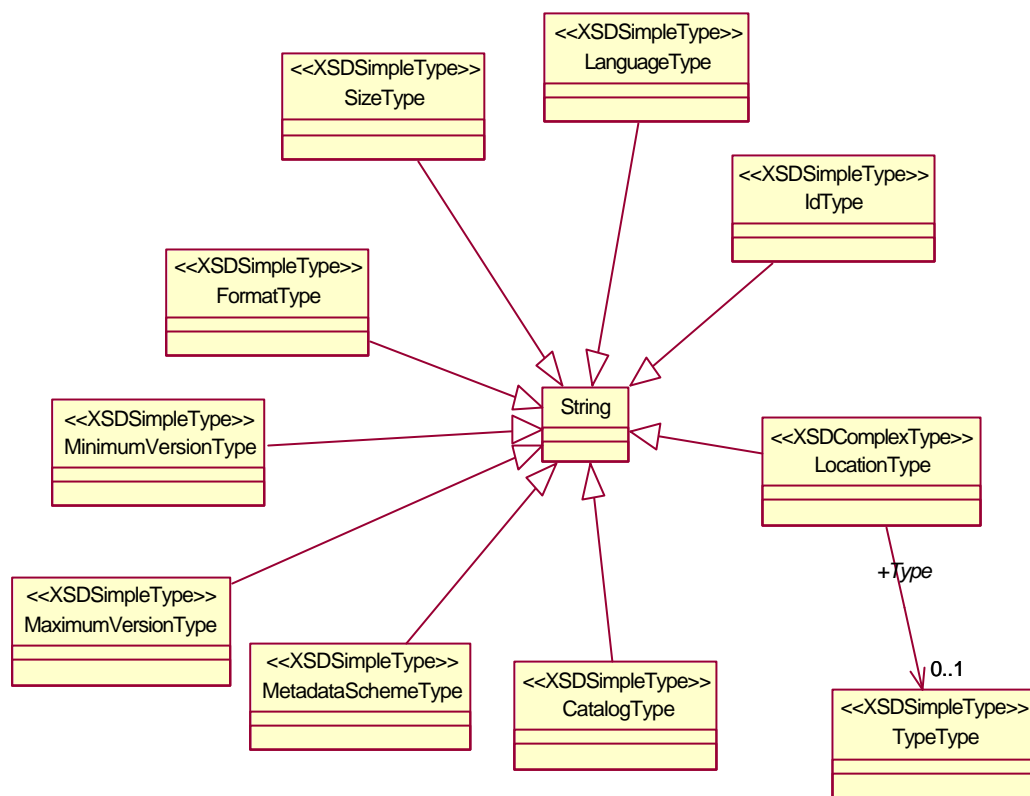


Figura 28: Domain Ontology Description (Parte 1/7).



**Figura 29: Domain Ontology Description (Parte 2/7).**



**Figura 30: Domain Ontology Description (Parte 3/7).**

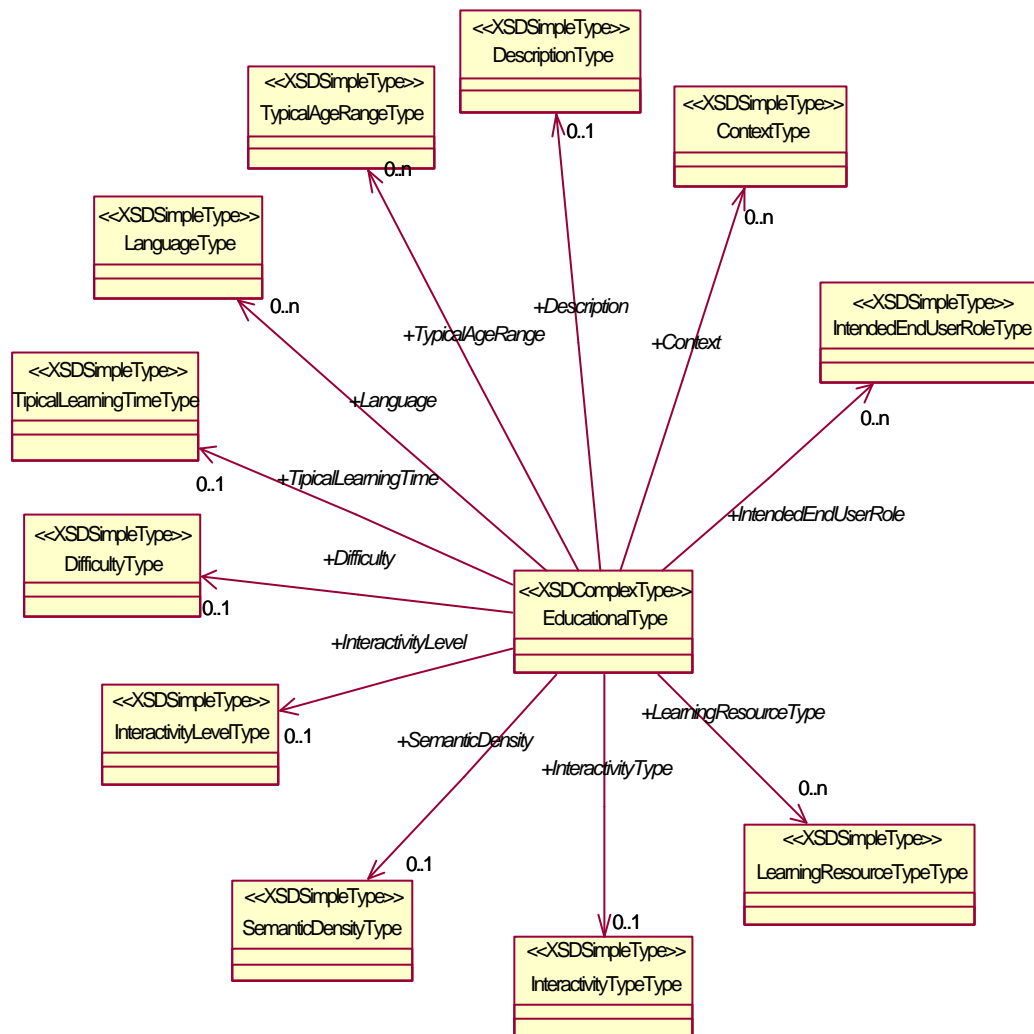
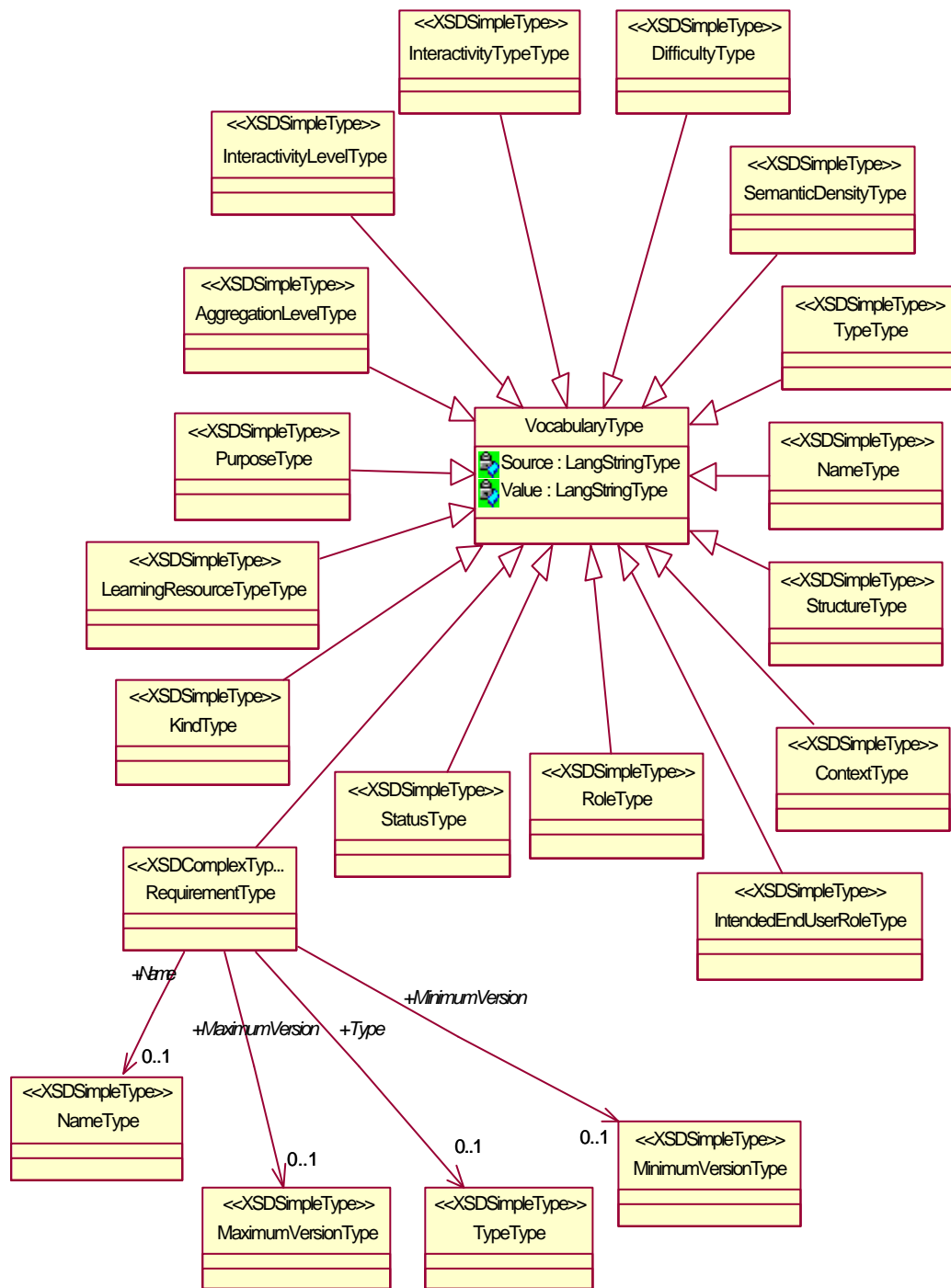


Figura 31: Domain Ontology Description (Parte 4/7).





**Figura 32: Domain Ontology Description (Parte 5/7).**

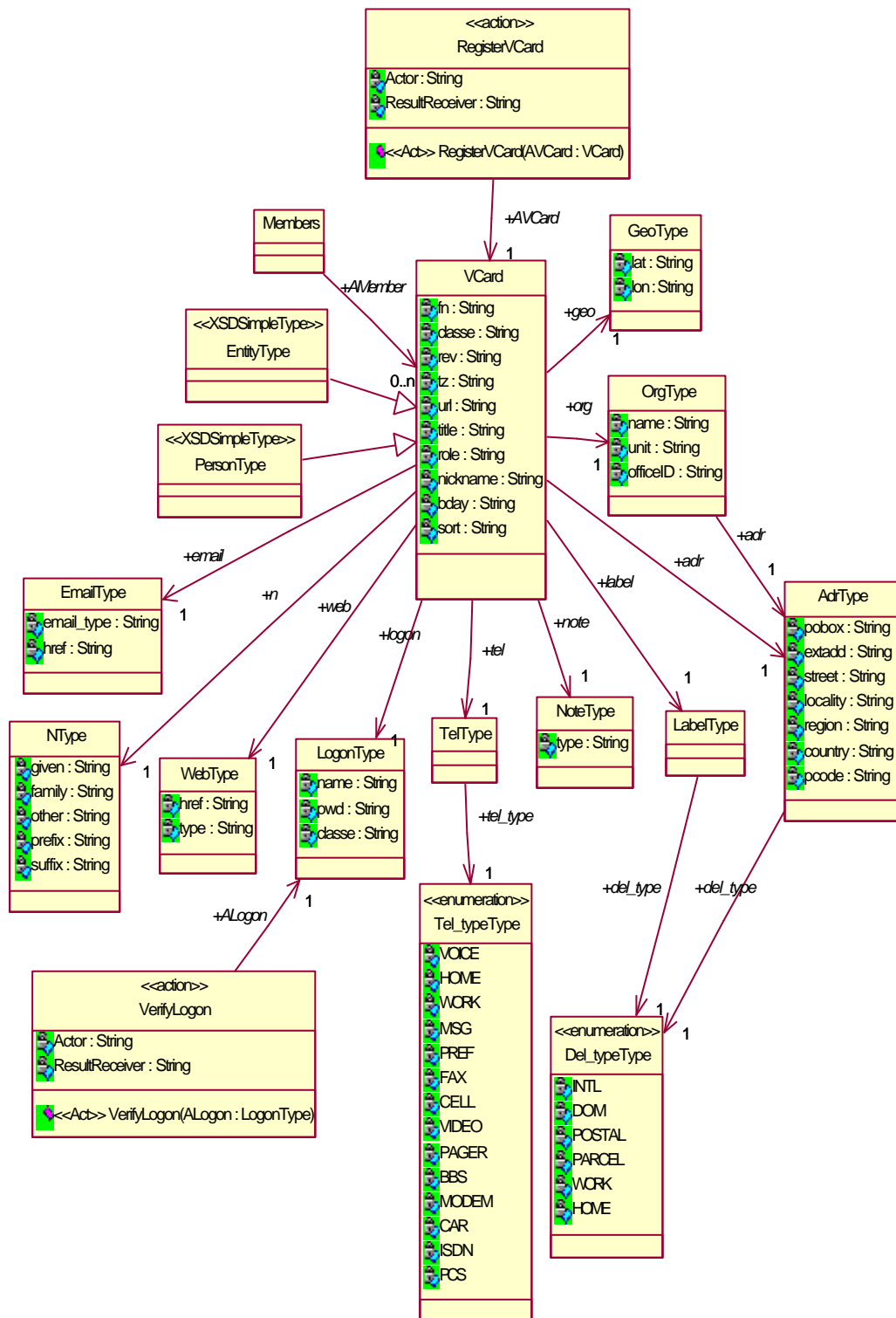


Figura 33: Domain Ontology Description (Parte 6/7).



			Item
InstallationRemarksType			SCORM Metadata Item
TypicalAgeRangeType			SCORM Metadata Item
OtherPlatformRequirementsType			SCORM Metadata Item
EntryType			SCORM Metadata Item
VersionType			SCORM Metadata Item
CoverageType			SCORM Metadata Item
KeywordType			SCORM Metadata Item
DescriptionType			SCORM Metadata Item
String			
SizeType			SCORM Metadata Item
FormatType			SCORM Metadata Item
MinimumVersionType			SCORM Metadata Item
MaximumVersionType			SCORM Metadata Item
MetadataSchemeType			SCORM Metadata Item
CatalogType			SCORM Metadata Item
LanguageType			SCORM Metadata Item
IdType			SCORM Metadata Item
TypeType			SCORM Metadata Item
LocationType			SCORM Metadata Item
VocabularyType	Source Value	LangStringType LangStringType	SCORM Metadata Item
InteractivityTypeType			SCORM Metadata Item
InteractivityLevelType			SCORM Metadata Item

AggregationLevelType			SCORM Metadata Item
PurposeType			SCORM Metadata Item
LearningResourceType			SCORM Metadata Item
KindType			SCORM Metadata Item
StatusType			SCORM Metadata Item
RoleType			SCORM Metadata Item
NameType			SCORM Metadata Item
SemanticDensityType			SCORM Metadata Item
IntendedEndUserRoleType			SCORM Metadata Item
ContextType			SCORM Metadata Item
StructureType			SCORM Metadata Item
DifficultyType			SCORM Metadata Item
RequirementType			SCORM Metadata Item
TypicalLearningTimeType			SCORM Metadata Item
EducationalType			SCORM Metadata Item
RightsType	Cost CopyrightAndOtherRestrictions	VocabularyType VocabularyType	SCORM Metadata Item
GeneralType	Identifier Title	String LangStringType	SCORM Metadata Item
TaxonType			SCORM Metadata Item
CatalogEntryType			SCORM Metadata Item
MetaMetadataType	Identifier	String	SCORM Metadata Item
EntityType			SCORM Metadata

			Item
TaxonPathType			SCORM Metadata Item
ResourceType	Identifier	String	SCORM Metadata Item
ContributeType			SCORM Metadata Item
PersonType			SCORM Metadata Item
DateType	DateTime Description	String LangStringType	SCORM Metadata Item
ClassificationType			SCORM Metadata Item
RelationType			SCORM Metadata Item
LifeCycleType			SCORM Metadata Item
AnnotationType			SCORM Metadata Item
LomType	General MetaMetadata Technical Rights	GeneralType MetaMetadata Type TechnicalType RightsType	SCORM Metadata Item
DurationType			SCORM Metadata Item
TechnicalType			SCORM Metadata Item
Metadata			SCORM Metadata Item
DefinitionPayload	Title Description Status Comment	String String Integer String	Contiene informazioni per i corsi, i sottocorsi e gli argomenti, tra cui il titolo della risorsa, una sua breve descrizione, lo stato di avanzamento e un commento utile per la contrattazione delle risorse da assegnare.
SearchAtomicInformation	Actor ResultReceiver	String String	Questa azione viene utilizzata per

	Keywords	String	richiedere la ricerca di informazioni atomiche già sviluppate sulla base di parole chiave.
SearchArgument	Actor ResultReceiver Keywords	String String String	Questa azione viene utilizzata per richiedere la ricerca di argomenti già sviluppati sulla base di parole chiave.
VerifyLogon	Actor ResultReceiver	String String	Questa azione viene utilizzata per richiedere la verifica dei dati di autenticazione dell'utente.
Tel_typeType	VOICE HOME WORK MSG PREF FAX CELL VIDEO PAGER BBS MODEM CAR ISDN PCS		VCard Item. Costanti che definiscono il tipo di numero telefonico presente nella VCard.
Del_typeType	INTL DOM POSTAL PARCEL WORK HOME		VCard Item. Costanti che individuano il tipo di indirizzo presente nella VCard.
EmailType	Email_type href	String String	VCard Item. Contiene l'email dell'utente.
NType	given family other prefix suffix	String String String String String	VCard Item. Contiene il numero telefonico dell'utente.

WebType	href type	String String	VCard Item. Contiene informazioni sul sito web dell'utente.
LogonType	name pwd classe	String String String	VCard Item. Contiene I dati di autenticazione dell'utente.
NoteType	type	String	VCard Item. Note informative supplementari sull'utente.
GeoType	Lat lon	String String	VCard Item. Informazioni sulla localizzazione dell'utente.
TelType			VCard Item.
OrgType	Name unit officeID	String String String	VCard Item. Informazioni sull'organizzazione cui appartiene l'utente.
LabelType			VCard Item.
AdrType	Pobox extadd street locality region country pcode	String String String String String String String	VCard Item. Indirizzo e localizzazione dell'utente.
RegisterVCard	Actor ResultReceiver	String String	Questa azione serve per richiedere la registrazione, modifica o cancellazione di un profilo utente nel sistema.
VCard	Fn classe rev tz url title role nickname bday	String String String String String String String String String	VCard Item. Contiene informazioni di carattere generale sull'utente, come nome, soprannome, data di nascita, ruolo sociale e altri.



	sort	String	
Members			Un'insieme di utenti.
Courses			Un'insieme di corsi.
AssembleCourse	Actor ResultReceiver	String String	Questa azione viene utilizzata per richiedere l'assemblamento di un corso i cui argomenti sono stati sviluppati.
DefineCourse	Actor ResultReceiver	String String	Questa azione viene utilizzata per richiedere l'organizzazione all'interno di un archivio di un corso definito dall'editorial central staff.
IsCourseStatus	Value	Integer	Questo predicato da informazioni sullo stato di avanzamento del corso.
AssignArgument	Actor ResultReceiver Author	String String String	Questa azione è utilizzata per assegnare gli argomenti ai vari autori.
AnswerAssignment	Actor ResultReceiver Author	String String String	Azione utilizzata per la contrattazione degli argomenti in fase di assegnazione.
Course	myMetadata myDefinitionPayload	Metadata DefinitionPayload	Rappresenta un corso caratterizzato da delle informazioni descrittive e dai metadati; all'interno del corso sono presenti sottocorsi e argomenti.
File	Path	String	
Arguments			Un'insieme di argomenti.

