Dal sistema informativo INFEA al sistema informativo SVS

Nicola Aloia, Cesare Concordia, Maria Teresa Paratore,
Sabrina Tardelli, Loredana Versienti,

Consiglio Nazionale delle Ricerche
Istituto di Scienza e Tecnologie dell'Informazione

Indice

Introduzione

Il sistema informativo INFEA, progettato, ed in gran parte realizzato presso il CNUCE (ora Istituto di Scienza e Tecnologie dell'Informazione – I.S.T.I.), del Consiglio Nazionale delle Ricerche, nasce, principalmente, dalla necessità, del Ministero dell'Ambiente e della Tutela del Territorio, di fornire un punto di riferimento nazionale per il reperimento e la gestione, integrata, di dati e risorse distribuite sul territorio dai vari centri e laboratori, finalizzata soprattutto alle attività di formazione ed educazione ambientale. La prima versione della componente informatica del sistema è stata completata agli inizi del 2002, e dopo una fase di test e popolamento del contenuto informativo, ad uso interno del Ministero, è stata resa ufficialmente fruibile, per l'intera comunità, dal 10 maggio 2002 (si rimanda alle pubblicazioni citate in Bibliografia per approfondimenti sulle funzionalità di INFEA).

La buona accettazione della soluzione adottata ha fatto scaturire nuove esigenze informative e gestionali che ha generato una proposta di estensione ad INFEA, che ha contemplato la sua graduale evoluzione verso il sistema informativo del Servizio Sviluppo Sostenibile del Ministero dell'Ambiente. Tale evoluzione è iniziata attraverso la definizione e realizzazione di strumenti utili sia per funzionalità di governo interno, che per le nuove esigenze di distribuzione dell'informazione correlate alla legge sulla trasparenza.

Il presente rapporto descrive le estensioni progettate e realizzate nell'ambito dell'accordo di collaborazione col Ministero dell'Ambiente (prot. n. SvS/C7/9655 del 19/12/2001) e le nuove problematiche affrontate. Occorre segnalare che alcune componenti software, qui descritte, sono state realizzate presso il Consorzio Pisa Ricerche, col coordinamento dell'I.S.T.I., che ne ha curato il collaudo e la rispondenza alle specifiche fornite.

1 Autenticazione e sicurezza.

Una delle più importanti problematiche che abbiamo affrontato è stata quella legata ai meccanismi di autenticazione e sicurezza del sistema informativo. Il sistema esteso (SVS) consente di gestire in maniera integrata, oltre ai dati di dominio pubblico, come le attività educative e formative promosse dal Ministero, anche dati gestionali riservati, quali ad esempio quelli relativi ai pagamenti di contratti o progetti finanziati, ecc.. Le componenti software coinvolte in questo processo comprendono: il gestore dei dati, il meccanismo di autenticazione, i certificati digitali ed i moduli specifici per la *Redazione* e l'*Amministrazione* del sistema informativo. Per quanto riguarda la prima componente abbiamo sfruttato le possibilità offerte dai sistemi di gestione di basi di dati, definendo opportune viste e credenzialità per l'accesso ai dati. Le altre componenti sono descritte di seguito.

1.1 Il meccanismo di autenticazione: utenti e ruoli

Abbiamo progettato e implementato il meccanismo di autenticazione del sistema SVS, sul concetto di *ruolo di utente*. Il ruolo determina per ogni utente un insieme dei servizi con relative modalità di fruizione. Il ruolo perciò definisce le funzioni disponibili per ogni tipologia di utente, e di conseguenza determina il sottoinsieme di dati di sua pertinenza, definendone le modalità di accesso (sola lettura, modifica, etc). Il meccanismo dei ruoli consente di presentare opportune interfacce uomo-macchina specifiche per le operazioni e per i servizi a cui l'utente ha accesso. Per esempio il servizio *Agenda* potrebbe essere fruibile da più ruoli di utenti, in cui ad uno è consentito l'inserimento e modifica di eventi, mentre all'altro è consentito la sola consultazione, in questo caso l'interfaccia per il primo ruolo contempla l'attivazione di azioni di inserimento/modifica che non sono, invece, disponibili nell'interfaccia per l'utente col secondo ruolo.

Dall'analisi delle attività, dei flussi di informazioni e da interviste intercorse con le varie tipologie dei possibili utilizzatori, sono stati individuati e definiti i seguenti ruoli:

1. *Ospite* questo ruolo è assegnato ad un utente generico Web, che accede al sistema per la consultazione di dati non riservati o di governo.

2. *Laboratorio* questo ruolo è ricoperto dagli utenti della rete dei laboratori territoriali istituiti dal ministero, e che, oltre alle possibilità di indagini sofisticate, svolgono attività di popolamento e modifica della base di dati.
3. *Redazione* questo ruolo è ricoperto da utenti (distribuiti sulla rete) che svolgono attività di filtro per la verifica della bontà dei dati, che affluiscono dai laboratori, e ne determinano la pubblicazione e la modalità di fruizione (es. se una determinata manifestazione debba comparire nell'agenda degli eventi rilevanti).
4. *Amministratore del sito*: è responsabile della gestione del sito, assegna i ruoli, comunica eventuali aggiornamenti e nuove funzionalità del sistema; attiva e coordina forum di discussione.
5. *Dirigente* ha strumenti di analisi dei dati e di supporto alle decisioni, definisce bandi e controlla le risorse finanziarie.
6. *Amministratore di sistema:* è responsabile della gestione dei componenti software del sistema.

Ad ognuno dei ruoli sopra elencati corrispondono specifiche interfacce utenti per l'accesso e la gestione delle informazioni. Per l'utente nel ruolo 6 non è stata studiata un'interfaccia grafica, in quanto molte delle attività svolte dal ruolo, si realizzano tramite l'interazione con i vari sistemi componenti (sistema operativo, gestore della base di dati, etc.), ma l'architettura dell'intero sistema prevede la possibilità di realizzare alcuni tools grafici di ausilio, per esempio: Compattamento del DB, Shutdown, Verifica di utilizzo, etc.
Il ruolo determina il tipo di operazione consentito sugli oggetti del sistema informativo, secondo la seguente tabella.

	CodOperazione	Ruolo
01	Lettura	Ospite
02	Proponi Inserimento	Laboratorio
03	Proponi modifica	Laboratorio
04	Proponi cancellazione	Laboratorio
05	Inserimento	Redazione/Amministrazione
06	Modifica	Redazione/Amministrazione
07	Cancellazione	Redazione/Amministrazione

Il numero d'ordine di un'operazione implica tutte le operazioni precedenti. Ogni ruolo può essere ricoperto da uno o più utenti. Il ruolo determina gli strumenti disponibili, le attività e le modalità di

accesso per l'utente che lo ricopre. Ad ogni utente del sistema informativo (tranne che per l'*UtenteOspite* che viene assunto per default) è assegnato un *identificativo*, una *password* ed un *ruolo*. Il ruolo dell'utente determina la tipologia dell'interfaccia e all'interno della categoria le funzionalità specifiche. La definizione delle varie porzioni di dati fruibili oltre che sul ruolo dell'utente si basa sul meccanismo delle viste del DBMS.

Per gli utenti col ruolo *Ospite*, sono disponibili le interfacce basate sul *Web browser* e quelle per i dispositivi mobili con microbrowser HTML o WAP. La scelta è stata principalmente motivata dal fatto che queste soluzioni non richiedono alcun software aggiuntivo (il *Plugin-Java*), oltre alla connessione Internet. Esiste comunque, la possibilità di accedere in sola lettura tramite l'interfaccia con *applet* Java anche per gli utenti ospiti. Per tutti gli altri ruoli le interfacce utenti sono realizzate con *applet* Java.

1.2 Base di dati per la gestione dei ruoli di utente

In figura 1 è mostrato lo schema concettuale per la gestione dei ruoli di utente. Gli utenti del sistema, tra le cui proprietà sono presenti un identificativo univoco ed una password, sono associati ad un gruppo, il cui obiettivo è di individuare determinate funzioni svolte dalle persone componenti (es. gruppo *laboratorio Sabaudia*). L'identificativo del gruppo, oltre a quello dell'utente, è associato ad ogni oggetto del sistema informativo SVS, al fine di poter controllare l'inserimento e la modifica dei valori (es. si potrà consentire di modificare un oggetto a tutti gli utenti appartenenti al gruppo ad esso associato, ma non a tutti gli altri utenti). Un utente può appartenere a più gruppi ed assume un ruolo relativo al gruppo (es. un determinato utente può essere redattore di più gruppi). È importante definire in maniera opportuna i vari gruppi di utenti, all'interno di un'organizzazione. Ad ogni ruolo sono associati determinati privilegi sugli oggetti del sistema informativo.

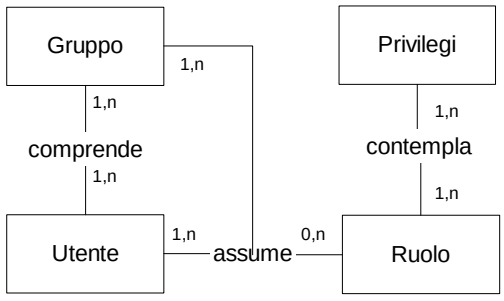


Figura 1 **Schema concettuale per la gestione dei ruoli di utente**

Nel seguito sono elencate le proprietà dettagliate dello schema concettuale.

Classe Utente: Contiene l'anagrafe degli utenti del sistema. I valori vengono forniti dall'utente all'atto della registrazione. Un utente registrato non è ancora abilitato a svolgere alcuna funzione fino al momento dell'assegnazione di un ruolo, da parte dell'amministrazione, che lo abilita ad operare.

Elenco proprietà:

1. ***IdGruppoPrimario*** –codice- gruppo primario di appartenenza, in assenza di specifica al login.
2. ***IdEnte*** –codice- identificativo dell'ente di appartenenza dell'utente
3. ***Utente*** –stringa- Identificativo dell'utente
4. ***Nome*** -stringa- nome dell'utente
5. ***Cognome*** -stringa- cognome dell'utente
6. ***Password*** -stringa-
7. ***Email*** -stringa- indirizzo di e-mail
8. ***CodStato*** –codifica- può assumere i valori: Abilitato (utente attivo a tutte le funzioni derivanti dal suo ruolo) o Disabilitato (utente registrato nel sistema ma non abilitato ad operare).

Classe Gruppo Raggruppa insieme di utenti abilitati ad operare su un insieme di oggetti del sistema informativo. Ad ogni gruppo è associato almeno un utente col ruolo *Redazione*, a cui arrivano le notifiche delle richieste di aggiornamento da parte degli utenti col ruolo *Laboratorio*. Nel caso di operazioni di inserimento di oggetti nel database da parte di utenti che appartengono a più gruppi (es. Redazione) occorre esprimere il gruppo con cui si intende operare, questo al fine di consentire la modifica dell'oggetto a tutti gli utenti appartenenti al gruppo. Come politica di gestione possiamo definire utenti appartenenti a più gruppi solo quelli col ruolo di Redazione.

Elenco proprietà:

1. ***NomeGruppo*** –stringa- Nome del gruppo
2. ***SiglaGruppo*** –codice- identificativo interno del gruppo, utile alla generazione automatica di chiavi.
3. ***Descrizione*** –stringa-

Classe Ruolo. Questa classe descrive i ruoli presenti nel sistema informativo SVS.

Elenco proprietà:

1. ***CodRuolo*** –codifica- assume uno dei valori: ospite, laboratorio, redazione, amministratore del sito, dirigente, amministratore del sistema.

Classe Privilegi. Questa classe descrive le operazioni consentite ad ogni gruppo per ogni oggetto del sistema informativo

Elenco proprietà:

1. ***IdGruppo*** –codice- identificativo interno del gruppo
2. ***CodOggetto*** -codifica- assume i valori della seguente tabella.

	CodOggetto
01	Agenda
02	Enti
03	Iniziative
05	Manifestazioni
06	Visite e soggiorni
07	Esperienze
08	Materiali
10	Progetti
11	Corsi di formazione
12	Corsi universitari
13	Corsi Master
14	Corsi istruzione secondaria
15	Corsi IFTS
16	Bando
17	Richiesta finanziamento
18	Valutazione richiesta
19	Progetto finanziato
20	Pagamento
21	Utenti e ruoli
100	Tutti

1.3 I certificati digitali in SVS

Per rendere più sicuri il meccanismo di autenticazione degli utenti e l'integrità dei dati da essi trattati, sono stati adottati i *certificati digitali*.
Un certificato digitale è un documento elettronico che contiene le informazioni che identificano in maniera certa il soggetto che lo possiede; in genere è rilasciato da una terza parte che ne garantisce la validità (Verisign, Thawte, etc).

L'uso dei certificati digitali per le comunicazioni Web può avvenire in vari modi, per il sistema SVS è stata scelto di implementare il protocollo Secure Socket Layer (SSL). Una volta che gli attori di una comunicazione hanno validato i certificati, i dati vengono criptati dal mittente e decodificati dal ricevente, che è il solo a possedere la chiave giusta.

Di seguito riportiamo la sequenza di visualizzazioni che documentano il verificarsi di una comunicazione sicura in SVS. In figura 2 il browser segnala che il server effettuerà una comunicazione criptata, mostrando le informazioni relative al certificato digitale e chiedendo conferma all'utente.

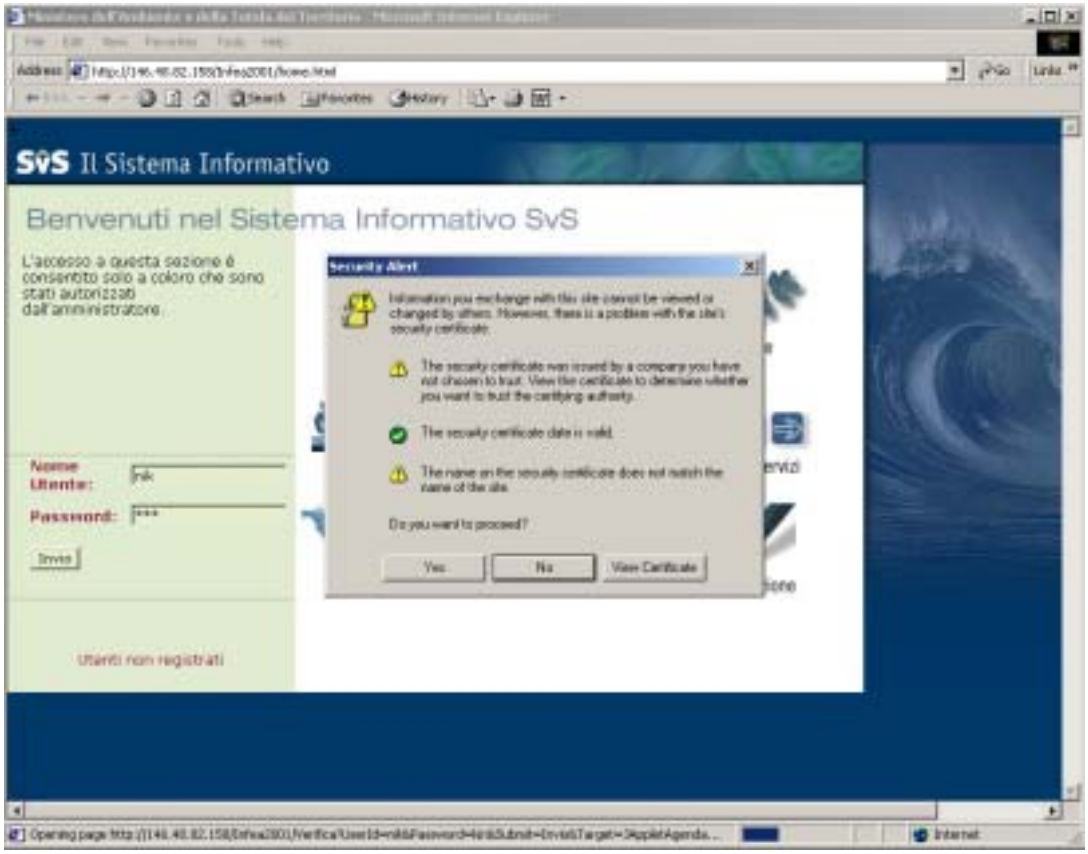


Figura 2 Informazioni sul certificato

Una volta accettato il certificato, la particolare natura della comunicazione (criptata) è evidenziata graficamente dal *lucchetto* (nella status bar del browser) che appare chiuso (Figura 3). Un secondo avviso arriva quando si carica il primo strumento di interazione con il server SVS, qualsiasi esso sia (Fig.4). L'utente può così verificare se gli strumenti sono quelli autentici e continuare o annullare la comunicazione.

Figura 3 Ambiente con comunicazione criptata

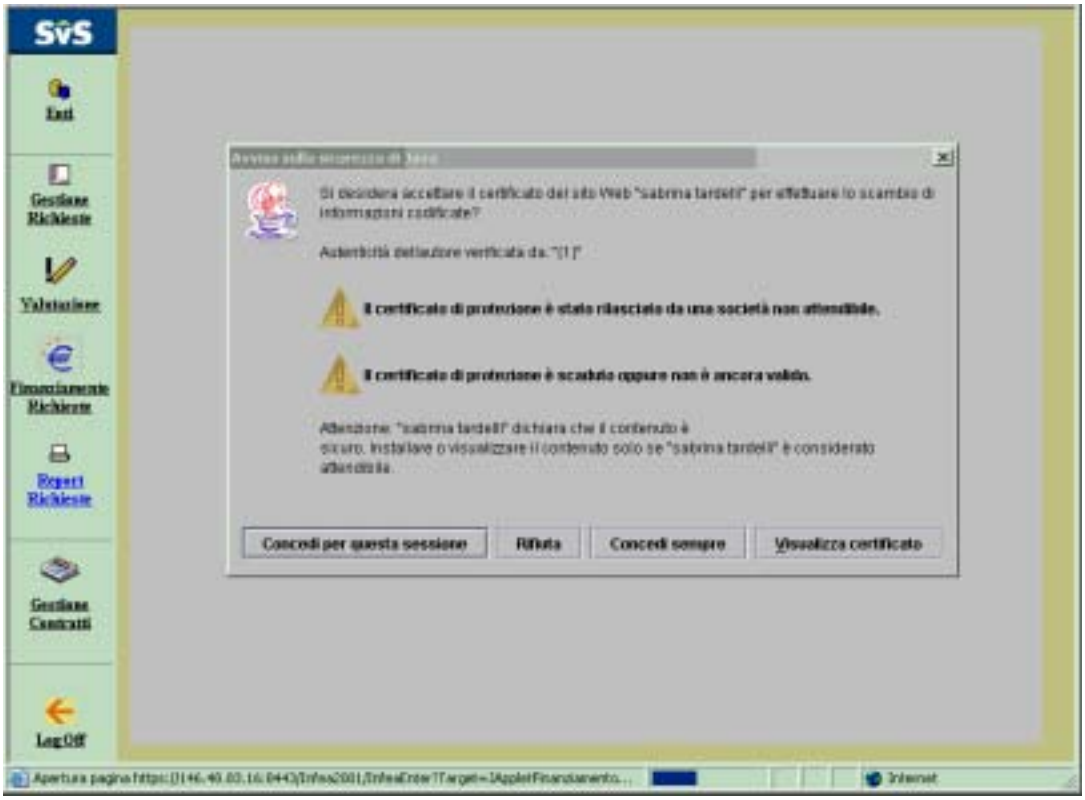


Figura 4Applet in ambiente criptato

I browser danno la possibilità di automatizzare questi controlli evitando il presentarsi delle finestre di dialogo descritte, in questo caso le scelte saranno fatte automaticamente in base a dei parametri impostati dall'utente.

Il certificato digitale adottato è stato generato dal gestore del server SVS.

1.4 Acquisizione di certificati digitali

In questo paragrafo illustriamo le modalità per l'acquisizione di certificati digitali. La politica adottata in fase di sviluppo del software (certificati autoprodotti) è, a nostro parere, inadeguata per la distribuzione ufficiale del sistema informativo, per cui occorrerebbe che il Ministero, attivasse le opportune pratiche per l'acquisizione di certificati rilasciati da una *Certification Authority (CA)*, ossia una delle organizzazioni iscritte al Registro dei Certificatori ufficiali che in Italia e mantenuto dall'AIPA e si può trovare all'URL: http://www.aipa.it/attivita[2/certifica[17/

1.5 Scelta del Certificatore

Nella scelta del certificatore ufficiale del sistema informativo, abbiamo deciso di adottare quanto stabilito dal centro tecnico della Rete Unitaria per la Pubblica Amministrazione (RUPA). Nella sezione dedicata alla firma digitale c'è un documento dal titolo "Certificatore" (http://www.ct.rupa.it/FIRMA-DIGI/Certificat/index.htm) nel quale si legge:

"*Il Centro Tecnico per la Rete Unitaria della Pubblica Amministrazione è preposto alle attività di certificazione e validazione temporale per i soggetti che utilizzano la Rete Unitaria per la Pubblica Amministrazione (RUPA).*"

[...]

"*Per espletare l'attività di certificazione, il Centro Tecnico si è avvalso della facoltà, prevista dall'articolo 62 dello stesso DPCM, di utilizzare i servizi offerti da un Certificatore ufficiale selezionato, attraverso una procedura concorsuale, tra quelli presenti nell'Elenco pubblico. La procedura si è conclusa nell'ottobre 2000, con l'aggiudicazione della fornitura alla* ***Società Postecom S.p.A.***"

1.6 Procedura per il rilascio del certificato.

Sul sito web della società Postecom S.p.A. la pagina relativa all'assegnazione dei certificati SSL si trova all'URL: http://www.poste.it/online/postecert/a_certificatiserver.shtml

Si tratta di certificati a chiave pubblica a 128 bit, conformi al protocollo SSL e quindi tecnicamente conformi ai meccanismi di sicurezza adottati nel sistema informativo. Il costo (gennaio 2003) è di 800 euro per certificato, l'acquisto dei certificati può avvenire on-line ed il pagamento anche tramite bonifico bancario.

Leggermente più complessa è la parte amministrativa: è necessario che la richiesta del certificato contenga l'indicazione di un responsabile legale e di un responsabile tecnico del server su cui il sistema è implementato. I responsabili devono essere esplicitamente indicati da un atto ufficiale dell'ente che richiede il certificato.

2 Il sottosistema dell'utente

L'introduzione dei meccanismi di autenticazione e sicurezza, descritti nel paragrafo precedente, hanno comportato notevoli cambiamenti nel software del sottosistema dell'utente, infatti tutte le componenti che realizzazione il lato *client* del sistema devono contemplare la nuova architettura.

Sono state inoltre arricchite le funzionalità dell'interfaccia basata sul browser, introducendo la possibilità di generare file PDF conteneti il risultato di interrogazioni.

2.1 Restituzione PDF

L'obiettivo è stato di introdurre nel sistema informativo la funzionalità di stampa. Il formato di visualizzazione sul browser infatti non risulta adatto per sua natura alla stampa dei dati (es. per stampare tutti i dettagli di un ente occorre visualizzare e stampare varie pagine). Per ovviare a questo inconveniente abbiamo progettato una nuova modalità di restituzione e realizzato una componente del sistema. Per l'implementazione abbiamo adottato la tecnologia Formatting Objects Processor (FOP), un processore basato su "formatting objects" XSL/FO. Con questo nuovo modulo, i dati estratti dal

database sono trasformati in formato PDF su richiesta del client. In figura a è mostrato, a destra, una pagina di restituzione generata dalla descrizione di sinistra.

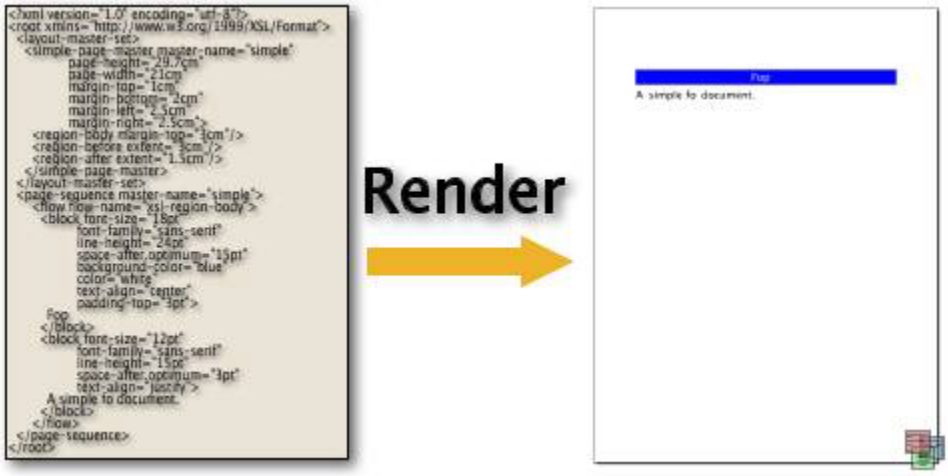


Figure a FOP

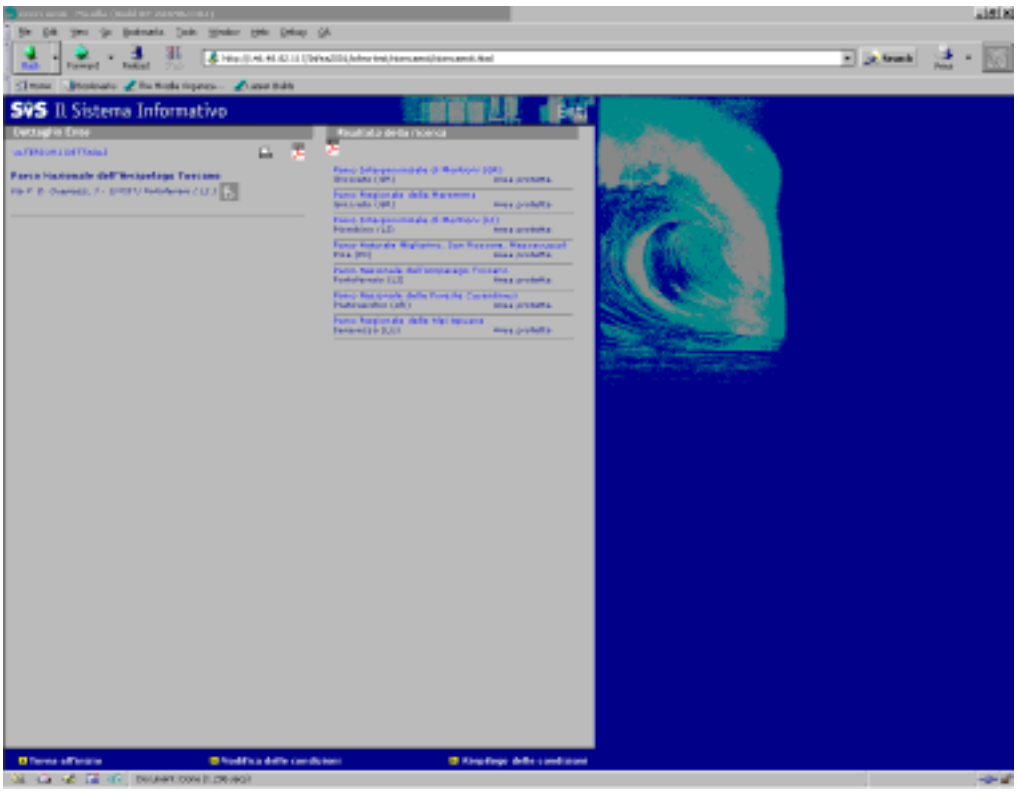


Figure b Modifiche alle pagine di restituzione

Alle pagine di restituzione generate dalla richiesta effettuata dall'utente (Fig. b), è stata aggiunta un'icona (che richiama il formato PDF), sia sul lato elenco (destra) che sul lato dettaglio (sinistra). Il

click dell'icona genera un processo di trasformazione del risultato dell'interrogazione, secondo le modalità di restituzione definite nel modulo FOP.

Nel modulo FOP si specificano le caratteristiche delle pagine e del loro contenuto, è possibile cambiare queste definizioni con tempi molto ragionevoli (nel modulo di restituzione reso fruibile abbiamo adottato le modalità di formattazione presenti nel catalogo cartaceo dei parchi). I dati possono essere inviati al client in formato PDF. È possibile ottenere il singolo oggetto (Fig. c)

Figure c Singolo oggetto

Oppure una lista di oggetti (Fig. d).

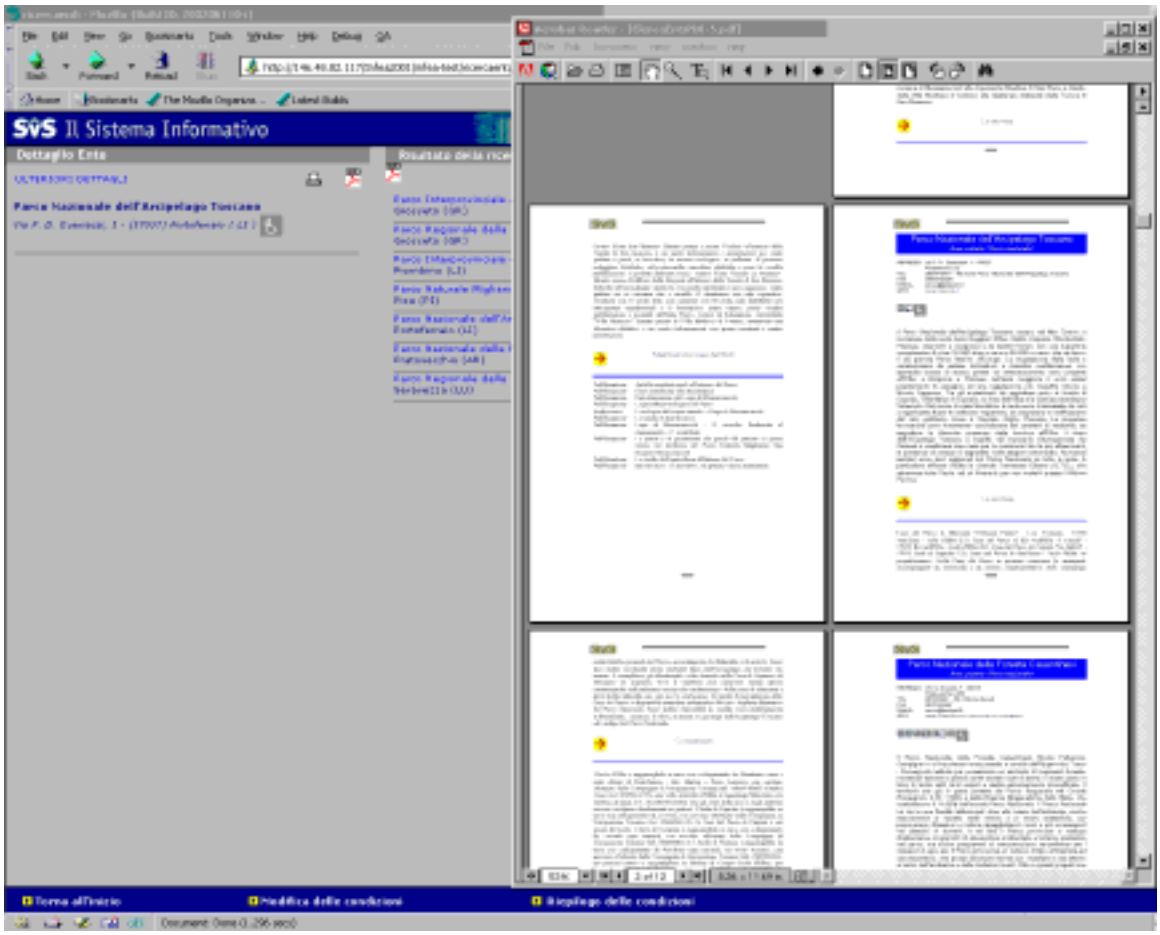


Figure d Elenco oggetti

Occorre segnalare che la produzione di lunghi elenchi di stampa, può caricare notevolmente il lavoro del server, per cui la fruibilità di questo processo dovrà essere determinata dalla politica dei ruoli.

3 Il sottosistema della redazione

Il sottosistema della redazione comprende un insieme di strumenti per le attività di governo del sistema informativo. L'I.S.T.I. ha curato la progettazione, la definizione delle specifiche implementative e il collaudo del software sviluppato dal Consorzio Pisa Ricerche.

La redazione del sistema informativo SVS è costituita da un insieme di persone, distribuite geograficamente sul territorio, che svolgono funzioni di coordinamento delle attività di produzione e distribuzione dell'informazione e fornisce le norme di compilazione dei vari documenti

Per agevolare l'attività delle persone coinvolte nel ruolo di redazione, abbiamo progettato un insieme di strumenti software a cui diamo il nome di "*sottosistema della redazione*". Il meccanismo fondamentale su cui si basa il sottosistema della redazione è quello del *ruolo dell'utente* e del meccanismo di *notifica degli eventi*.

3.1 Gestione del ruolo di utente.

Comprende l'insieme delle componenti software per la realizzazione del meccanismo di autenticazione presentato in 1.1. Sono state realizzate tutte le funzioni di interazione con la base di dati e i moduli per il controllo degli accessi.

3.1.1 Controllo degli accessi e dei diritti in SVS

L'approccio seguito è stato quello di implementare nel sistema informativo, i due principi che in letteratura prendono il nome di "ambiente chiuso" e "privilegio minimo" (conosciuti anche come primo e secondo principio di Denning).

Due sono le caratteristiche principali di un ambiente chiuso:

- Non esistono privilegi per default: un utente ha nei confronti di un oggetto *solo i privilegi che gli sono stati esplicitamente concessi*.
- Ogni accesso agli oggetti è completamente mediato dai meccanismi di protezione, compito di questi meccanismi è validare ogni richiesta di operazione su un oggetto da parte degli utenti.

Questa scelta si contrappone all'adozione del modello detto ad "ambiente aperto" nel quale un utente ha nei confronti di un oggetto *tutti i diritti che non gli sono stati esplicitamente negati*. Ne consegue che dimenticando di negare un diritto viene autorizzata una operazione potenzialmente illegale. Al contrario in un ambiente chiuso dimenticando di autorizzare un privilegio si disabilita una operazione legale.

Il principio del privilegio minimo rafforza ulteriormente gli effetti dell'ambiente chiuso. Esso prevede che un *soggetto possieda in ogni istante tutti e soli i diritti necessari per accedere agli oggetti riferiti nella fase di elaborazione*. Quindi ad ogni utente vengono dinamicamente assegnati e revocati i privilegi in modo che il suo dominio di protezione sia minimo.

I vantaggi di questo approccio sono enfatizzati soprattutto dalla natura distribuita del sistema SVS. Il sottosistema di interazione opera infatti via Internet dove non sono garantite le condizioni minime di sicurezza di una LAN o di una VPN; è richiesta quindi una maggiore attenzione alla sicurezza.

3.1.2 L'accesso al sistema SVS

Il controllo delle autenticazioni e la verifica dei privilegi utente nel sistema SVS sono effettuati dal modulo **Users manager**. Questo modulo ha la possibilità di assegnare e revocare dinamicamente i privilegi agli utenti ed implementa una matrice di protezione a *lista di capabilities*.

Descriviamo in dettaglio le varie fasi della procedura di autenticazione (Fig. e):

1. In fase di start up lo *Users manager* legge le informazioni sugli utenti, memorizzate nella base di dati, e costruisce i *profili* che vengono memorizzati in una struttura dati ad accesso veloce. Ogni operazione effettuata da un *amministratore* che va a modificare le informazioni sugli utenti richiede l'aggiornamento dei profili.
2. Quando viene effettuata una richiesta di accesso, gli identificativi (login e password) vengono verificati dallo *Users manager*. Se corretti, essi individuano univocamente il profilo utente, che contiene le informazioni necessarie alla definizione degli strumenti presenti nell'interfaccia utente ed alla gestione dei privilegi.
3. I profili degli utenti a cui è stata concessa l'autorizzazione sono copiati in una seconda struttura dati utilizzata per verificare a *run time* i permessi sulle singole operazioni. Ciascun utente a cui viene concessa l'autorizzazione sarà identificato univocamente da un identificativo interno assegnato dallo *Users manager*.

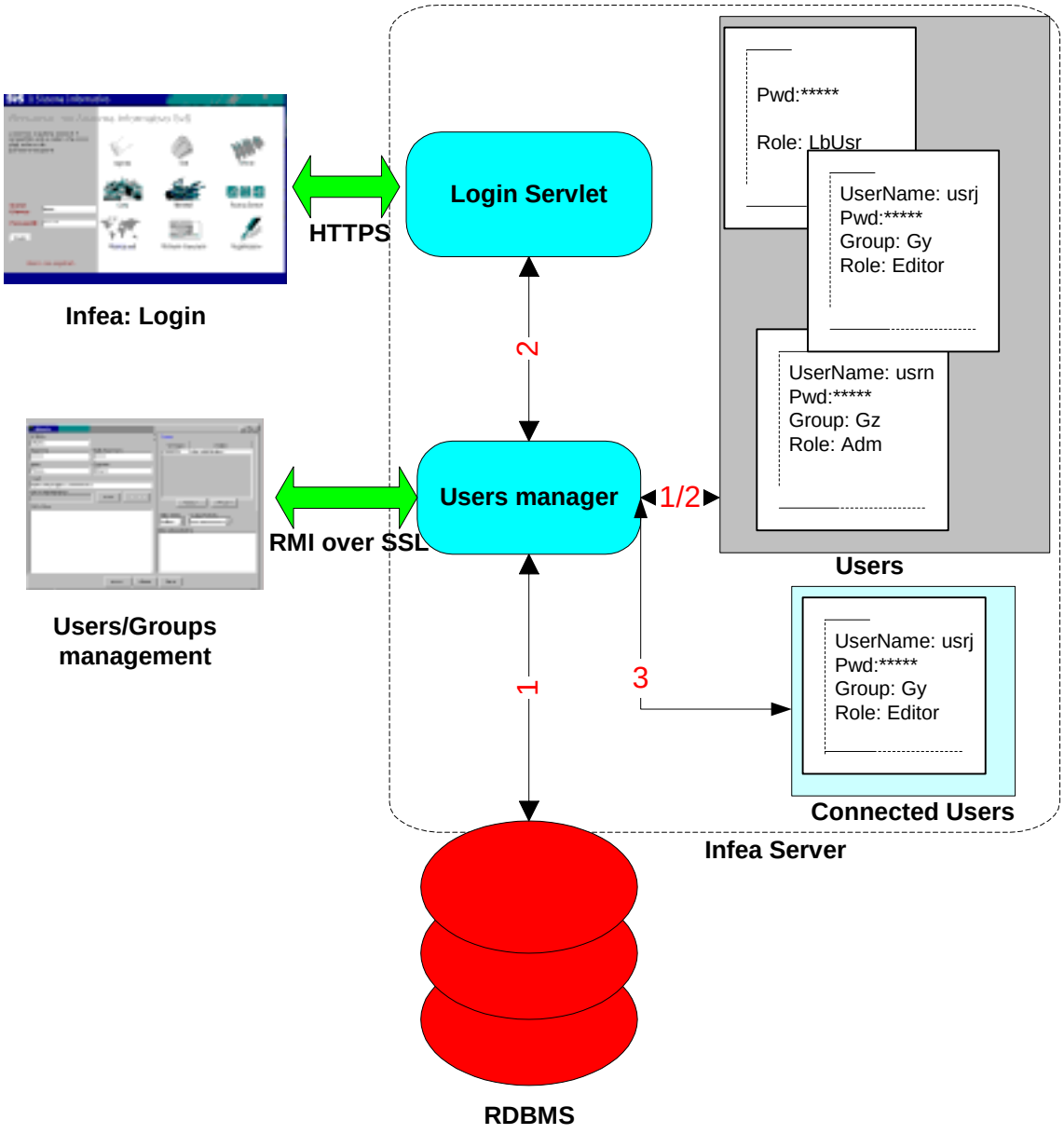


Figure e: Start Up e Verifica delle autorizzazioni alla connessione

3.1.3 Privilegio minimo

Ciascuna operazione che modifica il contenuto informativo del sistema SVS viene controllata a *run time* per verificare i diritti effettivi dello user che la esegue. La robustezza del sistema risulta esaltata da questo meccanismo, la contropartita è una perdita di efficienza nelle prestazioni. Abbiamo avuto modo

di valutare che il peso a livello di prestazioni del controllo esplicito di ogni operazione non influisce significativamente sul costo delle operazioni.

Descriviamo in maggiore dettaglio il meccanismo (Fig. f):

1. La richiesta effettuata dal client viene passata dal server RMI allo *Users manager* assieme all'identificativo univoco dell'utente. Lo *Users manager* controlla che la richiesta sia legittima ed assegna all'utente i privilegi per eseguirla.

2. In funzione della risposta dello *Users manager* il server RMI esegue l'operazione o segnala una situazione di errore. Una volta eseguita l'operazione i privilegi vengono revocati; l'esecuzione di una nuova richiesta del medesimo utente riattiverà la funzione di verifica.

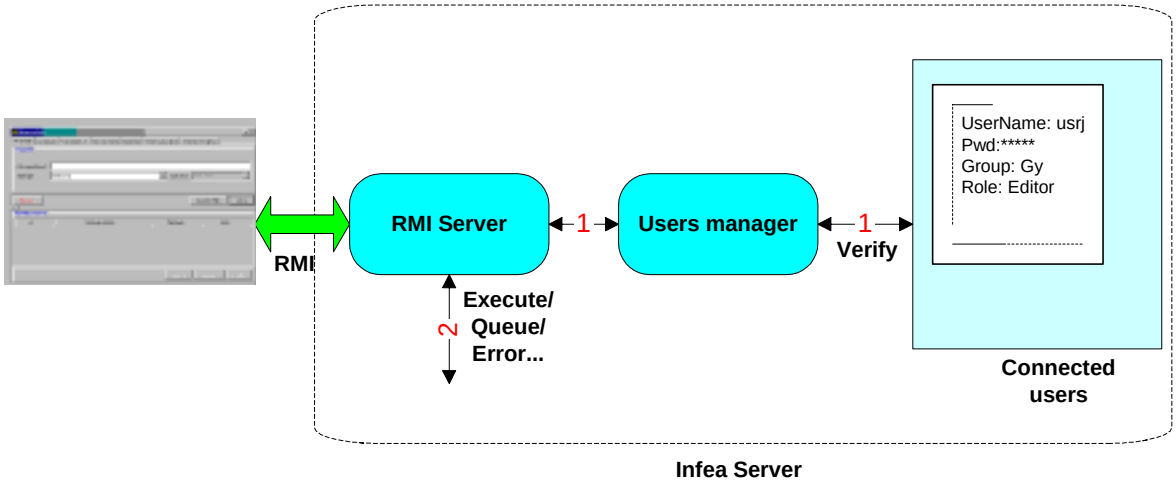


Figure f: Verifica a run time

3.2 Notifica degli eventi

Per evento si intende una operazione che modifica il contenuto informativo del sistema; gli eventi possono essere causati da operatori (aventi un qualsiasi ruolo) o da agenti automatici.
La maggior parte degli eventi devono essere controllati ed accettati/rifiutati dai componenti della redazione. Il sistema di notifica avvisa un redattore del verificarsi di un evento, consentendogli di controllarne il contenuto informativo e di agire accettando l'evento oppure rifiutandolo. Nel caso l'evento sia stato generato da un operatore questi riceverà notifica dell'azione svolta dal redattore. Occorre sottolineare che le funzioni di notifica sono state sviluppate con funzionalità minime per poter gestire le funzioni di redazione. L'architettura completa del sistema informativo prevede lo sviluppo di ciò che è stato chiamato "*Active Notifier*" (si rimanda alla documentazione in possesso di codesto Ministero) che non è oggetto dell'attuale contratto. Vediamo di seguito attraverso un esempio alcuni screenshots contenenti un caso d'uso che descrive il comportamento del sistema di notifica

L'inserimento da parte di un operatore di un nuovo *materiale* (Fig. 5), genera la notifica dell'evento sulla console del redattore (Fig. 6). In questa situazione i dati forniti tramite le interfacce di immissione e modifica dell'operatore, sono inseriti nella base di dati, ma non sono resi pubblici.

