SVILUPPO DI UNA APPLICAZIONE PER LA GESTIONE DI UNA RUBRICA DI CONTATTI

RELATORE: Ch.mo Prof. Girolamo Gradenigo

LAUREANDO: Roberto Minella

A.A. 2009-2010

UNIVERSITÀ DEGLI STUDI DI PADOVA
DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE
TESI DI LAUREA

SVILUPPO DI UNA APPLICAZIONE PER LA GESTIONE DI UNA RUBRICA DI CONTATTI

RELATORE: Ch.mo Prof. Girolamo Gradenigo

LAUREANDO: *Roberto Minella*

Padova, 12 Febbraio 2010

Alla mia famiglia

Indice

Elenco delle figure

Sommario

Obiettivo generale

Lo scopo di questa attività di tirocinio è stato quello di progettare e implementare un applicativo database per la gestione delle informazioni sui contatti dell'azienda promotrice dello stage.

Attività sviluppata

L' attività si è articolata in tre fasi principali:

1. Acquisizione delle competenze necessarie allo sviluppo dell' applicativo
2. Analisi requisiti funzionali del database e relativa implementazione
3. Analisi requisiti funzionali dell' applicativo e sua realizzazione

Strumenti utilizzati

Per lo svolgimento del tirocinio e la realizzazione dell' applicativo è stato fatto l'uso dell' ambiente di sviluppo Microsoft Visual Studio .NET 2003 del linguaggio di programmazione C# per quanto riguarda la parte dell' applicativo, mentre invece la parte relativa al DataBase è stata implementata attraverso il DataBase Management System MaxDB.

Risultati raggiunti

Il lavoro, nelle sue componenti principali, è stato portato a termine nelle scadenze previste ed è risultato attinente alle specifiche progettuali e requisiti funzionali minimi richiesti.

Introduzione

Il tirocinio, proposto dall'Associazione Industriali della Provincia di Belluno, è stato svolto presso la ditta Vemer S.p.A. situata in località Villapaiera di Feltre, per una durata di 500 ore, nel periodo febbraio - maggio 2007.

Vemer Spa nasce nel 1980 con il nome di Vemer Elettronica Srl, e sin da subito si colloca ai vertici del mercato italiano quale azienda produttrice di soluzioni per la misura e la gestione delle grandezze elettriche ed ambientali in ambito civile ed industriale. In questi anni Vemer è passata attraverso evoluzioni aziendali che hanno determinato la nascita, nel 2001, di un gruppo industriale europeo che prende il nome di Vemer Siber Group.

Nel 2007, specifiche fasi e processi storici ed economici determinano tuttavia lo sviluppo ed il cambiamento della struttura interna aziendale. Vemer si stacca infatti dal gruppo e, con il nome di Vemer Spa, torna a viaggiare da sola. Vemer da allora si rinnova e si presenta al mercato con una veste tutta nuova, coerente con gli obiettivi posti ed in linea con le reali potenzialità acquisite e sviluppate in oltre vent'anni di esperienza.
La sede è sempre quella di Feltre, in provincia di Belluno, ove oggi si concentrano sia l'area produttiva sia amministrativa, commerciale e direzionale dell'azienda.

La proposta di Vemer spazia oggi da un'ampia gamma di dispositivi ideali per adattare il clima domestico alle proprie esigenze, a dispositivi per la gestione domestica delle utenze elettriche, a rivelatori di gas per il controllo e la sicurezza negli ambienti domestici, dispositivi per la misura ed il controllo delle principali grandezze elettriche, termometri e regolatori di temperatura , pressione ed umidità e tutti gli accessori ed elementi che determinano il buon lavoro del moderno installatore.

Continui investimenti in Ricerca e Sviluppo, costituiscono gli elementi essenziali per consolidare e sviluppare la posizione di mercato di Vemer. Il carattere flessibile della struttura aziendale permette la progettazione e la realizzazione di

specifiche soluzioni in tempi brevi, garantendo una offerta di prodotti e soluzioni sempre attuale.
L'impegno aziendale nell'assicurare l'efficienza e l'efficacia della propria organizzazione e la qualità dei propri prodotti è garantita dalla condivisione degli obiettivi aziendali tra tutti i componenti del team e trova riscontro concreto nella certificazione del sistema qualità UNI EN ISO 9001:2000.

L'affidabilità del servizio è testimoniata dagli oltre 22.000 articoli che Vemer spedisce ogni giorno ai propri clienti in oltre 44 paesi del mondo. La vastità della propria offerta, composta da oltre 1.000 articoli suddivisi in specifiche famiglie che ne identificano il campo di applicazione, garantisce alla clientela la giusta proposta ad ogni esigenza.

E' proprio da questa vastità di contatti sparsi in molti paesi del mondo, che nasce da parte dell'azienda, la necessità di mantenere tutte le possibili informazioni sui propri clienti in modo ordinato e ben strutturato in modo da poter soddisfare a pieno le loro esigenze e richieste.

Il presente progetto, maturato da queste necessità aziendali, ha quindi come obiettivo lo sviluppo di un' applicativo che permetta una gestione efficace ed efficiente delle informazioni relative ai contatti dell'azienda.

L'attività si è svolta in più fasi distinte, la prima delle quali è stata acquisire le competenze tecniche necessarie allo sviluppo dell'applicativo.
Per realizzare il software è stato usato l'ambiente di sviluppo Microsoft Visual Studio .NET 2003 come IDE per il linguaggio di programmazione C#. Per l'interazione con la base di dati, realizzata con il DBMS MaxDB, è risultato molto utile e efficace l'uso di un ricco insieme di oggetti resi disponibili dalla piattaforma .NET; queste classi di oggetti sono raggruppate assieme e vengono riferite come ADO.NET. Si è scelto di utilizzare queste tecnologie poiché sono quelle attualmente impiegate presso l'azienda, per le quali la ditta ha maturato una buona esperienza ed è quindi anche in grado di dare un supporto adeguato.