ExportArgument	Actor ResultReceiver	String String	Questa azione è utilizzata per esportare gli argomenti completati dagli autori.
IsArgumentStatus	Value	Integer	Predicato che da informazioni sullo stato di avanzamento dell'argomento.
Argument	myMetadata myDefinitionPayload	Metadata DefinitionPayload	Un argomento contiene delle informazioni descrittive e dei metadati. Un argomento è composto da files e informazioni atomiche.
AtomicInformations			Un'insieme di informazioni atomiche.
AtomicInformation	MyFile myMetadata	File Metadata	Un'informazione atomica è composta da un file e dai metadati che lo descrivono.
SendMeAtomicInformation	Actor ResultReceiver AssetName	String String String	Questa azione è utilizzata per richiedere l'invio di una data informazione atomica già sviluppata selezionata a seguito di una ricerca.
UploadAtomicInformations	Actor ResultReceiver fileNames ftpHost remoteDir	String String String[] String String	Azione utilizzata per l'esportazione di informazioni atomiche da parte dell'autore.
BuildMetadata	Actor ResultReceiver myMetadata	String String Metadata	Azione che serve per richiedere la memorizzazione dei metadati in un

		opportuno formato.
--	--	--------------------

## 5.2 Communication Ontology Description

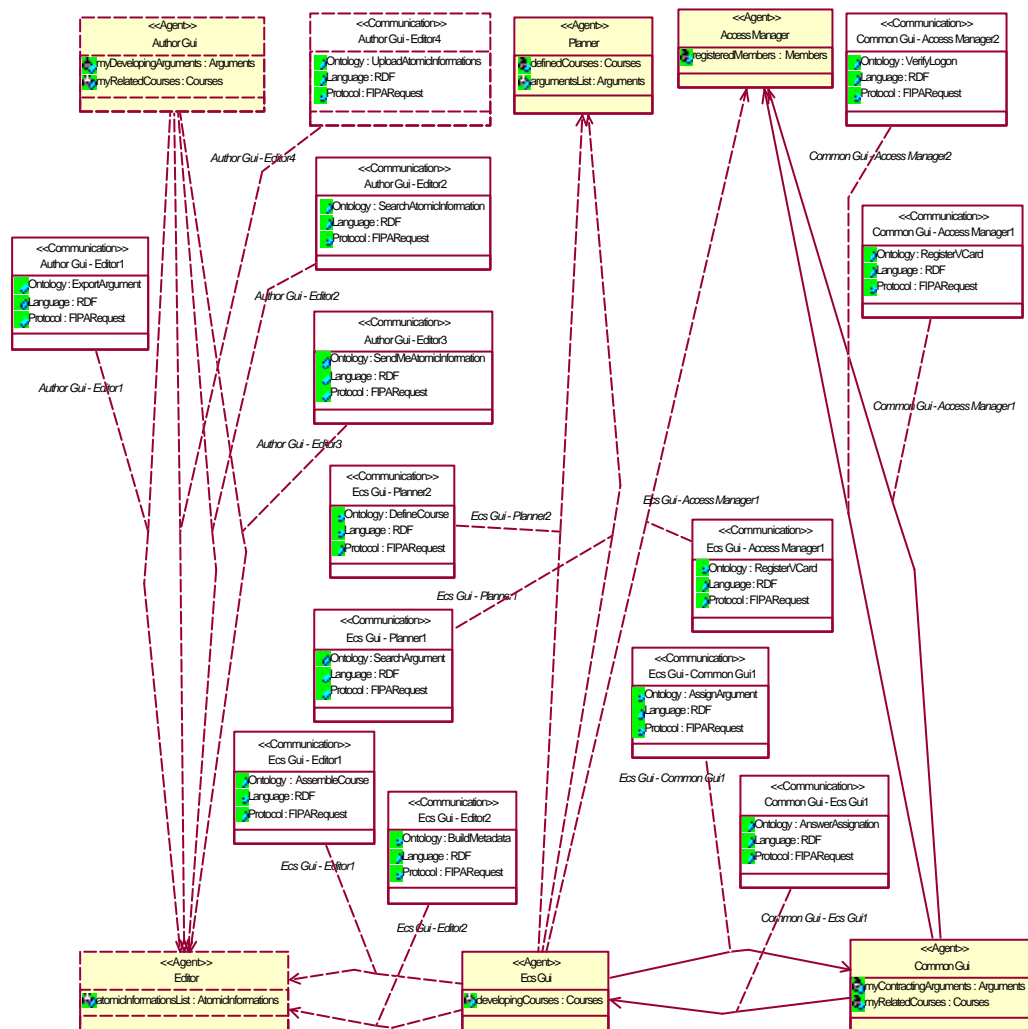


Figura 35: Communication Ontology Description

### 5.2.1 Conoscenza degli agenti

Agent	Field Name	Data Type
Common Gui	myContractingArguments myRelatedCourses	Arguments Courses

Access Manager	registeredMembers	Members
Planner	definedCourses argumentsList	Courses Arguments
Ecs Gui	developingCourses	Courses
Editor	atomicInformationsList	AtomicInformations
Author Gui	myDevelopingArguments myRelatedCourses	Arguments Courses

## 5.2.2 Contenuto dei messaggi

Name	MsgID	From Agent	To Agent	Language	Protocol	Data
Common Gui - Access Manager1	3	Common Gui	Access Manager	RDF	FIPARquest	RegisterV Card
Common Gui - Access Manager2	4	Common Gui	Access Manager	RDF	FIPARquest	VerifyLog on
Common Gui - Ecs Gui1	5	Common Gui	Ecs Gui	RDF	FIPARquest	AnswerAss ignment
Ecs Gui - Common Gui1	6	Ecs Gui	Common Gui	RDF	FIPARquest	AssignArg ument
Ecs Gui - Access Manager1	7	Ecs Gui	Access Manager	RDF	FIPARquest	RegisterV Card
Ecs Gui - Planner1	9	Ecs Gui	Planner	RDF	FIPARquest	SearchArg ument
Ecs Gui - Planner2	10	Ecs Gui	Planner	RDF	FIPARquest	DefineCou rse
Ecs Gui - Editor1	8	Ecs Gui	Editor	RDF	FIPARquest	AssembleC ourse
Ecs Gui - Editor2	0	Ecs Gui	Editor	RDF	FIPARquest	BuildMeta data
Author Gui - Editor1	1	Author Gui	Editor	RDF	FIPARquest	ExportArg ument

Author Gui - Editor2	2	Author Gui	Editor	RDF	FIPARquest	SearchAtomicInformation
Author Gui - Editor3	11	Author Gui	Editor	RDF	FIPARquest	SendMeAtomicInformation
Author Gui - Editor4	12	Author Gui	Editor	RDF	FIPARquest	UploadAtomicInformations

## 6 Fase 6: Role Description

Il diagramma descrive il ciclo di vita di ogni agente, focalizzando l'attenzione nei ruoli che questo gioca, nelle collaborazioni che necessita e nelle comunicazioni in cui partecipa. In questo diagramma vengono evidenziati anche i cambiamenti di ruolo all'interno di uno stesso agente e la condizione di *trigger* che scatuisce il cambiamento di ruolo.



## 6.1 Comunicazione

From	To	Name	Refer to C.O.D. msg (MsgID)
Common Gui:Logger	Access Manager:Authentifier	Common Gui - Access Manager2	4
Common Gui:Modify Member Gui	Access Manager:Members Administrator	Common Gui - Access Manager1	3
Ecs Gui:New Member Gui	Access Manager:Members Administrator	Ecs Gui - Access Manager1	7
Author Gui:Atomic Information Production Gui	Editor:Atomic Information Manager	Author Gui - Editor2	2
Author Gui:Atomic Information Production Gui	Editor:Atomic Information Manager	Author Gui - Editor3	11
Ecs Gui:Definition Gui	Planner:Course Definer	Ecs Gui - Planner2	10
Author Gui:Arguments Production Gui	Editor:Learning Resources Manager	Author Gui - Editor1	1
Author Gui:Arguments Production Gui	Editor:Learning Resources Manager	Author Gui - Editor4	12
Ecs Gui:Course Management Gui	Planner:Course Definer	Ecs Gui - Planner1	9
Ecs Gui:Course Management Gui	Editor:Learning Resources Manager	Ecs Gui - Editor1	8
Ecs Gui:Course Management Gui	Editor:Learning Resources Manager	Ecs Gui - Editor2	0
Common Gui:Arguments Trader	Ecs Gui:Course Management Gui	Common Gui - Ecs Gui1	5
Ecs Gui:Course Management Gui	Common Gui:Arguments Trader	Ecs Gui - Common Gui1	6

## 6.2 Dipendenze

Dependent/Client	Dependee/server	Kind of dependency	Description	Attached Note
Arguments Production Gui	Atomic Information Production Gui	[ROLE CHANGE]	Cambiamento di ruolo all'interno dell'agente Author Gui	
Arguments	Atomic	TRIGGER:	La condizione di	La

Production Gui	Information Production Gui	add atomic information	trigger che scaturisce il cambiamento di ruolo è l'aggiunta di un'informazione atomica all'argomento che si sta sviluppando.	condizione si riferisce all'inserimento di un'informazione atomica già sviluppata o di una nuova informazione atomica.
Arguments Production Gui	Learning Resources Manager	(service)	L'agente Author Gui esporta l'argomento che viene ricevuto e completato dall'agente Editor.	L'agente Editor completa l'argomento memorizzando i metadati in un apposito formato e organizzando i contenuti in un archivio.
Atomic Information Production Gui	Atomic Information Manager	(resource)	L'agente Author Gui richiede all'editor un'informazione atomica.	
Logger	Authentifier	(resource)	I dati di autenticazione inseriti dall'utente devono essere confrontati con quelli presenti in archivio.	
Modify Member Gui	Members Administrator	(service)	L'inserimento dei dati del profilo di un utente deve essere memorizzato dall'agente Access Manager all'interno di un archivio.	
Arguments Trader	Course Management Gui	(service)	Durante la contrattazione	



			degli argomenti questi ruoli scambiano i messaggi di contrattazione tra ecs e autore.	
Definition Gui	Course Definer	(service)	I dati di un corso appena definito dall'utente devono essere opportunamente organizzati e memorizzati in un archivio dall'agente Planner.	
New Member Gui	Members Administrator	(service)	Quando si registra un nuovo utente i dati che vengono passati dall'agente Ecs Gui all'agente Access Manager devono essere registrati da quest'ultimo all'interno di un archivio.	
Course Management Gui	Learning Resources Manager	(service)	La richiesta di assemblamento di un corso dall'agente Ecs Gui all'agente Editor richiede che quest'ultimo esegua una serie di operazioni per assemblare il corso, memorizzarne i metadati e prepararlo alla pubblicazione.	
Course Management Gui	Arguments Trader	(service)	Durante la contrattazione degli argomenti questi ruoli	

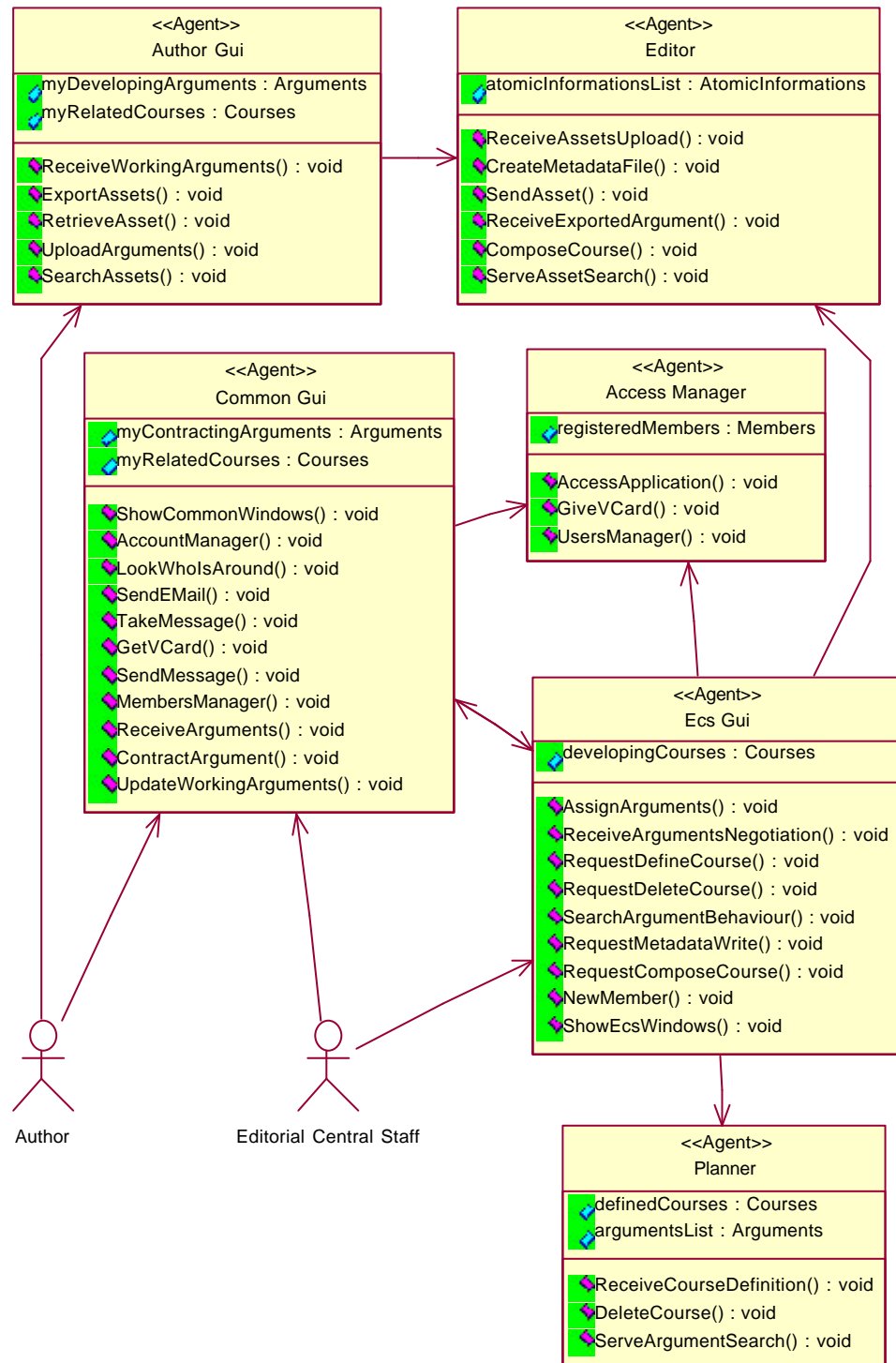
			scambiano i messaggi di contrattazione tra ecs e autore.	
Course Management Gui	Course Definer	(service)	L'agente EcsGui richiede all'agente Planner di ricercare argomenti già sviluppati sulla base di parole chiave e dopo aver effettuato la ricerca l'agente Planner invia all'agente Ecs Gui il risultato della ricerca.	

## 7 Fase 7: Protocol Description

In questa fase vanno illustrati i protocolli utilizzati per la comunicazione tra gli agenti. Poiché in questo progetto sono stati utilizzati solo protocolli dello standard FIPA non verranno ulteriormente descritti. Per inserire nuovi protocolli ad ogni modo si procede seguendo la notazione dei diagrammi di sequenza secondo l'estensione AUML del linguaggio UML.

## 8 Fase 8: Multi-Agent Structure Definition

In questo diagramma viene mostrato l'interno sistema multi agente e gli attori che interagiscono con il sistema. Ogni agente è rappresentato da una classe e all'interno le operazioni della classe rappresentano i task dell'agente. Ogni associazione può rappresentare sia una comunicazione sia una dipendenza.



**Figura 37: Multi-Agent Structure Definition**

## **9 Fase 9: Multi-Agent Behaviour Description**

Nel diagramma Multi-agent Behaviour Definition viene rappresentato il flusso degli eventi, i metodi invocati e i messaggi che si scambiano gli agenti. Tale diagramma non viene riportato di seguito per motivi di visualizzazione, ci si riferisca dunque alla documentazione del progetto allegata a questo lavoro per consultare tale diagramma.

### **9.1 Fase Single Agent Structure Definition**

#### **9.1.1 Agente: Author Gui**

L'agente che si interfaccia all'autore e si occupa di smistare tutte le sue richieste verso gli altri agenti.

In particolare si occupa di:

- Gestire l'arrivo di nuove assegnazioni
- Gestire la comunicazione con gli altri agenti per l'esportazione di argomenti completati
- Gestire l'esportazione delle risorse multimediali condivisibili
- Gestire l'interfaccia di ricerca di risorse multimediali condivise
- Gestire la comunicazione con gli altri agenti per il download e l'utilizzo di risorse multimediali condivise

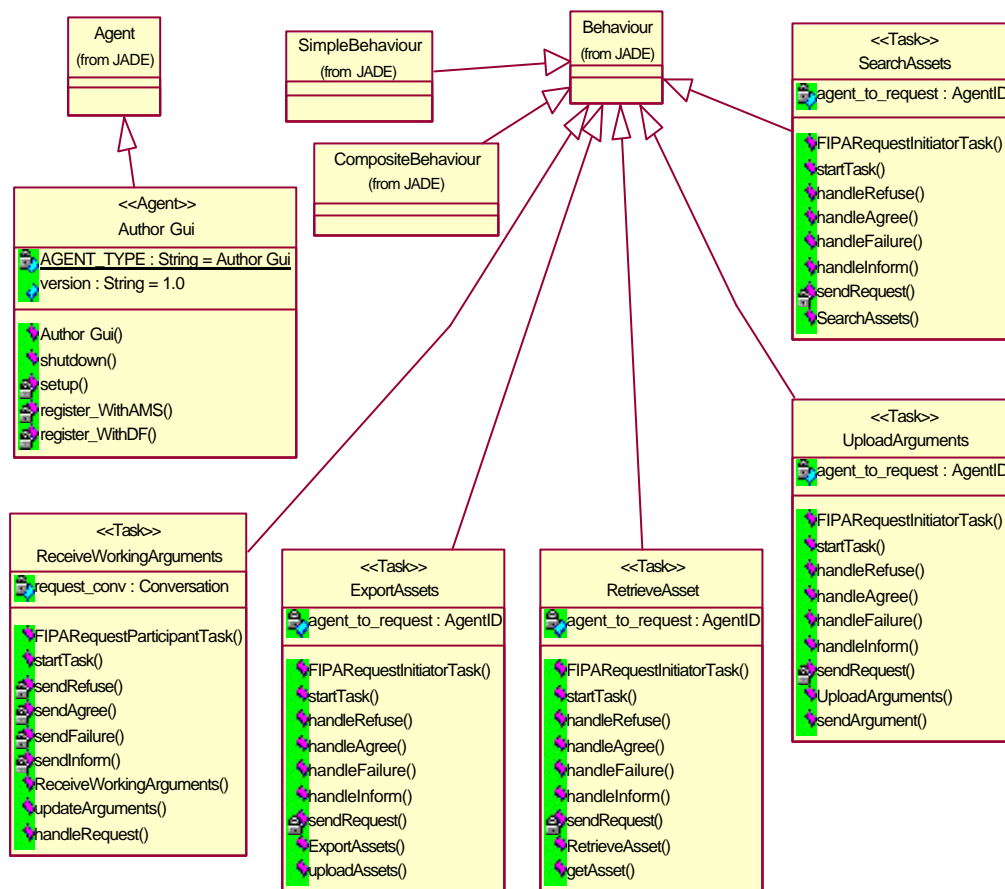


Fig.38: Single Agent Structure Definition - Author Gui

### 9.1.1.1 Attributi

Class	Attribute	Data Type	Description
Author Gui	AGENT_TYPE version	String String	
Simple Behaviour CompositeBehaviour ReceiveWorkingArguments	request_conv	Conversation	Behaviour per la ricezione degli argomenti da sviluppare.
ExportAssets	agent_to_request	AgentID	Behaviour per l'esportazione di informazioni

			atomiche.
RetrieveAsset	agent_to_request	AgentID	Behaviour per la richiesta di un'informazione atomica già sviluppata.
UploadArguments	agent_to_request	AgentID	Behaviour per l'esportazione di argomenti.
SearchAssets	agent_to_request	AgentID	Behaviour per la ricerca di informazioni atomiche già sviluppate.

### 9.1.1.2 Metodi

Class	Method	Parameters	Description
Author Gui	Author Gui shutdown setup register_WithAMS register_WithDF	platform name ownership	Author Gui è il costruttore dell'agente. Shutdown è il metodo che viene richiamato quando viene terminata l'esecuzione dell'agente. Setup è il metodo che viene richiamato quando viene inizializzato l'agente. Register_withAMS e register_withDF sono dei metodi che registrano gli indirizzi e i

SimpleBehaviour  
CompositeBehaviour  
ReceiveWorkingArguments

FIPAResponseParticipantTask  
startTask  
sendRefuse  
sendAgree  
sendFailure  
sendInform  
ReceiveWorkingArguments  
updateArguments  
handleRequest

conv  
data  
data  
data  
data

servizi  
dell'agente. Gli  
agenti AMS e  
DF forniscono  
servizi di  
pagine bianche  
e pagine gialle.

ReceiveWorkingArguments è il costruttore del behaviour. I metodi sendxxx servono per l'invio di messaggi in accordo con il protocollo FIPAResponse. Il metodo handleRequest serve per gestire il messaggio in arrivo di richiesta del protocollo FIPAResponse. Il metodo updateArguments riceve gli argomenti accettati e i corsi relativi. ExportAssets è il costruttore del behaviour. I metodi handlexxx servono per gestire l'arrivo di messaggi in accordo al protocollo FIPAResponse. Il metodo sendRequest

ExportAssets

FIPAResponseInitiatorTask  
ask  
startTask  
handleRefuse  
handleAgree  
handleFailure  
handleInform  
sendRequest  
ExportAssets  
uploadAssets

id  
conv  
conv  
conv  
conv  
data

			invia il messaggio di richiesta in accordo a protocollo FIPAResult. Il metodo uploadAssets esporta le informazioni atomiche relative ad un argomento.
RetrieveAsset	FIPAResultInitiatorT	id	RetrieveAsset è
	ask	conv	il costruttore del
	startTask	conv	behaviour. I
	handleRefuse	conv	metodi
	handleAgree	conv	handlexxxx
	handleFailure	data	servono per
	handleInform		gestire l'arrivo
	sendRequest		di messaggi in
	RetrieveAsset		accordo al
	getAsset		protocollo FIPAResult. Il metodo sendRequest invia il messaggio di richiesta in accordo a protocollo FIPAResult. Il metodo getAsset prende l'informazione atomica richiesta e la aggiunge all'argomento che si sta sviluppando.
UploadArguments	FIPAResultInitiatorT	id	UploadArgume
	ask	conv	nts è il
	startTask	conv	costruttore del
	handleRefuse	conv	behaviour. I
	handleAgree	conv	metodi
	handleFailure	data	handlexxxx
	handleInform		servono per



	sendRequest UploadArguments sendArgument		gestire l'arrivo di messaggi in accordo al protocollo FIPAResult. Il metodo sendRequest invia il messaggio di richiesta in accordo a protocollo FIPAResult. Il metodo sendArgument esporta l'argomento con i contenute i metadati.
SearchAssets	FIPAResultInitiatorT ask startTask handleRefuse handleAgree handleFailure handleInform sendRequest SearchAssets	id conv conv conv conv data	SearchAssets è il costruttore del behaviour. I metodi handlexxx servono per gestire l'arrivo di messaggi in accordo al protocollo FIPAResult. Il metodo sendRequest invia il messaggio di richiesta in accordo a protocollo FIPAResult, contenente l parole chiave sulla base delle quali si desidera effettuare la ricerca di informazioni atomiche.

## 9.1.2 Agente: Common Gui

Agente che riunisce funzionalità comuni offerte ad Author ed Editorial Central Staff interfacciandosi con essi e smistando le richieste agli altri agenti.

In particolare questo agente si occupa di:

- Offrire all'utente la possibilità di modificare i propri dati (autore) o quelli di un qualsiasi altro utente del sistema (ECS)
- Gestire l'interfaccia di login
- Scambiare messaggi (sia instant messaging che emails)
- Gestire la coda dei messaggi ricevuti
- Gestire la contrattazione degli argomenti da sviluppare

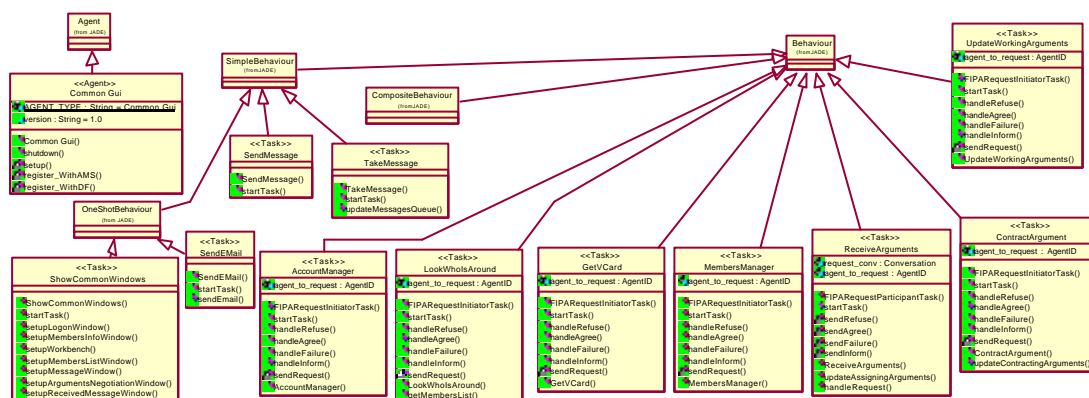


Fig. 39: Single Agent Structure Definition - Common Gui

### 9.1.2.1 Attributi

Class	Attribute	Data Type	Description
Common Gui	AGENT_TYPE version	String String	

Simple Behaviour  
CompositeBehaviour  
OneShotBehaviour  
ShowCommonWindo  
ws

Behaviour per  
la gestione delle  
interfacce che  
offrono  
funzionalità  
comuni ad ecs e

AccountManager	agent_to_request	AgentID	autori. Behaviour per richiedere la verifica dei dati di autenticazione dell'utente.
LookWhoIsAround	agent_to_request	AgentID	Behaviour per richiedere all'AMS informazioni sugli agenti presenti nella piattaforma.
SendEMail			Behaviour per l'invio di email.
SendMessage			Behaviour per l'invio di messaggi.
TakeMessage			Behaviour per la ricezione dei messaggi e per la gestione della coda dei messaggi ricevuti.
GetVCard	agent_to_request	AgentID	Behaviour per la richiesta di un profilo utente.
MembersManager	agent_to_request	AgentID	Behaviour per la modifica dei dati di un profilo utente.
ReceiveArguments	request_conv agent_to_request	Conversation AgentID	Behaviour per la ricezione di argomenti in fase di contrattazione.
ContractArgument	agent_to_request	AgentID	Behaviour per la contrattazione di argomenti assegnati.
UpdateWorkingArguments	agent_to_request	AgentID	Behaviour per l'invio all'agente AuthorGui degli

argomenti  
accettati che si  
vogliono  
sviluppare

### 9.1.2.2 Metodi

Class	Method	Parameters	Description
Common Gui	Common Gui shutdown setup register_WithAMS register_WithDF	platform name ownership	Common Gui è il costruttore dell'agente. Shutdown è il metodo che viene richiamato quando viene terminata l'esecuzione dell'agente. Setup è il metodo che viene richiamato quando viene inizializzato l'agente. Register_withA MS e register_withDF sono dei metodi che registrano gli indirizzi e i servizi dell'agente. Gli agenti AMS e DF forniscono servizi di pagine bianche e pagine gialle.
SimpleBehaviour CompositeBehaviour OneShotBehaviour ShowCommonWindows	ShowCommonWindows startTask setupLogonWindow setupMembersInfoWin		ShowCommon Windows è il costruttore del behaviour. Il metodo

dow  
setupWorkbench  
setupMembersListWin  
dow  
setupMessageWindow  
setupArgumentsNegoti  
ationWindow  
setupReceivedMessag  
eWindow

setupLogonWin  
dow gestisce  
l'interfaccia di  
autenticazione.  
Il metodo  
setupMembersI  
nfoWindow  
gestisce  
l'interfaccia per  
il profilo di un  
utente. Il  
metodo  
stupWorkbench  
avvia  
l'interfaccia di  
default del  
sistema in cui  
sono presenti le  
funzionalità  
messe a  
disposizione  
dell'utente. Il  
metodo  
setupMembersL  
istWindow  
gestisce  
l'interfaccia per  
la  
visualizzazione  
della lista degli  
utenti. I metodi  
setupMessageW  
indow e  
setupReceived  
MessageWindo  
w gestiscono le  
interfacce per  
l'invio e la  
ricezione di  
messaggi e per  
l'invio di email.  
Il metodo setup  
ArgumentsNeg  
otiationWindow  
gestisce la  
finestra di  
contrattazione  
degli argomenti.