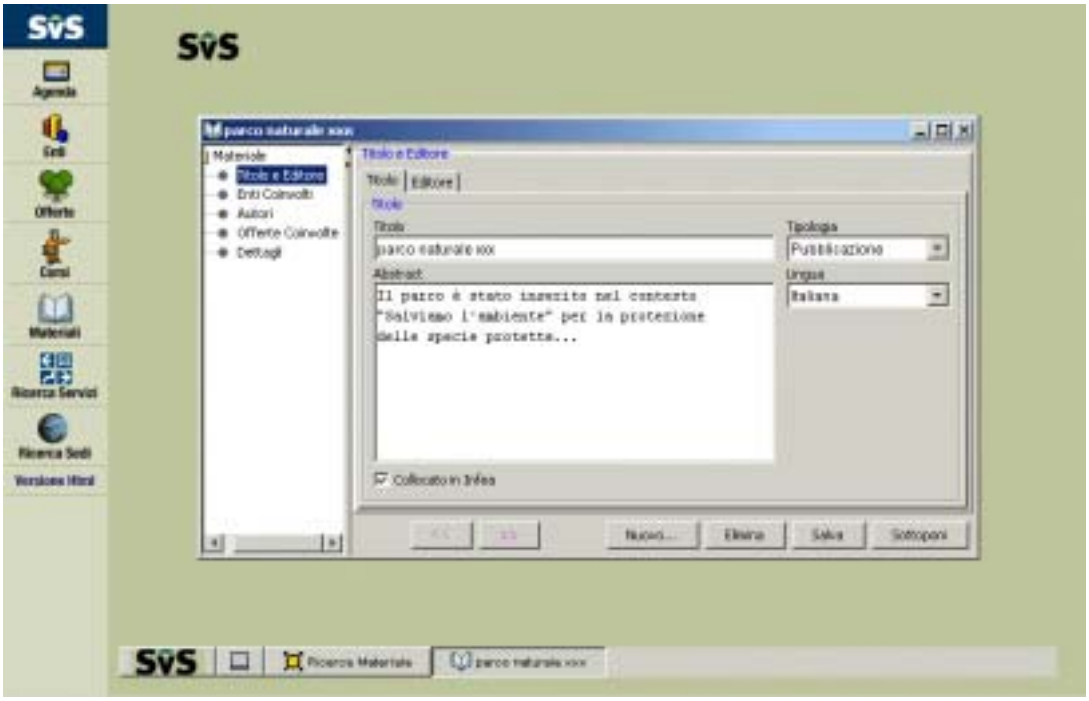


Figura 5 Desktop dell'operatore

Sul desktop del redattore, la presenza dell'icona gialla a forma di rombo a destra sulla barra dei comandi, segnala il verificarsi di eventi. Ricordiamo che nella coda di eventi di uno specifico redattore arriveranno solo quelli generati dagli utenti del gruppo da egli coordinato (si rimanda al paragrafo "Utenti e Ruoli"). In questa versione del sistema, le segnalazioni sono raggruppate per le classi di oggetto (es. Enti, Gestione Richieste, ecc.).

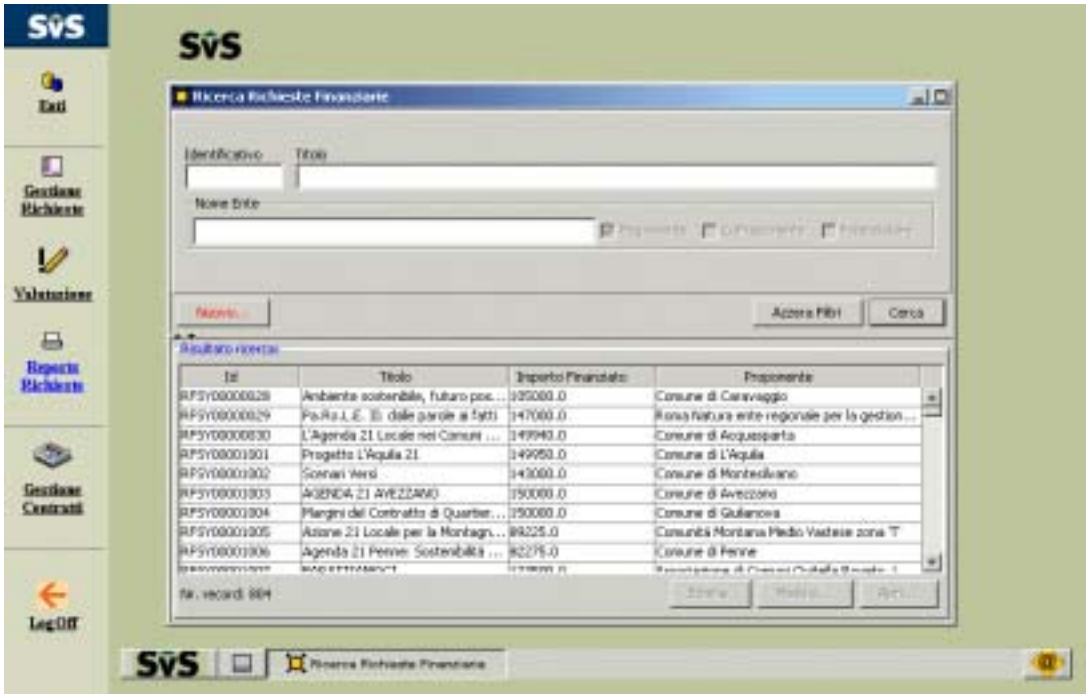


Figura 6 Dektop del redattore

Un click del mouse sull'icona romboidale apre la finestra (Fig. 7) contenente la sequenza di eventi verificatesi, che non sono ancora stati controllati, riportando la natura dell'operazione, il tipo di oggetto, l'operatore che ha generato l'evento e la data.

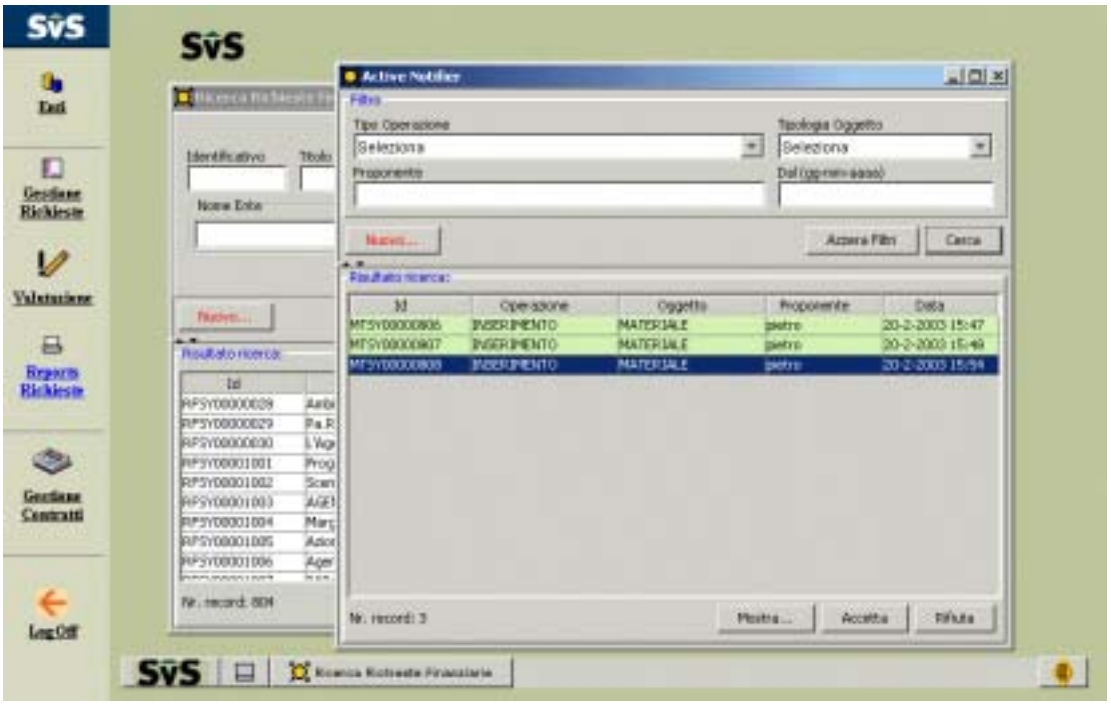


Figura 7 Elenco di eventi

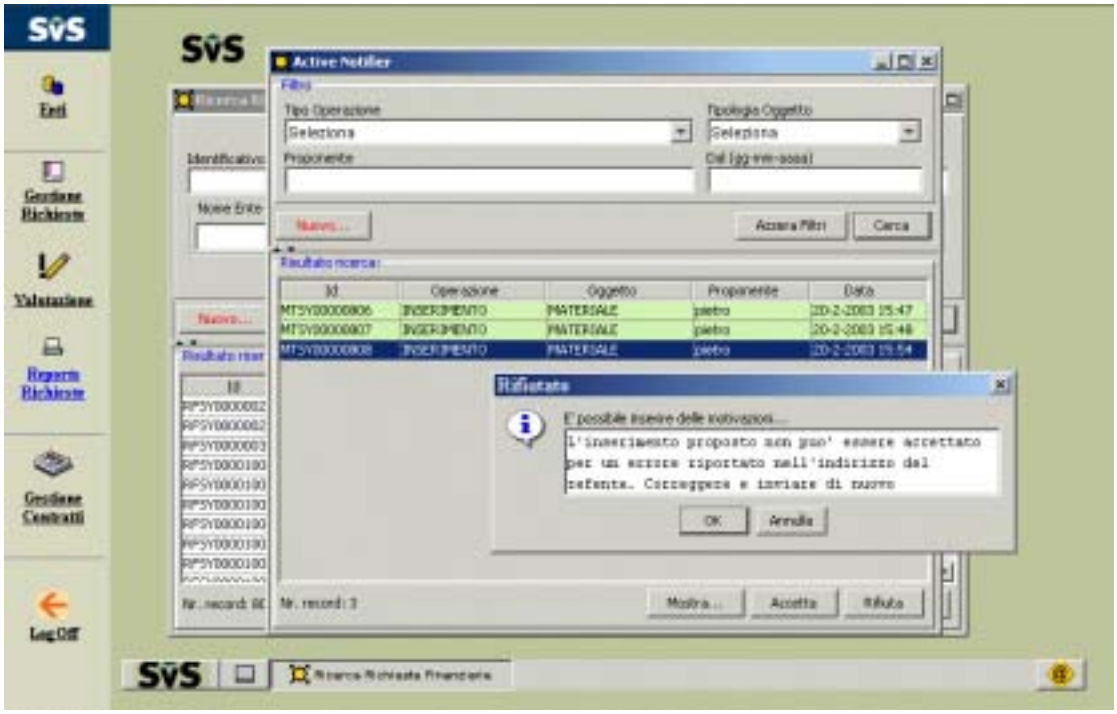


Figura 8 Invio messaggio

Figura 9 Visualizzazione evento

Il redattore, dopo aver controllato i dati (click su **Mostra...**), può decidere di accettare (click su **Accetta**) o rifiutare (click su **Rifiuta**) l'azione proposta; in entrambi i casi egli può decidere di segnalare l'azione all'operatore inviando un messaggio. In Fig. 8 è illustrato un rifiuto motivato del redattore.

L'operatore che ha inserito il materiale, riceverà sulla sua console la notifica sull'esito dell'operazione (Fig. 9). L'azione di rifiuto lascerà immutato il contenuto informativo del sistema annullando l'operazione.

4 Il sottosistema per l'amministrazione.

Il sottosistema per l'amministrazione fornisce strumenti per il trattamento dei dati riservati e contempla alcune funzioni di supporto alle decisioni. Oltre alle funzioni di verifica e controllo dei dati che possono affluire da una qualsiasi postazione Internet (funzione svolta dalla redazione), l'amministratore definisce le modalità di accesso al sistema per i vari utenti.

4.1 Gestione Utenti e Gruppi

La gestione dei soggetti che interagiscono con il sistema SVS (Utenti) e dei loro diritti (ruoli e privilegi) è effettuata tramite una serie di interfacce che possono essere accedute solo dagli utenti con ruolo di *Amministratore*. L'accesso a queste funzionalità avviene utilizzando il protocollo sicuro https (Fig.10).

L'amministratore ha a disposizione due classi di strumenti per:

- Aggiunta e gestione dei **Gruppi** di utenti
- Aggiunta e gestione degli **Utenti**

La console principale per la gestione dei Gruppi (Fig. 11) consente di crearne uno nuovo, oppure di accedere in visualizzazione/modifica/cancellazione di un gruppo esistente. Ad ogni gruppo è associato un nome, una descrizione ed una sigla di 2 caratteri che sarà utilizzata come prefisso degli identificatori interni (chiavi) creati dal sistema per ogni oggetto inserito (Fig. 12). Tramite questo prefisso il sistema è in grado di riconoscere, per ogni utente, l'insieme degli oggetti che egli potrà modificare (cioè solo quelli inseriti da tutti gli utenti appartenenti al suo stesso gruppo).

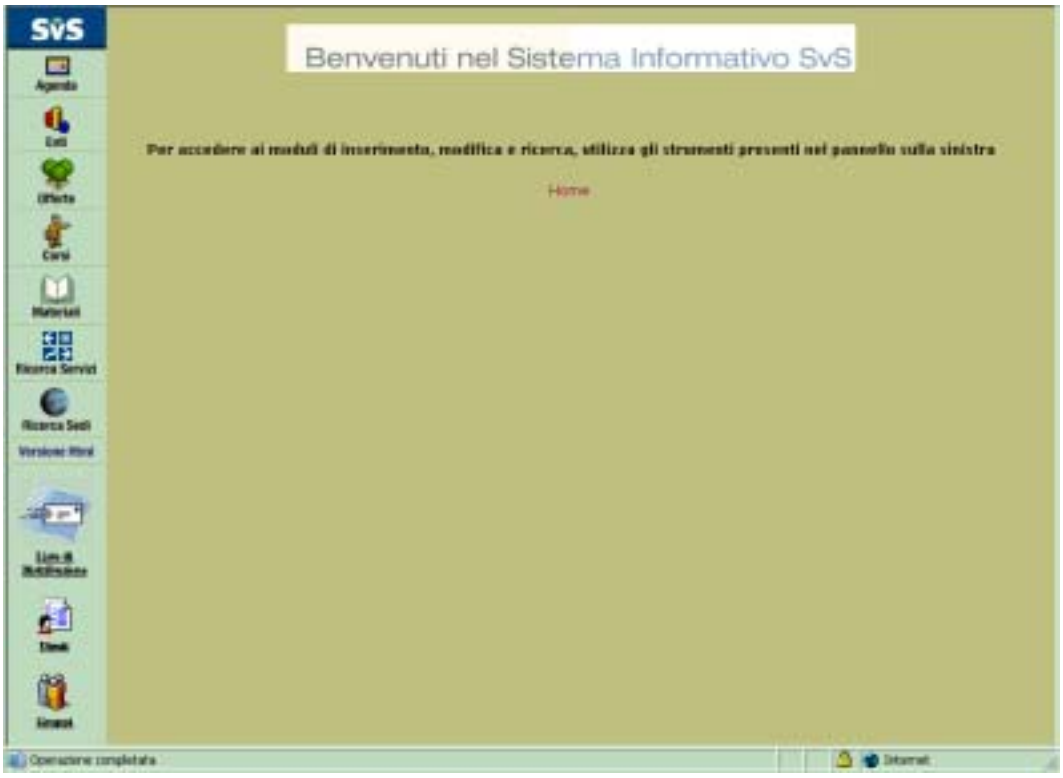


Figure 10 Nuove funzionalità per amministratori

Figure 11 Console di gestione gruppi

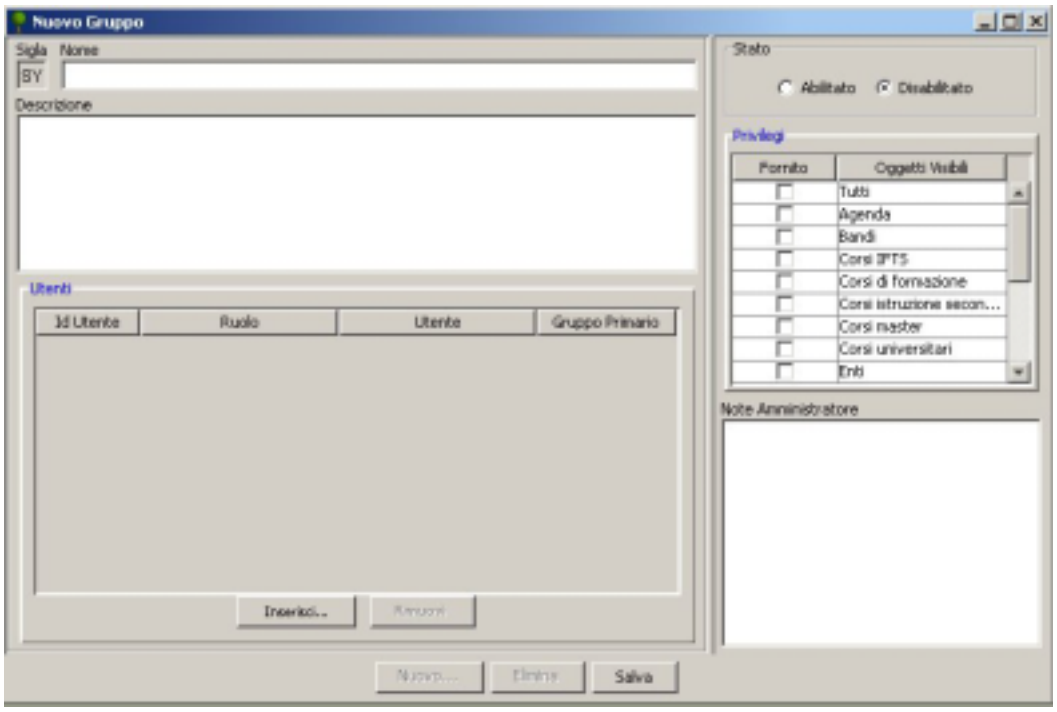


Figure 12 Nuovo gruppo

Ogni gruppo può trovarsi nello stato di *Abilitato* o *Disabilitato*. Con i tasti "*Inserisci*" e "*Rimuovi*" si aggiungono o eliminano utenti dal gruppo (Fig. 13). Per l'inserimento di un utente, viene proposta la finestra di ricerca utenti (Fig 13a), in cui si possono esprimere condizioni di ricerca e selezionare dal risultato ottenuto, l'utente interessato. Dopo che l'utente è stato inserito in un gruppo è necessario assegnargli un ruolo (Fig. 13b). Ricordiamo che un utente può appartenere a più gruppi ed avere ruoli diversi in ognuno di essi. Col *check* su *Gruppo Primario* si specifica per un utente, che per gli oggetti da egli inseriti, sarà utilizzata la sigla del gruppo in questione. Questo meccanismo è necessario per risolvere l'ambiguità derivante da utenti appartenenti a più gruppi e per supportare la politica adottata, cioè di consentire ad ogni utente di poter modificare tutti gli oggetti inseriti da tutti gli utenti appartenenti al suo stesso gruppo (l'alternativa di consentire la modifica di un oggetto solo all'utente che l'ha inserito ci è parsa molto restrittiva e poco pratica in situazioni reali, comunque in caso di necessità l'amministratore può definire un gruppo con un solo utente).

Ciascun utente deve appartenere almeno ad un gruppo. Per ogni gruppo di devono inoltre definire i *Privilegi* sugli oggetti del sistema (Fig. 13). Tramite i privilegi del gruppo ed il meccanismo dei ruoli si può determinare per ogni utente gli oggetti cui ha accesso e con quali modalità.

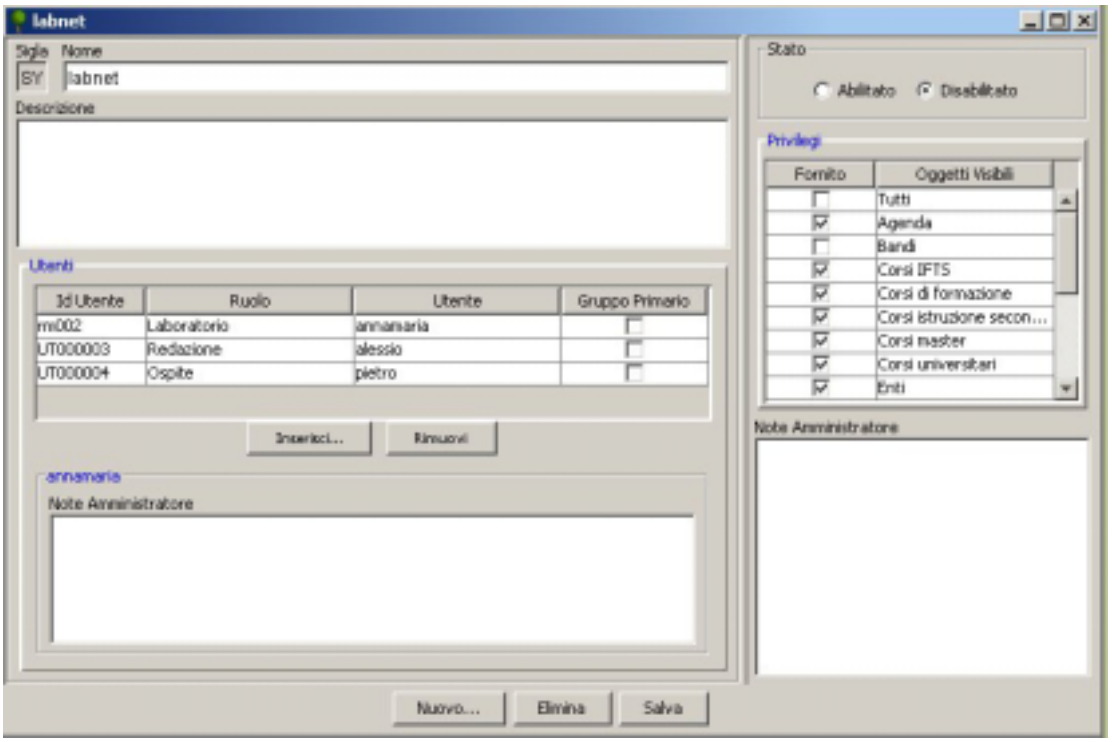


Figure 13 Modifica gruppo

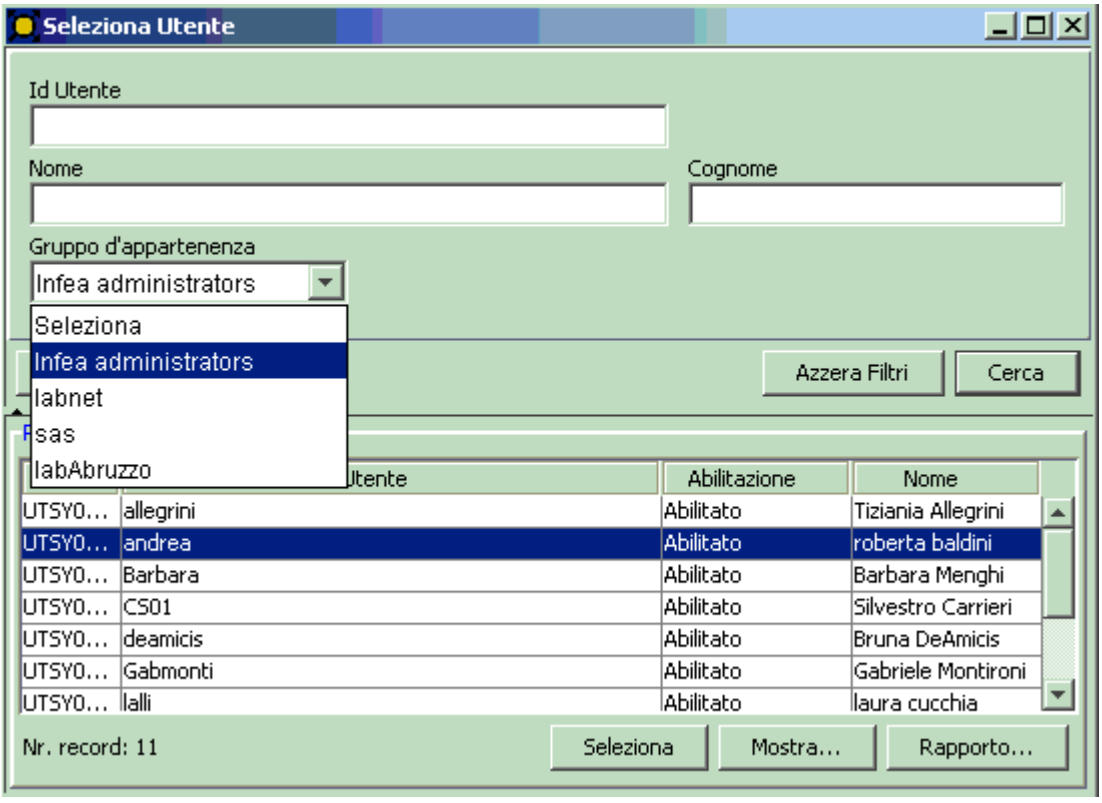


Figure 13a Ricerca e selezione utente

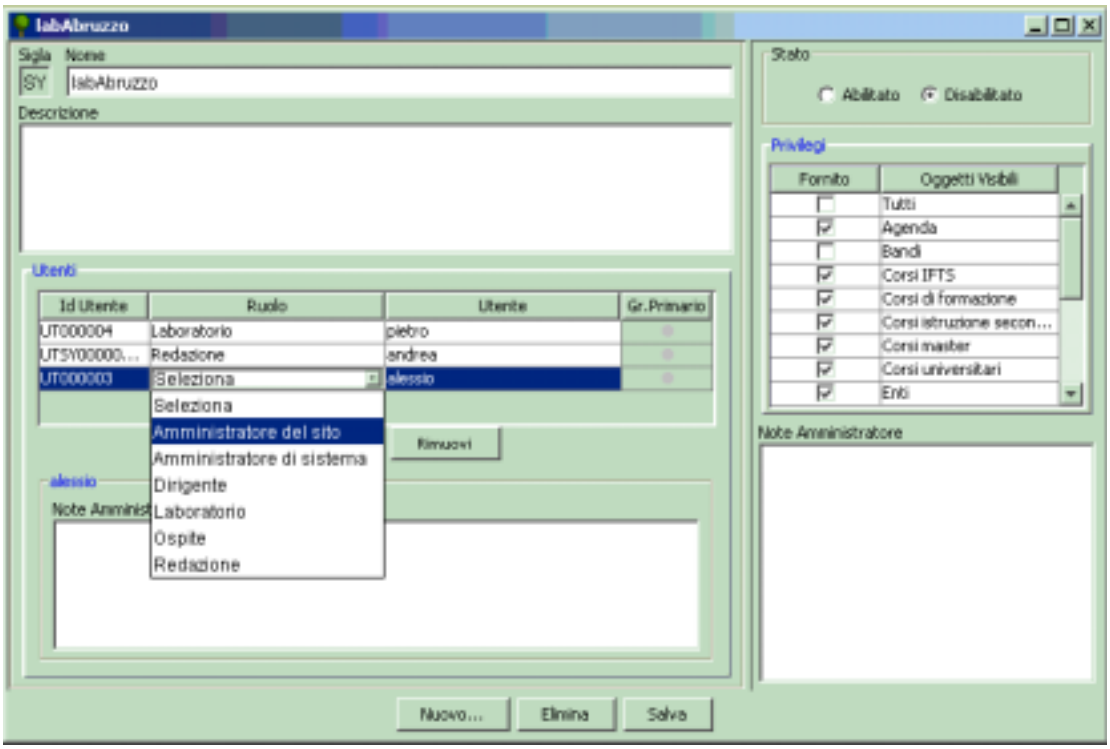


Figure 13b Assegnazione del ruolo di utente

La console di gestione utenti (Fig. 14) consente di inserire un nuovo utente (Fig. 15) o di modificare le caratteristiche di uno esistente (Fig. 16) selezionandolo dalla lista ottenuta come risultato di una ricerca con opportune condizioni. Per ogni utente si fornisce un identificativo univoco e i dati anagrafici: nome, cognome, indirizzo di e-mail (utile per generare liste di distribuzione, cap. 7) e l'eventuale ente di appartenenza, selezionato dall'elenco presente nel sistema. Ad ogni utente si può associare una lista di gruppi di appartenenza (almeno uno) e tra questi si deve scegliere quello primario, cioè quello con cui l'utente opererà in fase di inserimento/modifica di oggetti del sistema. Nella definizione dell'utente sono previsti due campi testuali: *Descrizione* e *Note Amministratore* di cui il primo deve essere compilato dall'utente stesso in fase di registrazione (comparirà la stessa finestra di Fig. 15 con il bottone di inserimento gruppi e le *liste a scomparsa* disabilitati, se la registrazione viene effettuata da un utente che non ha il ruolo di amministratore), ed il secondo ad uso interno dell'amministratore di sistema.

L'inserimento dei gruppi di appartenenza, per ogni utente, avviene scegliendoli tra quelli già definiti tramite il modulo di ricerca gruppi (Fig.11).

Figure 14 Console di gestione utenti

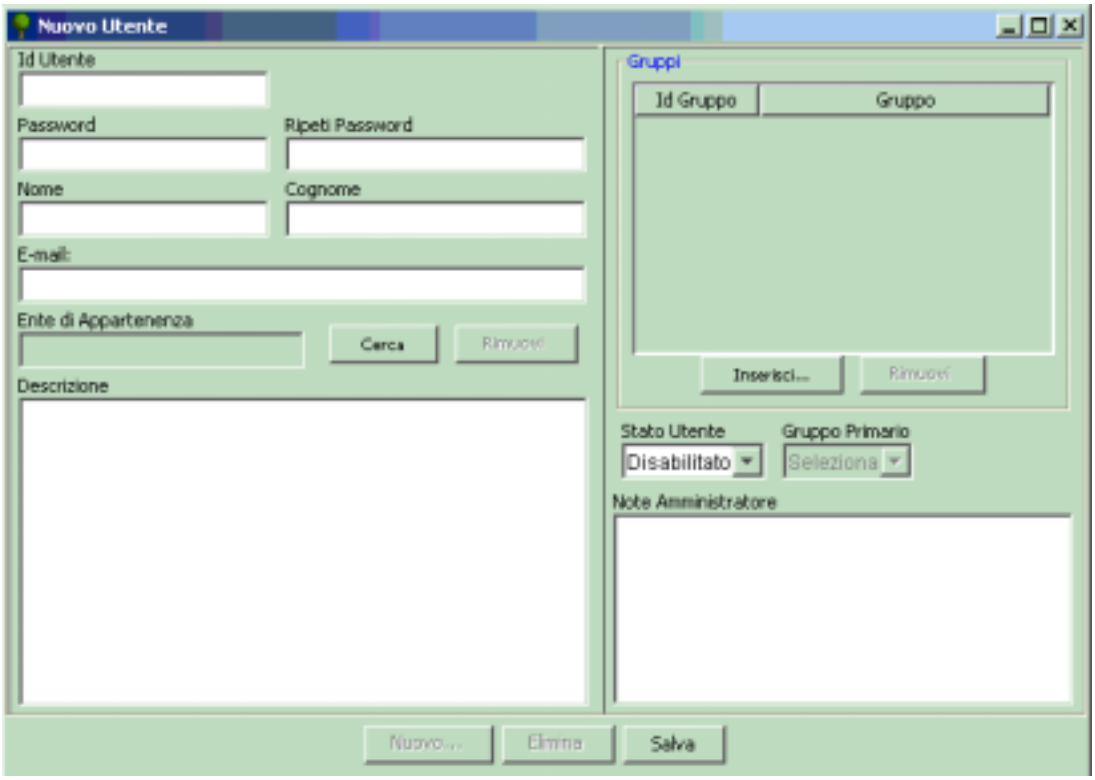


Figure 15 Nuovo utente

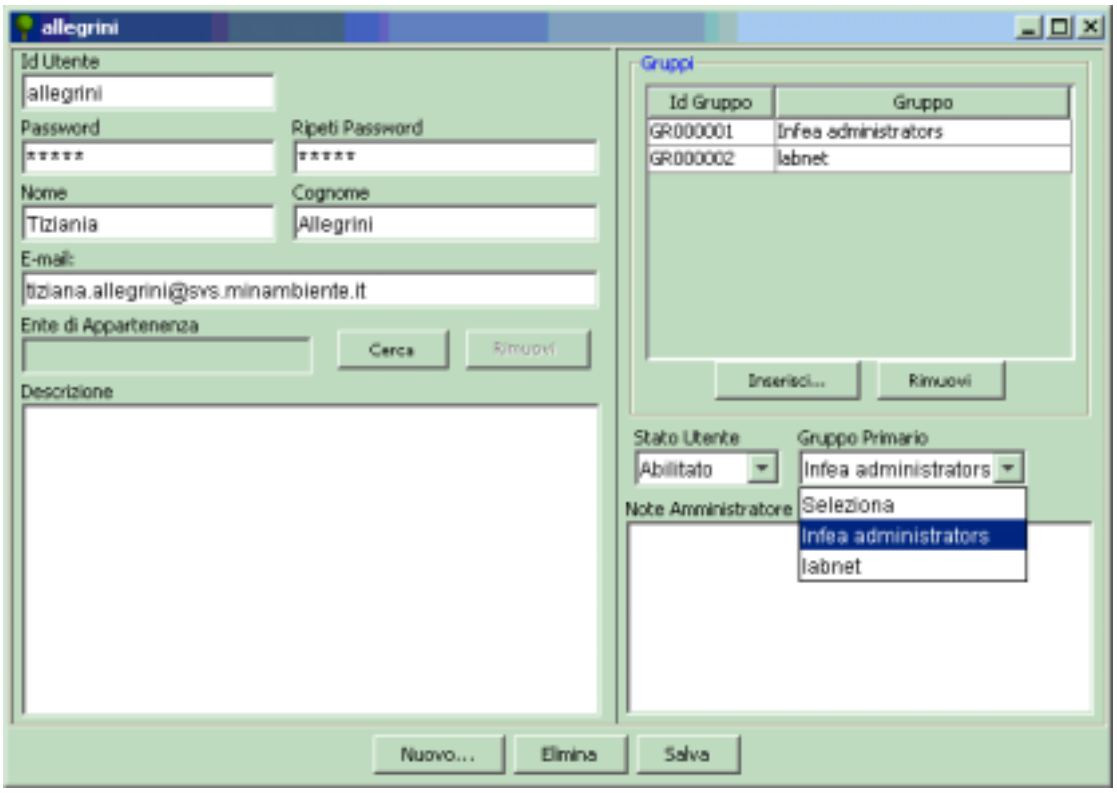


Figure 16 Modifica dati utente

4.2 Definizione Bandi

L'idea su cui si basava questa attività era di progettare e realizzare un sistema generalizzato per la gestione di bandi, per il finanziamento delle attività di pertinenza del Ministero. L'attività non è stata completata principalmente per i due seguenti motivi:

1) non è stato possibile allo stato attuale delle nostre conoscenze individuare le caratteristiche fondamentali delle azioni di finanziamento del Ministero, soprattutto per la mancanza di procedimenti standardizzati, in quanto la maggior parte delle azioni fanno riferimento a leggi, decreti, ecc. promulgati ad hoc.
2) la necessità del Ministero di ottenere uno strumento completo per la gestione delle azioni relative alle *Agende 21 locali* (vedi. Cap. 5), ha convogliato su quest'ultima attività buona parte delle risorse disponibili.

Di seguito presentiamo un prototipo che abbiamo realizzato, prefigurando un iter procedurale più o meno standard, che potrebbe essere la base per ulteriori approfondimenti ed eventuali realizzazioni.

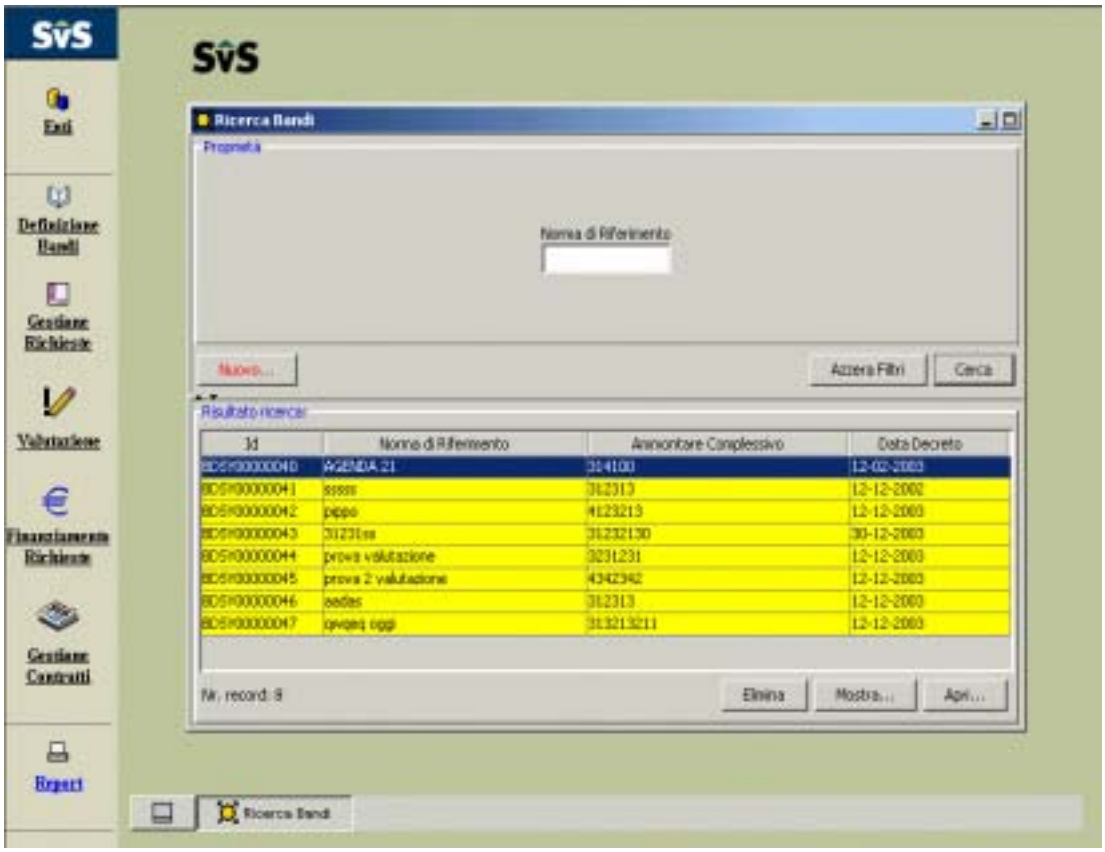


Fig. 17 Modulo per la ricerca e definizione dei bandi

Dalla barra degli strumenti del desktop dell'amministratore, si può attivare il modulo *Definizione Bandi* (Fig. 17). È possibile visualizzare un elenco di bandi, oppure ricercarne uno specifico per "Norma di riferimento" (cioè la legge o decreto, ecc su cui si basa la definizione del bando). La selezione di un bando dall'elenco o l'attivazione del bottone "Nuovo..." apre una finestra per l'inserimento/modifica dei dati; in Fig. 18 è riportato un esempio di definizione di bando che fa riferimento alle specifiche di *Agende 21 locali*.

In figura 19 è visualizzata la definizione di criteri di valutazione (l'esempio è sempre relativo ad Agende 21), si possono aggiungere "Categorie di valutazione" (A,B,C nell'esempio) e per ognuna di esse si possono aggiungere gli "Item di valutazione" (A1..An, B1,..Bn,…), con la definizione del relativo punteggio.

In figura 20 è visualizzata la finestra per la definizione delle attività ("Categorie di intervento") ammesse a finanziamento per il bando in questione.