Nella prima parte dell'elaborato verranno illustrate le principali caratteristiche degli strumenti software utilizzati, si illustreranno i concetti principali sia del linguaggio C#, del componente ADO.NET e del DBMS MaxDB. Nel secondo capitolo invece si passeranno in rassegna i vari requisiti che dovranno soddisfare sia la base di dati che l'applicativo che ad essa si interfaccia. Il terzo capitolo invece illustrerà la fase di progettazione del database, partendo da uno schema con-

cettuale fino ad arrivare con raffinazioni successive allo schema relazionale finale. Il quarto capitolo mostra invece la fase di progettazione dell'applicativo, facendo vedere come il componente ADO.NET renda l'applicazione in grado di interagire con il database. Nel quinto ed ultimo capitolo verranno presentate le conclusioni e i possibili sviluppi della soluzione software sviluppata. Per fornire al lettore una migliore comprensione del seguente documento è stata fornita un'appendice contenente ulteriori esempi di codice e specifiche sulle tecnologie adottate.

Capitolo 1

Tecnologie utilizzate

Nel seguente capitolo vengono brevemente illustrati gli strumenti e le metodologie adottati per la progettazione e implementazione dell'intero progetto. Le scelte effettuate garantiscono una riduzione dei tempi di produzione e di aggiornamento dell'applicativo. Nella prima parte del capitolo verranno esposte la struttura generale e le principali caratteristiche del DBMS scelto per la realizzazione del database, mentre nella seconda una panoramica degli strumenti utilizzati per l'implementazione del software.

1.1 MySQL MaxDB

MaxDB, sviluppato da SAP AG in collaborazione con MySQL AB sotto licenza GNU GPL (GNU General Public License), è un sistema di gestione di basi di dati relazionale o anche RDBMS (Relational DataBase Management System), usato per creare utilizzare e gestire istanze di database di tipo OLTP (OnLine Transaction Processing) e LiveCache SAP.

MaxDB è conforme alle specifiche ANSI SQL-92 ed è mirato per grandi ambienti SAP ad esempio mySAP Business Suite e altre applicazioni che richiedono funzionalità di database enterprise-level, offre per questo alta affidabilità, disponibilità e scalabilità ed anche i seguenti vantaggi:

- Facile configurazione e amministrazione attraverso interfaccia grafica
- Sofisticate funzionalità di backup e ripristino (online backup e backup incrementali)
- Supporto per un grande numero di utenti e decine di terabyte di volumi di dati
- Gestione automatica dello spazio dei volumi, nessuna necessità di riorganizzazione dei dati dopo ristrutturazioni
- Alta affidabilità attraverso configurazioni hot standby e supporto a cluster

Tra le caratteristiche principali del sistema,come specificato in [7], troviamo la presenza nel kernel di precompilatori PHP, Perl, Python, WebDAV, OLE DB, ADO, DAO, RDO e .NET tramite ODBC, Delphi e Tcl tramite interfacce di programmazione di terze parti. La figura 1.1 illustra le principali componenti del DBMS MaxDB.

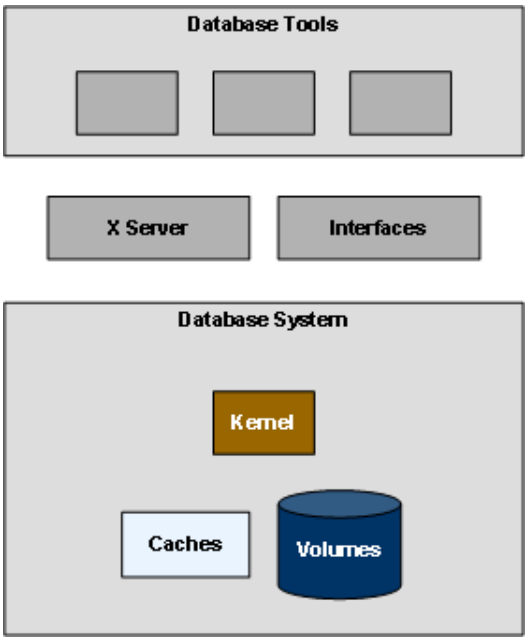

Figura 1.1: Componenti DBMS

In una istanza di database i dati vengono divisi in due categorie: dati relativi alla realtà modellata e metadati cioè l'informazione che li descrive. I dati vengono memorizzati in modo persistenti in volumi e durante il funzionamento parte di quelli utilizzati si trovano in diverse memorie cache presenti nella memoria principale.

Il sistema mette a disposizione diversi strumenti per lavorare con le istanze di database, come ad esempio il Database Manager e l'SQL Studio che interagiscono con le istanze attraverso il Server X.
E' possibile connettere anche altre applicazioni alle istanze attraverso l'uso di interfacce come ad esempio l'interfaccia ODBC (Open DataBase Connectivity) o JDBC (Java DataBase Connectivity).

L'istanza di un database si può trovare in diversi stati operativi (offline ,admin e online) e ad ognuno di questi stati corrisponde un relativo stato del kernel del DBMS che viene lanciato come un servizio del sistema.
Il kernel durante il funzionamento crea threads diversi per ogni task specifico lanciato dal sistema (query ,connessioni al database, backup) che vengono eseguiti in parallelo dal DBMS.

1.2 Microsoft .NET Framework

Il framework .NET è l'infrastruttura che costituisce la nuova piattaforma creata da Microsoft per lo sviluppo di applicazioni component-based, n-tier, per internet, per l'accesso ai dati, o semplicemente per le classiche applicazioni desktop. La piattaforma .NET è composta da diverse tecnologie, strettamente accoppate fra loro. In questa sezione verranno introdotte le nozioni fondamentali della piattaforma e dello strumento utilizzato per l'interazione dell'applicativo con i dati presenti nel database.

1.2.1 Architettura .NET

L'architettura del framework .NET è illustrata nella figura 1.2. Essa si appoggia direttamente al sistema operativo, nella figura viene indicato Windows, ma esistono e sono a buon punti progetti per il porting su ambienti diversi, ad esempio Linux.