AccountManager	FIPAResponseInitiatorT	id	AccountManager
	ask	conv	er è il
	startTask	conv	costruttore del
	handleRefuse	conv	behaviour. I
	handleAgree	conv	metodi
	handleFailure	data	handlexxx
	handleInform		servono per
	sendRequest		gestire l'arrivo
	AccountManager		di messaggi in
			accordo al
			protocollo
			FIPAResponse. Il
			metodo
			sendRequest
			invia il
			messaggio di
			richiesta in
			accordo a
			protocollo
			FIPAResponse,
			contenente i
			dati di
			autenticazione
			dell'utente che
			si intendono
			verificare.
LookWhoIsAround	FIPAResponseInitiatorT	id	LookWhoIsAro
	ask	conv	und è il
	startTask	conv	costruttore del
	handleRefuse	conv	behaviour. I
	handleAgree	conv	metodi
	handleFailure	data	handlexxx
	handleInform		servono per
	sendRequest		gestire l'arrivo
	LookWhoIsAround		di messaggi in
	getMembersList		accordo al
			protocollo
			FIPAResponse. Il
			metodo
			sendRequest
			invia il
			messaggio di
			richiesta
			all'AMS in
			accordo a
			protocollo
			FIPAResponse. Il
			metodo

			getMembersList prende le informazioni sulle istanze degli agenti Common Gui presenti nella piattaforma. SendEmail è il costruttore del behaviour. Il metodo sendEmail invia l'email mediante protocollo smtp.
SendEmail	SendEmail startTask sendEmail		SendMessage è il costruttore del behaviour. In questo behaviour viene inviato un ACL message in cui il contenuto è il testo del messaggio che si desidera inviare.
SendMessage	SendMessage startTask		TakeMessage è il costruttore del behaviour. Il behaviour rimane in attesa di messaggi in arrivo e il metodo updateMessages Queue gestisce la coda dei messaggi in arrivo.
TakeMessage	TakeMessage startTask updateMessagesQueue		GetVCard è il costruttore del behaviour. I metodi handlexxx servono per gestire l'arrivo
GetVCard	FIPAResponseInitiatorT ask startTask handleRefuse handleAgree handleFailure handleInform	id conv conv conv conv data	

	sendRequest GetVCard		di messaggi in accordo al protocollo FIPAResult. Il metodo sendRequest invia il messaggio di richiesta in accordo a protocollo FIPAResult, contenente un identificatore dell'utente di cui si richiede il profilo completo.
MembersManager	FIPAResultInitiatorT ask startTask handleRefuse handleAgree handleFailure handleInform sendRequest MembersManager	id conv conv conv conv data	MembersManag er è il costruttore del behaviour. I metodi handlexxx servono per gestire l'arrivo di messaggi in accordo al protocollo FIPAResult. Il metodo sendRequest invia il messaggio di richiesta in accordo a protocollo FIPAResult, contenente i dati del profilo utente che si intende modificare.
ReceiveArguments	FIPAResultParticipa ntTask startTask sendRefuse sendAgree	conv data data data data	ReceiveArgume nts è il costruttore del behaviour. I metodi sendxxx



	sendFailure	id	servono per l'invio di messaggi in accordo con il protocollo FIPARquest. Il metodo handleRequest serve per gestire il messaggio in arrivo di richiesta del protocollo FIPARquest. Il metodo updateAssignin gArguments gestisce la lista degli argomenti in fase di contrattazione.
	sendInform	conv	
	ReceiveArguments	conv	
	updateAssigningArgu ments	conv	
	handleRequest	data	
ContractArgument	FIPARquestInitiatorT ask	id	ContractArgum ent è il costruttore del behaviour. I metodi handlexxx servono per gestire l'arrivo di messaggi in accordo al protocollo FIPARquest. Il metodo sendRequest invia il messaggio di richiesta in accordo a protocollo FIPARquest. Il metodo updateContracti ngArguments gestisce lo scambio di contrattazioni tra autori ed ecs
	startTask	conv	
	handleRefuse	conv	
	handleAgree	conv	
	handleFailure	data	
	handleInform		
	sendRequest		
	ContractArgument		
	updateContractingArg uments		

UpdateWorkingArguments	FIPAResultInitiatorT	id
	ask	conv
	startTask	conv
	handleRefuse	conv
	handleAgree	conv
	handleFailure	data
	handleInform	
	sendRequest	
	UpdateWorkingArguments	

per un determinato argomento in fase di contrattazione. UpdateWorkingArguments è il costruttore del behaviour. I metodi handlexxx servono per gestire l'arrivo di messaggi in accordo al protocollo FIPAResult. Il metodo sendRequest invia il messaggio di richiesta in accordo a protocollo FIPAResult, contenente gli argomenti accettati e i relativi corsi di appartenenza.

### 9.1.3 Agente: Access Manager

Agente che si occupa della gestione degli utenti del sistema. In particolare questo agente si occupa di:

- Interfacciarsi con gli agenti preposti alla lettura dei dati di login immessi dall'utente
- Verificare i dati di login
- Effettuare le modifiche richieste ai dati personali di un utente

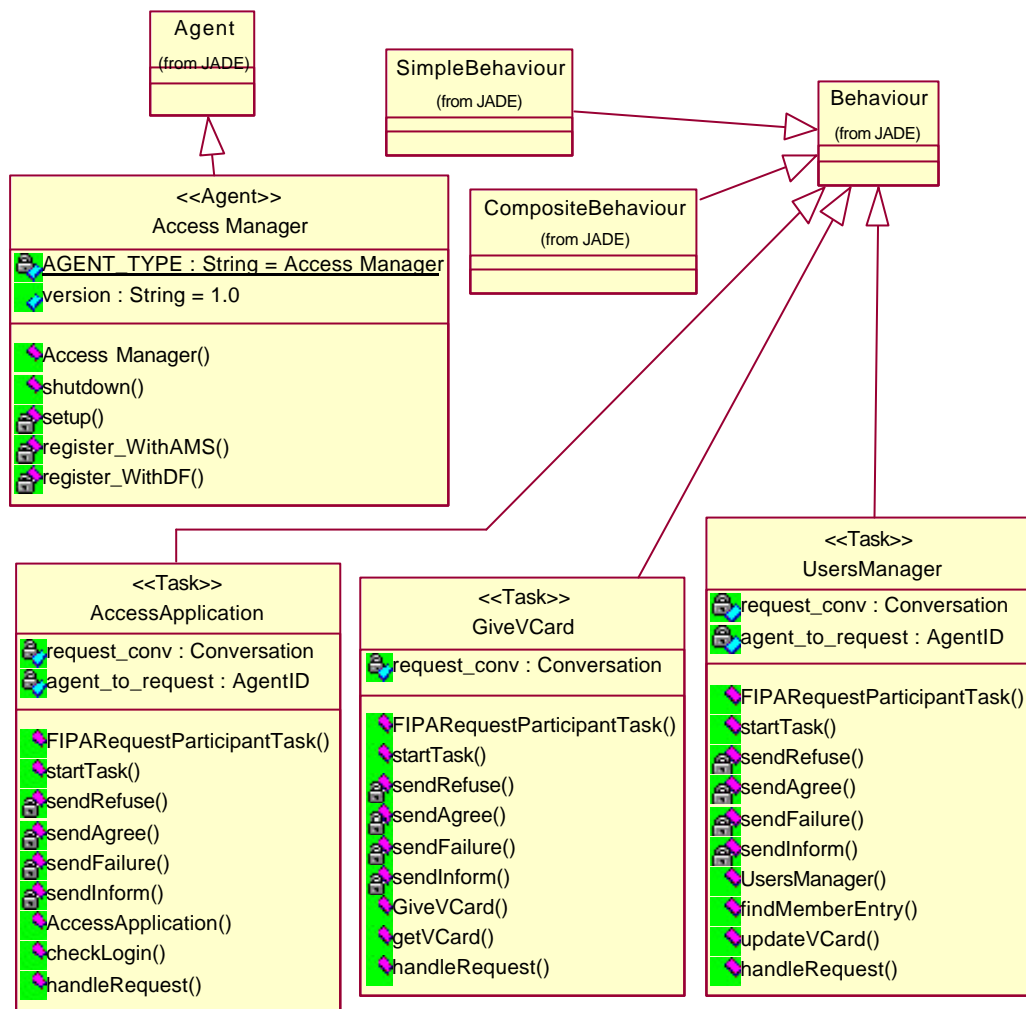


Fig. 40: Single Agent Structure Definition - Access Manager

### 9.1.3.1 Attributi

Class	Attribute	Data Type	Description
Access Manager	AGENT_TYPE version	String String	
SimpleBehaviour			
CompositeBehaviour			
AccessApplication	request_conv agent_to_request	Conversation AgentID	Behaviour per verificare i dati di autenticazione di un utente con

GiveVCard	request_conv	Conversation	quelli presenti in un archivio. Behaviour per la consegna di profili utente richiesti dagli altri agenti.
UsersManager	request_conv agent_to_request	Conversation AgentID	Behaviour per la gestione dell'archivio degli utenti.

### 9.1.3.2 Metodi

Class	Method	Parameters	Description
Access Manager	Access Manager shutdown setup register_WithAMS register_WithDF	platform name ownership	Access Manager è il costruttore dell'agente. Shutdown è il metodo che viene richiamato quando viene terminata l'esecuzione dell'agente. Setup è il metodo che viene richiamato quando viene inizializzato l'agente. Register_withAMS e register_withDF sono dei metodi che registrano gli indirizzi e i servizi dell'agente. Gli agenti AMS e DF forniscono servizi di pagine bianche e pagine gialle.

SimpleBehaviour  
CompositeBehaviour  
AccessApplication

FIPAResponseParticipa  
ntTask conv  
data  
startTask data  
sendRefuse data  
sendAgree data  
sendFailure id  
sendInform conv  
AccessApplication conv  
checkLogin conv  
handleRequest conv  
data

AccessApplicati  
on è il  
costruttore del  
behaviour. I  
metodi sendxxx  
servono per  
l'invio di  
messaggi in  
accordo con il  
protocollo  
FIPAResponse. Il  
metodo  
handleRequest  
serve per gestire  
il messaggio in  
arrivo di  
richiesta del  
protocollo  
FIPAResponse. Il  
metodo  
checkLogin  
confronta i dati  
di  
autenticazione  
dell'utente  
ricevuti con  
quelli presenti  
all'interno  
dell'archivio  
degli utenti.  
GiveVCard è il  
costruttore del  
behaviour. I  
metodi sendxxx  
servono per  
l'invio di  
messaggi in  
accordo con il  
protocollo  
FIPAResponse. Il  
metodo  
handleRequest  
serve per gestire  
il messaggio in  
arrivo di  
richiesta del

GiveVCard

FIPAResponseParticipa  
ntTask conv  
data  
startTask data  
sendRefuse data  
sendAgree data  
sendFailure  
sendInform  
GiveVCard  
getVCard  
handleRequest

			protocollo FIPARquest. Il metodo getVCard accede all'archivio degli utenti e preleva il profilo utente richiesto.
UsersManager	FIPARquestParticipa ntTask	conv data	UsersManager è il costruttore del behaviour. I metodi sendxxx servono per l'invio di messaggi in accordo con il protocollo FIPARquest. Il metodo handleRequest serve per gestire il messaggio in arrivo di richiesta del protocollo FIPARquest. Il metodo findMemberEnt ry ricerca nell'archivio utenti l'utente desiderato. Il metodo update VCard registra idati del profilo utente che sono stati inviati, all'interno dell'archivio.
	startTask	data	
	sendRefuse	data	
	sendAgree	data	
	sendFailure	id	
	sendInform	conv	
	UsersManager	conv	
	findMemberEntry	conv	
	updateVCard	conv	
	handleRequest	data	

#### 9.1.4 Agente: Editor

Agente che si occupa della produzione di contenuti didattici. In particolare questo agente si occupa di:

- gestire l’upload delle risorse multimediali condivisibili e rendere accessibili a tutti gli utenti
- gestire la composizione finale dei corsi completati
- gestire l’upload degli argomenti prodotti dagli autori e renderli accessibili
- gestire le richieste di ricerca e dowload di risorse multimediali condivise da parte degli autori

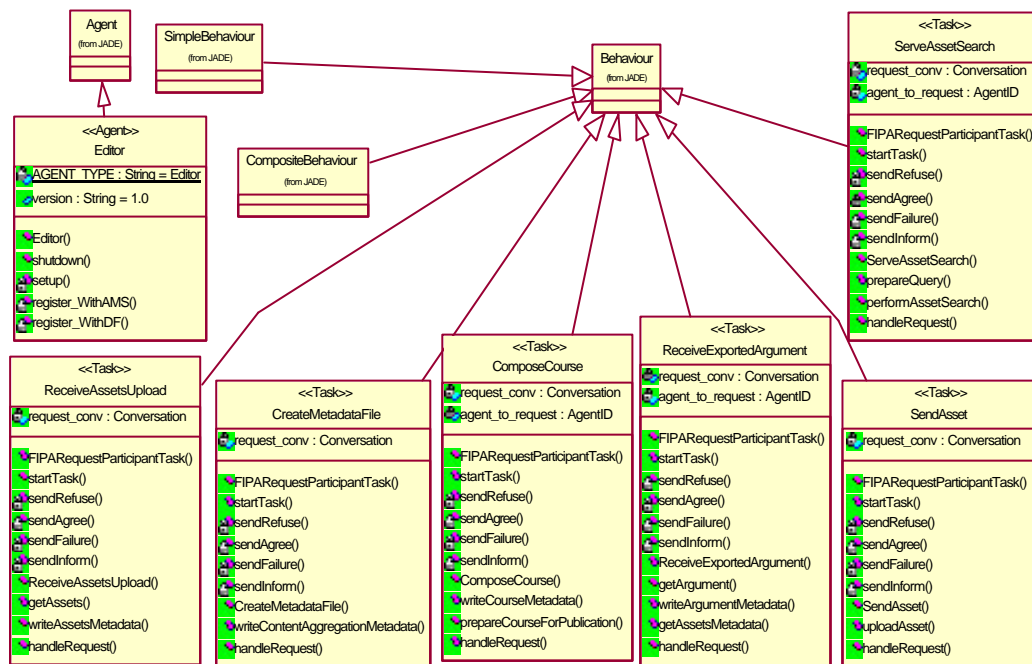


Fig. 41: Single Agent Structure Definition - Editor

### 9.1.4.1 Attributi

Class	Attribute	Data Type	Description
Editor	AGENT_TYPE version	String String	
SimpleBehaviour			
CompositeBehaviour			
ReceiveAssetsUpload	request_conv	Conversation	Behaviour per la ricezione di informazioni atomiche.
CreateMetadataFile	request_conv	Conversation	Behaviour per la creazione del

SendAsset	request_conv	Conversation	file XML di metadati. Behaviour per l'invio di un'informazione atomica che è stata richiesta.
ReceiveExportedArgument	request_conv agent_to_request	Conversation AgentID	Behaviour per la ricezione di un argomento esportato da un autore.
ComposeCourse	request_conv agent_to_request	Conversation AgentID	Behaviour per l'assemblamento di un corso completato.
ServeAssetSearch	request_conv agent_to_request	Conversation AgentID	Behaviour per la ricerca di informazioni atomiche già sviluppate sulla base delle parole chiave ricevute.

#### 9.1.4.2 Metodi

Class	Method	Parameters	Description
Editor	Editor shutdown setup register_WithAMS register_WithDF	platform name ownership	Editor è il costruttore dell'agente. Shutdown è il metodo che viene richiamato quando viene terminata l'esecuzione dell'agente. Setup è il metodo che viene richiamato quando viene inizializzato l'agente. Register_withA



			MS e register_withDF sono dei metodi che registrano gli indirizzi e i servizi dell'agente. Gli agenti AMS e DF forniscono servizi di pagine bianche e pagine gialle.
SimpleBehaviour			
CompositeBehaviour			
ReceiveAssetsUpload	FIPAResultParticipa ntTask startTask sendRefuse sendAgree sendFailure sendInform ReceiveAssetsUpload getAssets writeAssetsMetadata handleRequest	conv data data data data	ReceiveAssetU pload è il costruttore del behaviour. I metodi sendxxx servono per l'invio di messaggi in accordo con il protocollo FIPAResult. Il metodo handleRequest serve per gestire il messaggio in arrivo di richiesta del protocollo FIPAResult. Il metodo getAssets riceve le informazioni atomiche di un argomento e li memorizza in un archivio; il metodo writeAssetsMet adata crea il file XML dei metadati degli asset.
CreateMetadataFile	FIPAResultParticipa ntTask	conv data	CreateMetadata File è il

	startTask	data	costruttore del behaviour. I metodi sendxxx servono per l'invio di messaggi in accordo con il protocollo FIPAResult. Il metodo handleRequest serve per gestire il messaggio in arrivo di richiesta del protocollo FIPAResult. Il metodo writeContentAg gregationMetad ata crea il file XML di metadati per l'aggregazione di contenuti relativa.
	sendRefuse	data	
	sendAgree	data	
	sendFailure		
	sendInform		
	CreateMetadataFile		
	writeContentAggregati onMetadata		
	handleRequest		
SendAsset	FIPAResultParticipa ntTask	conv data	SendAsset è il costruttore del behaviour. I metodi sendxxx servono per l'invio di messaggi in accordo con il protocollo FIPAResult. Il metodo handleRequest serve per gestire il messaggio in arrivo di richiesta del protocollo FIPAResult. Il metodo uploadAsset invia l'informazione
	startTask	data	
	sendRefuse	data	
	sendAgree	data	
	sendFailure		
	sendInform		
	SendAsset		
	uploadAsset		
	handleRequest		



	sendRefuse	data	metodi sendxxx
	sendAgree	data	servono per
	sendFailure	id	l'invio di
	sendInform	conv	messaggi in
	ComposeCourse	conv	accordo con il
	writeCourseMetadata	conv	protocollo
	prepareCourseForPublication	conv	FIPAResult. Il
	handleRequest	data	metodo
			handleRequest
			serve per gestire
			il messaggio in
			arrivo di
			richiesta del
			protocollo
			FIPAResult. Il
			metodo
			writeCourseMet
			adata crea il file
			XML per il
			corso che si sta
			assemblando; il
			metodo
			prepareCourseF
			orPublication
			organizza i file
			dei contenuti
			del corso genera
			un archivio (.zip
			o.jar) in cui è
			contenuto il
			corso. Tale
			archivio è
			pronto per
			essere utilizzato
			da un LMS
			SCORM
			compliant.
ServeAssetSearch	FIPAResultParticipa	conv	ServeAssetSear
	ntTask	data	ch è il
	startTask	data	costruttore del
	sendRefuse	data	behaviour. I
	sendAgree	data	metodi sendxxx
	sendFailure	id	servono per
	sendInform	conv	l'invio di
	ServeAssetSearch	conv	messaggi in
	prepareQuery	conv	accordo con il
	performAssetSearch	conv	protocollo
	handleRequest	data	FIPAResult. Il

metodo  
handleRequest  
serve per gestire  
il messaggio in  
arrivo di  
richiesta del  
protocollo  
FIPARquest. Il  
metodo  
prepareQuery  
prepara la  
richiesta per  
interrogare  
l'archivio delle  
informazioni  
atomiche; il  
metodo  
performAssetSe  
arch ricerca le  
informazioni  
atomiche  
nell'archivio  
sulla base delle  
parole chiave  
ricevute.

