Note di sviluppo

Sistema Editoriale

Version: R1A

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| <dd/mmm/yy> | R1A | <details> | <name> |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Introduzione e scopo delle "Note di Sviluppo" 4

2. Descrizione generale di Business 5

2.1 Definizioni, Acronimi ed Abbreviazioni 5

2.2 Riferimenti (documenti, siti web, etc.) 5

2.3 Descrizione e requisiti di alto livello 5

3. Architettura (opzionale) 6

3.1 Descrizione di alto livello 6

3.2 Elenco dei moduli da realizzare (prossimi capitoli) 6

4. Modulo da realizzare: Xxx 7

4.1 Descrizione di dettaglio (implementazione) del modulo 7

4.2 Questionario (opzionale) 7

4.3 Interfacce grafiche (se presenti) 7

4.3.1 Interfacce grafiche da realizzare 7

4.4 Use case diagram 7

4.5 Sequence diagram 8

4.6 Class diagram 8

4.7 Diagrammi, disegni e materiale addizionale utile (opzionale) 8

5. Documentazione 9

5.1 Manuale di installazione e configurazione 9

5.2 Manuale utente 9

5.3 Test Case 9

Note di Sviluppo

Introduzione e scopo delle "Note di Sviluppo"

*Lo scopo di questo documento è principalmente quello di centralizzare in un unico contenitore tutte quelle informazioni necessarie allo sviluppo di un intero modulo/progetto software, dai requisiti iniziali, ai diagrammi che servono per realizzarlo, alle note tecniche necessarie per lo sviluppo software, fino ad arrivare alla note per poi preparare la documentazione ufficiale utile all’utente finale (su documenti separati). In pratica raggruppa tutte quelle informazioni definite da tutti gli attori coinvolti nello sviluppo software: capi progetto, analisti e sviluppatori del sistema. Serve quindi da collettore unico per le informazioni di uno specifico sviluppo software. Una sorta di diario di bordo dedicato a descrivere la vita di un software. Può essere utilizzato sia per la realizzazione di un intero progetto software, composto di più moduli, oppure per realizzare un solo modulo di un progetto già esistente. E’ inteso per essere leggero ed essenziale nel contenuto, in modo da essere effettivamente utile al gruppo di progetto.*

***Verrà utilizzato in questo modo:***

* *All’avvio di un nuovo sviluppo software verranno compilati i capitoli relativi alla descrizione di Business del modulo da realizzare. In genere i capitoli relativi verranno compilati da chi avverte l’esigenza di implementare un nuovo software, o perché richiesto dal cliente oppure perché avvertito come necessario per la naturale evoluzione del sistema. In alcuni casi, se i requisiti sono già definiti su una fonte esterna (documento o altro), sarà sufficiente inserire nel presente documento un riferimento ai requisiti esterni*
* *Una volta completata la prima fase (definizione dei requisiti), gli analisti e sviluppatori del progetto dovranno compilare le parti relative alla descrizione tecnica del progetto e dei moduli, definendo anche i diagrammi realizzativi, necessari poi allo sviluppo vero e proprio del modulo*
* *Gli sviluppatori, se necessario, raffineranno le descrizioni ed i diagrammi realizzati nella fase precedente*

*Naturalmente, trattandosi di un documento che nasce con l’esigenza e l’obiettivo di essere utile piuttosto che prolisso ed inutilmente complicato, qualsiasi persona del team che voglia migliorare l’utilità del documento stesso, sia togliendo parti inutili che aggiungendo capitoli o paragrafi, è invitato a proporre, ogni volta che ne sentirà l’esigenza, i vari cambiamenti. Il template verrà quindi aggiornato e migliorato man mano che se ne avvertirà l’esigenza.*

***Strumenti consigliati per la preparazione degli schemi:***

* *Per i diagrammi UML: astah\_community (ex. JUDE)*
* *Per la modellazione DB (se necessaria): MySQL Workbench o altro*
* *Per disegnare eventuali pagine HTML (form e quant’altro): Qualsiasi*

*I documenti prodotti (ed in via di produzione) devono essere posizionati dentro il repository di sviluppo (cvs, svn, git, etc.) e gestito esattamente come codice sorgente.*