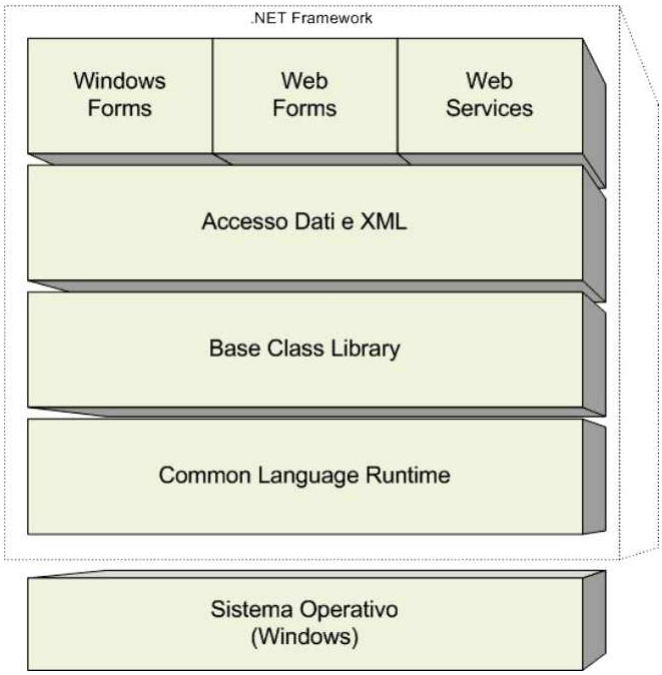

Figura 1.2: Architettura .Net Framework

Il framework .NET consiste di cinque componenti fondamentali.
Il componente anima e cuore dell'intero framework, è il Common Language Runtime (CLR). Esso fornisce le funzionalità fondamentali per l'esecuzione di una applicazione managed (gestita appunto dal CLR). Il CLR, a basso livello, si occupa inoltre dell'interfacciamento con il sistema operativo.

Lo strato immediatamente al di sopra del CLR, è costituito dalla Base Class Library (BCL) di .NET, cioè un insieme di classi fondamentali, utili e necessarie a tutte le applicazioni ed a tutti gli sviluppatori (ad esempio classi di input/output, per la connettività, per la gestione di collezioni di oggetti). Naturalmente altre classi specializzate saranno sicuramente mancanti nella BCL. Al di sopra della BCL, vengono quindi fornite le classi per l'accesso alle basi di dati e per la manipolazione dei dati XML.

Lo strato più alto del framework è costituito da quelle funzionalità che offrono un interfacciamento con l'utente finale, ad esempio classi per la creazione di interfacce grafiche desktop, per applicazioni web, o per i sempre più diffusi web service.

Come detto, il componente cardine dell'architettura è il CLR, che gestisce l'esecuzione dei programmi scritti per l'ambiente .NET. Principalmente si oc-

cupa dell'instanziazione degli oggetti, esegue dei controlli di sicurezza, ne segue tutto il ciclo di vita, e al termine di esso esegue anche operazioni di pulizia e liberazione delle risorse. In .NET ogni programma scritto in un linguaggi supportato dal framework viene tradotto in un linguaggio intermedio comune detto CIL (Common Intermediate Language), ed a questo punto esso può essere tradotto e assemblato in un eseguibile .NET specifico per la piattaforma su cui dovrà essere eseguito. In effetti, a run-time, il CLR non conosce e non vuole conoscere in quale linguaggio è stato scritto il codice, il risultato della compilazione è un modulo managed (che contiene il codice CIL e una descrizione dei tipi usati nel codice sorgente, i metadati), indipendente dal linguaggio utilizzato. Il codice CIL non è direttamente eseguibile dal microprocessore, dunque esso deve essere tradotto in codice nativo, in base alle informazioni contenute nei metadati. Questo compito viene svolto, a tempo di esecuzione, dal compilatore JIT (Just In Time).

L'architettura .NET, a differenza di Java, è multi linguaggio, cioè non è legata ad un linguaggio specifico: si può scrivere un programma in un qualunque linguaggio supportato, compilarlo ed eseguirlo su di una piattaforma .NET.Inoltre, oggetti scritti in un linguaggio , possono essere compilati e riutilizzati come moduli in programmi scritti in un altro linguaggio.Per garantire questa forma di interoperabilità sono stati sviluppati all' interno di .NET un insieme di tipi comuni, detto Common Type System (CTS), che definisce appunto i tipi comuni e le loro politiche di utilizzo e gestione da parte del CLR.

1.2.2 Microsoft ADO.NET

ADO.NET (ActiveX Data Objects) è una collezione di classi, interfacce, strutture, e tipi che gestisce l'accesso ai dati all'interno nel .NET framework.

Come tutti gli altri componenti del .NET framework, ADO.NET consiste di un insieme di oggetti che interagiscono fra loro per svolgere una determinata funzione. La figura sottostante mostra una vista semplificata degli elementi che compongono ADO.NET.

