

SUPSI

CAM - Centralized Access Map

Studente/i

Bonomi Niko

Relatore

Landolfi Giuseppe

Correlatore

-

Committente

**Ente Ospedaliero Cantonale
Bettosini Fabio**

Corso di laurea

Ingegneria Informatica

Modulo

M00002P Progetto di diploma

Anno

2018

Data

31 agosto 2018

STUDENTSUPSI

Indice

Abstract	1
Progetto assegnato	3
1 Analisi dei requisiti	5
1.1 Requisiti funzionali	5
1.1.1 Mappatura	5
1.1.2 Systems Integration	7
1.1.3 Gestione della mappa	8
1.1.4 API	8
1.2 Requisiti non funzionali	9
1.2.1 Tecnologie utilizzate	9
1.2.2 Sicurezza	10
2 Design del sistema	11
2.1 Architettura	11
2.2 Data Model	12
2.2.1 Interfacce	14
2.3 Plugin GUI	17
2.3.1 Lettura	17
2.3.2 Modifica/Inserimento	17
2.4 CAM	19
2.4.1 Property	19
2.4.2 Plugin	20
2.4.3 Legacy System Manager	22
2.5 Funzionalità del sistema	24
2.5.1 Mappatura	24
2.5.2 Gestione	24
2.5.3 API	26

3	Sistemi integrati	27
3.1	Software Risorse Umane	27
3.2	Legacy Systems	28
3.2.1	Active Directory	28
3.2.2	Polypoint	30
3.2.3	xLnet3	31
3.2.4	XpertLine	33
3.2.5	xLeoc	34
3.2.6	WinScribe	35
3.2.7	AS400	38
3.2.8	Service Desk Manager	40
3.2.9	LAB400	45
3.2.10	GECO	48
3.2.11	Opale	53
3.2.12	ViewPoint	53
3.2.13	Isteric	54
3.2.14	Cartella Pazienti Informatizzata	55
3.2.15	CARL Source	56
4	Interfaccia WEB	61
4.1	Login	61
4.2	Barra di navigazione	62
4.3	Home	63
4.4	Ricerca	63
4.4.1	Filtri	63
4.4.2	Risultato	65
4.5	Amministrazione	66
4.5.1	Annunci	67
4.5.2	Ruoli	68
4.5.3	Utenti	72
4.5.4	Gestione dei Plugin	75
4.5.5	Impostazioni	79
5	Application Programming Interface	83
5.1	Autenticazione	83
5.2	Risposta	84
5.3	Plugin	85
5.4	Utenti	85
5.5	Ruoli	87
5.6	Annunci	87

6 Conclusioni e sviluppi futuri	89
6.1 Sviluppi futuri	90
6.1.1 Integrazione sistemi aggiuntivi	90
6.1.2 Funzionalità aggiuntive sui plugin esistenti	90
6.1.3 Comparazione più user friendly	90
6.1.4 Documentazione sviluppo plugin	90
6.1.5 Tracciabilità modifiche	91
6.1.6 Messa in produzione	91

Elenco delle figure

1.1	Schema mappatura	7
2.1	Schema architettura	12
2.2	Right DataModel	13
2.3	User DataModel	14
2.4	Interfacce	15
2.5	Diagramma di sequenza modifica mappa	19
2.6	Data model Property	20
2.7	Data model Right	20
2.8	Legacy System DataModel	22
2.9	Diagramma di sequenza mappatura	24
3.1	Active Directory Data Model	30
3.2	xLnet3 ER	32
3.3	XLnet3 DataModel	32
3.4	XpertLine accessi	33
3.5	xLeoc accessi	34
3.6	WinScribe ER	36
3.7	WinScribe DataModel	37
3.8	AS400 DataModel	39
3.9	SDM ER	40
3.10	SDM DataModel	43
3.11	LAB400 ER	46
3.12	LAB400 DataModel	46
3.13	GECO ER	48
3.14	GECO DataModel	51
3.15	IstERIC ER	54
3.16	IstERIC DataModel	54
3.17	CPI ER	55
3.18	CPI DataModel	56
3.19	CARL Source ER	57

3.20 CARL Source DataModel	58
4.1 Login	61
4.2 Barra di navigazione	62
4.3 Menu utente	62
4.4 Home	63
4.5 Ricerca	64
4.6 Dettaglio ricerca	65
4.7 Calendario	65
4.8 Risultato ricerca - sistema	66
4.9 Amministrazione - News	67
4.10 Amministrazione - News	67
4.11 Amministrazione - Dettaglio news	68
4.12 Amministrazione - Ruoli	68
4.13 Amministrazione - Dettaglio ruolo	69
4.14 Amministrazione - Modale mappatura ruolo	70
4.15 Amministrazione - Sistema mappato al ruolo	71
4.16 Amministrazione - Modifica Sistema mappato al ruolo	71
4.17 Amministrazione - Utenti	72
4.18 Amministrazione - Utente	73
4.19 Amministrazione - Utente comparazione diritti	75
4.20 Amministrazione - Icona plugin risorse umane	75
4.21 Amministrazione - Plugins	76
4.22 Amministrazione - Nuovo Plugin	77
4.23 Amministrazione - Dettaglio Plugin	77
4.24 Amministrazione - Aggiunta nuova versione al plugin	78
4.25 Amministrazione - Selezione plugin per il supporto alle risorse umane	78
4.26 Amministrazione - Impostazioni	79
4.27 Amministrazione - Generatore mappa	80
4.28 Amministrazione - Livelli di accesso	81
4.29 Amministrazione - Active Directory	82
5.1 UML APIResult	84

Elenco delle tabelle

2.1	Descrizione metodi LegacySystem	16
2.2	Descrizione metodi HRLegacySystem	16
2.3	Interfaccia script.js	18
2.4	Campi classe PluginLS	21
2.5	Campi classe PluginLSVersion	21
2.6	Campi classe LegacySystemManager	22
2.7	Campi classe LSContainer	23
3.1	Parametri API	28
3.2	Campi classe ADUser	30
3.3	Campi classe XLnet3User	33
3.4	Campi classe XLnet3Unit	33
3.5	Campi classe WinScribeUser	37
3.6	Campi classe WinScribeJob	38
3.7	Campi classe WinScribeDepartment	38
3.8	Esempio contenuto MENU00F	39
3.9	Campi classe AS400User	39
3.10	Campi classe WinScribeJob	40
3.11	Campi classe Link	44
3.12	Campi classe Contact	44
3.13	Campi classe Type	44
3.14	Campi classe AccessType	45
3.15	Campi classe RestAccess	45
3.16	Descrizione campi	46
3.17	Campi classe LAB400User	47
3.18	Campi classe Profile	51
3.19	Campi classe GecoUser	52
3.20	Campi classe GecoUserComplete	52
3.21	Campi classe Scope	52
3.22	Campi classe ScopeComplete	52

3.23 Campi classe Sermed	53
3.24 Campi classe Unit	53
3.25 Campi classe IstericUser	55
3.26 Campi classe CPIUser	56
3.27 Campi classe CPIUserCode	56
3.28 Campi classe CarlUser	58
3.29 Campi classe CarlSite	59
3.30 Campi classe CarlProfile	59
4.1 Diritti sulle pagine GUI	62
4.2 Icone sistema	66
4.3 Sottosezioni amministrazione	67
4.4 Icone lista ruoli	69
4.5 Mappatura icone - bottoni	74
5.1 API - Plugin	85
5.2 API - Utenti	86
5.3 API - Ruoli	87
5.4 API - Annunci	87

Abstract

Il problema principale dell'Area ICT presso l'Ente Ospedaliero Cantonale è la gestione utenti. Quando un nuovo collaboratore inizia a lavorare, il suo utente informatico viene creato manualmente dagli addetti al supporto all'utenza.

In base alla funzione/ruolo lavorativo è necessario abilitare il suddetto utente a software specifici per permettergli di svolgere le proprie mansioni lavorative.

Attualmente alcuni di questi software sono gestiti manualmente da collaboratori esterni all'Area ICT, ovvero delle persone di riferimento esterne all'informatica, il cui compito è quello di creare gli utenti e assegnarne i diritti nei software sotto la loro gestione in base alla funzione dell'utente.

I responsabili degli applicativi, vengono notificati direttamente dai collaboratori del service desk¹, quest'ultimi selezionano quali software sono necessari per il ruolo lavorativo del nuovo utente, questo in base alla loro esperienza e alla similitudine con quanto è stato assegnato in precedenza ad altri collaboratori con la stessa funzione. I responsabili hanno anche altre mansioni da svolgere durante l'arco della giornata, e questo spesso porta a grandi ritardi sull'abilitazione degli utenti con conseguente malumore nei confronti dell'ICT.

Lo scopo di questo progetto è quello di eliminare il problema partendo dall'origine. Il prodotto ha lo scopo di realizzare una mappatura di tutti i ruoli presenti in azienda e le varie tipologie di accessi su quei software la cui gestione viene eseguita manualmente.

In questo modo, quando un nuovo collaboratore inizierà a lavorare presso l'EOC² si potrà sapere immediatamente i software necessari i quali gli verranno assegnati automaticamente senza alcun intervento umano.

Questo sgraverà moltissimo il service desk, facendo guadagnare loro due risorse costantemente occupate alla gestione utenti, che potranno dedicarsi a tempo pieno alle altre attività che gli competono.

¹Nome utilizzato in azienda per descrivere gli addetti al supporto all'utenza

²Ente Ospedaliero Cantonale

Progetto assegnato

Descrizione

Nel contesto aziendale la gestione accessi dei collaboratori per software specifici è problematica. Purtroppo, spesso ci sono situazioni in cui all'utente non vengono assegnati tutti i diritti necessari per il lavoro.

Attualmente in azienda la creazione e la modifica di utenti, viene fatta in modo manuale da un collaboratore incaricato di eseguire una creazione basilare: casella di posta elettronica, utente Active Directory. Quest'ultimo deve inoltre selezionare quali programmi saranno necessari per supportare l'utente nello svolgere le sue mansioni lavorative. I diritti interni a questi software vengono assegnati dal responsabile dell'applicativo.

Questo flusso manuale spesso comporta ritardi nell'assegnazione dei permessi, oppure la mancata assegnazione di essi, ad esempio nel caso in cui il collaboratore sbaglia nella selezione degli applicativi richiesti dalla funzione lavorativa dell'utente.

Compiti

Il problema del flusso descritto in precedenza è dunque relativo alla mancanza di una chiara direttiva che metta in relazione la funzione specificata dalle risorse umane con gli accessi degli applicativi. Si chiede dunque di sviluppare una mappatura di tutte le funzioni delle RU in funzione dei diritti sui vari software, questo per facilitare la creazione di nuove utenze automatizzando l'assegnazione degli accessi. La mappa dovrà essere inizialmente popolata in modo automatico sulla base della situazione attualmente presente in azienda.

Obiettivi

Gli obiettivi del progetto sono i seguenti:

- Creazione di una struttura dati che possa contenere la mappa, permettendone una gestione rapida ed efficace.

- Creazione di un automatismo “reverse engineering” che popoli la mappa in funzione della situazione attualmente presente in azienda.
- Validazione dell'automatismo con il personale addetto alla creazione di utenti.
- Creazione di un'applicazione WEB che permetta la gestione da parte degli amministratori di sistema e renda disponibile a tutti la visione della situazione standard.
- Creazione di un servizio WEB che permetta in futuro l'integrazione con l'attuale software di creazione utenti
- Stesura di una documentazione completa.

Capitolo 1

Analisi dei requisiti

Il capitolo 1 ha lo scopo di illustrare l'analisi eseguita sui requisiti di progetto, descrivendone le componenti principali.

1.1 Requisiti funzionali

Questi sono i requisiti necessari che il CAM deve soddisfare per il suo funzionamento rispetto alle direttive del progetto.

1.1.1 Mappatura

Il CAM nasce come piattaforma per la mappatura degli accessi informatici rispetto ad un ruolo lavorativo, ovvero, si vuole identificare quali diritti devono venire assegnati ad un collaboratore per permettergli di svolgere le sue mansioni lavorative.

L'azienda è suddivisa in *istituti*, gli ospedali, i quali sono a loro volta suddivisi in *dipartimenti* per esempio:

- pediatria
- laboratorio
- chirurgia
- fatturazione
- segretariato
- ecc...

Il personale viene dunque assegnato ad un dipartimento in uno specifico istituto, per identificare le attività svolte da un collaboratore gli viene anche assegnata una *funzione*.

Una funzione la si può trovare anche su più dipartimenti in diversi istituti.

Il ruolo lavorativo di una persona si compone dunque di questi 3 elementi:

- Istituto
- Dipartimento
- Funzione

Presso l'Ente Ospedaliero Cantonale sono presenti molti software per la cura del paziente, gestione apparecchiature e del personale. Alcuni software hanno la gestione accessi interna, ovvero l'informazione risiede nei loro database, per quest'ultimi sarà dunque necessario mappare questo dato in funzione del ruolo che una persona ricopre all'interno dell'azienda.

Alla base della mappa si troverà dunque il ruolo lavorativo aziendale, alla quale saranno relazionati i software necessari per permettere all'utente di tale ruolo di svolgere le sue mansioni lavorative ed in seguito ogni software relazionato avrà la sua tipologia di diritti specifica.

La struttura viene semplificata come mostrato in figura 1.1, dove si può notare molto semplicemente come i due ruoli hanno software diversi a dipendenza della loro esigenza, e ogni programma ha dei diritti *fini* a dipendenza del ruolo aziendale.

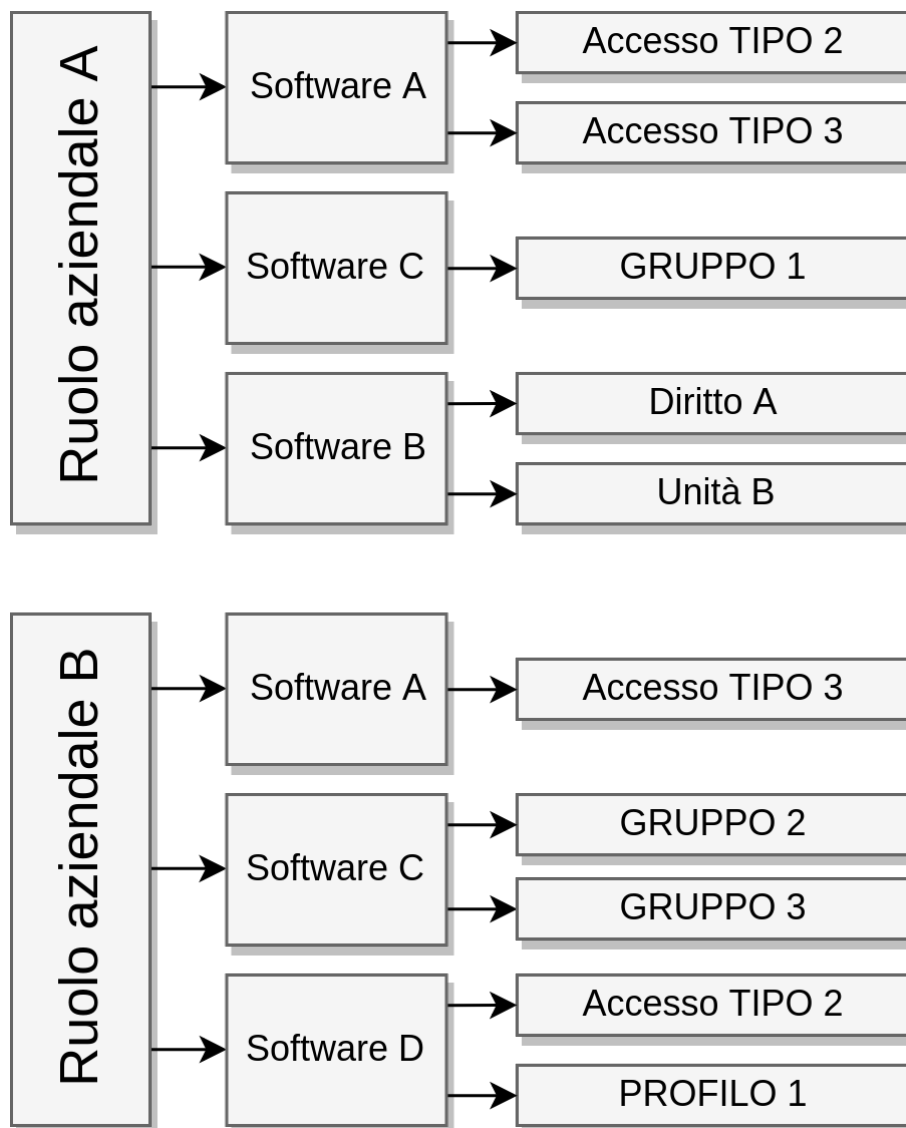


Figura 1.1: Schema mappatura

Viene dunque richiesto di modellare una struttura dati che possa supportare questa tipologia di mappatura.

La mappa dovrà venire generata automaticamente secondo lo stato attualmente in vigore in azienda.

1.1.2 Systems Integration

I software citati nella sezione precedente dovranno venire integrati nel CAM, l'integrazione permetterà al CAM di poter comunicare direttamente con il software in modo da estrarre tutte le informazioni necessarie per la costruzione automatica della mappa.

L'integrazione di un nuovo sistema dovrà avvenire senza modificare la struttura del CAM e senza dover ricorrere a modifiche future nel codice sorgente.

1.1.3 Gestione della mappa

Il CAM dovrà offrire un'interfaccia di gestione, questo per permettere agli amministratori del sistema di interagire con le informazioni contenute in esso.

In tal caso si rende necessario poter modificare la mappatura di un ruolo modificando i diritti per gli applicativi associati ad esso.

1.1.4 API

Uno degli obbiettivi primari del CAM è quello di essere integrabile con altri software che permettono la gestione degli accessi. A tale scopo il CAM fornisce le API necessarie con la quale i sistemi esterni potranno interagire.

Attualmente, in azienda per eseguire la creazione di un nuovo utente, i collaboratori del supporto all'utenza utilizzano un software denominato *EOCSigle*. Quest'ultimo permette di generare l'utente in Active Directory con una password generata casualmente, creare la casella di posta in exchange ed altre operazioni standard che sono da eseguire per ogni creazione di un'utenza informatica.

In seguito, mediante il software di ticketing, vengono notificati tutta una serie di responsabili per ogni software speciale. Quest'ultimi devono poi creare l'utente ed assegnare i diritti corretti nel software in loro gestione, il tutto manualmente.

Le API permetteranno, in un primo momento, di integrare *EOCSigle* nel CAM per automatizzare la gestione nei software "speciali", sfruttando la mappatura.

Le funzionalità principali richieste sono dunque:

- Creare un utente
- Assegnare diritti ad un utente
- Ottenere informazioni su un utente
- Notificare il CAM della creazione di un nuovo ruolo

1.2 Requisiti non funzionali

I requisiti non funzionali sono delle caratteristiche non citate nei requisiti di progetto, ma che sono comunque necessari per perseguirne l'obiettivo.

Infatti non descrivono cosa fa il sistema ma **come** lo fa.

1.2.1 Tecnologie utilizzate

Per lo sviluppo del CAM sono state impiegate le seguenti tecnologie

Spring MVC

Spring MVC è un framework per lo sviluppo di applicazioni web con backend in *JAVA*.

Nel progetto è stato utilizzato per sviluppare l'engine del CAM, dunque tutto il sistema viene eseguito in un webserver *Apache Tomcat*.

Inoltre Spring MVC è stato implementato per il backend delle API e dell'interfaccia web di gestione.

MongoDB

È un database *NoSQL* orientato a documenti, permette grande flessibilità quando si lavora con strutture dati non definibili a priori, il che lo rende la scelta migliore per l'esecuzione del progetto.

Al contrario di un *DBMS*¹ *SQL* comune è stato selezionato un *NoSQL*, questo perché la struttura dati contenuta nel CAM non è per nulla omogenea, infatti non è possibile descrivere in modo semplice la mappatura in quanto la struttura dei diritti di ogni sistema è diversa.

I *DBMS NoSQL* sono pensati apposta per questa tipologia di banca dati.

Bootstrap

È una collezione di stili *CSS* e anche alcuni *Script* in javascript per permettere un'abbellimento grafico semplice e uniforme su tutta l'interfaccia grafica del CAM.

¹Database management system

1.2.2 Sicurezza

La sicurezza dell'interfaccia web è gestita mediante **Spring Security** configurato per autenticare gli utenti tramite LDAP, nel caso specifico di questo progetto gli utenti accedono con le credenziali di *MS Active Directory*.

Come anticipato in precedenza la *GUI*² è suddivisa in 3 livelli:

- Utente base
- Utente informatico
- Amministratore

Mediante la configurazione con *LDAP* è possibile selezionare quali *gruppi AD* hanno accesso a quali aree dell'interfaccia, nel caso di questo progetto, essendo ancora in fase di sviluppo sono stati selezionati questi gruppi per selezionare i livelli di accesso:

- Amministratore: F_ICT_Governance_COLLABORATORE
- Utente informatico: Tutti gli utenti EOC
- Utente base: Tutti gli utenti LDAP

Di seguito è riportata la configurazione in *Spring Security*:

```

1  http
2  . authorizeRequests ()
3  . antMatchers ("/admin/**").hasAuthority(mainService.getProp("
    cam.ad.adminRole"))
4  . antMatchers ("/api/**").hasAuthority(mainService.getProp("cam
    .ad.servicesRole"))
5  . antMatchers ("/search/**").hasAuthority(mainService.getProp("
    cam.ad.adminRole"))
6  . antMatchers ("/").fullyAuthenticated()
7  . antMatchers ("/css/**").permitAll()
8  . antMatchers ("/webjars/**").permitAll()
9  . antMatchers ("/js/**").permitAll()
10 . antMatchers ("/images/**").permitAll()
11 . antMatchers ("/login").permitAll()

```

²Graphical User Interface (Interfaccia grafica)

Capitolo 2

Design del sistema

Il capitolo 2 ha lo scopo di illustrare le varie parti del CAM, nel dettaglio vengono mostrati: l'architettura modellata, il data model e le varie funzionalità del sistema.

2.1 Architettura

L'obiettivo principale di questa sezione è quella di mostrare l'architettura generale del sistema mostrando i principali componenti software coinvolti e l'interazione tra essi in termini di dati scambiati e funzionalità fornite.

È presente una libreria che contiene il data model e le interfacce che devono implementare i plugin.

In questa libreria è definito il modello dati che sarà condiviso fra il CAM e i vari plugin, e pure le interfacce che dovranno essere implementate per permettere al CAM di comunicare con i sistemi legacy.