Descrizione generale di Business

*Questo capitolo descrive, per quanto possibile e secondo quanto percepito con il cliente, lo sviluppo software che deve essere realizzato, più in termini di requisiti di Business che in termini di descrizione architetturale del software stesso. Definisce quindi i requisiti che sono stati avvertiti come utili per accrescere le funzionalità del sistema.*

*Il livello di dettaglio deve quindi essere appositamente generale per essere compreso da tutti e sufficientemente dettagliato per poter iniziare a definire i diagrammi che seguono nel presente documento: Use Case, Sequenze, etc.*

*In alcuni casi, se i requisiti sono già definiti su una fonte esterna (documento o altro), sarà sufficiente inserire nel presente documento un riferimento ai requisiti esterni.*

Definizioni, Acronimi ed Abbreviazioni

I termini, gli acronimi, le abbreviazioni di interesse al progetto (quando non ovvi e/o già comunemente noti) sono definiti qui di seguito:

|  |  |
| --- | --- |
| Nome | Descrizione |
| N/A | N/A |

Riferimenti (documenti, siti web, etc.)

|  |  |
| --- | --- |
| Nome | Descrizione |
| Project Work UIIP- Sistema Editoriale 1.1 | Specifiche dei requisiti |

Descrizione e requisiti di alto livello (revisione rispetto ai req. originali)

Il sistema redazionale deve implementare i seguenti requisiti:

Dare ai giornalisti la possibilità di

* accedere alle funzionalità del sistema via web, tramite browser, previo inserimento di username e password
* visualizzare la Lista delle Notizie presenti nella redazione
* creare una nuova Notizia
* modificare in modalità esclusiva una Notizia presente nella redazione
* registrare le modifiche effettuate sui campi della Notizia
* visualizzare una Notizia presente nella redazione
* cancellare una Notizia presente nella redazione: la cancellazione è logica e non fisica.
* trasmettere una Notizia verso un sistema esterno.
* annulla le modifiche apportate ad una Notizia prese in modifica: delockare tutte le notizie prese in esclusiva dall’utente autenticato, che ha preso in modifica una o più notizie. Tale funzionalità và utilizzata sia nella voce relativa alla singola notizia che nella lista completa delle notizie, dove verranno annullate tutte le modifiche iniziate e non registrate.

Dare ad un amministratore la possibilità di

* creare un account per un giornalista
* cancellare un account per un giornalista: la cancellazione è logica e non fisica
* modificare un account per un giornalista
* visualizzare la lista degli account presenti.

Dare la possibilità ai sistemi Fornitori di inviare notizie al sistema redazionale su directory condivisa in formato xml.

Dare la possibilità sistemi Clienti di ricevere le notizie dal sistema editoriale su directory condivisa in formato xml.

Ulteriori assunzioni

* Il versionamento delle modifiche viene realizzato tramite l’utilizzo del file di log.
* Un utente registrato può appartenere a più gruppi, quindi può essere contemporaneamente sia giornalista che amministratore.
* A seconda dell’utente che accede al sistema, verrà visualizzata una pagina contenente le funzionalità relative al gruppo di appartenenza, ad es. un Giornalista visualizzerà solo le funzionalità che gli competono. Solo nel caso in cui un utente sia contemporaneamente Amministratore e giornalista, avrà la possibilità di visualizzare tutte le funzionalità del sistema.
* Il sistema dà la possibilità all’amministratore di creare account solo di tipo Giornalista, e nel momento di installazione del database, saranno già preconfigurati tre utenti con il ruolo di Amministratore.
* Il sistema dà la possibilità ad un amministratore di aggiungere al suo account l’apparteneza al gruppo Giornalista.
* Non appena una Notizia viene Trasmessa non è più modificabile.
* La chiave della tabella Account è l’Username rappresentato da un indirizzo e-mail.

# Architettura

Descrizione di alto livello

L’architettura del sistema rispetta la specifica JavaEE ed è composta da tre layer:

* Il Presentation Layer genera delle pagine web dinamiche in base ai dati forniti in input dal client che vengono interpretati e passati al Business Layer. I contenuti dinamici sono creati grazie alla realizzazione, a questo livello, di un pattern MVC (Model View Controller) che utilizza Servlet e JSP.

