Research Program

WEB-based management and representation of spatial and geographic data

SPADA@Web

Controllo dell'accesso a dati spaziali basati su ruoli: un primo prototipo

(Controlled access to spatial data : a first prototype)

Maria Luisa Damiani – UNIMI
Matteo Carati - UNIMI

Technical Report UNIMI.D.5
October 2005

Capitolo 1:
CONTROLLO DEGLI ACCESSI PER BASI DI DATI SPAZIALI

1.1 INTRODUZIONE

Il proliferare di Servizi Web per Informazioni Geografiche (Spatial Web Services) sta creando nuove opportunità per la diffusione e la condivisione di informazioni geografiche. In generale, un Web Service spaziale è un software che esegue operazioni geografiche e spaziali, come la visualizzazione di mappe, la conversione di sistemi di coordinate e l'analisi spaziale. Queste operazioni di solito sono richiesta da utenti Web Client attraverso interfaccie apposite. I Web Service spaziali sono molto importanti perchè rappresentano un'alternativa ma anche un'evoluzione ai monolitici sistemi GIS tradizionali, spesso utilizzati senza sfruttarne appieno le potenzialità. Un argomento che non è stato ancora molto discusso dalla comunità GIS riguarda come garantire accessi sicuri ai dati spaziali sul Web. Infatti in questo campo c'è un gran bisogno di sicurezza dei dati, motivato da diversi fattori: primo, i dati geografIci possono contenere informazioni sensibili, che non devono essere raggiunte facilmente da chiunque; inoltre agli utenti dei sistemi geografici distribuiti su Web vanno assegnati differenti permessi a seconda dal livello di responsabilità che hanno. In generale vengono considerati indispensabili due tipi di servizio per la sicurezza delle strutture su rete: il servizio per il controllo degli accessi e quello per la comunicazione sicura. Il servizio per il controllo degli accessi protegge le risorse di Internet da usi non autorizzati, mentre il servizio per la comunicazione sicura protegge i dati trasmessi sulla rete. Requisiti fondamentali per la sicurezza sono la riservatezza e l'integrità. Assicurare riservatezza significa impedire agli utenti non autorizzati l'accesso improprio alle informazioni, mentre si garantisce l'integrità proteggendo i dati da modifiche non autorizzate, cioè evitando agli utenti non autorizzati l'inserimento o la modifica dei dati.

In questo documento viene presentato un prototipo di sistema per il controllo dell'accesso a dati spaziali su Web basato su *ruoli* . Il lavoro e' organizzato come segue: nel capitolo 1 viene presentato lo stato dell'arte; nel capitolo 2 viene presentato il modello utilizzato; infine nel capitolo 3 e' illustrato il prototipo.

1.2 STATO DELL'ARTE

1.2.1 An Access Control Model for Geographic Data in a XML-based Framework

Purevjii, Amagasa, Imai e Kanamori propongono in [21] un modello per il controllo degli accessi a basi di dati spaziali in un sistema basato su XML.

Il lavoro propone un modello per il controllo degli accessi per sistemi informativi geografici distribuiti su Web. Gli accessi controllati dipendono dall'estensione spaziale dei dati geografici, utilizzando il formato XML ed il formato di grafica 2D SVG, anch'esso basato

su XLM. In particolare il meccanismo tiene in considerazione i limiti spaziali e il livello di dettaglio (LoD Level of Details) dei dati geografici per regolare l'accesso.

L'articolo propone uno scenario che aiuta a capire meglio le motivazioni che hanno spinto alla creazione di un modello con le caratteristiche specificate. Viene presentata infatti una struttura per la gestione di un disastro ambientale da parte di amministrazioni locali e nazionali. Viene ipotizzata la presenza di molti utenti raggruppati sotto diversi profili (amministrazione, medici, primo intervento, forze dell'ordine, semplici cittadini, ecc.) che interagiscono con il sistema dinamicamente a seconda dei rispettivi bisogni e responsabilità. Questi gruppi sono autorizzati ad accedere solo alle informazioni strettamente necessarie per svolgere i propri compiti. Ad esempio il personale medico può essere autorizzato ad accedere alle informazioni relative agli invalidi o ai cittadini che necessitano di cure tempestive in caso di disastro. I membri di uno stesso gruppo vengono classificati in vari livelli a seconda della loro posizione amministrativa da locale a nazionale (ad esempio una divisione può essere: area, zona, città, provincia, regione, nazione). Inoltre utenti dello stesso gruppo e dello stesso livello vengono ulteriormente divisi in regioni amministrative, a seconda di dove si trovano.

Vengono date alcune definizioni di base sulle quali si sviluppa poi il meccanismo di controllo degli accessi introdotto nel modello:

Authorization object: Nel modello le feature class e le informazioni non geografiche sono memorizzate entrambe con la rappresentazione XML.
Le autorizzazioni per gli oggetti sono definite in termini di identificatori di elementi XML. Per non perdere la semantica e l'organizzazione strutturale tuttavia le autorizzazioni sono definite anche attraverso gli identificatori delle features e dei livelli. Gli oggetti sono raggruppati nel database in livelli. Un Authorization object è dunque una collezione di livelli in cui sono raggruppati i vari oggetti del warehause geografico.

Authorization Subject: Nel modello si vuole supportare qualsiasi tipo di modellazione dei soggetti. Un soggetto è dunque rappresentato con una tupla <ID, profilo, Ruolo>. ID è l'identificativo univoco che rappresenta il soggetto, il concetto di ruolo è quello esposto nei modello RBAC e il profilo è una tupla <nome, indirizzo, lavoro> e dunque rappresenta un insieme di proprietà dell'utente. Si può supportare la gerarchia dei ruoli presentata nel modello RBAC.

Autorization: Un autorizzazione nel modello è rappresentata da una tupla < *Authorization Subject, Authorization object, oper_reg, LoD, md* > dove Authorization Subject e Authorization object sono il soggetto è l'oggetto dell'autorizzazione come appena definito, md è un insieme di modalità d'azione (ad esempio *view, update, delete, all, ecc*.), oper_reg definisce quale sono le regioni operative su cui sono validi i permessi e Lod definisce il livello di dettaglio a cui è permesso visualizzare gli oggetti.

Nel modello viene infine descritto brevemente ed a grandi linee come si comporta l'applicazione per il controllo spaziale degli accessi e quali sono i passi che devono essere eseguiti per rispondere ad una richiesta di accesso ai dati. Il modello viene proposto con uno scarso livello di dettaglio per quanto riguarda le scelte progettuali e le motivazioni. La modellazione è altresì inefficace e non è chiaro quali sono i vantaggi che tale modello introduce.

1.2.2 Secure Access Control in a Multi-user Geodatabase

Nel modello proposto da Sahadeb De, Caroline Eastman, Csilla Farkas in [12] si cerca di implementare concretamente un sistema per il controllo degli accessi con tecnologie esistenti.

Gli obbiettivi che gli autori vogliono raggiungere sono definire e valutare la realizzabilità di un modello per il controllo degli accessi per proteggere l'integrità e la riservatezza dei dati in database spaziali. Intendono poi implementare il modello con software abbastanza diffusi come ArcGIS 8.1, ArcSDE e MS SQL Server. L'architettura proposta è illustrata in figura 1. L'asterisco indica dove è richiesta una autenticazione dell'utente. Inizialmente gli utenti vengono autenticati dal sistema sotto cui lavora il software GIS. Dopodiché tutti gli accessi ai dati del Database Server SQL devono essere autenticati. La ragione per cui sono previsti due livelli di sicurezza è dovuta alla necessità di avere diversi livelli di granularità della sicurezza in tale sistema.

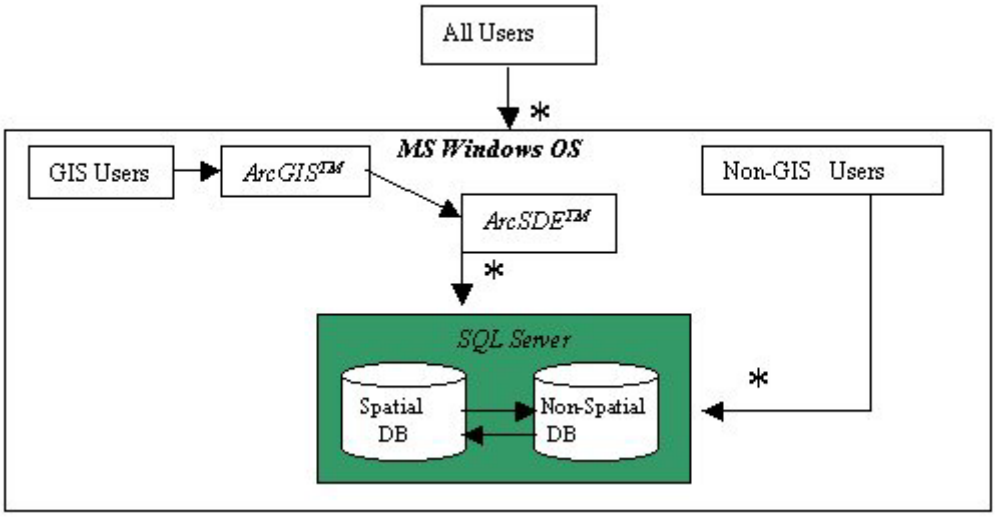


Figura 1: Architettura della sicurezza

Nel modello i permessi per l'accesso sono assegnati anche soltanto a sotto-tuple del database. Questo permette di avere una granularità di sicurezza a livello di tuple come nei controlli degli accessi basati sul contenuto. Le sottotuple per una data regola di sicurezza non sono altro che una vista del database, ciò permette di dire che il controllo degli accessi implementato sia basato sulle viste.

Tutti i tipi di utenti, sia quelli che accedono a dati geografi sia quelli che non hanno accesso ai dati spaziali, vengono suddivisi in ruoli secondo la definizione RBAC.

L'intento degli autori è dunque quello di creare un controllo degli accessi basato sulle viste. Gli utenti del sistema hanno il permesso di accedere ad un determinato insieme di viste a seconda delle autorizzazioni che possiedono. Le viste sono costruite a partire da un database multi-livello e possono essere aggiornate, rispettando i privilegi dell'utente. Ogni aggiornamento viene propagato in dietro alle relazioni mutli-livello. Il modello propone tre diverse soluzioni per tre differenti architetture di sicurezza:
1 Database Multi-livello Singolo
2 Database Multi-livello Replicato
3 Database Mono-livello Singolo

Nel primo tipo d'architettura (figura 2), il modello potrebbe essere un singolo database che memorizzi e gestisca i dati classificati in differenti livelli. Hanno utilizzato i comandi amministrativi di ArcSDE per definire diverse viste delle relazioni base con differenti privilegi. Gli utenti potranno accedere a queste diverse viste in base ai loro permessi d'accesso (privilegi). Gli aggiornamenti saranno propagati in dietro al database principale. Sebbene questo meccanismo risulti ottimale per la vista sicura dei dati, in fase di implementazione hanno incontrato dei problemi non riuscendo ad eseguire l'aggiornamento (aggiunta o eliminazione di feature) in modalità vista sul livello dati (feature class) con ArcMap.

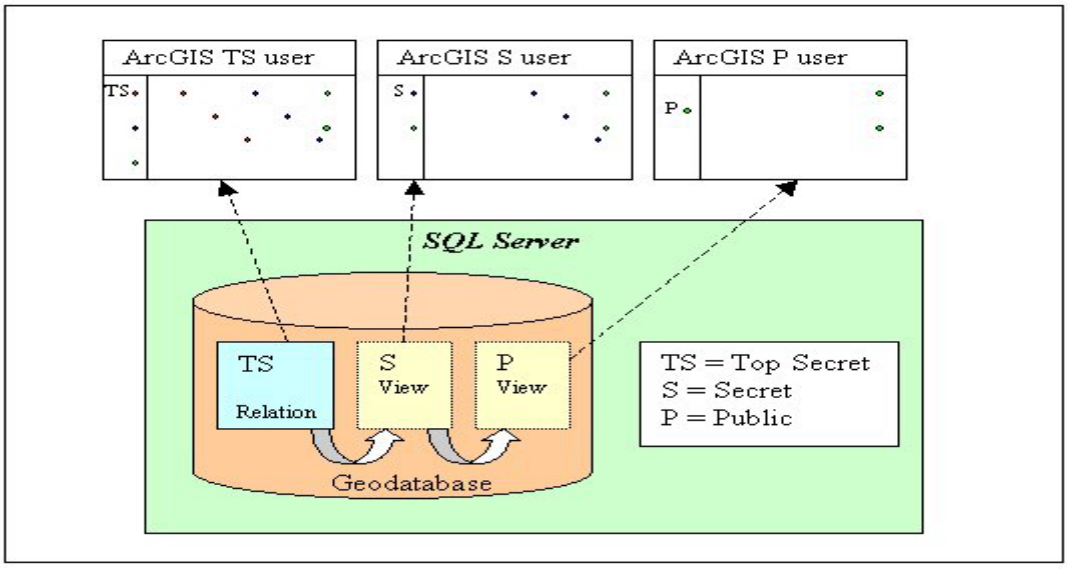


Figura 2: Database Multi-livello Singolo

Nel secondo tipo di architettura (figura 3) i dati vengono separati in diversi database, in base alla classificazione dei dati stessi. Ogni livello dei dati basso viene replicato nel database corrispondente al livello più alto. Questo permette ad ogni utente di vedere il database completo al suo livello di sicurezza. Ovviamente questo modello presenta un

consistente problema di ridondanza dei dati e di traffico elevato in fase di aggiornamento dai database di livello più basso verso quelli di livello più alto.

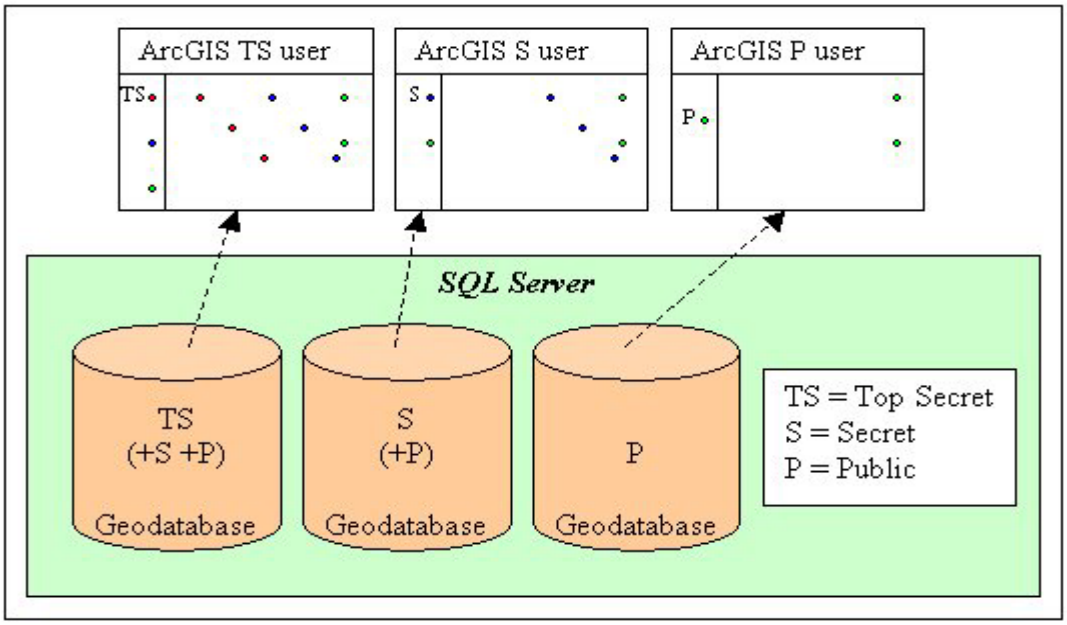


Figura 3: Database Multi-livello Replicato

Hanno ottenuto i risultati migliori con il terzo tipo di architettura (figura 4), in cui i dati sono separati in relazioni singolo-livello. In questo modello quando un utente accede al database, che è formato da relazioni singolo-livello, può avere accesso ai dati del solo livello appropriato. Dopo che la sessione dell'utente è terminata, il database composito viene scomposto in diverse relazioni singolo-livello in base alla classificazione di sicurezza dei dati. Gli utenti GIS in pratica caricano le relazioni una sull'altra. Questo dà la sensazione agli utenti di accedere ad una singola relazione che includa dati di diversi livelli, mentre in realtà si tratta di una proiezione di dati indipendenti. La presenza di una sola copia dei dati permette una gestione molto semplificata della concorrenza.

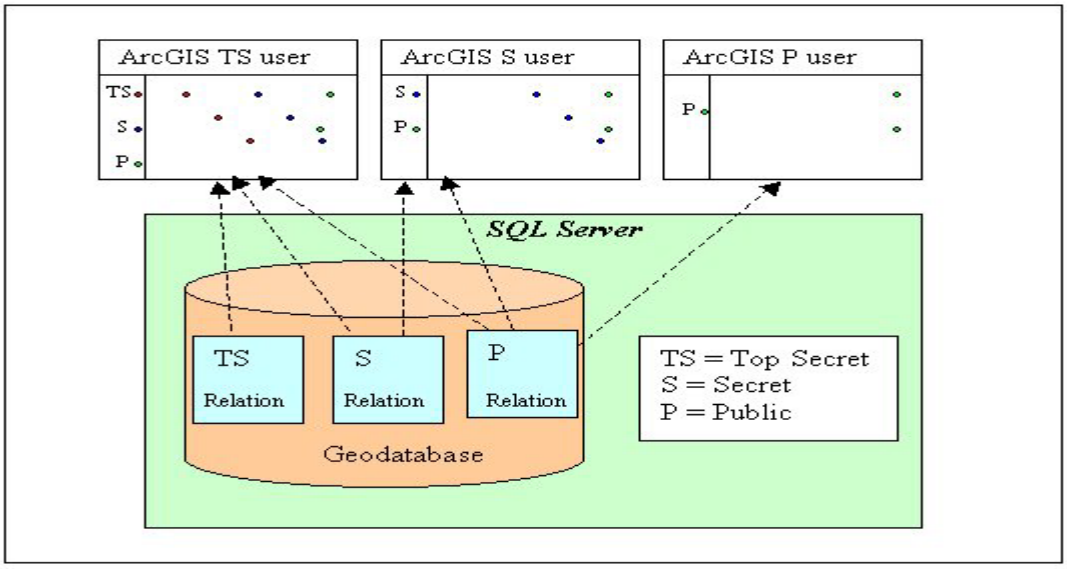


Figura 4: Database Mono-livello Singolo

Come gli autori stessi ammettono, il modello proposto nella fase implementativi ha prodotto solo risultati limitati. In particolare, le viste create da SQL Server non possono essere interpretate dal software GIS. D'altra parte le viste create dal software GIS (ArcSDE) non soddisfano i requisiti di sicurezza che si erano prefissi di raggiungere dato che le liste di basso livello non sono editabili.

1.2.3 Access control system for SVG Documents" in "Multimedia Security and Digital Rights Management Technology

Fernandez-Medina, De Capitani di Vimercati, Damiani, Piattini e Samarati propongono in [14] un modello per il controllo degli accessi per dati geografici SVG.

Gli autori presentano un approccio per la protezione a granularità fine delle feature di dati Scalable Vector Graphics (SVG). Anche questo modello si basa sulle regole d'autorizzazione, e in questo caso vengono espresse in un linguaggio basato su XML. Una regola d'autorizzazione indica quali azioni sono permesse o negate per un oggetto multimediale. Nel modello presentato, una regola d'autorizzazione ha 4 componenti: il soggetto (*subject*) a cui appartiene la regola, l'azione (*action*) a cui si riferisce, l'oggetto (*object*) a cui è applicata e un segno (*sign*) che descrive se tale regole indica un permesso(+) oppure un divieto(-).

In questo modello un soggetto viene definito sia come un singolo utente che come un gruppo di utenti. Come al solito, i gruppi sono definiti come insiemi di utenti eventualmente organizzati in una gerarchia. Questo semplice attributo permette un controllo degli accessi efficiente e una risoluzione rapida dei conflitti che si possono creare fra regole di permesso e regole di divieto.

In accordo con le specifiche SVG ci possono essere tre tipi di oggetti per cui è richiesta la protezione: definizioni (*definitions*), gruppi (*groups*) ed elementi SVG (*SVG elements*). Gli elementi SVG possono essere sia grafici che testuali, oppure essere dei riferimenti alle definizioni. Il modello proposto permette l'associazione delle autorizzazioni con ognuno di

questi elementi specifici contenuti in un documento SVG. Essendo SVG basato su XML, si può utilizzare le espressioni generiche Xpath sui documenti SVG per specificare l'elemento cui si riferisce l'autorizzazione. Ovviamente gli autori hanno introdotto anche un metodo per raggruppare gli elementi grafici da abbinare alle regole d'autorizzazione. Definire una regola per ogni elemento sarebbe infatti impraticabile. Il meccanismo che permette di definire classi di elementi è abbastanza semplice. Viene aggiunto un attributo all'elemento che si vuole raggruppare, tale attributo è *TypeElement*. Sarà possibile dunque definire regole d'autorizzazione per un TypeElement, valide cioè per ogni elemento appartenente al gruppo. Una regola può dunque essere associata ad un singolo elemento rappresentato attraverso il suo percorso Xpath all'interno del documento o tramite il suo identificatore univoco, oppure la regola può essere associata ad un gruppo di elementi. Un altro metodo fondamentale per raggruppare elementi in un sistema geografico è attraverso un'intersezione spaziale. Nel modello proposto è possibile infatti associare una regola alla geometria di un elemento attraverso la funzione perimetro. Ciò rende la regola d'autorizzazione valida per tutti gli elementi contenuti all'interno del perimetro dell'oggetto della regola. LA funzione perimetro può essere applicata anche agli elementi raggruppati, cioè ai TypeElement. La regola sarà valida per ogni elemento contenuto all'interno del perimetro di uno qualsiasi degli elementi del TypeElement oggetto della regola.
Il modello permette inoltre di selezionare elementi in base a determinate condizioni. Queste sono rappresentate da espressioni booleane che fanno uso dei predicati:

- *inside(obj)* che restituisce gli oggetti della regola d'autorizzazione solo se contenuti all'interno dell'elemento (o del TypeElement) *obj.*
- *Together_with(obj)* che restituisce gli oggetti della regola d'autorizzazione solo se figli di un elemento (o di un TypeElement) padre di *obj.*
- *Number_of(obj,n)* che restituisce gli oggetti della regola d'autorizzazione solo se ci sono *n* istanze dell'elemento (o del TypeElement) *obj.*

Anche la definizione delle condizioni ha una struttura basata su XML.
Il modello con le regole d'autorizzazione per SVG permette di specificare se gruppi d'utenti o singoli utenti hanno il permesso o il divieto di accesso ad oggetti singoli o raggruppati per tipo o raggruppati per posizione o rispondenti a determinate condizioni. Benché il modello proposto risponda in pieno ai requisiti di sicurezza per il controllo degli accessi risulta molto oneroso da implementare e troppo vincolato alla sintassi verbosa di SVG. Nel modello poi non viene tenuta in considerazione una caratteristica fondamentale dei sistemi informativi geografici quale la dipendenza (o gerarchia) che spesso esiste tra gli elementi spaziali.