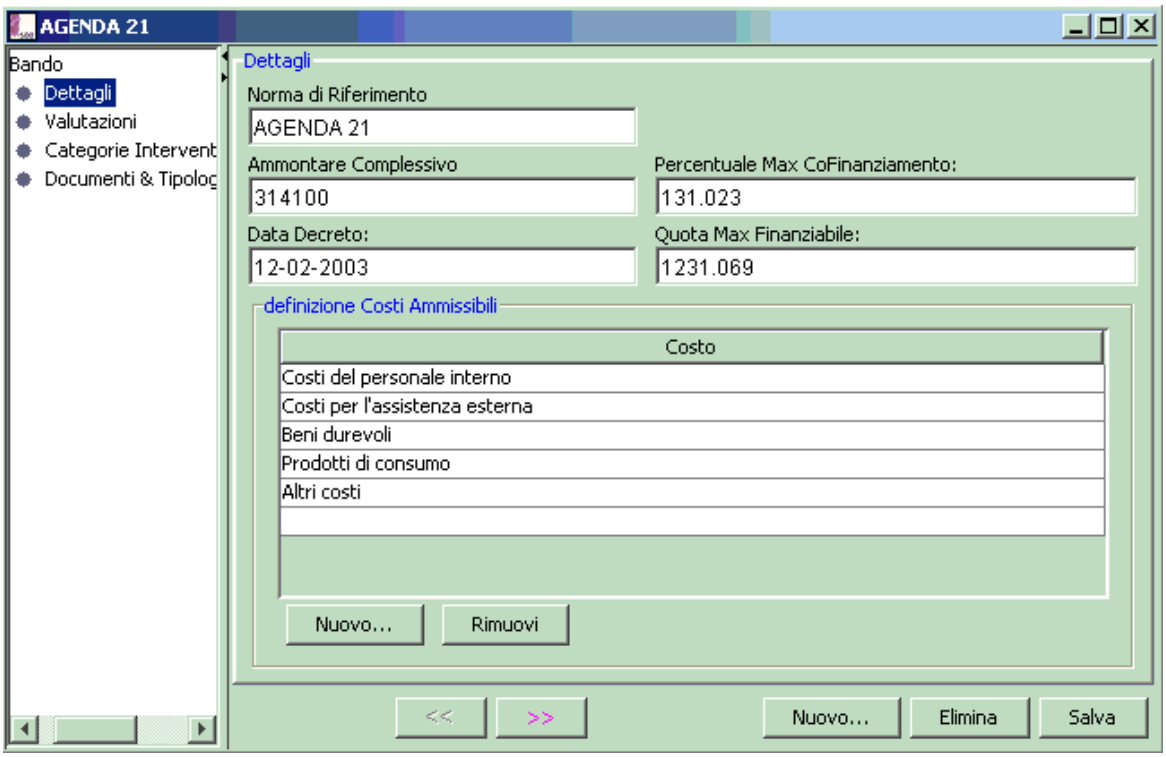


Fig. 18 Frame di inserimento bando

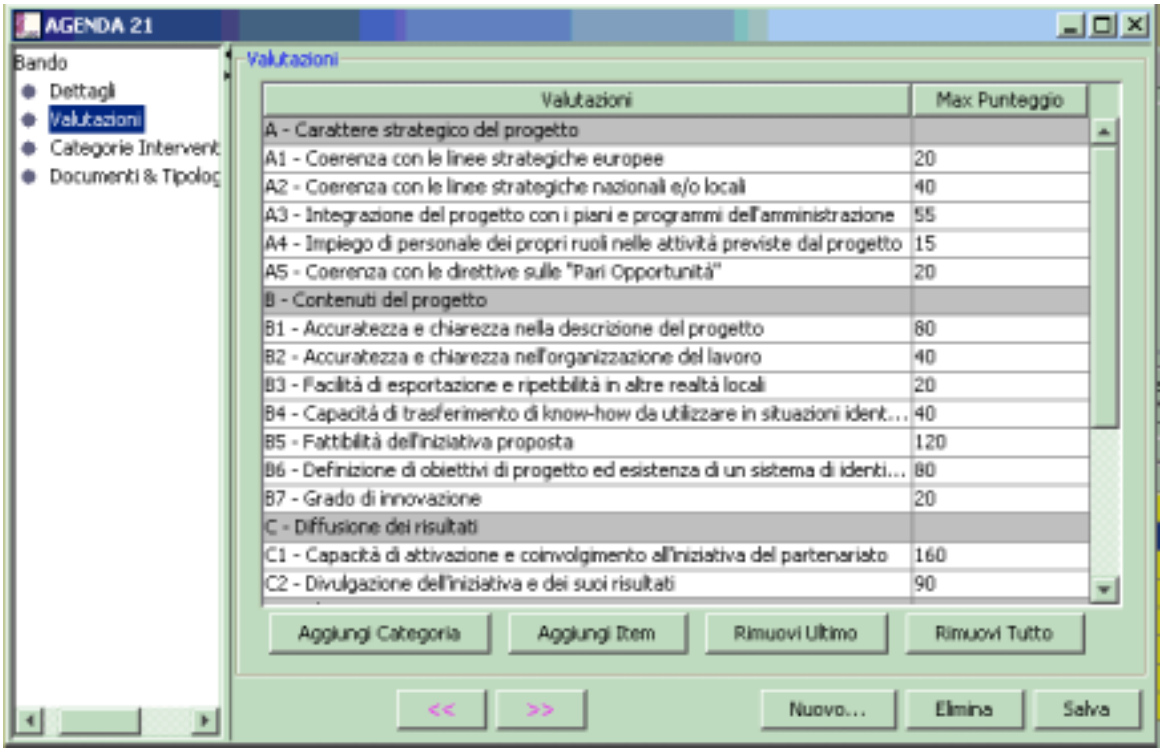


Fig. 19 Frame di inserimento bando

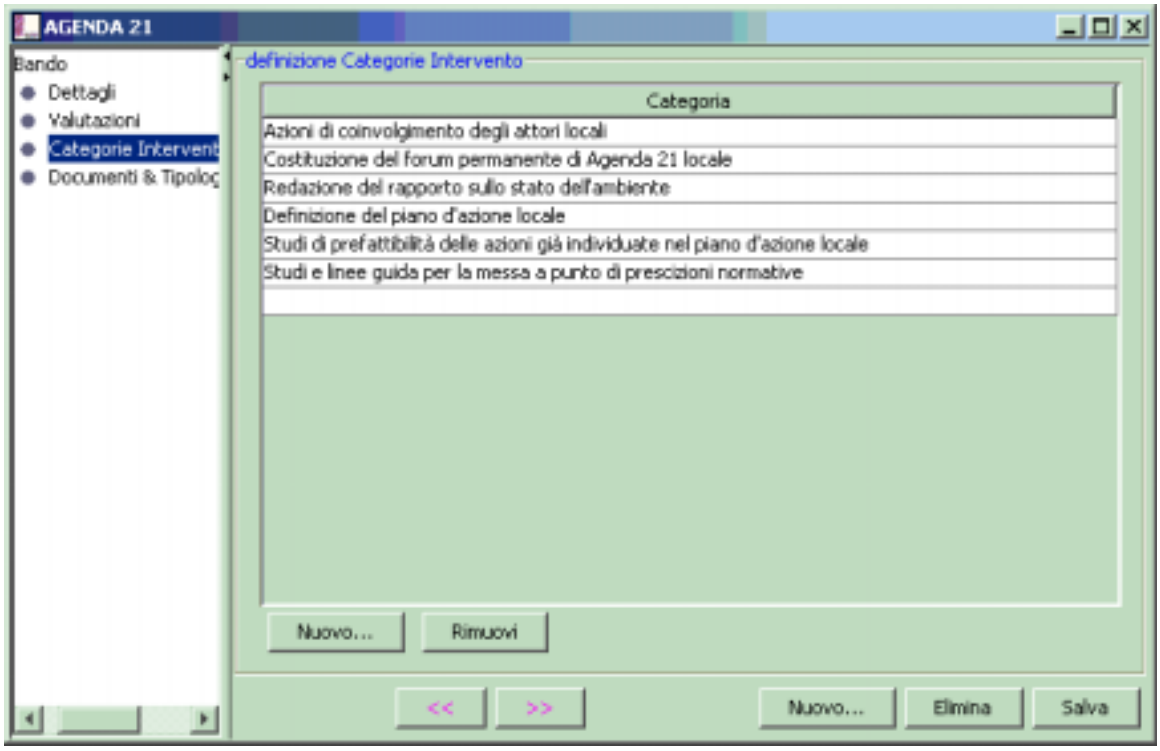


Fig. 20 Frame di inserimento bando

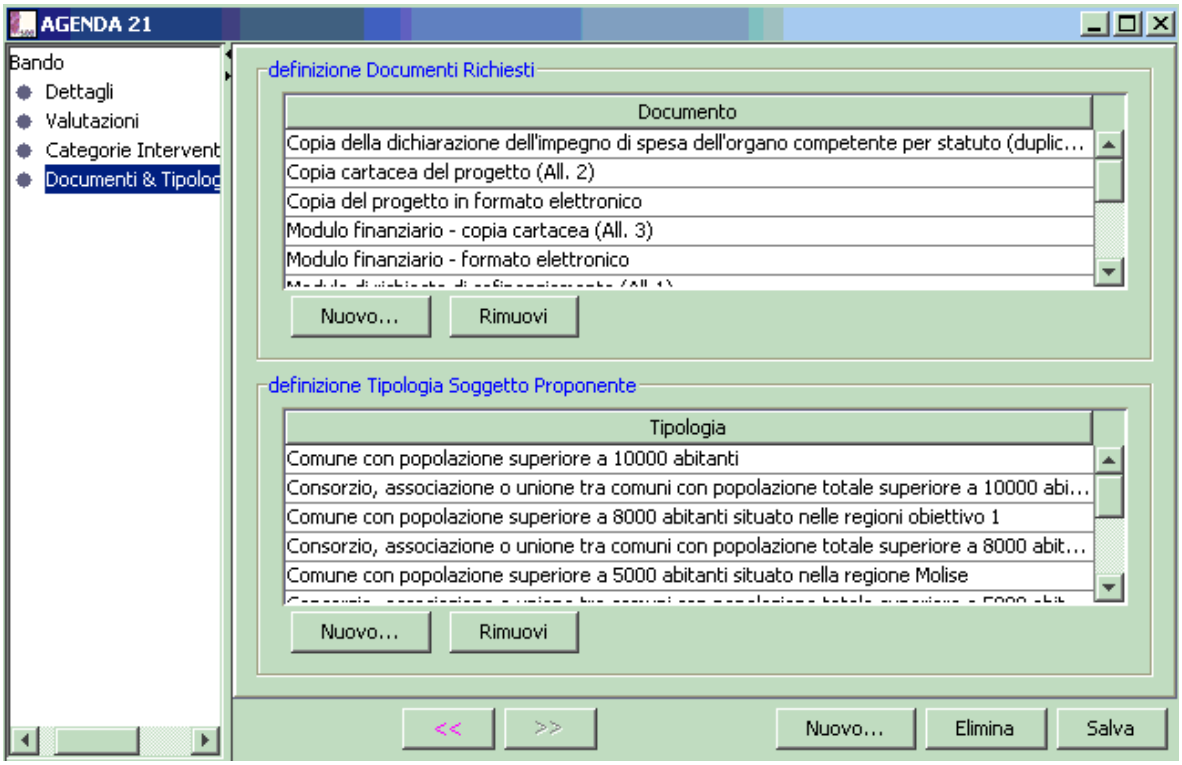


Fig. 21 Frame di inserimento bando

In figura 21 è visualizzata la finestra per la definizione dei documenti da allegare alla richiesta di finanziamento e le tipologie dei soggetti ammessi a formulare proposte.

Come si può notare dall'esempio precedente, le conoscenze acquisite nello studio dei finanziamenti relativi ad "*Agende 21 locali*", ha influenzato notevolmente la definizione del "Gestore dei bandi" e la realizzazione del prototipo presentato. Con questa attività abbiamo dimostrato che dato un sistema che permetta di definire un bando, con le caratteristiche esemplificate (il caso di Agende 21), è possibile generare l'insieme delle interfacce utente che costituiscono l'iter di una determinata gestione finanziaria: richiesta finanziamento, valutazione, finanziamento, gestione pagamenti; nel successivo capitolo 5 viene presentata nei dettagli l'interfaccia che è possibile ottenere utilizzando uno strumento come quello che abbiamo proposto in forma prototipale.

4.3 Funzionalità di supporto alle decisioni

Ciò che abbiamo realizzato con questo modulo non è un sistema di supporto alla decisioni, che non era nei piani delle attività contemplate dal contratto di collaborazione, ma un insieme di strumenti che possono essere proficuamente utilizzati per l'eventuale integrazione con sistemi più complessi. Il sistema che abbiamo studiato e progettato consente di estrarre informazioni dalla base di dati esprimendo condizioni di ricerca sofisticate e di trasformare il risultato in formati utilizzabili da altri strumenti software. Un'applicazione di questa attività è costituito dal generatore di report per Agende21 locali, che al di là dall'essere uno strumento generalizzato per la generazione di report (in questa versione infatti consente di generare report solo su richieste predefinite), tuttavia ingloba tecnologia (es. il software per la trasformazione in formato Excel o PDF, ecc.) che può essere proficuamente riutilizzata per eventuali più complete e complesse soluzioni.

5 Il sistema per la gestione finanziaria

Il sottosistema per la gestione finanziaria comprende vari strumenti che consentono di gestire diversi tipi di contratti del Ministero dell'Ambiente e della Tutela del Territorio: dalle richieste di finanziamento, ai decreti, alle convenzioni. Tra questi, la gestione delle richieste finanziarie risulta la

più complessa, per tale motivo ad essa è stata riservata la maggior parte delle risorse: agli utenti della redazione viene permesso di inserire una richiesta finanziaria corrispondente ad un bando particolare e di sottoporla alle varie procedure di valutazione e selezione.
I dati derivanti da questa attività sono integrati nell'unica base di dati di supporto al sistema informativo, che viene così arricchito con nuove possibilità di indagini (è per esempio possibile elencare i materiali prodotti da enti che figurano come proponenti di richieste di finanziamento per Agende 21 locali, ecc).

Il desktop della redazione (Fig. 22) comprende quindi i moduli che consentono la gestione delle richieste finanziarie (*Gestione delle richieste*), la loro valutazione (*Valutazione*), il loro finanziamento (*Finanziamento richieste*), e la gestione dei Contratti, Decreti, Convenzioni e progetti finanziati (*Gestione contratti*).

Fig. 22 Desktop Redazione

Attualmente le richieste finanziarie che possono essere inserite e gestite sono quelle relative al Bando del 2002 "Agende 21". In seguito illustreremo passo per passo la gestione di una richiesta di finanziamento.

5.1 Inserimento di una richiesta

Il modulo Gestione delle Richieste (Fig. 23) consente di ricercare, inserire e modificare una richiesta finanziaria.

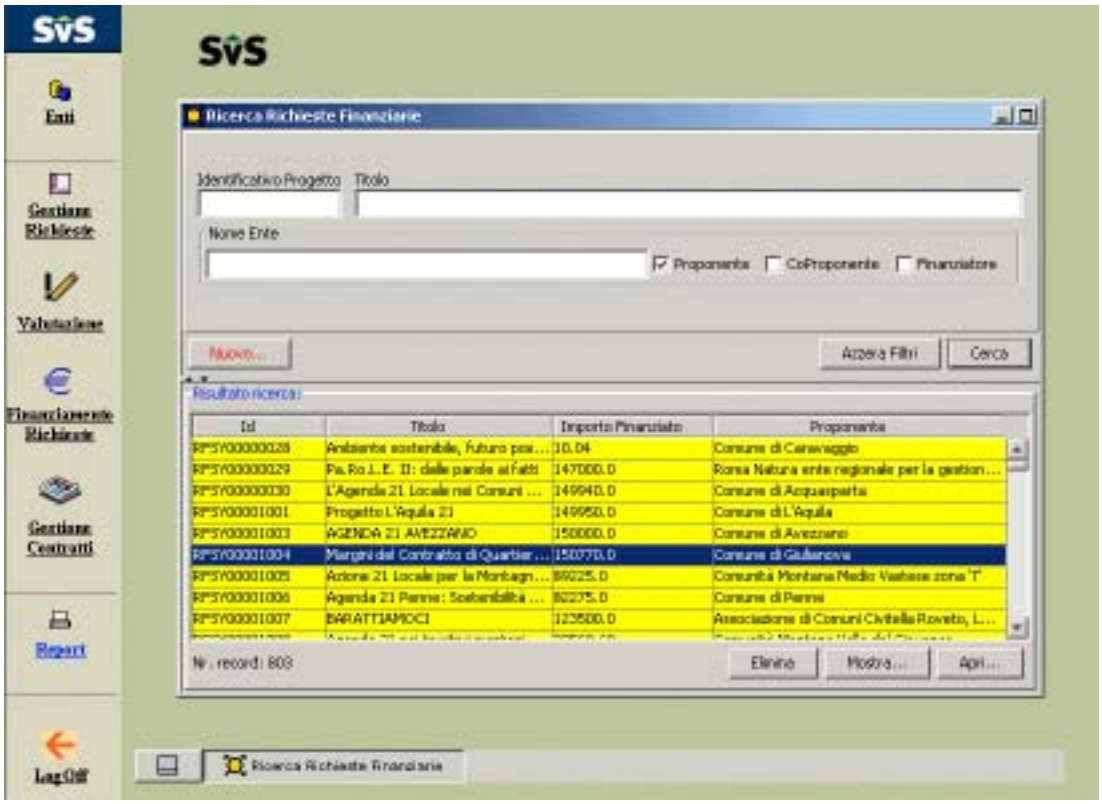


Fig. 23 Modulo Gestione Richieste

Premendo il bottone “Nuovo...” viene aperto un frame utilizzato per l’inserimento dei dati, (Fig. 24). Si richiede che vengano inseriti informazioni descrittive della richiesta (titolo, acronimo, protocollo, identificativo, data invio), informazioni di tipo finanziario (importo richiesto, costo totale, ripartizione dei costi), informazioni sugli enti coinvolti (ente proponente, ente capofila, enti coproponenti e cofinanziatori), informazioni sui vari referenti, informazioni sulle tipologie di intervento e informazioni sui documenti allegati.

Come si può notare dalla Fig. 24 la richiesta finanziaria che viene inserita si riferisce al Bando “Agenda 21”.

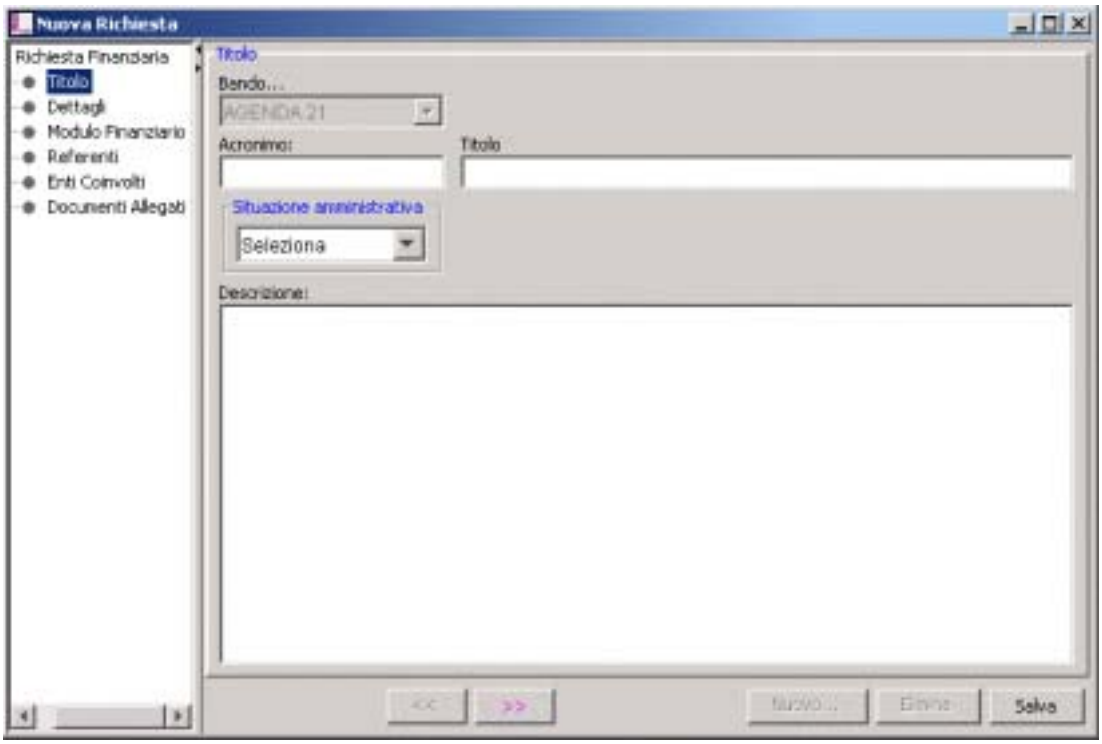


Fig. 24 Frame inserimento richiesta

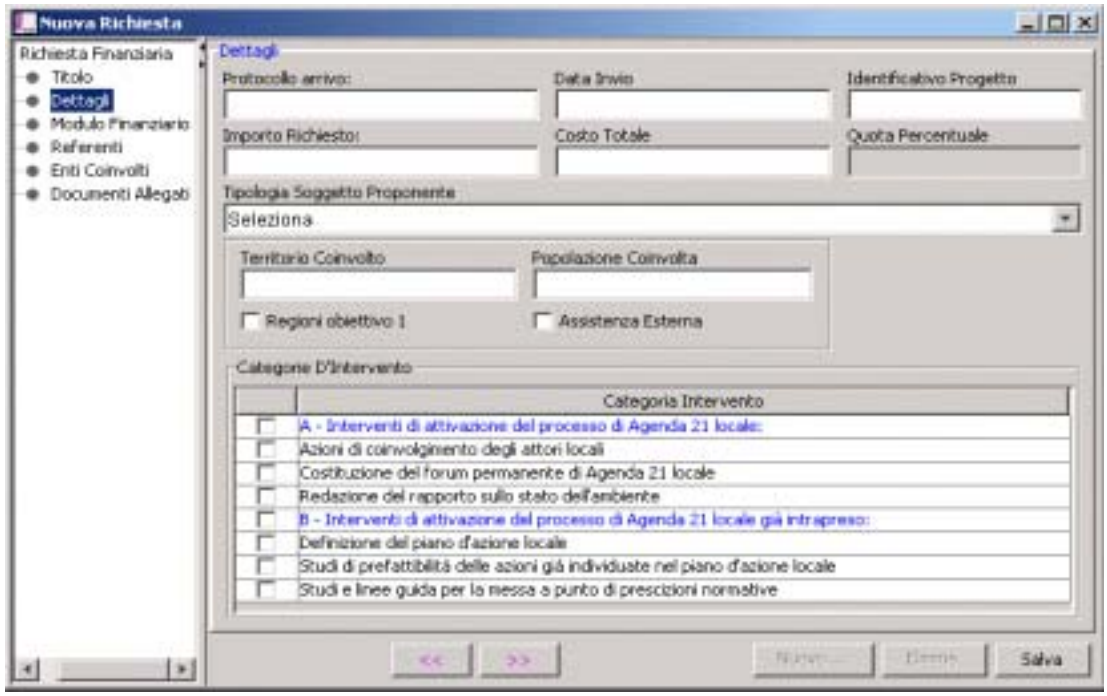


Fig. 24a Un dettaglio del frame di inserimento

La Fig. 24a mostra il secondo pannello di inserimento (Dettagli); da notare il menù a tendina che specifica la tipologia del soggetto proponente e la tabella per la scelta delle categorie di intervento: in base a quello che viene inserito dipenderà l'accettazione o l'esclusione della proposta.

Nella Fig. 24b possiamo vedere il pannello dedicato all'inserimento dei costi della richiesta.

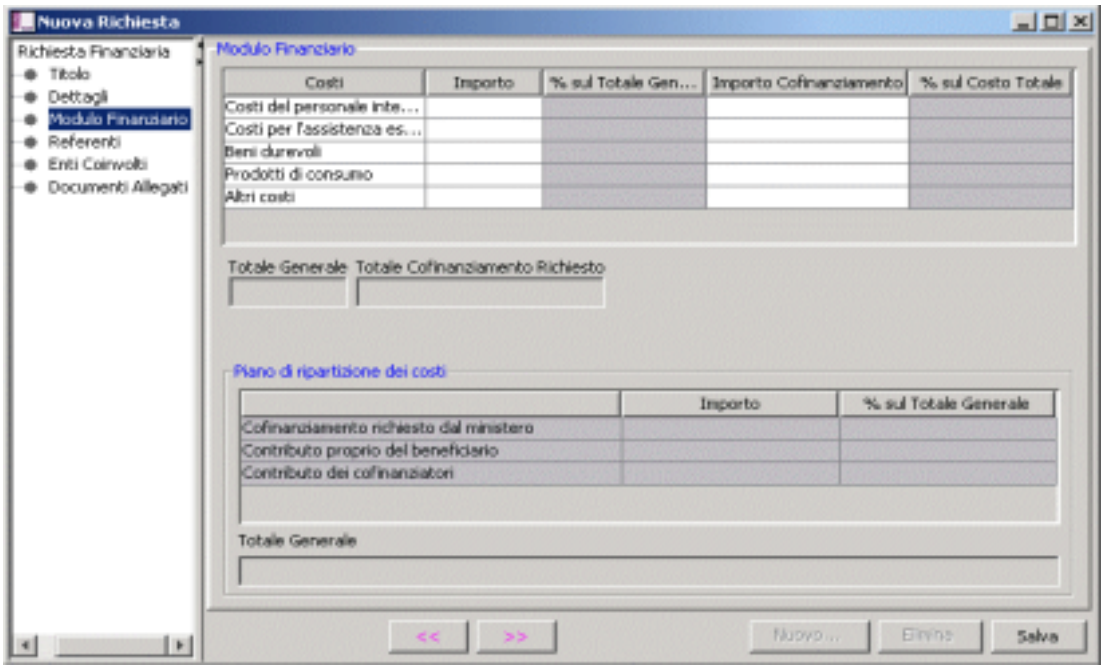


Fig. 24b Un dettaglio del frame di inserimento

Nei campi "Totale generale" e "Totale cofinanziamento richiesto" vengono caricati rispettivamente i valori inseriti nei campi "Costo totale" e "Importo richiesto" in Fig. 24a. Nella tabella Costi, vengono riportati, il "Costo totale" (colonna Importo) e l'"Importo richiesto" (colonna Importo Cofinanziamento) ripartiti per macrovoci di spesa. Nella tabella "Ripartizione Costi" viene mostrato come i costi sono ripartiti tra il Ministero, il beneficiario e gli eventuali cofinanziatori. L'importo del "cofinanziamento richiesto dal Ministero" corrisponde all'"Importo richiesto", il "contributo proprio del beneficiario" corrisponde alla differenza tra il "costo totale", l'"importo richiesto" e la somma delle quote finanziate dai cofinanziatori, valore quest'ultimo riportato anche nella voce "contributo dei cofinanziatori".

Le quote finanziate dai cofinanziatori vengono inserite nel pannello Enti coinvolti mostrato in Fig. 24c.

Infine uno sguardo al pannello (Fig. 24d) Documenti Allegati: la mancanza di alcuni documenti potrà determinare l'esclusione della richiesta..

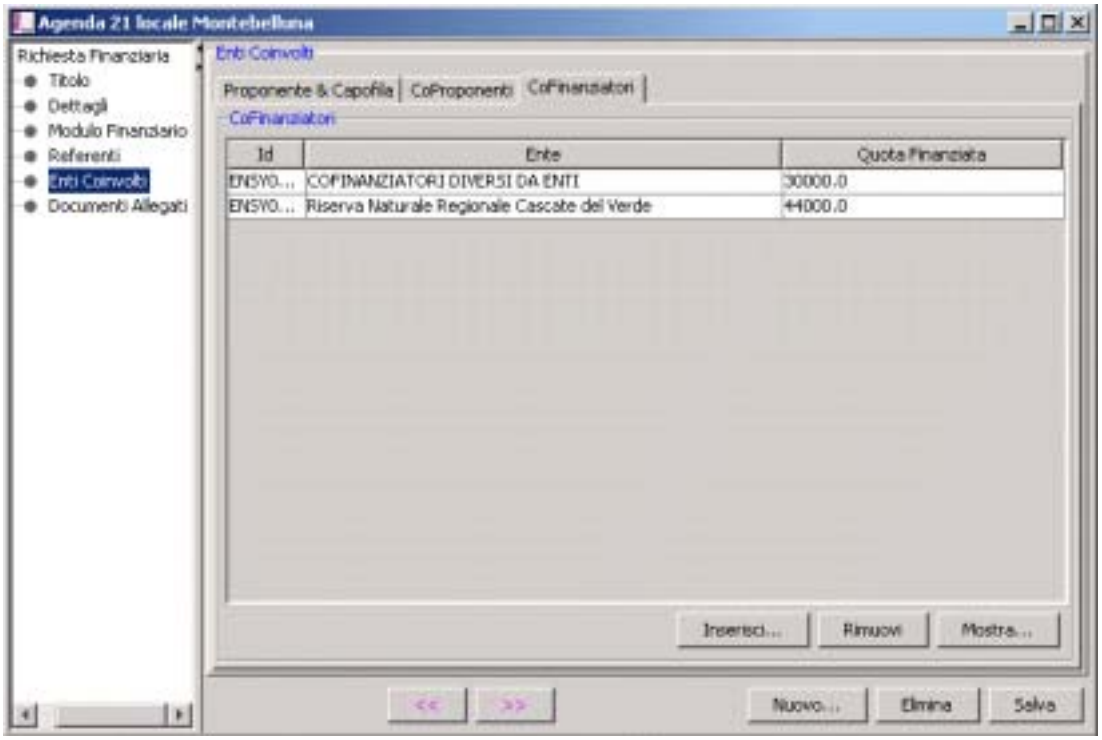


Fig 24c Un dettaglio del frame di inserimento

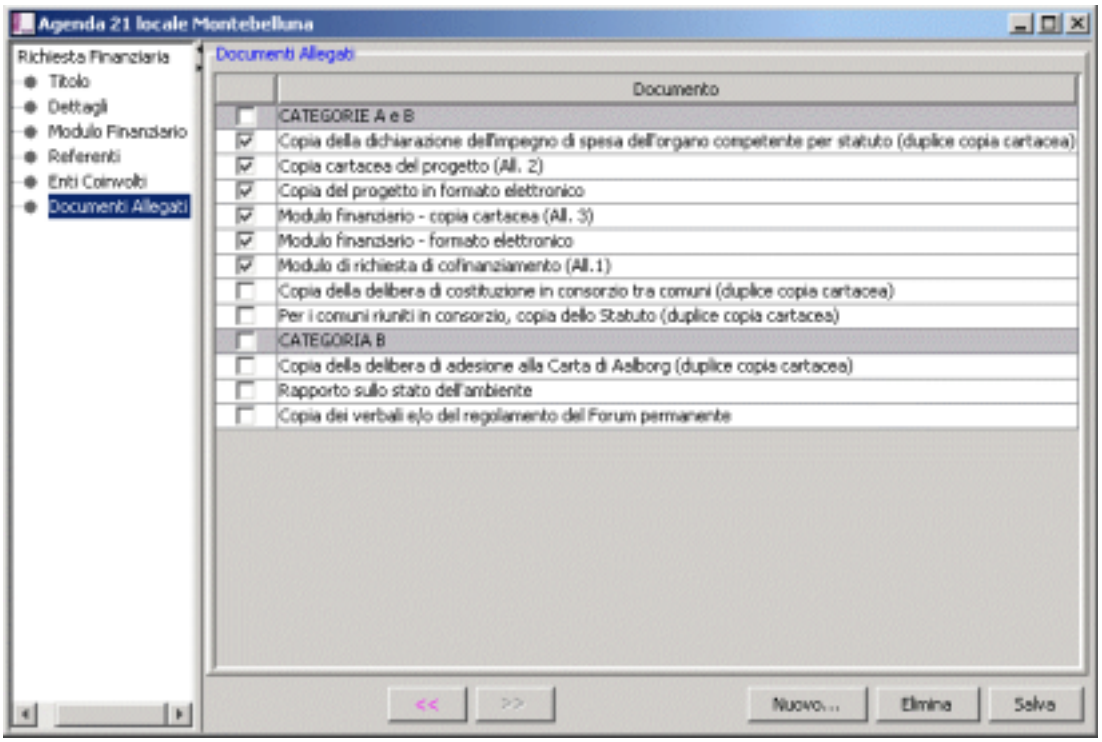


Fig. 24d Un dettaglio del frame di inserimento

5.2 Valutazione e selezione

Una volta che la richiesta di finanziamento è stata inserita viene sottoposta a verifica della conformità e della completezza della documentazione richiesta.

La valutazione delle richieste si articola in 4 fasi successive:

- Ammissibilità amministrativa;
- Valutazione secondo i criteri di selezione individuati;
- Valutazione di finanziamento;
- Assegnazione del cofinanziamento.

5.2.1 Fase I: ammissibilità amministrativa

I criteri per determinare l'ammissibilità delle richieste di finanziamento relative al bando Agenda 21 Locale sono stabiliti a priori. Il rispetto dei criteri da parte di richiesta ne determina l'ammissibilità, in questo caso viene assegnato lo stato di "ammissibile".
Il campo che permette di definire una proposta finanziaria ammissibile o non ammissibile si trova nel modulo di inserimento/modifica dati illustrato nella sezione precedente e di seguito riportato.

Solo i progetti che verranno valutati ammissibili passeranno alla fase successiva dove saranno sottoposti alla valutazione secondo dei criteri di selezione.

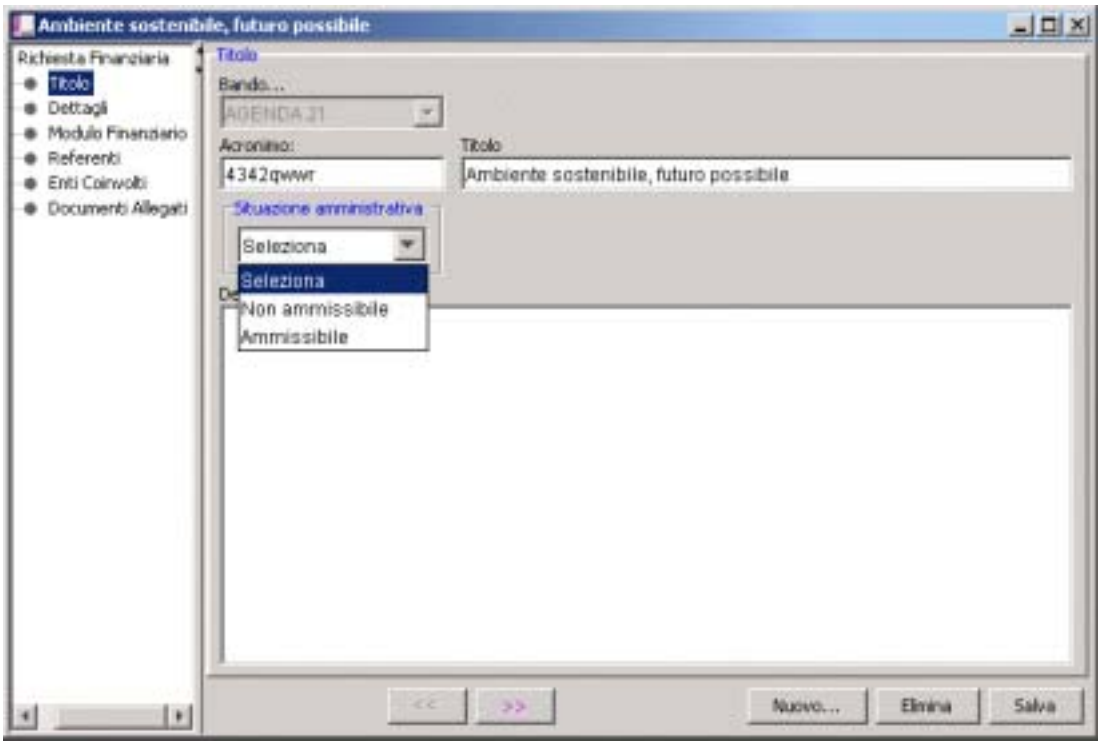


Fig. 25 Campo "situazione amministrativa"

5.2.2 Fase II: valutazione in base ai criteri di selezione

Durante questa fase ad ogni progetto ammissibile che soddisfa dei criteri prestabiliti verrà assegnato un punteggio di merito. Solo i progetti valutati positivamente, che cioè raggiungono o superano un certo punteggio complessivo, verranno considerati *idonei*.
L'interfaccia utente per l'assegnazione dei punti e per la valutazione della proposta in base al punteggio totale ottenuto è il modulo *Valutazione*, illustrato in Fig. 26. Il modulo *Valutazione* consente di ricercare tutte le richieste finanziarie inserite, indipendentemente dallo stato di ammissibilità, ma solo a quelle che sono risultate ammissibili viene data la possibilità di avanzare nel processo di valutazione: infatti nel pannello di inserimento esiste un checkbox chiamato "idoneo" che viene attivato solo per quelle richieste finanziarie che sono state ritenute ammissibili.

Le Fig. 27 e 27a mostrano rispettivamente come appare, in fase di valutazione, una richiesta finanziaria non ammissibile e una richiesta finanziaria ammissibile. Solo quest'ultima può procedere nel processo di valutazione in quanto solo per essa viene data la possibilità di cambiare la situazione amministrativa in *idonea*.

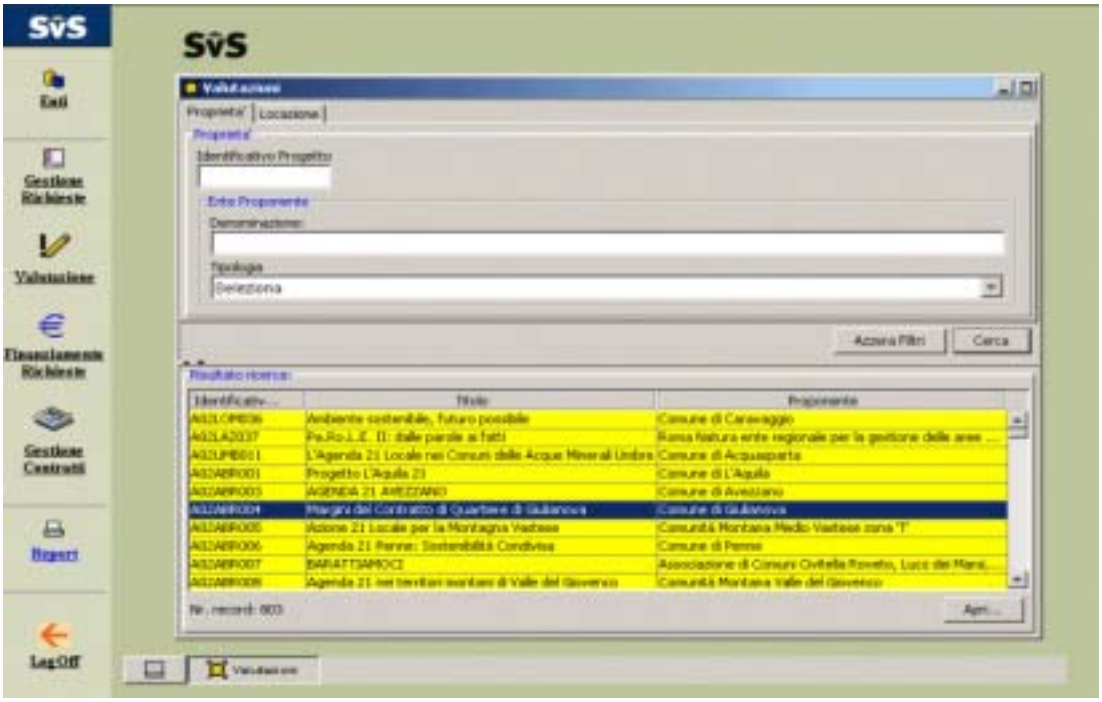


Fig. 26 Modulo Valutazioni

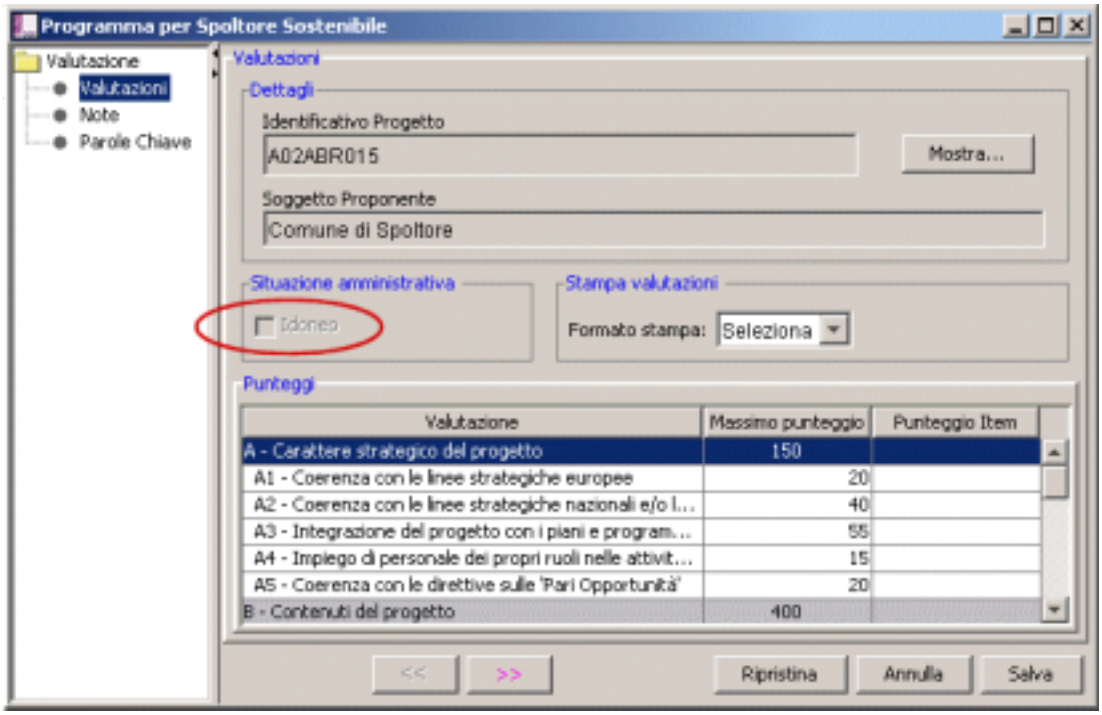


Fig. 27 Richiesta non ammissibile: impossibile procedere nella valutazione

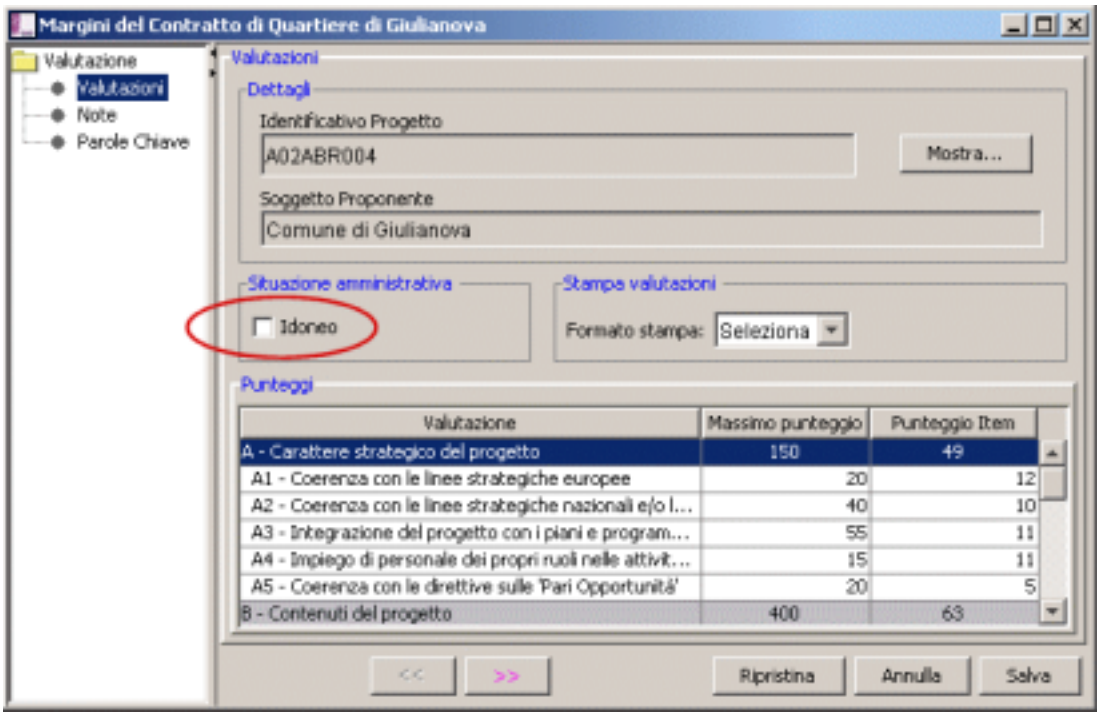


Fig. 27a Richiesta ammissibile: possibilità di procedere nella valutazione

La figura precedente mostra, come già accennato, il pannello preposto per assegnare i punteggi e la valutazione di idoneità ad una richiesta finanziaria ammissibile, di cui possiamo vedere nella sezione "Dettagli" alcune informazioni ("Identificativo" e "Soggetto porponente"). Per ulteriori dati circa la proposta in questione viene messo a disposizione il bottone "Mostra…". In basso, nella tabella "Punteggi" ad ogni riga corrispondente ad un determinato criterio (dipendente dal bando a cui la proposta di finanziamento si riferisce) può essere assegnato un punteggio, il cui valore massimo è mostrato nella colonna "Massimo punteggio". Viene data inoltre la possibilità di stampare la tabella "Valutazioni" in formato Pdf o Excel (menù a tendina "Formato stampa") come mostrato in figura 28, di aggiungere delle motivazioni ai punteggi assegnati (pannello Note) e di associare a tale progetto delle parole chiavi, attraverso il gestore del Thesaurus del sistema informativo (Fig. 27b).

Una volta inseriti tutti i punteggi il progetto può essere valutato *idoneo* o *non idoneo* selezionando o lasciando in bianco il campo "Idoneo" nella sezione "Situazione amministrativa". Solo i progetti che verranno resi idonei proseguiranno nel processo di valutazione.