Il Controller ricevendo le richieste HTML dal Client, le inoltra alle Servlet di competenza, che utilizzando il Model accede alla logica di Business dell’applicazione richiamando il Web Service appropriato. La risposta, gestita dal Controller, viene inoltrata alla View che permette la visualizzazione al Client.

Dopo aver valutato varie tecnologie si è scelto di utilizzare il framework opensource Struts per implementare il design pattern Model View Controller. Tale frame work garantisce modularità, riusabilità, manutenibilità e rapidità di sviluppo in quanto l’applicazione, costituita da livelli logici distinti, consente di scrivere codice modulare, riutilizzabile e sviluppabile in parallelo.

In questo layer viene anche gestito il controllo sui dati inseriti e sui permessi di un utente per compiere delle funzionalità.

Il Presentation Layer viene implementato su Web Server Apache Tomcat versione 6.

* Il Business Layer contiene la logica applicativa che interpreta ed elabora i dati forniti in input dal livello superiore. Il Layer è costituito da un’Application Server che mette a disposizione una serie di servizi implementati come dei WebServices utilizzando la tecnologia Apache Axis 2.

Tra i Web Services che verranno implementati nell’architettura, GestioneLogin e GestioneAccount si occupano dell’accesso dell’utente al sistema, della creazione/modifica di nuovi account e dei relativi controlli utilizzando le informazioni nel Data Layer.

Il Web Service GestioneNotizia implementa la logica per l’inserimento, la modifica oppure la cancellazione delle notizie.

Ogni Web Services indicato prevederà un oggetto Validator che si occupa dei controlli sui dati inseriti e sui permessi di un utente per compiere delle funzionalità.

In tale Layer viene inserito un oggetto Service Locator per il caricamento univoco dei file di configurazione relativi ai vari servizi da implementare e per la gestione della lingua dei messaggi di errore allo startup.

È previsto un Business Layer Interface per disaccoppiare il Front-End dell’applicazione con il Business Layer.

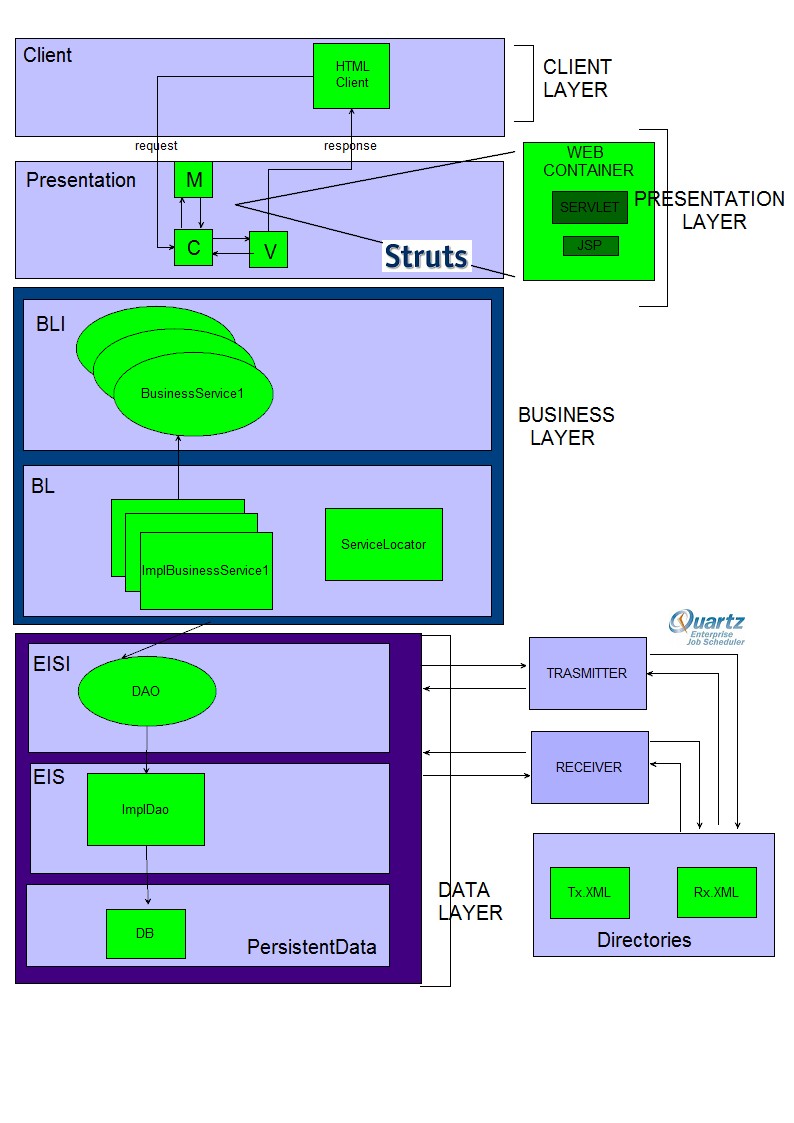
* Il Data Layer è il punto d’accesso alle risorse utilizzate. Al suo interno un ruolo importante viene ricoperto dal Data Access Object (DAO) il quale richiamando le Stored Procedures implementate in Oracle PLSQL gestisce la persistenza dei dati.