1.2.4 SRBAC: Un Controllo degli Accessi Spaziale Basato sui Ruoli per Sistemi Mobili

Introduzione

Sebbene il modello RBAC sia stato oggetto di molti studi non è ancora stato investigato l'aspetto per cui il sistema possa prendere le decisioni sugli accessi in base alla collocazione spaziale in cui si trova l'utente. La crescita dell'utilizzo di dispositivi mobili e di reti wireless in molte organizzazioni ha spinto Hansen e Oleshchuk alla formulazione del modello Spatial Rolebased Access Control (SRBAC) [16]. Si tratta di una estensione del noto modello RBAC che permette di specificare vincoli spaziali per abilitare o disabilitare ruoli. SRBAC può essere utilizzato per vincolare l'insieme dei permessi disponibili per i ruoli che un utente può attivare in una determinata collocazione. Nel modello proposto dagli autori si suppone che il sistema sia in grado di identificare e verificare la locazione di ogni utente legittimo sulla base di una architettura di rete fidata. Le locazioni descrivono aree identificabili dal sistema.

Il Modello SRBAC

Per includere il modello tradizionale RBAC in una piattaforma mobile bisognerebbe definire ruoli per ogni settore del dominio dell'organizzazione. Tuttavia in sistemi con molti domini di locazioni, la definizione dei ruoli è molto onerosa. Inoltre, i ruoli definiti per tali locazioni possono avere molti permessi in comune. Dunque, gli autori ottengono più flessibilità definendo una politica di sicurezza dove i permessi sono assegnati dinamicamente ai ruoli limitatamente alla locazione dove sono situati gli utenti. Quindi si può dire che un ruolo sia dinamico, nel senso che può avere associato permessi diversi in locazioni distinte.

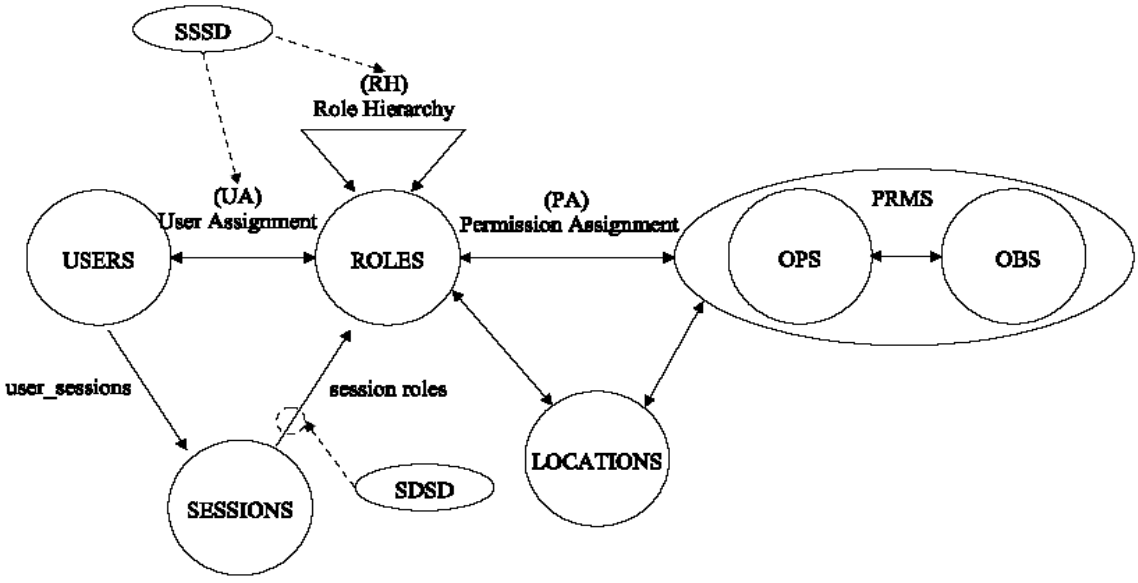


Figura 5: Schema SRBAC

Core SRBAC

Il modello SRBAC (figura 5) è formato da 5 insiemi: Utenti, Ruoli, Permessi, Sessioni e Locazioni. Gli utenti vengono considerati unità mobili che possono stabilire una connessione (eventualmente wireless) con le risorse del sistema per eseguire una qualche attività. I ruoli sono descritti come insiemi di permessi per accedere alle risorse

del sistema (gli oggetti). I permessi sono autorizzazioni ad eseguire una o più operazioni sugli oggetti RBAC e dipendono dal ruolo e dalla locazione dell'utente che ricopre il ruolo. Le locazioni sono rappresentate per mezzo di espressioni simboliche denominate espressioni di posizione che descrivono i dominii di posizioni identificabili dai sistemi.
Nel modello si assume che le aree definite nell'insieme delle locazioni coprono l'intero dominio delle responsabilità Z di SRBAC. Tale dominio Z è diviso su livelli fisici in sottoaree, chiamate *primary location cells* e denotate con πi, i = 1, .., k, che riflette la capacità dell'architettura sottostante di identificare l'utente in una determinata cella. Tuttavia utilizzare le *primary location cells* nel modello SBRAC può essere impraticabile poiché tali celle rappresentano l'infrastruttura del sistema di localizzazione mentre in realtà si vuole modellare una struttura delle locazioni che rifletta l'organizzazione. Nel modello viene allora introdotto il concetto di locazione logica che rifletta la struttura delle locazioni e la politica di sicurezza dell'organizzazione. Generalmente una stessa posizione può appartenere a differenti domini logici di locazione. Per semplificare la fase di definizione e di implementazione, una locazione logica è formata da una o più *primary location cell.*
Sono definite diverse funzioni sugli insiemi Utenti, Ruoli, Permessi, Sessioni e Locazioni. La relazione UA (user assignment) rappresenta l'appartenenza di un utente ad un ruolo. La relazione PA (permission assignment) rappresenta l'assegnamento di un permesso ad un ruolo in base alla locazione. L'assegnamento di un utente ad una sessione avviene attraverso la funzione user_sessions dove ogni utente può essere assegnato ad una sola sessione.

1.2.5 Un modello per il controllo degli accessi per i dati spaziali su Web basato sul concetto di finestra

Un modello per il controllo degli accessi per i dati spaziali su Web che presenta interessanti soluzioni è quello proposto da Bertino, Damiani e Momini nel 2003 in [5] e [6].
Il modello è basato sui seguenti presupposti: in primo luogo, i dati spaziali consistono in oggetti con contorni ben definiti situati in uno spazio geografico; in secondo luogo, i dati sono maneggiati dagli utenti a distanza attraverso le funzioni fornite da un Web Map Managemant Service che funge da mediatore fra gli utenti ed i dati. L'obiettivo del sistema è controllare il modo in cui i dati sono raggiunti dagli utenti che hanno profili differenti. Il metodo propone un'estensione del modello classico di controllo degli accessi basato sulla nozione di *regole di autorizzazione*.
L'idea centrale è quella di assegnare ad ogni autorizzazione una *portata geografica* , vale a dire una regione limitata in cui l'autorizzazione è valida. Di conseguenza, le funzioni che gli utenti possono eseguire sui dati spaziali possono variare, a seconda dell'identità dell'utente e della posizione dell'oggetto.
I problemi riguardanti la protezione di dati sono stati ampiamente studiati nei sistemi per l'amministrazione delle basi di dati convenzionali. Inoltre, ci sono stati parecchi sforzi per estendere i modelli convenzionali di controllo degli accessi con i nuovi tipi di dati e

modelli. Tali sforzi includono modelli per il controllo degli accessi per le pagine Web ed i dati in XML, modelli temporali di controllo degli accessi, i modelli per il controllo degli accessi estesi alle basi di dati relazionali e modelli per il controllo degli accessi per le librerie digitali. Questo nuovo metodo è il primo a proporre un modello completo di controllo degli accessi per i dati spaziali. Tale modello inoltre è basato su standard esistenti; è stato implementato e la sua architettura è basata sui moderni paradigmi di Web Service. L' applicazione che utilizza un'implementazione del modello, viene presentata in seguito come un caso di studio. Essa consiste in un Web Map Management Service che supporta l'aggiornamento e la condivisione di dati ambientali riguardanti i bacini del più importante fiume in Italia. I dati ambientale possono essere acquisiti da differenti unità organizzative e perfino di gruppi di volontari sparsi sul territorio. La base di dati spaziale può essere modificata così da differenti generi di utente, che possiedono diritti di accesso differenti.

Una possibile architettura per il Web Service che includa il meccanismo per il controllo degli accessi appena introdotto si basa,sul noto modello a tre livelli: Presentazione, Applicazione e Memorizzazione dei Dati (Figura 6).
Il livello di Applicazione è formato da due servizi principali il servizio per controllo degli accessi (ACS Access Control Service) e il servizio applicazione (AS Application Service). Il primo presenta e implementa le operazioni per fare i controlli sulle regole d'autorizzazione. Il secondo servizio presenta ed implementa la parte logica dell'applicazione ed accede ai dati. Quando un utente invoca un'operazione attraverso un WebClient, l'Application Service interagisce con l'Access Control Service per verificare se l'operazione può essere effetuata, se il controllo ha esito positivo l'operazione viene eseguita.

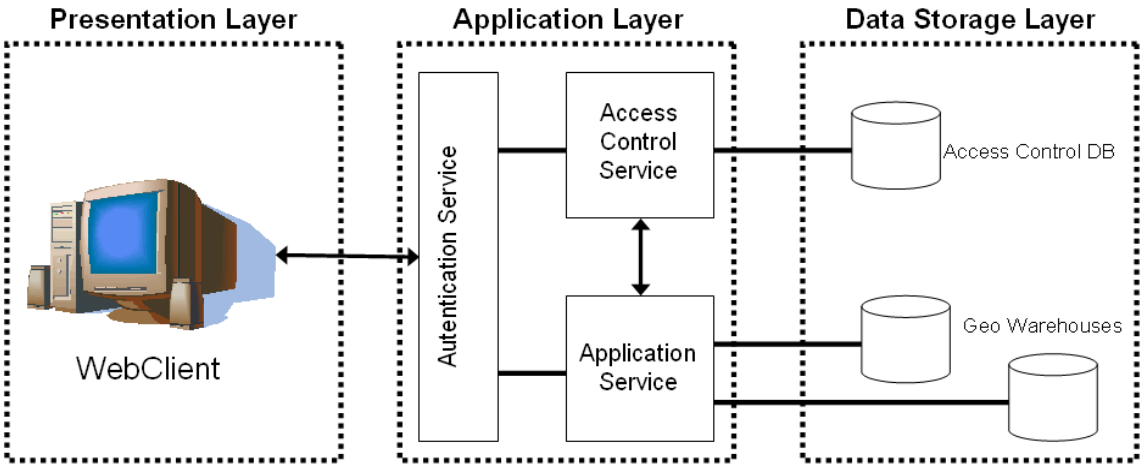


Figura 6: Architettura generale per il Web Service

Un modo conveniente per organizzare l'insieme delle regole d'autorizzazione è quello di memorizzarle in un database spaziale compatibile con il modello per le feature semplici del consorzio OpenGIS. In questo modo si può mappare direttamente le regole

d’autorizzazione con finestra sul database spaziale che contiene le informazioni geografiche. Lo schema delle regole d’autorizzazione può essere mappato su una Simple Feature Class definendolo come segue:

Feature Class Authorization Rule
ID: ObjrctID
Role: String
Privilege: String
FeatureClass: String
Window: OpenGIS Geometry Type
Grantor: String
GrantOption: Boolean

Secondo la definizione appena data, ogni ruolo ha un identificatore univoco e un attributo per ogni elemento della regola. Si noti che la finestra d’autorizzazione è definita come un attributo geometrico. In tale maniera sia i dati spaziali che le informazioni per il controllo degli accessi sono modellate uniformemente in termini di Simple Feature. Il livello applicazione include anche un Servizio d’Autenticazione che può essere basato sul modello username/password, Secure Socket Layer (SSL) o servizi più complessi.
L’interazione tipica tra cliente e servizio può essere descritta come segue: il cliente si connette al sistema attraverso il servizio di Autenticazione. Il cliente decide dunque quali ruoli attivare. Ogni richiesta dell’utente è mappata su una o più operazioni dell’Application Service che, interagendo con l’Access Control Service, verifica se e dove le operazioni possono essere eseguite.

Capitolo 2:
IL MODELLO PER IL CONTROLLO DELL'ACCESSO

In questo capitolo viene descritto un nuovo modello per il controllo degli accessi a basi di dati spaziali distribuite sul web. Il capitolo è organizzato come segue: nella prima parte vengono presentati i presupposti del modello e le entità che non cambiano o poco si discostano dal modello RBAC; vengono successivamente introdotti e descritti i concetti nuovi di window, geo-permesso, gerarchia dei permessi, ruolo parametrico e ruolo dinamico; infine vengono enunciate le specifiche funzionali necessarie per il corretto funzionamento del sistema e lo schema riassuntivo del modello. Per la rappresentazione di tale modello e degli schemi dei vari componenti presentati si utilizza la notazione UML brevemente riassunta nel capitolo 1. Per comodità si utilizza una notazione semplificata di UML riportando unicamente i nomi e le associazioni delle classi. Si rimanda invece alla fine del capitolo per la specifica completa delle singole classi.

2.1 INTRODUZIONE

Le applicazioni correnti richiedono sempre più che i dati spaziali siano gestiti attraverso il web. I sistemi d'informazione sul traffico, la fornitura di mappe ed i servizi basati sulla posizione del cliente sono soltanto poche della applicazioni coinvolte nella diffusione dei dati spaziali su web. Tuttavia, capita spesso che gli utenti non debbano soltanto visualizzare i dati ma anche modificarli e interagire con essi. È anche importante notare che in molti casi gli aggiornamenti sono effettuati da utenti collegati a distanza e che, inoltre, i dati spaziali sono presentati solitamente agli utenti attraverso mappe. L'uso di mappe è cruciale per georeferenziare correttamente i dati. Si pensi per esempio a degli ispettori forniti di dispositivi mobili che raccolgono i dati sul campo e aggiornano una base di dati a distanza; questo funzionamento richiede in primo luogo un trasferimento dal sistema centrale della mappa con gli strati di interesse, dopo la creazione dei dati spaziali ed infine l'upload in remoto attraverso il web dei nuovi oggetti spaziali nella base di dati.

Attualmente, la visualizzazione e la pubblicazione degli oggetti spaziali su web è supportato da parecchi sistemi di amministrazione di mappe commerciali. Tuttavia, un argomento che non è stato ancora molto studiato è quello di fare rispettare accessi controllati ai dati spaziali, per garantire riservatezza ed integrità delle informazioni. Assicurare riservatezza significa impedire rilevazione improprie di informazioni ad utenti non autorizzati a vederle. Garantire integrità significa proteggere i dati da modifiche non autorizzate cioè evitare ad utenti non autorizzati l'inserimento o la modifica di informazioni nella base di dati.

I suddetti obbiettivi vengono raggiunti proponendo un modello di controllo degli accessi per i dati spaziali su web.

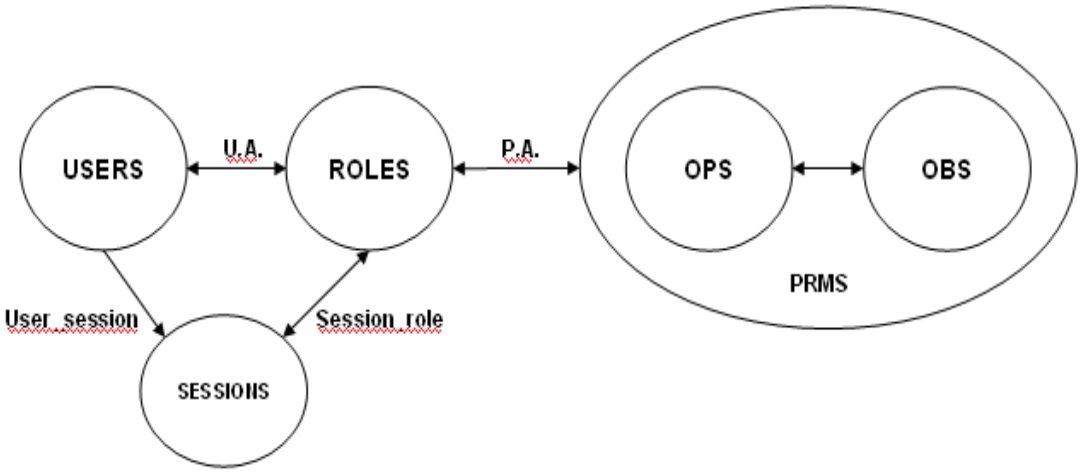


Figura 1: Core RBAC

Partendo dal modello di controllo degli accessi basato sui ruoli RBAC (figura 1) è stato elaborato un nuovo modello che includa il concetto geografico caratteristico delle basi di dati spaziali denominato Geographic Role Based Access Control (G-RBAC). Nel modello base (figura 2), i permessi per effettuare operazioni su determinati oggetti includono anche una specifica di validità spaziale, ovvero una window entro la quale il permesso è abilitato.

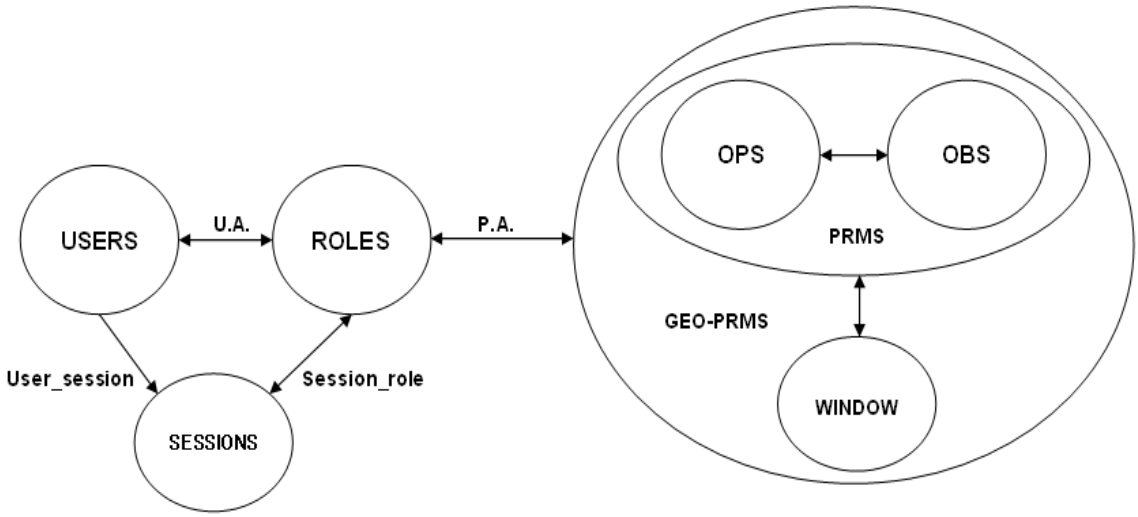


Figura 2: G-RBAC - Modello Base

Il modello base viene successivamente esteso, ampliando la nozione di ruolo col concetto ruolo parametrico e dinamico. Il ruolo parametrico permette di definire una classe di ruoli con determinati permessi da poter utilizzare su diverse window da definire al momento dell'istanzazione. Monitorando la localizzazione geografica dell'utente connesso è possibile definire un ruolo dinamico che si attiva per tale utente solo se esso si trova in una determinata window. Viene implementata infine una gerarchia dei permessi che permette di definire delle dipendenze tra le operazioni e tra gli oggetti, generando così delle restrizioni sulle window.
Nel modello vengono fatte alcune scelte su cui basare l'architettura per quanto riguarda degli aspetti dei sistemi informativi geografici e i web server che li supportano. E' stato adottato il modello vettoriale definito dal consorzio OpenGIS (OGC) basato sulla nozione di *simple feature class*, sfruttando il vantaggio di essere supportato dalle più diffuse piattaforme commerciali per l'amministrazione di dati spaziale. Una *feature* descrive un reale entità attraverso un insieme di attributi definiti sui tipi semplici (*attributi tematici*) e su uno o più attributi definiti sui tipi geometrici (*attributi geometrici*). I tipi geometrici sono quelli definiti dal consorzio OpenGIS: punto, linea, poligono, collezione di geometrie. Le features omogenee sono raccolte in *features class* ciascuna denotata da un nome unico. Un *sistema di riferimento spaziale* è specificato per ogni *features class*, così che i valori degli attributi geometrici possono essere correttamente interpretati come una posizione sulla superficie della terra. Le features spaziali possono essere trasferite dal server al cliente come una immagine o come una mappa vettoriale, a seconda delle capacità dal Web Map Service. Le featurs sono trasferite in in formato vettoriale ed elaborate lato client da un browser integrato con un plug-in o un JavaScript. Poiché questo architettura assicura prestazioni migliori quando le mappe sono usate in maniera interattiva, supponiamo che le features siano trasferite in formato vettoriale e che le procedure geografiche siano elaborate sia sul client che sul server.