Come mostrato in figura 2.1, il sistema è suddiviso in tre componenti principali:

- **CAM:** è il motore, al suo interno troviamo tutta la logica per interagire con i vari plugin e la GUI.
- **Data Model Condiviso:** è un set di classi e interfacce comuni al CAM e ai plugin, le quali vengono utilizzate per permettere alle due parti di comunicare.
- **Plugin:** si interfacciano direttamente con i *sistemi legacy*, questo dunque include l'utilizzo di API, connessioni database e/o accessi *MS Active Directory* per ogni sistema.

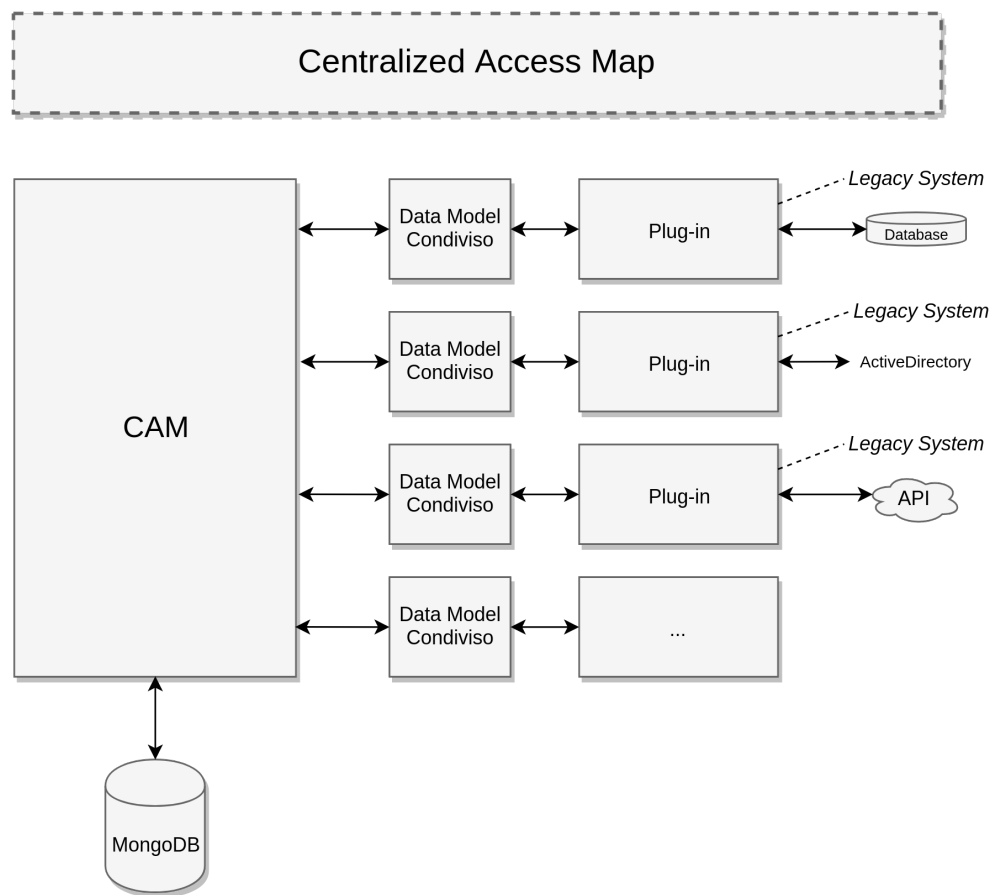


Figura 2.1: Schema architettura

2.2 Data Model

Il modello dati formalizza un insieme di classi e interfacce che rappresentano la struttura dati degli oggetti utilizzati dal CAM.

Il **Data Model** è condiviso tra il CAM e i vari plugin, in questo modo tutti saranno allineati sulla stessa gerarchia di oggetti.

Al suo interno si può trovare la struttura per gli oggetti di tipo *Right*, con una classe astratta e due classi che estendono *Right*, come pure la struttura per i dati degli utenti.

La classe **RightEntry** rappresenta un diritto specifico. **RightGroup** raggruppa più *entry* in un gruppo, per esempio i software in cui l'accesso di un utente è identificato da più diritti. **RightValue** è un container per i valori delle singole *entry*.

In figura 2.2 viene mostrato il diagramma completo per quanto riguarda i diritti.

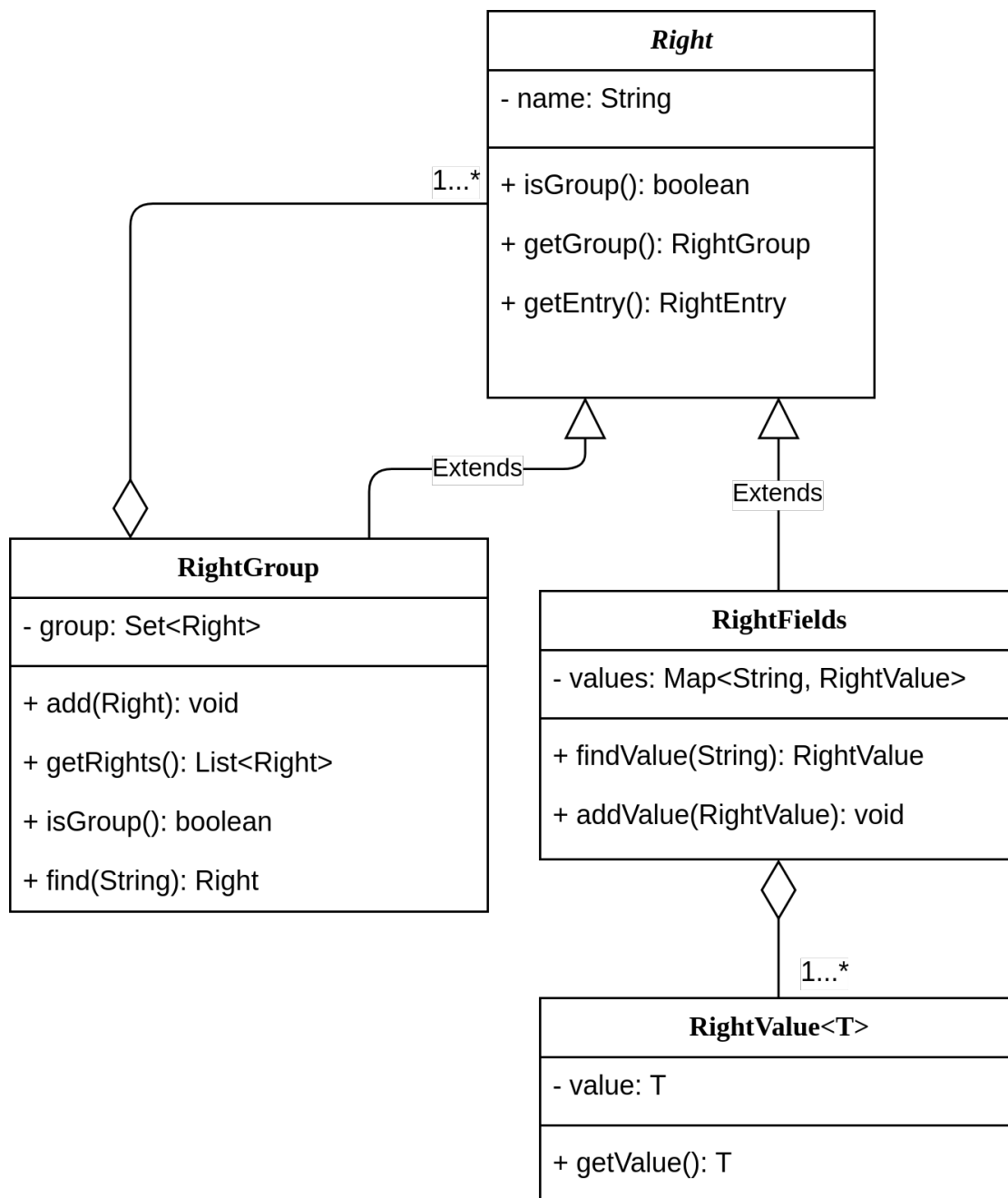


Figura 2.2: Right DataModel

Mediante questa struttura può essere modellato qualunque genere di diritto presente in azienda, i *generics* di **RightValue** permettono di estendere il modello per corrispondere alle proprie esigenze.

In aggiunta alle classi inerenti la struttura dei diritti, sono presenti due classi che definiscono un utente e un ruolo RU: **User** e **HRRole**.

La classe **User** rappresenta un utente informatico, il campo *collabID* rappresenta il numero

collaboratore assegnato dalle RU, mentre *username* rappresenta la sigla di accesso ai sistemi informatici.

La classe **HRRole** identifica il ruolo lavorativo di un utente all'interno dell'azienda.

In figura 2.3 viene mostrato il diagramma completo per quanto riguarda gli utenti, dove:

- *function*: è il nome della funzione lavorativa
- *functionCode*: è il codice della funzione lavorativa
- *department*: è il nome del dipartimento in cui si trova il ruolo
- *departmentCode*: è il codice del dipartimento in cui si trova il ruolo
- *organization*: è il nome dell'organizzazione/istituto in cui si trova il ruolo
- *organizationCode*: è il codice dell'organizzazione/istituto in cui si trova il ruolo

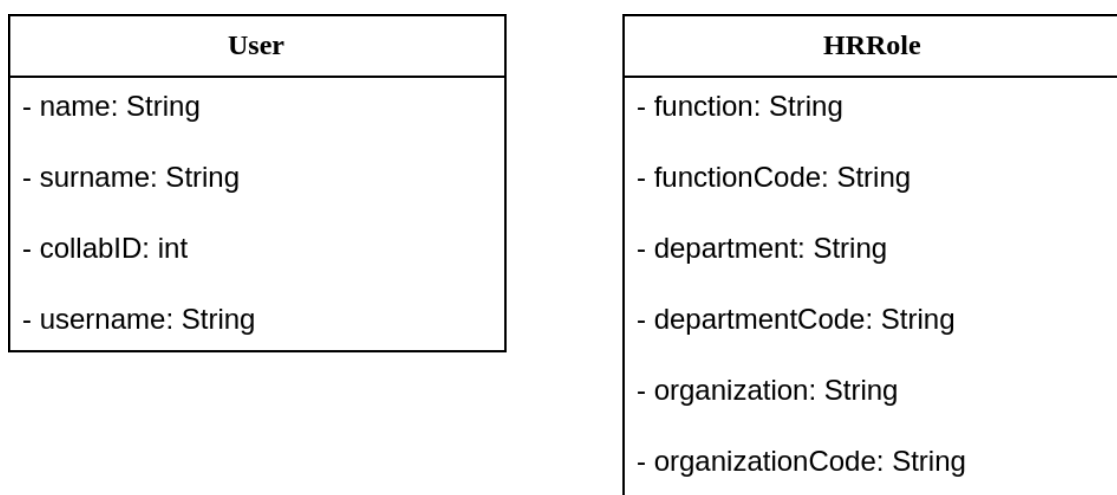


Figura 2.3: User DataModel

L'oggetto che verrà applicato a questo template è validato mediante il metodo *validateRight* per permettere massima compatibilità.

2.2.1 Interfacce

Sono state modellate le seguenti interfacce.

- LegacySystem
- RULegacySystem

La figura 2.4 rappresenta lo schema *UML* delle interfacce.

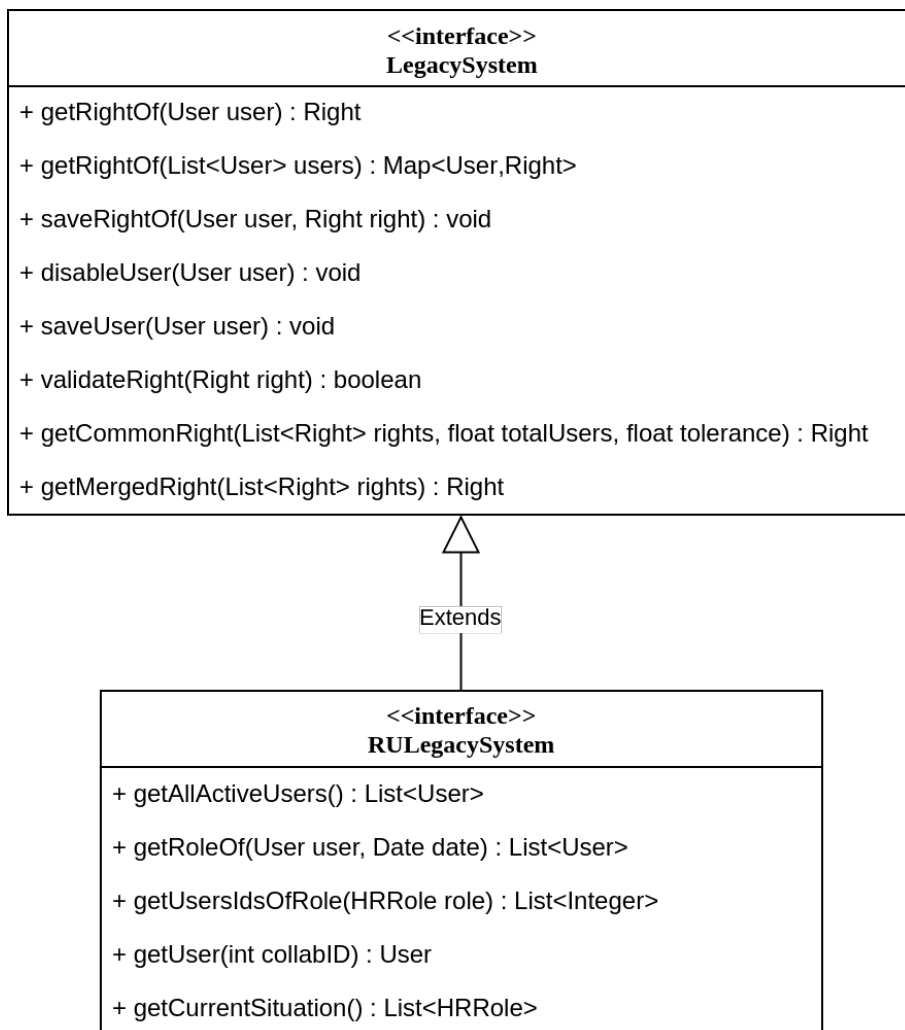


Figura 2.4: Interfacce

L'interfaccia **LegacySystem** mette a disposizione i metodi per permettere al CAM di interagire con i sistemi legacy mediante delle operazioni standard, la tabella 2.1 descrive tutti i metodi.

Tabella 2.1: Descrizione metodi LegacySystem

Metodo	Descrizione
Right getRightOf (User user)	Il risultato è un'istanza di <i>Right</i> che raffigura i diritti dell'utente <i>User</i> passato in parametro
Map<User,Right> getRightOf (List<User> users)	Il risultato è una mappa di <i>User</i> e <i>Right</i> che raffigura i diritti per ogni utente <i>User</i> passato in parametro
void saveRightOf (User user, Right right)	Se l'utente <i>User</i> passato in parametro non ha diritti gli vengono assegnati secondo <i>Right</i> passato in parametro, altrimenti vengono aggiornati
void saveUser (User user)	Se l'utente <i>User</i> passato in parametro esiste, viene aggiornato altrimenti viene creato nel sistema
boolean validateRight (Right right)	Valida il <i>Right</i> passato in parametro secondo gli standard del LegacySystem
Right getCommonRight (List<Right> rights, float totalUsers, float threshHold)	Ritorna un'istanza di <i>Right</i> con gli elementi in comune delle istanze nel parametro <i>rights</i> . Il parametro <i>totalUser</i> identifica il numero di utenti nel gruppo. Il parametro <i>threshHold</i> identifica la soglia minima per cui un diritto viene considerato <i>common</i> , per esempio se il valore è 0.6 almeno il 60% di <i>totalUsers</i> deve possedere il diritto
Right getMergedRight (List<Right> rights)	Unisce i <i>Right</i> passati in parametro in una sola istanza di <i>Right</i>
JsonNode getProperty (String name)	Ritorna una <i>property</i> del sistema, per esempio la lista degli accessi, la lista dei gruppi, ecc. Questo metodo sarà utilizzato per modificare l'oggetto <i>Right</i> del plugin.

L'interfaccia **RULegacySystem** eredita da *LegacySystem*, ed espone alcuni metodi che permettono al CAM di interagire con il *Master Data RU*¹. Quest'ultima dovrà dunque essere implementata dal plugin che comunicherà con il software in dotazione alle RU per la gestione dei collaboratori.

Tabella 2.2: Descrizione metodi HRLegacySystem

Metodo	Descrizione
List<User> getAllActiveUsers ()	Ritorna una lista di tutti gli utenti attivi nel sistema
List<HRRole> getRoleOf (User user, Date date)	Ritorna un'istanza di <i>HRRole</i> per ogni ruolo assegnato all'utente <i>User</i> passato in parametro alla data <i>Date</i>
List<Integer> getUsersIdsOfRole (HRRole role)	Ritorna una lista di interi che identificano gli ID degli utenti appartenenti al ruolo <i>HRRole</i> passato in parametro
User getUser (int collabID)	Ritorna l'utente identificato dall'id <i>collabID</i> passato in parametro
List<HRRole> getCurrentSituation ()	Ritorna una lista di <i>HRRole</i> per ogni ruolo attualmente attivo

¹ Software centrale di gestione dei collaboratori.

2.3 Plugin GUI

Il plugin è responsabile di fornire un template che permette al CAM di "disegnare" la struttura dell'istanza di *Right* la quale rappresenta un diritto.

Questo permette flessibilità allo sviluppatore del plugin che può rappresentare i dati in modo più simile al sistema con cui il plugin andrà a comunicare.

2.3.1 Lettura

Il template è un file denominato **template.mst** e deve essere posizionato nelle risorse del plugin.

Il CAM processerà il template con *mustache*, dunque si dovrà rispettare la sintassi di questo motore.

Questo è un esempio di *template* con la sintassi mustache.

```
1 <table>
2   <thead>
3     <tr>
4       <th>Tipo di accesso</th><th>Tipo di contatto</th>
5     </tr>
6   </thead>
7   <tbody>
8     <tr>
9       {{#rights}}
10        <td>{{values.name.value}}</td>
11        {{/rights}}
12      </tr>
13    </tbody>
14  </table>
```

2.3.2 Modifica/Inserimento

Il template descritto in precedenza ha il solo scopo di *visualizzazione*.

Per permettere agli amministratori di interagire con il sistema e dunque modificarne la mappatura è necessario fornire anche una seconda parte di *HTML* e *Javascript*.

A questo scopo è dunque necessario creare una cartella denominata **edit** all'interno delle risorse del plugin, contenente i seguenti file:

- script.js
- view.html

In questo caso, al contrario della visualizzazione non è sufficiente fornire solo un template in cui il CAM andrà a inserire le informazioni, questo perché per modificare la struttura è necessaria la conoscenza dello sviluppatore che ha creato il plugin.

Il CAM metterà a disposizione del plugin un'intera pagina, dove andrà a inserire il codice in *view.html* e *script.js*.

Lo script dovrà implementare due funzioni, che permetteranno di comunicare con il CAM:

- onReady(json)
- onSave()

La funzione onReady verrà chiamata quando la pagina è *renderizzata* e come argomento troviamo un oggetto json che descrive la struttura di *Right* generata dal plugin.

Invece, la funzione onSave() verrà chiamata dal CAM quando l'utente preme il tasto *salva* nella pagina.

La figura 2.5 rappresenta un diagramma di sequenza per questa funzionalità.

In aggiunta il CAM mette a disposizione dello sviluppatore alcune funzioni *javascript* come riportato in tabella 2.3 come pure il supporto delle seguenti librerie:

- Mustache
- jQuery
- FontAwsome
- Bootstrap

Tabella 2.3: Interfaccia script.js

Funzione	Descrizione
getProperty (property, success, error)	Chiede al CAM di richiamare il metodo getProperty del plugin, il parametro property identifica il nome della property mentre success e error sono delle funzioni che verranno chiamate al termine della richiesta. La chiamata verrà eseguita in modo asincrono e verrà ritornato un oggetto JSON.
save (json)	Comunica al CAM l'oggetto finale modificato, questo metodo sarà da chiamare in seguito all'evento <i>onSave</i> come conferma che la modifica è andata a buon fine da parte dello script js.

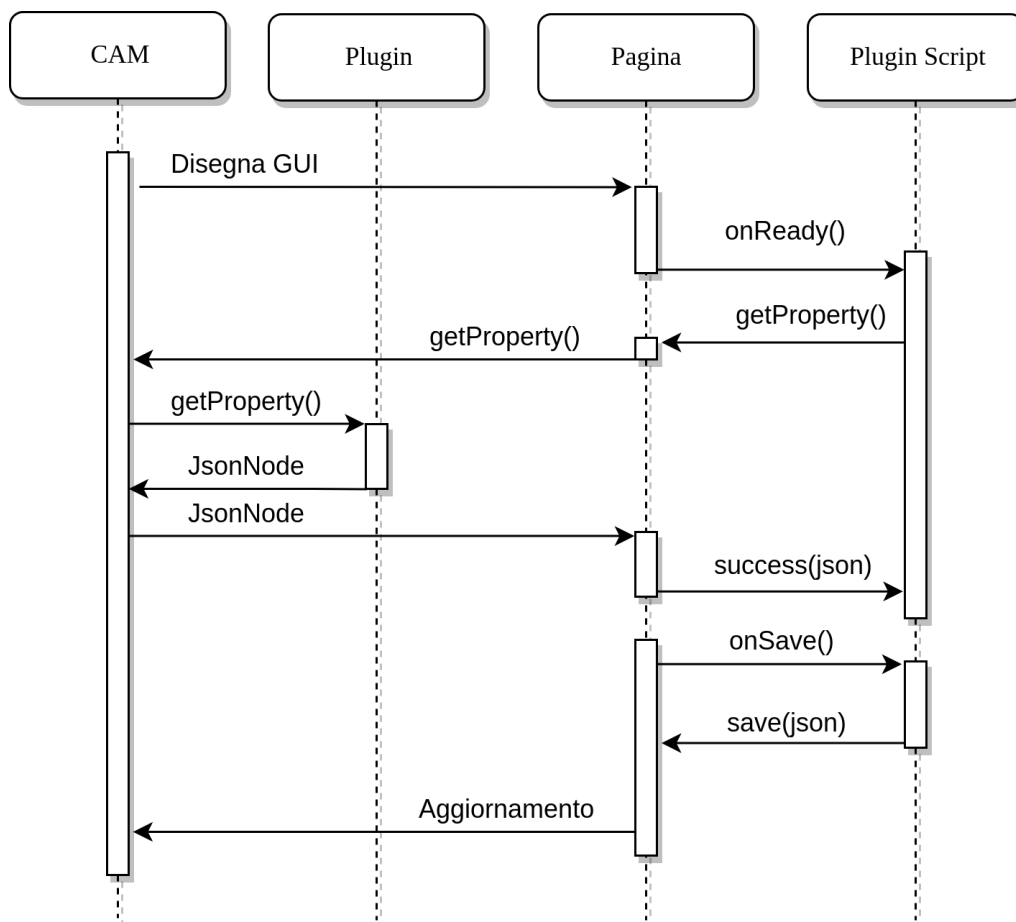


Figura 2.5: Diagramma di sequenza modifica mappa

2.4 CAM

Il CAM ha un suo modello dati che sfrutta per permettergli la persistenza dei dati nel database e di gestire al meglio le informazioni che provengono dai plugin.