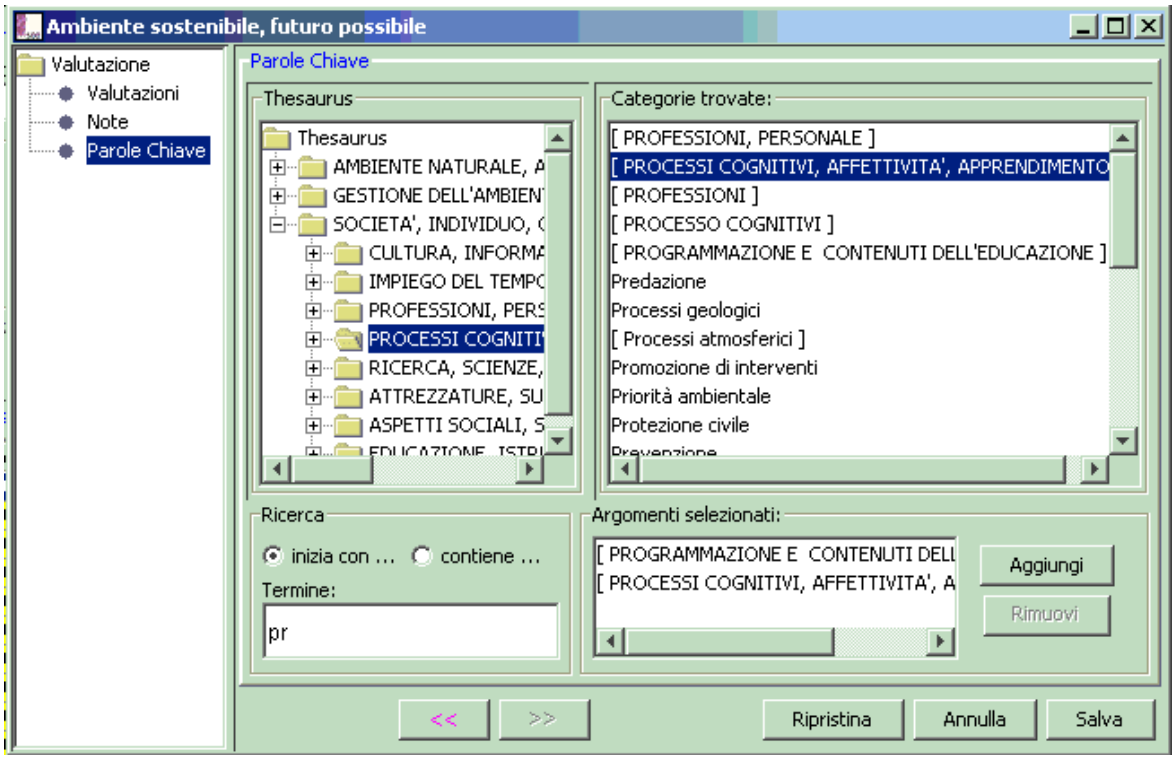


Fig. 27b Assegnazione di parole chiavi

(A02ABR004)

Comune di Giulianova
Margini del Contratto di Quartiere di Giulianova - MCaG

B - Interventi di attivazione del processo di Agenda 21 locale gia' intrapreso:
- Azioni di coinvolgimento degli attori locali
- Costituzione del forum permanente di Agenda 21
- Redazione del rapporto sullo stato dell'ambiente

	Criteri di Valutazione	Massimo punteggio	Punteggio attribuito
A	Carattere strategico del progetto	150	49
A1	Coerenza con le linee strategiche europee	20	12
A2	Coerenza con le linee strategiche nazionali e/o locali	40	10
A3	Integrazione del progetto con i piani e programmi dell'amministrazione	55	11
A4	Impiego di personale dei propri ruoli nelle attività previste dal progetto	15	11
A5	Coerenza con le direttive sulle Pari Opportunita'	20	5
B	Contenuti del progetto	400	63
B1	Accuratezza e chiarezza nella descrizione del progetto	80	6
B2	Accuratezza e chiarezza nell'organizzazione del lavoro	40	7
B3	Facilità di esportazione e ripetibilità in altre realtà locali	20	8
B4	Capacità di trasferimento di know-how da utilizzare in situazioni identiche e simili	40	9
B5	Fattibilità dell'iniziativa proposta	120	10
B6	Definizione di obiettivi di progetto ed esistenza di un sistema di identificatori di performance	80	11
B7	Grado di innovazione	20	12
C	Diffusione dei risultati	250	27
C1	Capacità di attivazione e coinvolgimento all'iniziativa del partenariato	180	13
C2	Divulgazione dell'iniziativa e dei suoi risultati	90	14
D	Valutazione economica	200	45
D1	Capacità di coinvolgimento finanziario di soggetti pubblici e privati	70	15
D2	Congruità dei costi rispetto agli obiettivi attesi	100	16
D3	Affidabilità finanziaria dei partner o dei cofinanziatori	30	17
	TOTALE	**1000**	**187**

	Punteggi aggiuntivi		
E			78
E1	soggetti che hanno conseguito idoneità al precedente bando		18
E2	Comuni che ricadono all'interno della zona di attraversamento appenninico		19
E3	Minor onere finanziario		20
E4	Partecipazione finanziaria allargata a pluralità di soggetti		21
	TOTALE		**78**
	TOTALE GENERALE		**265**

Esito: finanziato

Il Responsabile
(roberta baldini)

Li 8 aprile 2003

pag. 1

Fig. 28 Esempio di stampa in Pdf della valutazione

5.2.3 Fase III: valutazione di finanziamento

Tutte le richieste idonee vengono ordinate in base al punteggio assegnato, stabilendo così una graduatoria. Successivamente, nel rispetto del bando, vengono scelte le prime n richieste della graduatoria che saranno quelle effettivamente finanziate. Quando una richiesta è finanziata nella nostra terminologia prende il nome di progetto.

In questa fase vengono inseriti i dati relativi agli aspetti amministrativi delle sole richieste ritenute idonee.

L'interfaccia usata per questa operazione è il modulo *Finanziamento richieste* illustrato in Fig. 29. La ricerca effettuata da tale modulo consente di trovare tutte le richieste finanziarie inserite, a prescindere dalla situazione amministrativa. L'ultima colonna nella tabella dei risultati riporta la situazione di ogni richiesta (*non ammissibile, ammissibile, idonea, finanziata*). Ma solo aprendo una proposta *idonea* si ha la possibilità di cambiarne lo stato in *finanziata* e di inserire altri dati (e quindi di procedere nella fase di assegnazione del cofinanziamento) dal momento che le richieste *ammissibili/non ammissibili* sono in sola lettura.

Fig. 29 Modulo Finanziamento

La Fig. 30 mostra un esempio di richiesta non idonea (per la precisione *non ammissibile*): tutti i campi di entrambi i pannelli (Posizione e Dati Amministrativi) non sono editabili e i valori caricati si riferiscono a quelli inseriti nelle fasi precedenti e qui riportati per conoscenza.

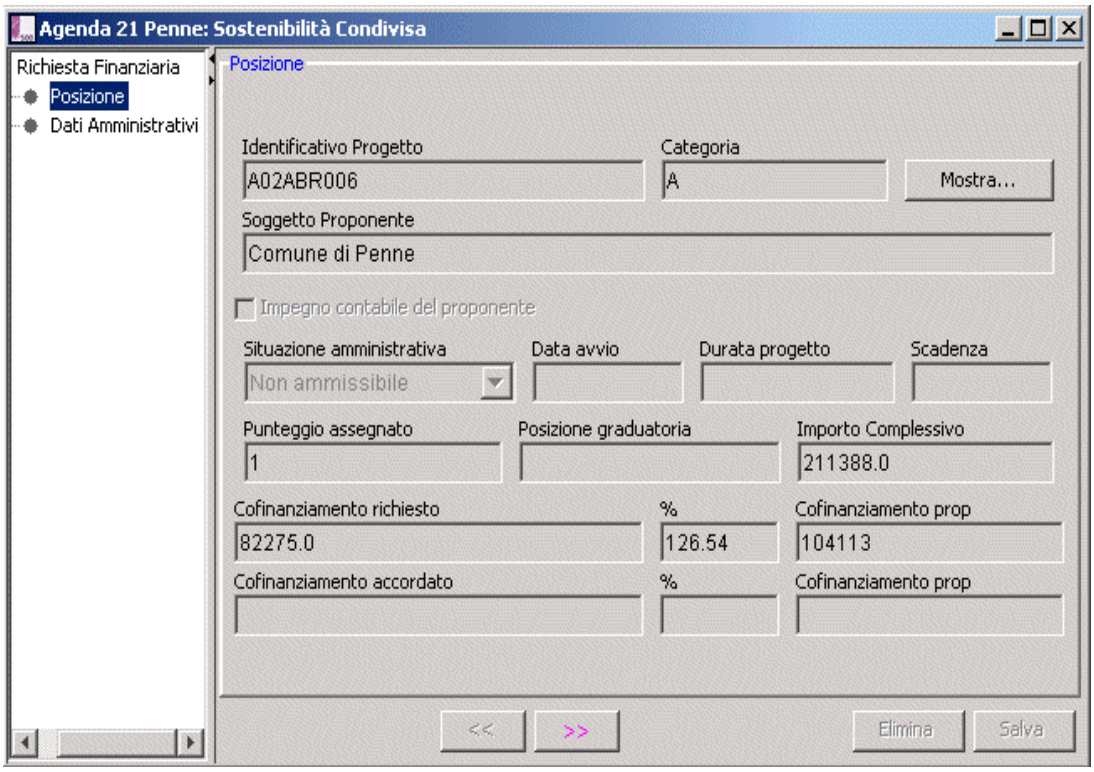


Fig. 30 Esempio di richiesta non idonea: impossibile procedere nel finanziamento

La Fig. 31 riporta invece un esempio di progetto ritenuto *idoneo* nella fase precedente: come si può vedere, in questo caso l'unica possibilità che viene data all'utente è di poter cambiare la situazione amministrativa da *idoneo* a *finanziato*.

Se il progetto è ritenuto *finanziato* vengono abilitati i campi in cui si ha la possibilità di inserire nuovi dati, come mostra la Fig. 32. Nel primo pannello vengono inserite informazioni sulla data di avvio, scadenza e posizione in graduatoria del progetto e sul cofinanziamento e cofinanziamento del proponente accordato, che può essere minore o uguale a quello richiesto. Mentre nel secondo pannello vengono richieste informazioni sui dati amministrativi dell'ente capofila (Fig. 32a).

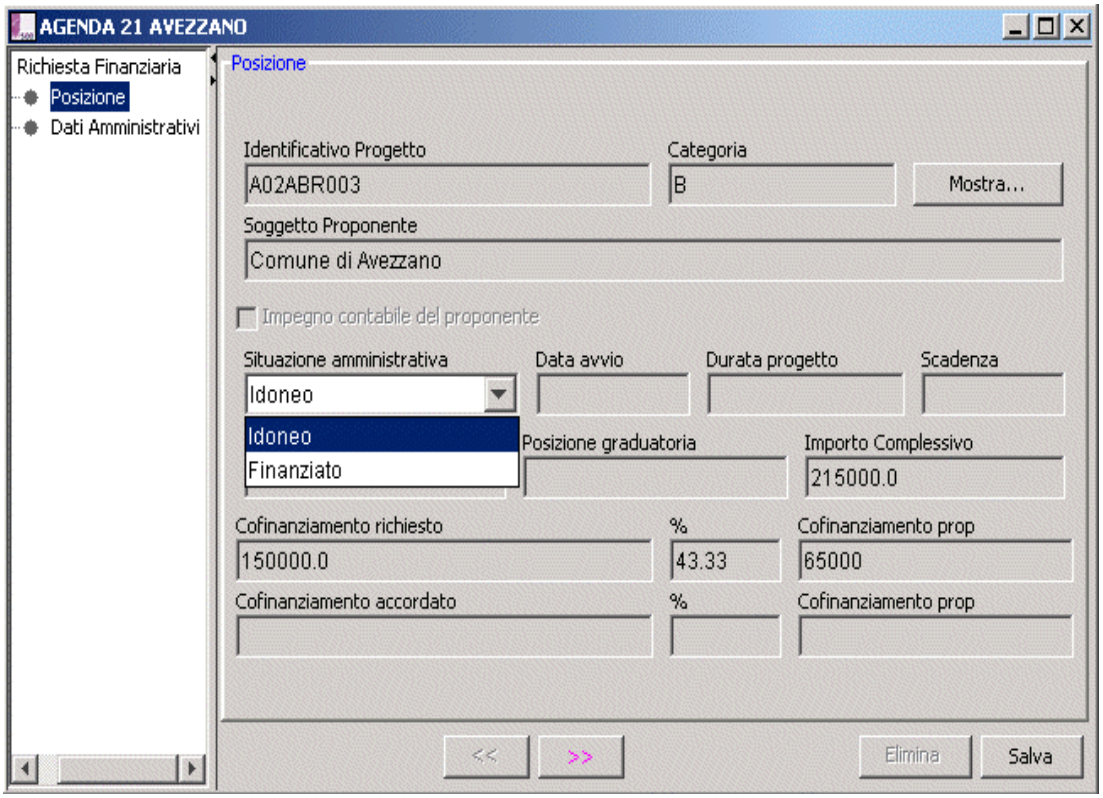


Fig. 31 Esempio di richiesta idonea: possibilità di procedere nel finanziamento

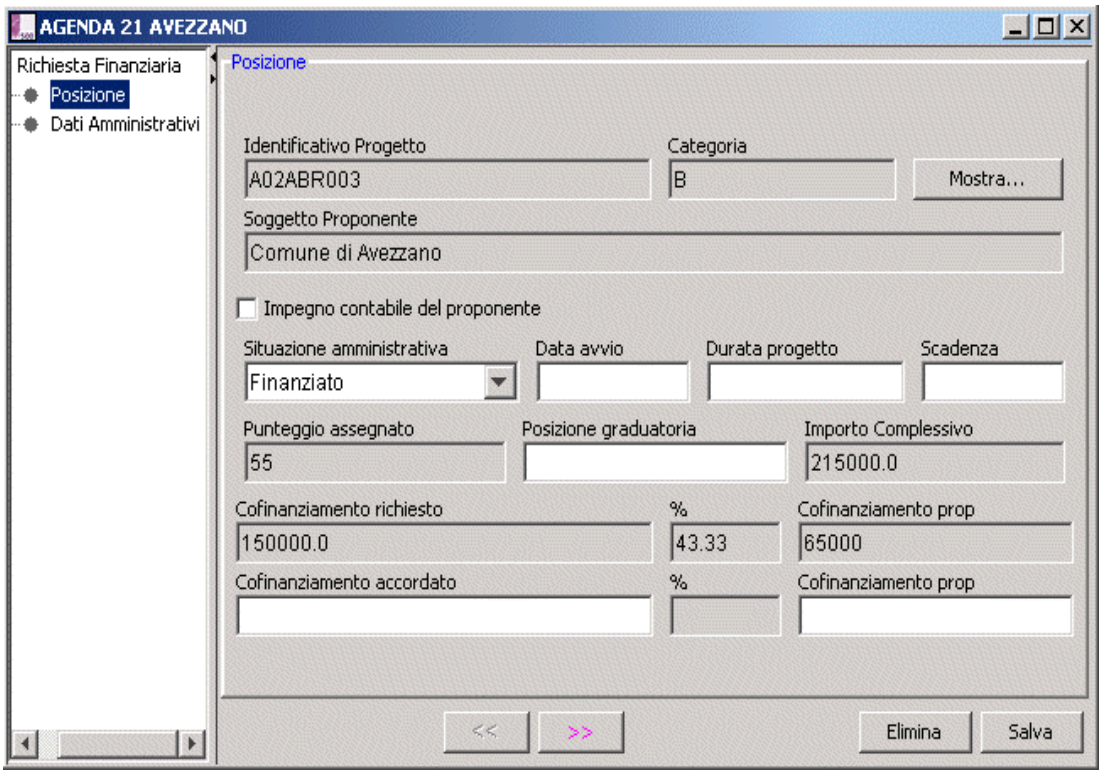


Fig. 32 Esempio di richiesta finanziata

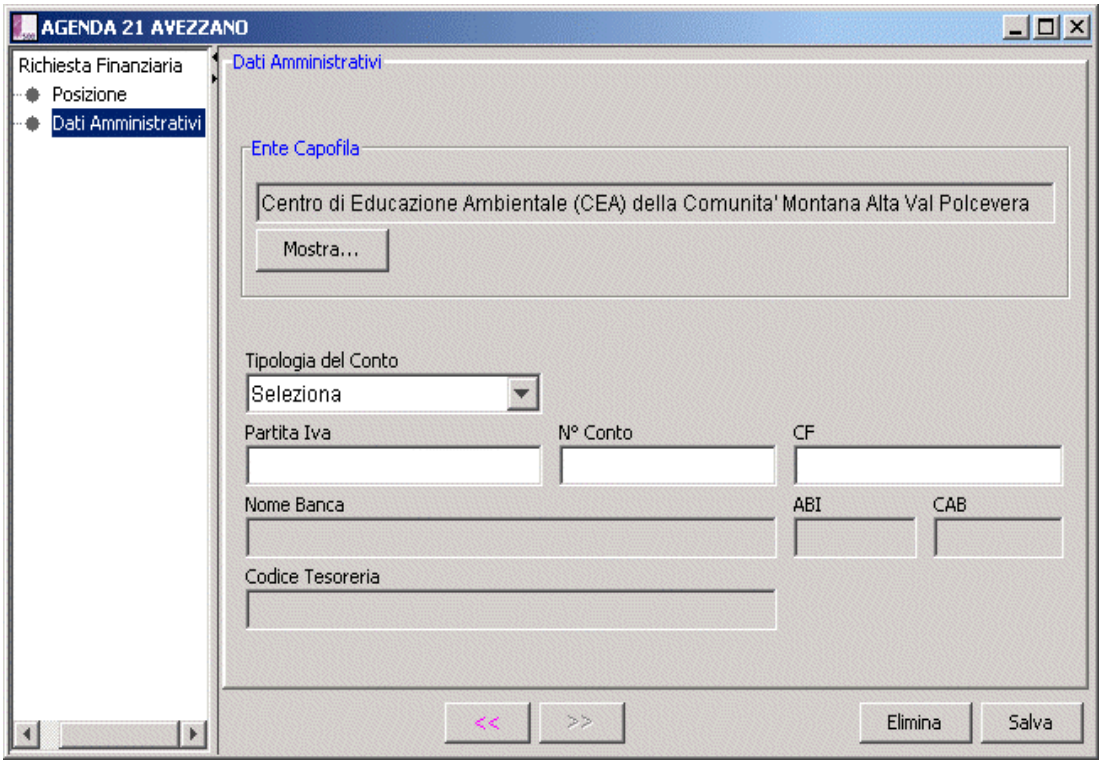


Fig. 32a Esempio di richiesta finanziata (pannello Dati Amministrativi)

5.2.4 Fase IV: assegnanzione del cofinanziamento

Un progetto salvato come finanziato passa all'ultima fase della valutazione e selezione, cioè alla fase di assegnazione del cofinanziamento. Tale tipologia di progetto viene ricercata attraverso il modulo *Gestione Contratti* impostando, nella sezione filtri, la tipologia *Agenda 21 locale* (Fig.33).
La Fig. 34 mostra il frame risultante dall'apertura di un progetto finanziato e composto da due pannelli.

Nel primo pannello (Posizione) vengono caricati alcuni dati relativi al progetto, inseriti durante le fasi precedenti, e riportati per conoscenza: la situazione amministrativa e varie informazioni di tipo finanziarie; mentre altri dati, tra cui quelli amministrativi dell'ente capofila, vengono visualizzati premendo il pulsante "Mostra…".
Il secondo pannello (Dettagli) illustrato in Fig. 35 consente l'inserimento dei dati: in alto abbiamo la tabella "Capitoli", vi vengono inseriti gli importi disponibili per ogni capitolo di spesa selezionato tra i valori dei capitoli di spesa inerenti al bando a cui si riferisce la richiesta finanziaria (nel caso attuale il bando "Agenda 21"). In basso si ha la possibilità di inserire le tre rate in cui viene erogato il

cofinanziamento del Ministero. È obbligatorio che queste tre rate vengano inserite in ordine, per tale motivo i campi di una tranche si attiveranno solo nel momento in cui viene inserita la tranche immediatamente precedente.

Questa rappresenta l'ultima fase della valutazione dei progetti e con essa si chiude la trattazione della gestione delle richieste finanziarie. La sezione successiva illustrerà l'inserimento e la modifica delle altre tipologie di progetti: contratti, convenzioni, decreti.

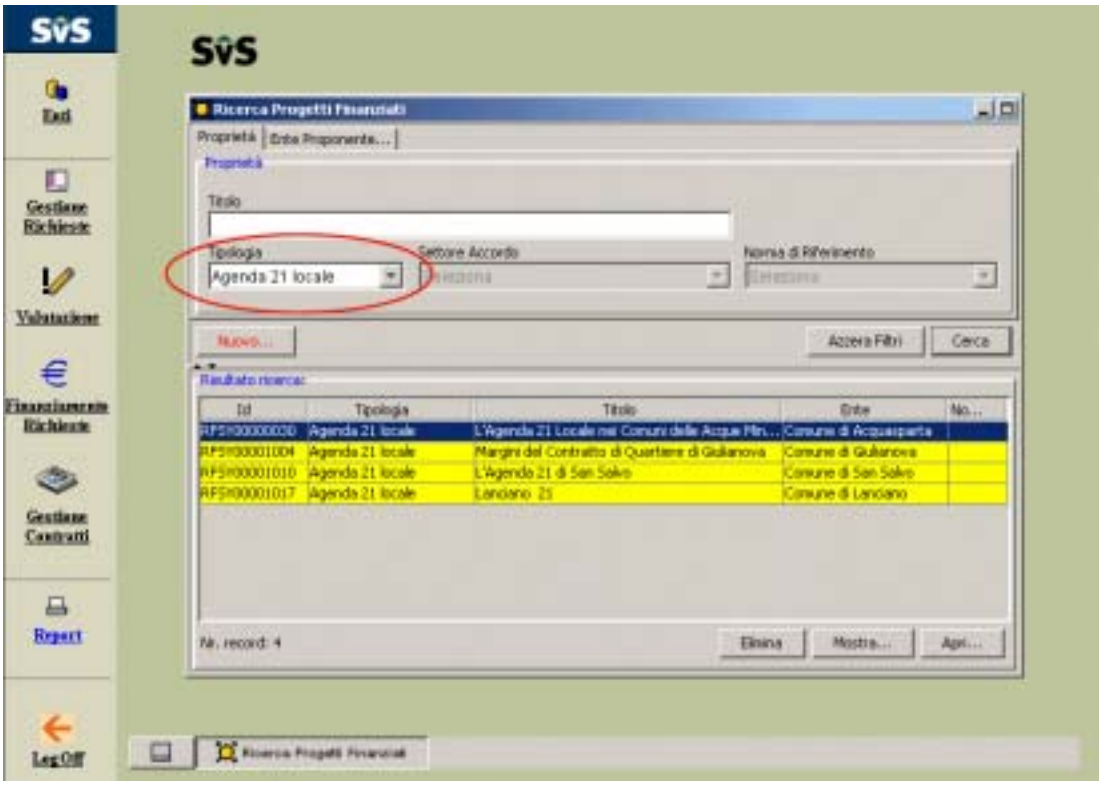


Fig. 33 Modulo Gestione Contratti

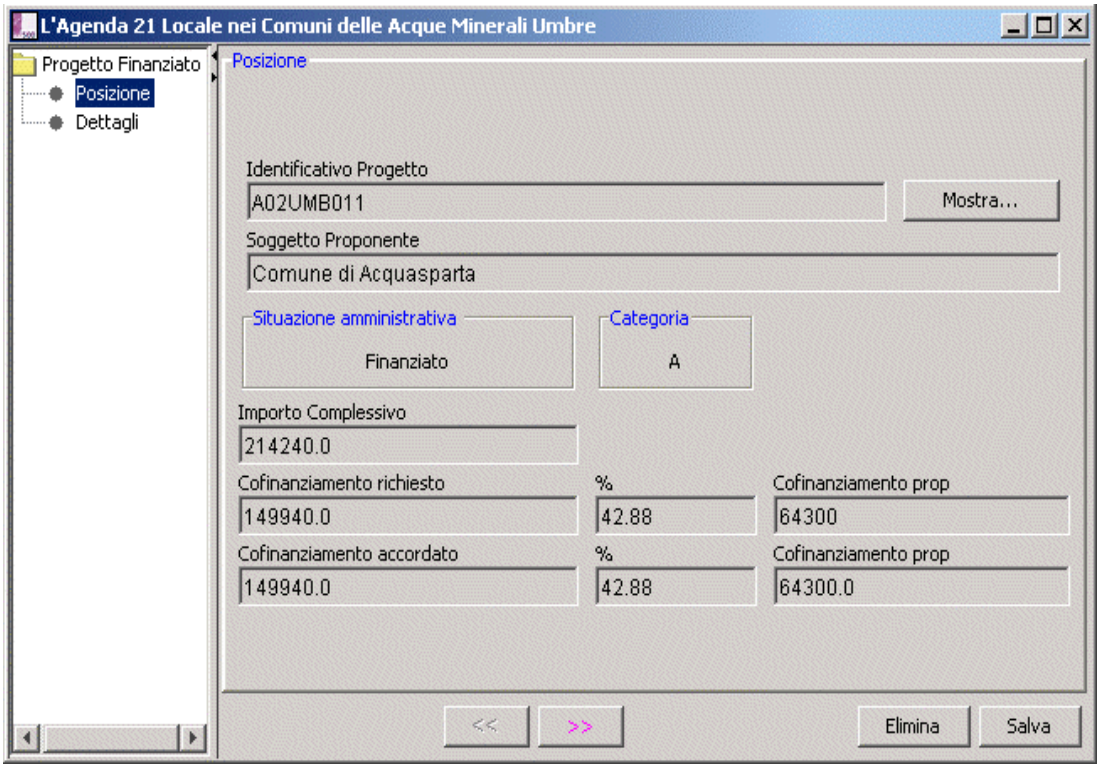


Fig. 34 Progetto Finanziato

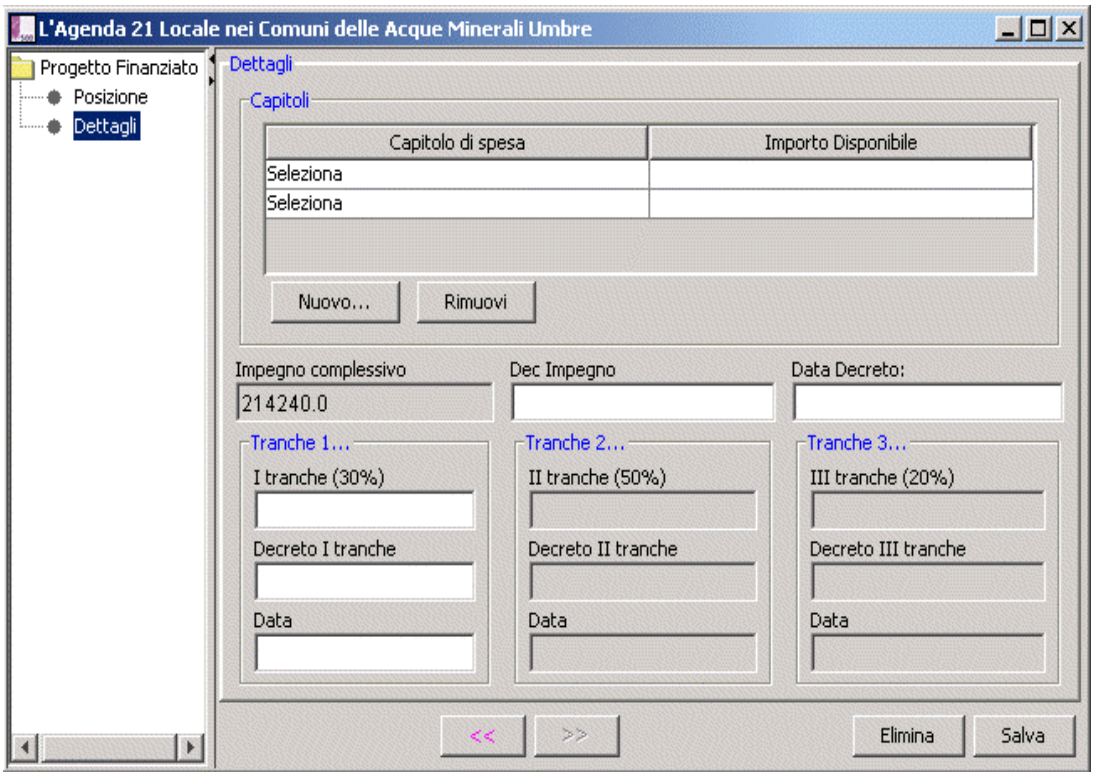


Fig. 35 Progetto Finanziato

5.3 Gestione dei contratti/convenzioni/decreti

Il modulo da utilizzare per ricercare, inserire e modificare un contratto, una convenzione o un decreto è il modulo *Gestione contratti* riportato in figura 33. In seguito verrà utilizzato il termine generico di progetto per riferirsi alle tipologie suddette.

Premendo il bottone "Nuovo…" viene aperto un frame da utilizzare per inserire un nuovo progetto (Fig.36).

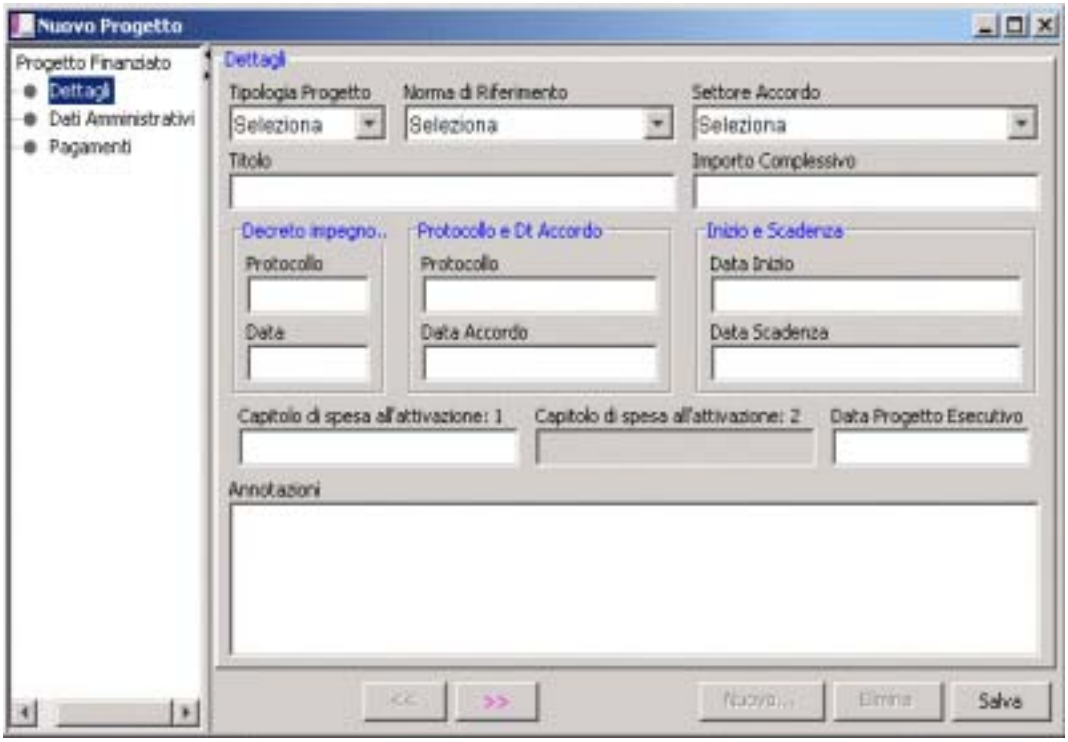


Fig. 36 Frame per l'inserimento di un progetto

Esiste un menù a tendina ("Tipologia progetto") con cui specificare il tipo di progetto che si vuole inserire. Il campo "Capitolo di spesa all'attivazione: 2" è un'informazione propria del progetto Decreto e viene attivato solo se viene scelta questa tipologia.

Nel secondo pannello (Dati Amministrativi) viene selezionato un ente proponente di cui si richiede l'immissione dei dati amministrativi (Fig.37).

Nel terzo pannello (Pagamenti) viene data la possibilità di inserire informazioni circa un numero variabile di pagamenti. Per ogni pagamento viene inserita la "Data prevista", la "Data di pagamento" e l'"Importo", nella tabella, e altre informazioni dettagliate ("Stato registrazione", "Protocollo", ecc) nel

relativo pannellino sottostante (Fig. 38).

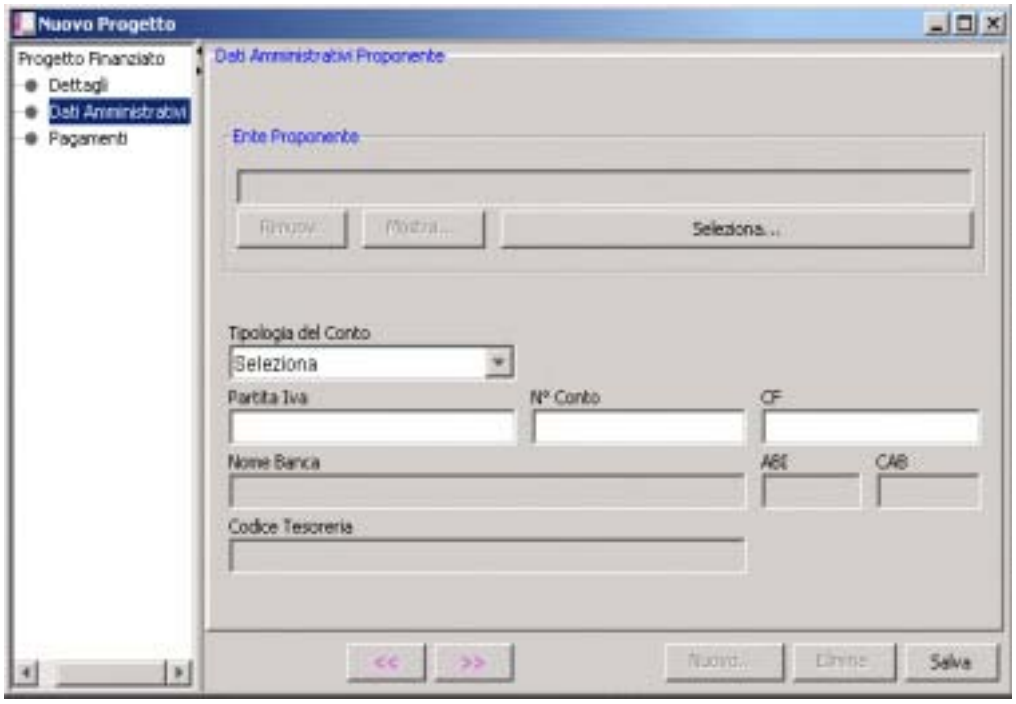


Fig. 37 Dettaglio del frame di inserimento/modifica di un progetto

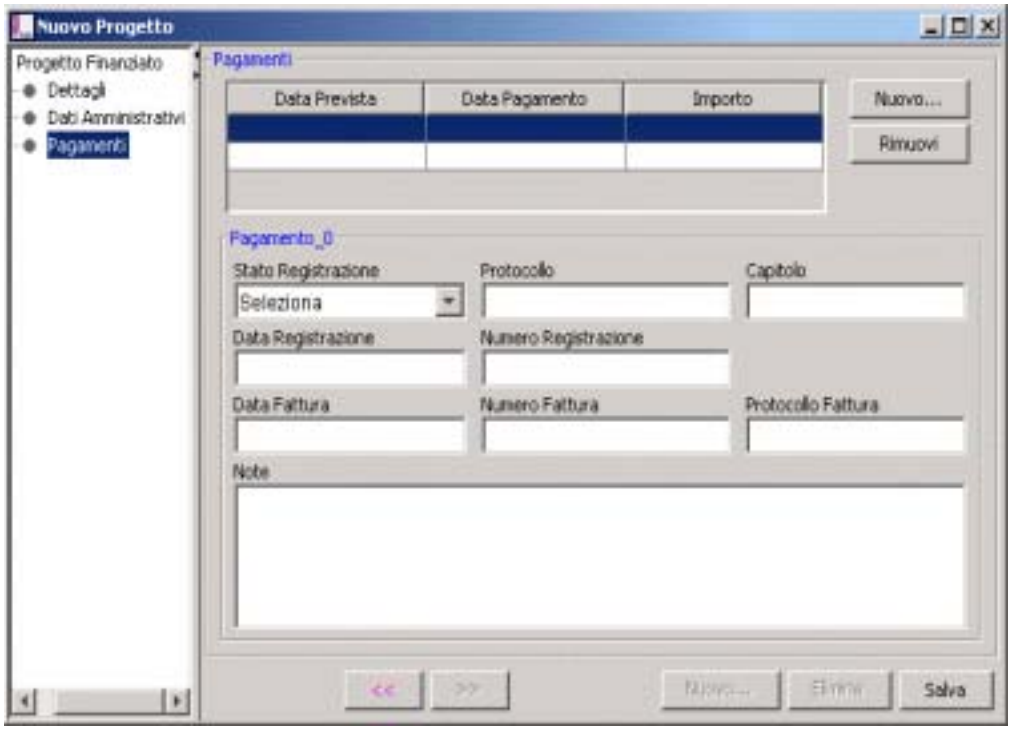


Fig. 38 Dettaglio del frame di inserimento/modifica di un progetto

6 Il modulo Report di Agende 21

Il Sistema Sviluppo Sostenibile è stato arricchito con il modulo per la generazione di report, questo strumento è stato messo a disposizione dei soli utenti aventi il ruolo di redattore. Nella scrivania (desktop) dell'utente redattore è presente l'icona (Fig. 39) per l'accesso al modulo di richiesta report.

Figura 39 La scrivania

Il modulo report si presenta diviso in due frame, nella parte superiore si possono impostare i filtri per la ricerca, in quella inferiore si possono scegliere il formato per la presentazione delle informazioni, l'ordinamento ed il layout.

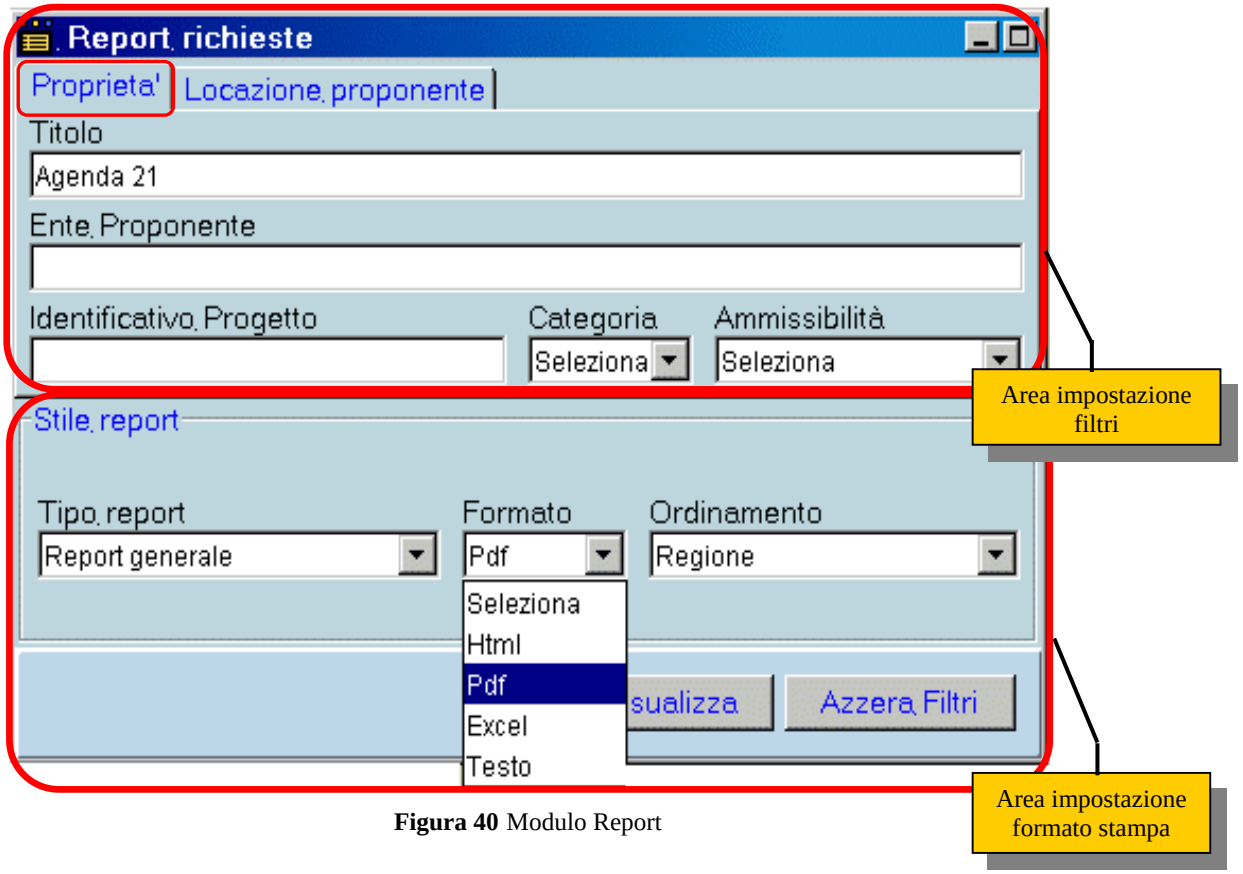


Figura 40 Modulo Report

La ricerca si effettua impostando nel panello *Proprietà* (Fig. 40) e nel panello *Locazione* (Fig 41) le condizioni di ricerca. L'utente può richiedere al sistema informativo il report impostando il solo filtro sul titolo del progetto, oppure può rendere più raffinata l'interrogazione aggiungendo ulteriori filtri nel campo Ente Proponente, Identificativo esterno, etc.

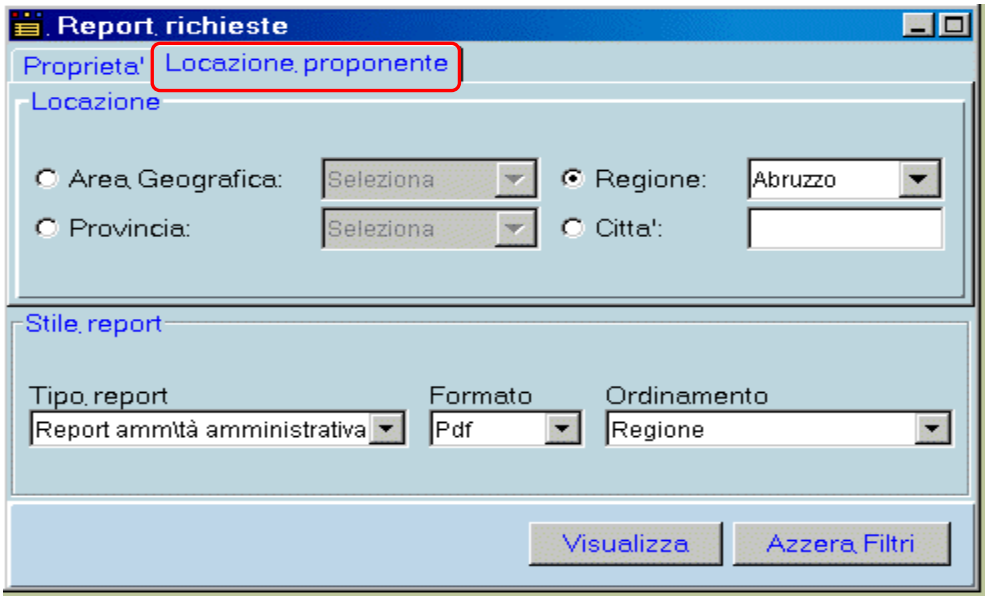


Figura 41 Modulo Report Locazione

Una volta stabiliti i filtri l'utente può:

- scegliere il formato di output, selezionando una voce dal combo box *"Formato"*; sono previsti 4 formati: HTML, PDF, EXCEL e TESTO
- scegliere il layout della risposta tra i quelli proposti nel combobox *"Tipo Report"*
- scegliere l'ordinamento del risultato.selezionando una delle voci del combo box *"Ordinamento"*,(Fig 42).

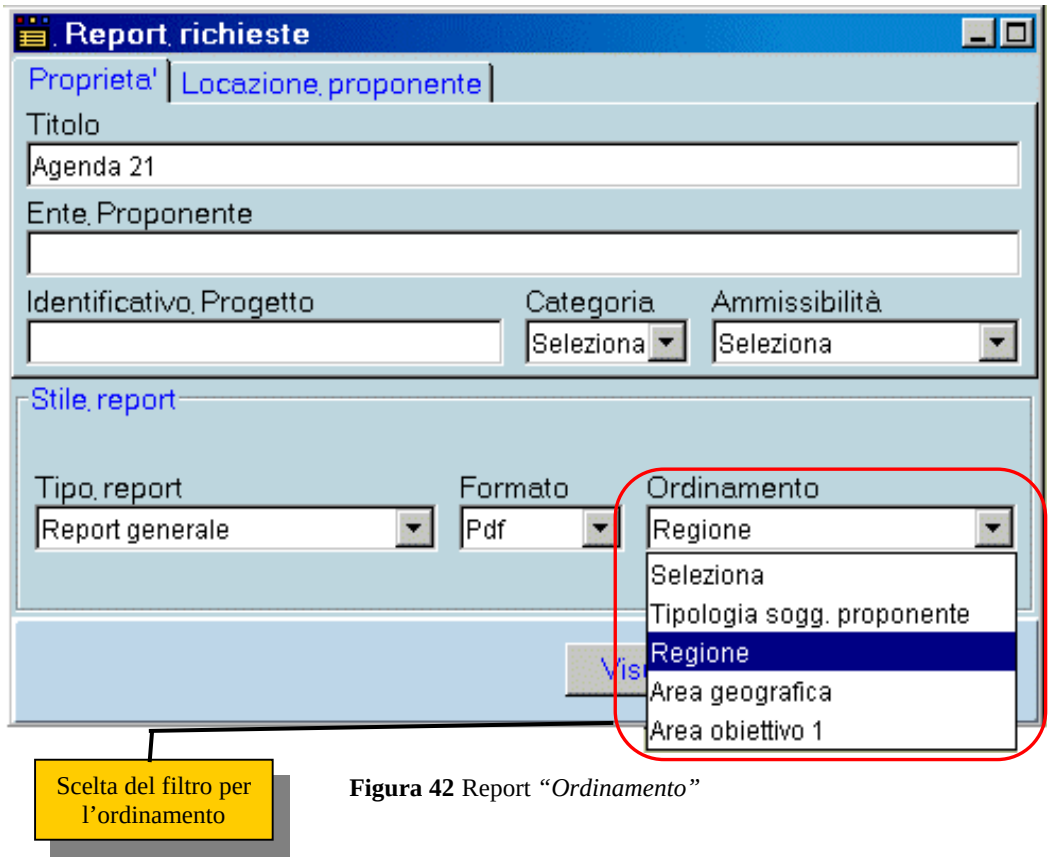


Figura 42 Report *"Ordinamento"*

Decisi i filtri e il tipo di presentazione dei dati, (click su "Visualizza") ottiene il risultato; prima di effettuare una nuova ricerca l'utente può o modificare i campi interessati al cambiamento oppure azzerare i filtri impostati avvalendosi del tasto "Azzera Filtri", in tal caso tutti i campi saranno ripuliti, Fig 43.