2.2 G-RBAC – MODELLO BASE

In questo paragrafo vengono descritte le componenti che formano il nuovo modello proposto, nella versione base. Vengono spiegati dapprima i concetti di utente, ruolo, sessione, operazione ed oggetto simili al modello RBAC. Infine vengono enunciati i nuovi concetti di geo-permesso e window.

2.2.1 UTENTI, RUOLI e SESSIONI

Nel modello G-RBAC proposto, i concetti di utente, ruolo e sessione rimangono uguali a come sono definiti nel modello RBAC. Un raffinamento geografico del concetto di ruolo verrà introdotto in seguito. Un *utente* rappresenta un essere umano (potrebbe anche essere un programma o una macchina) che accede al sistema informativo attraverso il meccanismo degli accessi. Tale meccanismo lo identifica e lo associa ad una *sessione* di lavoro. Un *ruolo* rappresenta un gruppo di utenti che hanno uguale livello di autorizzazione. Le modalità di accesso vengono dunque abbinate ai ruoli. I vari utenti acquisiscono i permessi facendo parte di un ruolo. Attraverso la sessione infatti, ogni utente è abbinato ad un ruolo e può dunque esercitare i permessi abbinati a quel ruolo. Un utente può rivestire più ruoli contemporaneamente. Nel momento in cui un utente effettua la connessione specifica quale ruolo, fra quelli per cui è abilitato, vuole ricoprire e la sessione lo attiva per tale ruolo.

2.2.2 OPERAZIONI e OGGETTI

Un *operazione* è un'insieme di funzioni che l'utente può effettuare sugli oggetti del sistema informativo. Nel caso specifico di sistemi distribuiti su web, le funzioni sono quelle messe a disposizione del web Map Service. Le funzioni vengono raggruppate per ridurre il numero di permessi da definire.

Un *oggetto* nel modello rappresenta una collezione di feature class. Rispetto al modello RBAC è stato introdotta l'idea di raggruppare diversi oggetti, anche non attinenti fra loro, in gruppi per semplificare ulteriormente l'implementazione. Se su due diverse feature si possono eseguire le stesse operazioni non sarà necessario definire due permessi diversi, basterà invece raggruppare le feature nello stesso *oggetto*.

2.2.3 GEO-PERMESSI e WINDOW

Nel modello G-RBAC il concetto di permesso rappresenta una coppia operazione oggetto e viene associata ad uno o più ruoli. Un permesso specifica dunque per un determinato ruolo quali funzioni può eseguire e su quali oggetti. In un sistema geografico è di fondamentale importanza poter specificare anche dove tali operazioni siano consentite. Viene allora introdotto il concetto di geo-permesso. Un geo-permesso è formato da un permesso e da una window che specifica dove l'autorizzazione è valida. Il geo-permesso è dunque formato da una tripla: operazione – oggetto – window (figura 3). La window è,

come nel modello di Bertino, Damiani e Momini in [5 e 6], una area geografica che specifica dove il permesso è autorizzato. Supponiamo che un utente abbia attivato attraverso una sessione un determinato ruolo r1, cui è abbinato un determinato geo-permesso p1. Sia il geo-permesso p1 così definito < <*visualizzazione, CaseAbbandonate*>, *AreaZ*>. Quando l'utente cerca di *visualizzare* gli oggetti della classe *CaseAbbandonate*, di tutte le istanze memorizzate nel WareHouse gli saranno mostrate solo quelle localizzate all'interno della *AreaZ*.

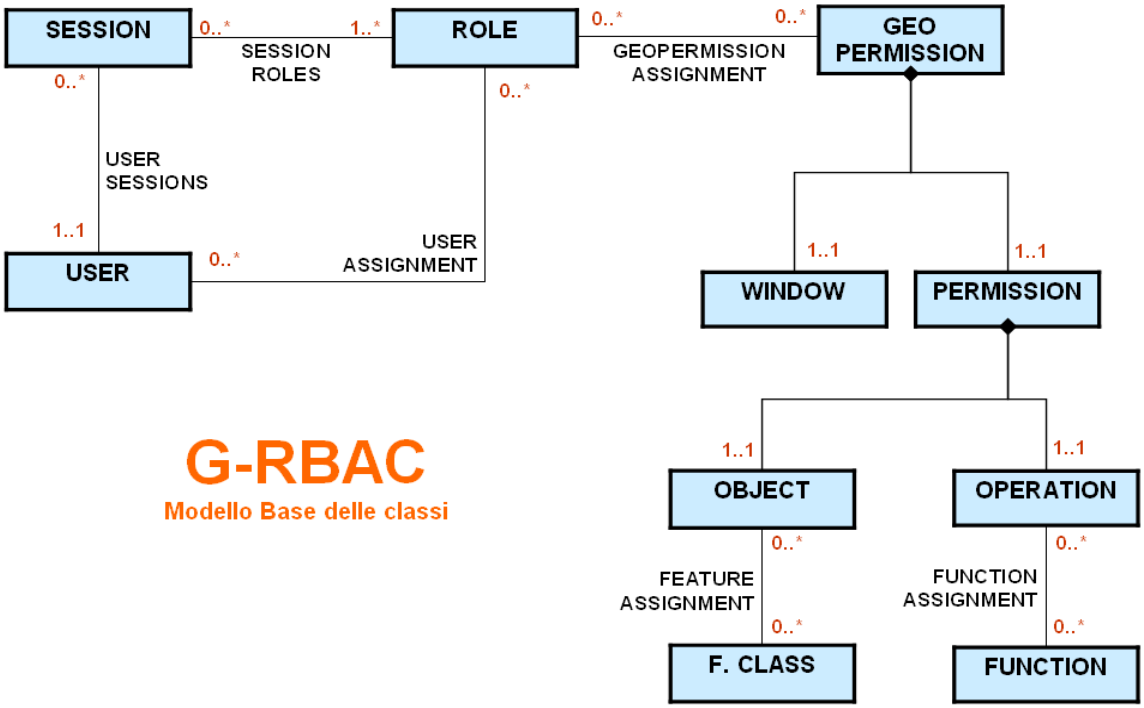


Figura 3: G-RBAC, modello base

Vediamo dunque quello che è il comportamento base del modello G-RBAC proposto. Il modello base è descritto in figura 3. Per comodità si utilizza una notazione semplificata di UML riportando unicamente i nomi e le associazioni delle classi. Si rimanda invece alla fine del capitolo per la specifica completa delle singole classi.Un utente è abilitato a ricoprire zero o più ruoli. Al momento della connessione l'utente avvia una sessione di lavoro. Un utente può avviare più sessioni contemporaneamente ma ogni sessione è associata ad un singolo utente. Attraverso la sessione l'utente attiva uno o più ruoli tra quelli per cui è abilitato. Ogni ruolo ha zero o più geopermessi abbinati. Ogni geo-permesso, ovviamente, può essere abbinato a più ruoli. Un geo-permesso è formato da una sola window e da un solo permesso. Tale permesso è formato a sua volta da un oggetto e una operazione. L'oggetto è un'insieme di Feature Class, mentre l'operazione è un insieme di funzioni messe a disposizione dal web Service.

2.3 GERARCHIA dei PERMESSI

La gerarchia dei permessi viene introdotta nel modello per semplificare e snellire la gestione dei permessi. Inoltre la gerarchia può anche essere vista come uno strumento per porre dei vincoli sulle operazioni che un utente può compiere. Si possono supportare due tipi di gerarchie, quella basata sulle operazioni e quella basata sugli oggetti.
Nella gerarchia delle *operazioni* vogliamo modellare il fatto che la possibilità per un utente di effettuare una determinata funzione implichi la possibilità di effettuare altre operazioni. Analogamente si può dire che si vuole schematizzare il fatto che un'operazione può essere compiuta solo se si ha il permesso di farne alcune altre specificate. Per un determinato oggetto, ad esempio, non si potrà avere il permesso di *modificarlo* se non si detiene almeno il permesso di *visualizzarlo*. Oppure posso dire che se ho il privilegio per *scrivere* un determinato oggetto, è sott'inteso che ho anche il privilegio di *visualizzarlo*.
Analogamente, nella gerarchia degli *oggetti* si formalizza il fatto che la possibilità per un utente di effettuare una funzione su un determinato oggetto implichi la possibilità di effettuarla anche su altri oggetti. Si può dire altresì che un'operazione su un oggetto può essere compiuta solo se si ha il permesso di farla anche su alcuni altri oggetti specificati. Per un determinato ruolo, ad esempio, la visualizzazione dei *tombini* di una rete urbana, dovrebbe implicare la visualizzazione anche della *strade*.
Gli aspetti dell'implicazione e della costrizione, che caratterizzano il concetto di gerarchia sia per le operazioni che per gli oggetti, servono in due fasi distinte nell'amministrazione del controllo degli accessi. Nel momento in cui si assegna ad un ruolo un certo permesso utilizzo la gerarchia per far rispettare le costrizioni: si potrà associare il nuovo permesso al ruolo solo se tale ruolo possiede già tutti i permessi da cui dipende quello nuovo. Invece nel momento in cui si verifica se la richiesta di un utente per eseguire una determinata operazione possa essere compiuta, si controlla che tale operazione sia associata al ruolo che sta ricoprendo l'utente, oppure che tale operazione dipenda da una di quelle associata all'utente. Per fare un esempio chiarificatore supponiamo una situazione in cui siano definiti i seguenti permessi:

P1 = < leggere, area_deposito >
P2 = < scrivere, deposito >

Supponiamo inoltre che sia stato definito un rapporto di gerarchia tra p1 e p2 tale per cui P2 dipende da P1, o se vogliamo P2 implica P1 (P2 → P1). In fase di gestione dei ruoli, volendo assegnare per un determinato ruolo R1 il permesso di scrittura P2 bisogna verificare che P1 sia già associato a R1, altrimenti l'operazione non può essere completata. Se R1 è già associato a P1 e dunque l'assegnamento di P2 a R1 è valido posso eliminare l'associazione di P1 per R1, essendo questa ricavabile attraverso P2. Infatti se un utente con ruolo R1 richiedesse di *leggere* le *area_deposito* il sistema trovando per R1 il permesso P2 e utilizzando la regola P2 → P1 concederebbe il permesso.

Formalmente definiamo tale aspetto della gerarchia dei permessi come un vincolo:

Definizione: Gerarchia dei Permessi

Siano:
Op1 e Op2 delle operazioni,
Obj1 e Obj2 delle collezioni di classi di feature,
P1 e P2 dei permessi.

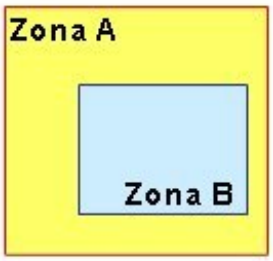
Diciamo che:
P1 = < Op1, Obj1 > dipende da P2 = < Op2, Obj2 > (scritto P1→ P2)

Se e solo se:
l'esistenza del permesso P1 = < Op1, Obj1 >
IMPLICA
l'esistenza del permesso P2 = < Op2, Obj2 > .

Fino ad ora non è stata tenuto in considerazione come la gerarchia dei permessi influisca sui geopermessi e in particolare sulla nuova componente introdotta: la finestra d'autorizzazione (window). Anch'essa infatti è molto importante, seppur in maniera passiva, nella formalizzazione del concetto di gerarchia dei permessi. Sia le costrizioni sugli oggetti che quelle sulle operazioni, che hanno portato alla formalizzazione del *Vincolo dei Permessi*, devono tener in considerazione l'aspetto geografico che tali vincoli implicano. Definendo infatti implicazioni tre permessi che non operano sulla stessa window si potrebbe creare una situazione in cui non sono rispettati i requisiti per cui il modello di controllo degli accessi è stato creato. Se un privilegio ne implica un altro, ci sono delle restrizioni sulla finestra che vanno tenute in considerazione, poiché la finestra d'autorizzazione del permesso implicato deve essere grande almeno come la finestra d'autorizzazione del permesso da cui dipende. In particolare è necessario che la window del permesso implicato contenga la window del permesso da cui dipende. Altrimenti un geo-permesso concederebbe ai suoi geo-permessi dipendenti di operare su una zona più ampia di quella per cui son stati definiti. Vediamo un esempio che esplicita meglio il concetto, supponiamo una situazione in cui siano definiti i seguenti geopermessi:

GeoP1 = < < scrivere, Obj >, zonaB >
GeoP2 = < < leggere, Obj >, zonaA >

Supponiamo inoltre che sia stato definito un rapporto di gerarchia tra i permessi P1=< scrivere, Obj > e P2=< leggere, Obj >, tale per cui P1 implica P2 (P1 → P2). La ZonaB deve essere inclusa nella ZonaA (figura 4.a) affinché non si creino situazioni indesiderate. Se la zonaB non fosse interamente contenuta nella ZonaA (figura 4.b) significherebbe concedere il permesso di *scrittura* in una zona dove non è concesso il permesso di *lettura*.



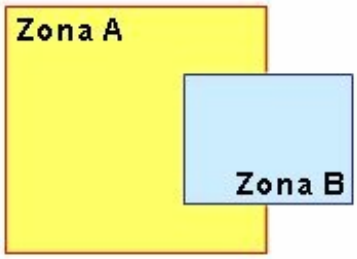


Figura 2.a : Vincoli Rispettati **Figura 2.b : Integrità NON Garantita**

Formalmente il vincolo sulle window che la gerarchia dei permessi implica viene così definito:

Proprietà: Vincolo sulle Window

Siano:
P1 = < Op1, Obj1 >
P2 = < Op2, Obj2 >
due permessi;
e
GeoP1 = < P1, W1 >
GeoP2 = < P2, W2 >
due geopermessi;

Se:
P1 → P2

Allora:
W1 ⊆ W2.

2.4 RUOLI PARAMETRICI

Un estensione che può essere fatta al modello per semplificare la fase di gestione dei ruoli e snellire la fase di definizione dei geo-permessi è l'introduzione del ruolo parametrico. Esso consiste in un ruolo a cui sono associati dei semplici permessi (che chiameremo geo-permessi parametrici) che non hanno una window definita. La window per tali geo-permessi sarà quella passata come parametro al ruolo (ruolo parametrico appunto) al momento della sua definizione (Figura 5). Il ruolo parametrico dunque è fortemente utile quando, nel sistema informativo geografico per cui si sta modellando il controllo degli accessi, ci sono molti ruoli che hanno gli stessi privilegi sugli stessi oggetti ma ognuno specializzato in una determinata zona. Invece di ridefinire i geopermessi per ogni zona, cambiando di volta in volta la window, e poi associarli ai vari ruoli specifici, si

crea un unico ruolo parametrico e il suo relativo geo-permesso parametrico (senza window specificata). Per ogni zona corrisponderà poi un'istanza del ruolo parametrico. Il ruolo parametrico può dunque essere visto come una superclasse astratta di ruoli. I permessi si associano a questa classe astratta, ma la finestra d'autorizzazione viene definita al momento in cui la classe concreta viene istanziata con il passaggio del relativo parametro (la window).

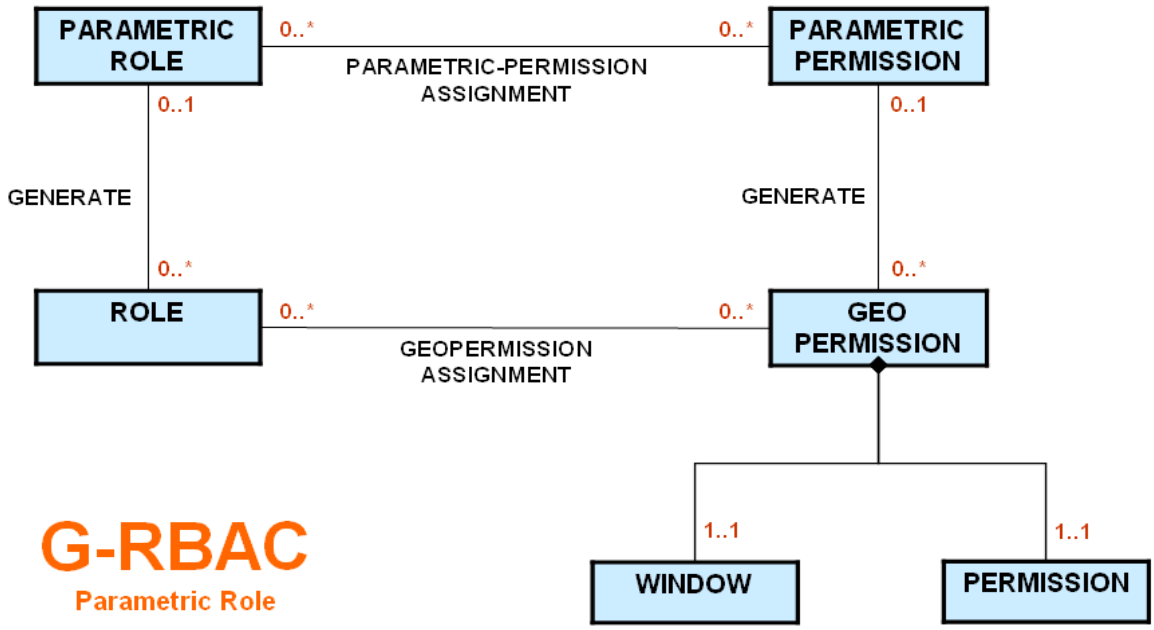


Figura 5: G-RBAC, Ruoli Parametrici

Vediamo dunque quello che è il comportamento in dettaglio del modello G-RBAC esteso coi ruoli parametrici (figura 5). Un ruolo paramerico parametrico è una classe astratta le cui istanze sono i georuoli. Ogni georuolo è un'istanza di un ruolo parametrico ed ogni ruolo parametrico può avere più istanze. Ad ogni ruolo parametrico è abbinato un permesso parametrico che specifica quali azioni e su quali oggetto potranno operare i ruoli una volta istanziati. Il permesso parametrico è infatti formato da un solo permesso. Nel momento in cui un permesso parametrico viene istanziato attraverso il passaggio del parametro window forma un georuolo, analogamente il permesso parametrico viene istanziato, con la stessa window come parametro, in un geo-permesso. Il nuovo georuolo creato è abbinato al nuovo geo-permesso creato.

Un esempio

Supponiamo che si voglia modellare e gestire l'accesso a un sistema informativo geografico distribuito su web. Supponiamo che il sistema si riferisca ad una data zona territoriale che è stata divisa in 4 regioni (A, B, C e D).

Per ogni regione sono presenti delle mansioni quali possono essere:

- **Amministratore**: supponiamo che possa vedere, modificare e analizzare tutte le features della propria regione (GET, INSERT, ANALYSE).
- **Funzionario:** supponiamo che possa vedere e modificare tutte le features della propria regione (GET, INSERT).
- **Impiegato**: supponiamo che possa vedere solo alcune features della regione in cui opera (GET).