Il Database relazionale è realizzato in Oracle DataBase 11g e fornisce ai Layer superiori i dati necessari per elaborare le risposte da fornire al client.

Anche in tale Layer è prevista una interfaccia per disaccoppiare il livello di Business Logic dalle risorse del sistema.

I processi di Trasmissione e Ricezione per le notizie batch sono esterni al sistema e sono gestite tramite uno scheduler che utilizza il frame work Quartz.

Tale frame work consente di eseguire tali processi con un periodo prefissato, comunicando con il Data Layer dell’applicazione e in particolare con il DAO.

Il processo Trasmissione consente di trasferire le notizie dal DB verso un’apposita directory dopo un’opportuna validazione del formato. L’oggetto Ricezione si occupa invece di prelevare le notizie inviate da un Fornitore esterno al sistema e renderle persistenti dopo la validazione.

**3.1.1 Tecnologie Utilizzate**

Web Server Apache Tomcat versione 6.0

Web Services con teconologia Apache Axis2 1.6.

Database: Oracle 11g.

Receiver e Transmitter: Framework Quartz.

Log: Log4j logger

Presentation: Apache Struts 2.3 per Web App

Gestione Lingue: Framework Internazionalizzazione e localizzazione (i18n).

Test: JUnit

**3.1.2 Realizzazione Fisica**

La suddivisione dell’architettura logica in Layer disaccoppiati grazie all’utilizzo di interfacce consente l’eventuale deployment di ogni singolo layer del sistema su web server distinti per garantire maggiore robustezza, modularità e scalabilità. In particolare il Data Layer potrebbe essere deployato su un server diverso rispetto al Presentation Layer e Business Layer affinchè le risorse siano dislocate in posti diversi rispetto a che le utilizza. I Web Services dell’applicazione implementata sono localizzati tutti su un unico server e inglobati in un unico Business Layer per motivi di semplicità.

Elenco dei moduli da realizzare (prossimi capitoli)

|  |  |
| --- | --- |
| Modulo | Descrizione |
| N/A | N/A |

# Modulo da realizzare: Xxx

Descrizione di dettaglio (implementazione) del modulo

*Questo paragrafo descrive, in modo dettagliato e preciso, il modo in cui verrà implementato il modulo software un in oggetto. Questa descrizione dovrà essere verificata dai responsabili del progetto per essere sicuri che quando descritto sopra, nella descrizione di Business, è effettivamente quello che si voleva realizzare. E’ molto importante che la descrizione contenga, definiti in modo chiaro, gli oggetti (intesi non come oggetti Java) con cui il modulo software dovrà trattare.*

***Esempio****: L’invocazione del* ***Web Service*** *esposto da sistema-editoriale, codificata in un messaggio* ***SOAP*** *che viaggia sul protocollo* ***HTTP,*** *verrà gestita da una* ***Servlet*** *sistema-editoriale, che analizzerà ….*

<Inserire qui la descrizione>

Questionario (opzionale)