Nelle sottosezioni seguenti vengono illustrate le sue componenti principali.

2.4.1 Property

La classe Property permette la gestione delle impostazioni, infatti mediante questa tipologia di oggetto vengono salvati nel database tutte le impostazioni inerenti gli accessi, i gruppi AD, ecc...

Come mostrato dalla figura 2.6 l'oggetto non è altro che l'associazione chiave-valore delle property.

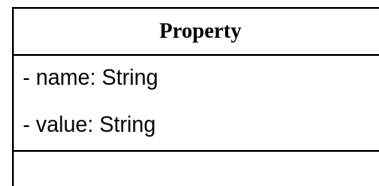


Figura 2.6: Data model Property

Questa struttura viene salvata nel database, nella collection *Properties*.

2.4.2 Plugin

Il plugin è uno dei componenti principali del CAM, esso infatti descrive un plugin con le sue versioni.

Questa struttura è composta da due classi:

- PluginLS
- PluginLSVersion

La classe **PluginLS** rappresenta un plugin, ovvero l'entità che poi conterrà tutte le versioni. Mentre **PluginLSVersion**, come suggerisce il nome rappresenta la versione specifica di un plugin.

La figura 2.7 rappresenta lo schema di queste due classi.

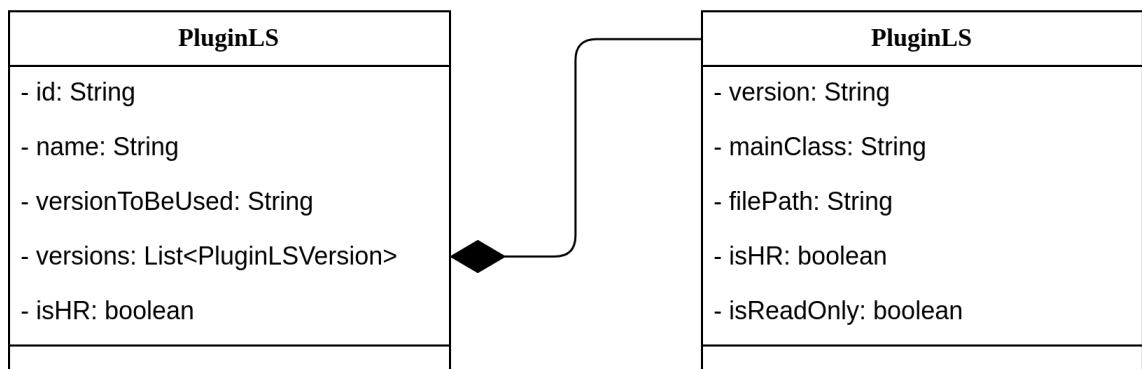


Figura 2.7: Data model Right

Nelle tabelle 2.4 e 2.5 sono descritti i campi delle classi presentate precedentemente.

Tabella 2.4: Campi classe PluginLS

Campo	Descrizione
id	ID del plugin
name	Nome del plugin
versionToBeUsed	Versione attualmente in uso
versions	Lista di tutte le versione installate
isHR	Identifica se il plugin è delle risorse umane

Tabella 2.5: Campi classe PluginLSVersion

Campo	Descrizione
version	Codice della versione
mainClass	Classe che implementa l'interfaccia dei plugin
filePath	Directory del file <i>jar</i> sul server
isHR	Identifica se la versione è compatibile con le risorse umane
isReadOnly	Identifica se il plugin è in sola lettura

Gli oggetti **PluginLS** sono salvati nella collection *Plugins* all'interno del database.

2.4.3 Legacy System Manager

Questo componente è necessario per ottenere le istanze dei vari plugin, il suo compito è quello di rendere disponibili le istanze di *LegacySystem* e *HRLegacySystem* (vedi sezione 2.2.1).

Come mostrato in figura 2.8, le classi principali sono 2: *LegacySystemManager* e *LSContainer*.

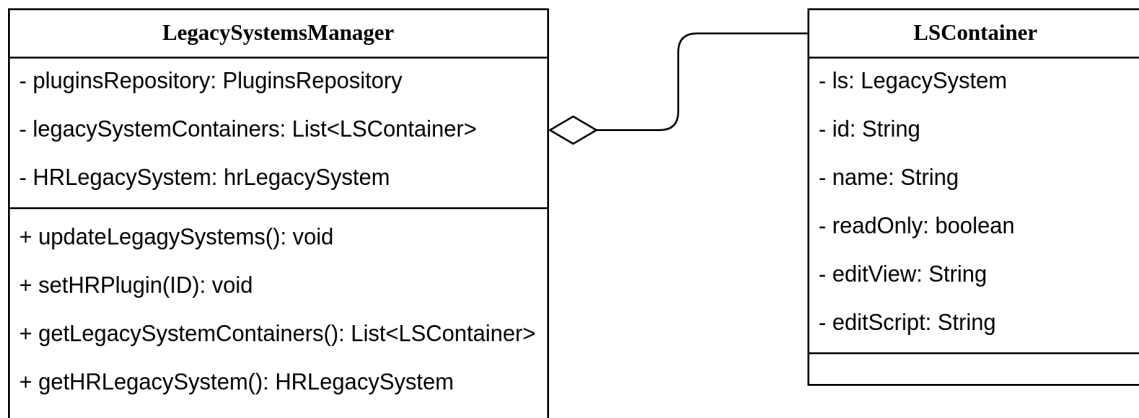


Figura 2.8: Legacy System DataModel

LegacySystemManager è un *Component* di Spring, in questo modo mediante la *dependency injection* è possibile delegare a Spring l'inizializzazione.

Di seguito sono descritti i campi delle classi precedentemente citate.

Tabella 2.6: Campi classe LegacySystemManager

Campo	Descrizione
pluginsRepository	Repository Spring che si occupa di interfacciarsi con il database per quanto riguarda i plugin.
legacySystemContainers	Lista di tutti gli LSContainer per i plugin installati
hrLegacySystem	Istanza di HRLegacySystem

Tabella 2.7: Campi classe LSContainer

Campo	Descrizione
ls	Istanza di <i>LegacySystem</i>
id	ID del plugin
name	Nome del plugin
readOnly	Identifica se il plugin è in sola lettura
editView	Visualizzazione per la modifica
editScript	Script per la modifica

2.5 Funzionalità del sistema

2.5.1 Mappatura

La mappatura dei diritti verrà eseguita sfruttando tutti i plugin che si trovano sul CAM, in figura 2.9 viene mostrato un diagramma di sequenza per il processo di creazione della mappa per un ruolo.

Per semplicità di rappresentazione lo schema mostra le interazioni con una sola entità *Plugin*, ovviamente verranno interrogati tutti i plugin disponibili.

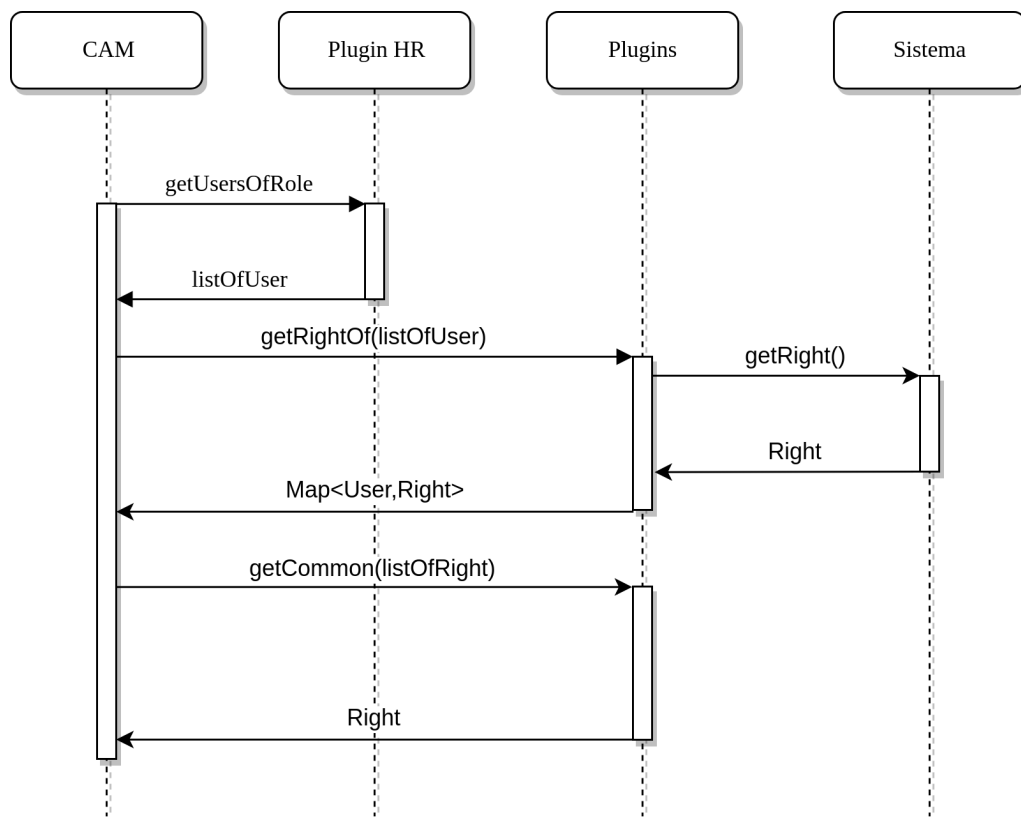


Figura 2.9: Diagramma di sequenza mappatura

Come si può notare, la vera mappatura viene effettuata mediante il metodo `getCommon` che ritorna un oggetto **Right**, il quale rappresenta il diritto di un determinato ruolo sul sistema.

La mappa viene in seguito salvata nel database MongoDB, essendo il database un *document based* l'oggetto *Right* viene serializzato in *JSON*.

2.5.2 Gestione

La gestione avviene mediante un'interfaccia grafica, la *GUI* è una pagina web raggiungibile con un browser.

Il *webserver* è all'interno dell'engine, ovvero direttamente incorporato nel CAM.
Una panoramica più dettagliata dell'interfaccia di gestione è disponibile nel Capitolo 4.

I principali utenti dell'interfaccia di gestione sono gli **Amministratori di sistema**, ma l'accesso è consentito a tutti gli utenti con una sigla di accesso informatica, ovvero in Active Directory.

Gli accessi alla piattaforma sono suddivisi in tre:

- Amministratore
- Informatico
- Utente

La distinzione viene eseguita mediante gruppi *Active Directory*.

Amministrazione

Ha pieno potere sul CAM e su tutte le sue funzionalità, l'amministratore può modificare la mappatura per ogni ruolo secondo le sue esigenze.

Questo ruolo ha la possibilità di assegnare o revocare diritti agli utenti, come pure installare un nuovo plugin.

Informatico

Questo ruolo permette una visione globale della mappa in sola lettura. È possibile dunque effettuare ricerche nei ruoli e visionarne utenti e sistemi mappati.

Permette anche una ricerca in tutti gli utenti per visionare, *on-demand*, i diritti su ogni sistema integrato nel CAM mediante il plugin.

Il ruolo non ha alcuna possibilità di modificare la mappa o i diritti dei singoli utenti.

Questo tipo di accesso è di supporto per i collaboratori informatici, i quali possono rapidamente verificare gli accessi assegnati ad ogni utente in caso di necessità.

Utente

Questo accesso è molto limitato, permette di visionare esclusivamente i sistemi a cui si ha accesso, senza però vedere nel dettaglio che tipo di diritto.

Questo ruolo non permette di visionare gli accessi dei colleghi e/o superiori.

2.5.3 API

Le API permetteranno di integrare il CAM in altre applicazioni nel contesto aziendale. Una panoramica più dettagliata sulle API è disponibile nel Capitolo 4.

Le API sono suddivise secondo le entità presenti nel CAM, ovvero:

- News
- Plugin
- Ruoli
- Utenti

Capitolo 3

Sistemi integrati

Lo scopo del capitolo 3 è quello di descrivere la maggior parte dei sistemi attualmente in utilizzo in azienda. Quest'ultimi sono i sistemi in cui la gestione degli accessi è interna e viene gestita da una persona di riferimento.

Dove possibile, è stato realizzato un plugin da integrare nel CAM.

3.1 Software Risorse Umane

Mediante il software delle risorse umane, che attualmente si chiama XpertLine, verranno estratte tutte le informazioni necessarie per identificare una funzione lavorativa. Questo software è principalmente utilizzato per la gestione degli stipendi.

La funzione lavorativa identificherà un gruppo di utenti che svolgono le medesime attività al lavoro e che dunque hanno gli stessi accessi sulle svariate piattaforme.

L'interrogazione alla banca dati avviene mediante un servizio web, denominato EOCUser, reperibile all'indirizzo <http://www-eocuser/wsEocUser/wseocuser.asmx>.

Il metodo che verrà sfruttato per ottenere il dettaglio di un singolo collaboratore è:

GetDipendenteXpertLine

La lista di tutti i collaboratori che attualmente lavorano presso l'Ente Ospedaliero Cantonale viene fornita da un altro database, molto più veloce e snello tramite un servizio WEB.

Una funzione lavorativa è identificata da tre campi nel programma in XpertLine:

- Istituto: Sede lavorativa
- Reparto: Sotto struttura dell'istituto

- Funzione: Tipologia di lavoro che si svolge

3.2 Legacy Systems

All'interno dell'azienda vi sono una moltitudine di software con la gestione accessi incorporata, questo implica una gestione degli accessi specifica per ognuno di loro. Questa gestione attualmente viene eseguita manualmente da diversi collaboratori.

Mediante un'intervista mirata allo SPOC (single point of contact), ovvero dove avviene la creazione iniziale di un nuovo utente, sono stati evidenziati i seguenti sistemi legacy nel quale vengono eseguite operazioni manuali.

Ognuno di questi sistemi è stato integrato¹ nel CAM mediante l'architettura a plugin descritta in precedenza.

3.2.1 Active Directory

Gli utenti in Active-Directory appartengono semplicemente a dei gruppi, mediante i quali viene dato accesso a determinate cartelle nella file system, vengono installati programmi, ecc...

In azienda AD è anche il sistema centrale per la gestione degli accessi ai computer.

API

Per accedere in lettura ai gruppi di un determinato utente sono disponibili delle API in SOAP, il dettaglio è visibile nell'elenco seguente:

- **URL:** www-eocuser/wsEocUser/wseocuser.asmx
- **Metodo:** WS_GetUserDataActiveDirectory_SPECIF_SE_LIKE

Per semplicità di utilizzo la richiesta viene fatta tramite *GET*.

I parametri passati sono:

Tabella 3.1: Parametri API

Campo	Valore	Descrizione
bolSearchLike	false	Specifica se la ricerca impiega un <i>like</i> , in questo caso non serve quindi di default viene passato <i>false</i>
strUserId	username	Nome utente da cercare in ActiveDirectory

¹I sistemi per la quale si necessita un intervento da parte di un'azienda terza non sono stati effettuati. Maggiori informazioni nelle conclusioni del capitolo

In risposta si ottiene un *XML* che descrive un utente con i relativi gruppi di appartenenza, di seguito è riportata una porzione della risposta.

```
1 <dtActiveDir diffgr:id="dtActiveDir1" msdata:rowOrder="0"
    diffgr:hasChanges="inserted">
2   <act_userad>EOC9952</act_userad>
3   <act_employeeNbr>9952</act_employeeNbr>
4   <act_familyname>Bonomi</act_familyname>
5   <act_givename>Niko</act_givename>
6   <act_memberof>
7     Gruppo_VDI_Users2;
        Gruppo_VDI_Users_Sopraceneri_Migration;
        Gruppo_VDI_Cure2;Gruppo_VDI_Users_Sopraceneri2;
        Gruppo_VDI_Users_Sopraceneri;Gruppo_VDI_Cure_AET;
        Gruppo_VDI_Users;APPL_GECO_BETA;Utenti_Win10;
        EOC_WANNACRY_ACCETTATO;EOC_User Over Exchange
8   </act_memberof>
9   <act_whencreated>21-10-2009</act_whencreated>
10  <act_statusad>ON</act_statusad>
11  <act_mail>Niko.Bonomi@eoc.ch</act_mail>
12  <act_expiringdate/>
13 </dtActiveDir>
```

Data Model del Plugin

Come mostrato in figura 3.1, il modello per il plugin è relativamente semplice, è basato sulla risposta ottenuta dalle API.

ADUser
- memberOf : List<String>
- userad : String
- employeeNbr : String
- givenName : String
- familyName : String
- whenCreated : String
- status : String
- mail : String
- expiringDate : String
+ addGroup(String): void

Figura 3.1: Active Directory Data Model

Tabella 3.2: Campi classe ADUser

Campo	Descrizione
memberOf	Gruppi di appartenenza
userad	Nome utente presente in ActiveDirectory
employeeNbr	Numero dipendente RU
givenName	Nome
familyName	Cognome
whenCreated	Data di creazione
status	Stato, identifica se l'utente è attivo o inattivo
mail	Indirizzo email
expiringDate	Data in cui la sigla ActiveDirectory diventerà inattiva

3.2.2 Polypoint

Polypoint è un'azienda con sede in Svizzera che produce una suite di software, in azienda attualmente sono in funzione i seguenti:

- PEP: Gestione timbrature e turni di lavoro
- RAP: Gestione agende mediche (appuntamenti, visite, apparecchiature, ecc)

- DIS: Gestione agende per i letti ospedalieri

La gestione accessi è leggermente differente fra PEP e RAP/DIS. Per quanto riguarda il PEP vengono dati accessi individuali in base a dei modelli prestabiliti (lettura, scrittura, assegnazione diritti, ecc), in seguito si deve selezionare dove si possono esercitare gli accessi. Lato RAP/DIS gli utenti vengono assegnati a gruppi specifici in base alle attività che devono svolgere.

La problematica in questo caso è la frammentazione dei diritti, in quanto non è possibile assegnare 2 gruppi per un singolo utente, spesso si è costretti a creare un gruppo personalizzato con gli accessi di entrambi i gruppi.

Esempio: L'utente X si trova nel gruppo XYZ assieme all'utente W e Q. L'utente X deve svolgere mansioni anche del gruppo ABC in cui si trovano E e F. Gli utenti W e Q non devono vedere quello che vedono E ed F e viceversa. Attualmente viene quindi creato un nuovo gruppo denominato XYZ + ABC specifico per l'utente X.

Accedere al database, attualmente viola il contratto di manutenzione e garanzia dell'azienda produttrice, per ora è dunque impossibile accedere ai dati.

La piattaforma non mette a disposizione nessun genere di servizio web.

3.2.3 xLnet3

Questo è un software *multi-purpose*, infatti permette una gestione contabile per i dipendenti, gestisce il vecchio sistema di valutazione per il personale e per la comanda di medicinali. Per gli utenti vengono assegnate due tipologie di accesso:

- "Cosa può fare" identificato dal gruppo
- "Dove lo può fare" identificato da UNIT

Il database *SQL* si trova sul server **EOCSQL030** all'istanza **EOCSQL030**, denominato **xl-net3**.

Gli utenti sono contenuti nella tabella *XLuser*, il gruppo si trova in *XLuserGrp* e la relazione fra utente e *unit* è descritta nella tabella *XLuserUnit*.

La figura 3.2 mostra uno schema ER del database per illustrare le relazioni fra le tabelle e i campi interessati dal progetto.

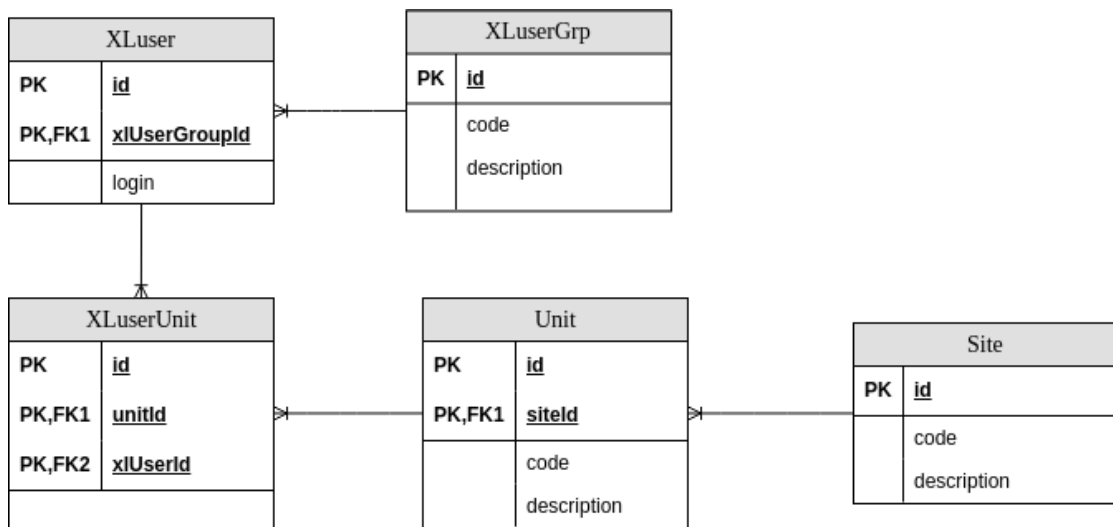


Figura 3.2: xLnet3 ER

I campi **code** identificano il codice di un'entità, mentre **description** è un testo descrittivo dell'entità.

Questo software non dispone di **API**, quindi l'accesso ai dati viene effettuato direttamente nel *database*.

Data Model del Plugin

In funzione dello schema ER (figura 3.2) raffigurante il database, è stato modellato il *Data-Model* visibile in figura 3.3.