### 9.1.5 Agente: Planner

Agente che si occupa della definizione e dell'organizzazione di corsi. In particolare questo agente si occupa di:

- gestire e registrare la struttura dei corsi definiti dall'ECS
- soddisfare le richieste di ricerca ed inserimento degli argomenti sviluppati e condivisi

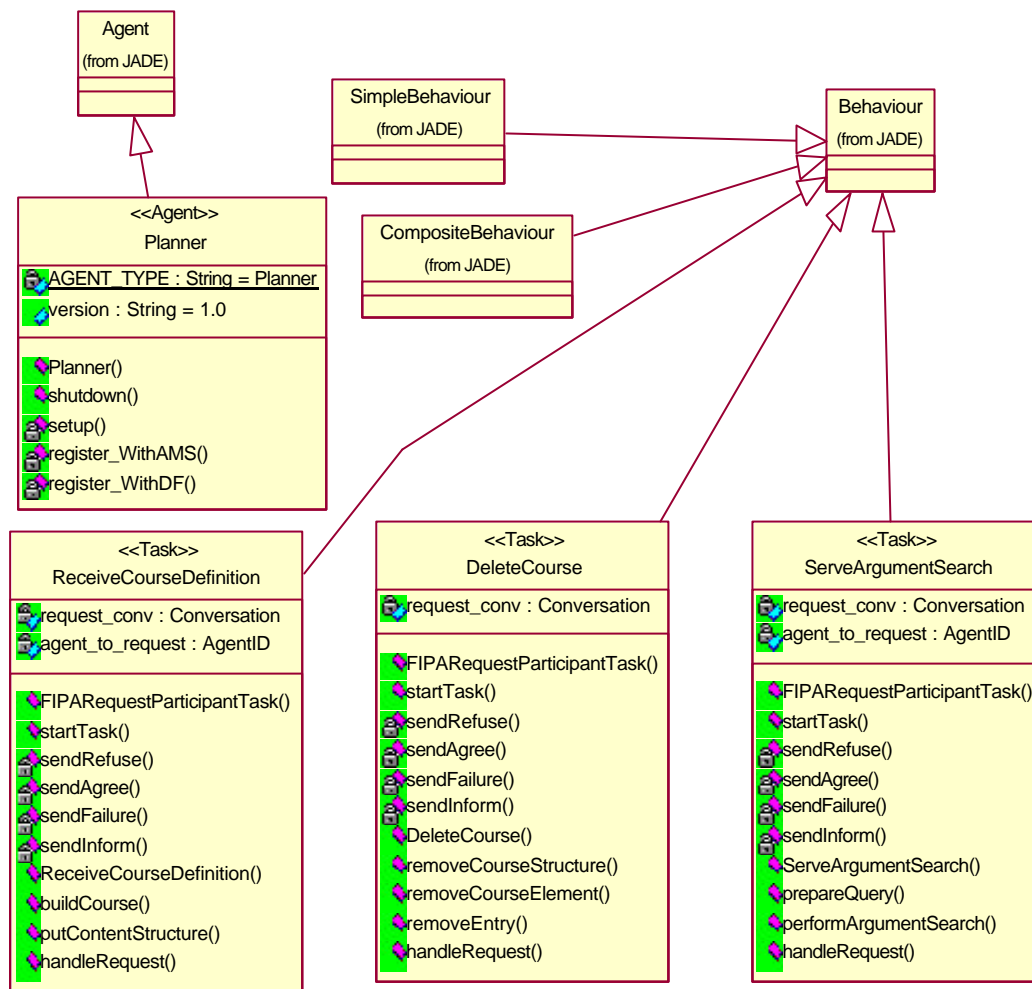


Fig 42: Single Agent Structure Definition - Planner

### 9.1.5.1 Attributi

Class	Attribute	Data Type	Description
Planner	AGENT_TYPE version	String String	
SimpleBehaviour CompositeBehaviour ReceiveCourseDefiniton	request_conv agent_to_request	Conversation AgentID	Behaviour per l'organizzazione e la memorizzazione in un archivio dei corsi definiti

DeleteCourse	request_conv	Conversation	dall'ecs. Behaviour per la cancellazione dei dati necessari allo sviluppo di un corso, per un corso hce è stato completato e già assemblato.
ServeArgumentSearch	request_conv agent_to_request	Conversation AgentID	Behaviour per la ricerca di argomenti già sviluppati da inserire nella definizione di un nuovo corso.

### 9.1.5.2 Metodi

Class	Method	Parameters	Description
Planner	Planner shutdown setup register_WithAMS register_WithDF	platform name ownership	Planner è il costruttore dell'agente. Shutdown è il metodo che viene richiamato quando viene terminata l'esecuzione dell'agente. Setup è il metodo che viene richiamato quando viene inizializzato l'agente. Register_withAMS e register_withDF sono dei metodi che registrano gli indirizzi e i servizi

			dell'agente. Gli agenti AMS e DF forniscono servizi di pagine bianche e pagine gialle.
SimpleBehaviour			
CompositeBehaviour			
ReceiveCourseDefinition	FIPAResultParticipationTask	conv data	ReceiveCourseDefinition è il costruttore del behaviour. I metodi sendxxx servono per l'invio di messaggi in accordo con il protocollo FIPAResult. Il metodo handleRequest serve per gestire il messaggio in arrivo di richiesta del protocollo FIPAResult. Il metodo putContentStructure struttura e memorizza in un archivio un corso definito dall'ecs; il metodo buildCourse ricostruisce la struttura del corso definito a partire dalle informazioni del corso memorizzate nell'archivio. DeleteCourse è il costruttore del behaviour. I metodi sendxxx
	startTask	data	
	sendRefuse	data	
	sendAgree	data	
	sendFailure	id	
	sendInform	conv	
	ReceiveCourseDefinition	conv	
	buildCourse	conv	
	putContentStructure	data	
	handleRequest		
DeleteCourse	FIPAResultParticipationTask	conv data	
	startTask	data	
	sendRefuse	data	



	sendAgree sendFailure sendInform DeleteCourse removeCourseStructure removeCourseElement removeEntry handleRequest	data	servono per l'invio di messaggi in accordo con il protocollo FIPARquest. Il metodo handleRequest serve per gestire il messaggio in arrivo di richiesta del protocollo FIPARquest. I metodi removeCourseS tructure, removeCourseE lement e removeEntry cancellano dall'archivio dei corsi in fase di sviluppo tutte le informazioni relative ad un corso che sia stato completato è già assemblato.
ServeArgumentSearch	FIPARquestParticipa ntTask startTask sendRefuse sendAgree sendFailure sendInform ServeArgumentSearch prepareQuery performArgumentSear ch handleRequest	conv data data data data id conv conv conv conv data	ServeArgument Search è il costruttore del behaviour. I metodi sendxxx servono per l'invio di messaggi in accordo con il protocollo FIPARquest. Il metodo handleRequest serve per gestire il messaggio in arrivo di richiesta del protocollo

FIPAResult. Il metodo prepareQuery prepara la richiesta per interrogare l'archivio argomenti; il metodo performArgumentSearch ricerca gli argomenti nell'archivio sulla base delle parole chiave ricevute.

### 9.1.6 Agente: Ecs Gui

Agente che si interfaccia all'Editorial Central Staff e smista le richieste agli altri agenti.

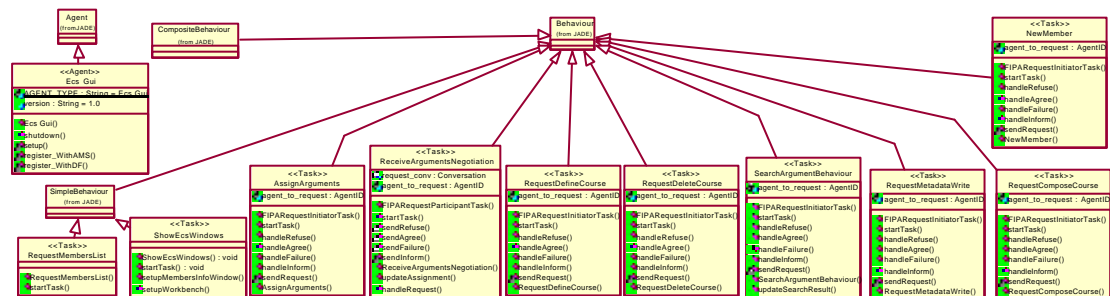


Fig. 43: Single Agent Structure Definition - Ecs Gui

#### 9.1.6.1 Attributi

Class	Attribute	Data Type	Description
Ecs Gui	AGENT_TYPE version	String String	
SimpleBehaviour			
CompositeBehaviour			
AssignArguments	agent_to_request	AgentID	Behaviour per l'assegnazione degli argomenti

			agli autori.
ReceiveArgumentsNegotiation	request_conv agent_to_request	Conversation AgentID	Behaviour per la gestione delle contrattazioni ricevute dagli autori.
RequestMembersList			Behaviour per richiedere la lista degli utenti registrati al sistema.
RequestDefineCourse	agent_to_request	AgentID	Behaviour per la richiesta di definizione di un corso.
RequestDeleteCourse	agent_to_request	AgentID	Behaviour per la richiesta di cancellazione delle informazioni per lo sviluppo di un corso che sia già stato completato e assemblato.
SearchArgumentBehaviour	agent_to_request	AgentID	Behaviour per la ricerca di argomenti già sviluppati da inserire nella definizione di un nuovo corso.
RequestMetadataWrite	agent_to_request	AgentID	Behaviour per la richiesta di scrittura dei metadati in un file XML.
RequestComposeCourse	agent_to_request	AgentID	Behaviour per la richiesta di assemblamento di un corso i cui argomenti sono stati completati.
NewMember	agent_to_request	AgentID	Behaviour per la richiesta di aggiunta di un nuovo profilo

utente  
all'interno  
dell'archivio.

ShowEcsWindows

Behaviour per  
la gestione delle  
interfacce  
relative alle  
funzionalità che  
il sistema mette  
a disposizione  
dell'ecs.

### 9.1.6.2 Metodi

Class	Method	Parameters	Description
Ecs Gui	Ecs Gui shutdown setup register_WithAMS register_WithDF	platform name ownership	Planner è il costruttore dell'agente. Shutdown è il metodo che viene richiamato quando viene terminata l'esecuzione dell'agente. Setup è il metodo che viene richiamato quando viene inizializzato l'agente. Register_withA MS e register_withDF sono dei metodi che registrano gli indirizzi e i servizi dell'agente. Gli agenti AMS e DF forniscono servizi di pagine bianche e pagine gialle.

SimpleBehaviour			
CompositeBehaviour			
AssignArguments	FIPAResultInitiatorT	id	AssignArgumen
	ask	conv	ts è il
	startTask	conv	costruttore del
	handleRefuse	conv	behaviour. I
	handleAgree	conv	metodi
	handleFailure	data	handlexxx
	handleInform		servono per
	sendRequest		gestire l'arrivo
	AssignArguments		di messaggi in
			accordo al
			protocollo
			FIPAResult. Il
			metodo
			sendRequest
			invia il
			messaggio di
			richiesta in
			accordo a
			protocollo
			FIPAResult, e
			contiene sia gli
			argomenti che i
			relativi corsi di
			appartenenza
			che l'ecs vuole
			inviare
			all'autore
			selezionato.
ReceiveArgumentsNe	FIPAResultParticipa	conv	ReceiveArgume
gotiation	ntTask	data	ntsNegotiation è
	startTask	data	il costruttore del
	sendRefuse	data	behaviour. I
	sendAgree	data	metodi sendxxx
	sendFailure	id	servono per
	sendInform	conv	l'invio di
	ReceiveArgumentsNe	conv	messaggi in
	gotiation	conv	accordo con il
	updateAssignment	conv	protocollo
	handleRequest	data	FIPAResult. Il
			metodo
			handleRequest
			serve per gestire
			il messaggio in
			arrivo di
			richiesta del
			protocollo

			FIPAResult. Il metodo updateAssignment gestisce le contrattazioni degli argomenti in arrivo dagli autori.
RequestMembersList	RequestMembersList startTask		RequestMembersList è il costruttore del behaviour. Il behaviour invia un messaggio per richiedere la lista degli utenti registrati al sistema.
RequestDefineCourse	FIPAResultInitiatorTask ask startTask handleRefuse handleAgree handleFailure handleInform sendRequest RequestDefineCourse	id conv conv conv conv data	RequestDefineCourse è il costruttore del behaviour. I metodi handlexxxx servono per gestire l'arrivo di messaggi in accordo al protocollo FIPAResult. Il metodo sendRequest invia il messaggio di richiesta in accordo al protocollo FIPAResult, contenente i dati della definizione del corso e dei suoi argomenti.
RequestDeleteCourse	FIPAResultInitiatorTask ask startTask handleRefuse handleAgree	id conv conv conv conv	RequestDeleteCourse è il costruttore del behaviour. I metodi

	handleFailure handleInform sendRequest RequestDeleteCourse	data	handlexxx servono per gestire l'arrivo di messaggi in accordo al protocollo FIPAResult. Il metodo sendRequest invia il messaggio di richiesta in accordo al protocollo FIPAResult, contenente la richiesta per la cancellazione delle informazioni temporanee relative al corso desiderato.
SearchArgument Behaviour	FIPAResultInitiatorT ask startTask handleRefuse handleAgree handleFailure handleInform sendRequest SearchArgumentBehaviour updateSearchResult	id conv conv conv conv data	SearchArgumentBehaviour è il costruttore del behaviour. I metodi handlexxx servono per gestire l'arrivo di messaggi in accordo al protocollo FIPAResult. Il metodo sendRequest invia il messaggio di richiesta in accordo al protocollo FIPAResult, contenente le parole chiave sulla base delle quali deve essere effettuata

RequestMetadataWrite	FIPAResponseInitiatorT	id
	ask	conv
	startTask	conv
	handleRefuse	conv
	handleAgree	conv
	handleFailure	data
	handleInform	
	sendRequest	
	RequestMetadataWrite	

RequestComposeCourse	FIPAResponseInitiatorT	id
	ask	conv
	startTask	conv
	handleRefuse	conv
	handleAgree	conv
	handleFailure	data
	handleInform	
	sendRequest	
	RequestComposeCourse	

la ricerca; il metodo updateSearchResult riceve e gestisce il risultato della ricerca degli argomenti.

RequestMetadataWrite è il costruttore del behaviour. I metodi handlexxx servono per gestire l'arrivo di messaggi in accordo al protocollo FIPAResponse. Il metodo sendRequest invia il messaggio di richiesta in accordo al protocollo FIPAResponse, contenente la richiesta per la scrittura dei metadati inseriti dall'utente.

RequestComposeCourse è il costruttore del behaviour. I metodi handlexxx servono per gestire l'arrivo di messaggi in accordo al protocollo FIPAResponse. Il metodo sendRequest invia il

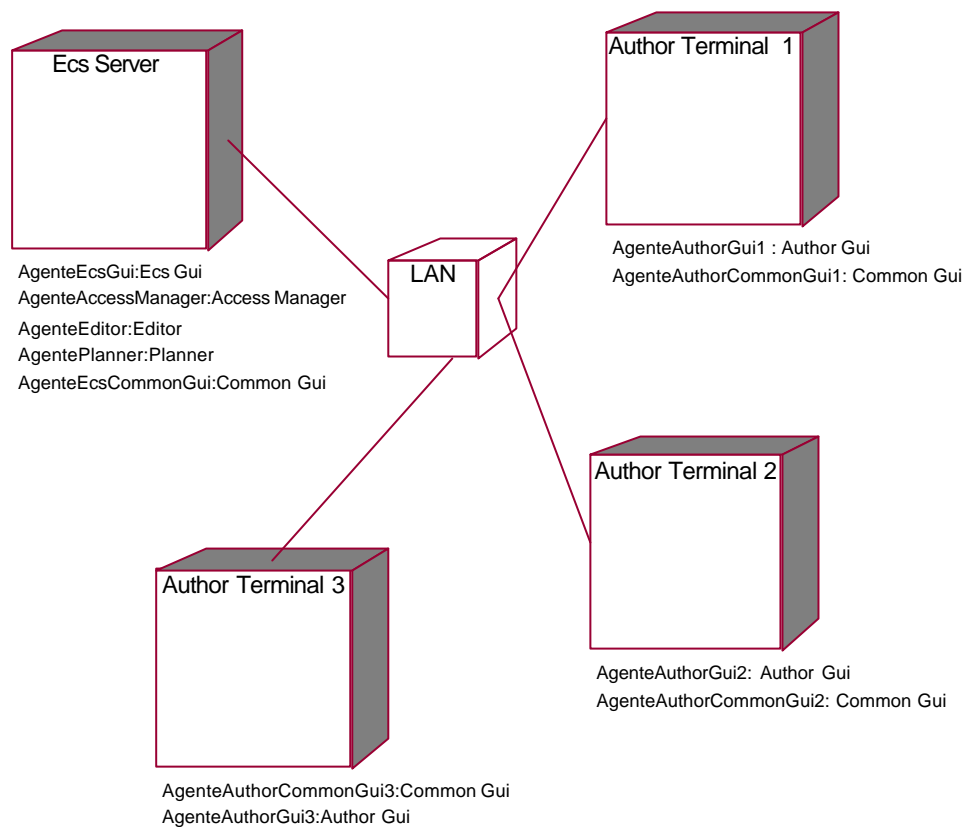


NewMember	FIPAResponseInitiatorT ask startTask handleRefuse handleAgree handleFailure handleInform sendRequest NewMember	id conv conv conv conv data	messaggio di richiesta in accordo al protocollo FIPAResponse, contenente la richiesta per l'assemblament o del corso desiderato. NewMember è il costruttore del behaviour. I metodi handlexxx servono per gestire l'arrivo di messaggi in accordo al protocollo FIPAResponse. Il metodo sendRequest invia il messaggio di richiesta in accordo al protocollo FIPAResponse, contenente i dati del profilo utente che si desidera registrare nel sistema. ShowEcsWindows
ShowEcsWindows	ShowEcsWindows startTask setupMembersInfoWindow setupWorkbench		ShowEcsWindows è il costruttore del behavior. Il metodo setupMembersInfoWindow gestisce l'interfaccia per i profili degli utenti; il metodo setupWorkbench

h gestisce  
l'interfaccia che  
mostra le  
funzionalità che  
il sistema mette  
a disposizione  
dell'ecs.

## **10 Fase 10 Deployment Configuration**

In questo diagramma viene mostrata la localizzazione degli agenti all'interno dei diversi computer coinvolti nel sistema. In tale diagramma si prevede l'utilizzo di un server per l'editorial central staff e un terminale per ogni autore che partecipa alla produzione dei contenuti. Nel server viene lanciata la piattaforma JADE con il main-container al cui interno si trovano cinque agenti; per ogni autore viene aggiunto alla piattaforma un container al cui interno vi sono due agenti. I computer sono connessi tra loro tramite una LAN e la piattaforma risulta distribuita tra i vari computer su ognuno dei quali è presente un container diverso.



**Figura 38: Deployment Configuration.**

### ***Ecs Server***

- AgenteEcsGui:Ecs Gui
- AgenteAccessManager:Access Manager
- AgenteEditor:Editor
- AgentePlanner:Planner
- AgenteEcsCommonGui:Common Gui

### ***Author Terminal 1***

- AgenteAuthorGui1 : Author Gui
- AgenteAuthorCommonGui1: Common Gui

## ***Author Terminal 2***

- AgenteAuthorGui2: Author Gui
- AgenteAuthorCommonGui2: Common Gui

## ***Author Terminal 3***

- AgenteAuthorCommonGui3:Common Gui
- AgenteAuthorGui3:Author Gui

## **11 Progetto dell'archivio**

### **11.1 Progetto Concettuale**

#### **11.1.1 Descrizione dell'archivio**

Si vuole realizzare un archivio contenente dei corsi.

Ogni corso è composto da uno o più argomenti e da zero o più sottocorsi, che sono a loro volta corsi.

Ogni argomento è composto da una o più informazioni atomiche ed è nello stato status argomento.

Lo status argomento è caratterizzato dallo stato.

Ogni informazione atomica può partecipare alla composizione di uno o più argomenti.

Ogni argomento può partecipare alla composizione di uno o più corsi.

Ogni corso è caratterizzato da un nome ed una introduzione.

Ogni argomento è caratterizzato da un nome, un nome del file di metadati, una introduzione ed un commento.

Ogni informazione atomica è caratterizzata da un nome e da un nome del file di metadati.

Ogni corso, così come ogni argomento, così come ogni informazione atomica, è descritta tramite una o più descrizioni.

Ogni corso, così come ogni argomento, così come ogni informazione atomica, possiede uno o più titoli.

Ogni corso, così come ogni argomento, così come ogni informazione atomica, è rintracciabile tramite una o più parole chiavi.

Ogni descrizione è caratterizzata dal testo della descrizione, dalla lingua in cui questa è espressa e dal tipo di risorsa a cui essa si riferisce.

Ogni titolo è caratterizzato dal testo del titolo, dalla lingua in cui questo è espresso e dal tipo di risorsa a cui esso si riferisce.

Ogni parola chiave è caratterizzata dal testo della parola, dalla lingua in cui questa è espressa, dal tipo di risorsa a cui si riferisce. Ad ogni parola chiave possono corrispondere uno o più corsi o argomenti o informazioni atomiche.

Ogni tipo di risorsa è caratterizzato dal nome del tipo.

### 11.1.2 Identificazione degli elementi dell'ERD

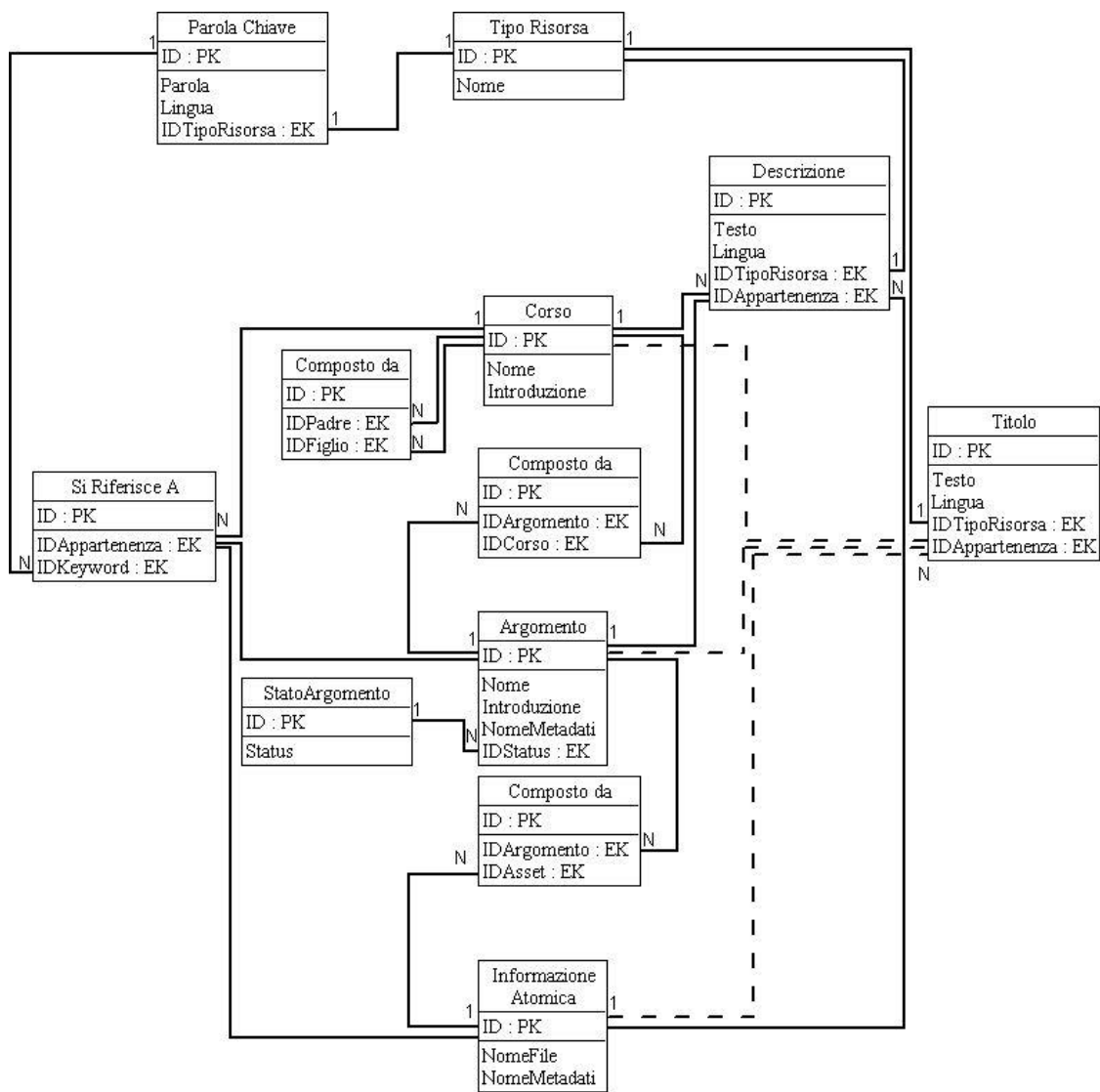
Dalla descrizione precedente si è proceduto all'individuazione di entità, relazioni ed attributi, che sono di seguito descritti.