L'implementazione di tale gerarchia con il modello G-RBAC senza ruoli parametrici per la regione C sarebbe:

Ruoli:	*Operazioni:*	*Oggetti:*
Amministratore_C	Get	Obj1
Funzionario_C	Insert	Obj2
Impiegato_C	Analyse	Obj3

GeoPermessi:	*GeoPermission Assignment:*
Gp_amm_C #1 = < <Get, All>, ZonaC >	Amministratore_C → GP_amm_C #1
GP_amm_C #2 = < <Insert, All>, ZonaC >	Amministratore_C → GP_amm_C #2
GP_amm_C #3 = < <Analyse, All>, ZonaC >	Amministratore_C → GP_amm_C #3
GP_funz_C #1 = < <Get, All>, ZonaC >	Funzionario_C → GP_funz_C #1
GP_funz_C #2 = < <Insert, All>, ZonaC >	Funzionario_C → GP_funz_C #2
GP_imp_C #1 = < <Get, Obj1> , ZonaC >	Impiegato_C → GP_imp_C #1
GP_imp_C #2 = < <Get, Obj2> , ZonaC >	Impiegato_C → GP_imp_C #2

Utilizzando tale modello le figure analoghe (stessi permessi sulle stesse features) delle altre regioni comporterebbero la creazione di nuovi ruoli e nuovi permessi:

Ruoli:	*Operazioni:*	*Oggetti:*
Amministratore_A	Get	Obj1
Funzionario_A	Insert	Obj2
Impiegato_A	Analyse	Obj3
Amministratore_B		
Funzionario_B		
Impiegato_B		
Amministratore_C		
Funzionario_C		
Impiegato_C		
Amministratore_D		
Funzionario_D		
Impiegato_D		

GeoPermessi:	*GeoPermission Assignment:*		
GP_amm_A #1 = < <Get, All>, ZonaA >	Amministratore_A	→	GP_amm_A #1
GP_amm_A #2 = < <Insert, All>, ZonaA >	Amministratore_A	→	GP_amm_A #2
GP_amm_A #3 = < <Analyse, All>, ZonaA >	Amministratore_A	→	GP_amm_A #3
GP_funz_A #1 = < <Get, All>, ZonaA >	Funzionario_A	→	GP_funz_A #1
GP_funz_A #2 = < <Insert, All>, ZonaA >	Funzionario_A	→	GP_funz_A #2
GP_imp_A #1 = < <Get, Obj1> , ZonaA >	Impiegato_A	→	GP_imp_A #1
GP_imp_A #2 = < <Get, Obj2> , ZonaA >	Impiegato_A	→	GP_imp_A #2
GP_amm_B #1 = < <Get, All>, ZonaB >	Amministratore_B	→	GP_amm_B #1
GP_amm_B #2 = < <Insert, All>, ZonaB >	Amministratore_B	→	GP_amm_B #2
GP_amm_B #3 = < <Analyse, All>, ZonaB >	Amministratore_B	→	GP_amm_B #3
GP_funz_B #1 = < <Get, All>, ZonaB >	Funzionario_B	→	GP_funz_B #1
GP_funz_B #2 = < <Insert, All>, ZonaB >	Funzionario_B	→	GP_funz_B #2
GP_imp_B #1 = < <Get, Obj1> , ZonaB >	Impiegato_B	→	GP_imp_B #1
GP_imp_B #2 = < <Get, Obj2> , ZonaB >	Impiegato_B	→	GP_imp_B #2
GP_amm_C #1 = < <Get, All>, ZonaC >	Amministratore_C	→	GP_amm_C #1
GP_amm_C #2 = < <Insert, All>, ZonaC >	Amministratore_C	→	GP_amm_C #2
GP_amm_C #3 = < <Analyse, All>, ZonaC >	Amministratore_C	→	GP_amm_C #3
GP_funz_C #1 = < <Get, All>, ZonaC >	Funzionario_C	→	GP_funz_C #1
GP_funz_C #2 = < <Insert, All>, ZonaC >	Funzionario_C	→	GP_funz_C #2
GP_imp_C #1 = < <Get, Obj1> , ZonaC >	Impiegato_C	→	GP_imp_C #1
GP_imp_C #2 = < <Get, Obj2> , ZonaC >	Impiegato_C	→	GP_imp_C #2
GP_amm_D #1 = < <Get, All>, ZonaD >	Amministratore_D	→	GP_amm_D #1
GP_amm_D #2 = < <Insert, All>, ZonaD >	Amministratore_D	→	GP_amm_D #2
GP_amm_D #3 = < <Analyse, All>, ZonaD >	Amministratore_D	→	GP_amm_D #3
GP_funz_D #1 = < <Get, All>, ZonaD >	Funzionario_D	→	GP_funz_D #1
GP_funz_D #2 = < <Insert, All>, ZonaD >	Funzionario_D	→	GP_funz_D #2
GP_imp_D #1 = < <Get, Obj1> , ZonaD >	Impiegato_D	→	GP_imp_D #1
GP_imp_D #2 = < <Get, Obj2> , ZonaD >	Impiegato_D	→	GP_imp_D #2

Introducendo invece il concetto di ruolo parametrico sarebbe possibile definire solo una volta il ruolo che si ripete nelle varie regioni:

Ruoli parametrici:	*Operazioni:*	*Oggetti:*
Amministratore( )	Get	Obj1
Funzionario( )	Insert	Obj2
Impiegato( )	Analyse	Obj3
Istanze:		
Amministratore(ZonaA)		
Funzionario(ZonaA)		
Impiegato(ZonaA)		
Amministratore(ZonaB)		
Funzionario(ZonaB)		
Impiegato(ZonaB)		
Amministratore(ZonaC)		
Funzionario(ZonaC)		
Impiegato(ZonaC)		
Amministratore(ZonaD)		
Funzionario(ZonaD)		
Impiegato(ZonaD)		

Permessi parametrici:	*Permission Assignment:*		
GP_amm _1 = < Get, All>, **** >	Amministratore ( )	→	GP_amm_1
GP_amm_2 = < Insert, All>, **** >	Amministratore ( )	→	GP_amm_2
GP_amm_3 = < Analyse, All>, **** >	Amministratore ( )	→	GP_amm_3
GP_funz_1 = < Get, All>, **** >	Funzionario ( )	→	GP_funz_1
GP_funz_2 = < Insert, All>, **** >	Funzionario ( )	→	GP_funz_2
GP_imp_1 = < Get, Obj1> , **** >	Impiegato ( )	→	GP_imp_1
GP_imp_2 = < Get, Obj2> , **** >	Impiegato ( )	→	GP_imp_2

Come si vede anche in questo esempio ridotto i vantaggi introdotti nella fase di amministrazione sono notevoli. All'aumentare delle zone da istanziare cresce solo la tabella dei ruoli istanziati. In caso di una modifica del geo-permesso poi, senza i ruoli parametrici ci sarebbero centinaia di geopermessi da modificare. Col ruolo parametrico, una qualsiasi modifica, riguarderebbe il solo geo-permesso parametrico.

2.5 RUOLI DINAMICI

Un sistema informativo geografico distribuito su Web presuppone l'utilizzo del sistema da parte di utenti con locazione geografica potenzialmente mobile se non addirittura fortemente dinamica. Si pensi alla possibilità di avere utenti connessi attraverso telefoni cellulari che supportino la tecnologia Wap, PDA o anche portatili con collegamento Wireless alla rete. Questi presupposti implicano l'eventualità di utenti che modificano la propria locazione anche durante una sessione di lavoro. In questo caso potrebbe essere utile avere dei ruoli che un per un determinato utente connesso si attivano solo se tale utente si trova in una determinata finestra d'autorizzazione.

Questi tipi di ruoli che si è voluto introdurre nel modello sono i ruoli dinamici. Se al momento della connessione l'utente si trova in una determinata zona gli potrebbe venir richiesto se intende attivare uno dei ruoli dinamici disponibili per lui solo in quella particolare window in cui si trova. Eventualmente l'attivazione può avvenire in automatico senza richiedere nulla all'utente. Analogamente, se durante la connessione l'utente dovesse entrare in una window per cui possiede dei ruoli dinamici, questi potrebbero essere attivati in automatico dal sistema o, come al momento della connessione, su conferma dell'utente. Ovviamente un requisito fondamentale per implementare i ruoli dinamici è la capacità di localizzare il luogo da cui l'utente è connesso.

L'introduzione del ruolo dinamico implica la definizione di un nuovo stato in cui un utente può settare un ruolo. Nel modello standard RBAC un ruolo per cui un utente è abilitato può essere attivo o inattivo per tale utente. L'utente può decidere in sostanza se ricoprire o no il ruolo. Supponiamo invece che un utente decida di ricoprire un ruolo dinamico, ma che non si trovi nella finestra d'autorizzazione abbinata a tale ruolo dinamico. Descriviamo questa situazione dicendo che il ruolo è *selezionato*. Un ruolo selezionato è un ruolo che un utente connesso al sistema ha deciso di ricoprire tra quelli per cui è abilitato. Se si tratta di un ruolo non dinamico esso diventa automaticamente un ruolo attivo. Se si tratta invece di un ruolo dinamico, verrà attivato solo se l'utente sarà localizzato all'interno della window. Se durante la sessione di lavoro l'utente entra nella finestra d'autorizzazione di un ruolo dinamico selezionato, questo passa automaticamente nello stato di attivato. Viceversa, se durante la sessione di lavoro l'utente esce dalla finestra d'autorizzazione di un ruolo dinamico attivato, questo passa automaticamente nello stato di selezionato.

Vediamo dunque quello che è il comportamento in dettaglio del modello G-RBAC esteso coi ruoli dinamici (figura 6). Un ruolo dinamico è una sottoclasse del ruolo semplice. Ne eredita infatti tutti gli attributi e le operazioni. Un utente può essere abilitato a ricoprire più ruoli dinamici. L'utente possiede inoltre una posizione, che ovviamente è unica. Questa posizione determina l'attivazione del ruolo dinamico.

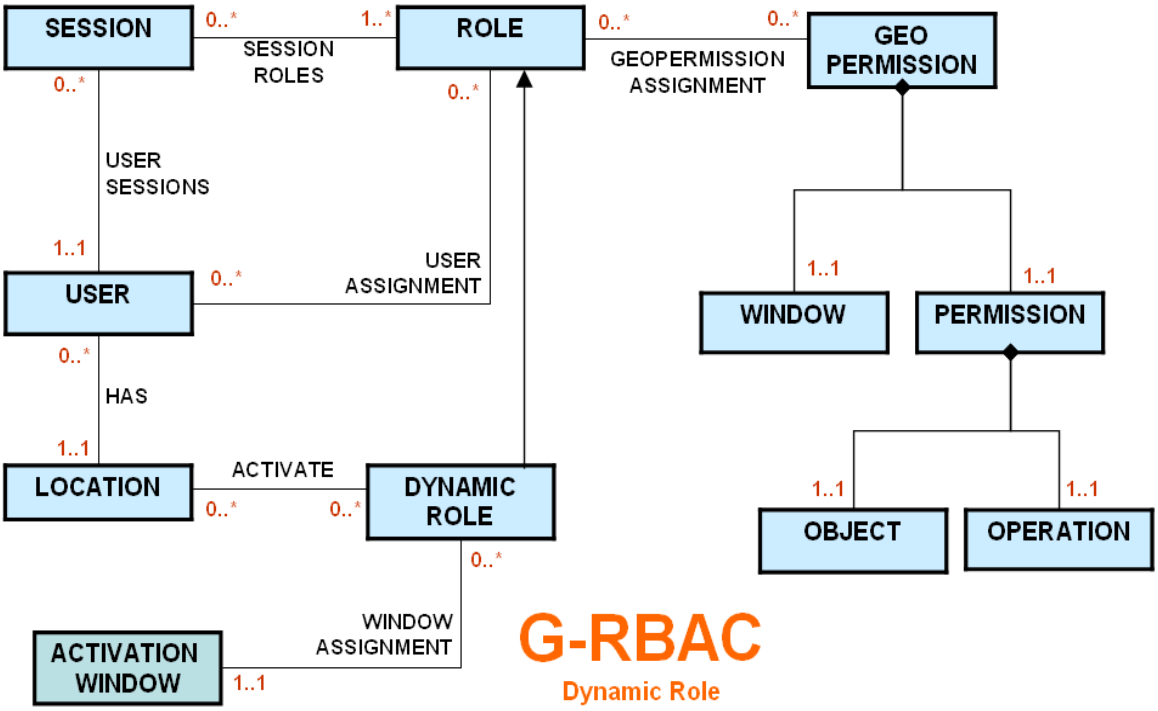


Figura 6: G-RBAC, Ruoli Dinamici

Per memorizzare la posizione dell'utente si può utilizzare la sessione. La sessione infatti, può essere estesa con una componente geografica per memorizzare la posizione da cui sta effettuando la connessione l'utente. L'entità che memorizza informazioni sulla connessione dell'utente è la sessione ed è dunque in essa che la posizione dell'utente verrà allocata. La sessione diventa allora una Feature spaziale e può essere utilizzata per interagire con altri elementi di tipo spaziale come la window d'autorizzazione.
Rimane da stabilire che tipo di geometria associare alla posizione di un utente. Identificando un utente con delle coordinate in un sistema di riferimento, la sua posizione potrebbe essere rappresentata come un punto. Benché una geometria di tipo punto sarebbe la scelta più intuitiva e logica, in realtà, anche una geometria di tipo area o di tipo lineare potrebbero descrivere la locazione di un utente. Nel caso di connessioni Wap attraverso dispositivi cellulari ad esempio, può essere identificata al massimo la cella di copertura che contiene l'utente. In tal caso la geometria da abbinare alla sua sessione sarebbe di tipo area. Oppure la locazione di utenti in movimento su rotte prestabilite, come quelle di aerei o treni, potrebbe essere descritta in termini di tratte. In tal caso la geometria da abbinare alla sessione di tali utenti sarebbe di tipo lineare.

Un Esempio

Supponiamo che esista una figura che abbia un ruolo preciso ma da svolgere in diverse regioni. Ad esempio un funzionario ispettore che svolga la propria mansione in itinere nelle varie regioni. Utilizzando i ruoli semplici si potrebbero creare dei permessi validi per una window che ricopra tutte le regioni. Questa soluzione tuttavia, comporterebbe un aumento rilevante di traffico peraltro inutile.

Utilizzando invece i ruoli parametrici si potrebbe autorizzare l'utente per tutti i ruoli istanziati ma rimarrebbe il problema del cambio dinamico di regione durante la sessione aperta. Nel caso in cui il cambio di regione sia abbastanza frequente risulterebbe scomodo anche riassegnare ogni volta il ruolo all'utente. Si potrebbe utilizzare allora un ruolo dinamico per il funzionario ispettore tale per cui ogni volta che cambia regione, anche durante una sessione in corso, automaticamente cambia il ruolo che ricopre. Il luogo da cui è attiva la connessione identifica la regione su cui si lavora. Ovviamente il sistema deve monitorare costantemente, o con una frequenza prestabilita, la posizione da cui l'utente è connesso.

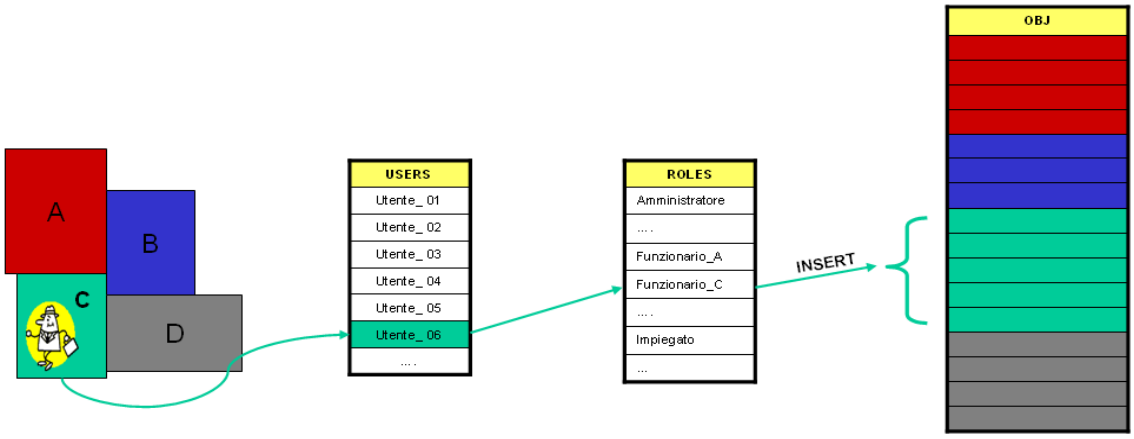


Figura 7.a: Utente localizzato nella zona C

Dapprima l'utente effettua la connessione e viene localizzato nella regione C. Allora verrà attivato il ruolo dinamico di funzionario itinerante con window d'autorizzazione corrispondente alla regione C.
Se durante la sessione l'utente cambia regione di lavoro, passando ad esempio nella regione A, il sistema automaticamente disattiva il ruolo dinamico di funzionario itinerante della regione C e attiva quello della regione A:

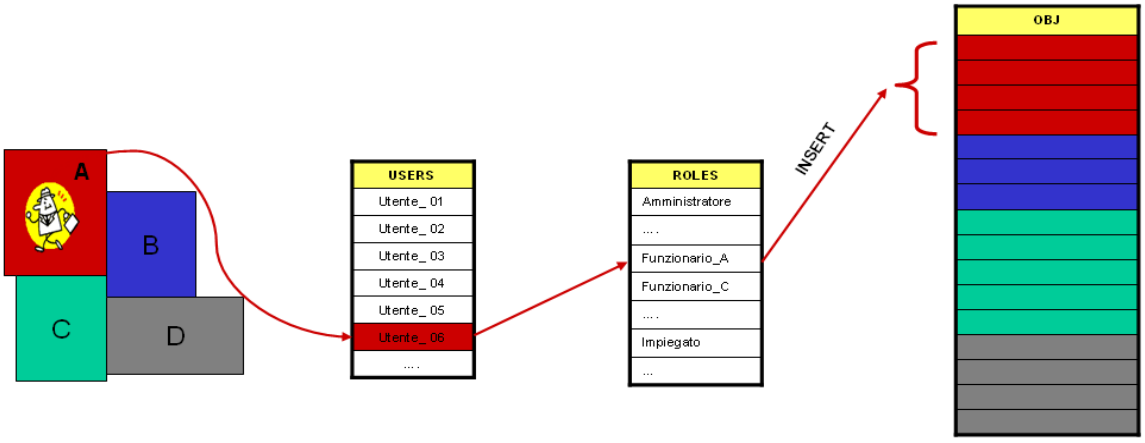


Figura 7.b: Utente localizzato nella zona A

Ovviamente non c'è nessun vincolo tra la finestra del ruolo dinamico e le finestre dei permessi associati al ruolo dinamico.

2. 6 SPECIFICHE FUNZIONALI

In questo paragrafo vengono presentate le specifiche necessarie al corretti funzionamento del sistema. Dapprima sono elencate le funzioni amministrative di base che devono essere implementate. Seguono poi le funzioni amministrative per i ruoli parametrici, per i ruoli dinamici e per la gerarchia dei permessi. La presenza ed il corretto funzionamenti di questo insieme di funzioni garantisce la consistenza del modello. Vengono presentate poi le funzioni per la gestione a runtime del sistema che devono essere implementate per permettere una corretta interazione dell'applicativo in fase d'esecuzione. Vengono elencate infine, delle funzioni facoltative per il monitoraggio del sistema.

4.6.1 Funzioni amministrative di base

AddUser Questo comando genera un nuovo utente G-RBAC ed è valido solo se l'utente che si vuole creare non appartiene già all'insieme USERS che verrà aggiornato dopo l'inserimento. Al momento della creazione il nuovo utente non ha nessuna sessione aperta.

DeleteUser Questo comando elimina un utente appartenente all'insieme USERS dal database G-RBAC. L'operazione aggiorna le tabelle USERS, UA, e la funzione *assigned_users*. Per quanto riguarda le sessioni aperte dall'utente viene lasciata libera scelta su come comportarsi: si può aspettare la chiusura della sessione oppure forzarla.

AddRole Questo comando crea un nuovo ruolo ed è valido solo se il ruolo che si vuole creare non appartiene già all'insieme ROLES. Verranno aggiornati dopo l'inserimento l'insieme ROLES e le funzioni *assigned_users* e *assigned_geopermission*. Inizialmente al nuovo ruolo non sono associati nessun utente e nessun permesso.

DeleteRole Questo comando elimina un ruolo appartenente all'insieme ROLES dal database G-RBAC. Per quanto riguarda le sessioni aperte in cui il ruolo da eliminare è attivo viene lasciata libera scelta su come comportarsi: si può aspettare la chiusura delle sessioni coinvolte, forzarla alla chiusura oppure eliminare il ruolo lasciando attive le sessioni.

AssignUser*(User, Role)* Questo comando assegna un utente ad un ruolo. Il comando è valido solo se l'utente appartiene all'insieme USERS, il ruolo appartiene all'insieme ROLES e non è già stato generato quest'assegnamento. In seguito all'operazione verranno aggiornati l'insieme UserAssignement e la funzione *assigned_users*.

DeassignUser *(User, Role)* Questo comando cancella l'appartenenza di un utente ad un ruolo. Il comando è valido solo se l'utente appartiene all'insieme USERS, il ruolo appartiene all'insieme ROLES e l'utente è assegnato al ruolo. Per quanto riguarda le

sessioni aperte in cui l'utente ricopre il ruolo da cui lo si vuole togliere viene lasciata libera scelta su come comportarsi: si può aspettare la chiusura delle sessioni coinvolte, forzarla alla chiusura oppure eliminare l'assegnamento lasciando attive le sessioni.

AddGeoPermission*(Operation, Object, Window)* Questo comando crea un nuovo permesso geografico ed è valido solo se il geo-permesso che si vuole creare non appartiene già all'insieme GEOPERMISSIONS, l'insieme di comandi *Operation* appartiene all'insieme OPERATIONS, la raccolta di oggetti *Object* appartiene all'insieme OBJECTS e la finestra d'autorizzazione *Window* appartiene all'insieme WINDOWS. Verranno aggiornati dopo l'inserimento l'insieme GEOPERMISSIONS e la funzione *assigned_permission*. Inizialmente il nuovo permesso non è associato a nessun ruolo.

DeleteGeoPermission*(GeoPermission)* Questo comando elimina il permesso *GeoPermission* appartenente all'insieme GEOPERMISSIONS dal database G-RBAC. Per quanto riguarda le sessioni aperte in cui è attivo qualche ruolo associato al permesso da eliminare viene lasciata libera scelta su come comportarsi: si può aspettare la chiusura delle sessioni coinvolte, forzarla alla chiusura oppure eliminare il permesso lasciando attive le sessioni.

GrantGeoPermission*(Role, GeoPermission)* Questo comando autorizza un ruolo ad eseguire un operazione su un oggetto del database (concede un geo-permesso). L'operazione è valida solo se la tripla operazione-oggetto-window rappresenta un geo-permesso e il ruolo appartiene all'insieme ROLES.

RevokeGeoPermission*(Role, GeoPermission)* Questo comando revoca il permesso di eseguire un'operazione su di un oggetto. Il comando è valido solo se la tripla operazione-oggetto-window rappresenta un geo-permesso ed il ruolo appartiene all'insieme ROLES. Per quanto riguarda le sessioni aperte in cui l'utente sta eseguendo l'operazione tramite il permesso che si vuole revocare viene lasciata libera scelta su come comportarsi: si può aspettare la chiusura delle sessioni coinvolte, forzarla alla chiusura oppure eliminare l'assegnamento lasciando attive le sessioni.

AddWindow Questo comando crea una nuova finestra d'autorizzazione. Verrà aggiornato dopo l'inserimento l'insieme WINDOW. Inizialmente la nuova finestra non è associata a nessun geo-permesso.

ModifyWindow Questo comando modifica la geometria di una finestra d'autorizzazione già esistente all'interno dell'insieme WINDOW. La presenza della gerarchia dei permessi obbliga a fare alcune considerazioni sulle conseguenze di tale operazione.

2.6.2 Funzioni amministrative per i ruoli parametrici

AddParametricRole Questo comando crea un nuovo ruolo parametrico ed è valido solo se il ruolo che si vuole creare non appartiene già all'insieme PARAMETRICROLES. Verranno aggiornati dopo l'inserimento l'insieme PARAMETRICROLES e la funzione *assigned_parametric_permission*. Inizialmente al nuovo ruolo parametrico non è associato nessun permesso.