*Il seguente questionario è stato pensato per facilitare la comprensione del modulo stesso e per permettere una valutazione degli impatti che il nuovo modulo può avere sulle varie aree sistema-editoriale. L’utilità del presente paragrafo, è pensata per progetti a lungo termine, che prevedono la successiva realizzazione di moduli aggiuntivi (adapter, plug-ins, moduli di business complessi, etc.) ad un progetto esistente. In questi casi il presente template (da cui vengono generati i documenti specifici) deve essere custom-izzato per lo specifico progetto.*

|  |  |
| --- | --- |
| Domande (alcune saranno N/A: Not Applicable) | Risposte e note |
| Di che tipo di modulo si tratta: war, ear, etc.? |  |
| Il modulo necessita di configurazione? |  |
| Quanti e quali sono, se presenti, i tipi di input su cui dovrà lavorare il modulo? |  |
| Quanti e quali sono, se presenti, i tipi di output che il modulo dovrà produrre? |  |
| Il modulo utilizzerà delle librerie di terze parti complesse? Es. Axis, etc. |  |
| La complessità del codice può essere giudicata bassa, media o alta? |  |
| ... |  |

Interfacce grafiche (se presenti)

*Sono riportate di seguito le informazioni ed i disegni utili per la definizione delle interfacce grafiche. A tale scopo possono essere utilizzati strumenti di html authoring, etc., in modo da avere un indicazione, già in fase di definizione dei requisiti e dell’architettura, di cosa conterranno (anche se approssimativamente) le interfacce grafiche.*

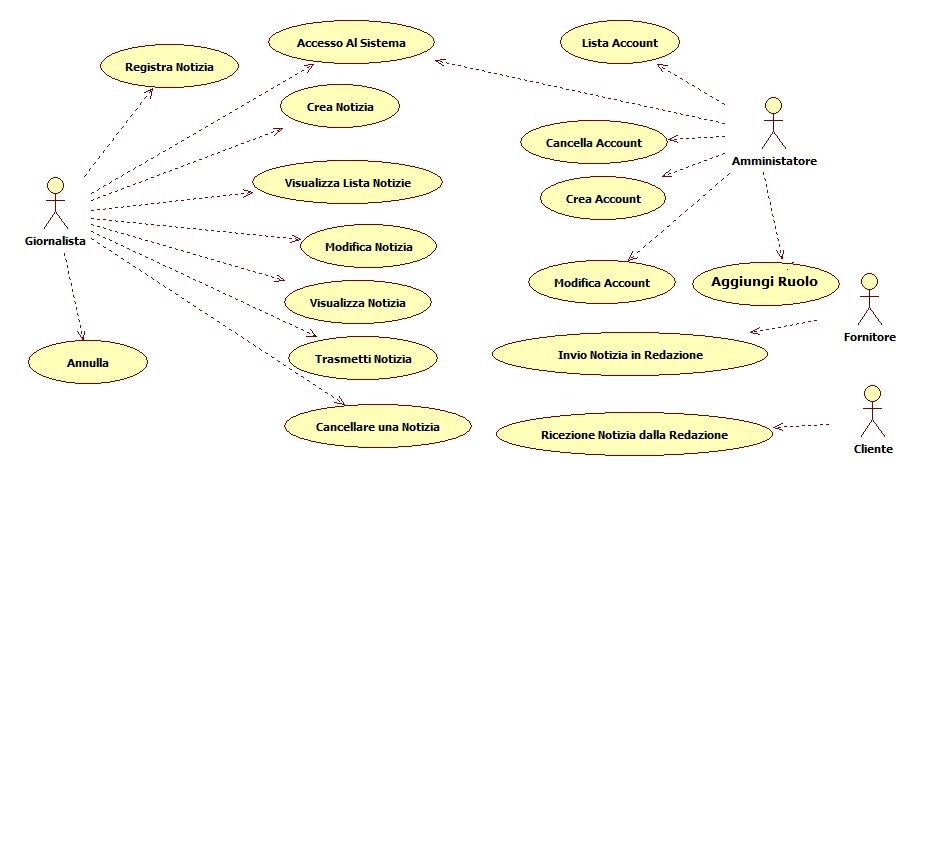
### Interfacce grafiche da realizzare

<Inserire qui i disegni>

Use case diagram

*Lo Use Case Diagram è realizzato, per quanto possibile ed utile, per tutti gli attori che sono significativi per il modulo da realizzare.*