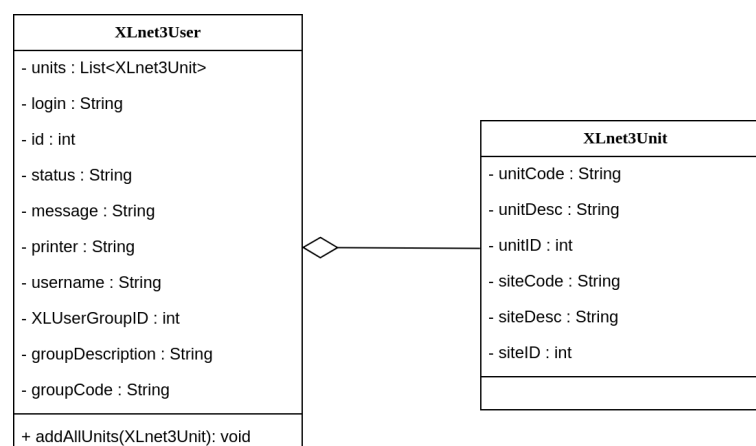


Figura 3.3: XLnet3 DataModel

Nelle tabelle 3.3 e 3.4 vengono descritti i campi delle due classi.

Tabella 3.3: Campi classe XLnet3User

Campo	Descrizione
units	Unità di appartenenza
login	Nome utente in XLnet3
status	Identifica se l'utente è attivo oppure inattivo
printer	Stampante di default per l'utente
username	Nome completo dell'utente
XLUserGroupID	Id gruppo di appartenenza
groupDescription	Descrizione gruppo di appartenenza
groupCode	Codice identificativo del gruppo

Tabella 3.4: Campi classe XLnet3Unit

Campo	Descrizione
unitCode	Codice descrittivo unità
unitDesc	Descrizione unità
unitID	ID unità
siteCode	Codice descrittivo sito
siteDesc	Descrizione sito
siteID	ID sito

3.2.4 XpertLine

XpertLine è attualmente lo strumento principale per i collaboratori che lavorano nelle risorse umane. Infatti permette di gestire i contratti di lavoro, i salari e ovviamente i dipendenti.

Il sistema è abbastanza datato, quindi il suo database si trova su un server AS400.

Il database è **CHEOC/IASP_PROD**.

XpertLine è suddiviso in moduli, ogni gruppo ha diritti fini per ciascuno di essi. La struttura è dunque rappresentata nella figura 3.4.

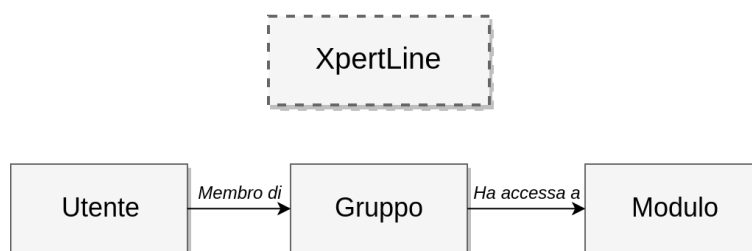


Figura 3.4: XpertLine accessi

Come si può notare un utente appartiene a uno o più gruppi, e quest'ultimo ha accesso su uno o più moduli.

La limitazione di questo software è che non si possono dare accessi in sola lettura, dunque ogni accesso permette la scrittura, inoltre quando un gruppo ha accesso a un modulo i suoi membri potranno vedere tutto di tutti.

La libreria che contiene gli utenti è denominata **XSECDTA01** e la tabella **SE59T**. Il campo con la sigla utente è **SE5KA**, le sigle sono tutte in *upper-case*.

Essendo questo un software acquistato da un'azienda, non è disponibile la struttura del database. In questo caso non si può integrare l'ottenimento dei diritti per questo sistema. Tuttavia è stato comunque sviluppato un plugin in quanto XpertLine è il sistema per la gestione del personale, quindi vengono estrapolate le funzioni lavorative specifiche per ogni collaboratore.

3.2.5 xLeoc

Il sistema xLeoc è stato introdotto come *workaround* alla limitazione dei diritti di *xpertLine*. xLeoc è una piattaforma per permettere l'accesso in sola visualizzazione ai dati di *xpertLine*, inoltre si può limitare la visualizzazione ai propri subordinati.

Il database si chiama **XLEOC**, è un SQL e si trova sul server **SVSQLCL041** all'istanza **SVSQLCL041**.

Come mostrato in figura 3.5 gli accessi vengono assegnati alla singola persona. Viene assegnato un ruolo e un modulo sulla quale lo si può esercitare.

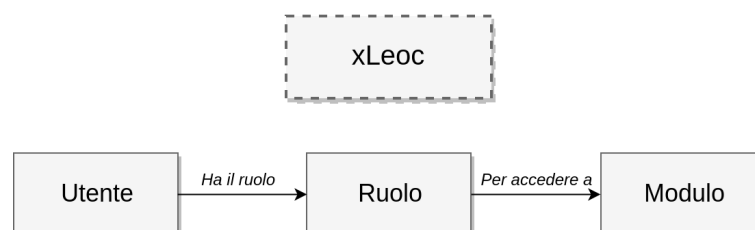


Figura 3.5: xLeoc accessi

Essendo questo un software sviluppato da un'azienda terza, non siamo in possesso di alcun genere di documentazione. Questo impossibilita l'implementazione del sistema nel CAM.

3.2.6 WinScribe

WinScribe è il software di gestione dei dittafoini, essenzialmente questi apparecchi sono dei registratori audio.

I medici li impiegano per registrarsi mentre esprimono una diagnosi per un paziente, in seguito una segretaria sarà incaricata di trascrivere quanto dettato dal medico.

Il software gestisce le segretarie (typist) e i medici (author), le segretarie vengono amministrate dal loro superiore e dunque i loro accessi sono coordinati esternamente all'Area ICT, flusso diverso per i medici perché gli accessi vengono dati manualmente dai nostri collaboratori.

Il database SQL si chiama **WinScribe** ed è localizzato sul server **SVSQLCL040** all'istanza **SVSQLCL040**.

Gli utenti sono creati in automatico in base ad un gruppo AD da un *job* eseguito periodicamente, i diritti invece vengono assegnati manualmente quando viene consegnato un nuovo dittafono.

Quest'ultimi sono suddivisi in due campi:

- Department
- JobType

Department identifica l'istituto, mentre *JobType* è il diritto.

Alla creazione di un utente i valori di default sono:

- Department: **EOC**
- JobType: **999**

Questi identificano un utente senza diritti.

La figura 3.6 mostra uno schema ER del database per illustrare le relazioni fra le tabelle e i campi interessati dal progetto.

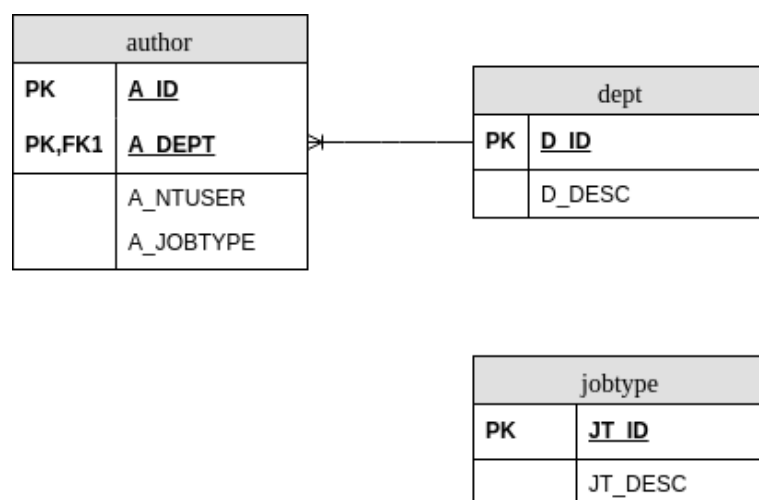


Figura 3.6: WinScribe ER

Come si nota dallo schema la tabella *author* non ha relazioni mediante chiave esterna con la tabella *jobtype*, in ogni caso il campo *A_JOBTYPE* si riferisce al *JT_ID*.

L'applicativo non dispone di **API** quindi gli accessi ai dati verranno effettuati direttamente nel *database* SQL.

Data Model del Plugin

Il modello per questo *plugin* è stato strutturato come illustrato in figura 3.7.

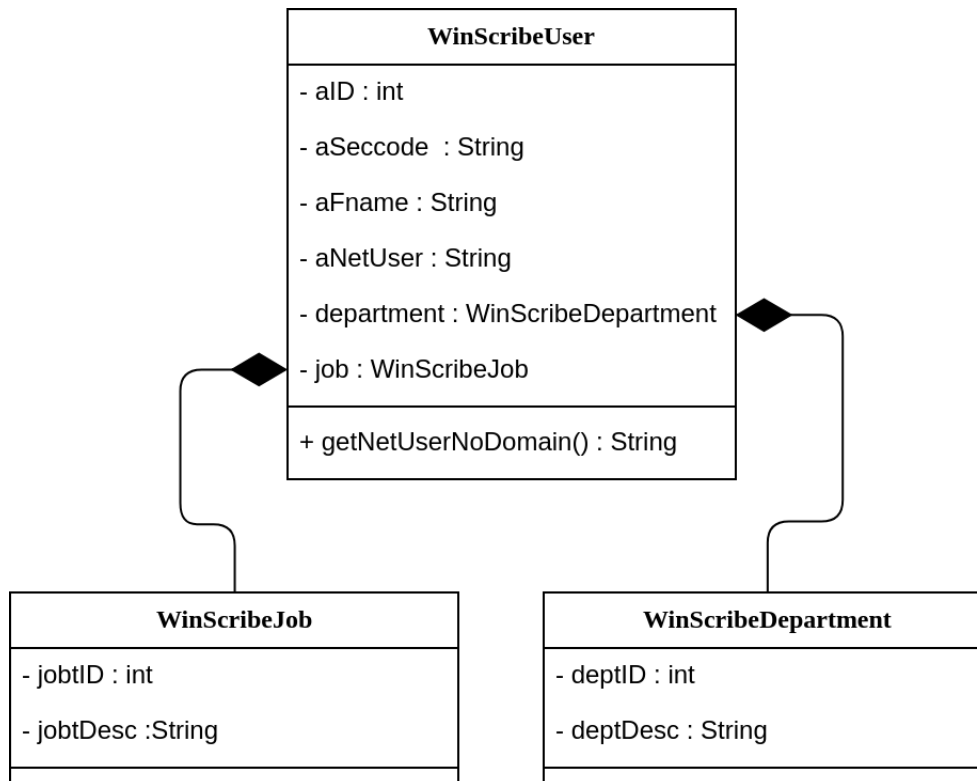


Figura 3.7: WinScribe DataModel

Nelle tabelle 3.5, 3.6 e 3.7 sono elencati i campi del datamodel con relativa descrizione.

Tabella 3.5: Campi classe WinScribeUser

Campo	Descrizione
aID	ID utente
aSeccode	Codice identificativo interno
aFname	ID Nome utente
aNetUser	Sigla di accesso all'applicativo
department	Dipartimento di appartenenza
job	Tipo di Job

Tabella 3.6: Campi classe WinScribeJob

Campo	Descrizione
jobtID	ID interno
jobtDesc	Descrizione del tipo di attività

Tabella 3.7: Campi classe WinScribeDepartment

Campo	Descrizione
deptID	ID dipartimento
deptDesc	Descrizione dipartimento

3.2.7 AS400

Attualmente AS400 è in via di dismissione, tuttavia ci sono ancora una serie di persone che lo utilizzano.

Questa piattaforma funziona a menu, dove a una persona viene dato accesso ad uno specifico menu per eseguire operazioni sui dati.

Il database contenente tutte le librerie si trova al seguente indirizzo **172.30.113.1:446**.

Per ogni istituto è stata creata una libreria con la nomenclatura seguente: «*ISTITUTO*» + *DB*.

Il file contenente i profili si trova nella libreria **YPSYS** e si chiama **PROFI10F**.

Gli utenti che vengono generati "standard" utilizzano menu dinamici, dunque per il fine di questo progetto verranno tenuti in considerazione solo gli utenti con menu dinamico. A tale scopo si utilizza la colonna *PRINI* dove il valore **D** identifica un profilo utente con menu dinamico.

I menu sono contenuti nella libreria specifica per il profilo, tale libreria la si può trovare nel campo **PRINIL** del profilo utente. Il nome del file con i menu assegnati ai profili è denominato **MENUD00F**.

Il file **MENUD00F** contiene la definizione di tutte le voci presenti nel menù di un profilo.

La tabella 3.8 è un esempio del contenuto del file per il profilo *LBMU00*.

Tabella 3.8: Esempio contenuto MENUD00F

MENOME	MERIGA	MECALL	MEDESC
LBMU00	0		LBMU00 - LABORATORIO
LBMU00	5	LB406C	GESTIONE REPARTI FATT./LAB400
LBMU00	7	GPMU90M	GPMU90 - GESTIONE PAZIENTI
LBMU00	13	.	FORMULARI LETTI IN AMMINISTRAZ
LBMU00	14	PF173	-> NON ANCORA FATTURATI
LBMU00	15	PF179	-> GIA' FATTURATI

MERIGA identifica la riga della voce nel menu, *MECALL* identifica la procedura che verrà chiamata quando la voce viene selezionata, *MEDESC* è una descrizione testuale del menu. La voce alla riga 0 identifica il nome del menu.

Data Model del Plugin

Il modello in figura 3.8 è stato strutturato sulla base di quanto presente nel database.

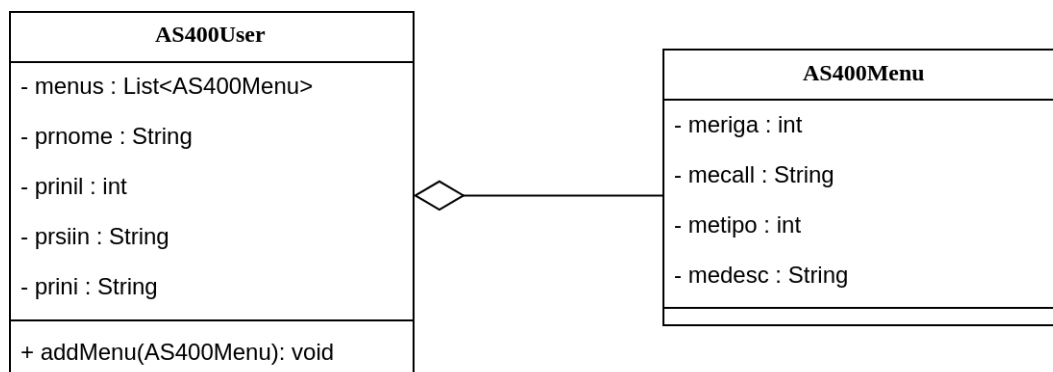


Figura 3.8: AS400 DataModel

Le tabelle 3.9 e 3.10 descrivono i campo del modello.

Tabella 3.9: Campi classe AS400User

Campo	Descrizione
menus	Menu assegnati ad un utente
prnome	Nome profilo
prinil	Menu default per l'utente
prsiin	Campo alternativo per il menu default dell'utente
prini	Identifica il tipo di menu

Tabella 3.10: Campi classe WinScribeJob

Campo	Descrizione
meriga	Numero di riga nel menu
mecall	Procedura che viene chiamata alla selezione della voce nel menu
metipo	Tipologia di elemento del menu
medesc	Descrizione testuale dell'entry nel menu

3.2.8 Service Desk Manager

Abbreviato con SDM è il software di ticketing aziendale, inoltre contiene l'inventario di tutte le apparecchiature informatiche installate presso l'Ente Ospedaliero Cantonale.

Gli utenti utilizzano le credenziali AD per accedere alla piattaforma, però gli accessi sono gestiti internamente. In SDM gli utenti sono contenuti nella tabella **cnt**, dove la sigla di accesso è nel campo *userid*.

Come mostrato in Figura 3.9, ci sono due campi che identificano gli accessi di un determinato utente:

- Tipo di contatto: **ctp**
- Tipo di accesso: **acctyp**

Il *tipo di accesso* definisce i diritti che un utente ha all'interno del sistema, mentre il *tipo di contatto* definisce semplicemente di che tipo è l'utente (Analista, Dipendente, Cliente). Il campo *sym* è la descrizione testuale dell'elemento.

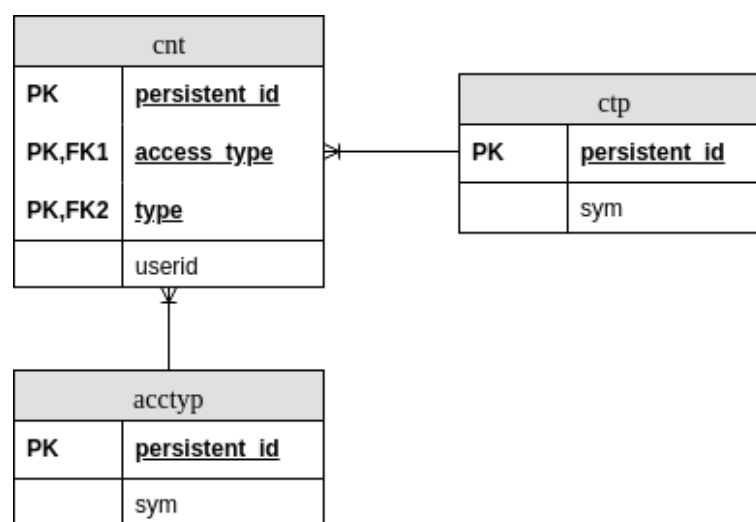


Figura 3.9: SDM ER

API

[1] Per accedere ai dati ci sono delle API Rest che permettono di manipolare qualunque oggetto della banca dati.

L'endpoint del servizio si trova al seguente indirizzo: <http://svcasdm001:8050/caisd-rest/>

Per lo svolgimento di questo progetto si renderanno necessari due metodi, in particolare:

- **rest-access**: necessario per ottenere la chiave di accesso per utilizzare il servizio.
- **cnt**: permette di ricercare i contatti ed ottenerne i diritti di accesso.

La risposta è in formato JSON oppure XML, per scegliere il tipo di standard con cui si vuole comunicare lo si deve specificare nell'header della richiesta l'elemento **Accept** e **Content-Type**.

Per ottenere la chiave d'accesso è necessario passare nell'header della richiesta anche le credenziali, bisogna infatti utilizzare l'elemento **Authorization** passando come valore la parola chiave *Basic* seguita dalla stringa codificata in base 64 *username:password*, per esempio *Base aGVsbG86d29yZA==*.

Se i valori vengono passati in modo corretto si otterrà in risposta un oggetto serializzato in *JSON* o *XML* contenente la chiave che ci permetterà di utilizzare tutti gli altri metodi del servizio, come mostrato nella porzione di codice seguente:

```
1 {
2   rest_access:{
3     @id: "401127",
4     @REL_ATTR: "401127",
5     @COMMON_NAME: "431910484",
6     link:{
7       @href: "http://svcasdm001:8050/caisd-rest/
8             rest_access/401127",
9       @rel: "self"
10    },
11    access_key: 431910484,
12    expiration_date: 1536222406
13 }
```

Per utilizzare il metodo *cnt* bisogna aggiungere nell'header anche i seguenti elementi:

- **X-AccessKey**: chiave ottenuta mediante il metodo *rest-access*.
- **X-Obj-Attrs**: Lista di campi che si vogliono ricevere in risposta.

L'applicazione di una query per filtrare gli elementi viene eseguita mediante la variabile *WC* nel *query string*, la sintassi per il filtro è quella di SQL.

Il metodo *cnt* ritorna una collections di contatti, come mostrato nella porzione di codice seguente:

Alcuni elementi sono stati rimossi per semplicità di lettura e comprensione.

```

1  {
2  collection_cnt:{
3    @COUNT:"1",
4    @START:"1",
5    @TOTAL_COUNT:"1",
6    cnt:{
7      @id:"U'1EDAE4A3F70465488160F8969CA69B83 '",
8      @REL_ATTR:"U'1EDAE4A3F70465488160F8969CA69B83 '",
9      @COMMON_NAME:"Bonomi, Niko ",
10     access_type:{
11       @id:"10002",
12       @REL_ATTR:"10002",
13       @COMMON_NAME:"Amministrazione",
14     },
15     contact_num:9952,
16     type:{
17       @id:"2307",
18       @REL_ATTR:"2307",
19       @COMMON_NAME:"Analista",
20     }
21   }
22 }
```

Come si può vedere nelle API *acctype* viene identificato da *type*.

Data Model del Plugin

Il modello in figura 3.10 è stato strutturato sulla base di quanto ritornato dalle API.

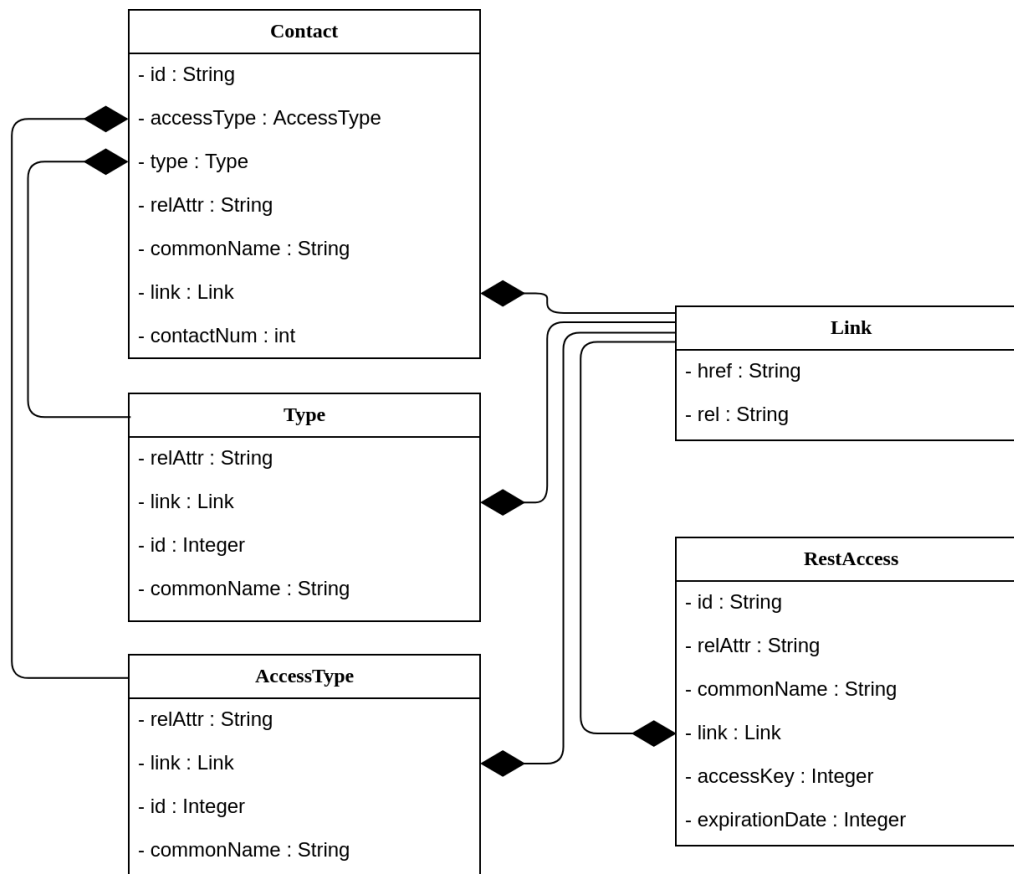


Figura 3.10: SDM DataModel

Link

Questa classe è utilizzata solo a livello di API, ogni oggetto ritornato dai servizi contiene un attributo *link*, quest'ultimo contiene un collegamento alla risorsa. La tabella 3.11 descrive i campi della classe.