DeleteParametricRole Questo comando elimina un ruolo parametrico e tutte le sue istanze appartenenti all'insieme ROLES dal database G-RBAC. Per quanto riguarda le sessioni aperte in cui il ruolo da eliminare è attivo viene lasciata libera scelta su come comportarsi: si può aspettare la chiusura delle sessioni coinvolte, forzarla alla chiusura oppure eliminare il ruolo lasciando attive le sessioni.

AddIstanceParametricRole (*ClassRole, Window*) Questo comando istanzia un nuovo ruolo parametrico della classe *ClassRole* associandolo alla finestra d'autorizzazione *window*. E' valido solo se il ruolo parametrico è stato definito, la *window* è stata creata e non è ancora stato istanziato alcun ruolo parametrico di classe *ClassRole* su finestra *Window*. Verranno aggiornati dopo l'inserimento l'insieme ROLES e le funzioni *assigned_users* e *assigned_permission*. Inizialmente al nuovo ruolo non sono associati nessun utente e nessun permesso.

DeleteIstanceParametricRole (*Role*) Questo comando elimina un istanza di un ruolo parametrico appartenente all'insieme ROLES dal database G-RBAC. Per quanto riguarda le sessioni aperte in cui il ruolo da eliminare è attivo viene lasciata libera scelta su come comportarsi: si può aspettare la chiusura delle sessioni coinvolte, forzarla alla chiusura oppure eliminare il ruolo lasciando attive le sessioni.

AddParametricPermission*(Operation, Object)* Questo comando crea un nuovo permesso parametrico ed è valido solo se il permesso che si vuole creare non appartiene già all'insieme PARAMETRICPERMISSIONS, l'insieme di comandi *Operation* appartiene all'insieme OPERATIONS e la raccolta di oggetti *Object* appartiene all'insieme OBJECTS. Verranno aggiornati dopo l'inserimento l'insieme PARAMETRICPERMISSIONS e la funzione *assigned_parametric_permission*. Inizialmente il nuovo permesso non è associato a nessun ruolo parametrico.

DeleteParametricPermission*(ParametricPermission)* Questo comando elimina il permesso *ParametricPermission* appartenente all'insieme PARAMETRICPERMISSIONS dal database G-RBAC. Per quanto riguarda le sessioni aperte in cui è attivo qualche ruolo associato al permesso da eliminare viene lasciata libera scelta su come comportarsi: si può aspettare la chiusura delle sessioni coinvolte, forzarla alla chiusura oppure eliminare il permesso lasciando attive le sessioni.

GrantParametricPermission*(ParametricRole, ParametricPermission)* Questo comando autorizza un ruolo ad eseguire un operazione su un oggetto del database (concede un permesso). L'operazione è valida solo se la tripla operazione-oggetto-window rappresenta un permesso, il ruolo appartiene all'insieme ROLES.

RevokeParametricPermission*(ParametricPermission)* Questo comando revoca il permesso di eseguire un'operazione su di un oggetto. Il comando è valido solo se la tripla operazione-oggetto-window rappresenta un permesso ed il ruolo appartiene all'insieme ROLES. Per quanto riguarda le sessioni aperte in cui l'utente sta eseguendo l'operazione tramite il permesso che si vuole revocare viene lasciata libera scelta su come comportarsi: si può aspettare la chiusura delle sessioni coinvolte, forzarla alla chiusura oppure eliminare l'assegnamento lasciando attive le sessioni.

2.6.3 Funzioni amministrative per i ruoli dinamici

AddDynamicRole *(Window)*Questo comando crea un nuovo ruolo dinamico attivabile se l'utente si connette dall'interno della *Window*. E' valido solo se il ruolo che si vuole creare non appartiene già all'insieme DYNAMICROLES e la *Window* appartiene all'insieme WINDOW. Verranno aggiornati dopo l'inserimento l'insieme DYNAMICROLES e la funzione *assigned _permission*. Inizialmente al nuovo ruolo dinamico non è associato nessun permesso.

DeleteDynamicRole Questo comando elimina un ruolo dal database G-RBAC. Per quanto riguarda le sessioni aperte in cui il ruolo da eliminare è attivo viene lasciata libera scelta su come comportarsi: si può aspettare la chiusura delle sessioni coinvolte, forzarla alla chiusura oppure eliminare il ruolo lasciando attive le sessioni.

GrantDynamicPermission*(DynamicRole, GeoPermission)* Questo comando associa un ruolo dinamico ad un permesso. E' identico per specifiche e requisiti a *GrantGeoPermission(Role, GeoPermission).*

RevokeDynamicPermission*(DynamicRole, GeoPermission)* Questo comando revoca un permesso ad un ruolo dinamico. E' identico per specifiche e requisiti a *RevokeGeoPermission(Role, GeoPermission).*

4.6.4 Funzioni amministrative per la gerarchia dei permessi

Nelle seguenti specifiche ci si riferisce col termine permessi indistintamente sia ai permessi geografici normali che a quelli parametrici. Non ci sono vincoli all'appartenenza di entrambi alla gerarchia dei permessi.

AddHierarchy *(Perm_asc, Perm_desc)* Questo comando genera una relazione di implicazione diretta tra due permessi esistenti (*Perm_asc* → *Perm_desc*). L'operazione è valida solo se *Perm_asc* e *Perm_desc* appartengono all'insieme PERMISSIONS*, Perm_asc* non è già un ascendente immediato di *Perm_desc* e *Perm_desc* non implica *Perm_asc* (per evitare cicli nel grafo delle gerarchie).

DeleteHierarchy*(Perm_asc, Perm_desc)* Questo comando elimina una relazione di implicazione diretta esistente tra due permessi (Perm_asc → Perm_desc). L'operazione è valida solo se Perm_asc e Perm_desc appartengono all'insieme PERMISSIONS e Perm_asc è un ascendente immediato di Perm_desc.

AddAscendent*(*Perm*_asc, Perm_desc)* Questo comando crea un nuovo permesso Perm_asc e lo inserisce nella gerarchia dei permessi definendolo ascendente immediato di Perm_desc. L'operazione è valida solo se Perm_asc non appartiene all'insieme PERMISSIONS e Perm_desc appartiene all'insieme PERMISSIONS. Le condizioni di validità sono verificate da due funzioni che l'operazione richiama: AddPermission(Perm_asc) e AddHierarchy(Perm_asc, Perm_desc).

AddDescendent*(Perm_asc, Perm_desc)* Questo comando crea un nuovo permesso Perm_desc e lo inserisce nella gerarchia dei permessi definendolo discendente immediato di Perm_asc. L'operazione è valida solo se Perm_desc non appartiene all'insieme PERMISSIONS e Perm_asc appartiene all'insieme PERMISSIONS. Le condizioni di validità sono verificate da due funzioni che l'operazione richiama: AddRole(Perm_desc) e AddInheritance(Perm_asc, Perm_desc).

2.6.5 Funzioni per la Gestione a Runtime del Sistema

CreateSession*(user, roles, session)* Questa funzione crea una nuova sessione con proprietario l'utente dato e un insieme di ruoli attivi. L'operazione è valida solo se l'utente appartiene all'insieme USERS e l'insieme dei ruoli attivi è un sott'insieme dei ruoli assegnabili all'utente.

DeleteSession*(user,session)* Questa funzione cancella la sessione data che ha come proprietario l'utente dato. L'operazione è valida solo se la sessione appartiene all'insieme SESSSIONS delle sessioni aperte, l'utente appartiene all'insieme USERS ed è il proprietario della sessione in questione.

AddActiveRole*(user,session,role)* Questa funzione attiva un ruolo aggiungendolo all'insieme dei ruoli attivi di una sessione per un dato utente. L'operazione è valida solo se l'utente appartiene all'insieme USERS, il ruolo appartiene all'insieme ROLES, la sessione appartiene all'insieme SESSIONS, l'utente può essere assegnato al ruolo e l'utente è il proprietario della sessione.

DropActiveRole *(user,session,role)* Questa funzione disattiva un ruolo togliendolo dall'insieme dei ruoli attivi di una sessione per un dato utente. L'operazione è valida solo se l'utente appartiene all'insieme USERS, la sessione appartiene all'insieme SESSIONS, l'utente è il proprietario della sessione e il ruolo appartiene dall'insieme dei ruoli attivi della sessione.

CheckAccess*(session, operation,object)* Questa funzione ritorna un booleano specificando se il soggetto di una data sessione è autorizzato o meno ad eseguire una determinata operazione su un dato oggetto. L'operazione è valida solo se la sessione appartiene all'insieme SESSIONS, l'oggetto appartiene all'insieme OBJS e l'operazione appartiene all'insieme OPS. Il soggetto della sessione ha il permesso di eseguire l'operazione solo se il permesso è assegnato ad almeno uno dei ruoli attivi della sessione.

AddDynamicRole*(User, Session, DynamicRole)* Questa funzione attiva un ruolo dinamico aggiungendolo all'insieme dei ruoli attivi di una sessione per un dato utente. L'operazione è valida solo se l'utente appartiene all'insieme USERS, il ruolo appartiene all'insieme DYNAMICROLES, la sessione appartiene all'insieme SESSIONS, l'utente può essere assegnato al ruolo e l'utente è il proprietario della sessione.

DropDynamicRole*(User, Session, DynamicRole)* Questa funzione disattiva un ruolo dinamico togliendolo dall'insieme dei ruoli attivi di una sessione per un dato utente. L'operazione è valida solo se l'utente appartiene all'insieme USERS, la sessione appartiene all'insieme SESSIONS, l'utente è il proprietario della sessione e il ruolo appartiene dall'insieme dei ruoli attivi della sessione.

2.6.6 Funzioni per il Monitoraggio del Sistema

AssignedUsers*(role)* Questa funzione restituisce l'insieme degli utenti assegnati ad un ruolo dato. La funzione è valida solo se il ruolo appartiene all'insieme ROLES o DYNAMICROLES o PARAMETRICROLES.

AssignedRoles*(user)* Questa funzione restituisce l'insieme dei ruoli assegnati ad un utente dato. La funzione è valida solo se l'utente appartiene all'insieme USERS.

RolePermissions*(role)* Questa funzione restituisce l'insieme dei permessi (tripla operazione-oggetto-window) assegnati ad un ruolo dato. La funzione è valida solo se il ruolo appartiene all'insieme ROLES o DYNAMICROLES o PARAMETRICROLES.

UserPermissions*(user)* Questa funzione restituisce l'insieme dei permessi (tripla operazione-oggetto-window) che un dato utente può ottenere attraverso i ruoli che può ricoprire. La funzione è valida solo se l'utente appartiene all'insieme USERS.

SessionRoles*(session)* Questa funzione restituisce l'insieme dei ruoli attivi di una data sessione. L'operazione è valida solo se la sessione appartiene all'insieme SESSIONS.

SessionPermissions*(session)* Questa funzione restituisce l'insieme dei permessi di una data sessione, ovvero i permessi assegnati ai ruoli attivi della sessione. L'operazione è valida solo se la sessione appartiene all'insieme SESSIONS.

2.7 SCHEMA RIASSUNTIVO DEL MODELLO G-RBAC

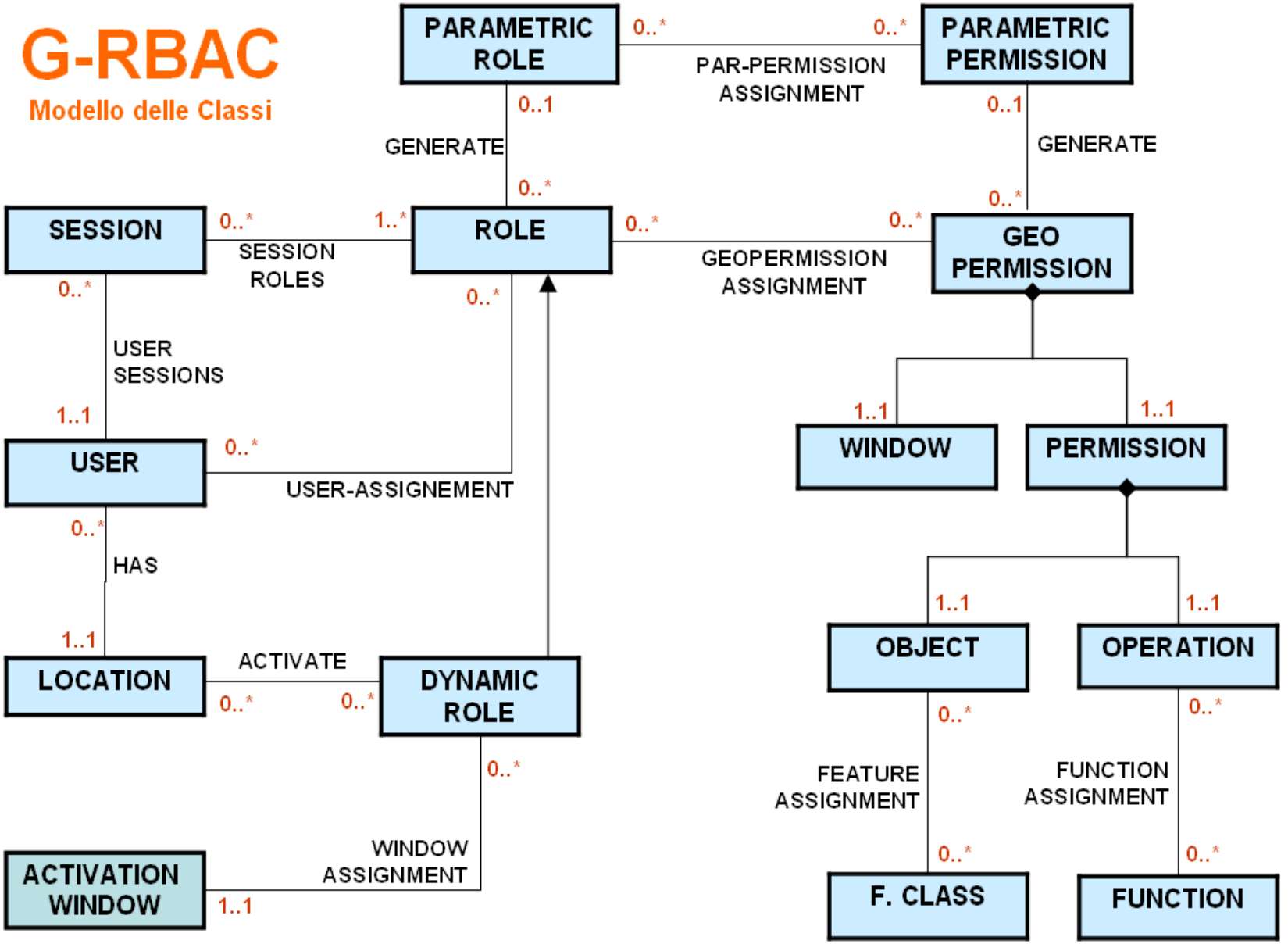
Viene utilizzato lo schema di modellazione UML per presentare quali sono i requisiti e le funzionalità introdotte in G-RBAC.

Modello delle classi

Come è stato detto G-RBAC è un'estensione geografica del modello noto di controllo degli accessi basato sui ruoli RBAC. Ai ruoli vengono associati dei geopermessi anziché dei semplici permessi. Un geo-permesso è formato da un permesso e da una finestra d'autorizzazione (window). Un permesso è costituito a sua volta da una operazione (insieme di funzionalità messe a disposizione dal Web Service) e da un oggetto (insieme di Feature). Di seguito viene presentato il diagramma con il modello delle classi. Successivamente vengono elencate le specifiche dettagliate delle singole classi che sono state omesse per motivi di spazio nel diagramma.

G-RBAC

Modello delle Classi



USER
FirstName SecondName
AddUser DeleteUser AssignUser(Role) DeassignUser(Role)

SESSION
Id Date
AddSession(User) DeleteSession AddActiveRole(Role) DropActiveRole(Role)

LOCATION
Coordinate
UpdateLocation AddDynamicRole(DynamicRole) DropDynamiceRole(DynamicRole)

ROLE
Name
AddRole DeleteRole GrantGeoPermission(GeoPermission) RevokeGeoPermission(GeoPermission)

PARAMETRIC ROLE
ParametricName
AddParametricRole DeleteParametricRole GrantParamPermission(ParametricPermission) RevokeParamPermission(ParametricPermission)

PARAMETRIC PERMISSION
ParametricName
AddParamPermission(Operation, Object) DeleteParamPermission

GEO ROLE
GeoName Window
AddGeoRole(ParametricRole, Window) DeleteGeoRole

GEO PERMISSION
Name Permission Window
AddGeoPermission(Permission, Window) DeleteGeoPermission

DYNAMIC ROLE
Role
AddDynamicRole DeleteDynamicRole SetDynamicRole(Window)

PERMISSION
Name Operation Object
AddPermission(Operation, Object) DeletePermission AddHierarchy(Permission, Permission) DeleteHierarchy AddAscendent(Permission, Permission) AddDescendent(Permission, Permission)

WINDOW
Name Geometry
AddWindow DeleteWindow SetWindow

OPERATION
Name
AddOperation DeleteOperation AddFunction(Function) DeleteFunction(Function)

FUNCTION
Name ServiceSpecification
AddFunction DeleteFunction

FEATURE CLASS
Name Geometry
AddFeatureClass DeleteFeatureClass

OBJECT
Name
AddObject DeleteObject AddFeature(FeatureClass) DeleteFeature(FeatureClass)

Capitolo 3*:*
PROTOTIPO PER G-RBAC

A partire dal modello proposto è stato realizzato un prototipo applicativo che implementa le specifiche introdotte nel capitolo precedente. Lo scopo è il controllo degli accessi degli utenti ad un sistema remoto contenente informazioni geografiche.
E' stata sviluppata la parte che implementa le funzioni specificate per il concetto di geo-permesso, di ruolo parametrico e di ruolo dinamico. Il capitolo si sviluppa come descritto di seguito: nella prima parte viene presentata l'architettura generale del prototipo e ne viene descritto sommariamente il funzionamento; viene presentato il livello Memorizzazione Dati con una descrizione dettagliata del database GRBAC contenete le informazioni per il controllo degli accessi; successivamente viene presentato il livello Applicazione elencando le principali funzioni utilizzate; segue una breve descrizione del livello Presentazione, un resoconto degli accorgimenti progettuali adottati e degli aspetti implementativi rilevanti; viene descritto poi il caso di studio utilizzato su cui è stato testato il modello per il controllo degli accessi; in conclusione viene presentata l'interfaccia del sistema e introdotti quelli che possono essere gli argomenti futuri di ricerca e sviluppo.

3.1 ARCHITETTURA GENERALE DEL PROTOTIPO

L'applicazione proposta è stata sviluppata affinché le features spaziali siano raggiungibili attraverso accessi interattivi con il Web, dunque l'architettura dell'amministrazione G-RBAC è organizzata sulla nota architettura a tre livelli, Presentazione, Applicazione e Archiviazione Dati.

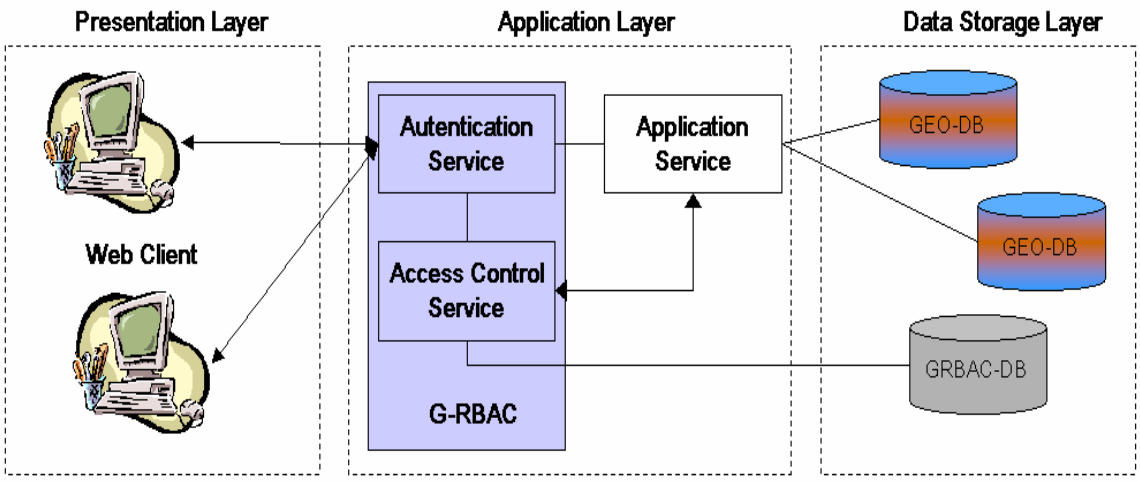


Figura 1: Architettura generale per il Web Service

Il livello Presentazione dal lato client, di solito, consiste in pagine HTML e ASP o in programmi specializzati, come possono essere del codice Java o dei plag-in.
Nel prototipo sviluppato le features spaziali sono trasferite dal server al client sotto forma di immagini vettoriali. E' necessario per l'esposizione dei dati lato client, un browser di Web esteso con un plug in o un programma Java. In particolare si richiede lato client la presenza del plag-in ACGM. In questo modo, il client è responsabile non soltanto della visualizzazione grafica ma anche di alcune semplici funzioni sulla mappa come lo zoom e l'editing locale.

Il livello di Applicazione è formato da due servizi principali: il servizio per controllo degli accessi (ACS Access Control Service) e il servizio applicazione (AS Application Service). Il primo presenta e implementa le operazioni per fare i controlli sulle regole d'autorizzazione. Il secondo servizio presenta ed implementa la parte logica dell'applicazione ed accede ai dati. Quando un utente invoca un'operazione attraverso un WebClient, l'Application Service interagisce con l'Access Control Service per verificare se l'operazione può essere effettuata, qualora il controllo abbia esito positivo l'operazione viene eseguita.