- L'entità *corso* è in relazione *compostoDa* con l'entità *argomento* e con l'entità *corso*, in relazione *descrittoDa* con l'entità *descrizione*, in relazione *possiede* con l'entità *titolo*, in relazione *rintracciabileCon* con l'entità *parolaChiave*; gli attributi sono: *nome*, *introduzione*
- L'entità *argomento* è in relazione *compostoDa* con l'entità *informazioneAtomica*, in relazione *descrittoDa* con l'entità *descrizione*, in relazione *possiede* con l'entità *titolo*, in relazione *rintracciabileCon* con l'entità *parolaChiave*, in relazione *èNelloStato* con l'entità *statusArgomento*; gli attributi sono: *nome*, *nomeMetadati*, *introduzione*, *commento*
- L'entità *informazioneAtomica* è in relazione *descrittoDa* con l'entità *descrizione*, in relazione *possiede* con l'entità *titolo*, in relazione *rintracciabileCon* con l'entità *parolaChiave*; gli attributi sono: *nome*, *nomeMetadati*
- L'entità *descrizione* è in relazione *siRiferisceA* con l'entità *tipoRisorsa*; gli attributi sono: *testo*, *lingua*
- L'entità *titolo* è in relazione *siRiferisceA* con l'entità *tipoRisorsa*; gli attributi sono: *testo*, *lingua*
- L'entità *parolaChiave* è in relazione *siRiferisceA* con l'entità *tipoRisorsa*; gli attributi sono: *parola*, *lingua*



## 11.3 Progetto Logico

Dall'ERD prima visto, si possono trarre le seguenti tabelle nelle quali sono state aggiunte le chiavi esterne necessarie alle relazioni; si noti che, per rendere il diagramma più leggibile, si è provveduto ad indicare con un tratteggio alcune linee e, dove possibile, a non indicare più di una volta la cardinalità in corrispondenza ad attributi coinvolti in più relazioni.



## 11.4 Tabelle

Corso:

Campo	Tipo
ID	Contatore
Nome	Testo (50) (richiesto:si, indice: no)
Introduzione	Testo (255) (richiesto: si, indice:no)

Argomento:

Campo	Tipo
ID	Contatore
Nome	Testo (50) (richiesto:si, indice: no)
Introduzione	Testo (255) (richiesto: si, indice:no)
NomeMetadati	Testo (50) (richiesto:si, indice: no)

InformazioneAtomica:

Campo	Tipo
ID	Contatore
Nome	Testo (50) (richiesto:si, indice: no)
NomeMetadati	Testo (50) (richiesto:si, indice: no)

Descrizione:

Campo	Tipo
ID	Contatore
IDAppartenenza	Intero lungo
Lingua	Testo (2) (richiesto:si, indice:no)
IDTipoRisorsa	Intero lungo
Testo	Testo (255) (richiesto:si, indice: no)

Titolo:

Campo	Tipo
ID	Contatore
IDAppartenenza	Intero lungo
Lingua	Testo (2) (richiesto:si, indice:no)



IDTipoRisorsa	Intero lungo
Testo	Testo (255) (richiesto:si, indice: no)

ParolaChiave:

Campo	Tipo
ID	Contatore
Lingua	Testo (2) (richiesto:si, indice:no)
IDTipoRisorsa	Intero lungo
Parola	Testo (50) (richiesto:si, indice: no)

TipoRisorsa:

Campo	Tipo
ID	Contatore
Nome	Testo (30) (richiesto:si, indice: no)

StatoArgomento:

Campo	Tipo
ID	Contatore
Status	Testo (30) (richiesto:si, indice: no)

CompostoDaCorsoArgomento:

Campo	Tipo
ID	Contatore
IDCorso	Intero lungo
IDArgomento	Intero lungo

CompostoDaCorsoCorso:

Campo	Tipo
ID	Contatore
IDCorsoPadre	Intero lungo
IDCorsoFiglio	Intero lungo

CompostoDaArgomentoInformazioneAtomica:

<b>Campo</b>	<b>Tipo</b>
ID	Contatore
IDArgomento	Intero lungo
IDInformazioneAtomica	Intero lungo

SiRiferisceA

<b>Campo</b>	<b>Tipo</b>
ID	Contatore
IDKeyword	Intero lungo
IDRisorsa	Intero lungo

## **Manuale dell'utente**

## 1 Utilizzo del tool

Il tool è stato sviluppato per essere installato su tutte le macchine coinvolte nello sviluppo di corsi, in configurazione autore o Ecs, a seconda del ruolo dell'utente.

All'avvio si presenta una procedura di login che richiede l'immissione di nome utente e password e che regola l'accesso al sistema.

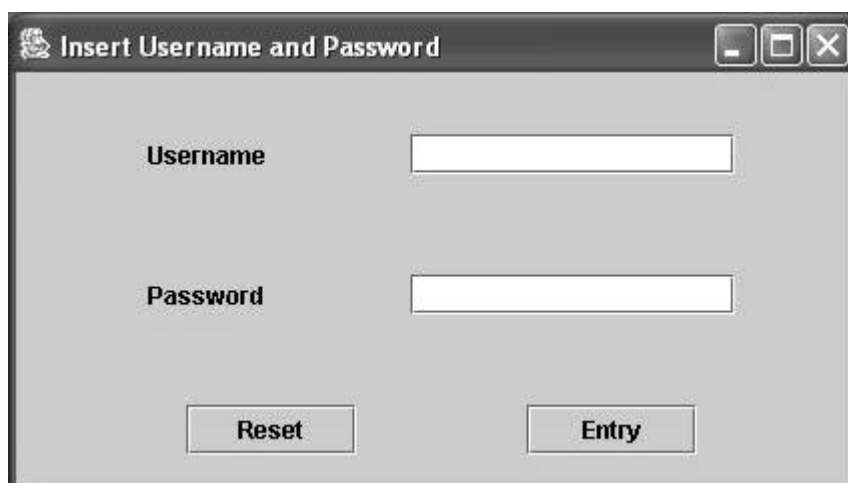


Figura 1: Finestra di Login per l'accesso al sistema

L'inserimento dei dati errati di Login, comporta la comparsa di una finestra di avviso di login errato.

Inserendo correttamente *username* e *password*, viene presentato all'utente il *Workbench*, che consta di un menù a tendina poco invadente, che presenta le opportune funzionalità a seconda che l'accesso sia stato effettuato da un autore o dall'Ecs.

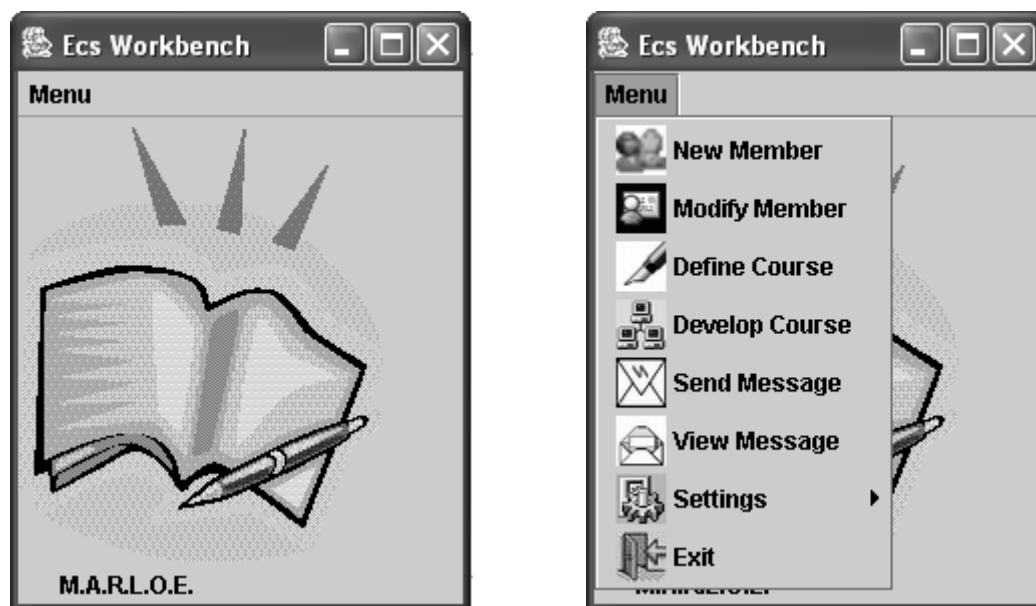


Figura 2: Workbench dell'Ecs prima e dopo la pressione del pulsante *Menu*

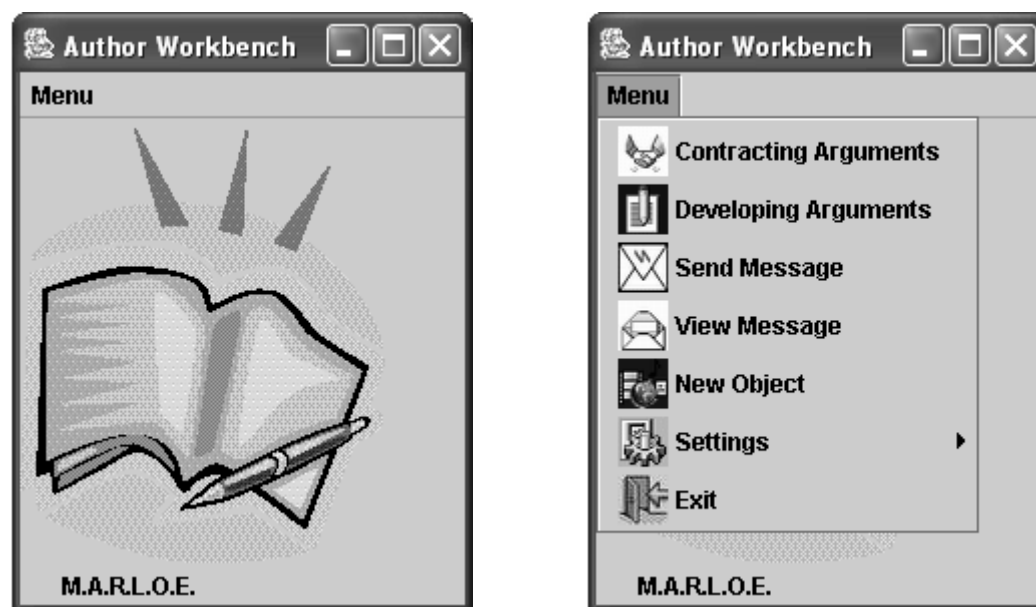


Figura 2: Workbench dell'Autore prima e dopo la pressione del pulsante *Menu*

Le funzionalità di immediato accesso dai relativi workbench sono le seguenti:

- ECS
  - *New member*: registrazione di un nuovo utente
  - *Modify member*: modifica dei dati di registrazione di un utente
  - *Define course*: definizione di nuovi corsi
  - *Develop course*: assegnazione e assemblamento dei corsi
  - *Send message*: invio di messaggi

- *View message*: lettura dei messaggi ricevuti
- *Settings*: impostazioni del sistema
- *Exit*: uscita dal sistema
- Autore:
  - *Contracting arguments*: contrattazione degli argomenti assegnati
  - *Developing arguments*: sviluppo degli argomenti assegnati
  - *New object*: sviluppo di una nuova informazione atomica a se stante
  - *Send message*: invio di messaggi
  - *View message*: lettura dei messaggi ricevuti
  - *Settings*: impostazioni del sistema
  - *Exit*: uscita dal sistema

Tali funzionalità verranno di seguito illustrate.

## **2 Funzionalità dell'ECS**

### **2.1 New member**

Mediante tale funzionalità è possibile per l'ECS aggiungere nuovi membri al gruppo di lavoro. La scelta di tale funzionalità comporta la comparsa della seguente finestra:

Personal Data			
Name	<input type="text"/>	Title	<input type="text"/>
Telephone	<input type="text"/>	Email	<input type="text"/>
Nickname	<input type="text"/>	Class	<input type="text"/>
Role	<input type="text"/>	Organization	<input type="text"/>
OfficeID	<input type="text"/>	URL	<input type="text"/>
Username	<input type="text"/>	Password	<input type="text"/>
Birthday	<input type="text"/>	Street	<input type="text"/>
Locality	<input type="text"/>	Region	<input type="text"/>
Country	<input type="text"/>	Postal Code	<input type="text"/>
Note	<input type="text"/>		
		Clear	Submit

Figura 4: Finestra di immissione dei dati personali di un utente

Mediante l'uso di tale finestra è possibile immettere tutti i dati personali dell'autore che si vuole aggiungere. Questi verranno successivamente utilizzati per riempire una struttura dati conforme allo standard *vCard* sviluppato da *Internet Mail Consortium* (IMC) [\*7\*] [\*8\*] [\*9\*].

L'immissione di alcuni dati come *email* e *note* è fondamentale per alcune funzioni come lo scambio dei messaggi e la scelta degli autori a cui assegnare lo sviluppo degli argomenti.

La pressione del pulsante *Submit* implica la registrazione dei dati; mediante il pulsante *clear* vengono ripuliti tutti i campi fino a quel momento riempiti.

## 2.2 Modify member

Tramite tale funzionalità è possibile per l'ECS modificare i dati personali dei membri del gruppo di lavoro. Quando tale funzione viene scelta, si apre la seguente finestra:

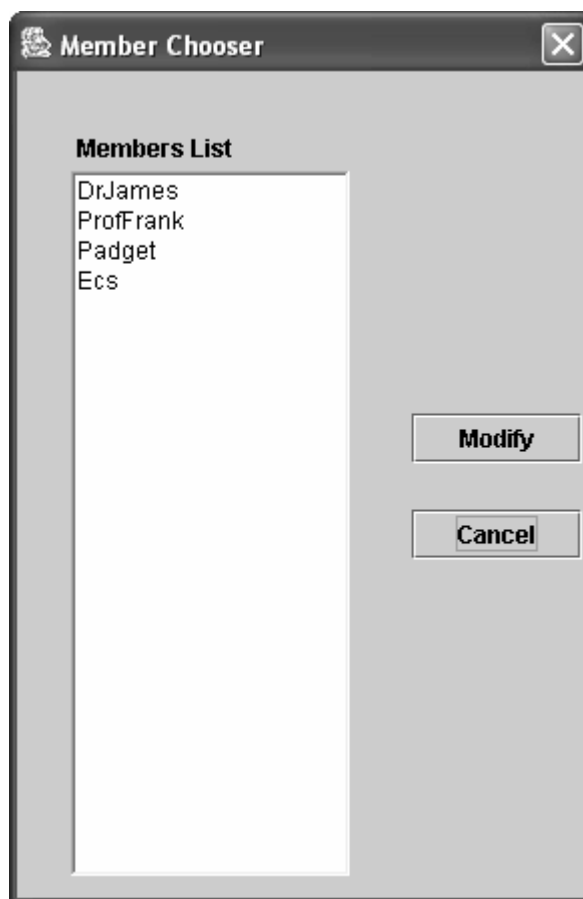


Figura 5: Finestra di scelta dell'utente di cui si vogliono modificare i dati

Dopo avere selezionato l'utente di cui si vogliono modificare i dati, sarà possibile cancellare la *entry* mediante il pulsante *Cancel* (comparirà una finestra di richiesta di conferma), oppure modificare solo alcuni dati mediante il pulsante *Modify*, che comporterà l'apertura di una finestra come quella di figura 4 in cui compariranno i dati memorizzati al momento.

## 2.3 Define course

Tale funzionalità permette la definizione di nuovi corsi. Inizialmente si dovrà immettere il titolo e la descrizione del corso nelle apposite aree di testo. Successivamente si potranno aggiungere gli argomenti che compongono il corso, semplicemente inserendone il titolo e la descrizione e premendo il pulsante *Add*.



**Define Course**

**Course Title** OOP

**Course Description** Programmazione orientata agli oggetti

Tree View:

- OOP
  - Java
  - C++

**Subcourse Title**

**Subcourse Description**

**Argument Title** Java vs C++

**Argument Description** Differenze fra Java e C++

Buttons: Delete, Add, Search, Clear, Commit

Figura 6: Definizione di un nuovo corso composto da tre argomenti

Allo stesso modo è possibile inserire sottocorsi, all'interno dei quali si dovrà poi inserire almeno un argomento.

Nella figura seguente viene mostrata la procedura di definizione di un corso (OOP) composto da due sottocorsi (Java e C++) ed un argomento (Java vs C++).

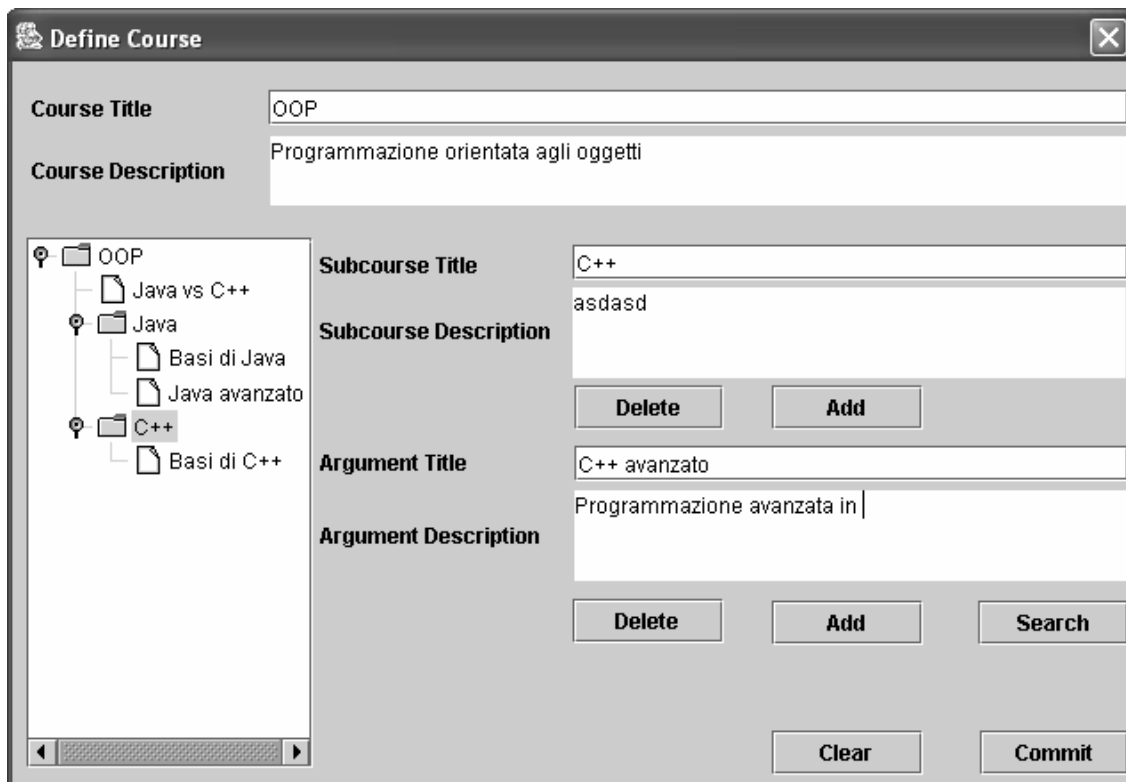


Figura 7: Definizione di un nuovo corso contenente due sottocorsi.

Gli argomenti possono essere inseriti anche scegliendo tra quelli già sviluppati; la pressione del pulsante *Search* provoca l'apertura della finestra di ricerca.



Figura 8: Ricerca di Argomenti già sviluppati e condivisi

All'apertura viene mostrato un elenco di tutti gli argomenti disponibili; è possibile selezionarne uno, visualizzarne la descrizione nell'apposita area di testo ed, eventualmente, vederne una preview utilizzando il pulsante *preview*, che comporterà l'apertura dell'argomento all'interno del *browser* di default.

Inserendo una o più parole chiave all'interno dello spazio apposito, verrà effettuata una ricerca all'interno dell'archivio di argomenti limitatamente all'elenco mostrato, che verrà aggiornato con il sottoinsieme di argomenti che possiedono le parole chiavi nel titolo o nella descrizione o tra le *keywords*; l'operazione può essere ripetuta più volte, limitando sempre più l'insieme di argomenti visualizzati. Quando tale insieme sarà troppo ristretto, l'utente potrà tornare all'elenco completo utilizzando il pulsante *reset*.

La pressione del pulsante *Commit* della finestra di definizione dei corsi (Figg. 6 e 7) comporta la memorizzazione del corso così come è stato definito. Il corso sarà dunque pronto per l'assegnazione degli argomenti agli autori.

Qualora si volesse ripulire la finestra di definizione (a causa di un errore ad esempio) si potranno utilizzare i pulsanti *Delete* o *Clear*. La pressione di uno dei due pulsanti *Delete* causa la rimozione dal corso del sottocorso o argomento selezionato. La pressione del pulsante *Clear*, invece, comporta la cancellazione dell'intero corso.

Dopo aver definito almeno un corso sarà possibile assegnarne gli argomenti agli autori tramite la funzionalità *Develop Course*; si noti che qualora si tentasse di accedere a tale funzionalità senza aver definito alcun corso, il sistema mostrerà un avviso che tale operazione non è eseguibile.

## 2.4 Develop Corse

Come esposto sopra, tale funzionalità permette l'assegnazione degli argomenti di un corso agli autori. Oltre a ciò sarà possibile tenere sotto controllo l'operato degli autori, verificando quando questi terminano la produzione degli argomenti.

Allorché tutti gli argomenti di un corso saranno completati, si potrà assemblare il corso e prepararlo per la pubblicazione.

Riassumendo da tale funzionalità si potrà accedere alle seguenti funzioni:

- assegnazione e contrattazione degli argomenti

- controllo del ciclo di vita degli argomenti
- completamento ed assemblamento dei corsi

Si supponga che si sia appena definito un solo corso (si veda la figura 6); la pressione del pulsante *Develop Course* dal *Workbench* dell'ECS comporta l'apertura della finestra mostrata in figura:

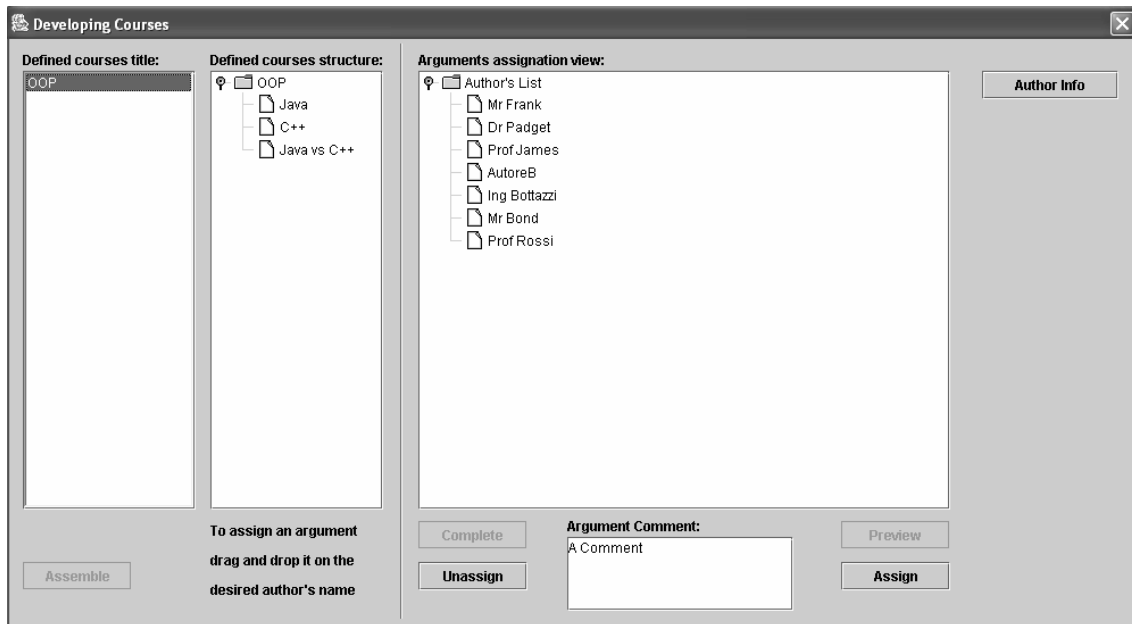


Figura 9: Assegnazione e contrattazione di argomenti 1

Si leggeranno:

- nella prima colonna i nomi dei corsi attualmente in sviluppo, in questo caso solo uno
- nella seconda colonna la struttura ad albero del corso, con sottocorsi ed argomenti, in questo caso tre argomenti
- nella terza colonna, la più grande, la lista degli autori

Per assegnare un argomento ad un autore sarà sufficiente effettuare la *drag and drop* dalla seconda colonna verso il nome dell'autore a cui si vuole assegnare l'argomento nella terza colonna; l'invio dell'assegnazione verrà effettuato con la pressione del pulsante *Assign*.