L'attributo *link* di un *cnt* è composto come segue:

```

1 {
2   "cnt":{
3     "link":{
4       "@href":"http://svcasdm001:8050/caisd-rest/cnt/U'1
          EDAE4A3F70465488160F8969CA69B83'",
5       "@rel":"self"
6     }
7   }
8 }

```

Tabella 3.11: Campi classe Link

Campo	Descrizione
href	Destinazione del link
rel	Identificativo della risorsa

Contact

Questa classe descrive un contatto, il contatto in SDM è l'utente, la descrizione dei campi è visibile alla tabella 3.12

Tabella 3.12: Campi classe Contact

Campo	Descrizione
id	Identificativo del contatto in SDM
accessType	Tipo di accesso
type	Tipo di contatto
commonName	Nome completo del contatto
link	URL che identifica la risorsa nelle API
contactNum	Numero di contatto, questo è il numero collaboratore presente nel programma delle RU

Type

Identifica il tipo di contatto di un utente, la tabella 3.13 descrive i campi.

Tabella 3.13: Campi classe Type

Campo	Descrizione
relAttr	Attributo utilizzato per le relazioni
link	URL che identifica la risorsa nelle API
id	Identificativo della risorsa
commonName	Nome descrittivo della risorsa

AccessType

Identifica il tipo di accesso di un utente, la tabella 3.14 ne descrive i campi.

Tabella 3.14: Campi classe AccessType

Campo	Descrizione
relAttr	Attributo utilizzato per le relazioni
link	URL che identifica la risorsa nelle API
id	Identificativo della risorsa
commonName	Nome descrittivo della risorsa

RestAccess

Struttura dati che permette di ottenere la chiave di accesso per l'utilizzo delle API, la tabella 3.15 ne descrive i campi.

Tabella 3.15: Campi classe RestAccess

Campo	Descrizione
id	Identificativo della richiesta di accesso
relAttr	Attributo utilizzato per le relazioni
commonName	Campo alternativo per ottenere la chiave di accesso come stringa
link	URL che identifica la risorsa nelle API
accessKey	Chiave di accesso per l'utilizzo delle API
expirationDate	Timestamp che identifica la scadenza della chiave di accesso

3.2.9 LAB400

LAB400 è un software che esegue su AS400, si può accedervi anche mediante i menù di AS400, ma anche in modo diretto tramite un collegamento.

Per avere accesso al LAB400 è necessario avere un utente e il rispettivo profilo su AS400 come pure l'utente su LAB400.

Il database si trova sul server AS400, come mostrato in figura 3.11 per estrapolare le informazioni degli utenti in LAB400 sono necessarie due librerie distinte:

- LABOGPL
- COMDB

All'interno di **LABOGPL** si trovano le informazioni inerenti agli utenti, quali: nome utente, menu, numero di istituto ecc...

COMDB invece viene utilizzata per tradurre in una sigla che identifica l'istituto in modo comprensibile.

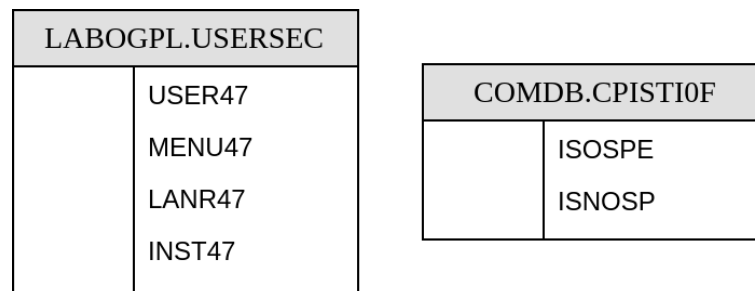


Figura 3.11: LAB400 ER

I campi nel database non sono parlanti, nella tabella 3.16 viene descritto ogni campo e il suo contenuto.

Tabella 3.16: Descrizione campi

Campo	Descrizione
USER47	Sigla utente
MENU47	Menu LAB400 assegnato
LANR47	Laboratorio di default
INST47	Numero istituto
ISNOSP	Numero istituto
ISOSPE	Sigla istituto

Data Model del Plugin

Il risultato di una *query* sulle due tabelle viene raggruppato in una sola classe come mostrato in figura 3.12

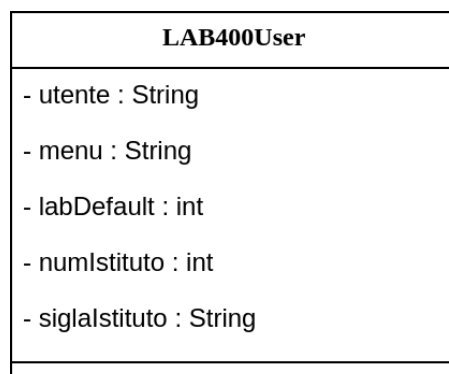


Figura 3.12: LAB400 DataModel

La tabella 3.17 descrive i campi riportati nel modello dati raffigurato in figura 3.12

Tabella 3.17: Campi classe LAB400User

Campo	Descrizione
utente	Sigla utente di LAB400
menu	Menu LAB400 assegnato
labDefault	Identificativo del laboratorio di default
numIstituto	Numero identificativo dell'istituto in LAB400
siglaIstituto	Sigla dell'istituto secondo l'id <i>numIstituto</i>

3.2.10 GECO

GECO è il software più utilizzato all'Ente Ospedaliero Cantonale, viene utilizzato per tutta una serie di operazioni mediche che variano dal dosaggio di farmaci, ammissione pazienti, gestione operazioni, ecc...

La gestione accessi di GECO è molto dettagliata, si può arrivare fino al diritto fine di una singola persona.

Gli accessi vengono gestiti tramite degli *scope*, delle *ZAM/ZAS* e dei *profili*. Le *ZAM/ZAS* sono zone in cui un determinato *profilo* viene applicato per un determinato utente, lo schema in Figura 3.13 mostra questa combinazione, con la tabella **USER** che rappresenta l'utente, **PROFILE** che rappresenta un profilo e **ZAM/ZAS** che rappresenta le zone di attività. La tabella **SCOPE** fa da ponte fra le altre entità collegandole fra loro.

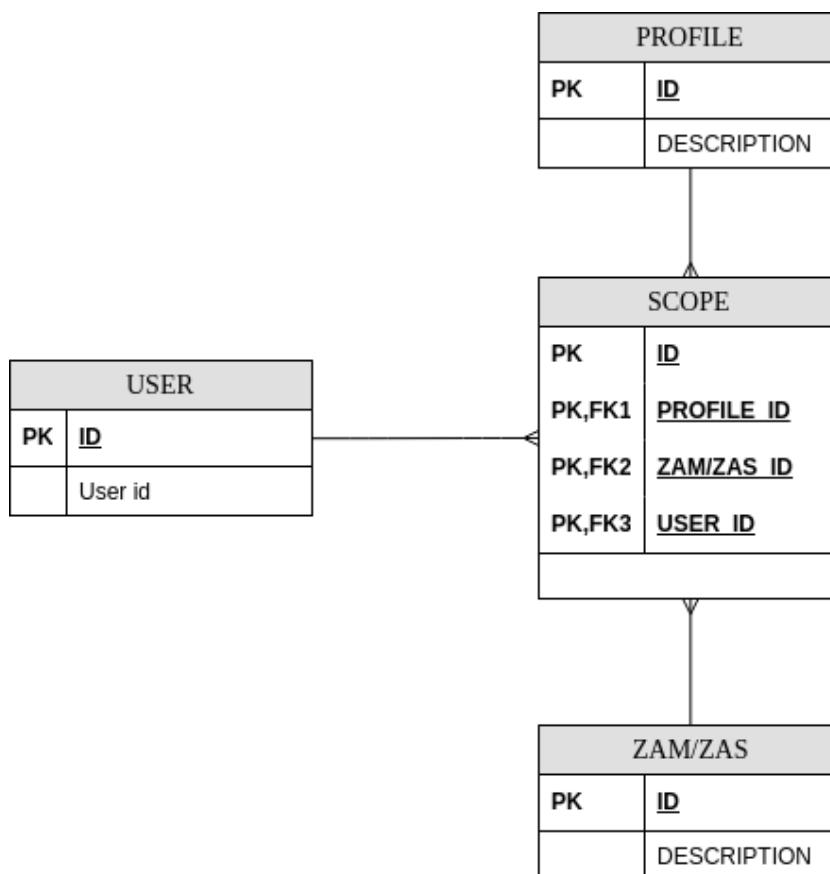


Figura 3.13: GECO ER

API

L'accesso ai dati viene eseguito mediante un webservice SOAP al seguente indirizzo:

<http://cmpwebupp1:8080/webup/UserFacade?wsdl>

Per ottenere tutte le informazioni riguardanti un utente si utilizza il metodo *getCompleteUser(userID)* passando l'id proprietario dell'utente. Quanto segue è un esempio di risposta fa *getCompleteUser*:

Il documento XML seguente è solo un estratto della risposta completa

```

1  <ns2:getCompleteUserResponse xmlns:ns2="http://web.webup.eoc.ch/">
2    <return>
3      <scopes>
4        <profile>
5          <lastupdateuserid>1</lastupdateuserid>
6          <profiletypeid>57</profiletypeid>
7          <statusreadercard>2</statusreadercard>
8          <typeabbr>FAT</typeabbr>
9          <typename>z-disattivato - Fatturazione</typename>
10         </profile>
11         <scope>
12           <enddate>2010-10-31T23:59:59+01:00</enddate>
13           <profiletypeid>57</profiletypeid>
14           <scopeareaid>415</scopeareaid>
15           <sermedareaid>25</sermedareaid>
16           <userinfoallid>8537</userinfoallid>
17         </scope>
18         <sermed>
19           <acronym>Amministrazione ORL</acronym>
20           <lastupdateuserid>12981</lastupdateuserid>
21           <longname>Zona amministrazione ORL</longname>
22           <sermedareaid>25</sermedareaid>
23           <servicechiefid>0</servicechiefid>
24           <type>ZAM</type>
25         </sermed>
26       </scopes>
27       <scopes>
28         ...
29       </scopes>
30     </return>
31 </ns2:getCompleteUserResponse>

```

Si nota dunque la relazione fra *profilo* e *ZAM/ZAS* per un determinato utente.

Gli scope da prendere in considerazione sono solo quelli validi, quindi con **enddate** supe-

riore alla data corrente.

Per ottenere l'id dell'utente è necessario invocare il metodo *getUserByMatricule(matricule)* passando in parametro il numero dipendente che figura nel programma stipendi. Qui di seguito è possibile vedere una risposta di esempio per il metodo *getUserByMatricule(matricule)*. Quanto segue è un estratto della risposta ottenuta all'invocazione del metodo:

```

1  <ns2:getUserByMatriculeResponse>
2    <return>
3      <birthdate>1994-02-14T00:00:00+01:00</birthdate>
4      <contractenddate>2200-01-01</contractenddate>
5      <contractstartdate>2014-08-01</contractstartdate>
6      <email>Niko.Bonomi@eoc.ch</email>
7      <eocid>9952</eocid>
8      <firstname>Niko</firstname>
9      <lastname>Bonomi</lastname>
10     <login>eoc9952</login>
11     <matricule>9952</matricule>
12     <userinfoallid>16766</userinfoallid>
13   </return>
14 </ns2:getUserByMatriculeResponse>

```

L'ID utente è dunque il campo *userinfoallid*

Data Model del Plugin

Il modello è stato generato in automatico mediante **wsimport**, questo comporta la generazione di molte classi.

In figura 3.14 sono rappresentati solo gli oggetti utilizzati dal progetto, per permetterne una visualizzazione pulita e ordinata.

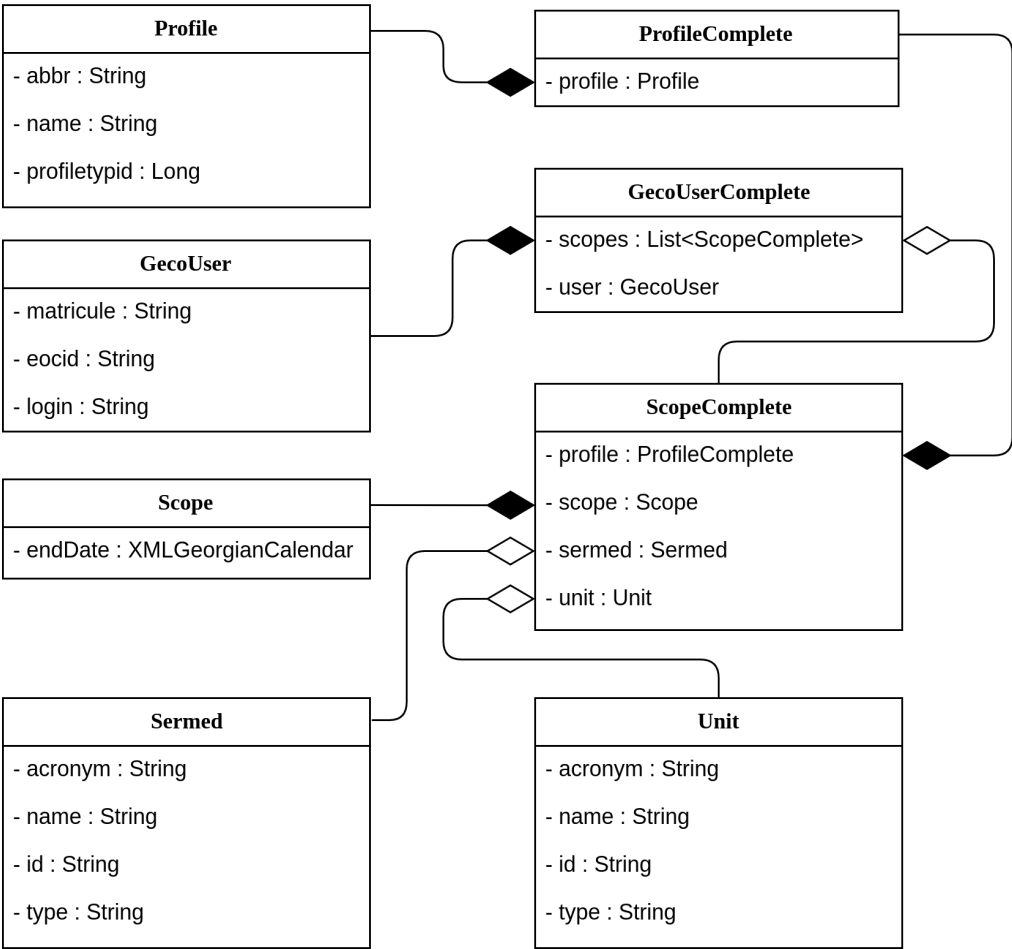


Figura 3.14: GECO DataModel

Profile

Tabella 3.18: Campi classe Profile

Campo	Descrizione
abbr	Abbreviazione del nome
name	Nome completo
profiletypeid	ID profilo

GecoUser

Tabella 3.19: Campi classe GecoUser

Campo	Descrizione
matricule	Numero collaboratore EOC
ecoid	Numero collaboratore EOC
login	Siglia di accesso EOC

GecoUserComplete

Tabella 3.20: Campi classe GecoUserComplete

Campo	Descrizione
scopes	Scopes assegnati ad un utente
user	Dettaglio specifico di un utente

Scope

Tabella 3.21: Campi classe Scope

Campo	Descrizione
endDate	Data di scadenza dello Scope

ScopeComplete

Tabella 3.22: Campi classe ScopeComplete

Campo	Descrizione
profile	Profilo relazionato allo scope
scope	Dettaglio specifico dello scope
sarmed	Sarmed assegnata allo scope
unit	Unità assegnata allo scope

Sarmed

Tabella 3.23: Campi classe Sermed

Campo	Descrizione
acronym	Abbreviazione del nome
name	Nome completo
id	Identificativo
type	Tipologia

Unit

Tabella 3.24: Campi classe Unit

Campo	Descrizione
acronym	Abbreviazione del nome
name	Nome completo
id	Identificativo
type	Tipologia

Le classi **SERMED** e **Unit** rappresentano una ZAS o una ZAM.

3.2.11 Opale

Questo software è soggetto a restrizioni da parte del produttore.

È disponibile un accesso in sola lettura su dati esportati del giorno precedente e la scrittura sul database non è concessa.

3.2.12 ViewPoint

Software per la gestione delle refertazioni di ginecologia.

Attualmente viene gestito tutto a mano, sia creazione che modifica.

Gli utenti vengono aggiunti ad alcuni gruppi, e quest'ultimi hanno le autorizzazioni.

Il database si trova sul server **EOCVPT001**.

Attualmente presso l'EOC nessuno conosce come accedere al database di quest'applicazione.

3.2.13 Isteric

Questo applicativo gestisce per intero il processo di sterilizzazione di apparecchiature mediche, dal trasporto al lavaggio e sterilizzazione.

I privilegi agli utenti vengono assegnati per livelli, un utente si può trovare solo in un livello.

Il database è molto semplice, è un *MySQL* e si trova sul server **SVSTERI001** con nome **steri_db**.

Come mostrato nella Figura 3.15, la tabella per la gestione accessi è la **utente** e contiene tutte le informazioni necessarie.

Nel campo **EOCUserName** si trova la sigla di accesso mentre in **livello** il relativo livello di accesso.

utente	
PK	<u>ID UTENTE</u>
	EOCUserName
	livello

Figura 3.15: Isteric ER

Non essendoci alcun genere di servizio web i dati verranno recuperati direttamente dal database.

Data Model del Plugin

In funzione dei dati presenti nel database il modello è stato strutturato come in figura 3.16.

IstericUser
- userID : int
- customerID : int
- name : String
- username : String
- status : int
- level : int

Figura 3.16: Isteric DataModel

La tabella 3.25 raffigura la descrizione testuale dei campi nel modello.

Tabella 3.25: Campi classe IstericUser

Campo	Descrizione
userID	ID Utente - numero collaboratore EOC
customerID	Campo per identificare il numero del cliente
name	Nome utente
username	Sigla di accesso all'applicativo
status	Stato dell'utente
level	Livello utente, identifica i diritti di un utente in Isteric

3.2.14 Cartella Pazienti Informatizzata

Spesso abbreviato con CPI, è il vecchio software per la gestione della cartella clinica informatizzata dei pazienti. Il software è in dismissione, però alcuni medici e alcune persone necessitano ancora di accederci per visionare le informazioni che non sono ancora state migrate.

Il database di questo software si trova sul server **ALSCPI** è un MS SQL. La particolarità è che per ogni istituto è stato creato un database a se stante, la struttura è sempre la medesima essendo che il software è uguale per tutti.

Sul server si trovano quindi i seguenti database:

- ORLDB - Ospedale Regionale Lugano
- OSGDB - Ospedale San Giovanni
- OBVDB - Ospedale Beata Vergine
- ODLDB - Ospedale Distrettuale La Carità
- CRNDB - Centro Riabilitazione Novaggio

La base dati è sprovvista di relazioni con chiavi esterne. Tuttavia gli accessi sono gestiti nella tabella **USERCODES** in cui è contenuto anche l'identificativo degli utenti informatici presso l'EOC, dunque come mostrato in figura 3.17, questa tabella è sufficiente.

USERCODES	
PK	<u>MEDNAME</u>
	TIPO

Figura 3.17: CPI ER

Data Model del Plugin

Ogni utente può avere uno o più *usercode*, come si può vedere in figura 3.18 il modello è stato dunque pensato in quest'ottica.

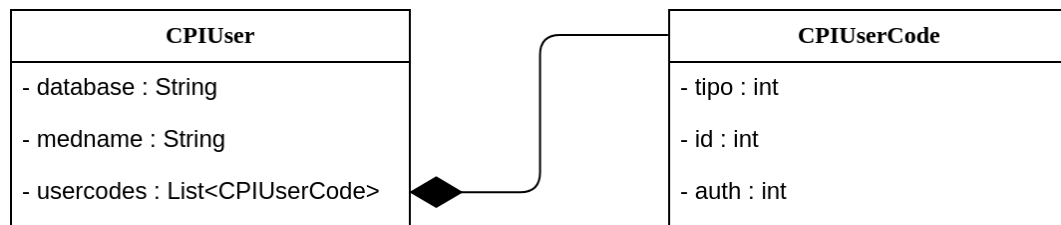


Figura 3.18: CPI DataModel

Non è possibile ottenere una descrizione testuale per ogni *userCode* perché quest'ultime sono scritte nel codice sorgente dell'applicativo.

Le tabelle 3.26 e 3.27 descrivono i campi del modello.

Tabella 3.26: Campi classe CPIUser

Campo	Descrizione
database	Database di appartenenza dell'utente
medname	Sigla di accesso dell'utente
usercodes	Codici utente

Tabella 3.27: Campi classe CPIUserCode

Campo	Descrizione
tipo	Numero che identifica il tipo di UserCode
id	Identificativo
auth	Identificativo tipologia di autenticazione

3.2.15 CARL Source

Il database di questo software si trova sul server **EOCSQL040** all'istanza **EOCSQL040**, il database è chiamato **carl_cs2**.

Gli utenti si trovano nella tabella **CSSY_USERS**, il numero identificativo dell'utente è nel campo *LOGIN*.