**Diagramma dei casi d’uso:**

Sequence diagram

<Inserire qui la descrizione>

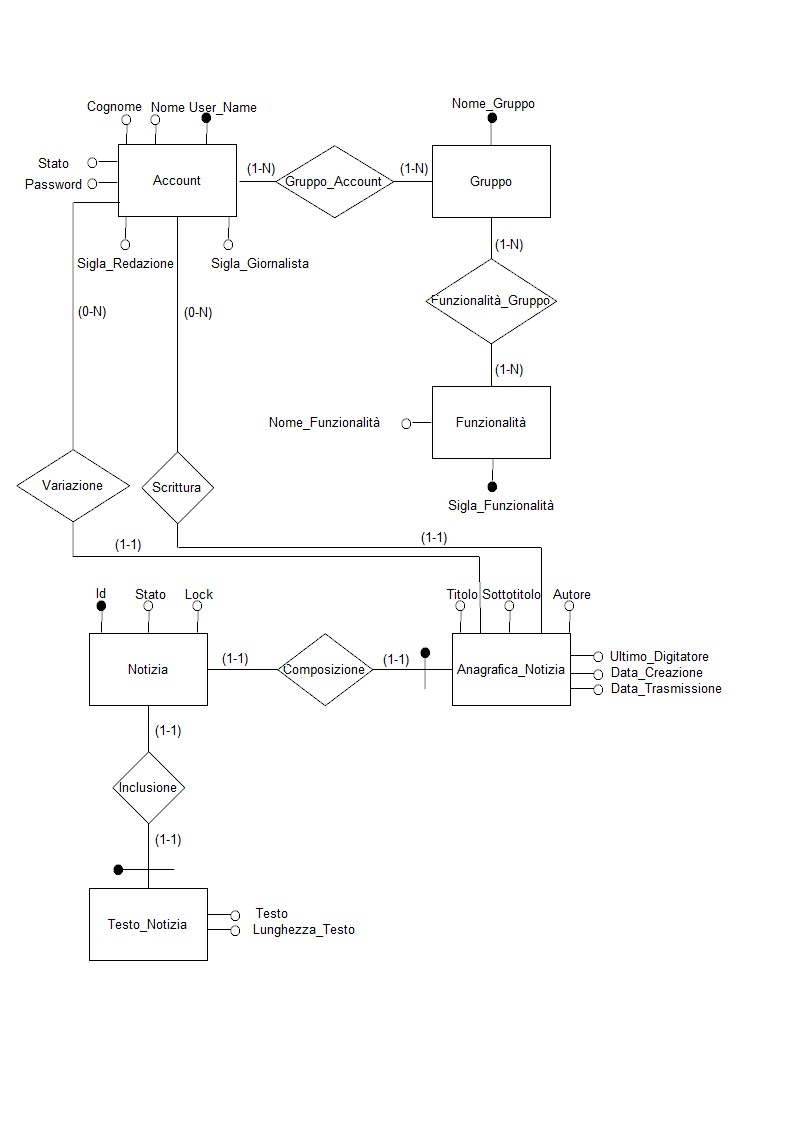
Class diagram

<Inserire qui la descrizione>

Diagrammi, disegni e materiale addizionale utile (opzionale)

Tutti i disegni, diagrammi ed in genere qualsiasi materiale utile alla comprensione del modulo da realizzare sono riportati di seguito. Ad esempio diagrammi E/R o altro possono essere inseriti qui.

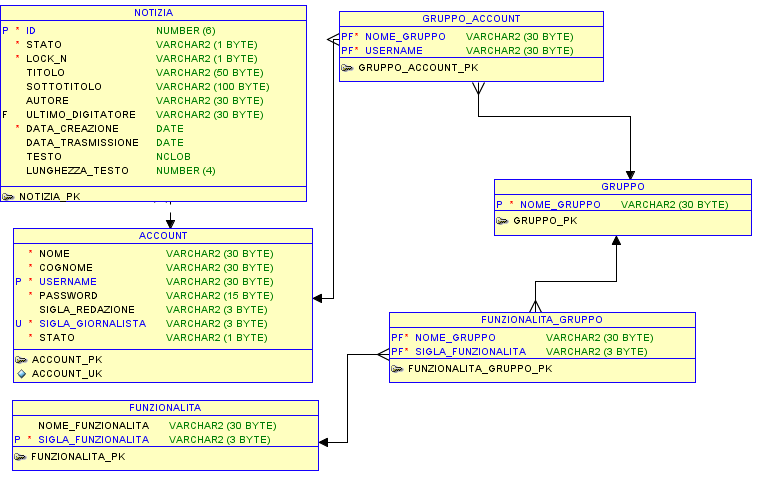
**Diagramma E-R Sistema Editoriale**

****

* Aggiunta la relazione Gruppo\_Account in quanto è necessario tener traccia di eventuali account appartenenti ad entrambi i gruppi (Giornalista e Amministratore).
* Aggiunta dell’attributo Stato all’entità Account per tener traccia di eventuali account cancellati logicamente.
* Aggiunta dell’attributo Id all’entità Notizia, il quale rappresenta la chiave primaria.