Nei paragrafi che seguono illustriamo in dettaglio i diversi tipi di report.

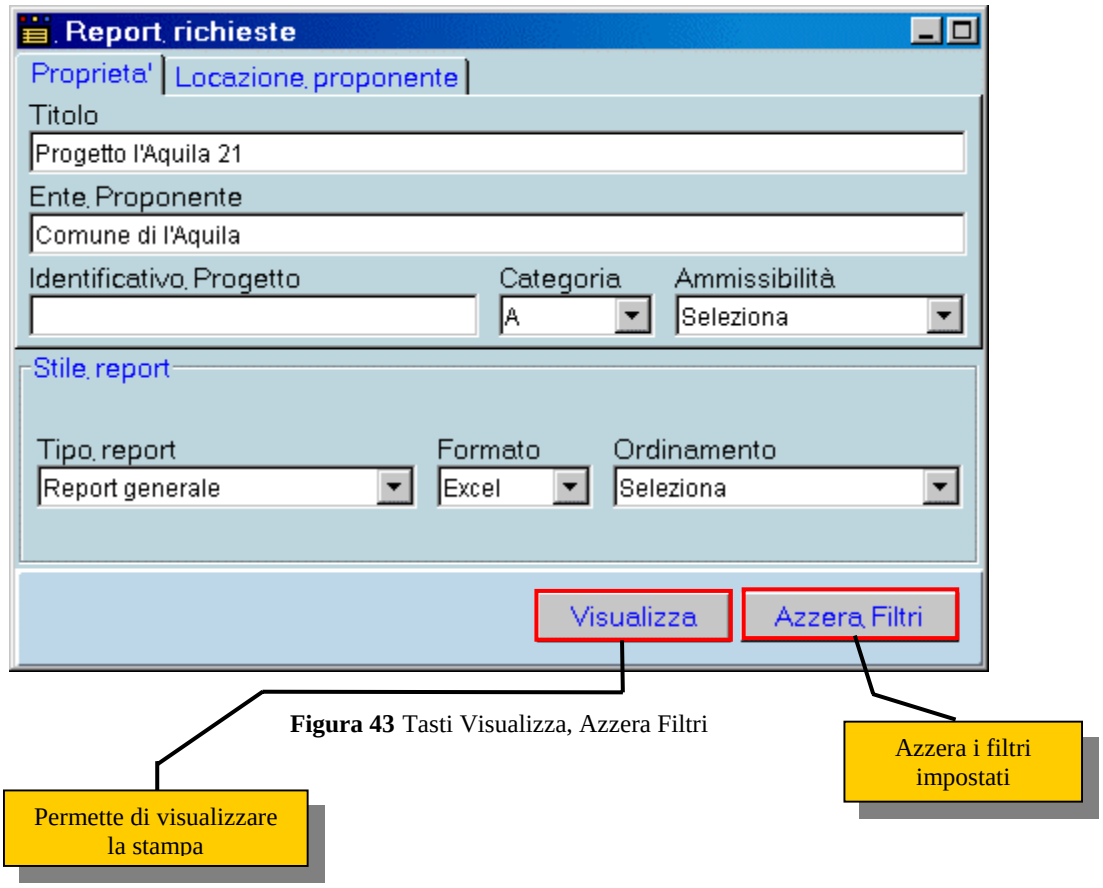


Figura 43 Tasti Visualizza, Azzera Filtri

1.1 Formato stampa PDF

Per visualizzare i documenti in formato PDF è sufficiente disporre del plug-in Acrobat Reader di Adobe.

1.1.2 Report Generale

Il report generale, sia in stampa singola sia per aggregazioni, fornisce informazioni dettagliate sulle richieste di finanziamento Ag 21L.

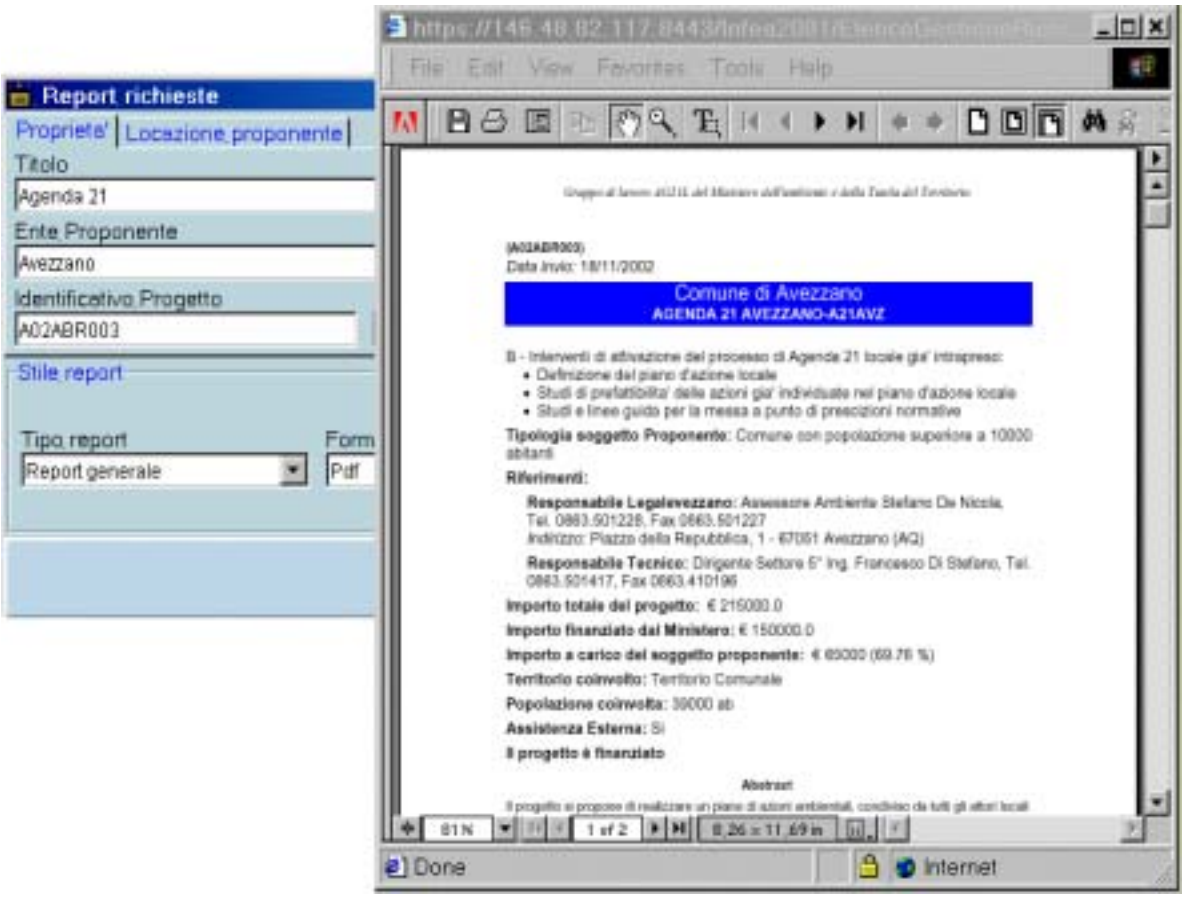


Figura 44 Modulo Report con output PDF

Le informazioni sono presentate nel seguente modo: in alto a sinistra è evidenziato l'identificativo del progetto, allo scopo di facilitare la catalogazione o la ricerca del documento in archivio. Di seguito sono inserite informazioni quali: Soggetto Proponente, Titolo del progetto, Acronimo e Categorie etc. La locazione è quella dell'ente capofila. In questo report sono riportate anche i nomi ed i ruoli dei referenti amministrativi e tecnici del progetto e gli importi richiesti e cofinanziati. In grassetto è evidenziato l'esito della valutazione del progetto.

(A02ABR019)
Data invio: 19/11/2002

Comunità Montana Valle Roveto
AGENDA 21 nei Territori Montani Marsicani-ATERR.VA.RO

A - Interventi di attivazione del processo di Agenda 21 locale:
- Azioni di coinvolgimento degli attori locali
- Costituzione del forum permanente di Agenda 21
- Redazione del rapporto sullo stato dell'ambiente

Tipologia soggetto Proponente: Comunita' montana

Ente Capofila: Comunità Montana Valle Roveto, *Diaz - Avezzano (AQ)*

Enti CoProponenti: Parco Naturale di Interesse Provinciale del Lago di Candia

Enti Cofinanziatori: COFINANZIATORI DIVERSI DA ENTI

Riferimenti:

Responsabile Legale: Presidente Enzo Palmerini, Tel. 0863.97308, Fax 0863.97613, info@valleroveto-aq.it
Indirizzo: Via Roma, 2 - 67054 Civitella Roveto (AQ)
www.valleroveto-aq.it

Responsabile Tecnico: Dirigente Stefano Di Rocco, Tel. 0863.97308, Fax 0863.97613, info@valleroveto-aq.it
Indirizzo: Via Roma, 2 - 67054 Civitella Roveto (AQ)

Importo totale del progetto: € 77468.52

Importo finanziato dal Ministero: € 43382.37

Importo a carico del soggetto proponente: € 34086.15 (56.00 %)

Territorio coinvolto: 305.05 kmq

Popolazione coinvolta: 18993 ab

Assistenza Esterna: Si

La richiesta di finanziamento è ammissibile

Abstract

BREVE DESCRIZIONE DEL PROGETTO
La CM della Valle Roveto (sup. 305.05 kmq, 80% montano, 19.121ab.) si propone con il progetto di continuare le azioni avviate sul proprio territorio con il programma Leader II e l'imminente Leader Plus, nonché dal Consorzio per il Patto Territoriale della Marsica e dal progetto APE.Il territorio in questione non ha ancora avuto lo sviluppo, soprattutto turistico, che altre aree montane hanno raggiunto; questa situazione, determina un territorio di per sé 'sostenibile'.La notorietà del comprensorio è legata alle tante risorse, come l'acqua, le castagne, le piante officinali, la vegetazione delle più disparate specie, e dai pregevoli contesti storico-architettonici. Il progetto intende predisporre in relazione al PSSE e alle azioni già avviate sul

Figura 45 Report Generale

6.1.1 Report Ammissibilità Amministrativa

Il report Ammissibilità amministrativa fornisce una visualizzazione tabellare e sintetica delle valutazione amministrativa della richiesta di finanziamento. Scelta la voce “Ammissibilità amministrativa” dal combo box *Tipo Report*, l’utente deve selezionare la “Categoria” a cui la richiesta di finanziamento appartiene. Se si inserisce anche un filtro sulla locazione, nel riquadro evidenziato in rosso è riportato oltre alla categoria anche il filtro scelto. Un esempio di stampa in Fig 46.

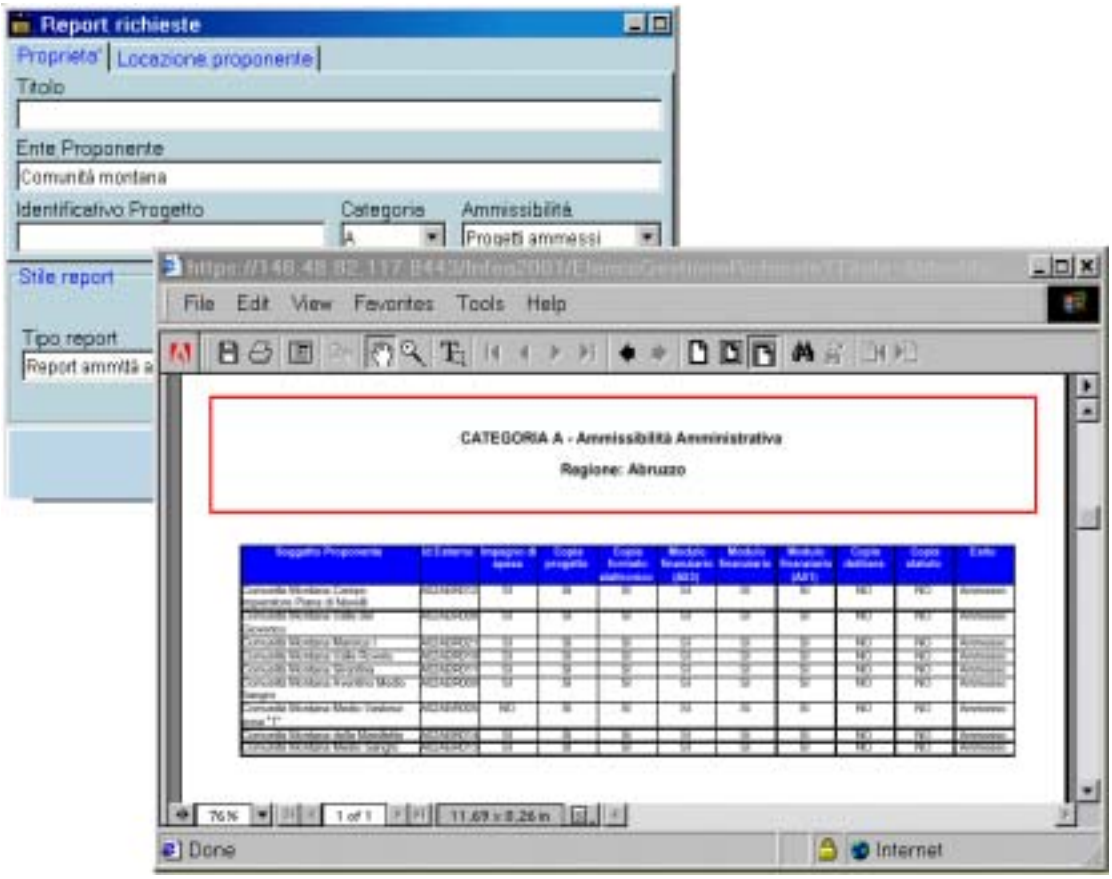


Figura 46 Report Amm\tà amministrativa

6.1.2 Report graduatoria

Il report graduatoria è un altro esempio di report in PDF, anche in questo caso abbiamo un'organizzazione tabellare e sintetica delle informazioni. Il layout presenta particolarità comuni a tutti gli altri layout nella presentazione delle informazioni: in alto a sinistra è riportato l'identificativo esterno del progetto, nel riquadro di colore azzurro sono inserite informazioni riguardanti il titolo del progetto, l'ente proponete, l'acronimo ecc. Fig 47.

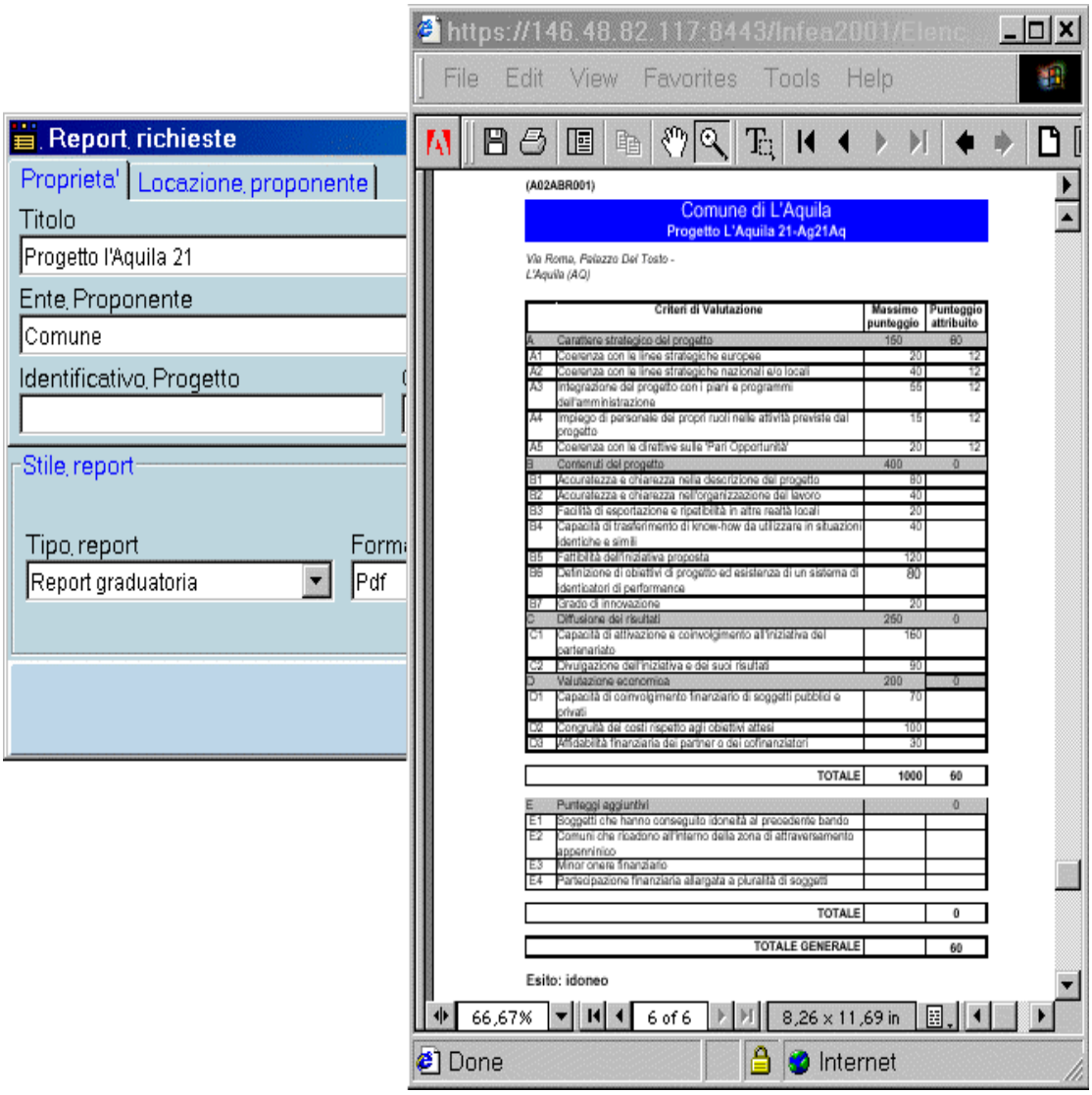


Figura 47 Report Graduatoria

6.1.3 Report ammissibilità amministrativa singola

Anche in questo caso, come nel caso del Report ammissibilità amministrativa, l'utente prima di richiedere al sistema l'output deve scegliere la categoria in base alla quale effettuare la ricerca. Il layout è stato studiato in maniera tale da mostrare all'utente se ha effettuato una ricerca selezionando la categoria A oppure B (Fig 48).

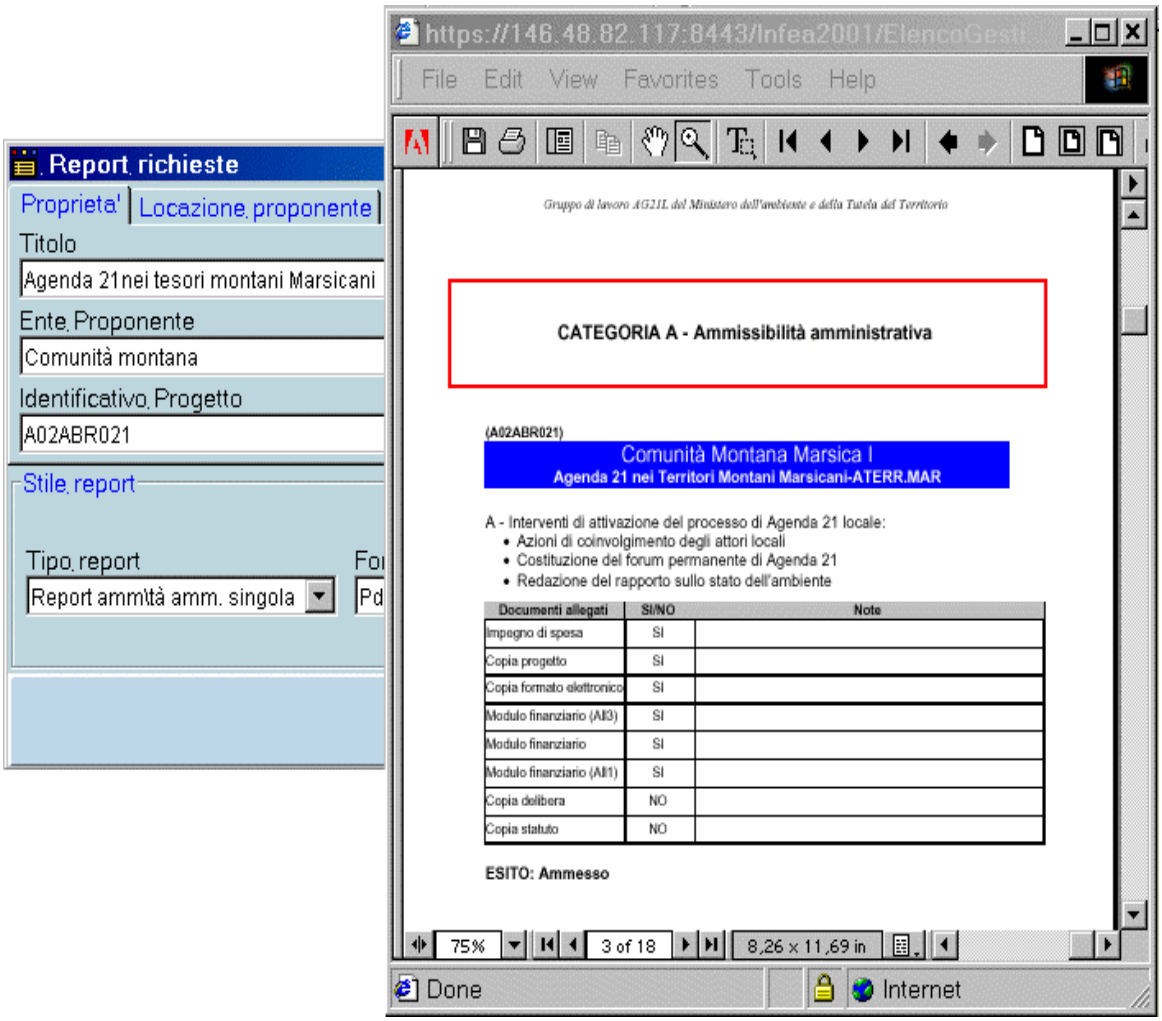


Figura 48 Report Ammissibilità amministrativa singola

1.2 Formato stampa HTML

Il formato HTML è usato per la visualizzazione “read only” delle infromazioni in INFEA. Non tutte i dati presenti negli altri formati sono riportati in HTML anche se i layout sono naloghi. L’utente per visualizzare il contenuto del sistema informativo deve impostare la voce del combo box *Formato* su HTML. Le modalità di ricerca sono analoghe a quelle presentate sopra.

Mostreremo di seguito alcuni risultati ottenuti impostando questa modalità di presentazione delle informazioni Fig 49 e 50.

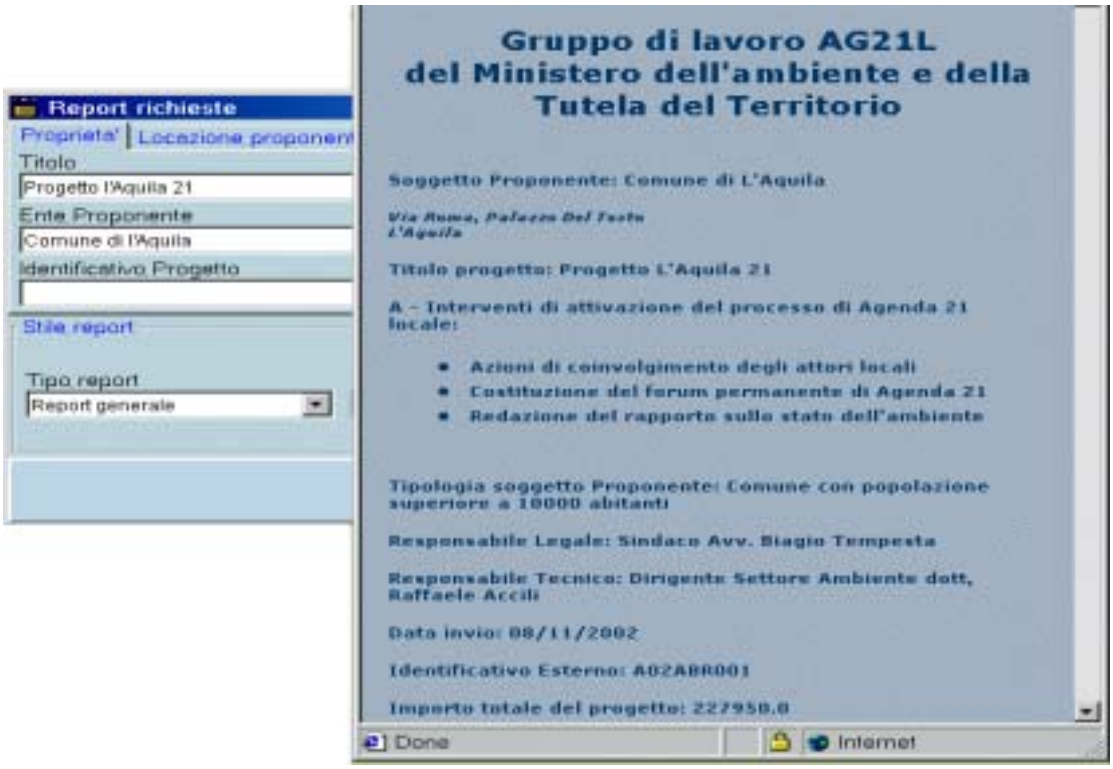


Figura 49 Report Generale HTML

Figura 50 Report Ammissibilità amministrativa singola HTML

1.3 Formato stampa Excel

Per visualizzare automaticamente i documenti in formato Excel, è necessario che sul browser sia configurato correttamente il tipo MIME corrispondente. In caso contrario i file potranno essere salvati sul disco ed aperti con il programma Microsoft Excel. Abbiamo scelto di inserire questa modalità di presentazione dei dati per permettere allo staff editoriale di effettuare elaborazioni statistiche sui dati contenuti nel data base. Anche in questo caso, le informazioni riportate rispecchiano le modalità di presentazione scelte per gli altri due formati. L'utente per ottenere la stampa in Excel deve impostare dal combo box *Formato* la voce Excel. Per quanto riguarda l'impostazione dei filtri la modalità è uguale a quella degli altri due formati di stampa. In Fig 51 un esempio di stampa in Excel.

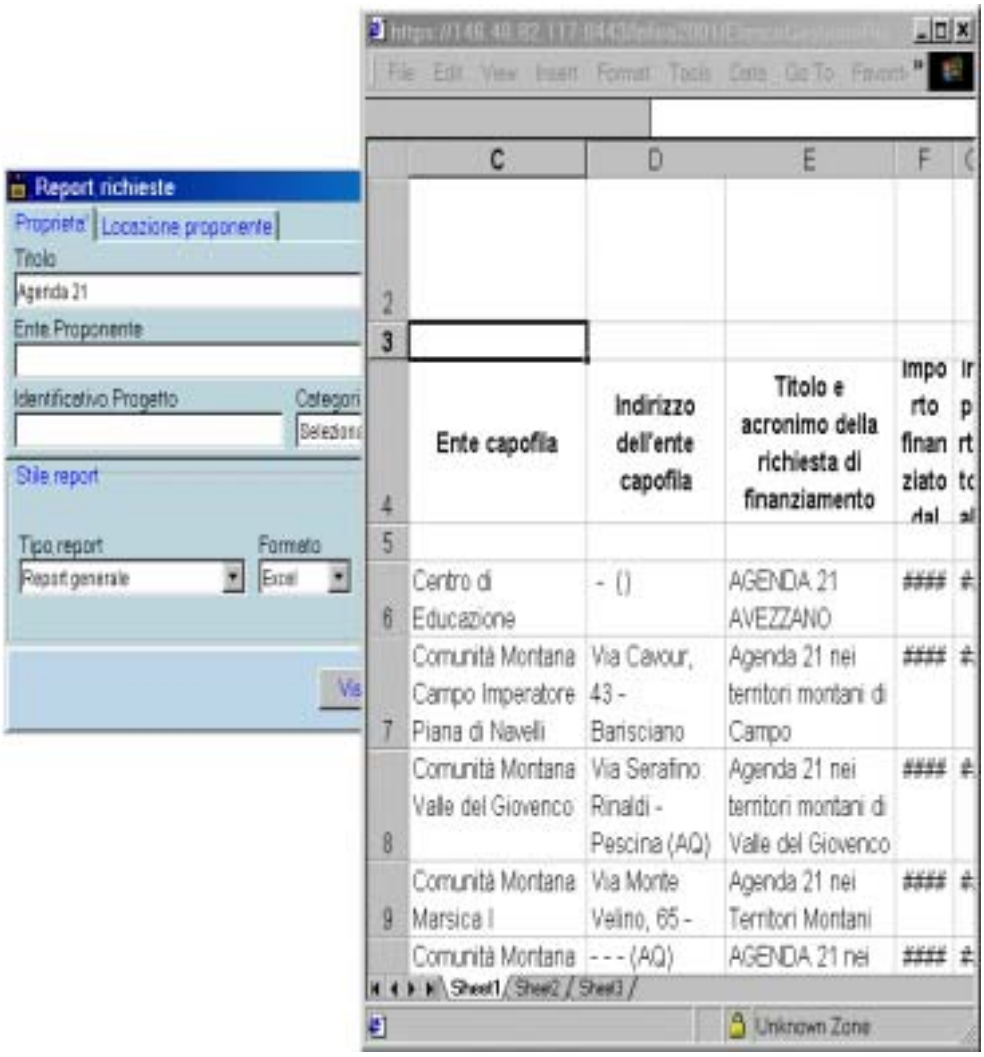


Figura 51 Stampa EXCEL

7 Liste di distribuzione

Una lista di distribuzione è un elenco di indirizzi di posta elettronica. Questo modulo gestisce liste di distribuzione di e-mail, software, documenti. Consente la creazione dinamica di liste temporanee o permanenti, selezionando, secondo determinati criteri, i destinatari dalla base di dati. Questa attività ha contemplato la progettazione e realizzazione delle componenti della base di dati per il supporto, la definizione delle specifiche implementative, e la realizzazione dell'interfaccia utente per la definizione e ricerca di liste di distribuzione.

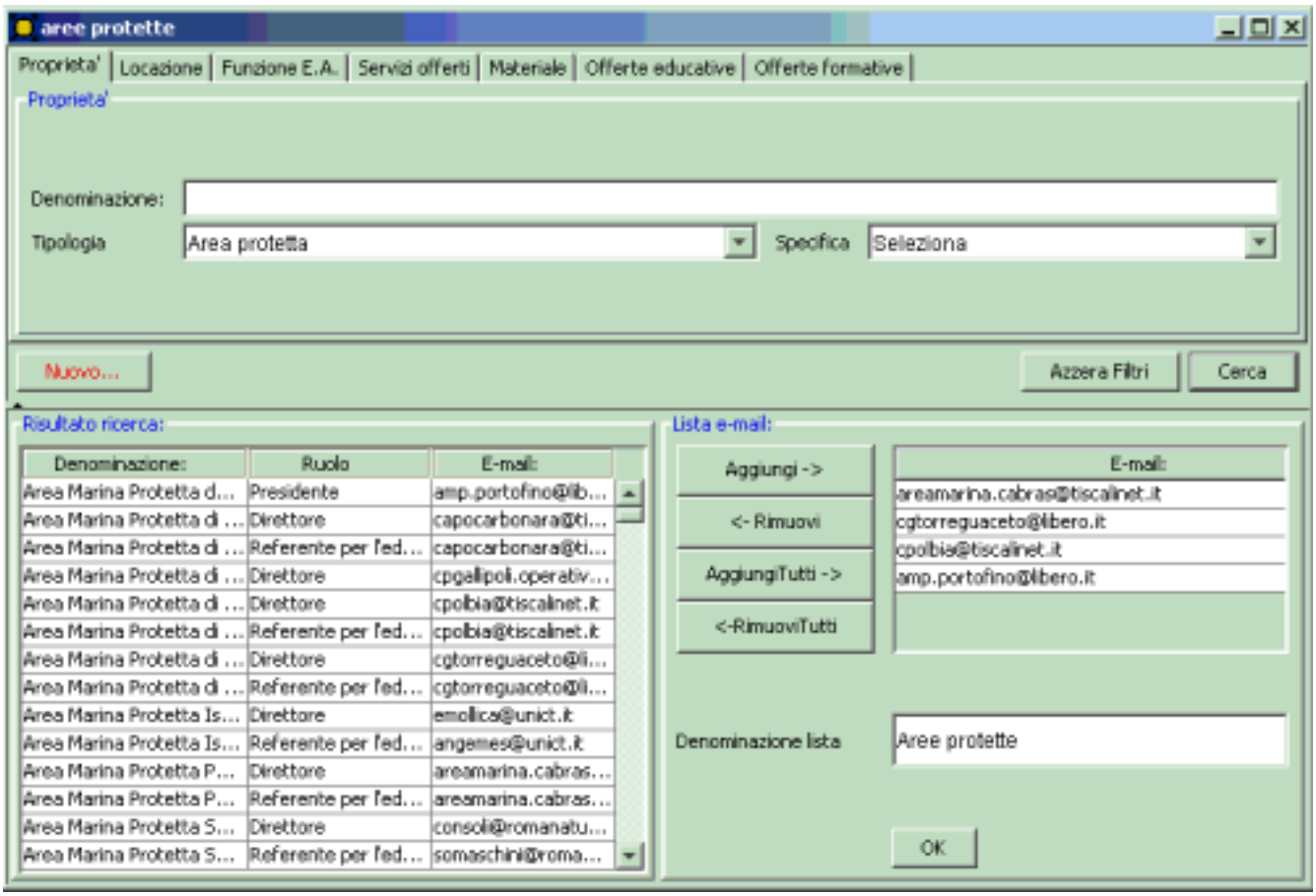


Figura 52 Definizione di una lista di distribuzione

In figura 52 è mostrata l'interfaccia per la definizione e modifica di una lista di distribuzione. La parte superiore della finestra consente di esprimere condizioni secondo l'articolata modalità di ricerca enti del sistema informativo SVS. La parte inferiore è composta da due liste e da un insieme di bottoni per lo spostamento di elementi da una lista all'altra. La lista di sinistra presenta l'elenco di tutti gli indirizzi

di posta elettronica relativi agli enti che hanno soddisfatto tutte le condizioni di ricerca espresse. È possibile selezionare da questa lista gli elementi che comporranno la lista di distribuzione, a cui viene assegnato un nome.

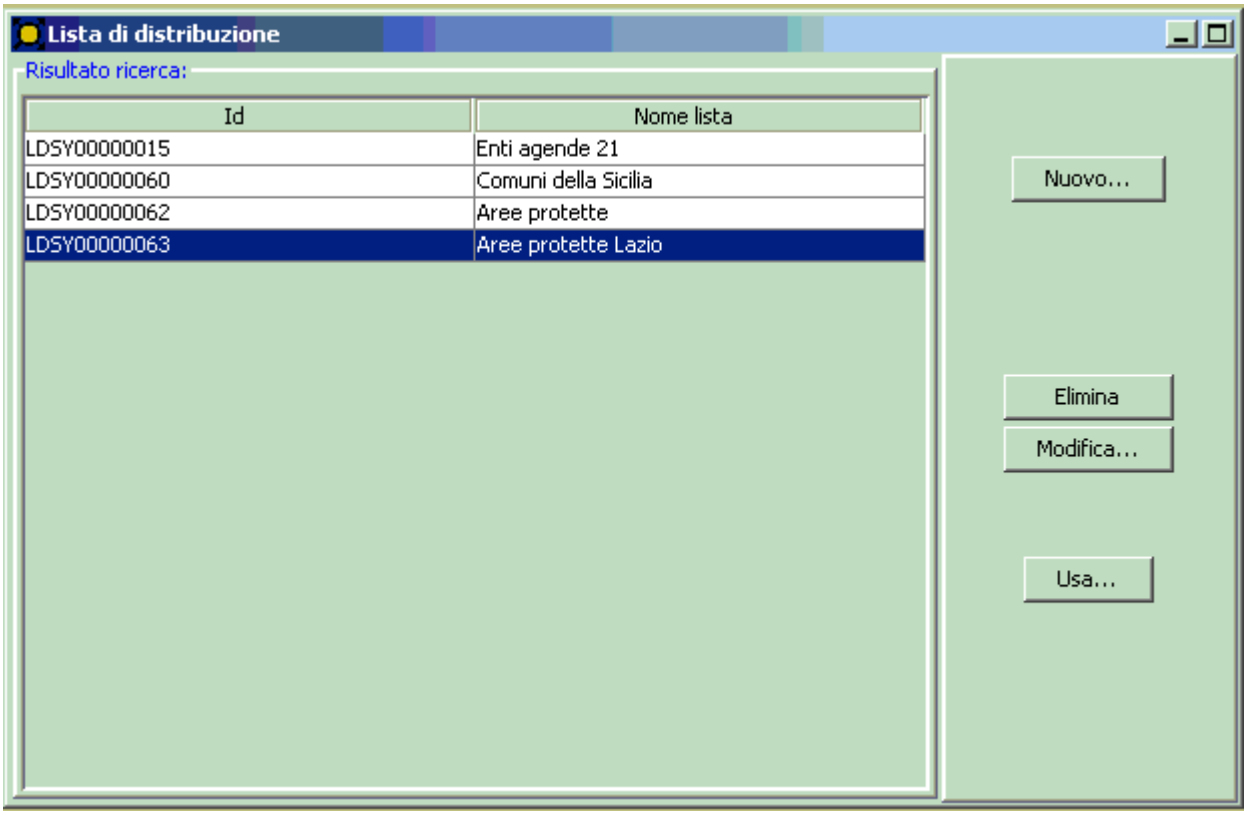


Figura 53 Gestione ed utilizzo di liste di distribuzione

È possibile creare liste di distribuzione per le necessità di gestione scelte dall'utilizzatore. In figura 53 è mostrato un elenco di liste di distribuzione. Il bottone "Usa…" attiva un client di posta elettronica, impostando come destinatari tutti gli elementi compresi nella lista di distribuzione selezionata; è così possibile compilare un messaggio, agganciare documenti da distribuire ed attivare la distribuzione.

8 Strumenti per lo sviluppo e l'implementazione del sistema Informatico

Questi i punti chiave nella scelta degli strumenti di sviluppo e gestione del progetto:

- Utilizzare, quando possibile, strumenti basati sulla tecnologia Java per soddisfare le specifiche di portabilità e distribuzione del codice fissate nel progetto.
- Focalizzare l'attenzione sulle tecnologie XML based
- Nella scelta degli strumenti OpenSource valutare il supporto della comunità e la presenza di documentazione.

8.1 Ambiente di sviluppo

Il criterio principale su cui abbiamo basato la scelta della Virtual Machine (VM) Java è stato l'aderenza alle specifiche del linguaggio, di conseguenza la scelta è caduta sul Software Development Kit (SDK) della Sun. La scelta degli IDE è stata lasciata ai singoli sviluppatori nel rispetto delle specifiche, mentre l'ambiente di test è stato creato ad hoc per ciascun modulo.

L'ambiente di sviluppo risulta così composto:

- Design tool: Microsoft Visio2000
- Integrated Development Environment (IDE): Borland Visual Café 4.0, Xinox Jcreator 2.5, NetBeans 3.4
- Java VM: J2SE 1.4.0 SDK
- CVS: WinCvs 1.2
- Test: abbiamo sviluppato tool per il test e per la valutazione di performance dei server e delle connessioni RMI, per le le connessioni HTTP ed i servlet è stato usato Flood.

8.2 XML

Oltre alle librerie per la gestione di file XML fornite dal SDK sono stati utilizzati i seguenti package:

- FOP: per l'implementazione del modulo di trasformazione XSL FO

- POI per l'implementazione del modulo di trasformazione dal formato XML al formato Microsoft OLE 2 Compound Document.

Entrambe le librerie sono state usate nel modulo di generazione di report: FOP per ottenere il formato PDF, POI per ottenere risultati in formato Excel.

8.3 Web Server e Servlet Container

Una importante modifica apportata all'architettura software del sistema INFEA è stata la realizzazione dell'indipendenza da funzionalita' di HTTP server e "JSP/servlet container" esterni. In precedenza queste funzioni erano svolte rispettivamente da Microsoft IIS e JRun. Nella attuale versione è stato incorporato il codice del server Jakarta Apache Tomcat release 4.0.6 che svolge entrambe le funzioni. Per le connessioni sicure è stato utilizzato il modulo Security Socket Layer (SSL) di Tomcat, i certificati sono stati generati in proprio.

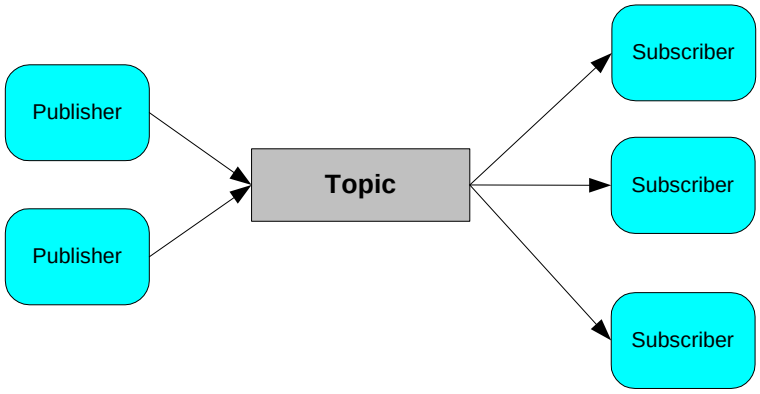
8.4 Gestore della base di dati

Sono stati sperimentati più gestori relazionali della base di dati per verificare l'aderenza alle specifiche di indipendenza del codice dal DBMS. Per i test sono stati utilizzati Oracle 7.1, Microsoft Acces (Versione per Windows 2000), diverse release di Microsoft SQL Server e alcune prove sono state fatte su MySQL. Attualmente nel server di produzione è installato Microsoft SQLServer2000.

8.5 Gestione degli eventi

Il sistema di gestione degli eventi è implementato adottando il paradigma MessageOriented Middleware (MOM). L'obiettivo è di creare canali (*topic*) di comunicazione bidirezionali; su questi canali i diversi client si scambiano *messaggi* rappresentati da documenti XML, ciascun messaggio contiene, oltre ai dati, anche le informazioni di *routing* (mittente destinatario, etc). Ciascun client si abbona (*subscribe*) ad uno o più topics e riceve o produce (*publish*) le informazioni in maniera *asincrona*.

Per la realizzazione è stata utilizzata la tecnologia *JMS* di Java implementando ed integrando nel codice un server per la gestione dei messaggi. Le librerie utilizzate sono quelle del progetto Joram relase 3.2.



Quando un utente accede al sistema informativo, il client da lui usato effettua il subscribe ai topics a lui assegnati e si pone in attesa 'silenziosa' della notifica di uno o più messaggi, quando un publisher pubblica un messaggio su uno dei topic l'utente viene immediatamente informato.

8.6 Altro

- Wusage 8.0: uno strumento per il monitoraggio e l'analisi degli accessi al server Web

8.7 Struttura delle directory

L'attuale radice Infea contiene 5 directory principali:

Infea

Bin

Lib

Conf

Sources

Webapps

Ciascuna delle directory contiene documenti o eseguibili dello “stesso tipo” a loro volta raggruppati in sottodirectory. I sorgenti sono contenuti tutti in “Sources”, le classi sono distribuite all’interno di “Webapps” che è così strutturata:

Webapps

Infea2001

InfeaAdmin

InfeaRedazione

Ciascuna sottodirectory contiene oltre alle classi Java del corrispondente modulo di interazione (utente, redattore, amministratore), anche le specifiche delle interfaccie utente. In “Lib” ci sono le librerie esterne, in “Conf” i file di configurazione (e i certificati digitali) ed in “Bin” gli eseguibili per la gestione del sistema.