CARL Source integra un concetto di *attore*, questo significa che l'utente è pure un attore. Gli attori si trovano sulla tabella **CSSY_ACTOR**, la relazione è di tipo *uno-uno* mediante il

campo *ACTOR_ID* nella tabella degli utenti.

Il funzionamento è molto simile a quello di xLnet3, ovvero si definisce il tipo di diritto e dove l'utente può accedere per lavorare.

La tabella **CSSY_PROFILE** identifica il tipo di profilo dell'utente, quindi i suoi diritti sul *sito*. Il sito è invece definito nella tabella **CSSY_SITE**, quest'ultimo è relazionato con l'attore mentre il profilo è relazionato direttamente con l'utente.

La figura 3.19 mostra uno schema ER del database che illustra le relazioni fra le tabelle e i campi interessati dal progetto.

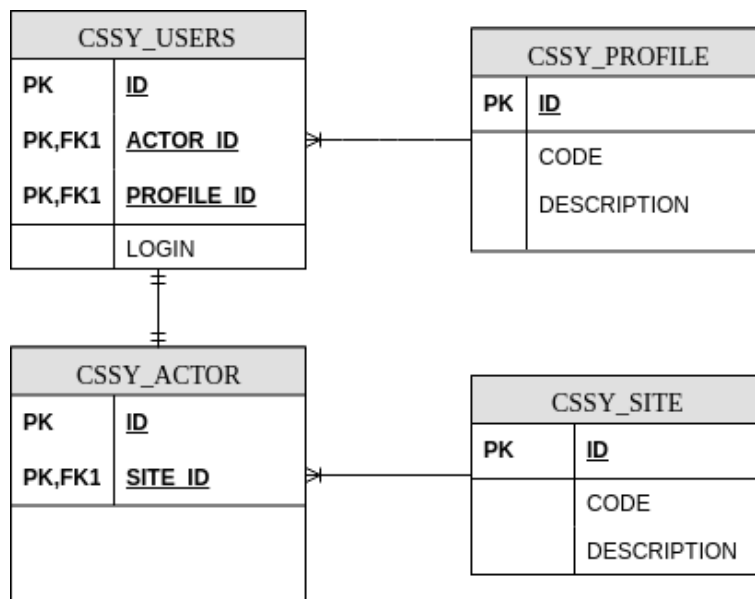


Figura 3.19: CARL Source ER

Data Model del Plugin

Il modello (figura 3.20) è stato strutturato secondo quanto è presente nel database e necessario per lo svolgimento del progetto, per semplicità d'implementazione l'attore e l'utente sono stati uniti in una sola entità.

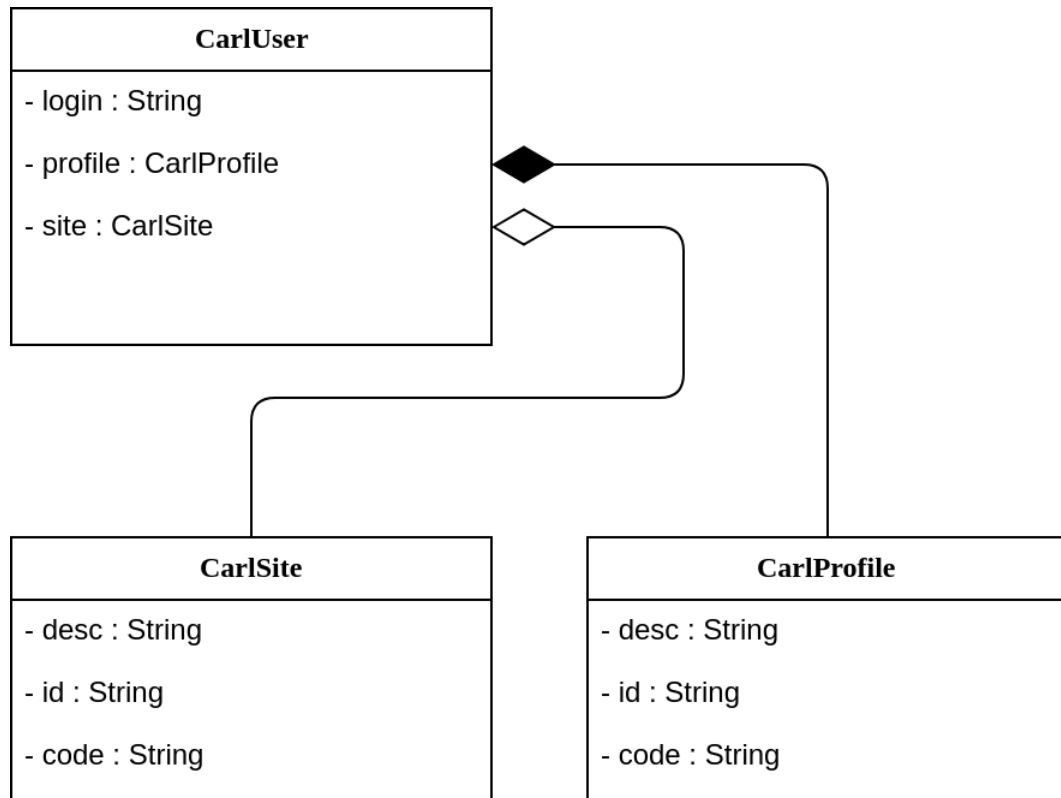


Figura 3.20: CARL Source DataModel

CarlUser

La tabella 3.28 fornisce la descrizione di tutti i campi per la classe *CarlUser*.

Quest'ultima raffigura un utente all'interno del CARL.

Tabella 3.28: Campi classe CarlUser

Campo	Descrizione
login	Sigla di accesso EOC
profile	Profilo relazionato all'utente
site	Sito relazionato all'utente

CarlSite

Questa classe rappresenta un sito nel CARL, la tabella sottostante (3.29) ne descrive i campi nel dettaglio.

Tabella 3.29: Campi classe CarlSite

Campo	Descrizione
desc	Descrizione testuale del sito
id	Identificativo
code	Codice identificativo del sito

CarlProfile

Con la classe *CarlProfile* si descrive un profilo del CARL, nella tabella 3.30 sono elencati tutti i campi con la relativa descrizione.

Tabella 3.30: Campi classe CarlProfile

Campo	Descrizione
desc	Descrizione testuale del profilo
id	Identificativo
code	Codice identificativo del profilo

Capitolo 4

Interfaccia WEB

Il *WEB server* è integrato nel CAM, l'applicativo è dunque sviluppato con *Spring MVC* e dunque esegue su un server *tomcat* sulla porta 8080.

4.1 Login

La prima schermata che si presenta all'apertura del CAM nel browser è quella di login (figura 4.1), in questa schermata è richiesto di inserire le proprie credenziali di accesso di *MS Active Directory*.

The image shows a login form for the Centralized Access Map (CAM). At the top, there is a blue circular logo with the letters 'eoc' in white. Below the logo, the text 'Centralized Access Map' is displayed. The form consists of two input fields: 'Nome utente' (Username) and 'Password'. Below these fields is a blue button labeled 'Log in'.

Figura 4.1: Login

4.2 Barra di navigazione

Questo è il componente principale che permette di navigare nell'interfaccia, come detto in precedenza i diritti sul CAM sono suddivisi in 3 gruppi, ogni gruppo ha degli accessi sempre maggiori.

In figura 4.2 è riportata un'immagine del menù.

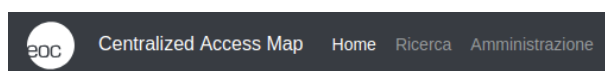


Figura 4.2: Barra di navigazione

L'accesso è suddiviso come riportato nella tabella seguente:

Tabella 4.1: Diritti sulle pagine GUI

Tipologia	Home	Ricerca	Amministrazione
Utente base	Si	No	No
Utente informatico	Si	Si	No
Amministratore	Si	Si	Si

Un utente informatico vedrà dunque solo le voci: *Home* e *Ricerca*.

Sulla barra di navigazione è disponibile anche una voce con la propria immagine, cliccando compare un menu a tendina, è possibile tornare alla schermata principale premendo il proprio nome oppure disconnettersi mediante l'apposito pulsante. Un esempio è riportato in figura 4.3.

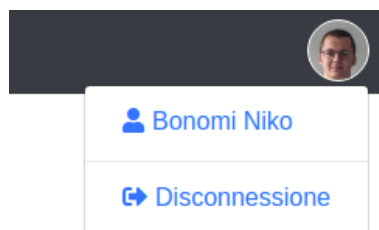


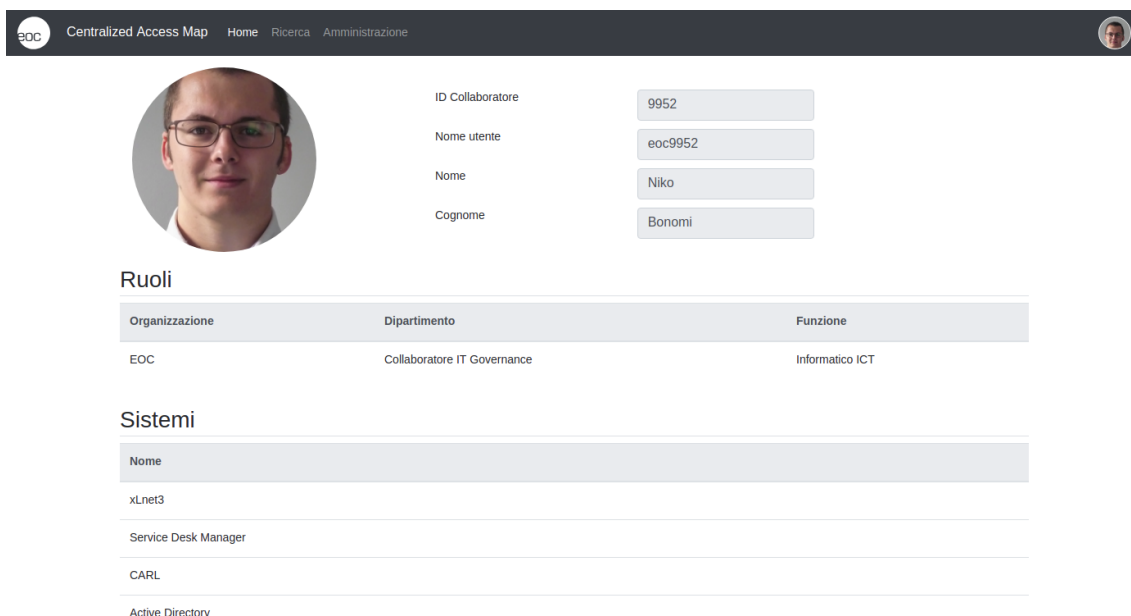
Figura 4.3: Menu utente

4.3 Home

A questa schermata tutti gli utenti hanno accesso, infatti qui si può vedere una panoramica del proprio utente, senza poter visualizzare nei dettagli i diritti fini assegnati.

Sono disponibili informazioni basilari quale: nome, cognome, id collaboratore e nome utente. È anche possibile visionare i propri ruoli ma anche in questo caso non è possibile visualizzare la mappatura dei ruoli.

Un esempio della schermata è visibile in figura 4.4.



Centralized Access Map Home Ricerca Amministrazione

ID Collaboratore 9952

Nome utente eoc9952

Nome Niko

Cognome Bonomi

Ruoli

Organizzazione	Dipartimento	Funzione
EOC	Collaboratore IT Governance	Informatico ICT

Sistemi

Nome
xLnet3
Service Desk Manager
CARL
Active Directory

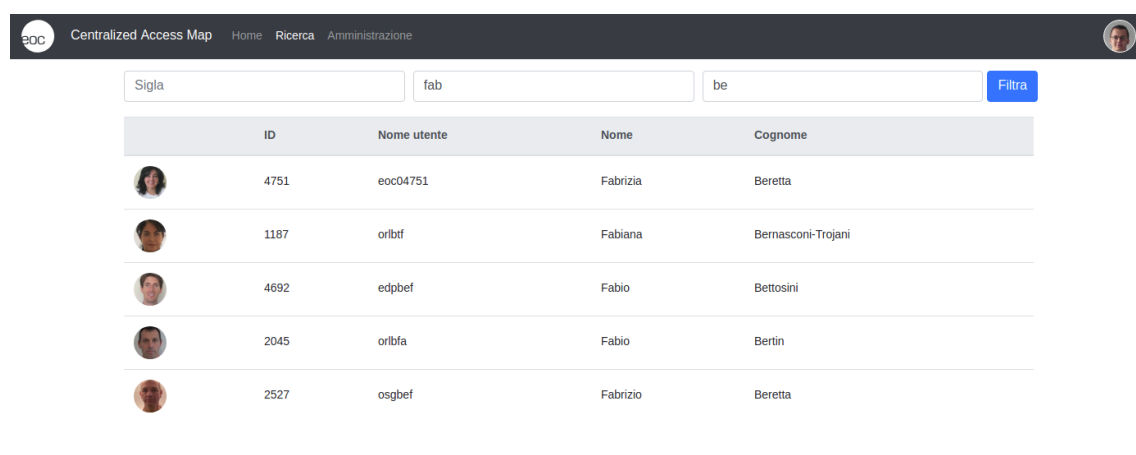
Figura 4.4: Home

4.4 Ricerca

L'accesso a questa schermata è permesso solo ad informatici ed amministratori.

4.4.1 Filtri

Qui è possibile cercare un qualunque collaboratore con un utente informatico. La schermata è visualizzabile alla figura 4.5








Sigla	ID	Nome utente	Nome	Cognome
	4751	eoc04751	Fabrizia	Beretta
	1187	oribtf	Fabiana	Bernasconi-Trojani
	4692	edpbef	Fabio	Bettosini
	2045	oribfa	Fabio	Bertin
	2527	osgbef	Fabrizio	Beretta

Figura 4.5: Ricerca

È possibile filtrare gli utenti mediante i campi:

- Sigla: username EOC collaboratore
- Nome: Nome collaboratore
- Cognome: Cognome collaboratore

Cliccando su una riga verrà mostrato il dettaglio dell'utente selezionato.

4.4.2 Risultato

Come si può notare in figura 4.6 è simile alla pagina *home* ma è possibile visualizzare un grado di dettaglio superiore.

The screenshot shows the user profile interface. At the top, there is a navigation bar with 'EOC', 'Centralized Access Map', 'Home', 'Ricerca', and 'Amministrazione'. A user profile picture is on the left. To the right, personal details are listed: ID Collaboratore (7521), Nome utente (edpmal), Nome (Luca), and Cognome (Martinelli). Below this, the 'Ruoli' (Roles) section shows a table with columns 'Organizzazione', 'Dipartimento', and 'Funzione'. The table contains one entry: EOC, Collaboratore IT Governance, and Informatico ICT. A date selector shows '17/08/2018'. The 'Sistemi' (Systems) section has a 'Carica' button and a list of systems: xLnet3 (warning icon), Service Desk Manager (checkmark), CARL (checkmark), and Active Directory (warning icon).

Organizzazione	Dipartimento	Funzione
EOC	Collaboratore IT Governance	Informatico ICT

Sistemi
xLnet3
Service Desk Manager
CARL
Active Directory

Figura 4.6: Dettaglio ricerca

Inoltre si possono anche visualizzare i ruoli anche in funzione del tempo, ovvero è possibile sapere quali sono i ruoli di un utente in una specifica data. Questo è possibile mediante il *calendario* che possiamo vedere in figura 4.7.

The screenshot shows the calendar interface. At the top, there is a date selector showing '17/08/2018'. Below it, a calendar for August 2018 is displayed. The calendar has columns for days of the week (Su, Mo, Tu, We, Th, Fr, Sa) and rows for weeks. The date 17 is highlighted in yellow. The background shows a blurred view of the user profile page.





August 2018						
Su	Mo	Tu	We	Th	Fr	Sa
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

Figura 4.7: Calendario

Oltre alla funzionalità aggiuntiva che permette di visualizzare in funzione del tempo i ruoli, è pure disponibile un maggiore dettaglio per quanto riguarda i sistemi.

Mediante un'icona è possibile capire se il sistema è conforme con quanto mappato nei ruoli. La tabella 4.2 descrive le icone disponibili.

Tabella 4.2: Icone sistema

Icona	Descrizione
	Il sistema è uguale a quanto mappato nei ruoli
	Il sistema è diverso da quanto mappato nel ruolo
	Il sistema non è mappato nei ruoli ma l'utente ha comunque accesso
	Il sistema è mappato nei ruoli ma l'utente non ha alcun accesso

Premendo il nome del sistema comparirà il diritto specifico che un utente ha su quest'ultimo, un esempio è visibile in figura 4.8.





	xLnet3				
	Service Desk Manager				
<table> <tr> <th>Tipo di accesso</th><th>Tipo di contatto</th></tr> <tr> <td>Amministrazione</td><td>Analista</td></tr> </table>		Tipo di accesso	Tipo di contatto	Amministrazione	Analista
Tipo di accesso	Tipo di contatto				
Amministrazione	Analista				
	CARL				
	Active Directory				

Figura 4.8: Risultato ricerca - sistema

Dato che ogni sistema ha la sua struttura, la visualizzazione di *Service Desk Manager* sarà differente da quella degli altri sistemi.

Il pulsante *carica* visibile nella figura 4.5 permette di caricare i sistemi dell'utente selezionato, questo vuol dire semplicemente mostrare la lista di sistemi aggiornata.

4.5 Amministrazione

Come suggerito dal titolo, in *amministrazione* solo gli amministratori di sistema potranno accedervi.

Questa sezione è suddivisa in 5 sottosezioni, come descritto in tabella 4.3.

Tabella 4.3: Sottosezioni amministrazione

Sezione	Descrizione
Annunci	Mostra gli annunci di fermi o problemi informatici attualmente in vigore
Ruoli	Gestione dei ruoli
Utenti	Gestione degli utenti
Plugin	Gestione dei plugin
Impostazioni	Da la possibilità di modificare le impostazioni di sistema

Il menu è riportato in figura 4.9.

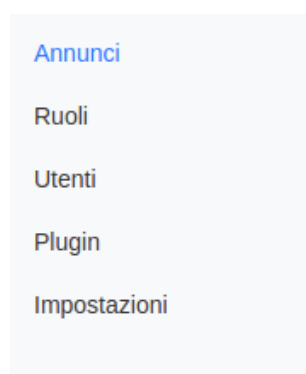


Figura 4.9: Amministrazione - News

4.5.1 Annunci

Questa sezione è unicamente utilizzata per visualizzare gli annunci attualmente in vigore. Gli annunci sono per fermi informatici, problemi o aggiornamenti. Questa funzione è molto utile in quanto il CAM interagisce con una moltitudine di sistemi, e spesso quest'ultimi potrebbero non funzionare correttamente.

La schermata degli annunci è mostrata in figura 4.10.

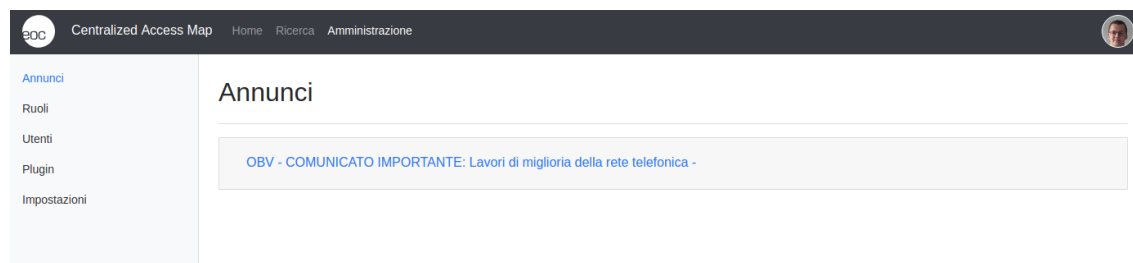


Figura 4.10: Amministrazione - News

Alla pressione di un titolo, verrà mostrato il contenuto dell'annuncio, il risultato è dunque mostrato in figura 4.11.

[OBV - COMUNICATO IMPORTANTE: Lavori di miglioria della rete telefonica -](#)

vi invitiamo a prestare la massima attenzione a questa comunicazione e a quelle che seguiranno nel corso delle prossime settimane.

Vi informiamo che il prossimo 30 agosto 2018 sono previste, a partire dalle ore 18.00, l'interruzione della linea telefonica OBV e l'installazione della nuova telefonia. Ne consegue che a partire dalle 18.00 e per alcune ore, l'ospedale non sarà raggiungibile tramite telefoni fissi, bensì solo tramite telefoni cellulari.

I reparti saranno orientati in merito al comportamento da assumere, in particolare anche per quanto attiene l'allarme REA.

L'installazione richiederà un paio di ore. Verranno cambiati tutti gli apparecchi fissi e portatili.

Alcune importanti informazioni:

- i telefoni fissi presenti nei reparti non saranno funzionanti. Ogni reparto avrà a disposizione una linea di soccorso che sarà attivata al momento del cambiamento. Verrà, pertanto, consegnata una lista contenente i numeri di telefono da contattare, unitamente ai numeri di cellulare delle persone-chiave coinvolte.
- i telefoni nelle camere dei pazienti verranno sostituiti e il personale infermieristico informa i pazienti presenti in reparto. Sul vassoio della colazione servita il 30 agosto verrà messo anche un breve comunicato da parte della Direzione.
- i telefoni DECT verranno sostituiti e i possessori sono responsabili del ritiro dei nuovi apparecchi. Seguiranno informazioni dettagliate.
- i nuovi telefoni fissi saranno posati parallelamente a quelli esistenti dai tecnici e saranno attivati al momento della migrazione.
- sono previsti dei momenti di formazione, a cui parteciperanno i collaboratori designati dai singoli reparti. Seguiranno informazioni dettagliate.
- la centralina telefonica verrà sostituita.

Vi comunichiamo, infine, che il 20 agosto 2018 procederemo ad altrettanti importanti lavori nella zona della ricezione OBV.

Figura 4.11: Amministrazione - Dettaglio news

4.5.2 Ruoli

In questa sezione è possibile gestire tutto quello che riguarda la mappatura dei ruoli all'interno dell'azienda.

La schermata che si presenta è visibile in figura 4.12.