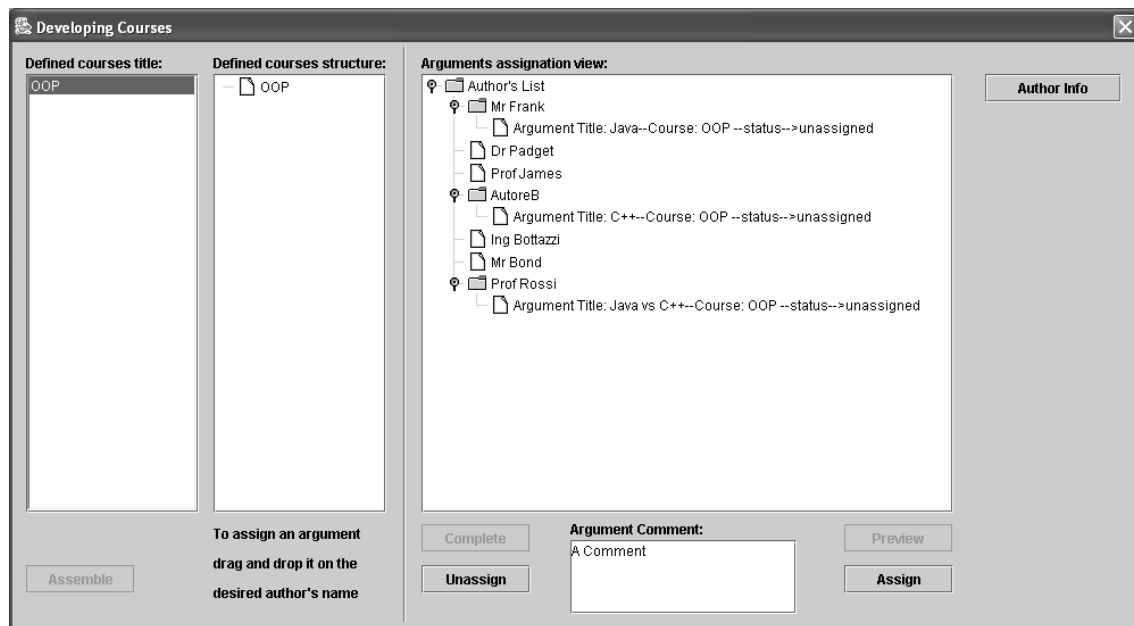


Figura 10: Assegnazione e contrattazione di argomenti 2

In figura 10 si può notare che l'argomento "C++" è stato assegnato all'autore "AutoreB", l'argomento "Java vs C++" all'autore "Prof Rossi", l'argomento "Java" all'autore "Mr Frank". Questi argomenti, inoltre, sono scomparsi dall'albero che rappresenta il corso il fase di sviluppo della seconda colonna.

Per facilitare le operazioni di assegnazione è anche presente il pulsante *Author Info* che permette la visualizzazione dei dati personali dell'autore selezionato dalla terza colonna, che verranno mostrati in una finestra uguale a quella di figura 4 (i cui campi di testo saranno riempiti con i dati dell'autore relativo).

E' possibile adesso inserire un commento ad ogni argomento assegnato, con lo scopo, ad esempio, di chiarire meglio quali contenuti ci si aspetta per esso. Per fare ciò sarà sufficiente selezionare un argomento dalla terza colonna e digitare il commento che si desidera inserire nello spazio contrassegnato dal nome *Argument Comment*. Una volta inseriti i commenti desiderati, sarà possibile far partire l'assegnazione premendo il pulsante *Assign*.

Come è possibile notare ogni argomento della terza colonna è contrassegnato con uno *status*, in questo caso *unassigned*. Tale stato passerà ad *assigned* se l'autore relativo deciderà di accettare l'assegnazione, o passerà a *refused* se questi deciderà di rifiutarla (si veda più avanti la funzionalità *Contracting Arguments* degli autori per approfondire la procedura di contrattazione).

Nel secondo caso si potrà leggere nell'apposita area di testo l'eventuale commento inserito dall'autore per modificare il suo rifiuto. Si potrà, dunque, decidere di riassegnare l'argomento allo stesso autore premendo nuovamente il pulsante *Assign* (eventualmente dopo aver inserito un ulteriore commento), oppure recedere dall'assegnazione premendo il pulsante *Unassign*; in questo modo l'argomento sparirà dalla colonna di destra e verrà riaggiunto all'albero che rappresenta il corso nella seconda colonna, in attesa di essere assegnato ad un altro autore.

Un ulteriore *status* in cui può trovarsi un argomento nella terza colonna è *Completed*; ciò vorrà dire che l'autore ha completato lo sviluppo dell'argomento e ne ha eseguito l'esportazione, rendendo disponibile per la condivisione l'argomento stesso e tutte le informazioni atomiche che lo compongono. Si noti che quando un autore esporta un argomento, viene visualizzata una notifica sul terminale dell'Ecs.

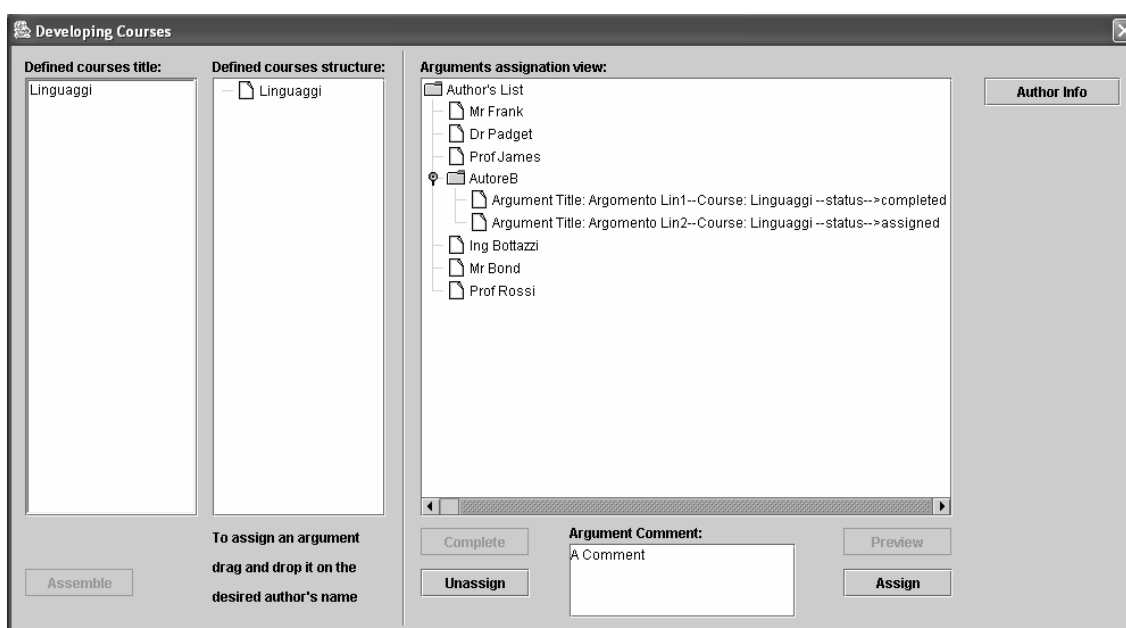


Figura 11: Completamento di un argomento

Nella figura 11, ad esempio, si nota come l'autore "AutoreB" ha esportato uno degli argomenti che gli erano stati assegnati ("Argomento Lin1"). Selezionandolo e premendo il pulsante *Complete*, questo verrà riportato al suo posto originale nella seconda colonna

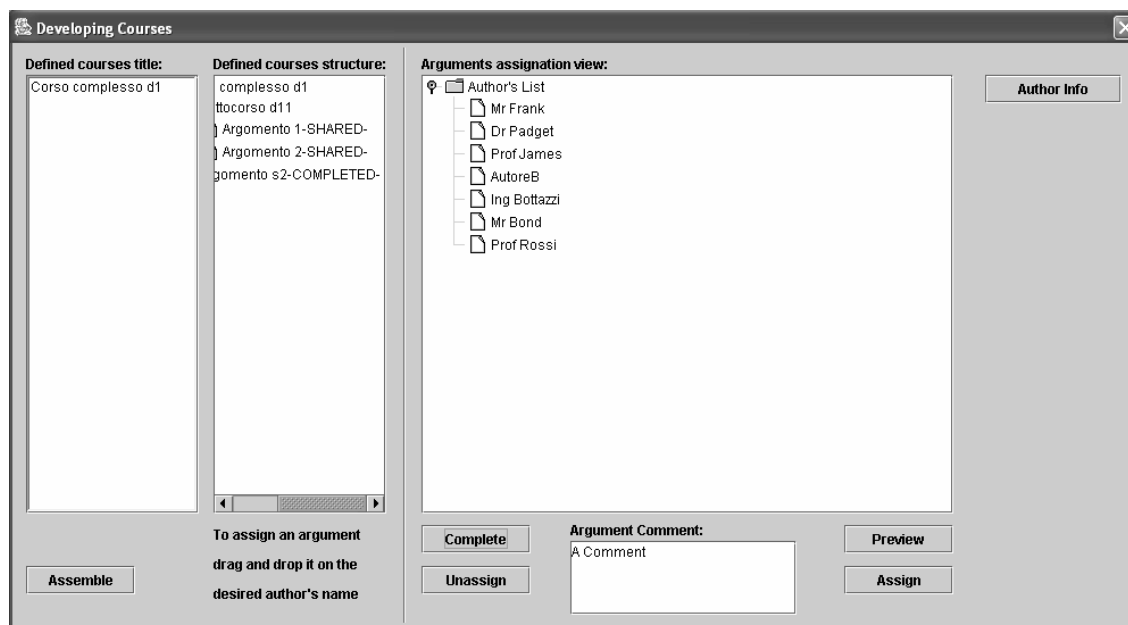


Figura 12: Completamento di un corso

In figura 12 è possibile vedere un corso ormai completato. Gli argomenti che lo compongono sono, come è possibile leggere dalla seconda colonna, tutti nello stato *Completed* (cioè appena esportati da un autore) o *Shared* (cioè già esportati in passato ed inseriti nel corso al momento della definizione scegliendo tra quelli condivisi). È dunque possibile selezionare l'opzione *Assemble* che permette il completamento definitivo del corso.

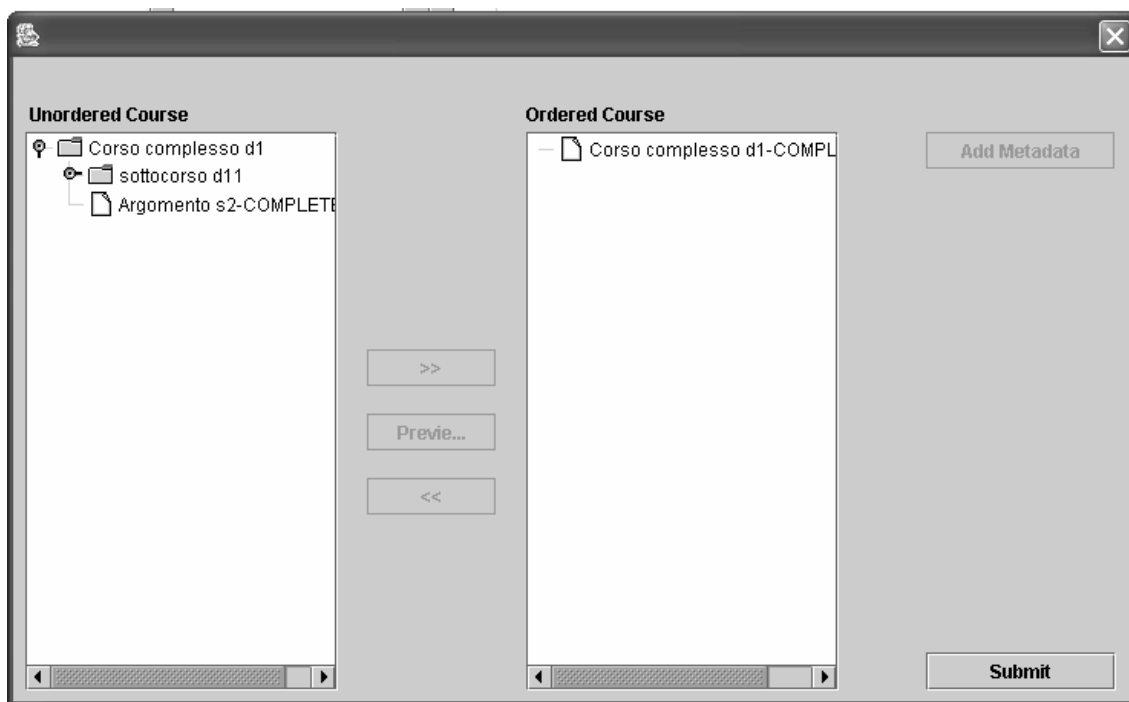


Figura 13: Assemblamento finale di un corso

A questo punto sarà possibile assemblare il corso a partire dagli argomenti che lo costituiscono; non è per forza necessario che gli argomenti vengano inseriti nel corso finale nello stesso ordine con cui sono stati definiti.

La prima colonna mostra tutti gli argomenti ed eventuali sottocorsi che costituiscono il corso; selezionando uno ad uno gli argomenti e premendo il pulsante >> questi verranno spostati verso la colonna di destra e ricostruiranno il corso definitivo.

Selezionando un argomento e premendo il pulsante *Preview* si potrà scorrere integralmente l'argomento all'interno del browser predefinito prima di inserirlo nel corso definitivo, mentre il pulsante << comporterà il ritorno dell'argomento selezionato dalla colonna di destra verso la colonna di sinistra.



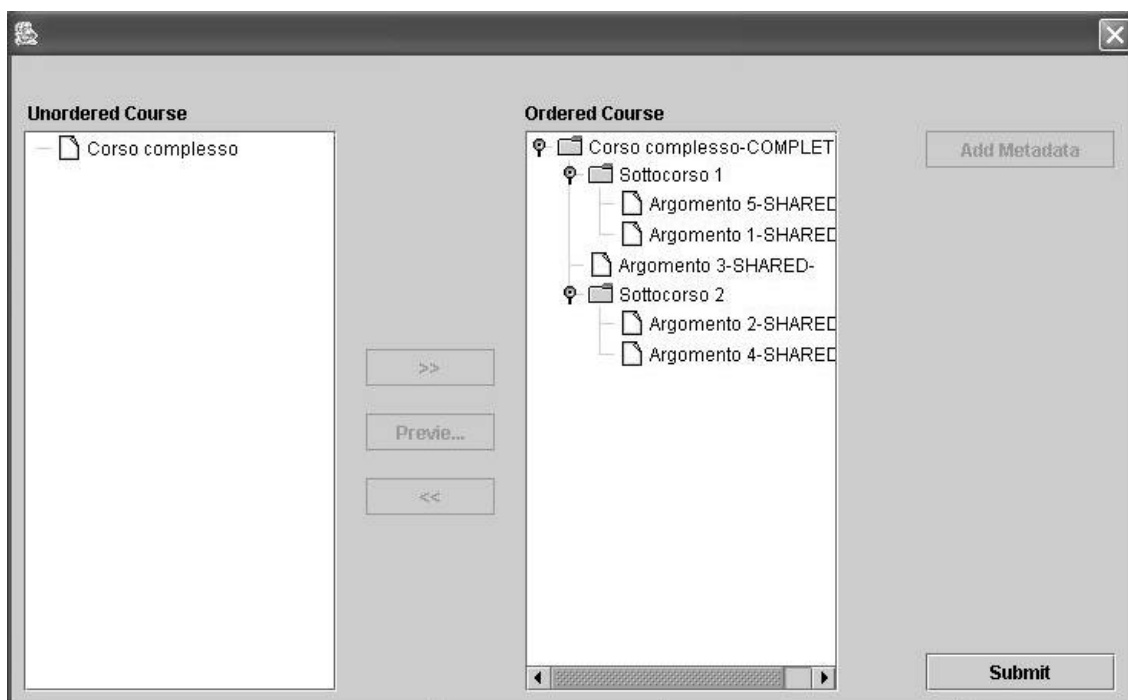


Figura 14: Assemblamento di un corso complesso

Si supponga di avere terminato la ricostruzione di un corso costituito da due sottocorsi (con due argomenti ciascuno) ed un argomento. Affinché l'assemblamento del corso sia davvero completo resta solo da aggiungere i metadati relativi a corsi e sottocorsi. Si dovranno, dunque, selezionare uno ad uno tutti i sottocorsi che costituiscono il corso e la radice dell'albero della colonna di destra, premendo per ognuno il pulsante *Add metadata*, che comporta l'apertura della finestra dei metadati riportata in figura 15.

The screenshot shows a software window titled "Technical" with a tabbed interface. The active tab is "General2". Below the tabs, there are four main sections for metadata entry:

- Title:** A dropdown menu set to "Italian", a text input field labeled "Resource title", and "Add" and "Remove" buttons.
- Catalog Entry:** A "catalog" dropdown, a text input field labeled "Catalog name", an "entry" dropdown set to "Italian", a text input field labeled "of the entry within the catalog", and "Add" and "Remove" buttons. Below this are "Add Catalog Entry" and "Remove Catalog Entry" buttons.
- Language:** A dropdown menu set to "Italian", an "Add" button, and a "Remove" button.
- Description:** A dropdown menu set to "Italian", a text input field labeled "Resource description", and "Add" and "Remove" buttons.

At the bottom of the window, there is a checkbox labeled "Show only mandatory items", and "Clear" and "Submit" buttons.

Figura 15: Inserimento dei metadati

La finestra è suddivisa in 12 pannelli, che ricoprono le 9 categorie di metadati di SCORM (si veda il capitolo SCORM) ed ogni pannello contiene una serie di aree di testo e pulsanti preposti all'inserimento dei metadati. Come noto dal capitolo SCORM non tutti i campi sono obbligatori, ed anche tra quelli obbligatori, ve ne sono alcuni non molto usati, specialmente da questa applicazione. La selezione dell'opzione *Show only mandatory items* permette, dunque, di limitare il lavoro di inserimento dei metadati ai soli pannelli "General" e "General2" (contenenti ad esempio i campi "Titolo", "Descrizione" e "keywords"); i rimanenti campi obbligatori verranno automaticamente riempiti mediante valori di default ("none") che permettono la produzione di un file xml di metadati valido e ben formato (si veda il capitolo XML) secondo lo standard SCORM.

Dopo avere inserito i metadati per tutti i sottocorsi e per il corso radice sarà possibile premere il pulsante *Submit* (figura 14) che causerà la produzione del corso definitivo, pronto per la fruizione tramite l'LMS.

## 2.5 Send Message

Questa funzionalità permette lo scambio di messaggi. Si tratta di una funzionalità molto utile, sebbene accessoria, perché facilita la collaborazione tra gli autori.

Ad esempio è possibile ad un autore chiedere chiarimenti all'Ecs circa i contenuti richiesti per un argomento che sta sviluppando; oppure un autore può chiedere ad un altro la produzione e condivisione di un'informazione atomica particolare; o, ancora, è possibile il semplice scambio di pareri e informazioni.

I messaggi che è possibile scambiare sono di due tipi: messaggi immediati, che vengono ricevuti subito dal destinatario, o emails.

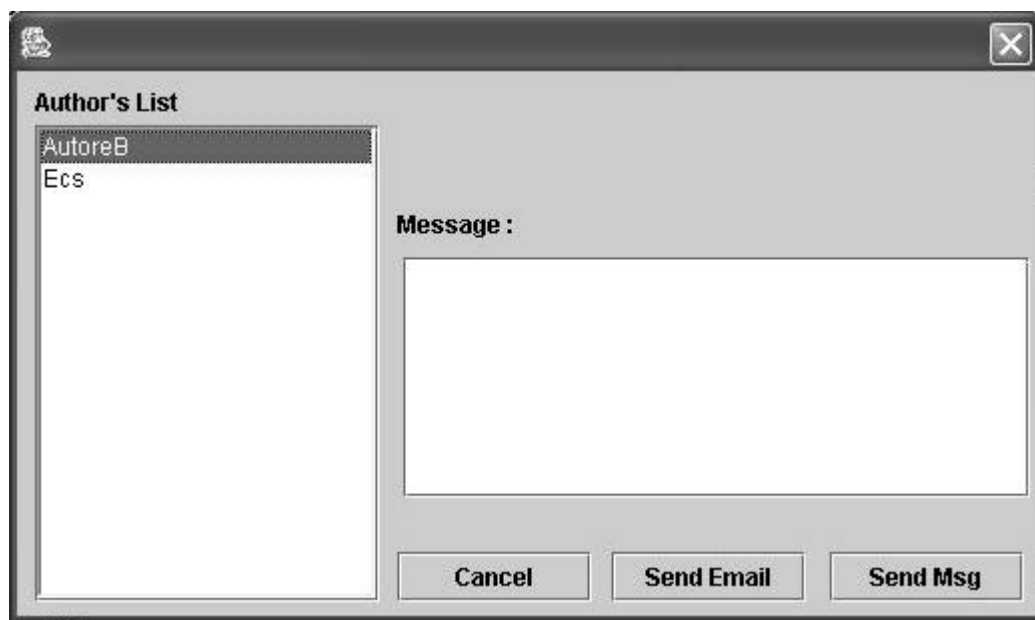


Figura 16: Invio di messaggi

Sulla sinistra, sotto *Author's List*, è possibile vedere la lista degli autori al momento online. Dopo aver inserito il testo del messaggio nell'area sotto *Message*, sarà possibile inviare il messaggio istantaneamente premendo il pulsante *Send Msg* o come email premendo il pulsante *Send Email*.

La pressione del pulsante *Cancel* permette, invece, di ripulire il testo del messaggio.

La ricezione viene notificata al destinatario tramite un'apposita finestra che permette di vedere subito il messaggio accedendo alla funzionalità *View Message*

## 2.6 View Message

Tale funzionalità è anche accessibile dal workbench, oltrechè dalla finestra di notifica di ricezione, e permette la visualizzazione dei messaggi ricevuti.