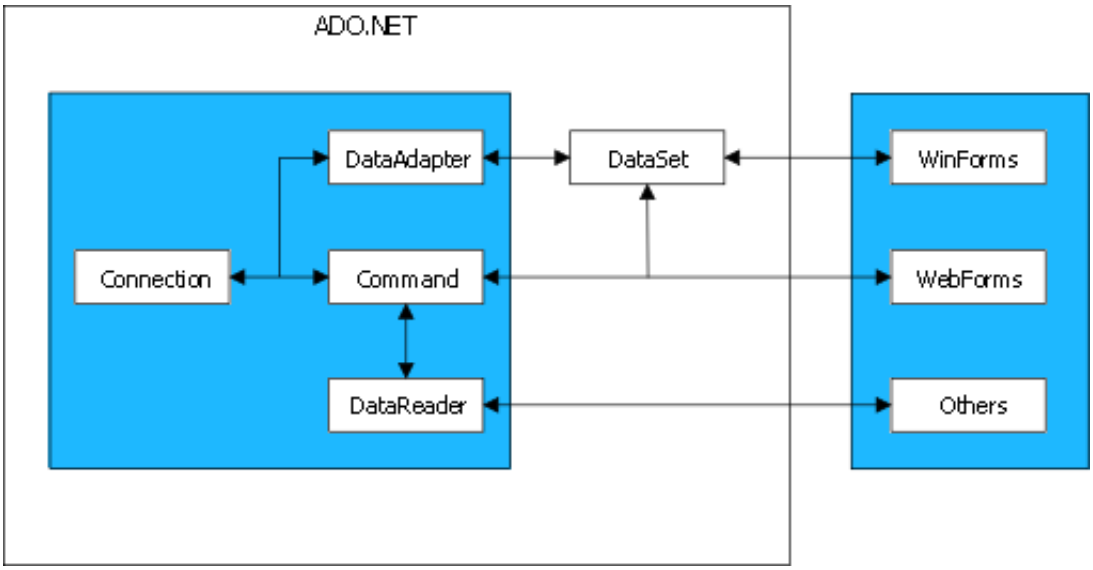

Figura 1.3: Struttura ADO.NET

Per una più facile interoperabilità, ADO.NET utilizza l' XML (eXtensible Markup Language) come formato nativo per i dati, mentre l'accesso a questi ultimi è stato progettato sulla base di un'architettura disconnessa. Ciò significa che le applicazioni rimangono connesse al database solamente per il tempo necessario per estrarre o aggiornare i dati, questo porta a numerosi vantaggi primo su tutti il minore carico di lavoro subito dal database, ed inoltre la maggior velocità di esecuzione del software, in quanto esso effettuerà le modifiche ai dati su di una copia degli stessi tenuta in memoria, senza attendere il database server.

ADO.NET , come illustrato in [6], è costituito da due componenti fondamentali, il Data Provider e il DataSet.

Il Data Provider si pone nella parte più bassa del flusso dei dati, in quanto ha il compito di comunicare direttamente con il database, stabilirne una connessione, eseguire comandi e recuperare risultati. Il .Net Framework viene fornito con due Data Provider : SQL Server Data Provider per le connessioni a SQL Server, e OLE DB .NET Data Provider per le connessioni a sorgenti OLE DB (Object Linking and Embedding Database).
Ognuno di questi Data Provider ha al suo interno diversi oggetti che forniscono le funzionalità essenziali per l'accesso disconnesso ai dati, dall'instaurazione della connessione con il database alla costruzione di un flusso di dati ad elevate prestazioni dall'origine di dati ed infine oggetti per l'esecuzione di comandi SQL sull'origine di dati.

L'altro componente fondamentale di ADO.NET è il DataSet. In un modello di dati disconnesso, non è pratico accedere al database ogni volta che l'applicazione deve elaborare un record successivo a quello in corso di elaborazione, è quindi preferibile lavorare su di una copia dei dati residente in memoria, questa replica dei dati viene chiamata DataSet.L'oggetto `DataSet` è fondamentale per il supporto offerto da ADO.NET in scenari di dati disconnessi e distribuiti. Il DataSet è una rappresentazione dei dati residente in memoria che fornisce un modello relazionale coerente e indipendente dall'origine di dati. Il DataSet rappresenta un insieme completo di dati che include tabelle correlate, vincoli e relazioni tra le tabelle.
L'oggetto `DataSet` è costituito da un insieme di oggetti `DataTable` (a sua volta divisi in oggetti `DataRow`, `DataColumn` e `Constraint`) che è possibile porre in relazione tra loro mediante oggetti `DataRelation`. E' inoltre possibile applicare l'integrità dei dati nell'oggetto DataSet utilizzando gli oggetti `UniqueConstraint` e `ForeignKeyConstraint`.

I dati in memoria vengono memorizzati, attraverso il DataSet, in formato XML, cosicché i dati e gli schemi possono essere trasferiti via http e utilizzati da qualsiasi piattaforma con supporto XML. Uno dei motivi per il quale si è scelto di memorizzare i dati dal DataSet attraverso documenti XML è il fatto che questo linguaggio rappresenta i dati in formato testuale, e non binario, ed è quindi possibile utilizzare qualsiasi protocollo di trasmissione per inviare informazioni in formato XML.

Capitolo 2

Analisi dei requisiti funzionali

Il primo e fondamentale passo per la progettazione di una base di dati è quello di raccolta e analisi dei requisiti funzionali che il database deve soddisfare.
Per raccolta dei requisiti, come specificato in [2],si intende la completa individuazione dei problemi che l'applicazione da realizzare deve risolvere e le caratteristiche che tale applicazione dovrà avere. Per caratteristiche del sistema si intendono sia gli aspetti statici (i dati) sia gli aspetti dinamici (le operazioni sui dati). I requisiti vengono inizialmente raccolti in specifiche espresse generalmente in linguaggio naturale e, per questo motivo, spesso ambigue e disorganizzate. L'analisi dei requisiti consiste nel chiarimento e nell'organizzazione delle specifiche dei requisiti. Le principali fonti di informazioni per la raccolta dei requisiti sono:

- Gli utenti dell'applicazione: informazioni ottenute mediante interviste, anche ripetute, o attraverso una documentazione scritta da utenti

- La documentazione esistente: operazione assistita dagli utenti ma a carico del progettista

- Eventuali realizzazioni preesistenti: applicazioni che si devono sostituire o interagire con il sistema da realizzare

In questa fase è molto importante individuare gli aspetti essenziali rispetto a quelli marginali, e procedure per raffinamenti successivi.

Caratteristica fondamentale che contraddistingue l'attività di analisi dei requisiti, risulta essere la stretta interazione che viene a crearsi fra l'analista e le varie figure aziendali, tra le quali spiccano:

- Un responsabile di settore in grado di fornire una buona visione di insieme della società e della relativa attività economica
- Un rappresentante degli utenti del sistema informativo con il quale dettagliare le procedure da eseguire
- Un amministratore della base dati a conoscenza delle politiche di sicurezza e gestione del sistema di archiviazione

2.1 Analisi del problema

Negli ultimi anni, l'impresa ha subito una forte espansione del proprio mercato, con un conseguente incremento della quantità di persone che contattano l'azienda sia a scopo puramente informativo sia a scopo commerciale.