Centralized Access Map Home Ricerca Amministrazione

Annunci
Ruoli
Utenti
Plugin
Impostazioni

Ruoli

Organizzazione Dipartimento Funzione Filtra

Precedente 1 2 3 4 5 Prossima

	Organizzazione	Dipartimento	Funzione
👤 (14)	OSG	Collaboratori anestesia	Inf. spec. in anestesia
👤 (3)	ORL	Allievo medicina 2 OIL	All. infermiere SUPSI
👤 (8)	OBV	Segretariato ambulatori medicina	Imp. amministrativo
👤 (6)	OSG	Collaboratori gruppo 3 segr. interdisciplinare e ASM ORBV	Assistente di studio medico
👤 (2) ⚠️	ORL	Capo servizio neurochirurgia	Medico caposervizio
👤 (19)	OBV	Consulente pronto soccorso	Medico accreditato
👤 (13)	ODL	Collaboratore MED C	Inf. dipl. in cure generali
👤 (35)	ORL	Collaboratore economia domestica OCL	Ausiliario economia domestica
👤 (1) ⚠️	OBV	Collaboratore diabetologia e endocrinologia	Inf. spec. clinico I
👤 (6)	OSG	Assistente traumatologia e ortopedia	Medico assistente

Precedente 1 2 3 4 5 Prossima



Figura 4.12: Amministrazione - Ruoli

Qui è possibile filtrare i ruoli mediante i campi proposti in alto:

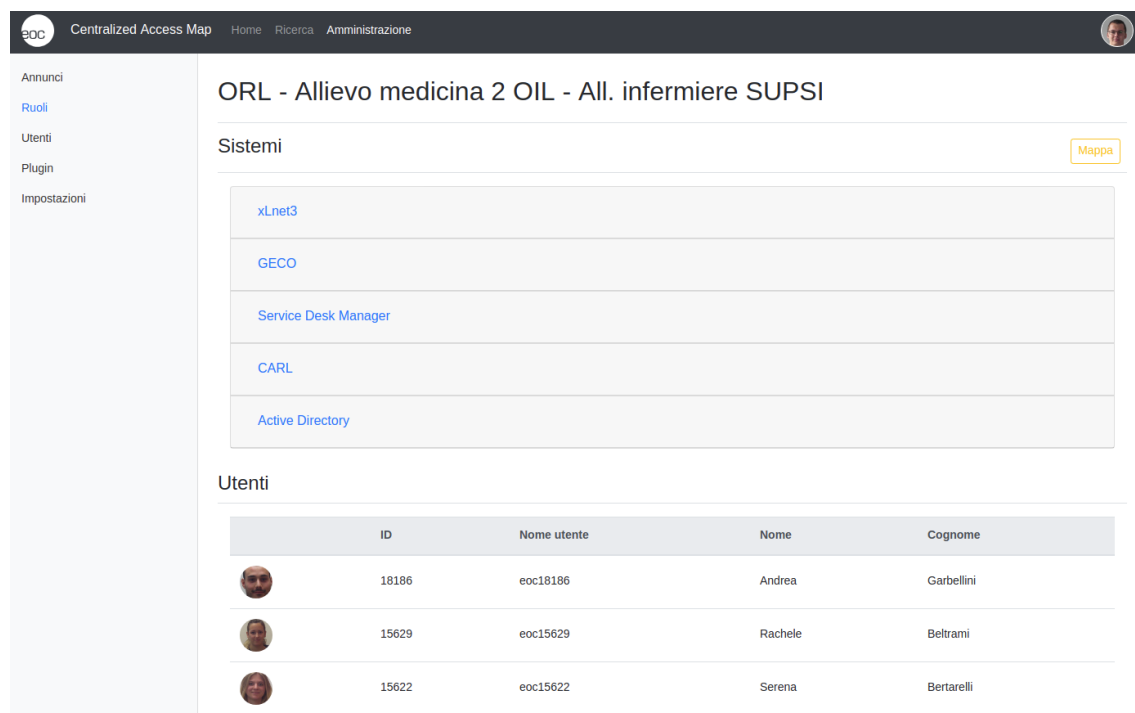
- Organizzazione
- Dipartimento
- Funzione

La lista di risultati contiene anche delle icone che permettono di visualizzare rapidamente i problemi e quanti utenti sono associati ad un determinato ruolo. Le icone sono descritte in tabella 4.4

Tabella 4.4: Icone lista ruoli

Icona	Descrizione
 (3)	Quantità di utenti nel ruolo, in questo caso 3
	Il ruolo non contiene abbastanza utenti per creare una mappatura affidabile

Premendo su un ruolo si verrà reindirizzati al suo dettaglio, una schermata d'esempio è visibile alla figura 4.13.



ORL - Allievo medicina 2 OIL - All. infermiere SUPSI

Sistemi Mappa

- xLnet3
- GECO
- Service Desk Manager
- CARL
- Active Directory

Utenti

ID	Nome utente	Nome	Cognome
18186	eoc18186	Andrea	Garbellini
15629	eoc15629	Rachele	Beltrami
15622	eoc15622	Serena	Bertarelli

Figura 4.13: Amministrazione - Dettaglio ruolo

La pagina mette a disposizione due pulsanti principali:

- Mappa

- Carica

Mediante il pulsante *Mappa* è possibile eseguire una mappatura completa di tutto il ruolo, mentre il pulsante *Carica* permette di caricare gli utenti che appartengono al ruolo e mostrarli nella sezione *Utenti*.

Premendo il pulsante *Carica* verranno mostrati tutti gli utenti che in questo momento hanno assegnato il ruolo selezionato, al contrario degli utenti, attualmente non è possibile avere una vista temporale.

Premendo un elemento della lista si verrà reindirizzate al suo dettaglio.

Alla pressione del pulsante *Mappa* verrà mostrato la seguente finestra modale:

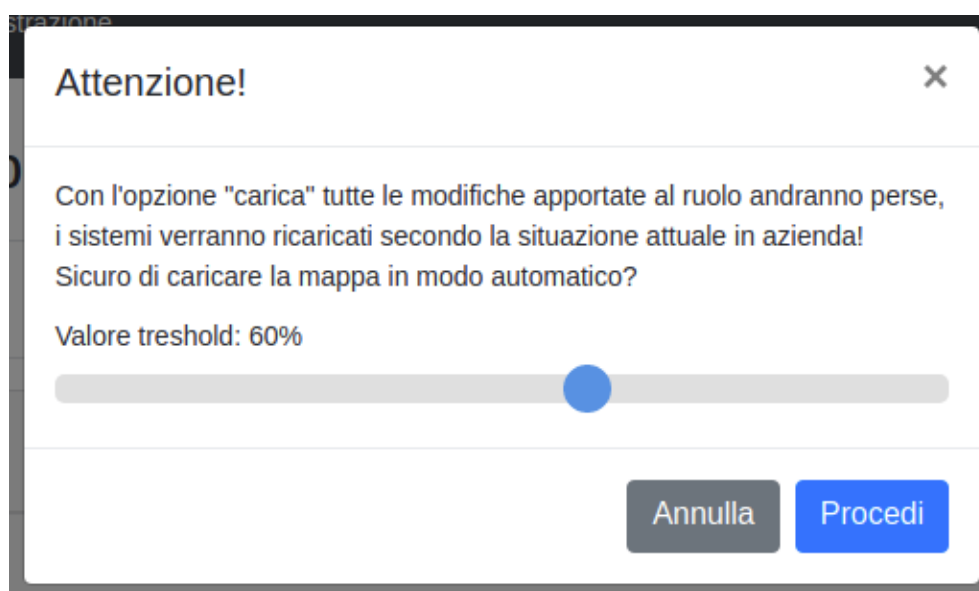


Figura 4.14: Amministrazione - Modale mappatura ruolo

Nel modale è possibile selezionare il valore di threshold preferito e quindi avviare la mappatura.

Il valore di threshold serve per specificare al plugin quanti utenti devono avere un determinato diritto per essere mappato, per esempio: se ci sono 10 utenti e un threshold di 60% almeno 6 di questi devono condividere lo stesso diritto, se lo condividono quest'ultimo viene mappato.

In questo modo è possibile scegliere con quanta precisione si vuole creare la mappatura. Come evidenziato nella finestra modale, qualunque modifica apportata alla mappatura del ruolo selezionato verrà persa.

Se il ruolo ha una mappa, la vedremo sotto la voce *sistemi*.

Per visualizzarne il dettaglio è necessario premere sul nome del sistema, allora comparirà la struttura della mappatura, un esempio è mostrato in figura 4.15.

xLnet3	
Gruppo	
Codice	Descrizione
PHL4	Utenti reparti
Unità	
Unità	Sito
Anestesia BZ (Stabile A 1*) - ANE	Ospedale Regionale Bellinzona e Valli - OSG
Terapia del dolore BZ (Stabile k 5*) - CDO	Ospedale Regionale Bellinzona e Valli - OSG
Modifica	
GECO	

Figura 4.15: Amministrazione - Sistema mappato al ruolo

Nello spazio riservato al dettaglio di un sistema viene mostrato pure un pulsante *Modifica*. Questo bottone permette di modificare la mappatura per il sistema selezionato. La modifica verrà eseguita in una nuova finestra.

Di seguito, in figura 4.16, è disponibile un esempio di modifica:

EOC

Centralized Access Map

Home

Ricerca

Amministrazione

Annunci

Ruoli

Utenti

Plugin

Impostazioni

OSG - Collaboratori anestesia - Inf. spec. in anestesia

Modifica - xLnet3

Salva Annulla

Gruppo Utenti reparti

Unità

Unità EOC -- Organico personale - STAFF

Aggiungi

Unità	Sito
Anestesia BZ (Stabile A 1*) - ANE	Ospedale Regionale Bellinzona e Valli - OSG
Terapia del dolore BZ (Stabile k 5*) - CDO	Ospedale Regionale Bellinzona e Valli - OSG

Figura 4.16: Amministrazione - Modifica Sistema mappato al ruolo

Questa pagina è personalizzata dal plugin stesso, pertanto non è univoca fra tutti i sistemi. I pulsanti *Salva* e *Annulla* permettono di salvare o annullare le modifiche apportate alla mappatura.

Una volta salvate le modifiche si verrà reindirizzati al ruolo precedentemente visualizzato. In caso di errore, si verrà notificati mediante un errore nella parte superiore della pagina.

Non tutti i sistemi sono modificabili, questo dipende dallo sviluppatore del plugin. In caso che un plugin non supporti la modifica il bottone non sarà cliccabile e sarà seguito da un messaggio che avvisa dell'impossibilità di modifica.

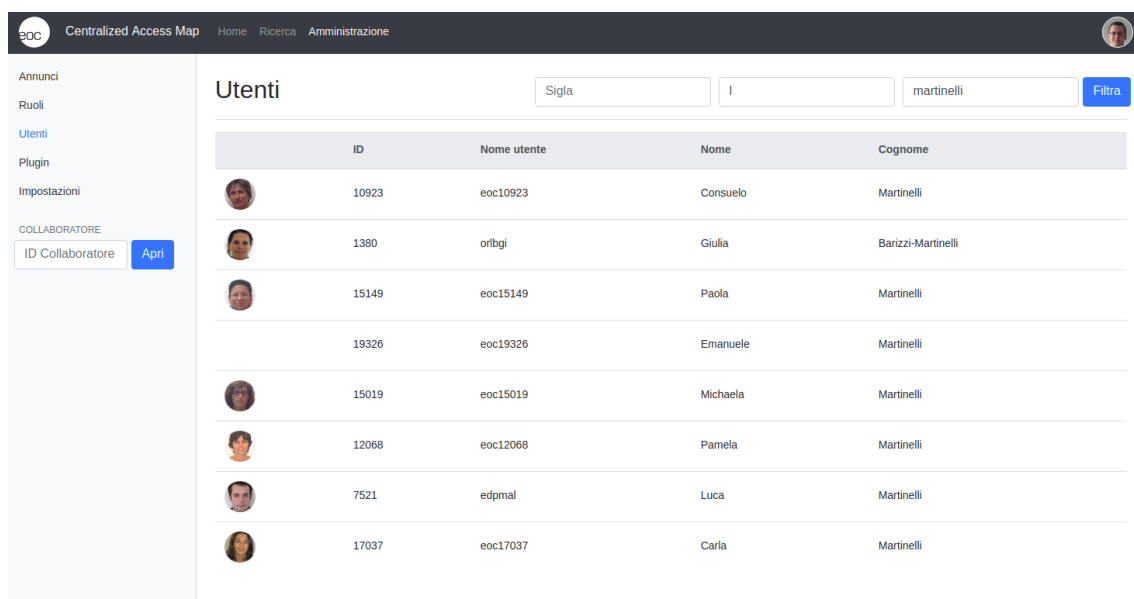
4.5.3 Utenti

La sezione dedicata agli utenti permette di manipolare i diritti degli utenti, in questo modo sarà possibile allineare gli utenti alla mappa, oppure concedere diritti speciali.

All'apertura della voce verrà mostrata una ricerca mediante i campi:

- Sigla
- Nome
- Cognome

La schermata, figura 4.17, è dunque la seguente:



ID	Nome utente	Nome	Cognome
10923	eoc10923	Consuelo	Martinelli
1380	orlbgi	Giulia	Barizzi-Martinelli
15149	eoc15149	Paola	Martinelli
19326	eoc19326	Emanuele	Martinelli
15019	eoc15019	Michaela	Martinelli
12068	eoc12068	Pamela	Martinelli
7521	edpmal	Luca	Martinelli
17037	eoc17037	Carla	Martinelli

Figura 4.17: Amministrazione - Utenti

Premendo su un risultato nella lista è possibile visualizzarne il dettaglio.

Oltre alla ricerca, nella pagina è anche disponibile la possibilità di accedere direttamente al dettaglio di un utente, per fare ciò è necessario conoscere l'*ID Collaboratore* ed inserirlo nell'apposito campo nel menù di sinistra, premendo il tasto *Apri* si verrà reindirizzati al dettaglio dell'utente selezionato.

Il dettaglio di un utente è molto simile a quanto già visto nella sezione *Ricerca*, in questo caso però si possono trovare molte più funzioni, una su tutte la modifica.

Un esempio di quello che è il dettaglio è visibile nella figura 4.18 (per semplicità di visualizzazione, dalla figura, sono rimosse le parti anagrafiche dell'utente):

The screenshot displays a user management interface. At the top, there is a 'Ruoli' (Roles) section with a date input field set to '17/08/2018'. Below this is a table with three columns: 'Organizzazione', 'Dipartimento', and 'Funzione'. The table contains one row with the values 'EOC', 'Collaboratore IT Governance', and 'Informatico ICT'. Below the table is a 'Sistemi' (Systems) section with two buttons: 'Carica' (Load) and 'Allinea tutto' (Align all). Below the buttons is a list of systems. The first system is 'xLnet3' with a blue triangle icon. The second system is 'Service Desk Manager' with a yellow 'x' icon. Below the list is a table with two columns: 'Tipo di accesso' (Access type) and 'Tipo di contatto' (Contact type). The table contains one row with the values 'Amministrazione' and 'Utente'. Below the table are four buttons: 'Modifica' (Modify), 'Compara' (Compare), 'Allinea' (Align), and 'Revoca' (Revoke).

Organizzazione	Dipartimento	Funzione
EOC	Collaboratore IT Governance	Informatico ICT

Sistemi

Carica Allinea tutto

xLnet3

Service Desk Manager

Tipo di accesso	Tipo di contatto
Amministrazione	Utente

Modifica Compara Allinea Revoca

Figura 4.18: Amministrazione - Utente

Come si può notare dalla figura, anche in questo caso è possibile ottenere una vista temporale di quello che sono i ruoli dell'utente e caricare i suoi sistemi mediante il pulsante *carica*.

In aggiunta alle funzionalità già note troviamo il pulsante *Allinea tutto*.

Questo bottone permette di allineare tutti i sistemi dell'utente secondo quanto mappato nei suoi ruoli, questo vuol dire che tutti i sistemi verranno modificati per combaciare con la mappa.





Questo per dare la possibilità all'amministratore di allineare rapidamente un utente secondo gli standard aziendali.

Ovviamente questo funziona solo per i sistemi il quale plugin supporta la scrittura.

Per ogni sistema inoltre, è presente una pulsantiera specifica. Quest'ultima dipende dalle differenze del sistema rispetto alla mappa dei ruoli.

La tabella mostra quali pulsanti saranno visibili in rapporto alle differenze dal ruolo.

Tabella 4.5: Mappatura icone - bottoni

Icona	Modifica	Compara	Allinea	Revoca
	Si	No	No	Si
	Si	Si	Si	Si
	Si	No	No	Si
	No	No	No	No

Come si può notare nel caso in cui non ci sono differenze dalla mappa non è possibile nè comparare nè allineare. Questo ovviamente perché il sistema è identico a quello presente nella mappa.

Il ragionamento si applica anche nel caso in cui il sistema è un'extra rispetto a quelli richiesti, questo perché non c'è nulla nella mappa con cui comparare il sistema.

Se il sistema fosse diverso dalla mappa è possibile utilizzare gli altri bottoni.

Il caso in cui l'utente non ha un sistema che invece è mappato non è possibile eseguire alcuna attività. Questo perché questa funzionalità non è ancora stata implementata.

Il pulsante *Modifica* permette di modificare il diritto di un utente nel sistema selezionato, questo andrà dunque ad impattare sull'operabilità dell'utente in caso che si selezionano diritti errati!

La form di modifica è uguale a quanto mostrato in figura 4.16.

Il pulsante *Compara* permette di visualizzare le differenze effettive fra il diritto dell'utente e quello mappato mediante una finestra modale.

Un esempio viene mostrato in figura 4.19.



Figura 4.19: Amministrazione - Utente comparazione diritti

A sinistra troviamo i diritti attualmente assegnati all'utente mentre a destra quelli che dovrebbero essergli assegnati secondo la mappatura.

Il pulsante *Allinea* assegna all'utente i diritti mappati per il sistema selezionato. Questi diritti verranno assegnati direttamente nel sistema.

Il pulsante *Revoca* permette di rimuovere il sistema all'utente, in poche parole revoca ogni accesso. Attualmente questa funzionalità non è ancora implementata, un errore verrà mostrato nel caso in cui il pulsante verrà premuto.

4.5.4 Gestione dei Plugin

In questa sezione è possibile manipolare tutti i plugin installati, ed installarne di nuovi.

Come detto in precedenza vi sono 2 tipi di plugin, quelli che interagiscono con sistemi che l'utente utilizza per lavorare e quelli in cui sono contenuti i dati degli utenti, ovvero i sistemi delle *risorse umane*.

Da questa seconda categoria verranno estratti i ruoli lavorativi e gli utenti per permettere al CAM di funzionare correttamente.

I *plugin* delle risorse umane sono identificate da un'icona raffigurante due utenti e un'ingranaggio come mostrato nella figura (4.20) successiva:



Figura 4.20: Amministrazione - Icona plugin risorse umane

La schermata principale di questa sezione è visibile nell'immagine 4.21.

The screenshot displays the 'Plugins' management interface. On the left, a sidebar lists navigation options: Annunci, Ruoli, Utenti, Plugin (highlighted), Impostazioni, and a section titled 'INSTALLATI' containing xpertLine, xLnet3, GECO, CARL, Active Directory, MedStat, iSteric, AS400, LAB400, WinScribe, and Service Desk Manager. The main area is titled 'Plugins' and features a table of installed plugins. A green 'Aggiungi' button is located in the top right corner of the table area.

Nome	Versione in uso
xpertLine	0.1.11
xLnet3	0.2.22
GECO	0.3.8
CARL	0.1.12
Active Directory	0.4.16
MedStat	1.2.2
iSteric	0.3.3
AS400	0.2.7
LAB400	0.1.2
WinScribe	0.1.2
Service Desk Manager	0.4.18

Figura 4.21: Amministrazione - Plugins

Come si può notare nell'immagine sopra, solo un plugin ha l'icona descritta in precedenza, questo perché solo un plugin può essere *eletto* come plugin per le risorse umane.

La schermata mostra una lista dei plugin installati con la loro versione attualmente in uso, sulla sinistra si trova la stessa lista, quest'ultima servirà per un accesso diretto da qualunque schermata di questa sezione.

Premendo il pulsante *Aggiungi* verrà proposto all'utente di aggiungere un nuovo plugin, a questo stadio si crea solo lo *slot*, il plugin vero e proprio verrà caricato nella schermata successiva.

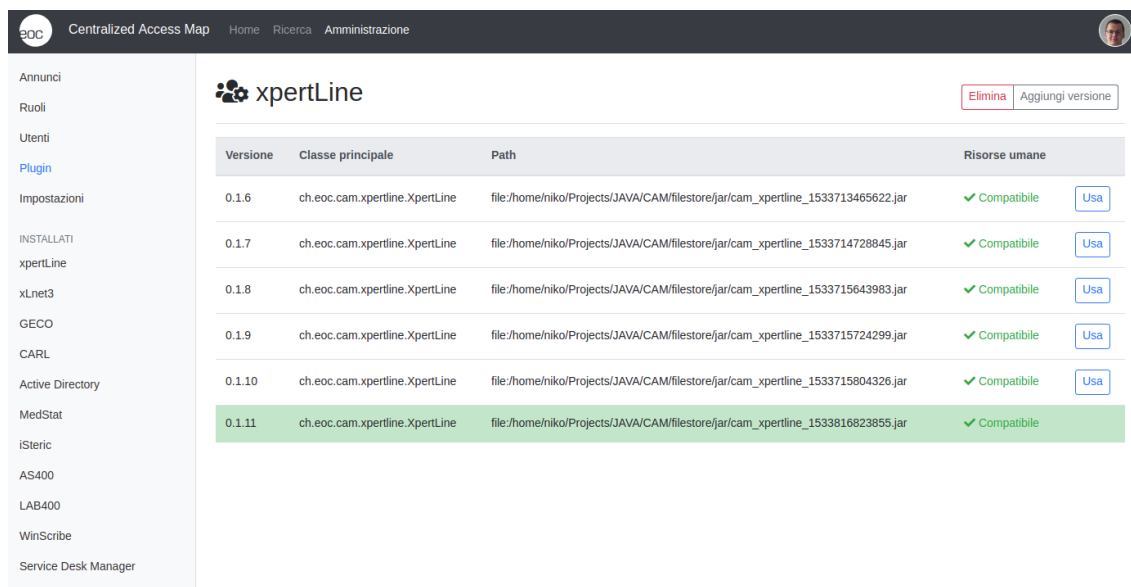
La creazione di un nuovo plugin avviene mediante una finestra modale come mostrato in figura 4.22.



Figura 4.22: Amministrazione - Nuovo Plugin

Premendo il tasto *Salva* il nuovo plugin verrà aggiunto alla lista.

Selezionando un plugin dalla lista (figura 4.21) si viene reindirizzati alla pagina di dettaglio del plugin, come si può notare in figura 4.23.