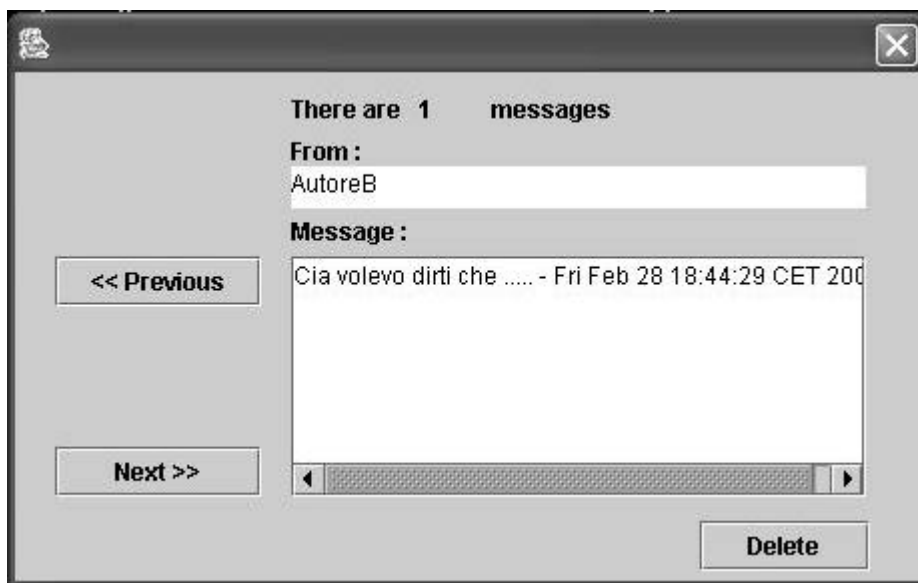


Figura 17: Lettura dei messaggi ricevuti

Tramite i pulsanti << *Previous* e *Next* >> è possibile scorrere la lista di messaggi, mentre il pulsante *Delete* permette la cancellazione dei messaggi. La data e ora del messaggio vengono automaticamente inseriti in coda allo stesso.

## 2.7 Settings

Questa funzionalità permette la modifica delle configurazioni fondamentali per l'invio dei messaggi di posta, ovvero l'alias del server di posta SMTP e l'eventuale username con cui si è registrati su esso.

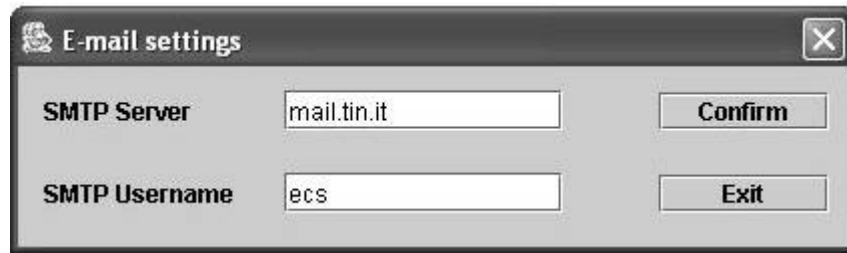


Figura 18: Setting dei parametri di posta in uscita

La correttezza di tali informazioni determina il buon funzionamento della funzionalità di invio di email.

### 3 Funzionalità dell'Autore

#### 3.1 Contracting Arguments

Tale funzionalità permette all'autore di visualizzare gli argomenti che gli sono stati assegnati dall'Ecs (tramite la funzionalità *Develop Course* – assegnazione) e decidere se accettarli o meno, ingaggiando eventualmente una sessione di contrattazione.

**Contracting Arguments**

**Argument:**

**Title:** argomento e1

**Description:** introduzione argomento e1

**Belonging Course:** corso di prova e

**Status:** contracting...

**Comment:**

A Comment

**Refuse**

**Accept**

Figura 19: Contrattazione degli argomenti assegnati

Quando l'ecs assegna ad un autore uno o più argomenti, viene visualizzata sul workbench di questo una notifica di assegnazione. L'autore potrà quindi accedere alla funzionalità di contrattazione mediante l'apertura della finestra in figura 19.

Sulla sinistra verrà mostrata una lista contenente i titoli degli argomenti assegnati. Selezionando uno tra questi titoli, verranno mostrati sugli spazi di testo sulla destra, oltre al titolo, la descrizione introduttiva inserita dall'ecs al momento della definizione, il nome del corso a cui questo appartiene e lo stato dell'argomento.

I due pulsanti sulla destra permettono di accettare o rifiutare l'assegnazione; in quest'ultimo caso l'autore potrà inserire un commento (le motivazioni per cui non accetta l'argomento ad esempio) nell'apposito spazio di testo sulla sinistra. L'accettazione o il rifiuto dell'argomento comportano la comparsa immediata di una notifica sul workbench dell'Ecs.

### 3.2 New Object

Mediante questa funzionalità l'autore può produrre e condividere immediatamente un'informazione atomica. Si tratta di una funzionalità molto comoda che permette agli autori di collaborare molto bene anche al di fuori del contesto di un corso.

Si pensi ad un autore che possiede una grande quantità di risorsa multimediali di un certo tipo, ad esempio filmati. Gli altri autori che sono a conoscenza di questo possono, tramite scambio di messaggi, richiedergli di produrre e rendere disponibili particolari filmati. L'autore in questione potrà, così, condividere le proprie risorse senza dover per forza trovarsi a sviluppare un argomento.

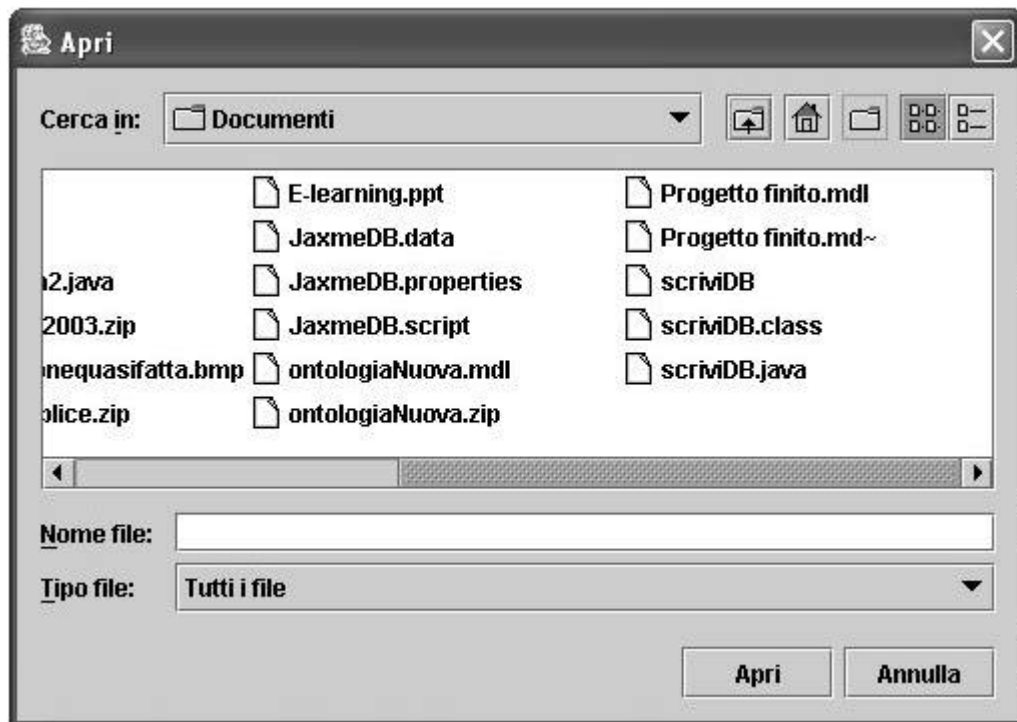


Figura 20: Scelta del file relativo ad un'informazione atomica

Dopo aver scelto di produrre una nuova informazione atomica verrà aperta la finestra in figura 20 che permette la scelta del file da condividere.

Una volta scelto il file si dovranno inserirne i metadati mediante la finestra di figura 15. La pressione del pulsante *Submit* di tale finestra comporta l'invio

immediato del file (accompagnato dai metadati) mediante la procedura descritta nel paragrafo *Invio dei files*.

### 3.3 Developing Arguments

Si tratta della funzionalità principale del tool e permette il vero e proprio sviluppo degli argomenti mediante l'inserimento di pagine *web* a contenuto multimediale.

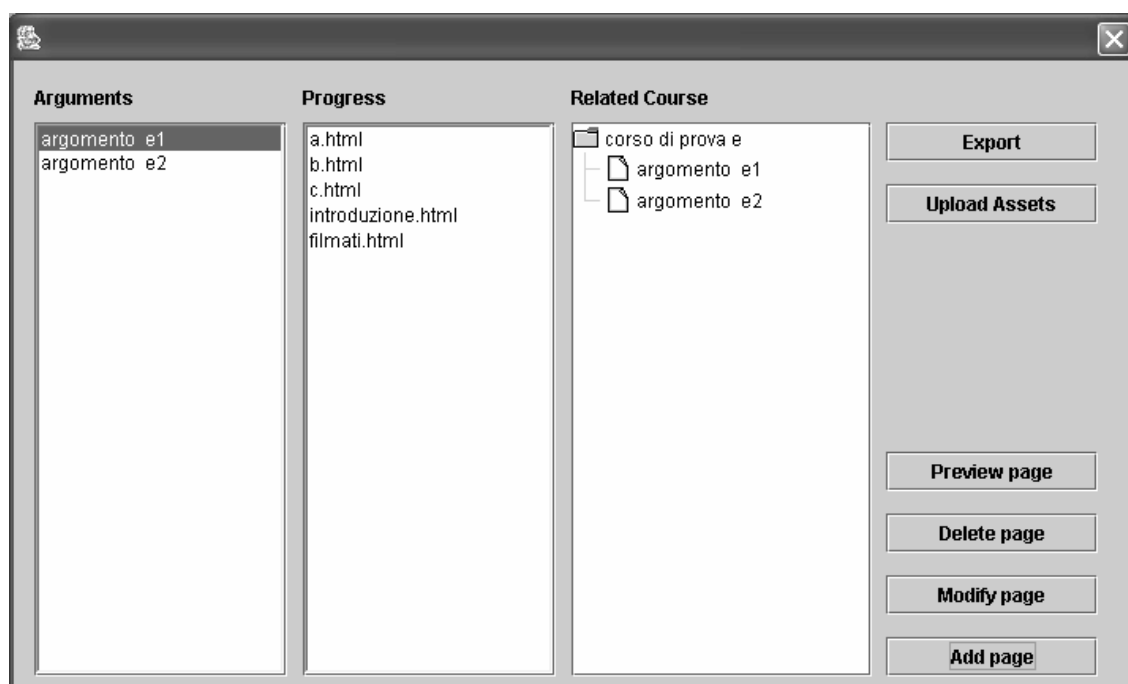


Figura 21: Sviluppo di argomenti

La finestra di lavoro è suddivisa in quattro parti: a partire da destra, tre colonne ed una zona di pulsanti.

La prima colonna (*Developing Arguments*) mostra la lista di titoli relativi agli argomenti che l'autore sta sviluppando e non ha ancora terminato.

La seconda colonna (*Progress*) mostra i titoli delle pagine *web* già prodotte per l'argomento selezionato dalla colonna di sinistra.

La terza colonna (*Related Course*) mostra la struttura ad albero del corso di cui fa parte l'argomento selezionato nella prima colonna.

I pulsanti che si trovano all'estrema destra della finestra sono divisi in due gruppi: in alto i pulsanti per il completamento dell'argomento e la condivisione



delle sue informazioni atomiche, in basso quelli necessari per la produzione delle pagine.

Partendo dal basso, il pulsante *Add page* permette l'aggiunta di una pagina *html* all'argomento selezionato mediante l'apertura dell'editor html di tipo *wysiwyg* (what you see is what you get).

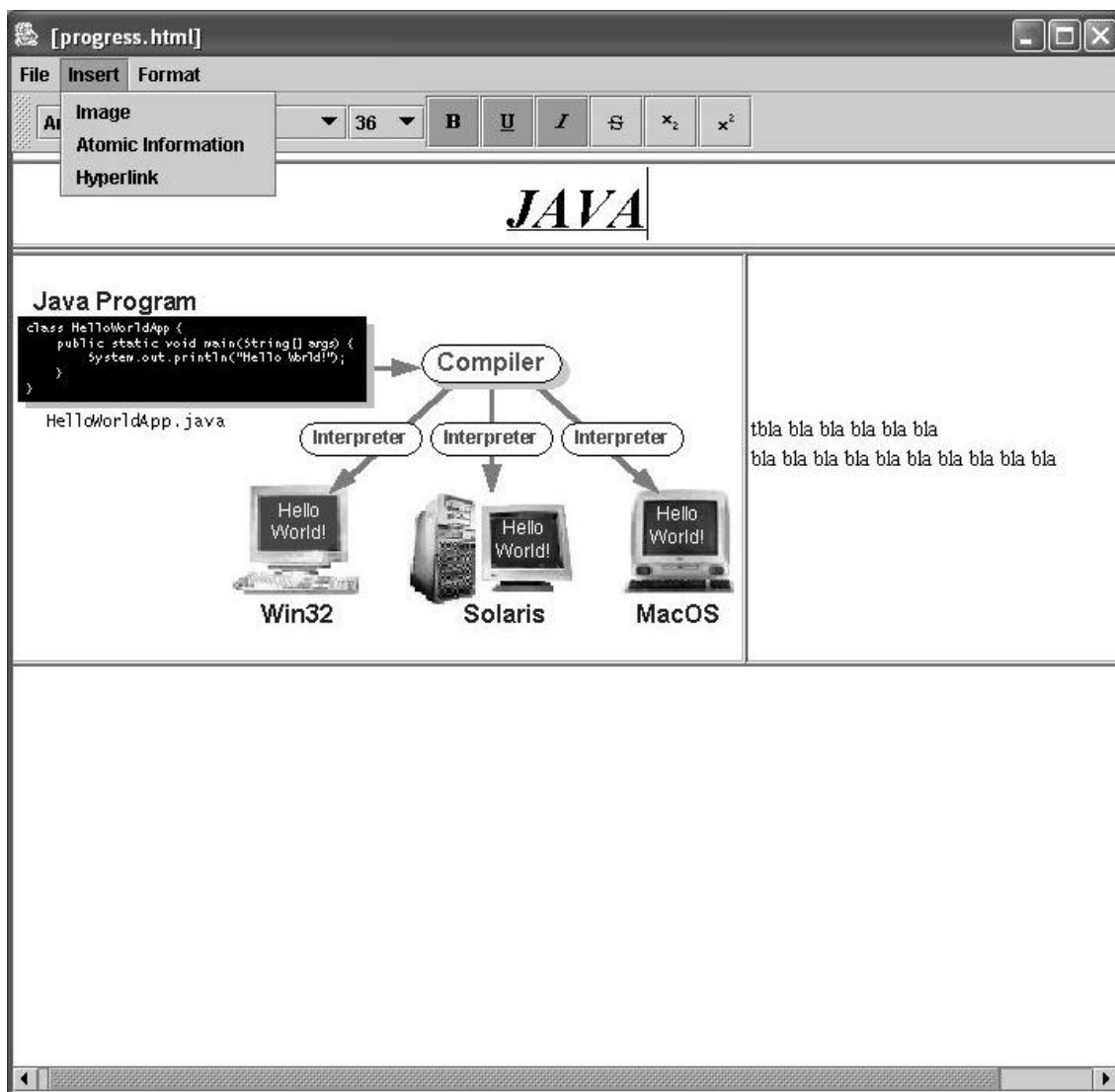


Figura 22: L'editor html

L'editor permette tutte le funzionalità base utili alla produzione di pagine a contenuto multimediale: apertura e salvataggio delle pagine, creazione di nuove pagine a partire da un layout predefinito, inserimento di immagini, inserimento di link ipertestuali, inserimento di risorse atomiche, formattazione del testo (allineamento, colore del testo, colore di background, font).

Si analizzeranno in seguito le caratteristiche più interessanti tra queste. L'inserimento di un layout predefinito facilita la produzione di pagine complesse tramite la scelta da un elenco fornito all'autore.

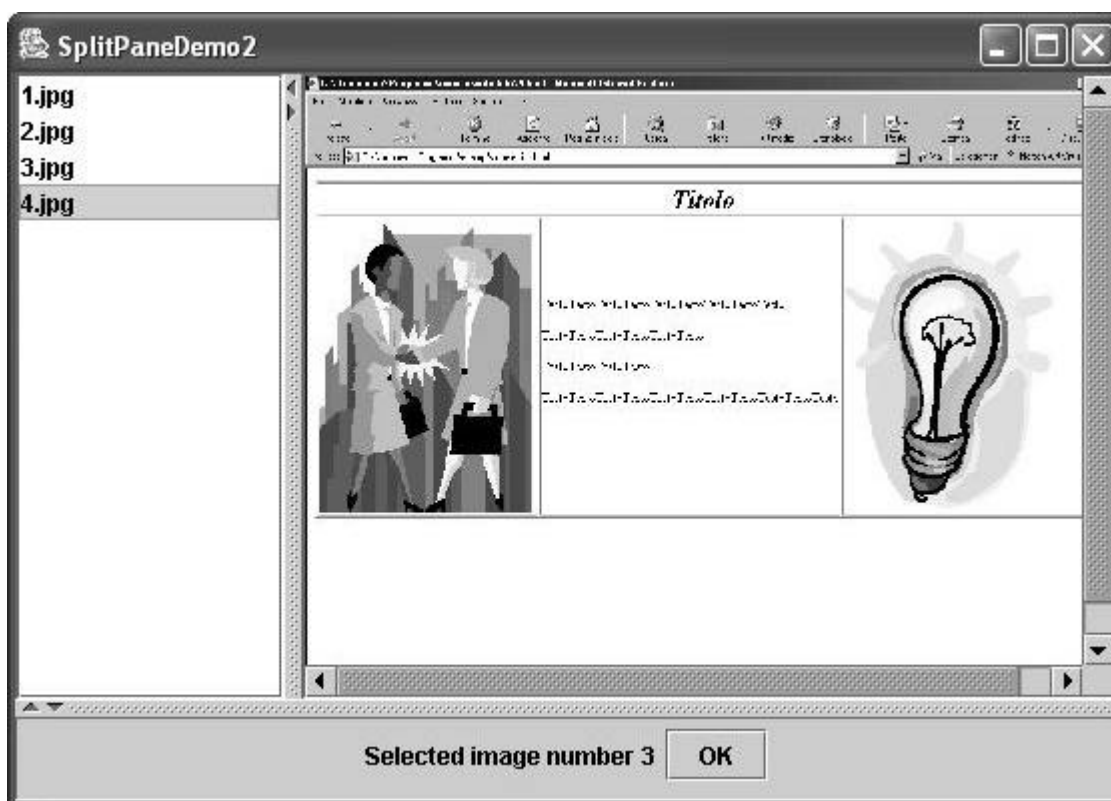


Figura 23: Scelta del layout della pagina

Scegliendo dal menù *File* l'opzione *New* viene richiesta la conferma di voler rinunciare al lavoro corrente; se si conferma tale scelta viene mostrata la finestra di figura 23, che propone sulla sinistra un elenco di nomi di immagini, che sono gli *snapshot* dei layout disponibili. Selezionando uno di questi files viene mostrata sul frame principale della finestra il layout corrispondente. Ingrandendo la finestra il layout verrà ingrandito permettendo una migliore visualizzazione. Dopo aver scelto il layout per la pagina, questo verrà accettato premendo il pulsante *Ok*.

Selezionando il menù *Insert* e l'opzione *Atomic Information* si potrà produrre ed inserire un'informazione atomica.



Figura 24: Inserimento di un'informazione atomica

Verrà proposta una finestra che permette di scegliere se produrre in proprio la risorsa da condividere o inserirla scegliendo dall'insieme di risorse condivise.

La pressione del pulsante *New* comporta l'inizio di una procedura identica a quella descritta sopra nel paragrafo relativo alla funzionalità *New Atomic Information*.

Premendo *Search* verrà aperta una finestra che permette la ricerca all'interno dell'archivio di risorse condivise.

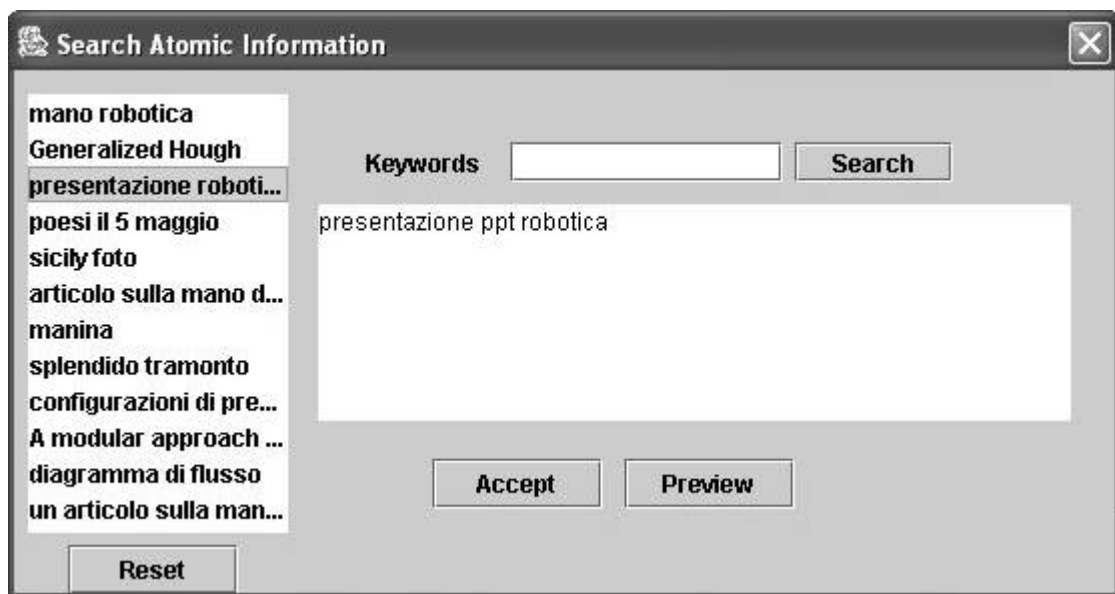


Figura 25: Ricerca di un'informazione atomica dall'archivio

Il funzionamento della finestra di ricerca è identico a quello della ricerca di argomenti condivisi descritto sopra nella sezione dedicata alla funzionalità *Define Course*.

Tornando alla figura 21 il successivo pulsante, partendo dal basso, è *Modify Page*; la pressione di esso comporta l'apertura del file html selezionato dalla colonna centrale che ne consente la modifica.

Il pulsante *Delete Page*, consente, previa conferma, la cancellazione della pagina selezionata.

Il pulsante *Preview Page* permette invece di visualizzare una preview della pagina selezionata all'interno del browser predefinito. Questa operazione facilita notevolmente le operazioni di produzione degli argomenti.

Il pulsante *Upload Assets* permette di condividere tutte gli oggetti condivisibili (le informazioni atomiche) presenti all'interno di un argomento, a prescindere dal fatto che questo sia stato completato o meno. L'operazione di condivisione delle risorse utilizzate viene eseguita automaticamente quando l'argomento, già completato, viene definitivamente esportato. Se l'autore in questione fosse molto produttivo ed avesse inserito molti oggetti in un argomento, potrebbe verificarsi un accumulo di risorse condivisibili molto interessanti che potrebbero essere utili agli altri autori; in questo caso l'autore condividerebbe gli oggetti subito.

Mediante il pulsante *Export* l'argomento selezionato verrà esportato; tale opzione viene scelta dall'autore quando questi crede di avere completato l'argomento.