8.8 Riferimenti

Di seguito riportiamo i riferimenti agli strumenti e librerie open source utilizzate nello sviluppo:

- Sun SDK: http://java.sun.com/
- Jakarta Tomcat: http://jakarta.apache.org/tomcat/index.html
- NetBeans: http://www.netbeans.org/
- FOP: http://xml.apache.org/fop/index.html
- POI: http://jakarta.apache.org/poi/index.html
- WinCVS: http://www.wincvs.org/
- Flood: http://httpd.apache.org/test/flood/

9 Progettazione logica e fisica della base di dati

In questo paragrafo è descritta la fase di progettazione logica e fisica della base di dati. Nel seguito viene descritta la trasformazione dello schema concettuale, nel modello dei dati relazionale. Per l'implementazione dello schema della base di dati è stato adottato il linguaggio SQL, perché esiste una definizione di standard da parte di organizzazioni internazionali (ANSI e ISO) e perché largamente adottato da molti sistemi di gestione di base di dati disponibili commercialmente. Questa scelta garantisce una buona portabilità in differenti ambienti senza generare dipendenze da specifici prodotti hardware e software.

9.1 Norme adottate nella realizzazione della base di dati.

1. Tutte le proprietà dello schema concettuale definite col tipo *codifica* sono state rappresentate nelle relazioni risultanti tramite valori numerici univoci. Nelle relazioni i nomi di tali proprietà hanno come prefisso la stringa *Cod*. La corrispondenza tra valore numerico e forma presentabile della proprietà (es. *10 = Dipartimento Ministeriale*) è mantenuta tramite un'unica relazione (la relazione *Codifica*). Questa scelta è stata adottata principalmente in funzione di una possibile variazione delle forme presentabili, in quanto vengono localizzati i possibili interventi di modifica (es. basta sostituire la relazione *Codifica* per avere la versione della base di dati in un'altra lingua). Inoltre tale soluzione consente di ridurre i tempi di trasferimento dei dati tra il Server della base di dati e le applicazioni in quanto le singole relazioni contenenti i valori codificati sono molto più compatte e infine consente di ridurre al minimo la ridondanza e quindi l'occupazione di spazio; tutte e tre le motivazioni determinano una maggiore efficienza del sistema. La scelta non penalizza l'usabilità, in quanto l'accesso alla base di dati non avverrà mai direttamente tramite il linguaggio nativo dello specifico DBMS, ma sempre tramite interfacce utenti che assumeranno il carico di trasformazione dei valori codificati internamente in valori presentati all'utente finale.

2. Tutti gli identificatori delle relazioni hanno un nome prefissato da *Id* e sono stati definiti di tipo stringa di 8 caratteri. I valori per tali proprietà consistono in un numero progressivo di 6 o 7 cifre prefissati da uno o due lettere che richiamano il nome dell'oggetto che identificano (es. EU000025 è l'identificatore di un Ente Universitario).

3. Tutte le proprietà di codifica (attributi il cui nome inizia con *Cod*) sono definiti sul tipo SQL *smallint*.

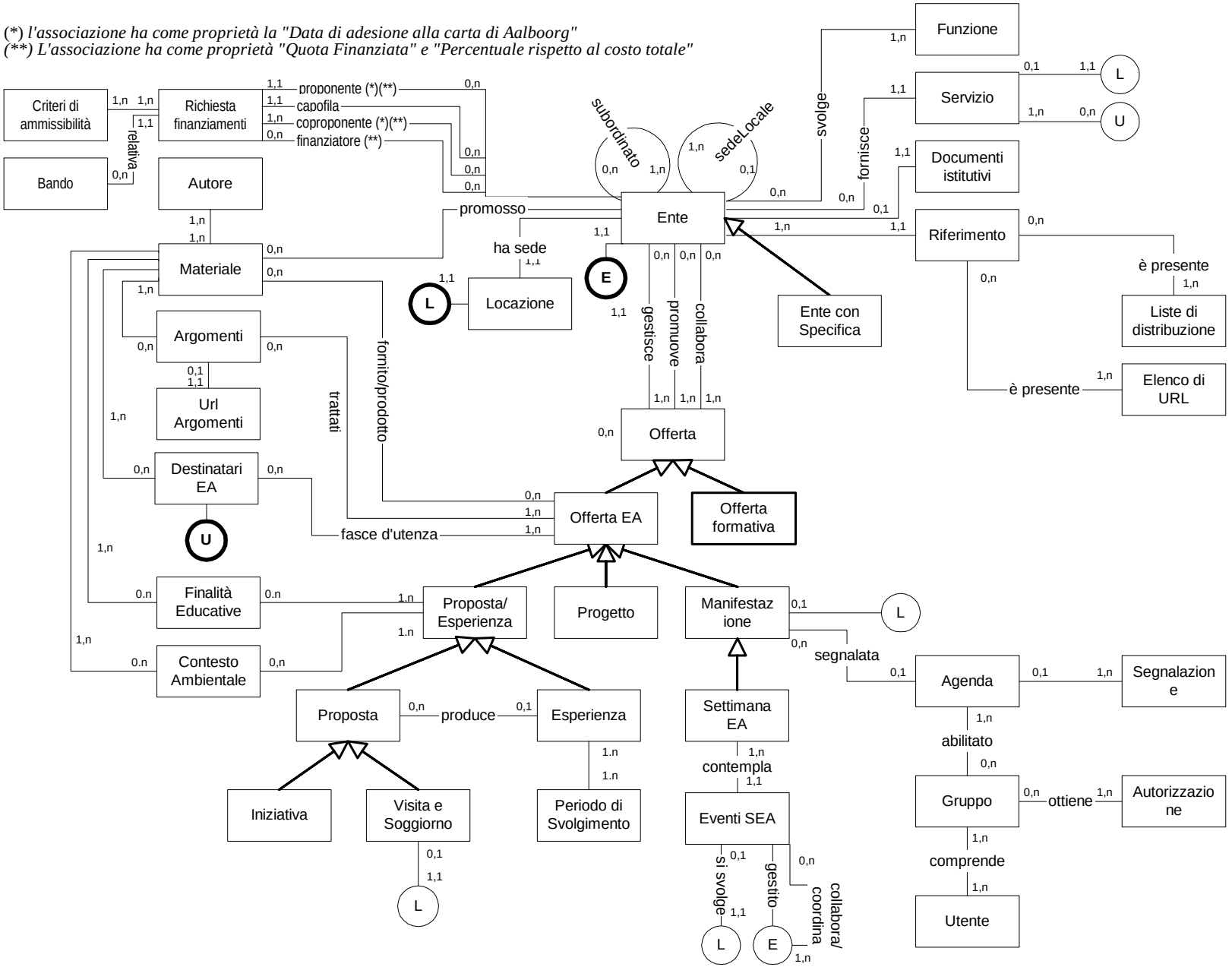
4. In alcune relazioni particolarmente importanti dal punto di vista implementativo sono state aggiunte, essenzialmente per problemi gestionali, le seguenti proprietà:
    - *DataPrimoInserimento* data e ora del primo inserimento della specifica istanza nella base di dati.
    - *DataUltimaModifica* data e ora dell'ultima modifica apportata ai valori della specifica istanza.

- *UtentePrimoInserimento* Identificativo dell'utente che ha sottoposto l'inserimento della specifica istanza.
- *UtenteProponente* Identificativo dell'utente che ha sottoposto l'ultima modifica della specifica istanza.
- *UtenteApprovante* Identificativo dell'amministratore che ha approvato l'inserimento, o l'ultima modifica della specifica istanza.
- *NoteAmministratore* attributo testuale per eventuali note ad uso dell'amministratore.

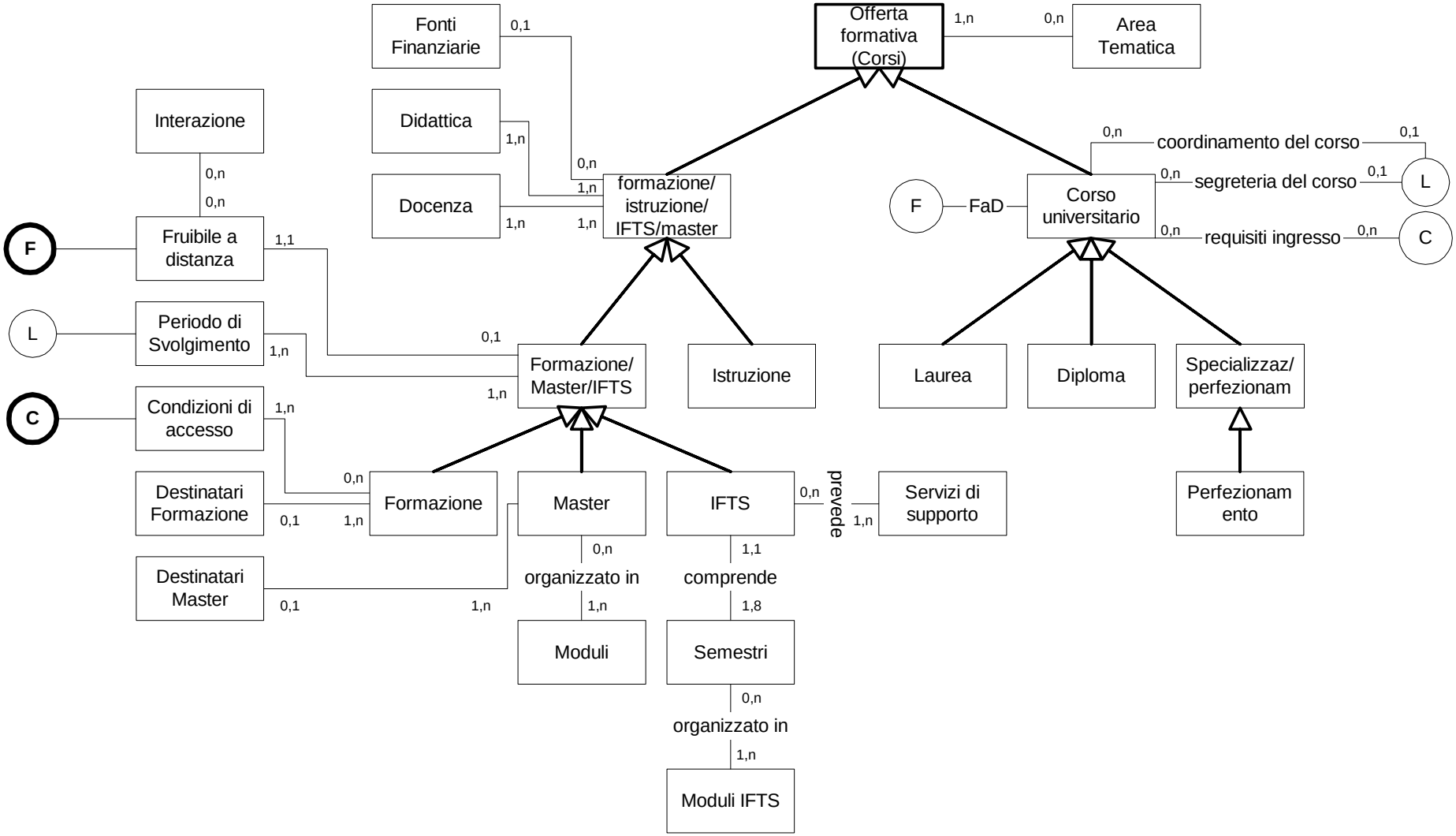
Queste proprietà sono accessibili solamente da utenti con ruolo di amministratore.

Nelle figure seguenti è riportato lo scheletro dello schema concettuale del sistema informativo SVS, per descrizioni dettagliate sulla fase di progettazione concettuale, si rimanda a rapporti in possesso di codesto Ministero.

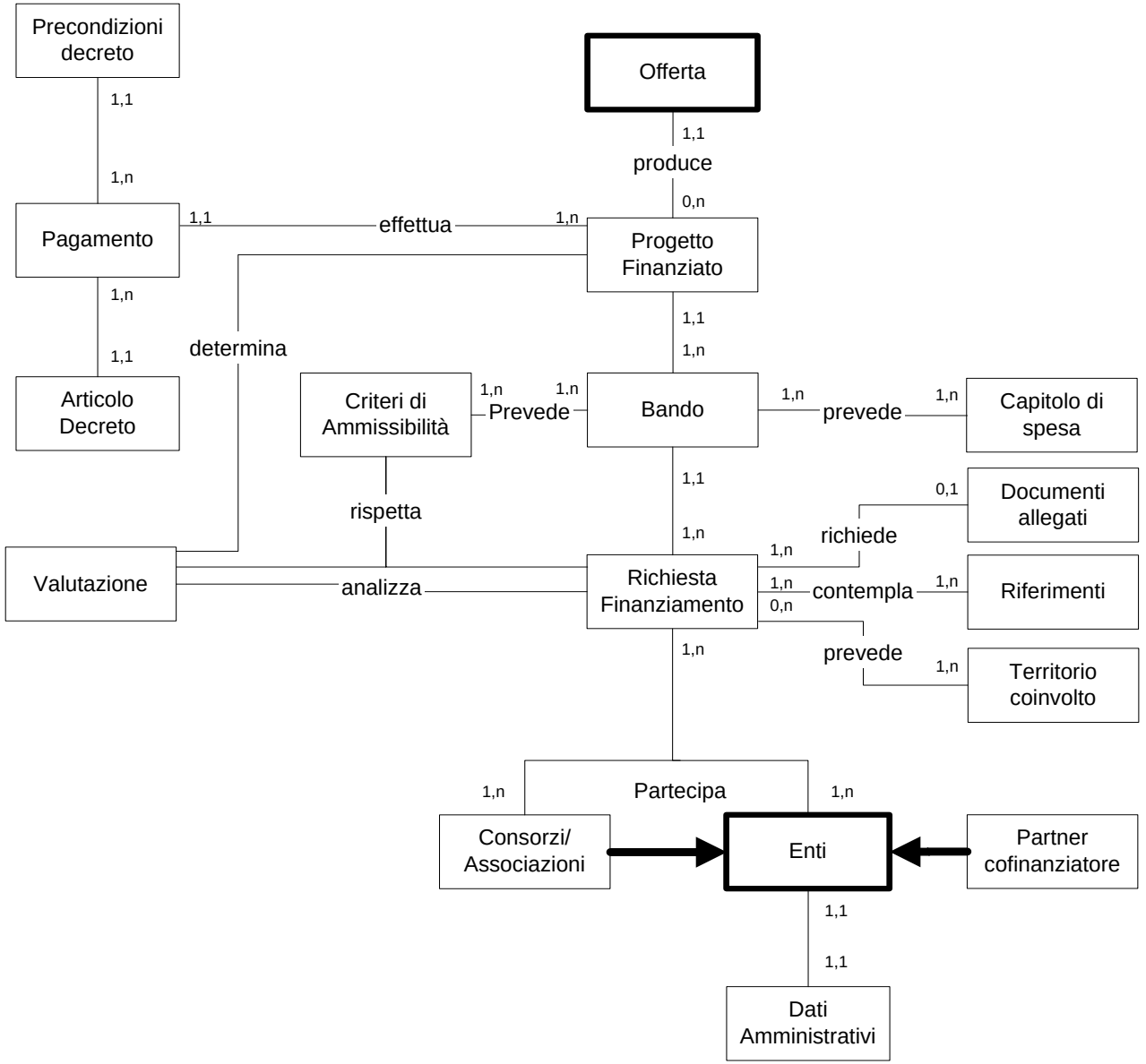
(*) l'associazione ha come proprietà la "Data di adesione alla carta di Aalboorg"
(**) L'associazione ha come proprietà "Quota Finanziata" e "Percentuale rispetto al costo totale"



Schema Concettuale della Base di dati



Schema Concettuale della Base di dati



Schema Concettuale della Base di dati

9.2 Trasformazione dello schema concettuale

Il primo passo della traduzione dal modello concettuale al modello relazionale è consistito nella risoluzione delle gerarchie IS-a.

9.2.1 Trasformazione della classe Ente

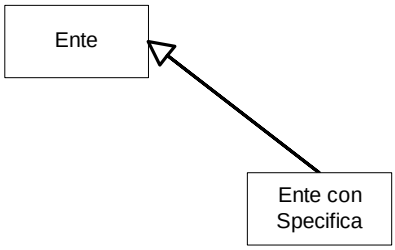


Figura 1. Gerarchia Enti dello schema concettuale

La gerarchia di Figura 1 è stata risolta, per ottimizzazione implementativa, definendo un'unica relazione *Enti*.

Di seguito compare la definizione della relazione corrispondente espressa in linguaggio SQL.

Relazione Enti

```
CREATE TABLE Enti (
      IdEnte char (12) NOT NULL ,
      Denominazione varchar (255) NOT NULL ,
      CodTipologiaIstituzionale smallint NOT NULL ,
      CodSpecificaTipologia smallint NULL ,
      Privato char (1) NULL ,
      SitoWeb varchar (255) NULL ,
      Descrizione text NULL ,
      Logo image NULL ,
      IdEnteSedeNazionale char (8) NULL ,
      CodStato smallint NOT NULL ,
      DataInserimento datetime NOT NULL ,
      DataModifica datetime NOT NULL ,
      IdGruppo char (12) NOT NULL ,
      IdUtenteInserimento char (12) NOT NULL ,
      IdUtenteModifica char (12) NOT NULL ,
      IdUtenteApprovante char (12) NOT NULL ,
      NoteAmministratore text NULL ,
      CONSTRAINT PK_Ente PRIMARY KEY  CLUSTERED (
            IdEnte
      )
)
```

CodStato rappresenta lo stato dell'oggetto rispetto alla sua fruibilità

	CodStato
01	Abilitato
02	Rifiutato
03	Sottoposto inserimento
04	Sottoposto cancellazione
05	In Progress

Rifiutato rappresenta la situazione in cui un oggetto sottoposto non è stato accettato dalla redazione poiché non soddisfa I requisiti per l'abilitazione, viene lasciato in questo stato per dare l'opportunità al proponente di modificarlo e di riproporlo.

In Progress rappresenta lo stato provvisorio di un oggetto che non è ancora stato sottoposto (azione save).

La tipologia di ente viene discriminata dall'attributo ***CodTipologiaIstituzionale*** che può assumere uno dei seguenti valori:

	CodTipologiaIstituzionale
01	Ente Pubblico Centrale
02	Ente Pubblico Territoriale
03	Ente Universitario
04	Servizio Culturale
05	Scuola
06	Ente o Istituto di Ricerca pubblico o privato
07	Area Servizio di pubblica utilità
08	Parte Sociale o enti di loro emanazione
09	Area Protetta
10	Terzo Settore
11	Organismo internazionale
12	Impresa
13	Agenzia/Authority

Privato segnala la natura dell'Ente, assume valore binario (vero, falso).
CodSpecificaTipologia assume valori che dipendono dal valore assunto da ***CodTipologiaIstituzionale*** della relazione *Enti*:

***CodTipologiaIstituzionale* = EntePubblicoCentrale** ➔

	CodSpecificaTipologia
01	Ministero economia e finanze
02	Ministero ambiente e tutela del territorio.
03	Presidenza Del Consiglio
04	Ministero affari esteri
05	Ministero interno
06	Ministero giustizia

07	Ministero attività produttive
08	Ministero istruzione, università e ricerca scientifica
09	Ministero lavoro e politiche sociali
10	Ministero difesa
11	Ministero politiche agricole e forestali
12	Ministero infrastrutture e trasporti
13	Ministero salute
14	Ministero beni culturali
15	Ministero comunicazioni

CodTipologiaIstituzionale = EntePubblicoTerritoriale ➔

	CodSpecificaTipologia
01	Regionale
02	Provinciale
03	Comunale
04	Interterritoriale
05	Centro di orientamento
06	Centro di formazione
07	Camera di commercio

CodTipologiaIstituzionale = EnteUniversitario ➔

	CodSpecificaTipologia
01	Rettorato
02	Facoltà
03	Scuola di specializzazione
04	Scuola di perfezionamento
05	Dipartimento
06	Istituto

Dato un ente universitario, non necessariamente occorre inserire tutta la gerarchia di enti (es. se interessa l'ente ScuolaDiSpecializzazione, non necessariamente occorre creare anche gli enti a cui esso è subordinato), ma si può utilizzare la classe Riferimenti. Lo stesso discorso vale per la segreteria studenti, ove presente. La distinzione tra università pubblica e privata viene risolta lasciando a livello di interfaccia utente questa caratteristica e creando a livello fisico 2 tipologie: UniversitàPubblica e UniversitàPrivata che verranno gestite opportunamente, in maniera trasparente per l'utente.

CodTipologiaIstituzionale = AreaProtetta ➔

	CodSpecificaTipologia
01	Oasi
02	Parco nazionale
03	Riserva naturale marina
04	Altri parchi

CodTipologiaIstituzionale = Scuola ➔

	CodSpecificaTipologia
01	Scuola dell'infanzia
02	Scuola elementare
03	Scuola media
04	Scuola secondaria

CodTipologiaIstituzionale = ServizioCulturale ➔

	CodSpecificaTipologia
01	Biblioteca
02	Centro di Documentazione
03	Centro di Educazione Ambientale
04	Museo
05	Orto botanico/giardino zoologico

CodTipologiaIstituzionale = AreaServiziDiPubblicaUtilità ➔

	CodSpecificaTipologia
01	Gestione energia (Enel,gas,…)
02	Gestione rifiuti
03	Telecomunicazioni
04	Sicurezza e Prevenzione
05	Trasporti
06	Sanità

CodTipologiaIstituzionale = TerzoSettore ➔

	CodSpecificaTipologia
01	Associazione Ambientalista
02	Enti Formativi
100	Altre Organizzazioni

CodTipologiaIstituzionale = Agenzia/Authority ➔

	CodSpecificaTipologia
01	**Locale**
02	Nazionale

CodTipologiaIstituzionale = PartiSocialiEdEntiDiLoroEmanazione ➔

	CodSpecificaTipologia
01	Associazione imprenditoriale
02	Associazione sindacale
03	Associazione professionale
04	Associazione di categoria

9.2.2 Trasformazione delle associazioni che coinvolgono la classe Ente

Le seguenti associazioni dell'entità Ente con se stesso (Figura 3) sono state modellate aggiungendo un attributo alla relazione Enti (***IdEnteSedeNazionale***) e definendo la relazione *EntiSovraordinanti*.

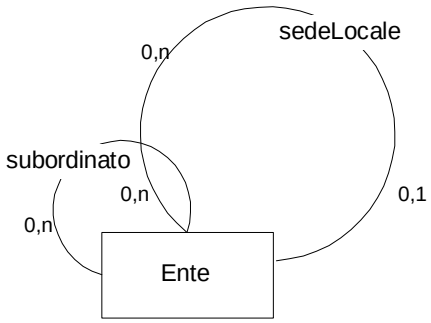


Figura 3. Associazioni di Ente con se stesso

Relazione EntiSovraordinanti

CREATE TABLE ***EntiSovraordinanti*** (
IdEnte char (12) NOT NULL ,
IdEnteSovraordinante char (12) NOT NULL ,
CONSTRAINT PK_ EntiSovraordinanti PRIMARY KEY CLUSTERED
(
IdEnte,EntiSovraordinane,
)
)

Tutte le associazioni fra *Ente* e *Offerta* (Figura 4) vengono modellate tramite la relazione *OffertaEnte*, in cui l'*IdOfferta* è relativa all'*Id* di una sottoclasse di *OffertaEa* o di *OffertaFormativa*.

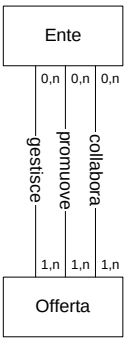


Figura 4.

Relazione OffertaEnte

```
CREATE TABLE OffertaEnte (
        IdOfferta char (12) NOT NULL ,
        IdEnte char (12) NOT NULL ,
        CodTipoAssociazione smallint NOT NULL ,
        CONSTRAINT PK_OffertaEnte PRIMARY KEY  CLUSTERED
        (
                IdEnte,
                IdOfferta,
                CodTipoAssociazione
        ),
        CONSTRAINT FK_OffertaEnte FOREIGN KEY
        (
                IdEnte
        ) REFERENCES dbo.Enti (
                IdEnte
        )
)
```

La relazione modella l'associazione multipla bidirezionale tra *Offerta* ed *Ente*.

La tipologia della associazione viene discriminata dall'attributo *CodTipoAssociazione* che può assumere uno dei seguenti valori:

	CodTipoAssociazione
01	Collabora
02	Promuove
03	Gestisce

Chiave Primaria:
IdOfferta+IdEnte+CodTipoAssociazione

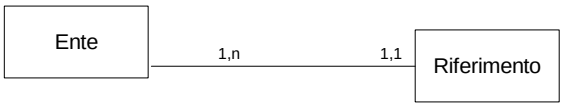


Figura 4a. Associazione Ente/Riferimento

L'associazione (1:n) tra la classe ***Ente*** e la classe ***Riferimento*** dello schema concettuale viene realizzata tramite la seguente relazione.

```
CREATE TABLE Riferimenti (
        IdEnte char (12) NOT NULL,
        NumProgressivo smallint NOT NULL,
        Ruolo varchar (255) NOT NULL ,
        Nome varchar (255) NULL,
        Telefono varchar (255) NULL,
        Fax varchar (255) NULL,
        Email varchar (255) NULL,
        SitoWeb varchar (255) NULL,
        Indirizzo varchar (255) NULL,
        Cap integer NULL,
        Citta varchar (255) NULL,
        SiglaProvincia char (2) NULL,
        CONSTRAINT PK_Riferimenti PRIMARY KEY  CLUSTERED (
                IdEnte, NumProgressivo
        ), CONSTRAINT FK_Riferimenti FOREIGN KEY
        (
                IdEnte
        )REFERENCES dbo.Enti (
                IdEnte
        )
)
```

I valori di *Indirizzo*, *Cap*, *Citta*, *SiglaProvincia* sono immessi solo se diversi da quelli della locazione dell'ente.

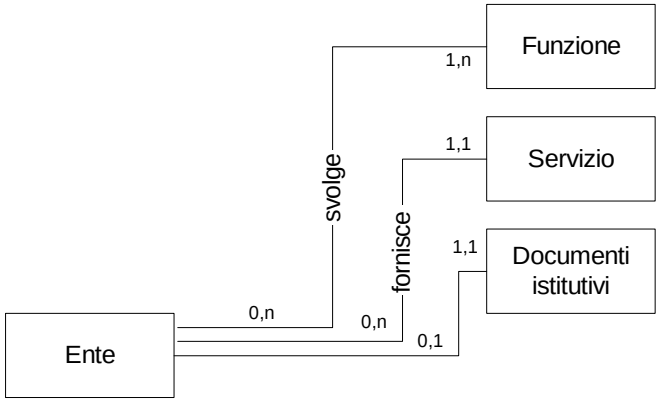


Figura 5.

Le classi *Funzione, Servizio, Documenti istitutivi* e le loro associazioni con la classe Ente (Figura 5) sono state realizzate tramite le relazioni *FunzioniEA, Servizi, DocumentiIstitutivi* (quest'ultima per eventuali scopi futuri, infatti l'associazione potrebbe diventare del tipo 1:n qualora si ritenesse di associare i documenti istitutivi reali per esempio in forma di file distinti).

Relazione FunzioniEA

```
CREATE TABLE FunzioniEA (
        IdEnte char (12) NOT NULL ,
        CodFunzione smallint NOT NULL ,
        CONSTRAINT PK_FunzioneEa PRIMARY KEY  CLUSTERED
        (
                IdEnte,
                CodFunzione
        ),
        CONSTRAINT FK_FunzioneEa FOREIGN KEY
        (
                IdEnte
        ) REFERENCES dbo.Enti (
                IdEnte
        )
)
```

	CodFunzione
01	Laboratorio territoriale
02	Centro esperienza
03	Coordinamento Regionale

Relazione Servizi

```
CREATE TABLE Servizi (
        IdServizio char (12) NOT NULL ,
        IdEnte char (12) NOT NULL ,
        CodTipologia smallint NOT NULL ,
        Contatto varchar (255) NULL ,
        Descrizione text NULL ,
        Iscrizione char (1) NOT NULL ,
        IdLocazione char (8) NOT NULL ,
        NoteAmministratore text NULL ,
        CONSTRAINT PK_Servizi PRIMARY KEY  CLUSTERED (
                IdServizio
        ),
        CONSTRAINT FK_Servizi FOREIGN KEY  (
                IdEnte
```

```
    ) REFERENCES dbo.Enti (
        IdEnte
    )
)
```

Questa relazione rappresenta i Servizi e l'associazione (1:n). con gli *Enti* tramite l'attributo *IdEnte*.

	CodTipologiaServizio
01	**Consultazione,**
02	**Guida alla ricerca di materiali, informazioni.**
03	**Supporto metodologico per la progettazione**
04	**Facilitazioni logistiche**
05	**Promozione informativa**
06	**Organizzazione di incontri (mostre, convegni, conferenze)**
07	**Assistenza per l'analisi e l'elaborazione. di dati e informazioni.**

La proprietà ***IdLocazione*** rappresenta l'associazione tra la classe ***Servizio*** e la classe ***Locazione***.dello schema concettuale, il valore assunto è uguale al valore di ***IdServizio*** o di ***IdEnte*** a seconda che la locazione del servizio coincide con quella dell'ente oppure no.

La proprietà ***IdEnte*** rappresenta l'associazione "fornisce" tra la classe ***Ente*** e la classe ***Servizio***.dello schema concettuale.

Relazione DocumentiIstitutivi

```
CREATE TABLE DocumentiIstitutivi (
    IdEnte char (12) NOT NULL ,
    IdDocumento char (12) NULL ,
    Documento text NOT NULL ,
    CONSTRAINT PK_DocumentiIstitutivi PRIMARY KEY  CLUSTERED
    (
        IdEnte
    ),
    CONSTRAINT FK_DocumentiIstitutivi FOREIGN KEY
    (
        IdEnte
    ) REFERENCES dbo.Enti (
        IdEnte
    )
)
```

Questa relazione stralci di documenti relativi agli aspetti giuridici dell'*Ente.* Come già detto, questa relazione viene tenuta per eventuali usi futuri (cioè migrazione verso l'associazione 1:n). Per il momento, data l'associazione 1:1 non sarebbe necessaria la proprietà *IdDocumento*, che viene tuttavia mantenuta per usi futuri (N.B. in caso di associazione 1:n *IdDocumento* dovrà far parte della chiave primaria).

9.2.3 Trasformazione della classe Offerta

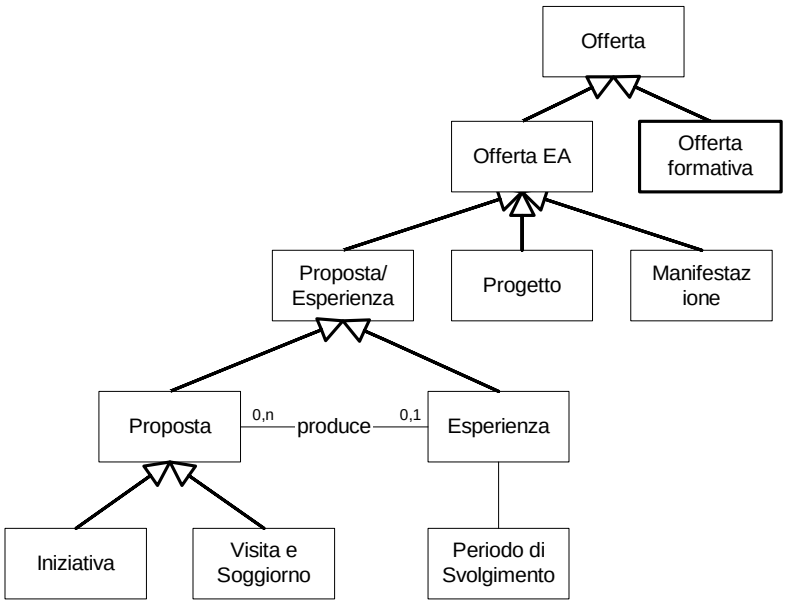


Figura 6

La gerarchia di Figura 6 è stata risolta definendo una relazione *OffertaEA,* che rappresenta il mondo Andrea e una relazione *OffertaFormativa* (*Corsi*) che comprende il mondo Anfora. Non si è ritenuto utile definire una relazione *Offerta* con le proprietà comuni delle due relazione, poiché, analizzando il traffico delle transazioni, non esistono casi in cui le due relazioni vengono coinvolte contemporaneamente, escluso per interrogazioni relative all'associazione di queste con l'Ente o relative a tutte le offerte a prescindere dalla loro tipologia (in questo caso l'operazione viene risolta tramite una Union tra gli attributi comuni). Per ciascuna delle classi è stata definita una relazione ad eccezione di *Proposta/Esperienza* in quanto il suo unico attributo ha la funzione di distinguere la *Proposta* dall'*Esperienza*. Per lo stesso motivo non viene definita una relazione *Proposta* in quanto la sua unica funzione sarebbe quella di discriminare tra *Iniziativa* e *Visita/Soggiorno* (conterrebbe solo gli ID delle differenti offerte e quindi costringerebbe comunque a fare un join con la relazione opportuna per ottenere informazioni specifiche della Proposta, con questa soluzione si elimina la necessità di un join tra tre relazioni). Poiché la classe *Esperienza* è caratterizzata da un periodo di svolgimento che può essere composto da più anni non consecutivi e, tale periodo, deve potere essere soggetto ad interrogazioni è stato ritenuto opportuno introdurre una nuova relazione *PeriodoDiSvolgimento* che riporti per ciascuna esperienza i periodi nei quali si è svolta. Lo schema diventa allora quello di Figura 7.

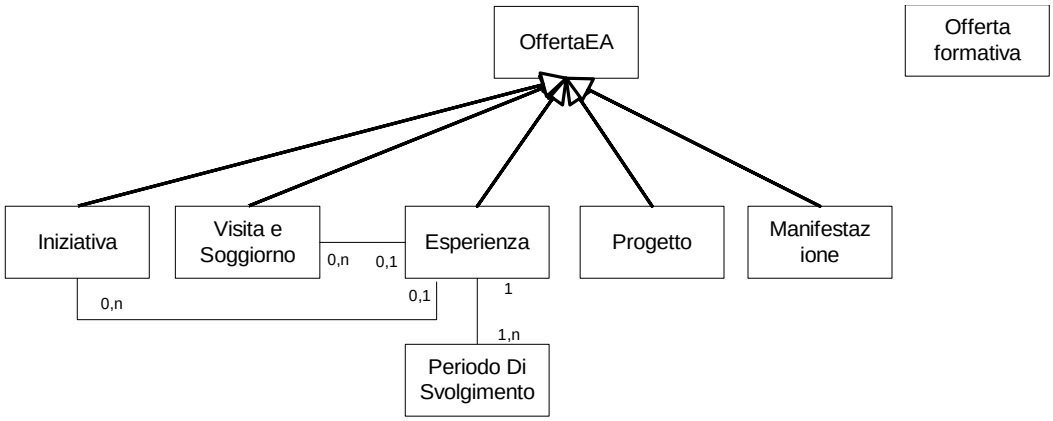


Figura 7

Il nome della relazione associata alla classe OffertaEA è ***Offerta***, mentre per la classe Offerta Formativa è stata definita la relazione ***Corsi***.

Relazione Offerta

CREATE TABLE **Offerta** (
IdOfferta char (12) NOT NULL ,
Titolo varchar (255) NOT NULL ,
CodTipologia smallint NOT NULL ,
CodTipologiaSpecifica smallint NOT NULL ,

Descrizione text NULL ,
CodStato `smallint NOT NULL ,`
DataInserimento `datetime NOT NULL ,`
DataModifica `datetime NOT NULL ,`
IdGruppo `char (12) NOT NULL ,`
IdUtenteInserimento `char (12) NOT NULL ,`
IdUtenteModifica `char (12) NOT NULL ,`
IdUtenteApprovante `char (12) NOT NULL ,`
NoteAmministratore text NULL ,
CONSTRAINT PK_Offerta PRIMARY KEY CLUSTERED
(
IdOfferta
)
)

Questa relazione contiene tutte le offerte. La tipologia di offerta viene discriminata dall'attributo *CodTipologiaOfferta* che può assumere uno dei seguenti valori:

	CodTipologia

01	Esperienza
02	Manifestazione
03	Progetto
04	Iniziativa
05	Visita o Soggiorno

Si è preferito non definire una relazione per la classe Proposta/Esperienza in quanto dotata di un unico attributo per la distinzione tra Proposta ed Esperienza. Le proprietà ***DataUltimaModifica*** e ***DataInserimento*** sono utili per la gestione del sistema (esempio per poter segnalare le modifiche o le novità).

Chiave Primaria:
IdOfferta

Relazione Iniziativa

CREATE TABLE **Iniziativa** (
IdOfferta char (12) NOT NULL ,
CodTipologia smallint NOT NULL ,
Referente varchar (255) NULL ,
SitoWeb varchar (255) NULL ,
CONSTRAINT PK_Iniziativa PRIMARY KEY CLUSTERED (
IdOfferta
), CONSTRAINT FK_ Iniziativa FOREIGN KEY
(IdOfferta
) REFERENCES dbo.Offerta (
IdOfferta
)

)
Questa relazione contiene tutte le Iniziative. . Il campo IdIniziativa è chiave primaria e contiene lo stesso valore del campo IdOfferta della generica offerta. La tipologia di iniziativa viene discriminata dall'attributo *CodTipologiaIniziativa* che può assumere uno dei seguenti valori:

	CodTipologia
01	Città dei bambini
02	Percorso didattico
03	Progetto educativo
04	Promozione
05	Ricerca educativa
06	Aggiornamento

Nota: Aggiornamento si riferisce alle attività di aggiornamento per insegnanti.

Chiave Primaria:
IdIniziativa

Relazione Visita/Soggiorno

CREATE TABLE **VisitaSoggiorno** (

IdOfferta char (12) NOT NULL ,
CodTipologia smallint NOT NULL ,
Durata varchar (255) NULL ,
Contatto varchar (255) NULL ,
CodIscrizione varchar (255) NULL ,
Accompagnatore char (1) NULL ,
MaxPersone smallint NULL ,
IdLocazione char (8) NULL ,
CONSTRAINT PK_VisitaSoggiorno PRIMARY KEY CLUSTERED
(
IdOfferta
) , CONSTRAINT FK_VisitaSoggiorno FOREIGN KEY
(IdOfferta
) REFERENCES dbo.Offerta (
IdOfferta
)
)

Questa relazione contiene tutte le Visite/Soggiorno.. Il campo IdVisita è chiave primaria e contiene lo stesso valore del campo IdOfferta della generica offerta La tipologia viene discriminata dall'attributo *CodTipologia* che può assumere uno dei seguenti valori:

	CodTipologia
01	Campo scuola
02	Escursione
03	Soggiorno
04	Soggiorno estivo
05	Visita guidata

Il tipo di iscrizione viene discriminata dall'attributo *CodIscrizione* che può assumere uno dei seguenti valori:

	CodIscrizione
01	Individuale
02	Di gruppo

L'attributo accompagnatore assume un valore di verità (si/no)

La proprietà ***IdLocazione*** rappresenta l'associazione tra la classe ***VisitaSoggiorno*** e la classe ***Locazione***.dello schema concettuale ed assume il valore di *IdVisita* oppure di *IdEnte* se la locazione corrisponde a quella dell'ente.

Chiave Primaria:
IdVisita

Relazione Esperienza

CREATE TABLE **Esperienza** (

```
        IdOfferta char (12) NOT NULL ,
        IdProposta char (8) NOT NULL ,
        CodTipologia smallint NOT NULL ,
        CodTipologiaProposta smallint NOT NULL ,
        Referente varchar (255) NULL ,
        NomeFileLocale varchar (255) NULL ,
        UrlFileLocale  varchar (255) NULL ,
        DataFileLocale  datetime NULL ,
        DimensioneFileLocale  int NULL ,
        CONSTRAINT PK_Esperienza  PRIMARY KEY  CLUSTERED
        (
                IdOfferta
        ) , CONSTRAINT FK_Esperienza FOREIGN KEY
        (       IdOfferta
        ) REFERENCES dbo.Offerta (
                IdOfferta
        )
)
```

Questa relazione contiene tutte le Esperienze. Il campo IdEsperienza è chiave primaria e contiene lo stesso valore del campo IdOfferta della generica offerta La tipologia di esperienza viene discriminata dall'attributo *CodTipologiaEsperienza* che può assumere uno dei seguenti valori:

	CodTipologia
01	Campo scuola
02	Città dei bambini
03	Escursione
04	Percorso didattico
05	Promozione
06	Ricerca educativa
07	Soggiorno
08	Soggiorno estivo
09	Progetto educativo
10	Visita guidata

L' *IdProposta* riferisce un'offerta che è nella relazione *Iniziative* o *VisiteSoggiorni*. Poiché il periodo di svolgimento può essere composto da più anni non consecutivi all'interno dei quali è necessario fare delle ricerche una soluzione può essere inserire gli anni in una relazione a parte (PeriodoDiSvolgimento).

Il *CodTipologiaProposta* serve a discriminare la tipologia della proposta, nel caso in cui un'esperienza è correlata ad una proposta e può assumere uno dei seguenti valori.

	CodTipologiaProposta
04	Iniziativa
05	Visita o Soggiorno

Chiave Primaria:

IdEsperienza

Relazione PeriodoDiSvolgimento

CREATE TABLE **PeriodoDiSvolgimento** (
IdOfferta char (12) NOT NULL ,
AnnoIniziale smallint NOT NULL ,
AnnoFinale smallint NULL ,
CONSTRAINT PK_PeriodoDiSvolgimento PRIMARY KEY CLUSTERED
(
IdEsperienza,
AnnoIniziale
) , CONSTRAINT FK_PeriodoDiSvolgimento FOREIGN KEY
(IdOfferta
) REFERENCES dbo.Offerta (
IdOfferta
)
)

- Se l'esperienza si svolge in un solo anno questo è riportato nel campo AnnoIniziale
- Se l'esperienza si svolge in un periodo di anni consecutivi l'anno iniziale e quello finale sono riportati rispettivamente nei campi AnnoIniziale ed AnnoFinale
- Se l'esperienza si svolge in più anni non consecutivi ciascun anno viene riportato nell'attributo AnnoIniziale
- Si possono avere combinazioni delle precedenti possibilità.

Chiave Primaria:
IdEsperienza+AnnoIniziale

Relazione Manifestazione

CREATE TABLE **Manifestazione** (
IdOfferta char (12) NOT NULL ,
CodTipologia smallint NOT NULL ,
Contatto varchar (255) NULL ,
DataInizio datetime NOT NULL **,**
DataFine datetime NOT NULL ,
IdLocazione char (8) NULL ,
SitoWeb varchar (255) NULL ,
UrlFileLocale varchar (255) NULL ,
NomeFileLocale varchar (255) NULL ,
DataFileLocale datetime NULL,
DimensioneFileLocale int NULL,
CONSTRAINT PK_Manifestazione PRIMARY KEY CLUSTERED
(
IdOfferta

```
        ) , CONSTRAINT FK_Manifestazione FOREIGN KEY
        (       IdOfferta
        ) REFERENCES dbo.Offerta (
                IdOfferta
        )
)
```

Questa relazione contiene tutte le Manifestazioni. . Il campo IdManifestazione è chiave primaria e contiene lo stesso valore del campo IdOfferta della generica offerta La tipologia di manifestazione viene discriminata dall'attributo *CodTipologiaManifestazione* che può assumere uno dei seguenti valori:

	CodTipologia
01	Campagna
02	Concorso
03	Conferenza/Convegno
04	Seminario
05	Mostra
06	Settimana ambientale

La proprietà ***IdLocazione*** rappresenta l'associazione tra la classe ***Manifestazione*** e la classe ***Locazione***.dello schema concettuale. *IdLocazione* non è obbligatorio per questa relazione. Per il valore 06 (Settimana ambientale) occorre indagare meglio.