Versione	Classe principale	Path	Risorse umane
0.1.6	ch.eoc.cam.xpertline.XpertLine	file:/home/niko/Projects/JAVA/CAM/filestore/far/cam_xpertline_1533713465622.jar	✓ Compatibile Usa
0.1.7	ch.eoc.cam.xpertline.XpertLine	file:/home/niko/Projects/JAVA/CAM/filestore/far/cam_xpertline_1533714728845.jar	✓ Compatibile Usa
0.1.8	ch.eoc.cam.xpertline.XpertLine	file:/home/niko/Projects/JAVA/CAM/filestore/far/cam_xpertline_1533715643983.jar	✓ Compatibile Usa
0.1.9	ch.eoc.cam.xpertline.XpertLine	file:/home/niko/Projects/JAVA/CAM/filestore/far/cam_xpertline_1533715724299.jar	✓ Compatibile Usa
0.1.10	ch.eoc.cam.xpertline.XpertLine	file:/home/niko/Projects/JAVA/CAM/filestore/far/cam_xpertline_1533715804326.jar	✓ Compatibile Usa
0.1.11	ch.eoc.cam.xpertline.XpertLine	file:/home/niko/Projects/JAVA/CAM/filestore/far/cam_xpertline_1533816823855.jar	✓ Compatibile Usa

Figura 4.23: Amministrazione - Dettaglio Plugin

In questa schermata, nel caso in cui il plugin fosse abilitato alle risorse umane, si può notare ancora l'icona descritta in precedenza.

Nella tabella centrale vengono esposte tutte le versioni attualmente disponibili per il plugin selezionato.

Nella colonna *Risorse Umane* viene evidenziato se la versione supporta anche le funzionalità di risorse umane, se la scritta *Compatibile* è presente allora la versione supporta le RU¹

Mediante il pulsante *Usa* possiamo selezionare la versione da utilizzare, questo può essere utile in caso di *rollback* per una qualunque ragione.

Il pulsante *Aggiungi versione*, permette di caricare un nuovo *jar* come nuova versione, questo viene eseguito mediante una finestra modale mostrata in figura 4.24.

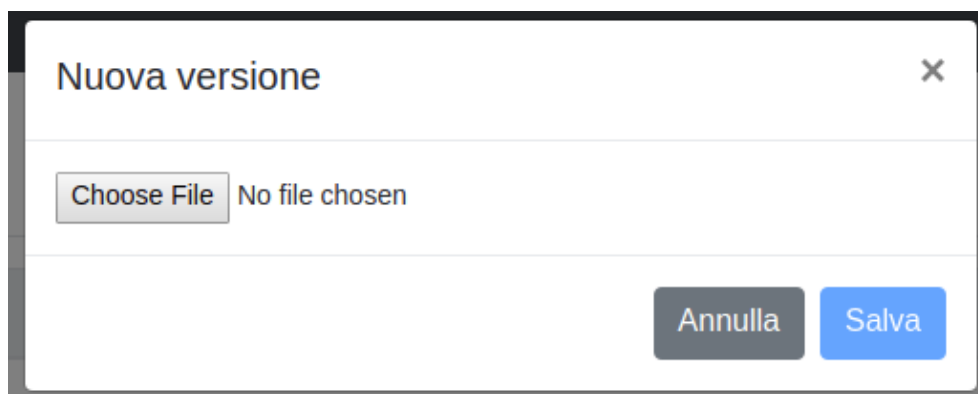


Figura 4.24: Amministrazione - Aggiunta nuova versione al plugin

Il solo scopo di questa finestra è quello di selezionare un nuovo *jar* e caricarlo sulla piattaforma.

Quando il nuovo *jar* verrà caricato, la sua versione diventerà quella di default per il plugin.

Per designare un nuovo plugin per le risorse umane, è prima necessario caricare un *jar* con il supporto per quest'ultima, poi premere il pulsante *Imposta come HR*, ed in seguito confermare l'azione mediante la finestra modale che si presenterà.



Figura 4.25: Amministrazione - Selezione plugin per il supporto alle risorse umane

Il bottone sarà visibile solo nel caso in cui l'attuale plugin non è già designato per il supporto alle RU e che la versione in uso abbia il supporto alle RU.

Il terzo tasto, *Elimina* rimuoverà dal sistema il plugin.

¹ Risorse Umane

4.5.5 Impostazioni

Le impostazioni sono suddivise in 3 voci:

- Mappa
- Livelli di accesso
- Active Directory

Come mostrato in figura, la pagina principale delle impostazioni mostra semplicemente un elenco delle voci disponibili.

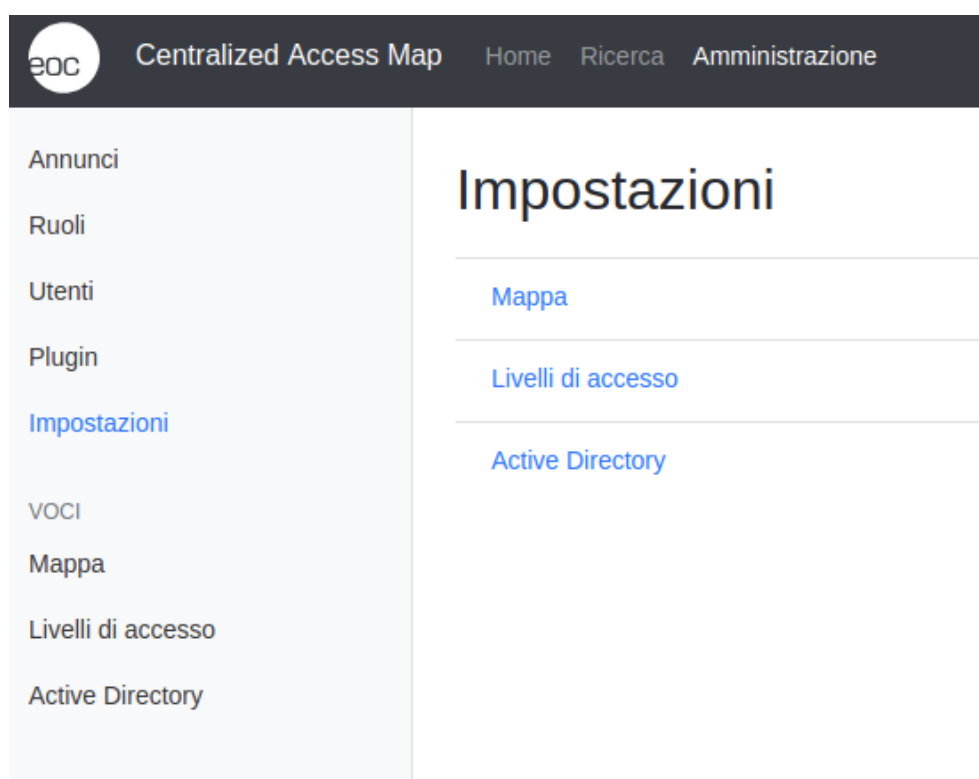


Figura 4.26: Amministrazione - Impostazioni

Mappa

Questa voce permette di generare la mappa al tempo 0², oppure ri-aggiornarla da capo. Come mostrato nella figura seguente (4.27), questa operazione sovrascriverà tutto quello che attualmente è stato modificato nella mappatura.

²Quando ancora non è presente nulla nel CAM

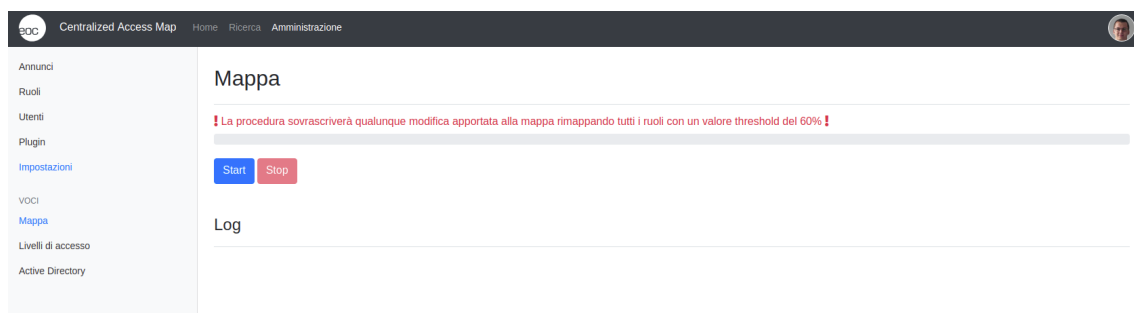


Figura 4.27: Amministrazione - Generatore mappa

Premendo il tasto *Start* si avvia la procedura, sotto al titolo *Log* verranno visualizzati i log dell'avanzamento.

La procedura può essere fermata in qualunque momento premendo il tasto *Stop*.

La funzionalità è stata implementata mediante chiamate *AJAX* alle API, vedi capitolo 5 alla sezione 5.5.

Livelli di accesso

Qui è possibile selezionare quali gruppi Active Directory sono relazionati agli accessi nel CAM.

The screenshot shows the 'Livello di accesso' (Access Level) configuration page in the Centralized Access Map (CAM) interface. The page has a dark header with the 'eoc' logo and navigation links: 'Home', 'Ricerca', and 'Amministrazione'. A left sidebar contains a menu with items: 'Annunci', 'Ruoli', 'Utenti', 'Plugin', 'Impostazioni' (highlighted in blue), 'VOCI', 'Mappa', 'Livelli di accesso' (highlighted in blue), and 'Active Directory'. The main content area is titled 'Livello di accesso' and contains three sections: 'Amministrazione' with a text input field containing 'F_ICT_Governance_COLLABORATORE', 'Ricerca' with a text input field containing 'Tutti gli utenti EOC', and 'Servizi' with a text input field containing 'F_ICT_Governance_COLLABORATORE'. A blue 'Salva' (Save) button is located at the bottom of the form.

Figura 4.28: Amministrazione - Livelli di accesso

Come mostrato in figura 4.28 si possono inserire quali gruppi AD corrisponderanno a quale tipologia di accesso.

Si può pure notare il campo *Servizi*, questo identifica quale gruppo ha accesso alle API elencate nel capitolo 5.

Active Directory

In questa schermata è possibile impostare i valori per collegare il CAM con AD e permettere l'accesso agli utenti.

La figura seguente mostra la schermata appena descritta:

Centralized Access Map Home Ricerca Amministrazione

Annunci
Ruoli
Utenti
Plugin
Impostazioni
VOCI
Mappa
Livelli di accesso
Active Directory

Active Directory

Dominio
EOC.NET

URL
ldap://eoc.net:389

Root Dn
OU=Istituti,DC=EOC,DC=NET

Salva

Figura 4.29: Amministrazione - Active Directory

Nel campo *Dominio* è necessario inserire il dominio aziendale configurato in AD, in *URL* va inserito l'URL del server Active Directory aziendale.

In *Root Dn* è necessario inserire la *Dn* di base in cui cercare gli utenti da autenticare.

Capitolo 5

Application Programming Interface

Le API del CAM permettono una gestione completa di tutta la piattaforma mediante chiamate http.

L'endpoint di accesso è *http://HOST:8080/api*

Gli oggetti principali sono quattro:

- Plugin
- Utenti
- Ruoli
- Annunci

Gli annunci vengono ottenuti da *Service Desk Manager* il quale è il sistema centrale per la gestione degli annunci informatici all'utenza, tuttavia il CAM mette a disposizione l'ottenimento degli annunci in modo semplice e veloce.

5.1 Autenticazione

Avviene tramite autenticazione **basic**, mediante le credenziali Active Directory.

Per potersi autenticare mediante *Basic* è necessario aggiungere l'header **Authorization** alla richiesta.

Il valore deve essere *Basic credenziali*, dove *credenziali* è la combinazione *username:password* codificato in *base64*.

Un esempio è il seguente:

Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==

Questo header deve essere fornito per tutte le richieste.

5.2 Risposta

Tutti gli endpoint rispondono con un oggetto comune, ovvero **APIResult**.

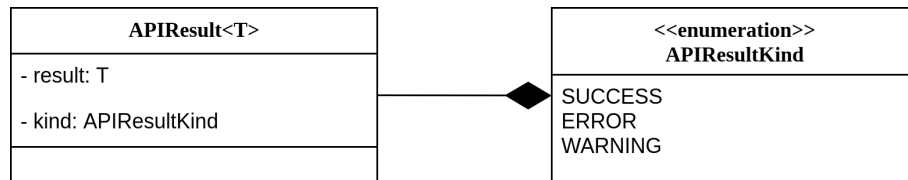


Figura 5.1: UML APIResult

Questo permette al richiedente di ottenere subito l'esito della sua richiesta controllando il campo *kind*.

Come mostrato in figura 5.1 ci sono 3 tipologie di *kind*:

- SUCCESS
- ERROR
- WARNING

Dove **SUCCESS** identifica una richiesta andata a buon fine, **ERROR** rappresenta un errore ed in fine **WARNING** rappresenta una richiesta andata a buon fine con un avviso.

Di seguito è riportato un esempio di risposta:

```

1 {
2   "kind": "SUCCESS",
3   "result": [
4     {
5       "username": "eoc5023",
6       "collabID": 5023,
7       "name": "Nikolina",
8       "surname": "Momcilovic"
9     },
10    {
11      "username": "eoc9952",
12      "collabID": 9952,
13      "name": "Niko",
14      "surname": "Bonomi"
  
```

```
15     },
16     {
17         "username": "eoc9376",
18         "collabID": 9376,
19         "name": "Nikola",
20         "surname": "Keller"
21     }
22 ]
23 }
```

5.3 Plugin

Con questo endpoint è possibile manipolare tutto quello che riguarda i plugin.

L'URL per questi metodi è il seguente:

http://HOST:8080/api/plugin

Tabella 5.1: API - Plugin

Endpoint	Tipologia	Parametri	Descrizione
/	PUT	name	name: Nome plugin, crea un nuovo plugin
/all	GET		Ottenere una lista di tutti i plugin installati con le relative versioni
/id	GET	id	Ritorna il dettaglio di un plugin
/id	DELETE	id	Elimina il plugin selezionato
/id/new	POST	id, file	Carica una nuova versione (jar) per il plugin specificato
/id/default	PUT	id, version	version: versione da utilizzare, imposta la versione selezionata come default per il plugin
/id/property/{prop}	GET	id, prop	id: ID plugin prop: Nome property Ritorna il risultato di <i>getProperty</i> del plugin selezionato.

5.4 Utenti

Con questo endpoint è possibile manipolare tutto quello che riguarda gli utenti all'interno del CAM.

L'URL per questi metodi è il seguente:

http://HOST:8080/api/user

Tabella 5.2: API - Utenti

Endpoint	Tipologia	Parametri	Descrizione
/find	GET	username, name, surname	Esegue una ricerca di tutti gli utenti
/id/roles	GET	id, date	Ritorna tutti i ruoli di un utente in una data specifica, il parametro date è facoltativo. Il formato della data è: <i>dd/MM/yyyy</i>
/id/legacysystems	GET	id	Ritorna tutti i sistemi di un utente, con i relativi diritti
/id/legacysystems/{pluginId}	GET	id, pluginID	Ritorna il dettaglio del singolo sistema specificato in parametro per l'utente selezionato
/id/legacysystems/{pluginId}	POST	id, pluginID, rights	Viene assegnato il diritto passato in parametro (rights) in forma di json all'utente selezionato mediante il plugin passato in parametro
/id/assign/own_role	PUT	id, pluginID	Assegna i diritti dei sistemi mappati ai ruoli dell'utente, se pluginID viene passato allora vengono assegnato solo per il sistema elaborato dal plugin
/id/assign/own_role_dated	PUT	id, date pluginID	Assegna i diritti dei sistemi mappati ai ruoli assegnati all'utente alla data specificata, se pluginID viene passato allora vengono assegnato solo per il sistema elaborato dal plugin
/id/assign/custom_role	PUT	id, roles, pluginID	Assegna i diritti dei sistemi mappati ai ruoli passati in parametro, se pluginID viene passato allora vengono assegnato solo per il sistema elaborato dal plugin. Roles è una lista json nel body della richiesta

5.5 Ruoli

Con questo endpoint è possibile manipolare tutto quello che riguarda i ruoli all'interno del CAM.

L'URL per questi metodi è il seguente:

http://HOST:8080/api/role

Tabella 5.3: API - Ruoli

Endpoint	Tipologia	Parametri	Descrizione
/loadAll	GET		Allinea i ruoli nel CAM secondo quanto presente nel sistema delle risorse umane
/all	GET	page	Ritorna tutti i ruoli impaginati. Page specifica quale pagina occorre visualizzare
/allNoPage	GET	page	Ritorna tutti i ruoli non impaginati
/map	POST	id, threshold	Mappa il ruolo richiamando tutti i plugin passando il parametro threshold
/id/users	GET	id	Ritorna tutti gli utenti con il ruolo passato in parametro
/id/legacysystems	GET	id	Ritorna tutti sistemi mappati con il ruolo selezionato
/id/legacysystems/{pluginID}	GET	id, pluginID	Ritorna il sistema richiesto, mappato al ruolo passato in parametro
/id/legacysystems/{pluginID}	POST	id, pluginID, rights	Modifica il sistema richiesto, mappato al ruolo passato in parametro. Rights deve essere una rappresentazione json del diritto

5.6 Annunci

Con questo endpoint è possibile ottenere gli annunci di fermi informatici presso l'azienda.

L'URL per questi metodi è il seguente:

http://HOST:8080/api/news

Tabella 5.4: API - Annunci

Endpoint	Tipologia	Parametri	Descrizione
/	GET		Ritorna tutti gli annunci attualmente pubblicati

Capitolo 6

Conclusioni e sviluppi futuri

Il progetto è stato eseguito rispettando gli obiettivi prefissati all'inizio.

Come descritto nel capitolo 2 alla sezione 2.2 l'obiettivo di creare una struttura dati flessibile per le esigenze dei vari sistemi è stato raggiunto, il modello supporta qualunque genere di struttura attualmente studiata.

Grazie alla classe astratta e all'impiego dei tipi generici tutti i plugin sono stati sviluppati senza alcun problema. Il grosso vantaggio dell'impiego dei tipi generici è quello di mettere a disposizione dello sviluppatore, dei plugin, ad avere un grado di libertà maggiore nel descrivere la propria struttura di *diritto*.

L'automatismo "reverse engineering" per popolare la mappa in funzione della situazione attuale è stato integrato nell'interfaccia grafica del CAM, vedi capitolo 4 nella sezione 4.5.5 dedicata alle impostazioni.

Grazie a questa funzionalità è dunque possibile generare la mappa da 0, questo è necessario per inizializzare il sistema appena messo in produzione.

Il lavoro è stato esposto ai membri dello *steering group* dell'Area ICT presso l'Ente Ospedaliero Cantonale, membro di questo gruppo è anche il Signor *Gian Maria Togni*, responsabile del team in supporto all'utenza.

Durante l'incontro è emerso che una validazione dei colleghi responsabili della creazione degli utenti informatici non è necessaria, questo perché la mappatura del CAM rispecchia la situazione attuale in azienda.

L'interfaccia grafica è semplice ed intuitiva e ne permette la comprensione da parte di tutti. Tutte le funzionalità richieste sono state integrate nell'interfaccia grafica, vedi capitolo 4.

I servizi WEB sono stati sviluppati e supportano chiamate HTTP da parte di chiunque si

autentichi con delle credenziali Active Directory valide, vedi capitolo 5.

6.1 Sviluppi futuri

Nella seguente sezione sono elencati gli sviluppi pianificati per il futuro, in azienda, al termine del lavoro di diploma.

6.1.1 Integrazione sistemi aggiuntivi

Per lo svolgimento del progetto sono stati presi in considerazione i sistemi *legacy* più importanti in azienda, tuttavia vi sono alcuni applicativi minori che necessitano comunque di essere integrati.

A tale scopo uno sviluppo futuro è proprio quello di integrare tutti i sistemi utilizzati.

In aggiunta sarà necessario demandare lo sviluppo del plugin alle aziende che non ne permettono lo sviluppo da parte dell'Ente Ospedaliero Cantonale.

6.1.2 Funzionalità aggiuntive sui plugin esistenti

Non tutti i plugin attualmente sviluppati supportano la modifica della mappatura e/o la scrittura nel loro sistema di appartenenza.

Uno sviluppo futuro è dunque di portare tutti i plugin, se possibile, a supportare anche la scrittura e pure ad implementare la modifica della mappatura, come descritto nel capitolo 2 alla sezione 2.2.

6.1.3 Comparazione più user friendly

Questa funzione può essere molto utile quando si comparano i diritti assegnati all'utente con quelli mappati nei suoi ruoli.

Attualmente questa funzione mette a disposizione una comparazione statica, ovvero mostra entrambi i diritti e sarà poi l'utente a dover analizzare le differenze.

Quello che si vuole fare con questo sviluppo futuro è evidenziare le differenze mediante colori in modo che balzino subito all'occhio.

6.1.4 Documentazione sviluppo plugin

Per permettere di delegare lo sviluppo ad una azienda terza, sarà necessario essere in possesso di una documentazione *ad-hoc* per lo sviluppo di plugin, questo sarà un punto di massima priorità in un futuro prossimo.

La documentazione dovrà spiegare esattamente come deve essere sviluppato un plugin, quali interfacce implementare e come strutturare il file di configurazione.

6.1.5 Tracciabilità modifiche

Il CAM permette di eseguire operazioni molto pericolose, è possibile assegnare o revocare diritti a tutti gli utenti aziendali.

Si renderà dunque necessario tracciare i nominativi delle persone che eseguono qualunque genere di modifica.

6.1.6 Messa in produzione

Allo stato attuale il progetto è ancora in fase di sviluppo, un primo step per la messa in produzione potrebbe essere quello di utilizzare *docker* e containerizzare l'applicativo tomcat e il database. Questo permetterebbe un deploy molto più semplice e rapido.

Bibliografia

- [1] Rest HTTP methods of CA SDM
<https://docops.ca.com/ca-service-management/14-1/en/reference/ca-service-desk-manager-reference-commands/technical-reference/rest-http-methods>.
- [2] Introduction to webjars
<http://www.baeldung.com/maven-webjars>.
- [3] Spring guides
<https://spring.io/guides>.
- [4] Bootstrap documentation
<https://getbootstrap.com/docs/4.1/getting-started/introduction/>.
- [5] Mustache documentation
<https://github.com/janl/mustache.js>.
- [6] AS400 connection
https://www.ibm.com/support/knowledgecenter/en/SSEPEK_10.0.0/java/src/tpc/imjcc_tjvjcccn.html.