Per poter mantenere la propria posizione all'interno del mercato del settore, l'azienda punta a conoscere in modo approfondito e dettagliato ogni singolo aspetto della propria clientela, per poter così garantire ai propri acquirenti di soddisfare le loro esigenze e richieste. A tale scopo si dovrà avere la consapevolezza di ampliare le conoscenze a tutti gli ambiti di interesse verso la ditta.

E' stato proficuo un colloquio preliminare con il responsabile commerciale e marketing dell'azienda, per conoscere quali aspetti aggiungono informazioni utili, e poter rendere più efficiente la gestione dei contatti.

Il primo aspetto considerato è stato quello del singolo contatto. Oltre alle basilari informazioni anagrafiche (nome, cognome, residenza ecc..) sono emersi alcuni aspetti fondamentali per la gestione del cliente, che sono:la sua qualifica, il suo settore di interesse e la sua attività principale.
La qualifica sta a indicare le competenze lavorative del soggetto, il settore di interesse riguarda invece il campo di applicazione dell'attività del contatto. I principali settori ai quali l'azienda fa riferimento sono: elettrico, termoidraulico ed export.

Il settore elettrico comprende tutti quei prodotti che possono soddisfare le esigenze di moderni elettricisti, quali ad esempio contatori, trasformatori,rivelatori di presenza, strumenti di misura, interruttori e accessori di cablaggio.

Nel settore termoidraulico invece si collocano prodotti come termometri, termoregolatori, termostati, rivelatori di gas e contabilizzatori di energia, articoli maggiormente usati da idraulici e installatori di caldaie termiche.

L'ultimo settore, quello dell'export, comprende invece tutti gli articoli presenti nei primi due settori, ma aventi localizzazione per il commercio estero.
Queste tre ulteriori caratteristiche del singolo contatto permettono ai responsabili marketing e commerciale di conoscere in modo più approfondito la clientela e soddisfare in modo radicale e incisivo richieste e bisogni.

Per mantenere e possibilmente aumentare il rapporto di stima e fiducia che hai nei propri clienti, l'azienda attua nei loro confronti una serie di promozioni su particolari prodotti. Risulta utile, quindi, sapere quali contatti sono stati "abilitati" a ricevere una promozione e, soprattutto, per ogni cliente le promozioni a lui attuate.
Scopo della promozione è quello di incentivare la vendita di determinati prodotti, ricevendo un omaggio, raggiunta la soglia minima di acquisto. Ogni offerta ha un periodo di validità, e quindi una data di inizio e di fine associata. Inoltre la promozione si compone di un regolamento, che ne spiega le norme, e di un volantino che la illustra. Volantino e regolamento vengono solitamente esposti alle fiere di settore a cui l'azienda partecipa, oppure, per avere più visibilità, nell'apposita sezione del sito internet della società.

Oltre che attraverso promozioni, gli omaggi o gadget vengono dati ai clienti in occasioni speciali che non sono vincolate necessariamente all'acquisto di prodotti. Ecco quindi che nasce la necessità di conoscere: quali gadget sono stati consegnati a quali contatti per evitare situazioni di omaggi ripetuti, il motivo per cui è stato assegnato l'articolo e l'indicazione della data di carico in magazzino dell'oggetto. Questa dà indicazioni precise riguardo la modernità, attualità e freschezza dell'omaggio.

Sempre continuando il colloquio con il responsabile commerciale, è apparso chiaro che il principale punto di contatto e tra azienda e futuri acquirenti si ha nelle fiere di settore. Qui i responsabili di vendita, dialogando con gli interessati, possono capire quali sono le loro richieste e necessità e quindi cercare di soddisfare

i loro requisiti proponendo il prodotto ad essi più adeguato. Esistono numerose fiere annuali, sia in ambito nazionale che internazionale, ognuna delle quali si rivolge ad un determinato settore elettrico, dai prodotti per comfort e sicurezza, agli articoli per misura e controllo di grandezze.
Nello stand dell'azienda presso la fiera, i visitatori hanno la possibilità di prendere accordi per un contratto di fornitura di prodotti oppure possono lasciare le loro credenziali per poter essere informati, tramite newsletter, su nuovi apparati o su promozioni esistenti.
Sapere a quale fiera è stato iniziato il rapporto con un cliente, costituisce per la ditta un' informazione utile a conoscere il principale settore di interesse dei suoi contatti.

Un altro mezzo che permette all'azienda di rimanere in contatto con i propri clienti è lo strumento della newsletter. Attraverso il sito internet, è possibile iscriversi gratuitamente alla newsletter inserendo i propri dati personali nonché il settore inerente la propria attività e gli interessi specifici verso l'azienda. La mail inviata conterrà aggiornamenti sulle novità riguardanti gli interessi specificati in fase di iscrizione e, indeterminati allegati, informazioni tecniche specifiche. Attraverso la registrazione, l'utente comunica così i suoi interessi all'azienda, la quale potrà così tenerlo sempre informato su tutti i prodotti e novità di mercato. Può risultare utile mantenere traccia delle newsletter inviate per verificare che tutti i contatti siano costantemente aggiornati sulle novità dell'offerta.

Gli interessi che un generico contatto può avere nei confronti della ditta, si possono sintetizzare in sei famiglie, che sono: gestione clima, comfort, sicurezza gas, misura e controllo, termoregolazione e installazione.