Chiave Primaria:
IdManifestazione

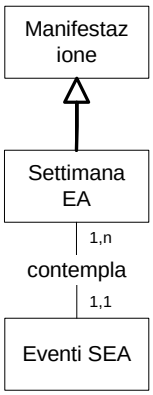


Figura 8

La gerarchia di Figura 8 è stata risolta definendo la relazione *EventiSea*, infatti la classe *Settimana EA* era stata introdotta a livello concettuale per via dell'associazione con *EventiSea*.

Relazione EventiSEA

CREATE TABLE.***EventiSEA*** (
IdSEA char (12) NOT NULL ,
IdEvento char (12) NOT NULL ,
IdEnte char (12) NOT NULL ,
Titolo varchar (255) NOT NULL ,
Contatto varchar (255) NOT NULL ,
Descrizione text NULL ,
IdLocazione char (8) NULL ,
CONSTRAINT PK_Sea PRIMARY KEY CLUSTERED
(
IdSea, IdEvento,
) , CONSTRAINT FK_Sea FOREIGN KEY
(IdSea
) REFERENCES dbo.Manifestazione (
IdManifestazione
)
)

IdLocazione assume lo stesso valore di *IdEnte* o di *IdEvento* a secondo della sede di svolgimento. Occorre indagare meglio su questa relazione alla luce dell'esperienza della III settimana Ea. IdSea identifica

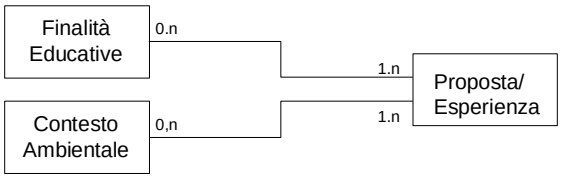


Figura 9

Le classi *ContestoAmbientale* e *FinalitàEducative* e le associazioni con la classe Proposta/Esperienza vengono realizzate tramite le relazioni *ContestoAmbientale* e *FinalitàEducative.*

Relazione ContestoAmbientale

```
CREATE TABLE ContestoAmbientale (
        IdOfferta char (12) NOT NULL ,
        CodContesto smallint NOT NULL ,
        CONSTRAINT PK_ContestoAmbientale PRIMARY KEY  CLUSTERED
        (
                IdOfferta,
                CodContesto
        )
)
```

	CodContesto
01	**Ambiente antropizzato**
02	**Acque dolci**
03	**Ambiente litoraneo**
04	**Ambiente marino**
05	**Ambiente roccioso**
06	**Ambiente urbano**
07	**Bosco**
08	**Prato**
09	**Zone umide**

Relazione FinalitaEducativa

```
CREATE TABLE FinalitaEducativa (
        IdOfferta char (12) NOT NULL ,
        CodFinalita smallint NOT NULL ,
        CONSTRAINT PK_FinalitaEducativa PRIMARY KEY  CLUSTERED
        (
                IdOfferta,
```

CodFinalita

)

)

	CodFinalita
01	**Conoscere i problemi dell'ambiente**
02	**Conoscere l'ambiente**
03	**Fare esperienza nell'ambiente**
04	**Intervenire sull'ambiente**
05	**Riflettere sull'educazione ambientale**

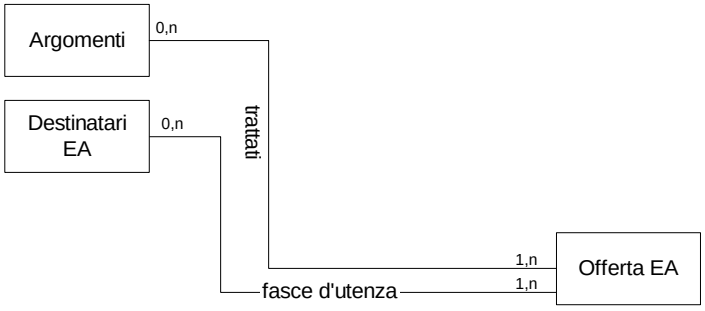


Figura 10

Le classi e le associazioni di Figura 10 sono realizzate con le seguenti relazioni. N.B.

Relazione DestinatariEA

CREATE TABLE ***DestinatariEA*** (
IdOfferta char (12) NOT NULL ,
CodDestinatari smallint NOT NULL ,
CONSTRAINT PK_DestinatariEA PRIMARY KEY CLUSTERED
(
IdOfferta,
CodDestinatari,
)
)

Contiene i destinatari delle istanze della classe Offerta EA, Materiali e Servizi. L'attributo ***IdOffertaEAMaterialeServizio*** permette di correlare gli elementi della relazione con gli elementi di una delle tre relazioni. I valori degli identificatori sono disgiunti.

	CodDestinatari
01	**Bambini**
02	**Giovani**
03	**Adulti**
04	**Anziani**
05	**Tutti**

Relazione Argomenti

CREATE TABLE.***Argomenti*** (
IdOfferta char (12) NOT NULL ,
CodArgomento smallint NOT NULL ,
IdArgomento smallint NOT NULL,
CONSTRAINT PK_Argomenti PRIMARY KEY CLUSTERED
(
IdOfferta,
CodArgomento
)
)

La relazione descrive la classe Argomenti associata ad OffertaEA e a Materiali. La definizione degli argomenti è quella del Thesaurus di ANDREA. *IdArgomento* è l'identificatore univoco presente in Andrea, *CodArgomento* è l'identificatore univoco creato nella costruzione dell'albero del thesaurus.

CREATE TABLE.***UrlsArgomenti*** (
IdArgomento smallint NOT NULL ,
Url varchar (255),
CONSTRAINT PK_UrlArgomenti PRIMARY KEY CLUSTERED
(
IdArgomento
)
)

Per Alcuni argomenti esiste una descrizione associata tramite url.

Trasformazione della classe Materiali

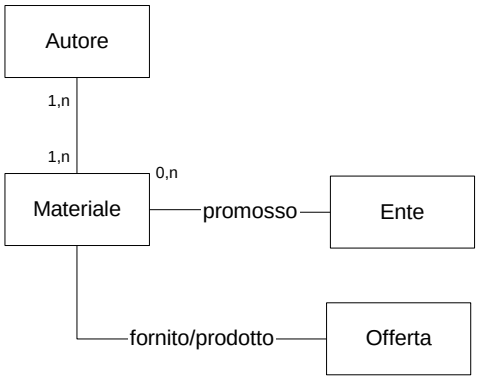


Figura 11

Le classi e le associazioni di Figura 11 sono state implementate tramite le seguenti relazioni.

Relazione Materiali

CREATE TABLE **Materiali** (
IdMateriale char (12) NOT NULL ,
Titolo varchar (255) NOT NULL ,
CodTipologia smallint NOT NULL ,
Abstract text NULL ,
CodLingua smallint NOT NULL ,
Editore varchar (255) NULL ,
AnnoEdizione smallint NULL ,
LuogoEdizione varchar (255) NULL ,
AltriRiferimentiBibliografici varchar (255) NULL ,
CollocatoInInfea char(1) NOT NULL ,
UrlFileLocale varchar (255) NULL ,
NomeFileLocale varchar (255) NULL ,
DataFileLocale datetime NULL,
DimensioniFileLocale int NULL,
CodStato `smallint NOT NULL ,`
DataInserimento `datetime NOT NULL ,`
DataModifica `datetime NOT NULL ,`
IdGruppo `char (12) NOT NULL ,`
IdUtenteInserimento `char (12) NOT NULL ,`
IdUtenteModifica `char (12) NOT NULL ,`
IdUtenteApprovante `char (12) NOT NULL ,`
NoteAmministratore text NULL ,
CONSTRAINT PK_Materiali PRIMARY KEY CLUSTERED (
IdMateriale
))

Questa relazioni contiene tutti i materiali, che vengono identificati univocamente tramite l'attributo IDMateriale che è costituito da una sequenza di 8 caratteri, come per tutti gli Id. Nella codifica dei valori per tale Id si usano i primi 3 caratteri per riferire l'oggetto (es. AUD per Audiovisivo, GIO per Gioco, etc) e gli atri 5 come numero progressivo.
L'attributo *CodTipologiaMateriale* può assumere uno dei seguenti valori:

	CodTipologiaMateriale
01	Audiovisivo
02	Gioco
03	Ipermedia
04	Pannello per mostra
05	Pubblicazione

L'attributo *CodLingua* può assumere uno dei seguenti valori:

	CodLingua
01	Italiano
02	Inglese
03	Francese
04	Spagnolo
05	Tedesco
06	…….

Relazione Autori

CREATE TABLE **Autori** (
IdMateriale char (12) NOT NULL ,
Autore varchar (255) NOT NULL ,
)

Questa relazione contiene tutti i dati relativi agli autori dei materiali. Per semplicità di realizzazione non è stata normalizzata la doppia associazione multipla, per cui la relazione può contenere valori duplicati e per questo motivo non viene definita nessuna chiave primaria.

Relazione MaterialiEnti

CREATE TABLE **MaterialiEnti** (
IdMateriale char (12) NOT NULL ,
IdEnte char (12) NOT NULL ,
CONSTRAINT PK_MaterialiEnti PRIMARY KEY CLUSTERED
(
IdEnte,
IdMateriale
)
)

Questa relazione rappresenta l'associazione "*promosso*" dello schema concettuale.

Relazione OffertaMateriali

CREATE TABLE **OffertaMateriali** (
IdMateriale char (12) NOT NULL ,
IdOfferta char (12) NOT NULL ,
CONSTRAINT PK_OffertaMateriali PRIMARY KEY CLUSTERED
(
IdMateriale,
IdOfferta
)
)

Questa relazione rappresenta l'associazione "*fornito/prodotto*" dello schema concettuale. L'interpretazione dell'associazione avviene tramite la relazione con cui si fa join (*fornito* nel caso di join con un'Offerta di tipo Proposta e *prodotto* nel caso di altre tipologie di Offerta). Con questa modellazione è possibile associare dei materiali a qualsiasi Offerta, per cui occorre prevedere a livello di interfaccia eventuali vincoli (se esistono per esempio delle Offerte che non producono materiali).

9.2.4 Trasformazione della classe Locazione

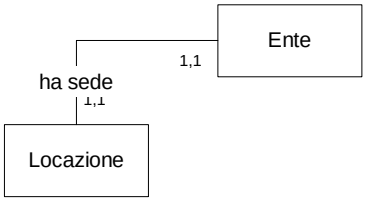


Figura 4b. Associazione Ente/Locazione

L'associazione "ha sede" tra la classe ***Ente*** e la classe ***Locazione***.dello schema concettuale viene realizzata tramite la corrispondenza IdEnte ed IdLocazione della relazione Locazione.
Questa relazione si tiene separata sebbene potrebbe confluire, per la classe Ente, nella relazione Enti essendo l'associazione che ad essa collegata del tipo (1:1) in entrambi i versi. Per uniformità di trattamento e di gestione con le classe *Servizi*, *Materiali*, ecc., poiché in questo caso l'associazione è opzionale, si preferisce la seguente scelta.

Relazione Locazione

CREATE TABLE **Locazione** (
IdLocazione char (12) NOT NULL ,
Stato varchar (255) NULL ,
SiglaProvincia char(2) NOT NULL ,

Citta varchar (255) NOT NULL ,
Cap integer NOT NULL,
Indirizzo varchar (255) NOT NULL ,
Collegamenti text NULL ,
Strutture text NULL,
BarriereArchitettoniche Char (1) NOT NULL ,
NoteAmministratore text NULL ,
CONSTRAINT PK_Locazione PRIMARY KEY CLUSTERED
(
IdLocazione
)
)

Questa relazione contiene tutte le specifiche delle locazioni, per cui l'*IdLocazione* corrisponde all'*IdEnte* se la locazione è relativa all'*Ente*, l'*IdServizio* se relativa al servizio, l'*IdMateriale* se relativo al *Materiale*, etc. Tutti i dati di questa tabella (escluso la chiave) possono essere duplicati per gli oggetti del DB che hanno l'identica specifica locazione. Questa scelta semplifica la sua gestione, infatti se supportassimo l'associazione multipla, dovremmo complicare notevolmente la gestione dell'aggiornamento (prevedendo ad esempio che l'aggiornamento dell'indirizzo riguarda solo alcuni degli oggetti che puntavano al vecchio indirizzo, etc).

Relazione ContestoTerritoriale

CREATE TABLE ***ContestoTerritoriale*** (
CodAreaGeografica smallint NOT NULL ,
CodRegione smallint NOT NULL ,
CONSTRAINT PK_ContestoTerritoriale PRIMARY KEY CLUSTERED
(
*CodAreaGeografica**,** CodRegione*
)
)

Questa relazione contiene i contesti territoriali, così come definita da *CodAreaGeografica,* tramite i quali selezionare le ennuple nella relazione *Locazione.* Si è preferito tenere separata questa relazione per fornire la possibilità di definire ulteriori contesti per cui raggruppare le ennuple della relazione *Locazione*.

La proprietà ***CodAreaGeografica*** assume i seguenti valori:

	CodAreaGeografica
01	**Italia centrale**
02	**Italia insulare**
03	**Italia meridionale**
04	**Italia nord occidentale**
05	**Italia nord orientale**

Relazione Regioni

CREATE TABLE Regioni (
CodRegione smallint NOT NULL ,
Regione varchar (30) NOT NULL ,
CONSTRAINT PK_Regioni PRIMARY KEY CLUSTERED
(
CodRegione
)
)

Relazione Provincie

CREATE TABLE Provincie (
SiglaProvincia char(2) NOT NULL ,
CodRegione smallint NOT NULL ,
Provincia varchar (30) NOT NULL ,
CONSTRAINT PK_Provincie PRIMARY KEY CLUSTERED
(
CodProvincia
)
)

9.2.5 Trasformazione della classe Agenda

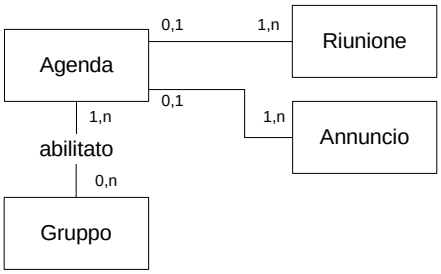


Figura 12

Le classi e le associazioni di Figura 12 sono state implementate tramite le seguenti relazioni.

Relazione Agenda

CREATE TABLE***.Agenda (***
IdOfferta char (8) NOT NULL ,
CodTipologia smallint NOT NULL ,
CodStato `smallint NOT NULL ,`
IdGruppo `char (12) NOT NULL ,`

IdUtenteApprovante `char (12) NOT NULL ,`
DataInserimentoInAgenda `datetime NOT NULL ,`
DataScadenza datetime NOT NULL ,
NoteAmministratore text NULL ,
CONSTRAINT PK_Agenda PRIMARY KEY CLUSTERED
(
IdEvento
)
)

Questa relazione contiene gli eventi in Agenda. I vari eventi sono identificati da ***IdEvento*** e fanno riferimento tramite la proprietà ***CodTipologiaEvento*** alla Manifestazioni o Segnalazioni.. La relazione viene popolata dall'Activity Manager dell'agenda selezionando gli eventi significativi per la segnalazione. La data di scadenza identifica il periodo di validità in Agenda che in generale può essere diverso dalle date di validità dell'evento. La propietà ***CodTipologiaEvento*** assume i seguenti valori.

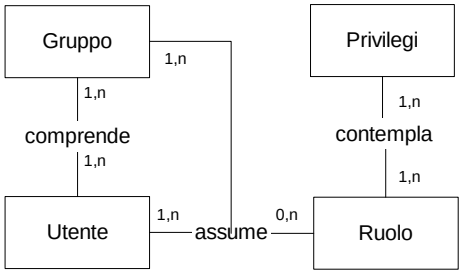
	CodTipologia
01	Manifestazione
02	Segnalazione

Relazione Segnalazioni

CREATE TABLE ***Segnalazioni*** (
IdSegnalazione char (12) NOT NULL ,
Titolo varchar (255) NOT NULL ,
Descrizione text NULL ,
SitoWeb varchar (255) NULL ,
UrlFileLocale varchar (255) NULL ,
NomeFileLocale varchar (255) NULL ,
DataFileLocale datetime NULL,
DimensioniFileLocale int NULL,
DataInizio datetime NOT NULL ,
DataFine datetime NULL ,
CodStato `smallint NOT NULL ,`
DataInserimento `datetime NOT NULL ,`
DataModifica `datetime NOT NULL ,`
IdGruppo `char (12) NOT NULL ,`
IdUtenteInserimento `char (12) NOT NULL ,`
IdUtenteModifica `char (12) NOT NULL ,`
IdUtenteApprovante `char (12) NOT NULL ,`
NoteAmministratore `text NULL ,`
CONSTRAINT PK_Segnalazioni PRIMARY KEY CLUSTERED
(
IdSegnalazione
)
)

Questa relazione contiene gli eventi da segnalare in agenda che non rientrano nelle categorie previste dalla classe Offerta (esempio: “il 28 cm il ministro inaugura il nuovo LT di Piazza Armerina”). L’eventuale Locazione dell’evento compare in forma descrittiva nella Descrizione.

9.2.6 Trasformazione delle classe Utenti per la gestione di utenti, gruppi e ruoli



Le classi e le associazioni in figura sono implementate tramite le seguenti relazioni.

Relazione Utenti

CREATE TABLE ***Utenti*** (
IdUtente char (12) NOT NULL ,
IdGruppoPrimario char (12) NOT NULL ,
IdEnte char(12) NOT NULL,
Utente varchar (20) NOT NULL,
Nome varchar (50) NOT NULL ,
Cognome varchar (50) NOT NULL,
Password varchar (20) NOT NULL,
Email varchar (255) NOT NULL,
CodStato smallint NOT NULL,
Descrizione varchar (255) NULL,
DataInserimento `datetime NOT NULL,`
DataModifica `datetime NOT NULL,`
IdUtenteApprovante `char (12) NOT NULL,`
NoteAmministratore text NULL,
CONSTRAINT PK_Utenti PRIMARY KEY CLUSTERED
(
IdUtente
)
)

	CodStato
01	Abilitato
02	Disabilitato
03	….. (usi futuri)

IdGruppoPrimario serve per associare la sigla del gruppo negli ID degli oggetti inseriti da utenti che appartengono a più gruppi.

Nel caso di inserimento da parte di utenti che appartengono a più gruppi (es. Redazione) occorre esprimere il gruppo con cui si intende operare, questo al fine di consentire la modifica dell'oggetto a tutti gli utenti appartenenti al gruppo. Come politica di gestione possiamo definire utenti appartenenti a più gruppi solo quelli col ruolo di Redazione. Per il ruolo di amministratore del sito si adotta la stessa politica della redazione per quel che riguarda l'inserimento di nuove istanze. L'amministratore ha in più la possibilità di definire gruppi ed utenti (questi forse possono essere definiti anche dalla redazione).

Relazione Gruppi

CREATE TABLE ***Gruppi*** (
IdGruppo char (12) NOT NULL ,
NomeGruppo varchar (20) NOT NULL ,
SiglaGruppo char (2) NOT NULL,
Descrizione varchar (255) NULL ,
DataInserimento `datetime NOT NULL ,`
DataModifica `datetime NOT NULL ,`
CodStato smallint NOT NULL,
IdUtenteApprovante `char (12) NOT NULL ,`
NoteAmministratore text NULL ,
CONSTRAINT PK_Gruppi PRIMARY KEY CLUSTERED
(
IdGruppo
)
)

La *SiglaGruppo* serve per eventuali realizzazioni distribuiti del sito per evitare la generazione di chiavi duplicati, il valore viene usato nella generazione degli identificatori dei vari oggetti del database. Ad ogni gruppo è associato almeno un utente col ruolo *Redazione*, a cui arrivano le notifiche delle richieste di aggiornamento. Verrà supportata la possibilità di fondere gruppi, facendo migrare utenti da un gruppo all'altro oppure utenti di più gruppi in un nuovo gruppo. Questa operazione può comportare l'attivazione di una procedura di aggiornamento su tutti gli oggetti del database, qualora un gruppo venga eliminato, per permettere la gestione degli oggetti associati al gruppo che si elimina.. La migrazione di un utente in un altro gruppo è possibile solo se l'utente non è connesso.

Relazione UtentiGruppi

CREATE TABLE ***UtentiGruppi*** (
IdUtente char (12) NOT NULL ,
IdGruppo char (12) NOT NULL ,
CodRuolo smallint NOT NULL,
DataInserimento `datetime NOT NULL ,`
DataModifica `datetime NOT NULL ,`
IdUtenteApprovante `char (12) NOT NULL ,`
NoteAmministratore text NULL ,
CONSTRAINT PK_UtentiGruppi PRIMARY KEY CLUSTERED
(
NomeGruppo,
Utente
)
)

Un utente assume un solo ruolo, non necessariamente il solito, per ogni gruppo di cui fa parte.

	CodRuolo
01	Ospite
02	Laboratorio
03	Redazione
04	Amministratore del sito
05	Dirigente
06	Amministratore di sistema
07	.... (usi futuri)

Relazione PrivilegiGruppo

CREATE TABLE. ***PrivilegiGruppo*** (
IdGruppo char (12) NOT NULL ,
CodOggetto smallint NOT NULL ,
DataInserimento `datetime NOT NULL ,`
DataModifica `datetime NOT NULL ,`
IdUtenteApprovante `char (12) NOT NULL ,`
NoteAmministratore text NULL ,

CONSTRAINT PK_Autorizzazioni PRIMARY KEY CLUSTERED
(
CodRuolo,
CodOggetto,
CodOperazione
)
)

CodOggetto corrisponde ad un oggetto del database. Per ogni ruolo è definito un template che specifica tutti gli oggetti e tutte le operazioni ammissibili su di esso. Nel caso si volessero imporre restrizioni o maggiori privilegi ai ruoli esistenti si definisce un nuovo ruolo con un nuovo template.

	CodOggetto
01	Agenda
02	Enti
03	Iniziative
05	Manifestazioni
06	Visite e soggiorni
07	Esperienze
08	Materiali
10	Progetti
11	Corsi di formazione
12	Corsi universitari
13	Corsi Master
14	Corsi istruzione secondaria
15	Corsi IFTS
16	Bando
17	Richiesta finanziamento
18	Valutazione richiesta
19	Progetto finanziato
20	Pagamento
21	Utenti e ruoli
100	Tutti

La registrazione di un utente è permessa a tutti i ruoli, la gestione di *Utenti e ruoli* è consentita solo al ruolo di redazione (o ad un ruolo più elevato).

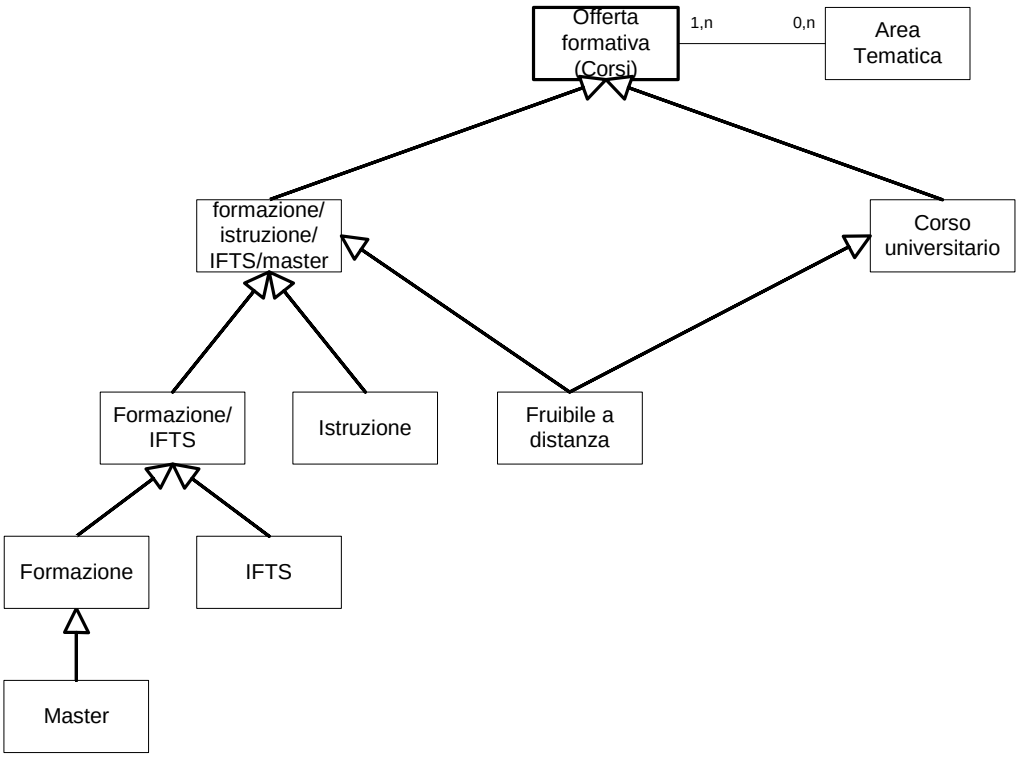
Offerta educativa contempla tutti gli oggetti che la compongono compreso Enti ed Agenda. *Offerta formativa* comprende tutti i corsi oltre ad Enti ed Agenda.

9.2.7 Trasformazione della classe Offerta Formativa

Il tentativo di trattare l'*Offerta Formativa* come tutte le altre offerte, si scontra con le differenti definizioni assunte dalle due grandi anime (mondo Andrea e mondo Anfora) per le classi *Argomenti,* e *Destinatari.* Tale convergenza non si è verificata in quanto la definizione di un approccio comune implicava un notevole impegno di studio, in cui venissero coinvolti esperti nella definizione di tesauri, non affrontabile con le risorse e gli impegni previsti per questo progetto.

La relazione OffertaEnti potrebbe essere clonata nella relazione CorsiEnti (qualora se ne presentasse la necessità per motivi di efficienza.

La risoluzione della gerarchia IS-A è avvenuta per partizionamento, dopo aver osservato che il traffico delle richieste che coinvolgono tutti i corsi a prescindere dalla loro natura coinvolge solo filtri su ContestoTerritoriale, Temi ed EntiCoinvolti e che le proprietà comuni a tutti i corsi si riducevano al solo Titolo..



La gerarchia in figura è stata risolta definendo le relazioni:

CorsoDiFormazione,
CorsiIstruzioneSecondaria,
CorsiMaster,
CorsiIFTS,
CorsiUniversitari
CorsiFad

Per problemi di efficienza, soprattutto per richieste che riguardano più tipologie di corsi è stata introdotta la relazione ridondante CORSI.

Relazione Corsi

CREATE TABLE **Corsi** (
IdCorso char (12) NOT NULL ,
CodTipologia smallint NOT NULL ,
CodTipologiaSpecifica smallint NOT NULL ,
Titolo varchar (255) NOT NULL ,
CodStato `smallint NOT NULL ,`
DataInserimento `datetime NOT NULL ,`
DataModifica `datetime NOT NULL ,`
IdGruppo `char (12) NOT NULL ,`
IdUtenteInserimento `char (12) NOT NULL ,`
IdUtenteModifica `char (12) NOT NULL ,`
IdUtenteApprovante `char (12) NOT NULL ,`
NoteAmministratore `text NULL ,`

CONSTRAINT PK_Corsi PRIMARY KEY CLUSTERED (
IdCorso
)
)

Relazione AreeTematiche

CREATE TABLE. AreeTematiche (
IdCorso char (12) NOT NULL ,
CodTema smallint NOT NULL ,
CONSTRAINT PK_Temi PRIMARY KEY CLUSTERED
(
IdCorso,
CodTema
)
)

Questa relazione rappresenta l'associazione multipla tra le classi *OffertaFormativa* e *AreaTematica*.

	CodTema
01	acqua
02	agricoltura
03	aria
04	beni culturali e ambientali
05	conservazione della natura
06	ecologia
07	informazione ed educazione ambientale
08	energia

09	gestione e pianificazione
10	igiene e sanità
11	impatto ambientale
12	normativa ambientale
13	rifiuti
14	rumore
15	sicurezza del lavoro e dell’ambiente
16	qualità e certificazione
17	tecniche e tecnologie ambientali
18	territorio
19	turismo ambientale
20	urbanistica
21	verde urbano

9.2.8 Trasformazione della classe Corso Universitario

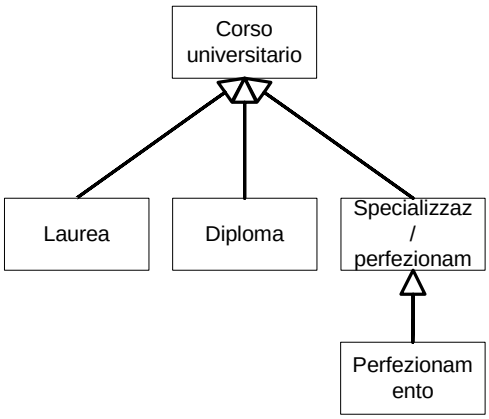


Figura 14

La gerarchia di figura 14 è stata risolta, per comodità di implementazione, definendo un'unica relazione (CorsiUniversitari), che contempla tutti i vari tipi di corsi universitari. Questa scelta deriva dalla constatazione che le classi specializzate di CorsoUniversitario contemplano poche proprietà aggiuntive, per cui sebbene la relazione risultante non è normalizzata, le semplificazioni implementative sono notevoli.

Relazione CorsiUniversitari

CREATE TABLE **CorsiUniversitari**(
IdCorso char (12) NOT NULL ,
Facoltà varchar (255) NULL ,
Dipartimento varchar (255) NULL ,
CodTipologia smallint NOT NULL ,
Titolo varchar (255) NOT NULL ,
IndirizzoDiStudioOrientamento varchar (255) NULL ,
PeriodoDiSvolgimento varchar (255) NULL ,
CodDurata smallint NULL ,
NumeroEsamiOModuli smallint NULL ,
AnnoAttivazione smallint NULL ,
NumeroChiuso char(1) NULL ,
NumeroPosti smallint NULL ,
TestIngresso char(1) NULL ,
QuotaIscrizione int NULL ,
RequisitiDiIngresso text NULL,
Materie text NULL,
CollaborazionePartenariato text NULL,
Referente varchar (255) NULL ,
DescrizioneContenuto text NULL,
AltreInformazioni text NULL,
CONSTRAINT PK_CorsiUniversitari PRIMARY KEY CLUSTERED (

IdCorso
)
)

Nota: la proprietà *IndirizzoDiStudioOrientamento* assume valore solo per Corso di Laurea (Indirizzo di studio) e per Diploma Universitario (Orientamento). Le proprietà *CondizioniDiAccesso, QuotaIscrizione* e *Materie* assumono valore solo per i Corsi di specializzazione o di perfezionamento. La proprietà *PeriodoDiSvolgimento* assume valore solo per *Corso di perfezionamento.*

I dati relativi alla *segreteria studenti* ed alla *presidenza/coordinamento del corso* appaiono nei riferimenti dell'Ente.

La proprietà ***CodTipologia*** assume i seguenti valori:

	CodTipologia
01	**Laurea**
02	**Diploma universitario**
03	**Scuola diretta a fini speciali**
04	**Scuola di specializzazione**
05	**Scuola di perfezionamento.**

La proprietà ***CodDurata*** assume i seguenti valori:

	CodDurata
01	**annuale**
02	**biennale**
03	**triennale**
04	**Quadriennale**
05	**Quinquennale**
06	**.....**

9.2.9 Trasformazione della classe Corso Istruzione

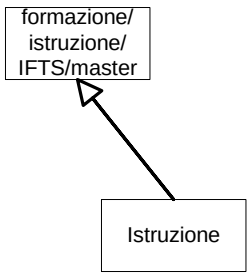


Figura 15

La gerarchia di figura 15 è stata risolta definendo un'unica relazione *CorsiIstruzioneSecondaria* che contiene le proprietà di entrambe le classi. Non è stato ritenuto utile definire due relazioni distinte, poiché è sembrata poco significativa la possibilità di esprimere condizioni di ricerca sulle proprietà comuni alle 4 tipologie di corsi.

Relazione CorsiIstruzioneSecondaria

CREATE TABLE **CorsiIstruzioneSecondaria** (
IdCorso char (12) NOT NULL ,
CodTipologia smallint NULL,
Titolo varchar (255) NOT NULL ,
DescrizioneContenuto text NULL ,
CodDurata smallint NULL ,
CodModalitaTemporali smallint NULL ,
CollaborazionePartenariato text NULL,
AltreInformazioni text NULL,
CONSTRAINT PK_CorsiIstruzione PRIMARY KEY CLUSTERED (
IdCorso
)
)

La proprietà ***CodTipologia*** assume i seguenti valori:

	CodTipologia
01	**Attestato di qualifica**
02	**Diploma**

La proprietà ***CodModalitaTemporali*** assume i seguenti valori:

	CodModalitaTemporali
01	**Tempo pieno**
02	**Tempo pieno residenziale**
03	**Tempo parziale**
04	**Fine settimana**
100	**Altra modalità (specificare)**

Per altra modalità viene memorizzata soltanto la specifica che assume un valore di codice da 4 in poi, in questo modo è possibile esprimere un filtro su tutti i corsi che assumono valore altra modalità (Cod > 4) che su una specifica altra modalità (es.Cod 18).

Relazione CorsiADistanza

CREATE TABLE **CorsiADistanza** (
	IdCorso char (12) NOT NULL ,
	OrganizzazioneInModuli char(1) NOT NULL,
	CorsiIndividualizzabili char(1) NOT NULL,
	NumeroSettimane smallint NULL,
	PercentualeOreADistanza smallint NULL,
	PercentualeOreInPresenza smallint NULL,
	FeedbackSuMateriali Char(1) NULL,
	CONSTRAINT PK_CorsiADistanza PRIMARY KEY CLUSTERED (
		IdCorso
	)
)

Questa relazione se popolata è sempre da considerarsi come estensione di una delle altre tipologie di corso.

Relazione ModuliIFAD

```
CREATE TABLE ModuliFAD(
        IdCorso char (12) NOT NULL ,
        NumProgressivo smallint NOT NULL,
        Titolo  varchar(255) NULL,
        CONSTRAINT PK_ModuliFAD PRIMARY KEY  CLUSTERED  (
                IdCorso , NumProgressivo
        ),
        CONSTRAINT FK_ModuliFAD FOREIGN KEY  (
                IdCorso
        ) REFERENCES dbo.CorsiADistanza (
                IdCorso

)
```

Relazione MaterialiDidatticiFAD

```
CREATE TABLE MaterialiDidatticiFAD (
        IdCorso char (12) NOT NULL ,
        CodMateriale smallint NOT NULL,
        CONSTRAINT PK_MaterialiFad PRIMARY KEY  CLUSTERED  (
                IdCorso , CodMateriale
        ),
        CONSTRAINT FK_MaterialiFad FOREIGN KEY  (
                IdCorso
        ) REFERENCES dbo.CorsiADistanza (
                IdCorso
)
```

	CodMateriale
01	Cartaceo
02	Audiovisivo (videocassette, lezioni TV)
03	Cd-Rom
04	Via web

Relazione DotazioniFAD

```
CREATE TABLE DotazioniFAD (
        IdCorso char (12) NOT NULL ,
        CodDotazione smallint NOT NULL,
        Specifica varchar (255) null,
        CONSTRAINT PK_DotazioniFad PRIMARY KEY  CLUSTERED  (
                IdCorso , CodDotazione
```

```
        ),
        CONSTRAINT FK_DotazioniFad FOREIGN KEY  (
                IdCorso
        ) REFERENCES dbo.CorsiADistanza (
                IdCorso
)
```

	CodDotazione
01	***Videoregistratore***
02	***Fax***
03	***Computer***
04	***Stampante***
05	***Collegamento Internet***
100	***altro***

Relazione DidatticaFAD

```
CREATE TABLE DidatticaFAD (
        IdCorso char (12) NOT NULL ,
        CodDidatticaFad smallint NOT NULL,
        CONSTRAINT PK_DidatticaFad PRIMARY KEY  CLUSTERED  (
                IdCorso , CodDidatticaFad
        ),
        CONSTRAINT FK_DidatticaFad FOREIGN KEY  (
                IdCorso
        ) REFERENCES dbo.CorsiADistanza (
                IdCorso
)
```

	CodDidatticaFad
01	***Test***
02	***Multimedia***
03	***Lavagna condivisa***
04	***Chat***
05	***Utilizzo banche dati***
06	***Forum di discussione***
07	***FAQ***
08	***Tour in rete***
100	***altro***

Relazione InterazioneCorsiADistanza

CREATE TABLE **InterazioneCorsiADistanza** (
IdCorso char (12) NOT NULL ,
CodModalita smallint NOT NULL,
CodSoggetto smallint NOT NULL,
CodStrumento smallint NOT NULL,
SpecificaInPresenza VarChar(255) NULL,
CONSTRAINT PK_InterazioneFad PRIMARY KEY CLUSTERED (
IdCorso , CodModalita, CodSoggetto, CodStrumento
),
CONSTRAINT FK_InterazioneFad FOREIGN KEY (
IdCorso
) REFERENCES dbo. CorsiADistanza (
IdCorso
)

La proprietà *SpecificaInPresenza* può assumere valore solo per *CodModalita* = "In presenza". *CodStrumento* indica gli strumenti di interazione con docenti, tutor e altri partecipanti e assume i valori presenti nella codifica solo per *CodModalita* = "A distanza", altrimenti assume valore –1. La relazione non è stata normalizzata per semplificazione dell'implementazione.

La proprietà ***CodModalita*** assume i seguenti valori:

	CodModalita
01	***A distanza***
02	***In presenza***

La proprietà ***CodModalita*** assume i seguenti valori:

	CodSoggetto
01	***Docenti***
02	***Tutor***
03	***Altri partecipanti***

La proprietà ***CodStrumento*** assume i seguenti valori:

	CodStrumento
01	***Via telefono***
02	***Via fax***
03	***Via e-mail***
04	***Forum***
05	***Via chat***
06	***Videoconferenza***
100	***altro***

9.2.10 Trasformazione della classe Corso Formazione/IFTS/Master

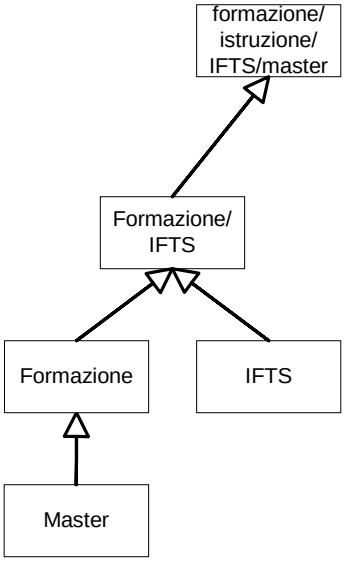


Figura 16

La gerarchia di Figura 16 viene risolta definendo una relazione Per i corsi di Formazione, una per i Master ed una per i corsi IFTS.

Relazione CorsiDiFormazione

CREATE TABLE **CorsiDiFormazione** (
IdCorso char (12) NOT NULL ,
Titolo varchar (255) NOT NULL ,
DescrizioneContenuto text null,
CodNaturaCorso smallint NULL,
QuotaIscrizione int NULL,
MaxAmmessi smallint NULL,
CostoOraAllievo int NULL,
CollaborazionePartnenariato text NULL,
CodTipoGestione smallint NULL,
CodTipologiaFormativa smallint NULL,
ModalitàDiAmmissione text NULL,
CodValutazioneCompetenze smallint NULL,
CodCertificazioneFinale smallint NULL,
CodModalitàDidatticaPrevalente smallint NULL,
CodModalitàTemporali smallint NULL,
CodDurata smallint NULL ,
AnnualitàInCorso smallint NULL,
OreComplessive smallint NULL,
Attivato char(1) NOT NULL,

```
        AltreInformazioni text NULL,
        CONSTRAINT PK_CorsiDiFormazione PRIMARY KEY  CLUSTERED  (
                IdCorso
        )
)
```

La proprietà ***CodModalitaTemporali*** assume i seguenti valori per tutti i Corsi (escluso Universitari):

	CodModalitaTemporali
01	**Tempo pieno**
02	**Tempo pieno residenziale**
03	**Tempo parziale**
04	**Fine settimana**
100	**Altra modalità (specificare)**

	CodNaturaCorso
01	**Pubblico**
02	**Privato**

	CodTipologiaFormativa
01	**Qualificazione**
02	**Specializzazione**
03	**Aggiornamento**
04	**Perfezionamento**
05	**Patente di mestiere**

	CodTipoGestione
01	**Diretta/delegata**
02	**Convenzionata pubblica**
03	**Convenzionata privata**
04	**Gestione di mercato**
100	**Altro**

	CodValutazioneCompetenze
01	**Si, in itinere**
02	**Si, finale**
03	**no**

	CodCertificazione finale
01	**Frequenza**
02	**Qualifica**
03	**Specializzazione**
04	**Patente di mestiere/abilitazione professionale**
05	**Nessuno**

	CodModalitàDidatticaPrevalente
01	**In aula**
02	**In alternanza**
03	**A distanza**
100	**Altro**

Relazione SedeDiSvolgimento

CREATE TABLE **SedeDiSvolgimento** (
IdCorso char (12) NOT NULL ,
NumProgressivo smallint NOT NULL,
IDLocazione Char(8) NULL,
Cap integer NOT NULL,
Citta varchar (255) NOT NULL ,
SiglaProvincia char(2) NOT NULL ,
Indirizzo varchar (255) NOT NULL ,
Collegamenti text NULL ,
Strutture text NULL,
BarriereArchitettoniche Char (1) NOT NULL ,
AvvioDaDefinire char(1)NOT NULL,
Dal datetime NULL,
Al datetime NULL,
CONSTRAINT PK_SedeDiSvolgimento PRIMARY KEY CLUSTERED (
IdCorso , NumProgressivo
)

)

una opportuna codifica delle proprietà Dal e Al segnala “avvio da definire”

Relazione CondizioniDiAccesso

CREATE TABLE **CondizioniDiAccesso** (

```
        IdCorso char (12) NOT NULL ,
        CodRequisito smallint NOT NULL,
        SpecificaRequisito varchar(255) NULL,
        CONSTRAINT PK_CondizioniDiAccesso PRIMARY KEY  CLUSTERED  (
                IdCorso, CodRequisito
        )
)
```

La proprietà ***CodRequisito*** assume i seguenti valori:

	CodRequisito
01	**Scuola dell'obbligo**
02	**Licenza media**
03	**Frequenza scuola secondaria**
04	**Attestato di qualifica**
05	**Diploma scuola secondaria**
06	**Diploma universitario**
07	**Laurea**
08	**Conoscenza lingua straniera**
100	**Altro**

La proprietà SpecificaRequisito assume un valore non a priori codificabile, dipendente dal valore assunto da CodRequisito (es. Laurea in Fisica)

Relazione DestinatariFormazione

```
CREATE TABLE DestinatariFormazione (
        IdCorso char (12) NOT NULL ,
        CodDestinatari smallint NOT NULL ,
        CodSpecifica smallint NOT NULL ,
        CONSTRAINT PK_DestinatariOFA PRIMARY KEY  CLUSTERED
        (
                IdCorso,
                CodDestinatari,
                CodSpecifica
        )
)
```

Contiene i destinatari delle istanze della classe corsi di Formazione. Tutte le specifiche relative alla codifica *Altro* vengono fornite nella proprietà del corso "Altre informazioni rilevanti". Per i corsi Master non sono ammesse le codifiche 02 e 100.