Il livello di Archiviazione Dati è formato da base di dati. In questo livello vengono raggruppati sia l'insieme di database spaziali, sia la base di dati GRBAC che memorizza le informazioni sugli utenti e le regole d'autorizzazione. Tale database sarà descritto nel paragrafo successivo mentre i database spaziali saranno descritti nel paragrafo che presenta il caso di studio.
Un modo conveniente per organizzare l'insieme delle regole d'autorizzazione è quello di memorizzarle in un database spaziale compatibile con il modello per le feature del database di cui si vuole regolare l'accesso. In questo modo si può mappare direttamente le regole d'autorizzazione con finestra sul database spaziale che contiene le informazioni geografiche. Il prototipo G-RBAC realizzato per il controllo degli accessi lavora infatti su una base di dati di tipo geografico come quella dei dati spaziali che si vuole amministrare. Ciò è dovuto al fatto che alcune delle informazioni memorizzate nel database per il controllo degli accessi, come la window e la sessione, hanno una componente spaziale. L'applicativo si pone come filtro tra l'utente finale e i dati geografici memorizzati nel warehouse. Vediamo infatti come avviene una comunicazione tra un utente e il warehouse geografico attraverso l'applicativo G-RBAC. L'utente si connette a G-RBAC venendo riconosciuto e autenticato. Una volta connesso è libero di attivare e disattivare i ruoli cui è assegnato. L'utente invia all'applicativo le richieste necessarie per eseguire le mansioni che deve svolgere con le risorse memorizzate. Per ogni richiesta ricevuta G-RBAC preleva dalle basi di dati spaziali i dati interessati e li confronta con i permessi associati al ruolo dell'utente. In particolare viene verificato che tra tutti i ruoli attivati dall'utente esista un geopermesso per compiere l'operazione. Se tale geopermesso esiste vengono considerati i dati spaziali contenuti nella finestra associata al geopermesso. I dati, dopo essere dunque filtrati, vengono inviati all'utente.

3.2 LIVELLO MEMORIZZAZIONE DATI

Come è stato preannunciato la base di dati che memorizza le informazioni per la gestione del controllo degli accessi è un database misto, con entità sia di tipo spaziale che di tipo non spaziale. Per compatibilità con le tecnologie utilizzate (Intergraph GeoMedia e Geomedia WebMap) è stato utilizzato un database in formato Microsoft Access. A partire dal modello delle classi proposto nel capitolo precedente sono state individuate le seguenti entità:

USER: nel prototipo realizzato un utente è un essere umano che si connette al sistema attraverso la rete Internet. Sono memorizzate per questa entità solo alcuni attributi significativi. In uno sviluppo reale bisognerebbe sicuramente includere altre informazioni aggiuntive. Gli attributi utilizzati per la descrizione dell'entità all'interno della base dati GRBAC sono:

Id	Di tipo contatore, è la chiave primaria che identifica univocamente il record.
Name1	E' una stringa di testo che descrive il nome dell'utente.
Name2	E' una stringa di testo che descrive il cognome dell'utente.
Username	E' una stringa di testo che rappresenta il nome univoco di ogni utente da inserire al momento della connessione.
Password	E' una stringa di testo che rappresenta il codice segreto per autorizzare la connessione.

WINDOW (spatial): l'entità window descrive la finestra d'autorizzazione. E' una feature di tipo area ed è per questo che viene considerata come un'entità spaziale. Una finestra d'autorizzazione è una regione delimitata della superficie terrestre. Permette, abbinata ad un permesso, di stabilire dove la regola d'autorizzazione è valida. Consente inoltre, abbinata ad un ruolo, di specificare dove si deve trovare l'utente che lo vuole attivare. Gli attributi utilizzati per la descrizione dell'entità all'interno della base dati GRBAC sono:

Id	Di tipo contatore, è la chiave primaria che identifica univocamente il record.
Name	E' una stringa di testo che descrive la finestra d'autorizzazione. Questo campo permette all'utente dell'applicativo di riconoscere facilmente le varie Window.
Geometry Data	E' un insieme di attributi di tipo BLOB in formato proprietario di Intergraph® che descrivono e posizionano sulla superficie terrestre una geometria di tipo area.

ROLE: un ruolo è un'entità molto semplice ma ricopre un ruolo fodamentale nell'applicativo realizzato. Esso descrive una mansione che può ricoprire uno o più utenti e permette di compiere una o più operazioni. Gli attributi utilizzati per la descrizione dell'entità all'interno della base dati GRBAC sono:

Id	Di tipo contatore, è la chiave primaria che identifica univocamente il record.
Name	E' una stringa di testo che descrive il ruolo.
Window	Di tipo numerico, è una chiave esterna. In questo campo viene memorizzato l'Id della finestra d'autorizzazione associata al ruolo, se è di tipo dinamico. Se il ruolo non è dinamico il valore di questo campo è 0.

SESSION (spatial): la sessione permette di tener traccia degli utenti connessi. Tale entità memorizza tutte le informazioni relative agli utenti collegati e al tipo di connessione effettuata. In particolare nella sessione viene memorizzata anche la posizione dell'utente (qualora sia possibile identificarla). Nel prototipo realizzato la posizione degli utente viene descritta come un punto, ed è per questo, che la sessione è un'entità spaziale. Gli attributi utilizzati per la descrizione dell'entità all'interno della base dati GRBAC sono:

Id	Di tipo contatore, è la chiave primaria che identifica univocamente il record.
User	Di tipo numerico, è una chiave esterna. In questo campo viene memorizzato l'Id dell'utente che ha avviato la sessione.
Date	Di tipo data, in questo campo viene memorizzata la data e l'ora in cui l'utente ha effettuato la connessione.
SessionId	Di tipo testo, questo campo memorizza il codice univoco associato alla variabile session dell'utente creata automaticamente al momento in cui carica la prima pagina dell'applicativo. Questo campo è molto importante perché ci permette di verificare, alla richiesta di ogni pagina dell'applicativo, se l'utente è autorizzato ad eseguire l'operazione.
Geometry Data	E' un insieme di attributi di tipo BLOB in formato proprietario di Intergraph® che descrivono e posizionano sulla superficie terrestre una geometria di tipo punto.

FUNCTION: si intende per funzione un qualsiasi servizio eseguibile lato server attraverso il Web Map service. Nella versione base viene inclusa una pagina ASP per ogni funzione base eseguibile da richiamare per eseguire la funzione. L'amministratore può creare nuove funzioni ad hoc per il warehouse che sta gestendo, aggiungendole a questa tabella. Gli attributi utilizzati per la descrizione dell'entità all'interno della base dati GRBAC sono:

Id	Di tipo contatore, è la chiave primaria che identifica univocamente il record.
Name	E' una stringa di testo che descrive la funzione.
Link	E' il collegamento ad una pagina che implementa la funzione, presentata automaticamente come link nel menu' degli utenti.

OPERATION: un'operazione è un insieme di funzioni. Specifica per ogni permesso l'insieme di funzioni del Web Service eseguibili. Gli attributi utilizzati per la descrizione dell'entità all'interno della base dati GRBAC sono:

Id	Di tipo contatore, è la chiave primaria che identifica univocamente il record.
Name	E' una stringa di testo che descrive l'insieme operazione.

WAREHOUSES: questa entità non è presente nel modello proposto nel capitolo precedente. E' stata introdotta per rendere automatiche le operazioni di richiesta dello strumento GIS. Specifica per ogni warehouse del sistema dove è memorizzato. Gli attributi utilizzati per la descrizione dell'entità all'interno della base dati GRBAC sono:

Alias	Di tipo stringa, è la chiave primaria che identifica univocamente il database.
Link	E' una stringa di testo che descrive la posizione fisica del warehouse.

FEATURE: una feature descrive una singola Feature Class contenuta in uno qualsiasi dei database spaziali che costituiscono il sistema. Vengono memorizzate informazioni per permettere di recuperare i dati e per la visualizzazione delle feature. Gli attributi utilizzati per la descrizione dell'entità all'interno della base dati GRBAC sono:

Id	Di tipo contatore, è la chiave primaria che identifica univocamente il record.
Warehouse	Di tipo stringa, è una chiave esterna. In questo campo viene memorizzato l'Alias del warehouse in cui è effettivamente memorizzata la feature.
FeatureName	Di tipo stringa. In questo campo viene memorizzato il nome con

	cui è memorizzata all'interno del warehouse la feature. Ciò permette di accedere alla tabella che contiene i dati della feature.
FeatureDescription	Di tipo stringa. Questo campo contiene la descrizione della feature così come è memorizzata all'interno del warehouse nella tabella GFeatures.
GeometryType	Di tipo numerico. Questo campo descrive il tipo di feature (Area, Punto, Retta) così come è memorizzata all'interno del warehouse nella tabella GFeatures.
COLOR_R	Di tipo numerico, serve in fase di visualizzazione della feature. Può assumere un valore compresi tra 0 e 255. Rappresenta la componente R dell'attributo di layout COLOR che esprime il colore della feature.
COLOR_G	Di tipo numerico, serve in fase di visualizzazione della feature. Può assumere un valore compresi tra 0 e 255. Rappresenta la componente G dell'attributo di layout COLOR che esprime il colore della feature.
COLOR_B	Di tipo numerico, serve in fase di visualizzazione della feature. Può assumere un valore compresi tra 0 e 255. Rappresenta la componente B dell'attributo di layout COLOR che esprime il colore della feature.
WEIGHT	Di tipo numerico, serve in fase di visualizzazione della feature. Rappresenta l'attributo di layout WEIGHT che esprime lo spessore della feature.
STYLE	Di tipo numerico, serve in fase di visualizzazione della feature. Può assumere un valore compresi tra 0 e 6. Rappresenta l'attributo di layout STYLE che esprime lo stile della feature.
FILLCOLOR_R	Di tipo numerico, serve in fase di visualizzazione della feature. Può assumere un valore compresi tra 0 e 255. Rappresenta la componente R dell'attributo di layout FILLCOLOR che esprime il colore di riempimento della feature(di tipo area).
FILLCOLOR_G	Di tipo numerico, serve in fase di visualizzazione della feature. Può assumere un valore compresi tra 0 e 255. Rappresenta la componente G dell'attributo di layout FILLCOLOR che esprime il colore di riempimento della feature(di tipo area).
FILLCOLOR_B	Di tipo numerico, serve in fase di visualizzazione della feature. Può assumere un valore compresi tra 0 e 255. Rappresenta la componente B dell'attributo di layout FILLCOLOR che esprime il colore di riempimento della feature(di tipo area).
Priority	Di tipo numerico, serve in fase di visualizzazione della feature. Può assumere un valore compresi tra 0 e 1000. Serve per assegnare una priorità di visualizzazione quando si sovrappongono le varie feature (0 = max).

ToolTips	Di tipo testo, specifica quale attributo della feature memorizzato nel warehouse visualizzare al passaggio del mouse.
Action	Di tipo testo, specifica un link da attivare se l'utente seleziona la feature.

OBJECT: un oggetto è un insieme di Feature. Specifica per ogni permesso l'insieme di Feature distribuite tra i vari warehouse su cui eseguire l'operazione. Gli attributi utilizzati per la descrizione dell'entità all'interno della base dati GRBAC sono:

Id	Di tipo contatore, è la chiave primaria che identifica univocamente il record.
Name	E' una stringa di testo che descrive l'insieme oggetto.

PERMISSION: l'entità permesso descrive un'autorizzazione come una coppia operazione - oggetto. E' in sostanza una regola che permette di eseguire tutte le funzioni dell'insieme operazione su una qualsiasi Feature dell'insieme oggetto. Gli attributi utilizzati per la descrizione dell'entità all'interno della base dati GRBAC sono:

Id	Di tipo contatore, è la chiave primaria che identifica univocamente il record.
Operation	Di tipo numerico, è una chiave esterna. In questo campo viene memorizzato l'Id dell'operazione che raggruppa le funzioni di cui si vuole concedere il permesso.
Object	Di tipo numerico, è una chiave esterna. In questo campo viene memorizzato l'Id dell'oggetto che raggruppa le feature su cui si vuole concedere il permesso.

GEOPERMISSION: un geopermesso descrive una regola d'autorizzazione stabilendo quale operazione può essere effettuata, su quale oggetto e dove. Questa entità è formata da un permesso e da una window. Gli attributi utilizzati per la descrizione dell'entità all'interno della base dati GRBAC sono:

Id	Di tipo contatore, è la chiave primaria che identifica univocamente il record.
Permission	Di tipo numerico, è una chiave esterna. In questo campo viene memorizzato l'Id del permesso.
Window	Di tipo numerico, è una chiave esterna. In questo campo viene memorizzato l'Id della finestra d'autorizzazione associata al permesso.

PARAMETRICROLE: un ruolo parametrico è una superclasse del ruolo e serve per velocizzare la fase di definizione dei ruoli. Gli attributi utilizzati per la descrizione dell'entità all'interno della base dati GRBAC sono identici a quelli del ruolo.

PARAMETRICPERMISSION: l'entità permesso parametrico è una superclasse del permesso e serve per velocizzare la fase di definizione dei ruoli. Gli attributi utilizzati per la descrizione dell'entità all'interno della base dati GRBAC sono identici a quelli del permesso.

Le relazioni fra le varie entità formulate nel modello delle classi presentato nel capitolo precedente hanno portato alla realizzazione delle seguenti tabelle-relazione:

USERASSIGNMENT: questa relazione descrive per ogni utente quale ruolo può ricoprire. Potendo ogni utente ricoprire più ruoli e ogni ruolo essere ricoperto da più utenti non è conveniente memorizzare un'entità come attributo chiave esterna dell'altra. Gli attributi utilizzati per la descrizione della relazione all'interno della base dati GRBAC sono:

User	Di tipo numerico, è una chiave esterna e forma la chiave primaria. In questo campo viene memorizzato l'Id dell'utente che si vuole autorizzare a ricoprire il ruolo.
Role	Di tipo numerico, è una chiave esterna e forma la chiave primaria. In questo campo viene memorizzato l'Id del ruolo per cui si vuole autorizzare l'utente.

SESSIONROLE: questa relazione descrive per ogni sessione di un utente quali sono i ruoli selezionati e quali quelli attivati. Potendo ogni utente tramite una sessione selezionare/attivare più ruoli e ogni ruolo essere selezionato/attivato da più utenti contemporaneamente, non è conveniente memorizzare un'entità come attributo chiave esterna dell'altra. Un utente può attivare più sessioni contemporaneamente e dunque bisogna tener traccia dei ruoli attivati separatamente in ogni sessione. Altrimenti, se l'attivazione di un ruolo in una sessione comportasse l'attivazione dello stesso ruolo in tutte le altre sessioni aperte, sarebbe inutile concedere la possibilità di aprire più sessioni contemporaneamente, dato che sarebbero tutte uguali. Gli attributi utilizzati per la descrizione della relazione all'interno della base dati GRBAC sono:

User	Di tipo numerico, è una chiave esterna. In questo campo viene memorizzato l'Id dell'utente che ha attivato il ruolo.
Role	Di tipo numerico, è una chiave esterna e forma la chiave primaria. In questo campo viene memorizzato l'Id del ruolo attivato dall'utente.
SessionId	Di tipo numerico, è una chiave esterna e forma la chiave primaria. In questo campo viene memorizzata la SessionId della sessione attraverso la quale l'utente ha attivato il ruolo.
Activable	Di tipo VERO/FALSO, questo attributo esplicita se il ruolo è attivabile, cioè se l'utente si trova nella finestra d'autorizzazione del ruolo dinamico. Ogni volta che l'utente cambia posizione questo campo viene aggiornato. Per i ruoli non dinamici il campo ha sempre il valore VERO.

FEATUREASSIGNMENT: questa relazione descrive per ogni oggetto quali sono le feature appartenenti all'insieme che rappresenta. Potendo ogni oggetto raggruppare più feature e ogni feature appartenere a più oggetti non è conveniente memorizzare un'entità come attributo chiave esterna dell'altra. Gli attributi utilizzati per la descrizione della relazione all'interno della base dati GRBAC sono:

Feature	Di tipo numerico, è una chiave esterna e forma la chiave primaria. In questo campo viene memorizzato l'Id della feature di cui si vuole dire a che oggetto appartiene.
Object	Di tipo numerico, è una chiave esterna e forma la chiave primaria. In questo campo viene memorizzato l'Id dell'oggetto a cui appartiene la feature.

FUNCTIONASSIGNMENT: questa relazione descrive per ogni operazione quali sono le fuzioni appartenenti all'insieme che rappresenta. Potendo ogni operazione raggruppare più funzioni e ogni funzione appartenere a più operazioni non è conveniente memorizzare un'entità come attributo chiave esterna dell'altra. Gli attributi utilizzati per la descrizione della relazione all'interno della base dati GRBAC sono:

Function	Di tipo numerico, è una chiave esterna e forma la chiave primaria. In questo campo viene memorizzato l'Id della funzione di cui si vuole dire a che operazione appartiene.
Operation	Di tipo numerico, è una chiave esterna e forma la chiave primaria. In questo campo viene memorizzato l'Id dell'operazione a cui appartiene la funzione.

PERMISSIONASSIGNMEMT: questa relazione descrive per ogni ruolo quali sono i geopermessi che gli sono stati assegnati. Potendo ogni ruolo esercitare più geopermessi e ogni geopermesso essere concesso a più ruoli non è conveniente memorizzare un'entità come attributo chiave esterna dell'altra. Gli attributi utilizzati per la descrizione della relazione all'interno della base dati GRBAC sono:

Role	Di tipo numerico, è una chiave esterna e forma la chiave primaria. In questo campo viene memorizzato l'Id del ruolo a cui è assegnato il geopermesso.
GeoPermission	Di tipo numerico, è una chiave esterna e forma la chiave primaria. In questo campo viene memorizzato l'Id del geopermesso cui è assegnato il ruolo.

PARAMETRICPERMISSIONASSIGNMEMT: questa relazione descrive per ogni ruolo parametrico quali sono i permessi parametrici che gli sono stati assegnati. Potendo ogni ruolo esercitare più permessi e ogni permesso essere concesso a più ruoli non è conveniente memorizzare un'entità come attributo chiave esterna dell'altra. Gli attributi utilizzati per la descrizione della relazione all'interno della base dati GRBAC sono:

ParametricRole	Di tipo numerico, è una chiave esterna e forma la chiave primaria. In questo campo viene memorizzato l'Id del ruolo parametrico a cui è assegnato il permesso parametrico.
ParametricPermission	Di tipo numerico, è una chiave esterna e forma la chiave primaria. In questo campo viene memorizzato l'Id del permesso parametrico cui è assegnato il ruolo.

PERMISSIONHIERARCHY: Questa relazione descrive la nozione di implicazione introdotta con il concetto di gerarchia dei permessi. Potendo ogni permesso implicare ed essere implicato da più permessi non è conveniente memorizzare un'entità come attributo chiave esterna dell'altra. Gli attributi utilizzati per la descrizione della relazione all'interno della base dati GRBAC sono:

PermissionSenior	Di tipo numerico, è una chiave esterna e forma la chiave primaria. In questo campo viene memorizzato l'Id del permesso implicante.
PermissionJunior	Di tipo numerico, è una chiave esterna e forma la chiave primaria. In questo campo viene memorizzato l'Id del permesso implicato.