Queste sei categorie altro non rappresentano che i diversi ambiti di mercato in cui si collocano gli articoli prodotti. Nella gestione clima, ad esempio si trovano prodotti per adattare il clima domestico alle proprie esigenze. Nella categoria comfort invece,ci sono dispositivi per la gestione elettrica delle utenze domestiche come interruttori crepuscolari o rivelatori di presenza e inseritori luci scale. Sicurezza gas racchiude componenti utili nella sicurezza degli ambienti domestici contro eventuali fughe di gas, ad esempio rivelatori di gas metano. Appartengono alla classe misura e controllo tutti quei dispositivi adatti al monitoraggio delle principali grandezze elettriche e alla misura di energia attiva e reattiva nei sistemi monofase e trifase AC. La categoria termoregolazione comprende una vasta

gamma di termometri per la visualizzazione della temperatura e di regolatori di temperatura, pressione e umidità. L'ultima famiglia, installazione, contiene una serie di prodotti per l'installazione di un sistema elettrico, come morsetti o passacavi.
La conoscenza, pertanto, di quali famiglie di interessi legano la clientela all'azienda, può risultare utile per ideare promozioni o newsletter specifiche mirate a soddisfare ogni particolare esigenza degli acquirenti. Non sono da sottovalutare inoltre gli interessi generici non orientati alle attrattive specifiche dell'azienda, ma rivolti ad altre soluzioni per la misura e la gestione delle grandezze elettriche ed ambientali. Anche questi possono essere utili per capire i gusti e aspettative dei contatti e formulare così le giuste risposte alle loro necessità.

Prendendo in considerazione tutti gli svariati aspetti emersi durante la fase di raccolta dei requisiti, è facile intuire che l'attuale sistema di archiviazione dei contatti non risulta adeguato. Esso infatti si compone di una serie di fogli di calcolo in formato Excel che memorizzano in forma di elenco tutte le informazioni anagrafiche e professionali dei clienti, nonché dati su fiere e promozioni. E' evidente che un tale sistema risulta inefficiente, sia in termini di prestazioni sia di affidabilità. Inoltre può soffrire di problemi di inconsistenza e integrità dei dati, in quanto il foglio di calcolo non permette di gestire relazioni tra concetti distinti, ma al contrario consente solo di rappresentare le caratteristiche di un singolo oggetto.

Si cercherà ora, per una più chiara lettura, di dividere l'attuale analisi dei requisiti in due parti: una legata in modo più stretto alla descrizione dei dati da memorizzare (analisi dei dati), l'altra più incentrata sulle operazioni fondamentali che si devono poter svolgere sulla base di dati da realizzare (analisi delle funzionalità). Alla fine della sezione invece, verrà elencato un glossario dei termini più significativi emersi durante la fase di analisi dei requisiti che riassumerà appunto i concetti principali che sono stati inferti.

2.1.1 Analisi dei dati

Esaminando, in uno stadio successivo, le informazioni ottenute durante la fase di raccolta dei requisiti, si possono derivare i seguenti concetti che modellano, in modo accurato, la realtà considerata. Per ogni astrazione identificata verranno elencate le principali caratteristiche che la descrivono, accompagnate da una

breve descrizione, ove il significato non fosse chiaro a priori.

CONTATTI

Rappresenta l'insieme di tutte le persone che hanno un qualsiasi tipo di rapporto con l'azienda, ad esempio clienti o persone che chiedono informazioni di supporto su prodotti od anche fornitori di materiale per la produzione.Le caratteristiche specifiche da considerare sono riportate di seguito:

- *Identificativo:* IdContatto
  Permette di identificare univocamente un contatto nell'insieme dei contatti.
- *Dati anagrafici:* nome, cognome, codice fiscale, indirizzo, cap, città, provincia, nazione
- *Dati di contatto:* telefono, fax, cellulare, e-mail, e-mail alternativa, eventuale sito web
- *Dati relativi all'attività:* ragione sociale, partita iva, qualifica, settore, attività
  Rappresentano i dati che consentono di distinguere i vari contatti in base alla loro posizione lavorativa.
- *Dati relativi ad azioni da svolgere sul contatto:* flag newsletter, flag promozione
  Questi flag stanno ad indicare l'abilitazione del contatto a ricevere la newsletter dell'azienda o promozioni specifiche.
- *Altri dati:* Nota.
  Note di carattere generale che si possono usare come eventuali informazioni aggiuntive sul contatto.

FIERE

Concetto che rappresenta le fiere di settore a cui l'azienda partecipa.
Principali attributi da considerare:

- *Identificativo:* IdFiera
  Permette di identificare una singola fiera sulla globalità delle fiere presenti.
- *Dati relativi al luogo:* città, nazione
- *Dati relativi al periodo di svolgimento:* data inizio, data fine
- *Dati generali:* descrizione, nota
  Rappresentano una breve descrizione della fiera in riguardo al settore merceologico al quale si riferisce ed eventuali note di carattere generale.

PROMOZIONI

Questa classe di oggetti rappresenta appunto le promozioni che vengono effettuate in merito a particolari prodotti e sono rivolte ai clienti dell'azienda.
Le principali proprietà che la caratterizzano sono:

- *Identificativo:* IdPromozione
  Identifica univocamente una promozione all'interno di tutte le promozioni.
- *Dati relativi al periodo di validità:* data inizio, data fine
- *Dati relativi all'omaggio:* codice gadget
  Rappresenta il codice dell'omaggio associato alla promozione.
- *Dati sulla regolamentazione:* link regolamento, link volantino
  Sono due link che puntano uno al file che contiene le norme che regolano la promozione e l'altro al volantino che la illustra, il quale verrà esposto in occasione delle fiere di settore ed anche sul sito web dell'azienda.
- *Dati generali:* descrizione, nota
  Descrizione contiene una rappresentazione sommaria delle caratteristiche della promozione, mentre nota viene usato per eventuali annotazioni generali.

GADGET

Il gadget rappresenta l'oggetto che viene dato in omaggio ad un cliente che usufruisce di una determinata promozione o che viene assegnato come segno di gratitudine per la clientela affezionata.
I parametri caratteristici che lo descrivono sono:

- *Identificativo:* IdGadget
  Permette di identificare un gadget nell'insieme.

- *Dati generali:* descrizione, nota
  Il primo attributo descrive le principali caratteristiche dell'oggetto.

- *Dati relativi alle motivazioni dell'offerta:* causale
  Rappresenta appunto il motivo per cui è previsto il gadget, ad esempio può essere stato abbinato ad una promozione oppure dato come premio per occasioni speciali per la clientela affezionata.