La pressione di tale pulsante comporta l'apertura della seguente finestra:

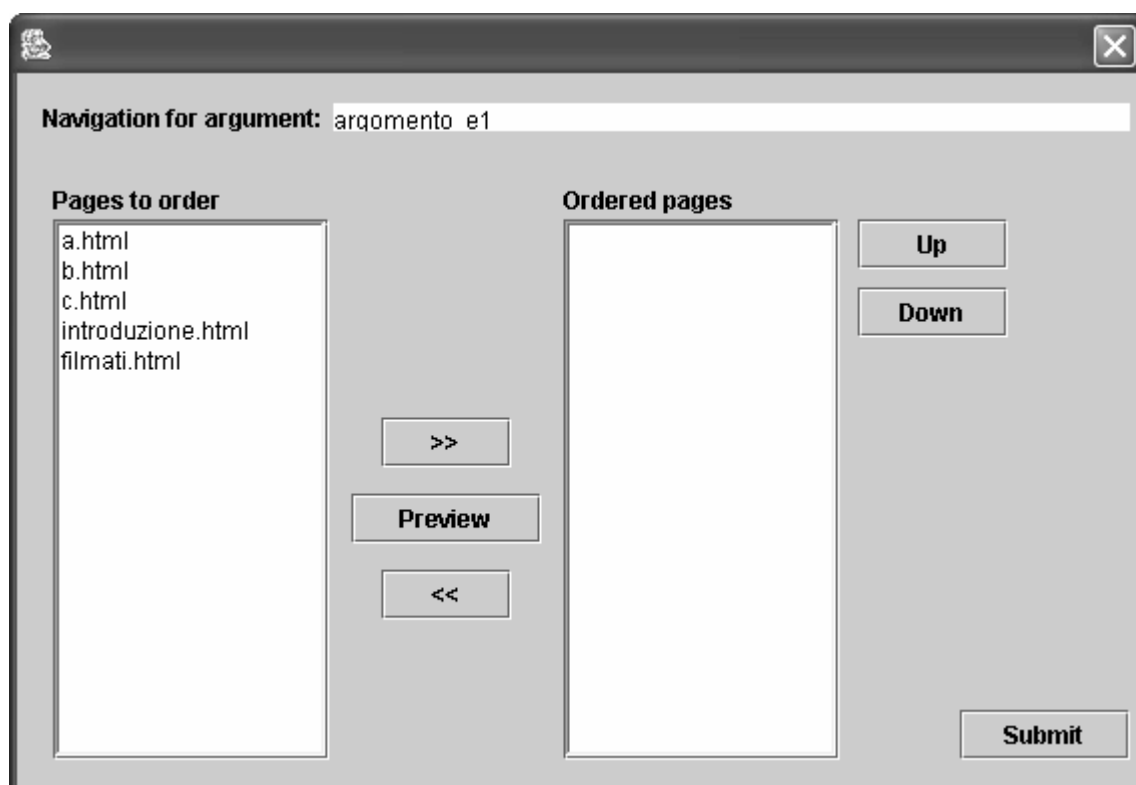


Figura 26: Completamento di un argomento 1

La lista di sinistra presenta tutte le pagine html prodotte per l'argomento il cui titolo è scritto in alto.

La finestra permette di decidere l'ordine con cui le pagine verranno presentate all'utente quando questi fruirà l'argomento.

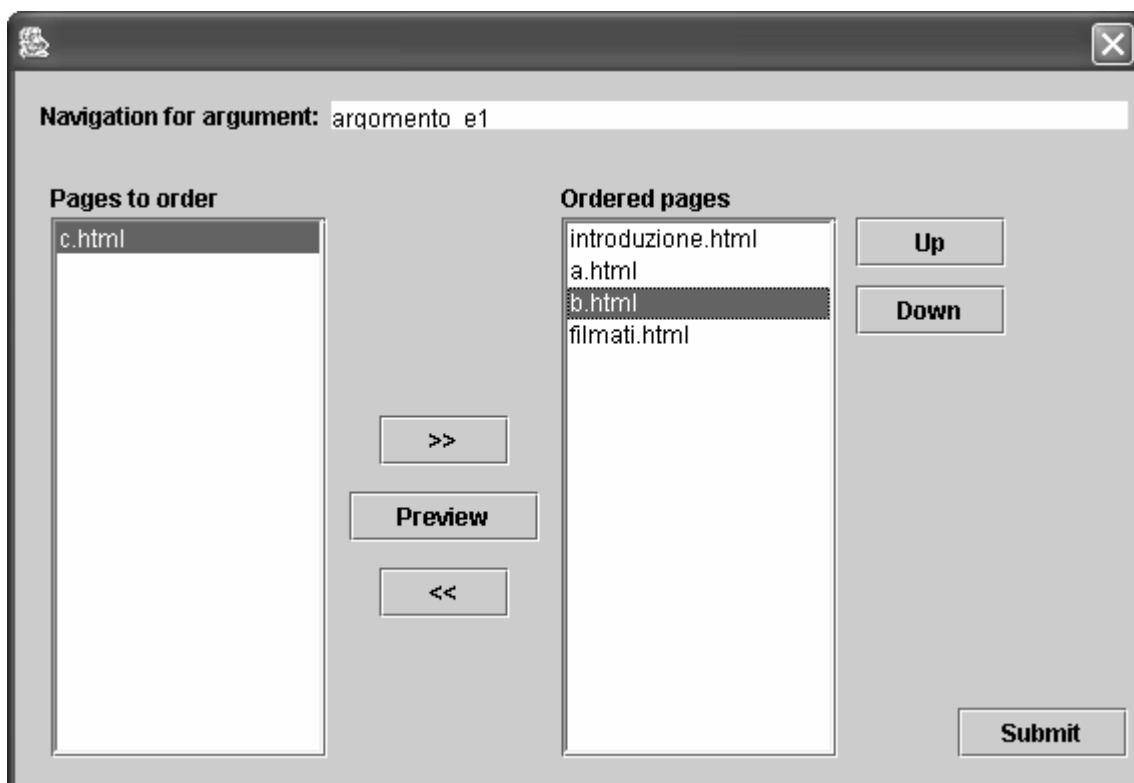


Figura 27: Completamento di un argomento 2

I pulsanti *>>* e *<<* causano il trasferimento delle pagine dalla lista di sinistra a quella di destra e viceversa. La pressione del pulsante *Preview* consente, invece, la visualizzazione della pagina selezionata dalla lista di sinistra all'interno del Browser predefinito, onde facilitare la scelta dell'autore.

I pulsanti *Up* e *Down* sulla destra, invece, permettono di spostare verso l'alto o verso il basso la pagina selezionata dalla lista di destra.

Una volta che tutte le pagine sono state inserite nell'ordine desiderato, premendo il pulsante *Submit* l'argomento verrà definitivamente inviato all'Ecs che riceverà una notifica di completamento di un argomento e potrà visualizzarne lo stato *Completed* nella finestra *Developing Courses* in figura 11.

### 3.4 Send message e View Message

Queste due funzionalità sono fornite all'utente allo stesso modo che all'Ecs. Si rimanda dunque alla descrizione già fatta alcuni paragrafi sopra.

### 3.5 Settings

Mediante questa funzionalità l'autore può modificare alcuni parametri del sistema; si può, infatti, modificare l'alias del server smtp utilizzato per l'invio delle email (per la descrizione si rimanda al paragrafo già descritto sopra) e modificare i propri dati personali.

Selezionando infatti l'opzione *VCard* del menù *Settings* del workbench, verrà mostrata una finestra come quella di figura 4 (alcune pagine sopra) contenente i dati al momento inseriti nel sistema per l'autore.

Si potranno, dunque, modificare i dati che sono cambiati o che per qualunque ragione non vadano più bene.

## **Appendice**



## 1 Materiale fornito

Il seguente materiale viene fornito insieme a questa documentazione: un CD-ROM contenente:

- DataBaseRisorse.mdb: file Microsoft Access contenente il DB delle risorse del tool
- DataBaseUtenti.mdb: file Microsoft Access contenente il DB degli utenti
- ProgettoAuthoringTool.mdl: file Rational Rose contenente il progetto del sistema
- PianificazioneAuthoringTool.mpp: file Microsoft Project relativo alla pianificazione del progetto
- DocumentazioneAuthoringTool.doc: file Microsoft Word contenente tutta la documentazione qui presentata
- DocumentazioneAuthoringTool.pdf: file Adobe Acrobat contenente tutta la documentazione qui presentata
- File \*.java: codice sorgente
- Programma.zip: archivio contenente:
  - File \*.class: file java compilati costituenti l'applicazione
  - File \*.gif, \*.jpg: immagini utilizzate dal tool

### 2.1 Istruzioni per l'installazione

Per poter installare correttamente il tool è necessario seguire i seguenti passi:

1. sulla macchina dell'ecs copiare il file DataBaseRisorse.mdb ed il file DataBaseUtenti.mdb dal CD-ROM in una cartella desiderata e creare i driver di connessione ad essi secondo la procedura illustrata in seguito. In tale procedura, quando verrà chiesto, inserire come “nome origine dati” relativamente: dataBaseAssets e UsersDB
2. su ogni macchina del sistema estrarre l'archivio Programma.zip in una cartella a propria scelta

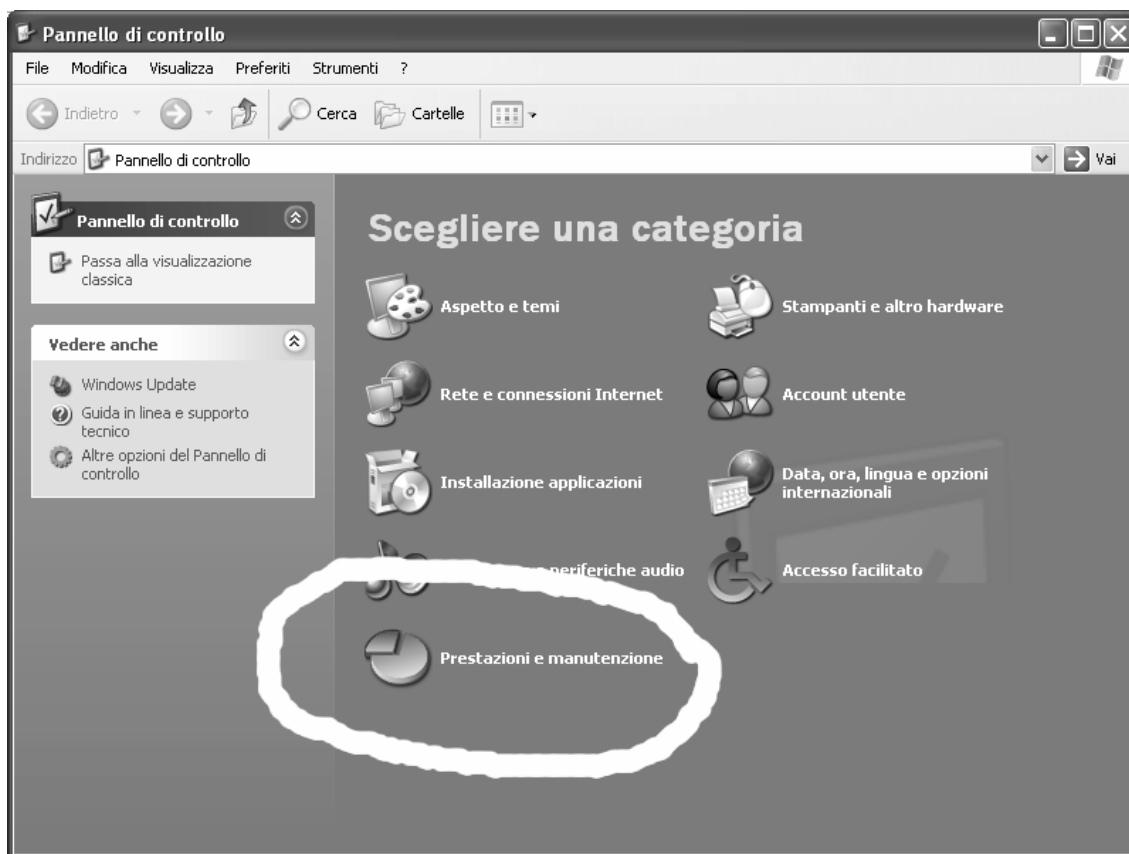
3. su ogni macchina del sistema inserire la cartella appena creata all'interno del classpath di java
4. sulla macchina dedicata all'ecs, creare un file di testo al cui interno scrivere:

```
java jade.Boot -gui
AgenteGui:Tesi.progetto.AgenteGui
AgenteAccessManager:Tesi.progetto.AgenteAccessManager
AgenteEditor:Tesi.progetto.AgenteEditor
AgentePlanner:Tesi.progetto.AgentePlanner
Ecs:Tesi.progetto.AgenteCommonGui
```
5. rinominare il file di testo appena creato come Server.bat
6. su ognuna delle macchine dedicate agli autori seguire la seguente procedura:
  - a. si supponga di essere sulla macchina di *autoreA*; creare un file di testo il cui contenuto sia:

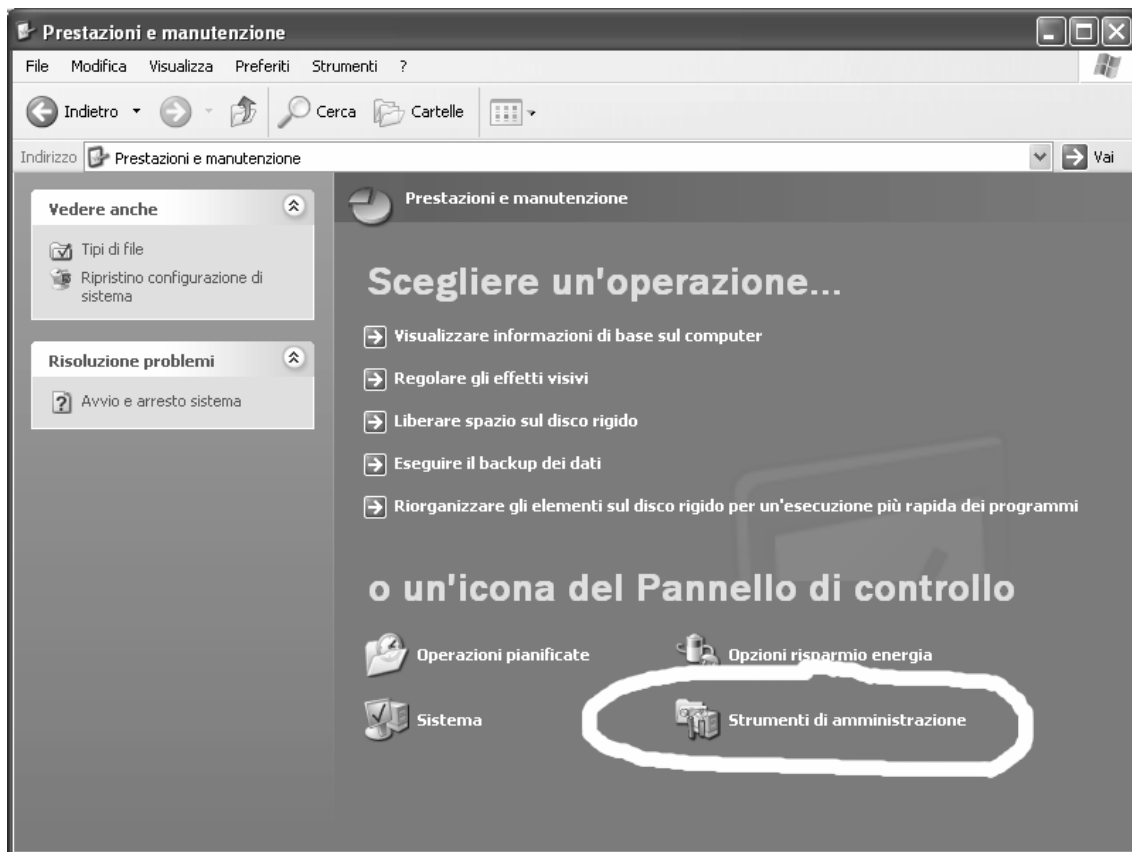
```
java jade.Boot -container -host nomeserver
AutoreA:Tesi.progetto.AgenteCommonGui
AutoreA2:Tesi.progetto.AgenteAuthorGui
```

sostituendo a *nomeserver* l'alias di rete della macchina che ospita l'ecs o il suo indirizzo IP
  - b. rinominare il file come Autore.bat

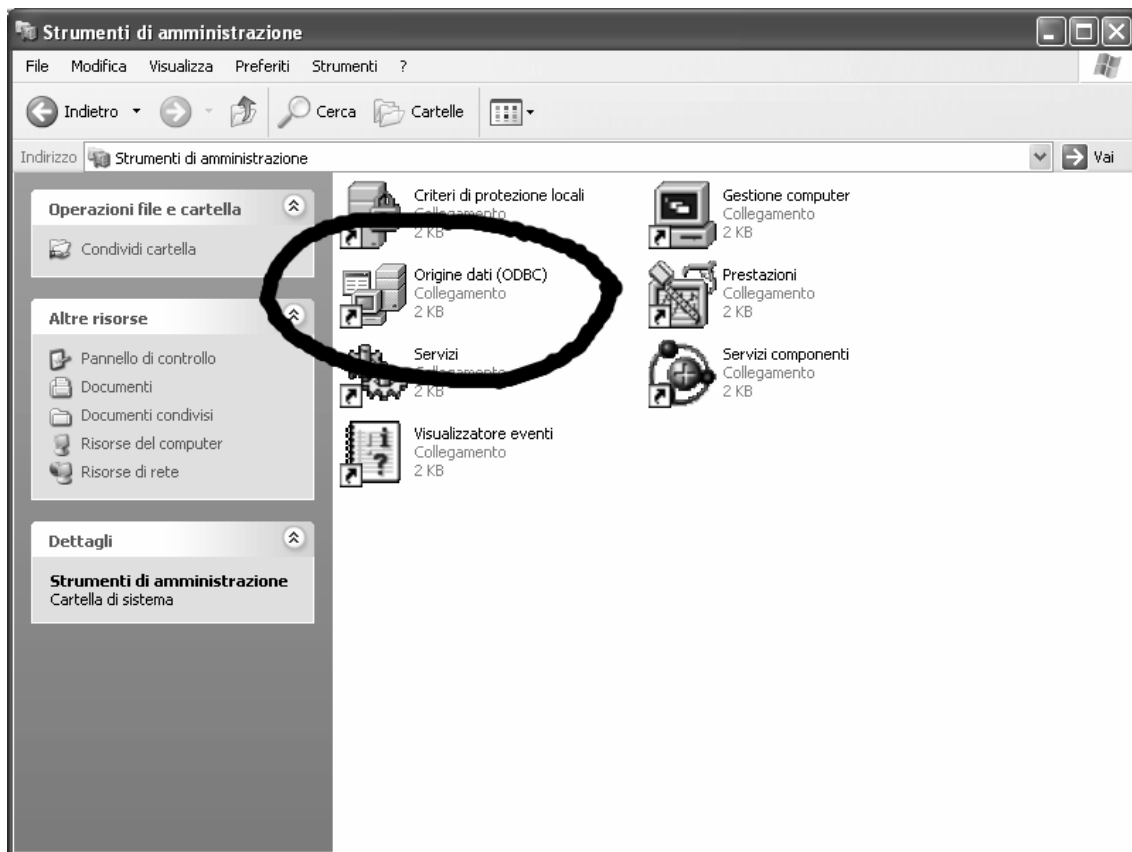
## 2.2 Installazione dei driver del DB



1. Selezionare dal pannello di controllo la voce “Prestazione e manutenzione”.



2) Cliccare su “Strumenti di amministrazione”.



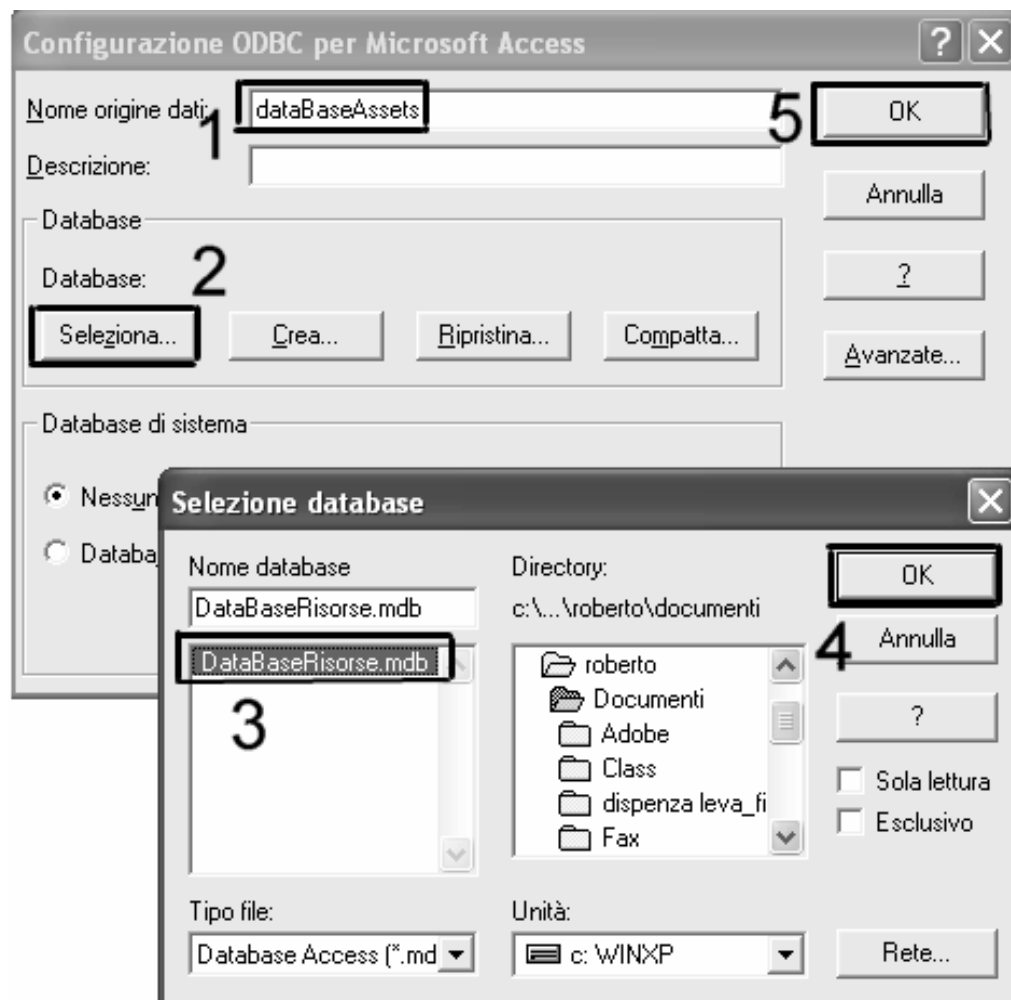
3) Selezionare “Origine dati (ODBC)”.



4) Cliccare su “DSN di Sistema” e poi su ”Aggiungi” per creare un nuovo driver di connessione.



5) Selezionare “Microsoft Access Driver (\*.mdb)” dalla liste e cliccare su “Fine”.



6) Inserire il nome del driver “dataBaseAssets” (passo 1), cliccare su “Seleziona...” (passo 2), selezionare il file Access “DataBaseRisorse” dalla cartella in cui è stato copiato (passo 3) e cliccare su “OK” (passo 4 e 5).

7) Nella finestra successiva premere OK

### 3 Statistiche

Di seguito sono riassunte le statistiche che riassumono la mole di lavoro affrontata:

- Files Java generati: 104
- Spazio occupato su disco dai sorgenti: 2.740.901 byte
- Righe di codice: 32156
- Classi: 149