Per riassumere lo schema della base di dati memorizzata nel file GRBAC.mdb utilizzata per la gestione del controllo degli accessi è quello illustrato nella seguente figura:

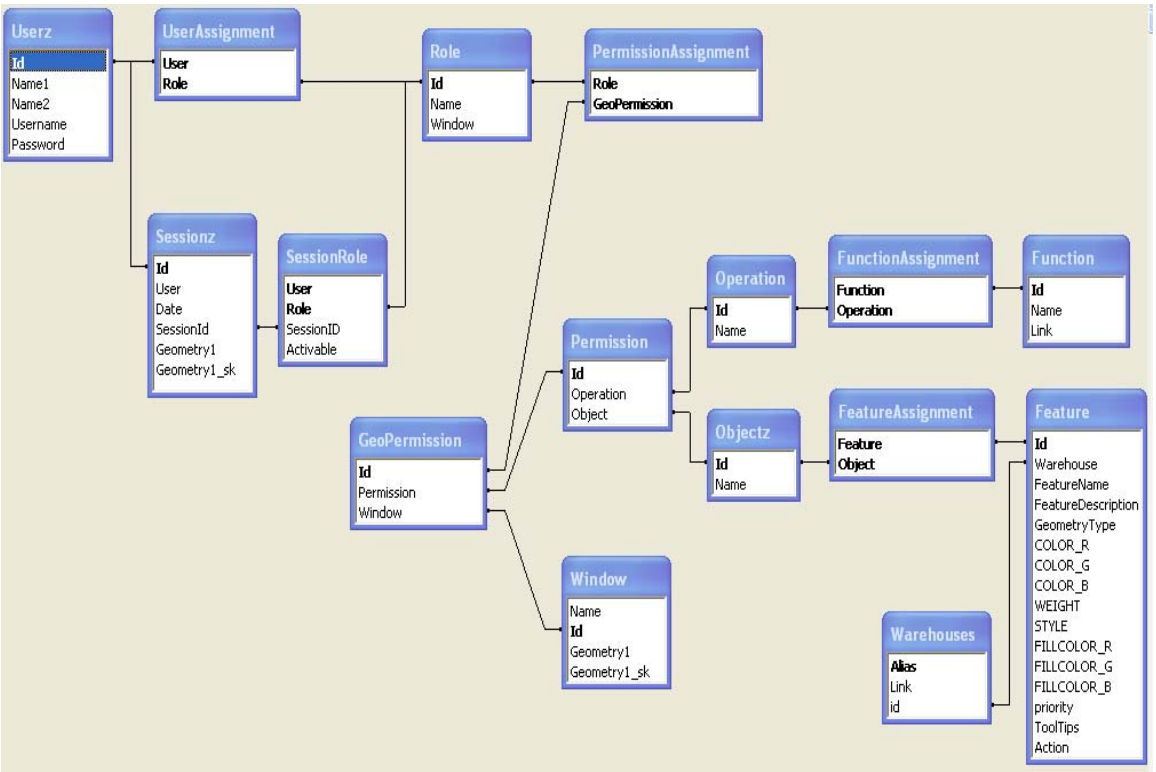


Figura 2: Schema per il Database GRBAC

3.3 LIVELLO APPLICAZIONE

A partire dalle specifiche funzionali presentate per il modello G-RBAC nel capitolo precedente, sono state implementate nell'applicativo proposto alcune procedure per la gestione del database GRBAC per il controllo degli accessi. Le principali vengono presentate qui di seguito:

3.3.1 Gestione Utenti

Add User (Nome, Cognome, NomeUtente, ParoladOrdine)
In tabella USER aggiungi record:
Id = AUTO-NUMBER
Name1 = Nome
Name2 = Cognome
Username = NomeUtente
Password = ParoladOrdine

Modify User (Id-utente, Nome, Cognome, NomeUtente, ParoladOrdine)
In tabella USER modifica record con Id = Id-utente:
Name1 = Nome
Name2 = Cognome
Username = NomeUtente
Password = ParoladOrdine

Assign User (Id-utente, Id-ruolo)
In tabella USERASSIGNMENT aggiungi record:
User = Id-utente
Role = Id-ruolo

Deassign User (Id-utente, Id-ruolo)
In tabella USERASSIGNMENT elimina record con User=Id-utente e Role=Id-ruolo.
Del User (Id-utente)
In tabella USER elimina record con Id = Id-utente
In tabella USERASSIGNMENT elimina record con User = Id-utente

3.3.2 Gestione Autorizzazioni

Add Role (Nome)
In tabella ROLE aggiungi record:
Id = AUTO-NUMBER
Name = Nome
Window = 0

Add Geo Permmission (Id-ruolo, Id-oggetto, Id-operazione, Id-window)
In tabella PERMISSION aggiungi, se non esiste già, il record:
Id = AUTO-NUMBER
Object = Id-oggetto
Operation = Id-operazione
In tabella GEOPERMISSION aggiungi, se non esiste già, il record:
Permission = PERMISSION.Id
Window = Id-window
In tabella PERMISSIONASSIGNMENT aggiungi il record:
Id = AUTO-NUMBER
Role = Iid-ruolo
GeoPermission = GEOPERMISSION.Id

Del Geo Permmission (Id-ruolo, Id-geopermesso)
In tabella PERMISSIONASSIGNMENT elimina record con Geopermission=Id-geopermesso e Role=Id-ruolo.

Del Role (Id-ruolo)
In tabella USERASSIGNMENT elimina record con Role = Id-ruolo
In tabella PERMISSIONASSIGNMENT elimina record con Role = Id-ruolo
In tabella ROLE elimina record con Id = Id-ruolo

Set Role Dynamic (Id-ruolo, Id-window)
In tabella ROLE modifica record con Id = Id-ruolo:
Window = Id-window

Back Role Static (Id-ruolo)
In tabella ROLE modifica record con Id = Id-ruolo:
Window = 0

Add ParametricRole (Nome)
In tabella PARAMETRICROLE aggiungi record:
Id = AUTO-NUMBER
Name = Nome

Add Parametric Permmission (Id-ruolo, Id-oggetto, Id-operazione)
In tabella PARAMETRICPERMISSION aggiungi, se non esiste già, il record:
Id = AUTO-NUMBER
Object = Id-oggetto
Operation = Id-operazione
In tabella PARAMETRICPERMISSIONASSIGNMENT aggiungi il record:
Id = AUTO-NUMBER
Parametric Role = Iid-ruolo
Parametric GeoPermission = GEOPERMISSION.Id

Del Parametric Permmission (Id-ruolo, Id-permesso)
In tabella PARAMETRICPERMISSIONASSIGNMENT elimina record con Parametricpermission=Id-permesso e ParametricRole=Id-ruolo.

Del Parametric Role (Id-ruolo)
In tabella PARAMETRICPERMISSIONASSIGNMENT elimina record con ParametricRole = Id-ruolo
In tabella PARAMETRICROLE elimina record con Id = Id-ruolo

Generate Role (Id-ruoloParametrico, Id-window)
In tabella ROLE aggiungi record:
Id = AUTO-NUMBER
Name = ruoloParametrico.Nome + window.Name
Window = 0
Per ogni permesso parametrico associato a Id-ruoloParametrico PP(Id-ruoloParametrico):
In tabella GEOPERMISSION aggiungi, se non esiste già, il record:
Permission = PP(Id-ruoloParametrico)
Window = Id-window
In tabella PERMISSIONASSIGNMENT aggiungi il record:
Id = AUTO-NUMBER
Role = ROLE.id
GeoPermission = GEOPERMISSION.Id

3.3.3 Gestione Operazioni

Add Operation (nome)
In tabella OPERATION aggiungi record:
Id = AUTO-NUMBER
Name = Nome

Assign Function (Id-operazione, Id-funzione)
In tabella FUNCTIONASSIGNMENT aggiungi il record:
Operation = Id-operazione
Function = Id-funzione

Del Operation (Id-operazione)
In tabella PERMISSIONASSIGNMENT elimina record con Geopermission con Permission con Operation = Id-operazione
In tabella GEOPERMISSION elimina record con Permission con Operation = Id-operazione
In tabella PERMISSION elimina record con Operation = Id-operazione
In tabella FUNCTIONASSIGNMENT elimina record con Operation = Id-operazione
In tabella OPERATION elimina record con Id = Id-operazione

3.3.4 Gestione Oggetti

Add object (nome)
In tabella OBJECT aggiungi record:
Id = AUTO-NUMBER
Name = Nome

Assign Feature (Id-oggetto, Id-feature)
In tabella FEATUREASSIGNMENT aggiungi il record:
Object = Id-oggetto
Feature = Id-feature

Del object (Id-oggetto)
In tabella PERMISSIONASSIGNMENT elimina record con Geopermission con Permission con Object = Id-oggetto
In tabella GEOPERMISSION elimina record con Permission con Object = Id-oggetto
In tabella PERMISSION elimina record con Object = Id-oggetto
In tabella FEATUREASSIGNMENT elimina record con Object = Id-oggetto
In tabella OBJECT elimina record con Id = Id-oggetto

3.4 LIVELLO PRESENTAZIONE

Come è stato detto l'utente finale necessita di un browser per poter accedere al sistema attraverso l'applicativo che gestisce il controllo degli accessi. Il Web Service Geografico utilizzato, GeoMedia WebMap Professional, elabora mappe in file ActiveCGM (ACGM), che è un formato proprietario. Questi file ActiveCGM possono essere visualizzati in browser come Internet Explorer o Netscape. Per essere visualizzati deve essere istallato un *ActiveCGM control*. Per Internet Explorer, si tratta di un componente ActiveX, per Netscape invece è un PlugIn. L'utente può interagire con la mappa attraverso i comandi locali del ActiveCGM control, attraverso l'esecuzione di script lato client, o richiamando processi WebMap addizionali dal server.

3.5 SCELTE PROGETTUALI e ASPETTI IMPLEMENTATIVI

In questo paragrafo vengono evidenziati e alcune scelte progettuali effettuate e alcuni aspetti implementativi rilevanti della fase di realizzazione del prototipo.

3.5.1 Gestione della finestra d'autorizzazione

La finestra d'autorizzazione da associare ad un permesso per ottenere un geopermesso è semplicemente una Feature di tipo AREA. La gestione di tale Feature è molto importante e sono state individuate tre possibilità inerenti la sua collocazione.

Si potrebbe memorizzare la window all'interno di uno dei Warehouse del sistema informativo. Utilizzando questa soluzione è possibile adoperare come window una Feature già esistente di un Warehouse rendendo la fase di inizializzazione più semplice. La seconda soluzione possibile è quella di creare un Warehouse indipendente dedicato unicamente alla Window. Se questa è la soluzione più lineare e robusta ci sono degli aspetti negativi rilevanti: traffico elevato, difficile integrazione con i Warehouses del sistema informativo. La terza soluzione prevede il collocamento della tabella nel database G-RBAC in modo da ottenere un accesso più rapido durante la fase di autenticazione e d'amministrazione poiché si adopera la stessa connessione utilizzata per accedere agli altri dati necessari. Inoltre a livello logico, la Window contiene informazioni inerenti il controllo degli accessi ed è quindi auspicabile che sia memorizzata insieme agli altri dati per la gestione degli accessi.

Questa sembra essere la soluzione migliore ed è anche quella effettivamente utilizzata nell'implementazione sebbene l'inserimento di Feature via Web non abbia la precisione di tutti quei tool (Intersection Snap, End Point Snap, Vertex Snap) che facilitano il disegno di Feature contigue. E' possibile tuttavia utilizzare programmi ad-hoc (come può essere Intergraph® GeoMedia) per la creazione e gestione delle window connettendosi contemporaneamente al database dei dati e a quello per il controllo degli accessi GRBAC.

3.5.2 Autenticazione e autorizzazioni

Per il riconoscimento degli utenti e la gestione delle concessioni è necessario memorizzare un'informazione che permetta di verificare, per ogni pagina richiesta, se si è autorizzati a visualizzarla. E' necessario memorizzare delle informazioni circa gli utenti connessi che rimangano valide non solo per la durata di visualizzazione di una singola pagina, ma per l'intera sessione di lavoro dell'utente stesso. A tale scopo, per la gestione di variabili valide per un'intera sessione di lavoro, si possono utilizzare tre oggetti messi a disposizione dall'architettura standard di rete:

L'oggetto Session Come fa intuire il nome stesso, la variabile definita come session vale per la sessione di navigazione attiva. Infatti ogni utente connesso, potrà leggere e variare solamente le proprie variabili di sessioni. L'uso di una variabile di sessione è abbastanza simile a quello di una variabile normale. Ogni variabile di sessione, appena creata viene a generare un numero univoco di identificazione da parte del Server. Questo numero è progressivo e serve appunto al Server per riconoscere il valore della sessione ogni volta che la utilizziamo. Bisogna ricordare che le variabili session si appoggiano ai cookies. Nel caso l'utente finale avesse i cookies disabilitati, il funzionamento di questa tipologia di variabili è garantita in quanto il server attiva un cookies speciale che ne consente la gestione.

L'oggetto Application Il problema principale utilizzando le sessioni è la durata nel tempo. Infatti, la sua vita per default è impostata a venti minuti. Inoltre non è possibile condividere questa variabile tra i diversi utenti connessi. Se vogliamo condividere una variabile tra tutti gli utenti collegati, basta utilizzare le application. Questa variabile, a differenza delle altre precedentemente illustrate, ha una vita a noi non prevedibile in quantità di tempo. Infatti questa vive sino a quando abbiamo un utente collegato al nostro sito web. Quando il nostro web resta deserto (si intende senza visitatori) la nostra application muore.

L'oggetto cookies Per mantenere un dato vivo (cioè attivo in memoria) per un lungo periodo di tempo bisogna utilizzare i cookies. Analizzando la visibilità dei cookies nel rapporto utente/tempo si può dire che tale visibilità è del singolo utente e la durata nel tempo è variabile. Questa durata dipende da una impostazione specifica che si impartisce durante la creazione fisica del cookies. Il cookies viene creato sul SS (Server Side - Lato Server) in base a delle direttive date dalla pagina asp programmata e viene inviato (tramite comandi appositi) sotto forma di file di testo al CS (Client Side - Lato Utente). E' appunto per questo motivo che il cookies ha una vita prolungata nel tempo. Ciò è possibile grazie al fatto che il cookies risiede NON sul server, ma sul computer del visitatore. E' sconsigliabile tuttavia memorizzare informazioni per il controllo degli accessi lato Client.

Necessitando di variabili che durino solo per la durata della sessione di lavoro viene utilizzato l'oggetto Session per la gestione degli accessi al prototipo. Quando un utente effettua il log-in con successo viene memorizzata nel database G-RBAC la sua SessionID. Tale variabile viene associata ovviamente all'utente che l'ha generata e ad ogni ruolo che attiva. In tale modo, per ogni pagina richiesta confrontando la SessionID del richiedente con quelle memorizzate nel database G-RBAC è possibile stabilire se autorizzare oppure no l'esecuzione della pagina.

Per quel che riguarda la fase di autenticazione, sono previsti tre stati per l'utente: *NotLogged, LoggedNoRole* e *LoggedWithRole* (Figura 3). In seguito al log-in l'utente passa dallo stato *NotLogged* allo stato *LoggedNoRole*. Nello stato *LoggedNoRole* può attivare uno o più ruoli e passare dunque allo stato *LoggedWithRole*. Effettuando il log-out torna nello stato *NotLogged.*

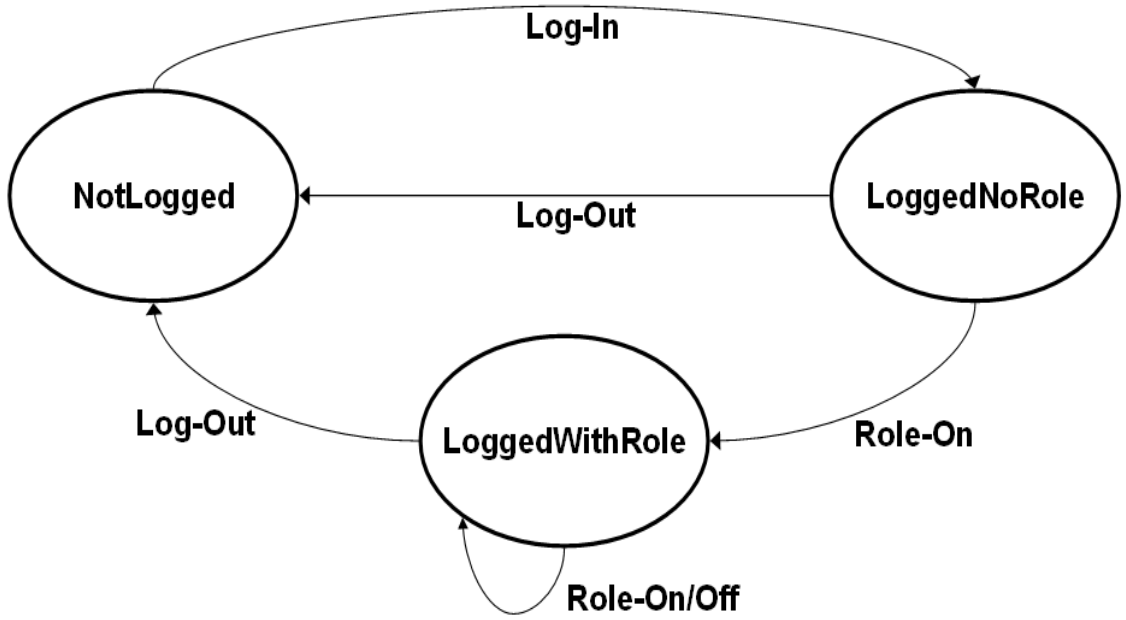


Figura 3: Stati utente

3.5.3 Localizzazione dell'utente

Come è stato detto la posizione dell'utente può essere memorizzata in diversi modi. A seconda del tipo di tecnologia a disposizione si possono utilizzare differenti tipi di geometria per rappresentare la locazione dell'utente. Nel prototipo realizzato la posizione dell'utente viene simulata lato Client. L'utente deve manualmente impostare la propria posizione sulla Basemap. Tale informazione viene memorizzata nel database GRBAC con una geometria di tipo punto.

3.5.4 Basemap

L'ampiezza della mappa che viene caricata in visualizzazione dipende dalla feature su cui si imposta il range e dalla collocazione degli elementi di tale feature. Per rendere l'applicativo più omogeneo e le mappe prodotte più comprensive, è stato deciso di includere in ogni mappa riprodotta una feature di sfondo. Questa feature prende il nome di Basemap.La feature Basemap indipendentemente dai permessi posseduti dall'utente e dalle funzioni che esso deve svolgere viene caricata in ogni mappa visualizzata. L'amministratore può scegliere quale feature, tra tutte quelle presenti nel sistema, utilizzare come Basemap. E' ovvio che bisognerebbe utilizzare una feature semanticamente interessante, in grado cioè di facilitare la comprensione della collocazione delle altre features caricate di volta in volta.

3.6 CASO DI STUDIO

3.6.1 Introduzione

Il prototipo deve essere implementato su di un caso di studio adatto ad enfatizzare le peculiarità del modello proposto. E' stato allora sviluppato un esempio in cui siano evidenziati i vantaggi introdotti dei geo-permessi e l'importanza dei ruoli statici e dei ruoli dinamici in un applicativo che regoli il controllo degli accessi in un database spaziale. La realtà ipotizzata è quella di un'impresa che integra nell'offerta per i propri clienti, turisti della città di Milano, la possibilità di accedere ad informazioni via web durante il periodo della visita. I servizi proposti da tale società potrebbero svariare dalle esigenze più semplici di trasporto fino alle visite guidate alla scoperta della metropoli. Un elenco di questi servizi potrebbe essere:

- Trasporto privato del cliente (taxi privato con terminale mobile per autista e cliente).
- Trasporto collettivo di clienti (mini bus con terminale mobile per autista e clienti).
- Visita culturale della Città.
- Visita commerciale della Città.
- Visita guidata a musei, edifici culturali, ecc.
- Servizio guida-automatica con fornitura di palmare al cliente.

Per realizzare un applicativo di questo tipo sono necessarie informazioni sulla rete stradale di Milano e sui punti d'interesse della città. A tal proposto è stato utilizzata la banca dati NAVTEQ STREET ITALIA. Tutte le Feature Class del Warehouse NavTeq sono state filtrate con un intersezione spaziale sul solo comune di Milano ed esportate in un nuovo warehouse . Per compatibilità con le tecnologie utilizzate (Intergraph GeoMedia e Geomedia WebMap) è stato utilizzato un database in formato Microsoft Access. Questo database più compatto (MilanTourist.mdb) è il warehouse contenente le informazioni geografiche di cui il prototipo deve amministrare il controllo degli accessi. Gli utenti otterranno le informazioni di questo database spaziale in base alla propria locazione, al ruolo ricoperto e ai geopermessi abbinati a tale ruolo.

3.6.2 Descrizione del database geografico

Le informazioni sono contenute, come è stato appena detto, nel database MilanTourist.mdb . Le Feature Class del warehouse sono descritte di seguito:

Major Highways (linea)

Contiene gli archi che compongono le strade ad alto scorrimento, di solito ad accesso controllato, con grande volume di traffico e di spostamenti. Di solito è utilizzata per canalizzare il traffico da e a strade di tipo Secondary Highways.

Secondary Highways (linea)
Contiene gli archi che compongono le strade a medio scorrimento, con grande volume di traffico ma bassa velocità di spostamento. Di solito collegano i vari quartieri della città.

Streets (linea)
Contiene la totalità degli archi che compongono l'intera rete stradale. Memorizza anche informazioni aggiuntive come la tipologia e il nome della strada, la numerazione delle abitazioni e tutte le informazioni aggiuntive utili per la navigazione.

Railroads (linea - poligono)
Contiene la totalità degli archi e dei poligoni che descrivono l'intera rete ferroviaria. Memorizza anche informazioni aggiuntive come la tipologia e il nome dell'elemento ferroviario.

Waterway Polygons (poligono), ***Waterway Segments*** (linea)
Contengono poligoni e linee che descrivono elementi d'acqua sulla superficie come possono essere fiumi, canali, cave, laghi artificiali e naturali.

Administrative Area Boundaries (1, 2, 3, 4, 5) (poligono)
Sono 5 Feature Class di tipo poligonale che rappresentano la divisione amministrativa dell'area. Per l'Italia la suddivisione è così rappresentata:

1. Nazione
2. Regioni
3. Province
4. Comuni
5. Frazioni

Ovviamente nel database compatto del solo comune di Milano è presenta la sola Feature Class AdministrativeAreaBoundaries5 delle frazioni.

Land Use Features A (poligono)
Contiene una descrizione di come è utilizzata una determinata area. In particolare gli usi possono essere Aeroporto, Cimitero, Ospedale, Complesso Industriale, Caserma Militare, Parco, Monumento Nazionale, Area ad uso Pubblico, Centro Commerciale, Complesso Sportivo, Università e Area Boschiva.
Nella base dati ridotta al comune di Milano sono presenti 77 oggetti appartenenti a questa Feature Class.

Hospitals (punto)
Localizza elementi di tipo ospedale e pronto soccorso. Contiene inoltre informazioni aggiuntive sulla struttura. Nella base dati ridotta al comune di Milano sono presenti 13 oggetti appartenenti a questa Feature Class.

Parks and Recreation (punto)
Localizza elementi di svago e divertimento. Contiene inoltre informazioni aggiuntive sulla struttura. Gli elementi raccolti sono suddivisi nelle seguenti categorie: Parco Divertimenti, Golf Club, Museo, Complesso Sportivo, Parco Marino, Centro Sportivo Pubblico, Aeroporto, Bowling, Parco Giochi, Casinò, Pattinaggio su Ghiaccio. Nella base dati ridotta al comune di Milano sono presenti 130 oggetti appartenenti a questa Feature Class.

Transportation Hubs (punto)
Localizza i terminali e le stazioni di mezzi di trasporto pubblico e privato. Contiene inoltre informazioni aggiuntive sulle strutture. Gli elementi raccolti sono suddivisi nelle seguenti categorie: Stazione Ferroviaria, Stazione Autobus, Aeroporto, Stazione di Interscambio. Nella base dati ridotta al comune di Milano sono presenti 18 oggetti appartenenti a questa Feature Class.