NEWSLETTER

La newsletter risulta un utile strumento per mantenere aggiornati i clienti sulle novità aziendali. I campi che la descrivono in maniera adeguata sono:

- *Identificativo:* IdNewsletter
  Identifica univocamente una newsletter all'interno della collezione delle newsletter.

- *Dati generali:* descrizione, nota, data
  Descrizione riassume brevemente l'oggetto e il contenuto della newsletter, mentre data indica la data di creazione.

- *Dati sull'oggetto:* testo
  Rappresenta il contenuto testuale della newsletter.

ALLEGATI

La definizione di allegato è distinta dal concetto di newsletter. Nel nostro caso l'allegato è un documento aggiunto all'e-mail che contiene informazioni dettagliate sull'oggetto della newsletter. L'unico attributo individuato che lo descrive in maniera adeguata è il link del file allegato contenuto nella e-mail inviata. Questo costituisce anche l'identificatore dell'entità in quanto si presuppone che ogni singolo allegato corrisponda a file con nomi distinti l'uno dall'altro.

FAMIGLIE INTERESSI

Con questo concetto si vogliono modellare per l'appunto gli interessi del contatto verso l'azienda, dove con il termine famiglia si vuole intendere l'ambito di mercato in cui si collocano gli articoli. Come specificato in precedenza sono state individuate sei famiglie di prodotti per i quali i contatti possono esprimere il loro interesse. Le famiglie di interessi sono state considerate come concetto a sé stante in quanto presentano una esistenza concettuale utile ai fini dell'applicazione.
Le proprietà principali sono:

- *Identificatore:* IdFamiglia

- *Dati generali:* descrizione
  Descrizione della famiglia, ad esempio termoregolazione, misura e controllo ecc...

INTERESSI

Questo concetto si può trattare in modo analogo al precedente, con l'unica distinzione che questo termine rappresenta gli interessi generici del contatto, rivolti ad altre soluzioni per la misura e la gestione delle grandezze elettriche ed ambientali. Come per il caso delle famiglie di interessi, attributi principali per questa entità sono il suo identificatore e la sua descrizione. Anche in questo caso, gli interessi sono stati modellati come elementi a sé stanti in quanto dotati di una esistenza concettuale ai fini dell'applicazione.

Per ogni entità considerata, inoltre saranno previsti alcuni attributi comuni utili a identificare l'utente del sistema che ha effettuato operazioni sulla base di dati. Questo per motivi di sicurezza e responsabilità aziendali.

Quella fatta fin'ora vuole essere una descrizione generale dei concetti che si sono dedotti da una prima analisi dei requisiti, che risulta utile per lo sviluppo di uno schema concettuale che rispecchi le specifiche acquisite. Maggiori informazioni dettagliate riguardo le proprietà delle entità modellate e sulle relazioni che le coinvolgono, verranno quindi date nella fase di progettazione logica della base di dati.

Con l'analisi degli interessi specifici, si chiude l'analisi dei dati, improntata come visto su una serie di liste di qualità specifiche. Si andranno ora ad analizzare e elencare quelle che dovrebbero essere le operazioni da eseguire sul database.

2.1.2 Analisi delle funzionalità

La base di dati dovrà fornire all'utente la possibilità di eseguire svariati tipi di operazioni, tra le quali l'inserimento di nuove tuple, la loro cancellazione ed eventuali modifiche a valori di campi all'interno di righe nelle tabelle. Sarà necessario, inoltre, effettuare interrogazioni per la ricerca di particolari valori di attributi che soddisfano a determinate condizioni.

Verranno considerate ora, per ogni entità, le principali operazioni che il database dovrà rendere disponibili all'utente.

CONTATTI

- Inserimento nuovo contatto
- Cancellazione contatto esistente
- Modifica di valori di determinati attributi
- Ricerca di un contatto in base a determinati attributi o proprietà derivanti da relazioni con altre entità. Per gli attributi propri del contatto si effettuerà la ricerca in base alla provincia di residenza e alla nazione di residenza nel caso si tratti di un contatto con domicilio estero. Altre proprietà che vengono usate come attributi di ricerca sono la presenza dei flag di abilitazione alla ricezione di newsletter e promozioni. Per quanto riguarda le proprietà

non intrinseche del contatto invece, si vorranno ricercare tutti i contatti che hanno partecipato a determinate fiere, ricevuto particolari promozioni o che hanno certi interessi specifici.

PROMOZIONI

- Inserimento nuova promozione
- Cancellazione promozione esistente
- Modifica di attributi della promozione

FIERE

- Inserimento nuova fiera
- Cancellazione fiera esistente
- Modifica valori di attributi di una fiera

GADGET

- Inserimento nuovo gadget
- Cancellazione gadget esistente
- Modifica valori di campi di un gadget esistente

NEWSLETTER

- Inserimento nuova newsletter
- Cancellazione newsletter esistente
- Modifica attributi newsletter

ALLEGATI

- Inserimento nuovo allegato
- Cancellazione allegato
- Modifica link allegato

INTERESSI

- Inserimento nuovo interesse
- Modifica descrizione interesse
- Cancellazione interesse

FAMIGLIE INTERESSI

- Inserimento nuova famiglia
- Cancellazione famiglia esistente
- Modifica famiglia

Oltre a queste classiche operazioni sui singoli concetti individuati, dall'analisi dei requisiti risulta chiaro che il database dovrà fornire anche altri tipi di operazioni, tra le quali le più importanti sono:

- Visualizzazione di tutte le fiere a cui ha partecipato un contatto
- Visualizzazione di tutte le promozioni di cui ha usufruito un contatto
- Visualizzazione, indipendentemente dalle promozioni, di tutti gadget ricevuti dai contatti
- Visualizzazione dello storico di newsletter spedite ad un determinato contatto
- Visualizzazione di tutti gli interessi di un determinato contatto ed inoltre le famiglie di interessi da lui specificate

In questa analisi sono state considerate tutte le possibili operazioni che la base di dati dovrà supportare anche se, come si può intuire facilmente, alcune di queste avranno una frequenza di utilizzo praticamente nulla. Questa scelta garantisce così eventuali sviluppi futuri della soluzione software in risposta alla comparsa di eventuali prossime esigenze.