**Modello logico Sistema Editoriale:**

Come si può evincere dal modello sottostante le entità Testo\_Notizia ed Anagrafica\_Notizia, con i relativi attributi, sono state incorporate nell’ entità Notizia

****

**Script DataBase:**

**Create User:**

create user editoriale identified by editoriale;

grant all privileges to editoriale;

disconnect;

connect editoriale;

**Create Table gruppo:**

CREATE TABLE gruppo (

nome\_gruppo varchar2(30),

CONSTRAINT gruppo\_pk PRIMARY KEY (nome\_gruppo)

);

**Create Table account:**

CREATE TABLE account (

nome varchar2(30) default '' NOT NULL,

cognome varchar2(30) default '' NOT NULL,

username varchar2(30) default '' NOT NULL,

password varchar2(15) default '' NOT NULL,

sigla\_redazione varchar2(3) default '',

sigla\_giornalista varchar2(3) default '' NOT NULL,

stato varchar2(1) default 'A' NOT NULL,

CONSTRAINT account\_pk PRIMARY KEY (username),

CONSTRAINT account\_uk UNIQUE (sigla\_giornalista)

) ;

**Create Table gruppo\_account:**

CREATE TABLE gruppo\_account (

nome\_gruppo varchar2(30),

username varchar2(30),

CONSTRAINT gruppo\_account\_pk PRIMARY KEY (nome\_gruppo,username),

CONSTRAINT gruppo\_account\_fk1 FOREIGN KEY (nome\_gruppo) REFERENCES gruppo(nome\_gruppo) ON DELETE CASCADE,

CONSTRAINT gruppo\_account\_fk2 FOREIGN KEY (username) REFERENCES account(username) ON DELETE CASCADE

) ;

**Create Table funzionalità:**

CREATE TABLE funzionalita (

nome\_funzionalita varchar2(30),

sigla\_funzionalita varchar2(3),

CONSTRAINT funzionalita\_pk PRIMARY KEY (sigla\_funzionalita)

) ;

**Create Table funzionalità\_gruppo:**

CREATE TABLE funzionalita\_gruppo (

nome\_gruppo varchar2(30),

sigla\_funzionalita varchar(3),

CONSTRAINT funzionalita\_gruppo\_pk PRIMARY KEY (nome\_gruppo, sigla\_funzionalita),

CONSTRAINT funzionalita\_gruppo\_fk1 FOREIGN KEY (nome\_gruppo) REFERENCES gruppo(nome\_gruppo) ON DELETE CASCADE,

CONSTRAINT funzionalita\_gruppo\_fk2 FOREIGN KEY (sigla\_funzionalita) REFERENCES funzionalita(sigla\_funzionalita) ON DELETE CASCADE

) ;

**Create Table notizia:**

CREATE TABLE notizia (

id number(6),

stato varchar2(1) default 'S' NOT NULL,

lock\_n varchar2(1) default 'N' NOT NULL,

titolo varchar2(50),

sottotitolo varchar2(100),

autore varchar2(30),

ultimo\_digitatore varchar2(30),

data\_creazione date default CURRENT\_TIMESTAMP NOT NULL,

data\_trasmissione date,

testo NCLOB,

lunghezza\_testo number(4),

CONSTRAINT notizia\_pk PRIMARY KEY (id),

CONSTRAINT notizia\_fk FOREIGN KEY (ultimo\_digitatore) REFERENCES account(username)

) ;

**Create sequence seq\_notizie:**

genera una sequenza che sarà utilizzata nel trigger successivo

create sequence seq\_notizie INCREMENT BY 1 START WITH 1 NOCACHE NOCYCLE ;