	CodDestinatari
01	**Giovani in cerca di I occupazione**
02	**Soggetti in situazione di svantaggio**
03	**Donne**
04	**Disoccupati**
05	**Occupati**
100	**Altro**

Per le offerte formative può avere un'ulteriore specifica negli attributi ***CodDestinatari*** e ***CodSottocategoriaDestinatari.***

Se ***CodDestinatari* = *Giovani in cerca di I occupazione***

	CodSpecifica
01	**Drop-out**
02	**Allievi**
03	**Giovani con qualifica**
04	**Giovani con diploma**
05	**Universitari e neo laureati**

Se ***CodDestinatari* = Soggetti in situazione di svantaggio**

	CodSpecifica
01	**Portatori di handicap**
02	**Tossicodipendenti**
03	**Detenuti ed ex detenuti**
04	**Immigrati**
05	**Nomadi**
06	**Disoccupati adulti in età avanzata**

Se ***CodDestinatari* = *Disoccupati***

	CodSpecifica
01	**Soggetti in riconversione**
02	**Di lunga durata**
03	**Mobilità**
04	**CIG**
100	**Altro**

Se ***CodDestinatari = Occupati***

	CodSpecifica
01	Apprendisti
02	Contratto formazione lavoro
03	Piccola e media impresa
04	Grande impresa
05	Istruzione e formazione professionale
06	Pubblica amministrazione
100	Altro

Relazione SpecificaDidattica

CREATE TABLE **SpecificaDidattica**(
IdCorso char (12) NOT NULL ,
CodModalitaDidattica smallint NOT NULL,
NumeroOre smallint NULL,
SpecificaDiAltro varchar (256) NULL,
CONSTRAINT PK_Didattica PRIMARY KEY CLUSTERED (
IdCorso,
CodModalitaDidattica
)
)

La proprietà ***CodModalitaDidattica*** assume i seguenti valori:

	CodModalitaDidattica
01	Lezioni in aula
02	Esercitazioni
03	Lezioni/esercitazioni a distanza
04	Lavoro in sottogruppo
05	Ricerca di gruppo
06	Discussione dei casi
07	Discussione guidata
08	Simulazione
09	Laboratori
10	Seminari
11	Visite guidate/viaggi di studio
12	Project work (progetti sul campo)
13	Stage
100	altro

Per il trattamento della modalità ***altro*** si adotta la stessa tecnica prevista per ***CodModalitaTemporali***

Relazione Docenza

CREATE TABLE **Docenza**(
IdCorso char (12) NOT NULL ,
CodTipoDocenza smallint NULL,
SpecificaDiAltro varchar (256) NULL,
CONSTRAINT PK_Docenza PRIMARY KEY CLUSTERED (
IdCorso, CodTipoDocenza
)
)

La proprietà ***CodTipoDocenza*** assume i seguenti valori:

	CodTipoDocenza
01	universitaria
02	aziendale
03	consulenza
04	testimonianza
05	tutoring
06	animatori
07	funzionari di P.A.
08	docenti interni
09	docenti iscritti all'albo regionale
100	altro (specificare).

Per il trattamento della modalità ***altro*** si adotta la stessa tecnica prevista per ***CodModalitaTemporali***

Relazione FontiFinanziarie

CREATE TABLE **FontiFinanziarie**(
IdCorso char (12) NOT NULL ,
CodOrigine smallint NOT NULL,
CodSpecifica smallint NOT NULL,
SpecificaDiAltro varchar (256) NULL,
CONSTRAINT PK_Fonti PRIMARY KEY CLUSTERED (
IdCorso,CodOrigine,CodSpecifica
)
)

La proprietà ***CodOrigine*** assume i seguenti valori:

	CodOrigine
01	Fondi Comunitari
02	Fondi pubblici
03	Fondi privati
100	Altro

La proprietà ***CodSpecifica*** assume i seguenti valori:

	CodSpecifica (origine = Fondi Comunitari)
01	***FSE, obiettivi, assi, etc***
02	***Programmi e iniziative comunitarie***
03	***Fondi comunitari di altra natura***

	CodSpecifica Origine = Fondi pubblici
01	***Nazionali***
02	***Regionali***
03	***Provinciali***
04	***Comunali***
05	***Comunità Montana***

	CodSpecifica Origine = Fondi privati)
01	***Associazioni imprenditoriali***
02	***Associazioni di categoria***
03	***Autofinanziamento***
100	***Altro***

Relazione CorsiMaster

CREATE TABLE **CorsiMaster** (
IdCorso char (12) NOT NULL ,
Titolo varchar (255) NOT NULL ,
Finalità text null,
OrganizzazioneInModuli char(1) NOT NULL,
PartecipazioneSingoliModuli char(1) NULL,
CodNaturaDelCorso smallint NULL,
CollaborazionePartnenariato text NULL,
MaxAmmessi smallint NULL,
QuotaIscrizione int NULL,
BorseDiStudio char(1) NULL,
ModalitaFruizioneBorsa text NULL,
Edizione smallint NULL,
CostoOraAllievo int NULL,
AltreCondizioniAccesso text NULL,
ModalitaAmmissione text NULL,
CodValutazioneCompetenze smallint NULL,
CodCertificazioneFinale smallint NULL,
CodModalitàDidatticaPrevalente smallint NULL,

CodModalitàTemporali smallint NULL,
CodDurata smallint NULL,
AnnualitàInCorso smallint NULL,
OreComplessive smallint NULL,
AltreInformazioni text NULL,
CONSTRAINT PK_CorsiMaster PRIMARY KEY CLUSTERED (
IdCorso
)
)

Nella relazione *CondizioniDiAccesso*, per questi corsi la proprietà ***CodRequisito*** assume i seguenti valori:

	CodRequisito
05	Diploma scuola secondaria
06	Diploma universitario
07	Laurea
100	Altro

	CodCertificazione finale
01	Frequenza
02	Qualifica
03	Specializzazione
04	Diploma
05	Nessuna
100	Altro

Relazione ModuliMaster

CREATE TABLE **ModuliMaster** (
IdCorso char (12) NOT NULL ,
NumProgressivo smallint NOT NULL,
Titolo varchar(255) NULL,
Ore smallint NULL,
Dal datemine null,
Al datetime null,
CONSTRAINT PK_ModuliMaster PRIMARY KEY CLUSTERED (
IdCorso , NumProgressivo
),
CONSTRAINT FK_ModuliMaster FOREIGN KEY (

```
        IdCorso
    ) REFERENCES dbo.CorsiMaster (
        IdCorso

)
```

Relazione CorsiIFTS

```
CREATE TABLE CorsiIFTS (
    IdCorso char (12) NOT NULL ,
    Titolo varchar (255) NOT NULL ,
    FiguraProfessionale  text NULL,
    SbocchiOccupazionali  text NULL,
    QuotaIscrizione int NULL,
    MaxAmmessi smallint NULL,
    CostoOraAllievo int NULL,
    OreComplessive smallint NULL,
    AgevolazioniFinanziarie  text  NULL,
    DestinatariCondizioniAccesso text NULL,
    ModalitàDiAmmissione text NULL,
    CodValutazioneCreditiIniziale  smallint  NULL,
    CodValutazioneCreditiFinale  smallint  NULL,
    PianiIndividualizzabili  char(1) NULL,
    Attivato  char(1) NOT NULL,
    CertificazioneIntermedia  varchar(255) NULL,
    CertificazioneFinale  varchar(255) NULL,
    NumeroSemestri  smallint NULL,
    NumeroModuli  smallint NULL,
    CodModalitàTemporali smallint NULL,
    AltreInformazioni text NULL,
    CONSTRAINT PK_CorsiIFTS PRIMARY KEY  CLUSTERED  (
        IdCorso
    )
)
```

OreComplessive non sempre è ricavabile dalla somma delle ore per semestre, in quanto non sempre viene fornito tale dettaglio, per cui occorre fornirlo.

NumeroModuli e *NumeroSemestri* sono uguali al numero dei moduli/semestri presenti nelle relazione *SemestriIfts* e *ModuliIFTS*, non possono essere campi calcolati, perché spesso viene solo fornito il numero di essi senza i dettagli.

	CodValutazioneCreditiIniziale
01	**Iniziale, per l'accesso al corso**

Questa codifica si poteva evitare in quanto descrive solo la presenza della valutazione, per cui è sufficiente dichiararlo char (1). Viene lasciato come codifica per uniformità di trattamento e per consentire ulteriori eventuali codifiche.

	CodValutazioneCreditiFinale
01	**Finale con riconoscimento certificato**
02	**Finale senza riconoscimento certificato**

La relazione non è normalizzata per semplificazione di implementazione e perché le codifiche di valutazione sembrano stabili.

Relazione ServiziSupporto

CREATE TABLE **ServiziSupporto** (
    IdCorso char (12) NOT NULL ,
    CodServiziDiSupporto smallint NULL,
    CONSTRAINT PK_ServiziSupportoIFTS PRIMARY KEY CLUSTERED (
        IdCorso , CodServiziDiSupporto
)
)

Questa relazione contiene i servizi di supporto per IFTS.

	CodServiziDiSupporto
01	**Informativo/orientativo**
02	**Psico-motivazionale**
03	**Professionale (tecnologico)**

Relazione SemestriFTS

CREATE TABLE **SemestriIFTS** (
    IdCorso char (12) NOT NULL ,
    Semestre smallint NOT NULL,
    Titolo varchar(255) NULL,
    Ore smallint NULL,
    CONSTRAINT PK_SemestriIFTS PRIMARY KEY CLUSTERED (
        IdCorso , Semestre
    ),
    CONSTRAINT FK_SemestriIFTS FOREIGN KEY (
        IdCorso
) REFERENCES dbo.CorsiIFTS (
        IdCorso

)

Ore non sempre è ricavabile dalla somma delle ore dei moduli per semestre, in quanto non sempre viene fornito tale dettaglio, per cui occorre fornirlo

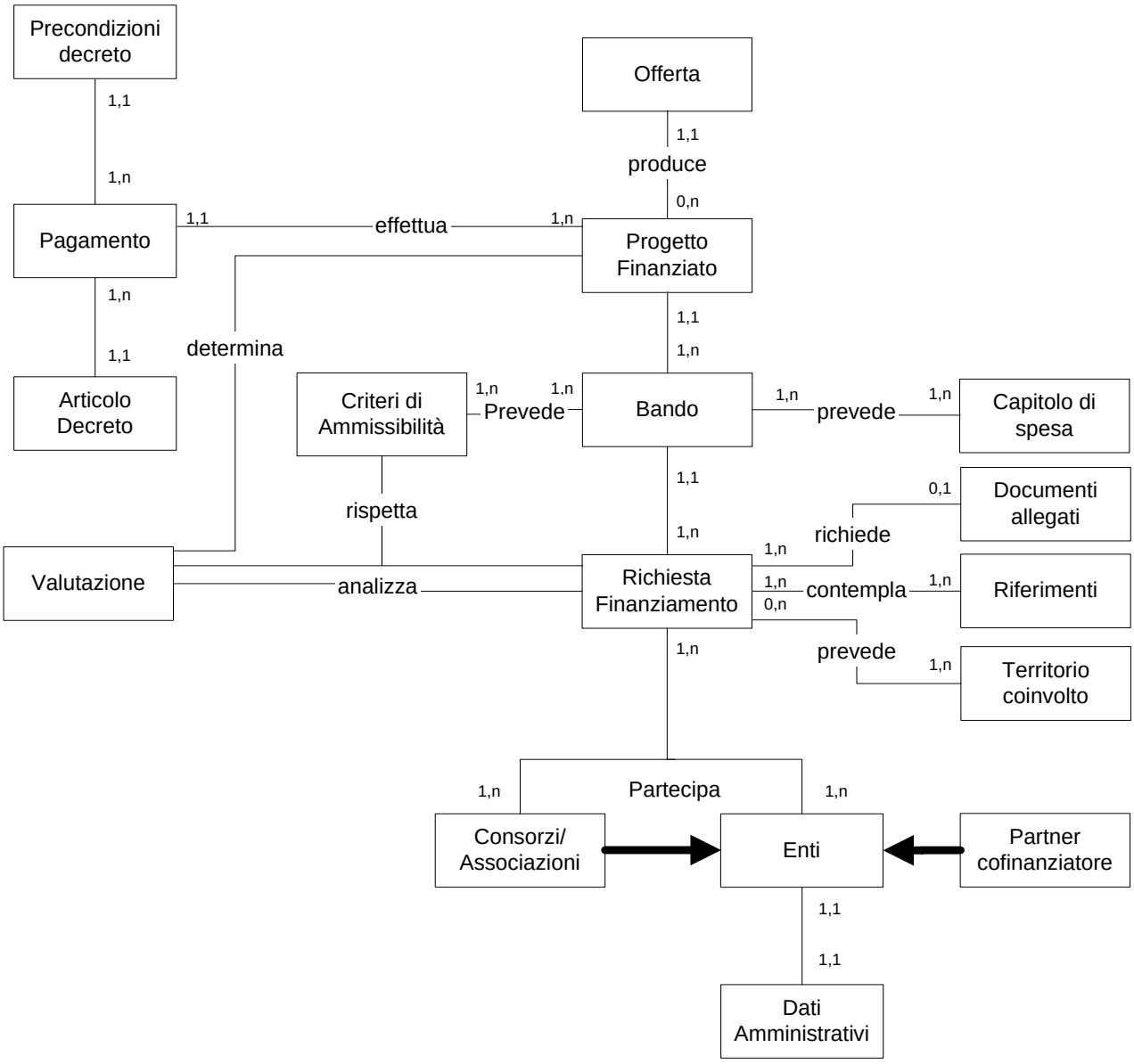
---

Relazione ModuliIFTS

CREATE TABLE **ModuliIFTS** (
IdCorso char (12) NOT NULL ,
Semestre smallint NOT NULL,
NumeroModulo smallint NOT NULL,
Titolo varchar(255) NULL,
Ore smallint NULL,
Dal datetime NULL,
Al datetime NULL,
CONSTRAINT PK_ModuliIFTS PRIMARY KEY CLUSTERED (
IdCorso , Semestre, NumeroModulo
),
CONSTRAINT FK_ModuliIFTS FOREIGN KEY (
IdCorso , Semestre
) REFERENCES dbo.SemestriIFTS (
IdCorso , Semestre

)

9.2.11 Trasformazione delle classi relative alle richieste finanziarie.



Le classi Offerta, Riferimenti ed Enti sono quelle dello schema globale.

Relazione Bandi

CREATE TABLE **Bandi** (
IdBando char (12) NOT NULL,
CodStato smallint NOT NULL,
CodNormaRiferimento smallint NOT NULL,
ImportoComplessivoDelBando float,
NumeroDecreto smallint,
CodValuta smallint,
QuotaMaxFinanziabile float,
PercentualeMaxFinanziamento float ,
DataDecreto datetime,
DataPubblicazioneDecreto datetime,
DataInserimento datetime,
DataModifica datetime,
IdGruppo char(12) NOT NULL,
IdUtenteInserimento char(12) NOT NULL,
IdUtenteModifica char(12) NOT NULL,
IdUtenteApprovante char(12) NOT NULL,
NoteAmministratore text NULL,
CONSTRAINT PK_Bandi PRIMARY KEY CLUSTERED (
IdBando
)
}

CodStato assume solo i valori di servizio, in quanto i bandi vengono inseriti solo all'atto della pubblicazione nella gazzetta ufficiale che corrisponde alla sua attivazione

QuotaMax e *PercentualMaxFinanziamento* sono al momento utili solo per Agende 21

NumeroDecreto è assegnato dal protocollo.

	CodNormaRiferimento
01	Agende 21
02	Poma
03	PTTA
04	Educazione ambientale
05	Formazione
06	Città dei bambini e delle bambine
08	…….

	CodValuta
01	Euro
02	Lira
03	……..

Relazione CapitoliDiSpesa

CREATE TABLE **CapitoliDiSpesa** (
IdBando char (12) NOT NULL,
CapitoloDiSpesa smallint NOT NULL,
Anno smallint NOT NULL,
Denominazione varchar (255),
ImportoDisponibile float,
CodStato smallint,
DataInserimento datetime,
DataModifica datetime,
IdUtenteInserimento char(12) NOT NULL,
IdUtenteModifica char(12) NOT NULL,
IdUtenteApprovante char(12) NOT NULL,
NoteAmministratore text NULL,
CONSTRAINT PK_ CapitoliDiSpesa PRIMARY KEY CLUSTERED (
IdBando,
CapitoloDiSpesa,
Anno
)
}

Lo stesso codice relativo al *CapitoloDiSpesa* può comparire in più anni

---

Relazione CriteriPrevistiDalBando

CREATE TABLE **CriteriPrevistiDalBando** (
IdBando char (12) NOT NULL,
CodCriterio smallint NOT NULL,
TestoCriterio text,
CONSTRAINT PK_ CapitoliDiSpesa PRIMARY KEY CLUSTERED (
IdBando,
CodCriterio,
)
}

CodCriterio è dipendente dall'Id del bando… per esempio per Agende 21 è definito dalla tipologia del soggetto proponente (nella tabella seguente):

	CodCriterio
01	**Comune con popolazione superiore a 10000 abitanti**
02	Consorzio, associazione o unione tra comuni con popolazione totale superiore a 10000 abitanti
03	Comune con popolazione superiore a 8000 abitanti situato nelle regioni obiettivo 1
04	Consorzio, associazione o unione tra comuni con popolazione totale superiore a 8000 abitanti situato nelle regioni ad obiettivo 1
05	Comune con popolazione superiore a 5000 abitanti situato nella regione Molise
06	Consorzio, associazione o unione tra comuni con popolazione totale superiore a 5000 abitanti situato nella regione Molise
07	Comune con popolazione superiore a 3000 abitanti situato nella regione Valle d'Aosta
08	Consorzio, associazione o unione tra comuni con popolazione totale superiore a 3000 abitanti situato nella regione Valle d'Aosta
09	Provincia
10	Comunità montana
11	Ente di gestione di area naturale protetta

Relazione CriteriRispettati

CREATE TABLE **CriteriRispettati** (
    IdBando char (12) NOT NULL,
    IdRichiesta char (12) NOT NULL,
    CodCriterio smallint NOT NULL,
    CONSTRAINT PK_CriteriRispettati PRIMARY KEY CLUSTERED (
        IdBando,
        IdRichiesta
)
}

Relazione RichiesteFinanziamento

CREATE TABLE **RichiesteFinanziamento** (
    IdBando char (12) NOT NULL,
    IdRichiesta char (12) NOT NULL,
    TitoloProgetto varchar (256),

Acronimo char (20),
ImportoRichiesto float,
CodTipologiaIntervento smallint,
CodTerritorioCoinvolto smallint,
ProtocolloArrivo varchar(255),
DurataProgetto smallint,
Descizione text,
CodStato smallint not null,
DataInserimento datetime not null,
DataModifica datetime not null,
IdGruppo char(12) not null,
IdUtenteInserimento char(12) not null,
IdUtenteModifica char(12) not null,
IdUtenteApprovante char(12) not null,
NoteAmministratore text null,
CONSTRAINT PK_CriteriRispettati PRIMARY KEY CLUSTERED (
IdBando,
IdRichiesta
)
}

CodTipologiaIntervento è per il momento definito solo per Agende 21

	CodTipologiaIntervento
01	A - azioni di coinvolgimento degli attori locali
02	A - costituzione del Forum permanente di Agende 21
03	A - redazione del RSA del territorio interessato
10	B - definizione del piano d'azione locale
11	B - studi di fattibilità delle azioni già individuate
12	B - studi e linee guida per la messa a punto di prescrizioni…

Relazione EntiConsorziati

CREATE TABLE **EntiConsorziati** (
IdEnte char (12) NOT NULL,
IdRichiesta char (12) NOT NULL,
CodTipoAssociazione smallint NOT NULL,
CONSTRAINT PK_EntiConsorziati PRIMARY KEY CLUSTERED (
IdEnte,
IdRichiesta,
CodTipoAssociazione
)
}

	CodTipoAssociazione
01	**Proponente**
02	CoProponente
03	Beneficiario

Relazione PartnersCofinanziatori

CREATE TABLE **PartnersCofinanziatori** (
	IdEnte char (12) NOT NULL,
	IdRichiesta char (12) NOT NULL,
	QuotaFinanziata int NOT NULL,
	CONSTRAINT PK_ PartnersCofinanziatori PRIMARY KEY CLUSTERED (
		IdEnte,
		IdRichiesta,
	)
}

Relazione DatiAmministrativi

CREATE TABLE **DatiAmministrativi** (
	IdEnte char(12) not null,
	PartitaIva char(20) not null,
	CodiceFiscale char(20) not null,
	CodTitologia smallint,
	CodiceTesoreria smallint,
	NumeroConto varchar (30),
	ABI int null,
	CAB int null,
	DataInserimento date
	DataModifica date
	IdGruppo char(12) not null
	IdUtenteInserimento char(12) not null
	IdUtenteModifica char(12) not null
	IdUtenteApprovante char(12) not null
	NoteAmministratore text null,
	CONSTRAINT PK_DatiAmministrativi PRIMARY KEY CLUSTERED (
		IdEnte,
	)
}

ABI e CAB solo per Banche, CodiceTesoreria solo per tesoreria

	CodTipologia
01	**Banca**
02	Tesoreria
03	CC.Postale

Relazione Valutazione

CREATE TABLE **Valutazione** (
IdRichiesta char (12) NOT NULL,
CodValutazione smallint,
PunteggioTotale varchar (255),
Motivazioni text,
DataInserimento datetime,
DataModifica datetime,
CodStato smallint,
IdGruppo char(12) not null
IdUtenteInserimento char(12) not null
IdUtenteModifica char(12) not null
IdUtenteApprovante char(12) not null
NoteAmministratore text null,
CONSTRAINT PK_ Valutazione PRIMARY KEY CLUSTERED (
IdRichiesta,
)
}

	CodValutazione
01	Idoneo
02	Cofinanziato

Relazione ProgettoFinanziato

CREATE TABLE **ProgettoFinanziato** (
IdProgetto char (12) NOT NULL,
CodNormaRiferimento smallint,
CodTipologiaAccordo smallint
Partecipanti text,
DataAttivazione datatime,
Protocollo smallint not null,
DataInizioIntervento datetime,

DataConclusionePrevista datetime,
AmmontareTotale int NOT NULL,
DataInserimento datetime not null,
DataModifica datetime not null,
CodStato smallint NOT NULL,
IdGruppo char(12) not null
IdUtenteInserimento char(12) not null
IdUtenteModifica char(12) not null
IdUtenteApprovante char(12) not null
NoteAmministratore text null,
CONSTRAINT PK_ProgettoFinanziato PRIMARY KEY CLUSTERED (
IdProgetto
)
}

	CodTipologiaAccordo
01	Decreto
02	Contratto
03	Convenzione
04	Comunicazione

	CodNormaRiferimento
01	Agende 21
02	Poma
03	PTTA
04	Educazione ambientale
05	Formazione
06	Città dei bambini e delle bambine
08	…….

CodNormaRiferimento era stata chiamata CodAzione nella precedente versione.

Relazione Pagamenti

CREATE TABLE **Pagamenti** (

IdProgetto char (12) NOT NULL,
CodStatoRegistrazione smallint NOT NULL,
DataPrevista datetime,
DataPagamento datetime,
Protocollo smallint NOT NULL,
Importo int,

DataRegistrazione datetime,
NumeroRegistrazione int,
CodStato smallint NOT NULL,
DataInserimento datetime NOT NULL,
DataModifica datetime NOT NULL,
IdGruppo char(12) not null
IdUtenteInserimento char(12) not null
IdUtenteModifica char(12) not null
IdUtenteApprovante char(12) not null
NoteAmministratore text null,
CONSTRAINT PK_Pagamenti PRIMARY KEY CLUSTERED (
IdProgetto
)
}

	CodStatoRegistrazione
01	In corso
02	effettuata

Relazione ArticoliDecreti

CREATE TABLE **ArticoliDecreti** (
IdProgetto char (12) NOT NULL,
Articolo varchar (255),
Testo text,
CONSTRAINT PK_ArticoliDecreti PRIMARY KEY CLUSTERED (
IdProgetto
)
}

Relazione PrecondizioniDecreto

CREATE TABLE *PrecondizioniDecreto* (
IdProgetto char (12) NOT NULL,
Precondizioni varchar (255),
Testo text,
CONSTRAINT PK_ *PrecondizioniDecreto* PRIMARY KEY CLUSTERED (
IdProgetto
)
}

Relazione DocumentiRichiesti

CREATE TABLE **DocumentiRichiesti** (
IdBando char (12) NOT NULL,
Documento smallint
TestoDocumento varchar (255),
CONSTRAINT PK_DocumentiRichiesti PRIMARY KEY CLUSTERED (
IdBando
)
}

Documento dipende dall'*IdBando*, per esempio per Agende 21 è:

	Documento
01	Copia della dichiarazione dell'impegno di spesa dell'organo competente per statuto (duplice copia cartacea).
02	Copia del progetto (Allegato 2, una copia cartacea ed una in formato elettronico)
03	Modulo finanziario con la ripartizione degli oneri (Allegato 3, una copia cartacea ed una in formato elettronico)
04	Copia della delibera di adesione alla Carta di Aalborg (duplice copia cartacea)
05	Copia della delibera di costituzione in consorzio tra comuni (duplice copia cartacea)
06	Per i comuni riuniti in consorzio, copia dello statuto (duplice copia cartacea)
07	Rapporto sullo stato dell'ambiente (solo per i soggetti che concorrono per la categoria "B")
08	Copia dei verbali e/o del regolamento del forum permanente (solo per i soggetti che concorrono per la categoria "B")

Relazione DocumentiAllegati

CREATE TABLE **DocumentiAllegati** (
IdRichiesta char (12) NOT NULL,
IdBando char (12) NOT NULL,
Documento smallint
TestoDocumento varchar (255),
CONSTRAINT PK_DocumentiAllegati PRIMARY KEY CLUSTERED (
IdRichiesta,
IdBando
)
}

Active Notifier

Il meccanismo di notifica delle operazioni sul database, viene realizzato con l'ausilio delle seguenti due relazioni.

Relazione Notifier

CREATE TABLE **Notifier** (
IdOggetto char (12) NOT NULL,
CodTipoOperazione smallint NOT NULL,
CodTipologiaOggetto smallint
IdMittente char (12) NOT NULL,
IdGruppo char (12) NOT NULL,
Data datetime NOT NULL,
CONSTRAINT PK_Notifier PRIMARY KEY CLUSTERED (
IdOggetto,
IdMittente,
Data
)
}

Questa relazione fornisce l'input per l'*active notifier* dell'utente con ruolo *Redazione*, è popolata dalle operazioni effettuate dagli utenti con ruolo *Laboratorio*.
IdGruppo permette di individuare tutti gli utenti con ruolo redazione (dalla relazione *UtentiGruppi*) che possono elaborare il messaggio.

	CodTipoOperazione
02	Proponi Inserimento
03	Proponi modifica
04	Proponi cancellazione

	CodTipologiaOggetto
01	Agenda
02	Enti
03	Iniziative
05	Manifestazioni
06	Visite e soggiorni
07	Esperienze
08	Materiali
10	Progetti
11	Corsi di formazione
12	Corsi universitari
13	Corsi Master
14	Corsi istruzione secondaria

15	Corsi IFTS
16	Bando
17	Richiesta finanziamento
18	Valutazione richiesta
19	Progetto finanziato
20	Pagamento
21	Utenti e ruoli

CREATE TABLE **EsitoNotifica** (
IdOggetto char (12) NOT NULL,
CodTipoOperazione smallint NOT NULL,
CodTipologiaOggetto smallint
IdMittente char (12) NOT NULL,
IdDestinatario char (12) NOT NULL,
DataProposta datetime NOT NULL,
DataElaborazione datetime NOT NULL,
CodEsito smallint NOT NULL,
Commento text,
CONSTRAINT PK_EsitoNotifica PRIMARY KEY CLUSTERED (
IdOggetto,
IdMittente,
DataElaborazione
)
}

	CodTipoEsito
01	Accettato
02	Verificare
03	Rifiutato

10 Notifiche in SVS: valutazione delle prestazioni del software di gestione

10.1 Introduzione

Come descritto in precedenza, il modulo di gestione delle notifiche è stato implementato usando il paradigma Message Oriented Middleware (MOM).

Il meccanismo delle notifiche introduce infatti comunicazioni asincrone fra gli utenti e l'uso di code di messaggi è uno dei paradigmi di implementazione chiave per gestire le interazioni asincrone. Pur essendo l'architettura basata su code di messaggi ampiamente utilizzata nei sistemi informativi tradizionali, mancano esperienze significative in ambito WIS. Coerentemente con le specifiche del progetto che stabilivano in Java il linguaggio di sviluppo, abbiamo usato il framework Java Message Service (JMS) per l'implementazione del sistema di gestione delle notifiche.

10.2 SOAP vs protocollo ad hoc

Il sottosistema di gestione della comunicazione del WIS Infea,da cui ha preso le mosse SVS, prevedeva l'uso di 3 protocolli di comunicazione e relativi server:

- HTTP ed HTTPS gestiti da servlet container
- RMI gestito da un server remote
- WTP gestito da un gateway

Nell'implementare il gestore delle notifiche ci si è trovati davanti alla necessità di scegliere un protocollo ed un server per l'interazione asincrona. Una rassegna comparativa delle principali implementazioni JMS esistenti ci ha portato ad individuare il prodotto open source Joram (http://joram.objectweb.org/). La comunicazione in Joram può avvenire secondo due protocolli: uno nativo che richiede l'uso di un server ad hoc e SOAP che invece si basa su http ed un servlet container. Mentre nel secondo caso non avremmo aggiunto server al sistema, nel primo si. Prima di decidere abbiamo provato a fare delle misure per cercare di capire le differenze in termine di prestazioni ed occupazione di banda tra le due implementazioni. In questa parte del documento descriviamo i principali risultati ottenuti.
Utilizzare il protocollo di comunicazione Soap per lo scambio delle informazioni in ambienti distribuiti comporta diversi vantaggi, la possibilità di scambiare messaggi XML utilizzando la connessione http e quindi la porta 80 è uno di questi. Transitare dalla comunissima porta 80 significa evitare di vedersi negato l'accesso da qualche firewall ad esempio, e la possibilità di poter comunicare con dispositivi wireless. Spesso però il prezzo di tutto ciò si paga in una diminuzione delle prestazioni delle trasmissioni.

10.3 Misurazioni

La tecnologia Soap è stata integrata nelle ultime versioni di Joram (dalla 3.4). Questo ha reso Joram una piattaforma di messaging utilizzabile anche da dispositivi che non supportano Java o i protocolli TCP. Il sistema di messaging ha quindi beneficiato dei vantaggi forniti da Soap, abbiamo cercato di capire però quali sono le prestazioni.
Lo scopo di questo studio consiste nel valutare le differenze di prestazioni tra l'uso del protocollo nativo e Soap.

L’analisi consta di un primo calcolo del carico di pacchetti jar che devono essere trasferiti sul client; seguono due tipologie di test improntate a misurare i tempi di trasferimento della richiesta di connessione al topic, del post di un messaggio dall’ambiente del laboratorio a quello della redazione e viceversa, del post di un messaggio dall’ambiente della redazione a quello del laboratorio.

10.4 Misura dei jar

Di seguito elenchiamo l’insieme delle librerie che risiedono sul client nel caso in cui Joram non fornisca il protocollo Soap.

JCup.jar	85 kb
jms.jar	26 kb
joram.jar	305 kb
jta.jar	9 kb
log4j.jar	343 kb
jndi.jar	97 kb
ow_monolog.jar	127 kb
	952 kb

Mentre l’insieme delle librerie richieste dal client nel caso in cui la piattaforma Joram includa la tecnologia Soap è:

JCup.jar	85 kb
jms.jar	26 kb
joram.jar	305 kb
jta.jar	9 kb
log4j.jar	343 kb
activation.jar	54 kb
jndi.jar	97 kb
mail.jar	299 kb
soap.jar	230 kb
ow_monolog.jar	127 kb
	1575 kb

Utilizzare la versione di Joram con SOAP appesantisce il client di 583 kb.

10.5 Test I

Il primo test consiste nel sottoporre al sistema MOM un insieme di richieste di connessione. Ciò che desideriamo misurare è il tempo che intercorre dalla richiesta di connessione al *topicconnectionfactory*, alla generazione del *listener* usato dal client effettuare il subscribe sul topic.

Vengono attivati dieci thread che simulano la richiesta di connessione da parte di un client. Ogni thread richiede e rilascia la connessione per venti volte.
Il test viene ripetuto sei volte. Il bootstrap del sistema precede la prima esecuzione del test. Per ogni esecuzione del test viene calcolata la media dei risultati delle varie valutazioni.
Infine l’intero test è stato ripetuto tre volte.

I tempi sono espressi in millisecondi.

L'intera verifica viene eseguita sulle due piattaforme Joram, in presenza di Soap e in assenza di Soap. Le tabelle seguenti riportano i relativi risultati.

10.5.1 Test senza SOAP

Prima Prova	
Test 1	MEDIA: 10.51
Test 2	MEDIA: 10.16
Test 3	MEDIA: 10.09
Test 4	MEDIA: 10.24
Test 5	MEDIA: 10.18
Test 6	MEDIA: 11.23

Seconda Prova	
Test 1	MEDIA: 10.31
Test 2	MEDIA: 10.87
Test 3	MEDIA: 10.56
Test 4	MEDIA: 10.06
Test 5	MEDIA: 10.01
Test 6	MEDIA: 10.25

Terza Prova	
Test 1	MEDIA: 10.08
Test 2	MEDIA: 10.25
Test 3	MEDIA: 10.71
Test 4	MEDIA: 10.29
Test 5	MEDIA: 10.06
Test 6	MEDIA: 13.86

Possiamo osservare che complessivamente i valori si mantengono intorno ai 10.50 millisecondi.

10.5.2 Test con SOAP

Prima Prova	
Test 1	MEDIA: 8.335
Test 2	MEDIA: 7.88
Test 3	MEDIA: 7.85
Test 4	MEDIA: 7.69
Test 5	MEDIA: 8.40
Test 6	MEDIA: 8.332

Seconda Prova	
Test 1	MEDIA: 7.68
Test 2	MEDIA: 7.75
Test 3	MEDIA: 7.87
Test 4	MEDIA: 8.30
Test 5	MEDIA: 8.07
Test 6	MEDIA: 7.80

Terza Prova	
Test 1	MEDIA: 8.27
Test 2	MEDIA: 8.75
Test 3	MEDIA: 9.21
Test 4	MEDIA: 8.99
Test 5	n.d.
Test 6	n.d.

Le prime due serie di esecuzioni presentano valori simili, mentre la terza impiega tempi più lunghi, probabilmente a causa di un sovraccarico del sistema.
Possiamo osservare comunque che complessivamente i valori si mantengono intorno agli 8.00 millesecondi.

10.5.3 Considerazioni sul test I

Confrontando i dati sopra riportati con quelli ottenuti dai test eseguiti in assenza di Soap possiamo notare un notevole abbassamento dei tempi, di oltre 2 millesecondi.
D'altro canto le esecuzioni effettuate in presenza del protocollo Soap hanno rivelato alcuni problemi. Durante la terza ripetizione dell'intero test il sistema è andato in crash impedendo la misurazione delle ultime due esecuzioni. Anche in seguito al restart del sistema non è stato possibile proseguire oltre la quarta esecuzione del test. Causa di tale malfunzionamento potrebbe essere un accumulo sul server MOM di un insieme di oggetti (connessioni) tale da risultare non gestibile. Un fatto osservato: lo shut down del servlet container Tomcat in un qualche modo sblocca il sistema SVS, il server rmi prosegue la sua esecuzione (nel test successivo si osserva la ripresa dell'esecuzione delle richieste di inserimento).
Un altro problema presentato durante la prima fase di test su Soap e in seguito risolto, consisteva nella generazione a tempo di esecuzione di una serie di eccezioni: l'eccezione, riportata sotto, appariva in modo imprevedibile e varie volte nel corso della stessa esecuzione. Si è scoperto in seguito che veniva provocata dalla richiesta di quelle connessioni che, rimaste inattive da più di sessanta secondi (valore settato da programma) erano state chiuse dal server. E' stato possibile superare tale ostacolo configurando opportunamente il timer che controlla le connessioni inattive di Soap e che ne causa la chiusura.
Aggiungiamo che questa non è una soluzione, ma un modo per aggirare il problema; infatti essendo http, su cui SOAP si basa, un protocollo *connectionless* non è possibile determinare quando un client non è più connesso ed un timeout è necessario.

```
[ prova-9] [2003-10-23 14:43:13,994] ERROR fr.dyade.aaa.joram.Client  - javax.jm
s.IllegalStateException: The SOAP service failed to process the call: Exception
from service object: Connection 179 has been removed.
Exception caught in TestSoap: javax.jms.IllegalStateException: The SOAP service
failed to process the call: Exception from service object: Connection 179 has be
en removed.
javax.jms.IllegalStateException: The SOAP service failed to process the call: Ex
ception from service object: Connection 179 has been removed.
        at fr.dyade.aaa.joram.soap.SoapConnection.send(SoapConnection.java:200)
        at fr.dyade.aaa.joram.Connection.syncRequest(Connection.java:541)
        at fr.dyade.aaa.joram.Connection.<init>(Connection.java:124)
        at fr.dyade.aaa.joram.TopicConnection.<init>(TopicConnection.java:47)
        at fr.dyade.aaa.joram.soap.TopicSoapConnectionFactory.createTopicConnect
ion(TopicSoapConnectionFactory.java:75)
        at infea.server.util.TestBench.openSoap(TestSoap.java:194)
        at infea.server.util.TestBench.doTest(TestSoap.java:104)
        at infea.server.util.TestBench.run(TestSoap.java:80)
        at java.lang. prova.run( prova.java:536)
```

10.6 Test II

Il secondo test consiste nella simulazione di una serie di scambi di messaggi tra utenti già connessi. Ciò che desideriamo valutare è il tempo che intercorre dalla generazione di un messaggio al relativo post.

Il test lancia dieci thread che simulano il comportamento di dieci utenti di Infea, dei quali cinque assumono il ruolo di utenti del laboratorio e cinque il ruolo di utenti della redazione. Per venti volte i cinque thread laboratorio inviano al server una richiesta di inserimento, ciò provoca la generazione di un messaggio e il post dello stesso sul relativo topic dei redattori, tale messaggio verrà ricevuto dai cinque redattori.Il test continua simulando l'invio da parte di tutti i redattori di un messaggio destinato ad un particolare utente del laboratorio.

Verranno calcolate due stime: il tempo impiegato per creare un messaggio destinato alla redazione fino alla realizzazione del post sul topic dei redattori; e il tempo impiegato per creare un messaggio destinato al laboratorio fino alla realizzazione del post sul topic del laboratorio.

L'intero test viene ripetuto quattro volte. Il bootstrap del sistema e la connessione dei dieci utenti ai rispettivi topic precedono la prima esecuzione del test. Per ogni esecuzione del test viene calcolata la media dei risultati ottenuti in ogni valutazione.
Lo stesso test è stato eseguito sulla piattaforma Joram in presenza di Soap e in assenza di Soap.
L'intero processo (bootstrap del sistema, connessione ai topic, quattro esecuzioni del test) viene ripetuto per tre volte.
I tempi sono espressi in millisecondi.

Le tabelle seguenti riportano i valori risultanti.

10.6.1 Test senza SOAP

Prima Prova	Post sul topic dei Redattori	Post sul topic del laboratorio
Test 1	MEDIA: 3.05	MEDIA: 12.29
Test 2	MEDIA: 4.31	MEDIA: 17.79
Test 3	MEDIA: 6.14	MEDIA: 24.89
Test 4	MEDIA: 7.01	MEDIA: 28.96

Seconda Prova	Post sul topic dei Redattori	Post sul topic del laboratorio
Test 1	MEDIA: 3.03	MEDIA: 11.79
Test 2	MEDIA: 4.62	MEDIA: 18.65
Test 3	MEDIA: 5.96	MEDIA: 23.95
Test 4	MEDIA: 6.89	MEDIA: 28.18

Terza Prova	Post sul topic dei Redattori	Post sul topic del laboratorio
Test 1	MEDIA: 3.00	MEDIA: 11.26
Test 2	MEDIA: 4.45	MEDIA: 18.55
Test 3	MEDIA: 5.66	MEDIA: 22.81
Test 4	MEDIA: 6.97	MEDIA: 29.21

Test 5	MEDIA: 7.87	MEDIA: 33.08
Test 6	MEDIA: 9.09	MEDIA: 38.13

Possiamo osservare che con l'aumentare del numero delle esecuzioni del test aumenta il tempo impiegato a eseguire i post. Tra i due post il più costoso risulta essere quello relativo al topic del laboratorio.

10.6.2 Test con Soap

L'esecuzione del test eseguito in presenza del protocollo Soap ha presentato diversi problemi.
Il primo riguardava la generazione dell'eccezione della cui soluzione abbiamo parlato sopra.

Rimane invece aperto un secondo problema consistente nella non terminazione del test. La prima esecuzione del test seguita al boostrap del sistema e alla connessione ai topic degli utenti non arriva a termine, delle prime 100 iterazioni ne vengono eseguite 38. Né una nuova esecuzione è possibile. Il sistema pur non segnalando alcun errore non dà segni di vita. Solo ripetendo il bootstrap e una nuova connessione ai topic è possibile riavviare il test che comunque, anche in questo caso, non termina. Per le considerazioni su questo argomento si vedano le conclusioni riportate alla fine del test I.

Si tenga quindi presente che i valori ottenuti in queste prove sono parziali e si riferiscono a elaborazioni che seguono immediatamente lo start up dell'intero sistema.

Prima Prova	Post sul topic dei Redattori	Post sul topic del laboratorio
Test 1 (parziale)	MEDIA: 5.16	MEDIA: 11.04

Seconda Prova	Post sul topic dei Redattori	Post sul topic del laboratorio
Test 1 (parziale)	MEDIA: 4.97	MEDIA: 10.32

Terza Prova	Post sul topic dei Redattori	Post sul topic del laboratorio
Test 1 (parziale)	MEDIA: 5.74	MEDIA: 11.16

Abbiamo provato a dimezzare il numero di iterazioni interne ad ogni thread, da venti a dieci, l'elaborazione continua a interrompersi. Mentre diminuendo ulteriormente il ciclo interno (passando a tre iterazioni) si ottengono tre esecuzioni del test, le prime due complete e la terza parziale. Di seguito riportiamo le stime relative a questa prova.

Quarta Prova	Post sul topic dei Redattori	Post sul topic del laboratorio

Test 1	MEDIA: 2.42	MEDIA: 10.87
Test 2	MEDIA: 2.48	MEDIA: 11.42
Test 3 (parziale)	MEDIA: 0.99	MEDIA: 23.95

***Osservazione*.**
Un'ulteriore misurazione valutata interessante era quella del tempo occorente ad accodare e a ricevere un messaggio. Non è stato possibile rilevare tali stime dal momento che attualmente il MessageListener riceve più copie dello stesso messaggio. Questo problema è stato segnalato agli sviluppatori di Joram.

10.7 Conclusione

I dati relativi alle prestazioni del protocollo Soap sono in linea di massima migliori di quelli del protocollo nativo. Esistono tuttavia dei limiti di affidabilità dell'attuale implementazione di SOAP in Joram che ci hanno sconsigliato dal suo utilizzo in fase di produzione. Stiamo cercando, in collaborazione con igli sviluppatori del server Joram, di superare i problemi di affidabilità per un futuro impiego di SOAP.

11 Bibliografia

1) *N. Aloia, S. Tardelli, L. Versienti:* **Un sistema informativo basato sul Web**, Proc. Decimo Convegno Nazionale su Sistemi Evoluti per Basi di Dati (SEDB 2002), pp-92-102, Portoferraio giugno 2002.

2) N. Aloia: **Il sistema informativo INFEA – Progettazione della base di dati,** Rapporto CNUCE-B4-2002-05, aprile 2002

3) N. Aloia, C. Concordia F. Furfari: **Technologies for Public Acces Web Information Systems**, EuroWeb2001, Pisa 2001.

4) N. Aloia, C. Concordia, F. Furfari, V. Miori: **Caratteristiche, problematiche e tecnologia dei sistemi informativi basati sul Web**, Rivista di Informatica AICA, vol. XXXI n.2 settembre 2001 (pp. 95-108)

5) *N. Aloia, C. Concordia, F. Furfari*, **A Public Access Web Information System**, Proc. of 2001 International Conference on Intelligent Agents, Web Technologies and Internet Commerce, July 2001, Las Vegas

6) N.Aloia, C. Concordia, V.Miori: **Web Based Information Systems,** Proc. of 16th IFIP World Computer Congress 2000, Information Technology for Business Management, August 2000, Bejing, (pp. 581-588)

7) N.Aloia, C.Concordia, F. Furfari, V.Miori: **Considerazioni per la realizzazione di sistemi informativi basati su Web accessibili ai disabili**, Proc. of 6th Convegno Nazionale Informatica, Didattica e Disabilità – Andria (BA), novembre 1999.

8) N.Aloia, C.Concordia, V.Miori: **Web Architectures for Database Access,** Proc. of WebNet '99 - Honolulu, Hawaii, ottobre 1999, (pp. 1473-1475)

9) N.Aloia, C.Concordia, V.Miori: **INFEA: un sistema informativo per il Ministero dell'Ambiente,** Proc. of AICA '99 (pp. 447-452), Abano Terme, settembre 1999