Travel Destinations (punto)
Localizza punti d'interesse turistico. Contiene inoltre informazioni aggiuntive sulle strutture. Gli elementi raccolti sono suddivisi nelle seguenti categorie: Hotels/Motels, Attrazioni Turistiche, Informazioni Turistiche. Nella base dati ridotta al comune di Milano sono presenti 366 oggetti appartenenti a questa Feature Class.

Shopping (punto)
Localizza punti d'interesse per fare shopping quali possono essere centri commerciali, zone famose ricche di negozi e ristoranti. Contiene inoltre informazioni aggiuntive sulle strutture. Nella base dati ridotta al comune di Milano sono presenti 9 oggetti appartenenti a questa Feature Class.

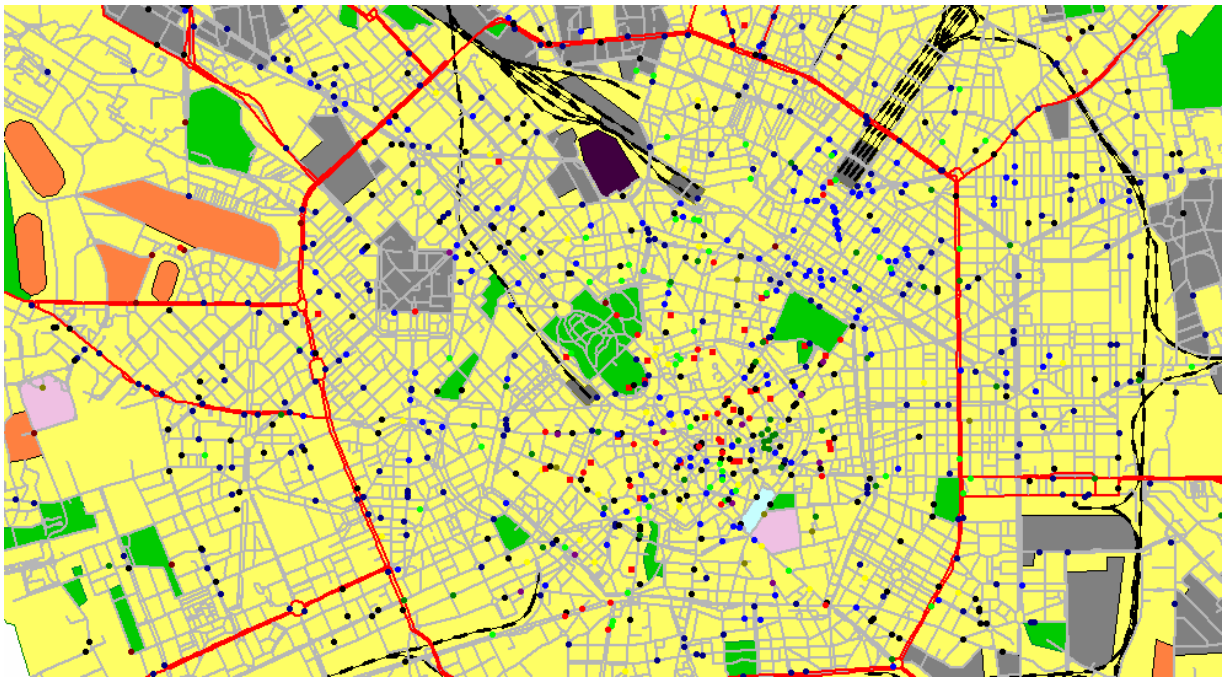
Restaurants (punto)
Localizza i ristoranti. Contiene inoltre informazioni aggiuntive sulle strutture. Nella base dati ridotta al comune di Milano sono presenti 1021 oggetti appartenenti a questa Feature Class.

Entertainment (punto)
Localizza punti d'intrattenimento. Contiene inoltre informazioni aggiuntive sulle strutture. Gli elementi raccolti sono suddivisi nelle seguenti categorie: Vita Notturna, Arte, Cinema e Teatro. Nella base dati ridotta al comune di Milano sono presenti 166 oggetti appartenenti a questa Feature Class.

Auto Maintenance, Service, and Petrol (punto)
Localizza punti d'utilità per gli autonobilisti. Contiene inoltre informazioni aggiuntive sulle strutture. Gli elementi raccolti sono suddivisi nelle seguenti categorie: Distributore Benzina/Gas, Auto officina, Rivenditori Automobili, Club Automobilistici, Rivenditori Motocicli. Nella base dati ridotta al comune di Milano sono presenti 664 oggetti appartenenti a questa Feature Class.

Educational Institutions (punto)
Localizza punti d'educazione. Contiene inoltre informazioni aggiuntive sulle strutture. Gli elementi raccolti sono suddivisi nelle seguenti categorie: Scuole, Biblioteche, Università. Nella base dati ridotta al comune di Milano sono presenti 19 oggetti appartenenti a questa Feature Class.

Parking (punto)
Localizza punti d'utilità per posteggiare. Contiene inoltre informazioni aggiuntive sulle strutture. Gli elementi raccolti sono suddivisi nelle seguenti categorie: Parcheggi, Garage a pagamento, Parcheggi a pagamento. Nella base dati ridotta al comune di Milano sono presenti 485 oggetti appartenenti a questa Feature Class.



Snapshoot di MilanTourist in GeoMedia

3.6.3 Descrizione Ruoli

Di seguito vengono descritti sommariamente quali sono i ruoli possibili per lo scenario introdotto. Vengono elencati inoltre, per ognuno di essi, i permessi associati. Solo alcuni ruoli sono effettivamente stai implementati nel prototipo.

Direttore *(Ruolo Statico)*

Tutti gli addetti della società che possono avere le funzioni amministrative. Ha i seguenti permessi/compiti:

- Gestione sistema
- Amministrazione ruoli-permessi.
- Statistiche
- Visualizzazione report movimenti dipendenti,
- Visualizzazione report movimenti clienti,

Segreteria *(Ruolo Statico)*

Addetti della società che si occupano della parte gestionale di clienti e addetti. Ha i seguenti permessi/compiti:

- gestione anagrafica dei clienti
- inserimento, modifica attributi del cliente,
- localizzazione del cliente durante la connessione.
- gestione addetti
- inserimento, modifica attributi del dipendente,
- localizzazione del dipendente durante la connessione.

Autista Taxi *(Ruolo Dinamico)*

Addetti della società che guidano un taxi con l'ausilio dell'applicativo tramite palmare o wap. Ha i seguenti permessi/compiti:

- Trasporto turisti
- Localizzazione destinazione,
- Visualizzazione Major Highways, Secondary Highways, Streets, Railroads, Parking, Transportation Hubs, Hospitals, Auto Maintenance, Service, and Petrol .
- Visualizzazione percorso, interruzioni e ingorghi,
- Segnalazione interruzioni e ingorghi.

Autista Bus *(Ruolo Dinamico)*

Addetti della società che guidano un bus con l'ausilio dell'applicativo tramite palmare o wap. Ha i seguenti permessi/compiti:

- Trasporto turisti
- Localizzazione destinazione,
- Visualizzazione Major Highways, Secondary Highways, Streets, Railroads, Parking, Transportation Hubs, Hospitals, Auto Maintenance, Service, and Petrol.
- Visualizzazione percorso, strade impraticabili agli autobus, interruzioni e ingorghi,
- Segnalazione interruzioni e ingorghi.

Guida Museo *(Ruolo Statico)*

Addetti della società che guida i turisti all'intero di un museo con l'ausilio dell'applicativo tramite palmare o wap. Ha i seguenti permessi/compiti:

- Guida turisti
- Visualizzazione grafica del museo, mappa, collocazione opere e relativi attributi,
- Visualizzazione informazioni e attributi sul museo,
- Localizzazione del cliente durante la connessione.

Guida Città *(Ruolo Dinamico)*

Addetti della società che guida i turisti per la città con l'ausilio dell'applicativo tramite palmare o wap. Ha i seguenti permessi/compiti:

- Guida turisti
- Visualizzazione grafica della città, mappa, collocazione POI ad uso della guida e relativi attributi: Parks and Recreation, Shopping, Restaurants, Entertainment, Land Use Features, Travel Destinations, Educational Institutions.
- Localizzazione del cliente durante la connessione.

Turista *(Ruolo Dinamico)*

Cliente che accede all'applicativo attraverso browser, wap o palmare. Ha i seguenti permessi:

Informazioni commerciali

- Visualizzazione grafica della città, mappa, collocazione POI di interesse turistico commerciale e relativi attributi: Parks and Recreation, Shopping, Restaurants, Entertainment.
- Localizzazione sedi ed assistenza della società.

Turista AutoGuida *(Ruolo Dinamico)*

Cliente che accede all'applicativo attraverso palmare fornito dalla società. Ha i seguenti permessi:

Informazioni turistico-culturali

- Visualizzazione grafica della città, mappa, collocazione POI di interesse turistico culturale e relativi attributi: Land Use Features, Travel Destinations, Educational Institutions.
- Localizzazione sedi ed assistenza della società.

Informazioni commerciali

- Visualizzazione grafica della città, mappa, collocazione POI di interesse turistico commerciale e relativi attributi: Parks and Recreation, Shopping, Restaurants, Entertainment.

3.7 INTERFACCIA e INTERAZIONE COL SISTEMA

Ogni utente può accedere all'applicativo collegandosi attraverso la rete e inserendo nome utente e password (Figura 4).

Figura 4: Schermata Principale

Se l'autenticazione è corretta viene attivata una nuova sessione per l'utente. Finché la sessione è attiva l'utente può visualizzare l'interfaccia che gli permette di interagire col sistema. Tale interfaccia è divisa i due parti. A sinistra vengono raggruppate le informazioni inerenti la sessione dell'utente ed i comandi per gestire la modalità d'esecuzione. In particolare nella parte superiore compaiono informazioni anagrafiche riguardanti l'utente e il tasto per uscire dall'applicativo (Figura 5-A). Immediatamente sotto, l'utente può impostare la propria posizione attraverso il link *SetLocation* (Figura 5-B) che attiva la procedura, descritta in seguito, per modificare la locazione dell'utente. Questa parte dell'applicativo è destinata a scomparire quando il prototipo verrà raffinato introducendo le tecnologie necessarie al riconoscimento automatico della posizione da cui avviene la connessione. Più in basso (Figura 5-C), è collocata l'interfaccia per la gestione dei ruoli. Sono elencati tutti i ruoli che l'utente può ricoprire e per ognuno di essi sono presenti un tasto per attivarlo (ON) e uno per disattivare (OFF). Un ruolo disattivato risulta scritto in nero e allineato a destra. Un ruolo attivo compare invece in rosso, grassetto e allineato a sinistra. I ruoli solo selezionati invece appaiono anche loro in grassetto e allineati a sinistra ma ancora di colore nero. Si ricordi che per ruoli selezionati si intende quei ruoli dinamici che l'utente ha attivato ma non sono ancora effettivamente validi poiché l'utente non si trova all'interno della finestra d'autorizzazione abbinata al ruolo.

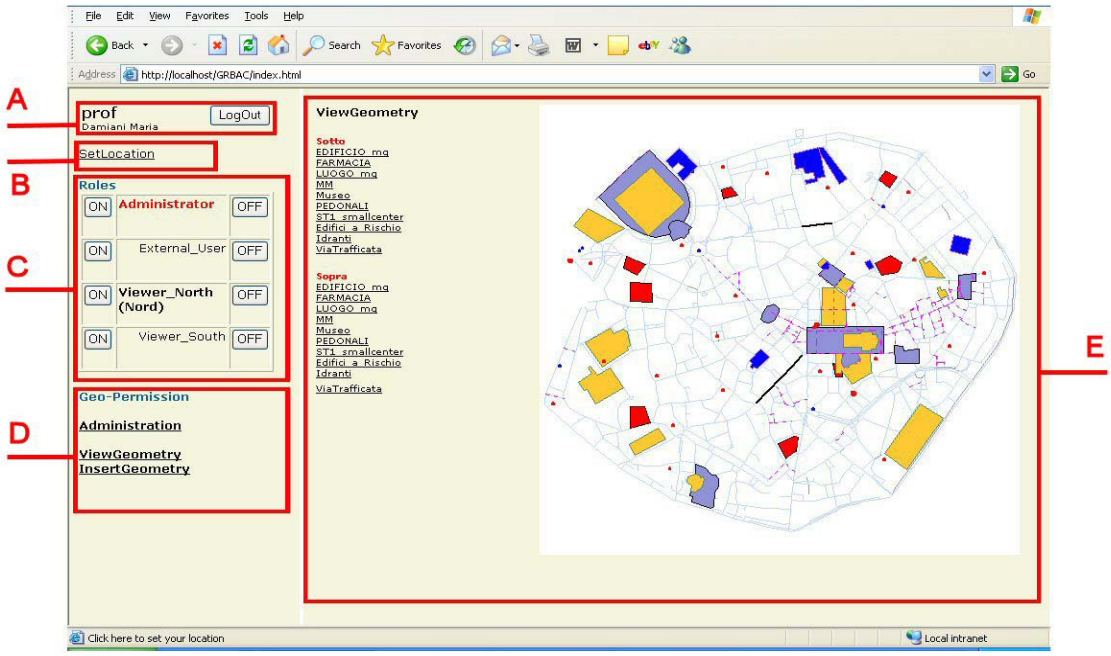


Figura 5: Funzionalità del prototipo

Infine nella parte inferiore compaiono i link alle funzioni che l'utente è autorizzato ad eseguire in base ai ruoli attivi cui è associato (Figura 5-D). Questi link permettono di caricare nella seconda parte dell'interfaccia, quella di destra, le pagine che effettivamente implementano le operazioni. Quella di destra è dunque la parte esecutiva dell'applicativo (Figura 5-E), in cui l'utente svolge effettivamente le mansioni per cui ha effettuato il collegamento al sistema.

L'amministratore ha a disposizione delle funzioni per la gestione del sistema del controllo degli accessi. Tali funzioni sono racchiuse nell'operazione Administration che appartiene di default al prototipo e non può essere modificata. Quello dell'amministratore diventa dunque un ruolo come gli altri. Per eseguire le operazioni di amministrazione, l'amministratore accede al prototipo come un utente normale e attiva il ruolo *Administrator*. L'attivazione di tale ruolo gli permette di accedere alle funzioni richieste (Figura 6-A).

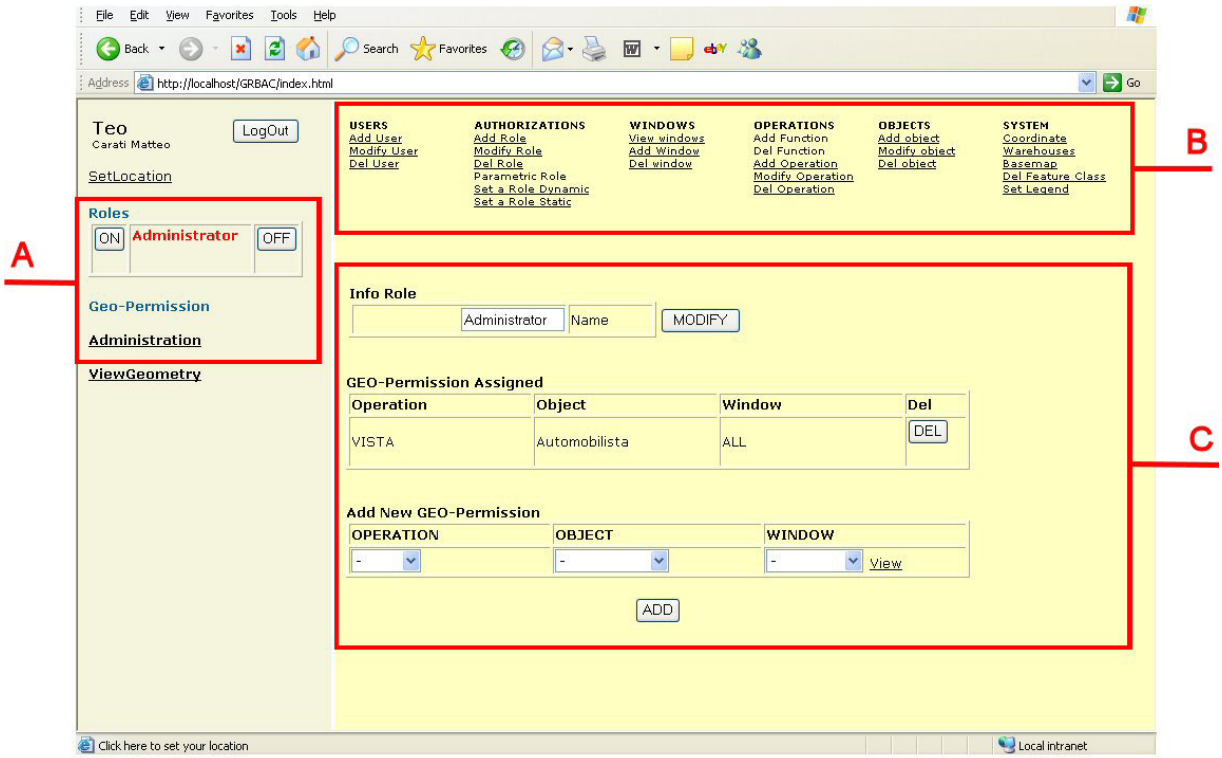


Figura 6: Interfaccia Amministratore

Tutte le funzionalità a disposizione dell'amministratore compaiono nella parte superiore destra (figura 6-B), raggruppate in maniera intuitiva per categoria. Nella parte inferiore di destra invece(figura 6-c), compare l'interfaccia per eseguire le funzionalità a disposizione. Ad esempio (figura 6-c), l'amministratore può modificare un ruolo varandone il nome, eliminando i Geo-permessi ad esso associati o definendo un nuovo Geo-permesso per tale ruolo selezionando l'operazione, l'oggetto e la finestra che lo compongono.

RIFERIMENTI BIBLIOGRAFICI

[1] R. Ahad et al. Supporting Access Control in an Object-Oriented Database Language. In *Proc. Int. Conf. on Extending Database Technology (EDBT)*, Vienna, Austria, Springer-Verlag LNCS 580, 1992.

[2] E. Bertino. Data Security. *Data & Knowledge Engineering*, 25(1-2):199–216, March 1998.

[3] E. Bertino, C. Bettini, E. Ferrari, and P. Samarati. A Temporal Access Control Mechanism for Database Systems. *IEEE Trans. on Knowledge and Data Engineering*, 8(1):67-80, February 1996.

[4] E. Bertino, F. Buccafurri, E. Ferrari, and P. Rullo. A Flexible Authorization Model and its Formal Semantics, In *Proc. of 5th European Symposium on Research in Computer Security*, to appear.

[5] Bertino, Damiani "A Controlled Access to Spatial Data on Web"

[6] Bertino, Damiani, Momini "An access control system for a web map management"

[7] E. Bertino, Elena Ferrari. Data Security. *COMPSAC1998*, 228-239

[8] E.Bertino, P.Samarati, and S.Jajodia. An Extended Authorization Model. *IEEE Trans. on Knowledge and Data Engineering*, 9(1):85-101, January/ February 1997.

[9] E. Bertino and H. Weigand. An approach to Authorization Modeling in Object-Oriented Database Systems. *Data and Knowledge Engineering*, 12(1), 1994.

[10] S. Castano, M.G. Fugini, G. Martella, and P. Samarati. *Database Security*. AddisonWesley, 1995.

[11] S. Cox , P. Daisey, R. Lake, C. Portele, A. Whiteside "OpenGIS Geography Markup Language (GML) Implementation Specification"

[12] Sahadeb De, Caroline Eastman, Csilla Farkas "Secure Access Control in a Multi-user Geodatabase"

[13] R. Fagin. On an Authorization Mechanism. *ACM Trans. on Database Systems*, 3(6):310–319, November 1976.

[14] Fernandez-Medina, De Capitani di Vimercati, Damiani, Piattini, Samarati "Access control system for SVG Documents" in "Multimedia Security and Digital Rights Management Technology"

[15] D.F. Ferraiolo, D.R. Kuhn, R. Chandramouli, "Role Based Access Control"

[16] Frode Hansen, Vladimir Oleshchuk; "SRBAC: A Spatial Role-Based Access Control Model for Mobile Systems"

[17] P. P. Griffiths and B. W. Wade. An Authorization Mechanism for a Relational Database System. *ACM Trans. on Database Systems*, 1(3):242–255, September 1976.

[18] L. Haas, W. Chang, and G.M. Lohman. Starbust Midflight: as the Dust Clears. *IEEE Trans. On Knowledge and Data Engineering*, 2:33–54, 1990.

[19] Jeff de La Beaujardiere "Web Map Service OpenGIS® Implementation"

[20] Tim Moses "eXtensible Access Control Markup Language 2 (XACML)"

[21] Purevjii, Amagasa, Imai, Kanamori “An Access Control Model for Geographic Data in a XML-based Framework”

[22] F. Rabitti, E. Bertino, W. Kim, and D. Woelk. A Model of Authorization for Next-Generation Database Systems. *ACM Trans. on Database Systems*, 16(1):88–131, March 1991.

[23] J. Richardson, P. Schwarz, and L.F. Cabrera. CACL: Efficient Fine-Grained Protection for Objects. In *Proc. Int. Conf. on Object-Oriented Programming Systems, Languages, and Applications (OOPSLA)*, Vancouver, Canada, 1992.

[24] P. A. Vretanos "Web Feature Service Adopted Implementation Specification"

[25] http://www.geoplace.com/gr/webmapping/sites.asp

[26] www.opengeospatial.org

[27] www.uml.org/ “OMG Unified Modeling Language Specification”

[28] www.w3.org/TR/SVG/ “Scalable Vector Graphics (SVG) 1.1 Specification”

[29] www.webmapping.org/onlinedemos.html

[30] François Yergeau, Tim Bray, Jean Paoli, C. M. Sperberg-McQueen, Eve Maler “Extensible Markup Language (XML) W3C Recommendation”