**Create trigger trigger1:**

utilizza seq\_notizie per assegnare un id univoco ed incrementale alla notizia

CREATE or REPLACE

TRIGGER trigger1

BEFORE INSERT ON notizia

FOR EACH ROW

BEGIN

SELECT seq\_notizie.nextval

INTO :new.id FROM dual;

END trigger1;

/

**Create trigger trigger2:**

Simula una foreign key tra le tabelle Account e Notizia in modo da tener tracia di eventuali aggiornamenti sull’attributo autore

CREATE or REPLACE

TRIGGER trigger2

AFTER UPDATE OF username ON account

FOR EACH ROW

BEGIN

UPDATE notizia SET autore= :new.username WHERE autore= :old.username;

END;

/

**Create trigger trigger3:**

Implementa un Update on cascade nella foreign keytra le tabelle Account e Notizia in modo da aggiornare automaticamente il contenuto del campo ultimo\_digitatore della tabella Notizia nel caso di modifiche nel corrispondente campo username della tabella Account

CREATE or REPLACE

TRIGGER trigger3

AFTER UPDATE OF username ON account

FOR EACH ROW

BEGIN

UPDATE notizia SET ultimo\_digitatore= :new.username WHERE ultimo\_digitatore= :old.username;

END;

/

**Create trigger trigger4:**

Implementa un Update on cascade nella foreign keytra le tabelle Account e Gruppo\_Account in modo da aggiornare automaticamente il contenuto del campo username della tabella Gruppo\_Account nel caso di modifiche nel corrispondente campo username della tabella Account

CREATE or REPLACE

TRIGGER trigger4

AFTER UPDATE OF username ON account

FOR EACH ROW

BEGIN

UPDATE gruppo\_account SET username= :new.username WHERE username= :old.username;

END;

/

**Create trigger trigger5:**

Simula il comportamento di una Foreign key nel caso di inserimento di tupla nella tabella Notizia, controllando la reale esistenza dell’autore inserito nella tabella Account, o nel caso in cui la notizia sia stata ricevuta da un sistema esterno (RCV) si controlla la presenza della sigla RCV all’interno del campo autore. Nel caso in cui nessuna delle due condizioni sia verificata viene generato un errore Oracle (ORA-20000 – ‘Autore Non Valido’)

CREATE or REPLACE

TRIGGER trigger5

BEFORE INSERT ON notizia

FOR EACH ROW

DECLARE counter number;

BEGIN

SELECT count(\*) INTO counter from account where username = :new.autore;

IF

:new.autore != 'RCV' and counter = 0

THEN

raise\_application\_error(-20000, 'Autore non valido.');

END IF;

END;

/

**Insert:**

Inserimento di tre account appartenenti al gruppo Amministratore

INSERT INTO account VALUES ('admin', 'admin', 'admin@aa.a', 'admin', 'aaa', 'ad1', 'A');

INSERT INTO account VALUES ('admin1', 'admin1', 'admin1@aa.a', 'admin', 'aaa', 'ad2', 'A');

INSERT INTO account VALUES ('admin2', 'admin2', 'admin2@aa.a', 'admin', 'aaa', 'ad3', 'A');

<dato, disegno, diagramma o altro>

Documentazione

*Questo paragrafo ha lo scopo di contenere informazioni relative ai manuali da rilasciare all'utente finale.*

*L'ideale sarebbe di arrivare ad un punto in cui questi paragrafi non forniscano solo informazioni per realizzare i vari manuali da rilasciare all'utente, ma rappresentino i manuali stessi. In questo modo la realizzazione dei manuali verrebbe demandata agli sviluppatori stessi (cosa che in ogni caso verrebbe richiesta comunque alle stesse persone ;-) una volta finito lo sviluppo) e diluita nel tempo. In molti casi probabilmente il cliente finale chiederà un documento ufficiale di installazione ed un manuale utente.*

Manuale di installazione e configurazione

<Inserire qui le informazioni: sotto paragrafi e tutto>

Manuale utente

<Inserire qui le informazioni: sotto paragrafi e tutto>

Test Case

*Vanno indicati i test di accettazione che deve effettuare il cliente finale per ritenere il modulo, nuovo o modificato, accettabile.*

|  |  |
| --- | --- |
| Test Case Name | Descrizione |
| N/A | N/A |