2.1.3 Stima del carico applicativo

Per quanto riguarda la stima del carico applicativo che dovrà supportare la base di dati, non ne è stata fatta un'analisi approfondita ma sono state tratte solo alcune considerazioni sul volume dei dati e sulla frequenza delle operazioni fondamentali.

Dai colloqui con il personale dell'azienda e l'analisi di alcune loro stime e statistiche, si è potuto stimare il probabile carico applicativo al quale andava incontro il database. Dalle statistiche è emerso che l'azienda attualmente conta circa 3500 contatti e partecipa mediamente a 10-15 eventi fieristici nazionali e internazionali nell'arco di un anno. Le newsletter verranno inviate con cadenza mensile, salvo ulteriori messaggi per eventi particolari, ai contatti che soddisfano alle condizioni di invio stabilite. Sicuramente quindi, si può affermare che l'azienda comporrà circa 12-15 newsletter annuali, ma nulla si può dire in riguardo al numero di destinatari delle e-mail. Anche in riguardo alle promozioni non si può stimare un dato preciso in quanto la scelta di attivare o meno una promozione dipende da molti fattori legati principalmente alla politica commerciale dell'azienda e all'andamento attuale del mercato di settore.

Il numero di gadget che l'azienda avrà a disposizione non è noto a priori ma si può cercare di darne una stima considerando che ad ogni promozione può venire associato un oggetto omaggio a scelta tra massimo quattro. Supponendo poi che vengano attivate annualmente una decina di promozioni, si può dare un limite superiore alla quantità di gadget presenti nella misura di 40 unità.

Considerando invece la frequenza con cui si andranno a compiere le operazioni, si può affermare che azioni di inserimento di contatti saranno molto più frequenti di operazioni di modifica e cancellazione. Infatti, la cancellazione del contatto si ha solamente in caso di termine del rapporto con l'azienda, mentre la modifica avviene solamente quando si riscontrano errori di inserimento dell'anagrafica.

Una volta al mese invece, avverrà l'inserimento della nuova newsletter e la ricerca dei contatti per l'invio e il successivo inserimento della e-mail nella lista

dei messaggi inviati a ogni contatto. Cancellazioni di newsletter non vengono considerate in quanto si vuole mantenere lo storico delle e-mail inviate.

Anche per quanto riguarda le fiere si può dire che l'inserimento di una nuova fiera avverrà con frequenza pressoché mensile, considerato il fatto che l'azienda partecipa a 12-15 eventi nell'arco di un anno. Anche in questo caso la modifica avviene solamente quando si riscontrano errori di inserimento di valori nella lista di attributi. Come per le newsletter, anche per le fiere è previsto il mantenimento di uno storico e quindi sono quasi nulle le operazioni di cancellazione. Mensilmente avverrà anche l'inserimento delle nuove informazioni aggiornate sulle fiere a cui hanno partecipato i contatti presenti nella base di dati. Analoghe considerazioni si possono trarre per le operazioni relative alle promozioni e gadget.

2.1.4 Glossario dei termini

E' molto utile, per la comprensione e precisazione dei termini usati, definire un glossario che, per ogni concetto, contenga: una breve descrizione, possibili sinonimi e altri termini contenuti nel glossario con i quali esiste un legame logico.

TERMINE	**DESCRIZIONE**	**SINONIMI**	**COLLEGAMENTI**
contatti	persone che hanno un determinato rapporto con l'azienda	clienti, fornitori, agenti	fiere, gadget, promozioni, newsletter, interessi, famiglie interessi
fiere	eventi espositivi periodici a cui l'azienda partecipa	mercato, esposizione, mostra	contatti
promozioni	attività di propaganda e incentivazione commerciale attuata su alcuni prodotti	offerte	contatti
gadget	oggetto omaggio dato come incentivo incluso nella promozione	regalo, presente	contatti
newsletter	notiziario scritto per posta elettronica che dà informazioni di interesse a determinati utenti	messaggio, e-mail	contatti, allegati
allegati	documenti aggiunti all'e-mail contenenti informazioni sulle novità incluse nel messaggio	documentazione a corredo	newsletter
interessi	oggetto o attività di particolare attenzione per soluzioni di misura e gestione di grandezze elettriche e ambientali	attrattiva, desiderio	contatti
famiglie interessi	rami specifici derivanti dalla suddivisione degli articoli prodotti, verso i quali si rivolge particolare attenzione	categorie, gruppi	contatti

Tabella 2.1: Glossario dei termini

2.2 Requisiti funzionali applicativo

Nello sviluppo di un prodotto sono importanti i bisogni degli utenti e non solamente lo stato delle tecnologie. Consultare gli utenti durante la fase di progettazione è fondamentale per riuscire a realizzare un sistema che consenta un buon livello di interazione con essi, che sia quindi usabile, capace di soddisfare gli scopi che una persona si prefigge quando si appresta ad utilizzarlo.

Il sistema software da realizzare utilizza hardware standard svolgendo attività di archiviazione, elaborazione e supporto alla consultazione delle informazioni principalmente in modalità interattiva. Il sistema deve poter essere usato da terminalisti mediante periferiche di data-entry e video consultazione di uso corrente (mouse, tastiera, video). L'utilizzo del sistema deve essere possibile da terminalisti dislocati in diverse postazioni collocate all'interno dell'azienda.

L'interfaccia utente deve essere necessariamente di facile comprensione anche per categorie di persone con un basso grado di preparazione informatica. Allo stesso tempo, le funzionalità di visualizzazione ed elaborazione, devono essere strutturate in maniera tale da consentire una continua evoluzione del software da realizzare. Un' altra funzionalità che l'applicazione dovrà prevedere sarà quella di permettere l'importazione di eventuali dati da fonti esterne, ad esempio da file excel. Questa funzione risulta utile per automatizzare il processo di inserimento dei dati qualora essi provengano già da fonti digitali come fogli di calcolo.

Considerando ora le caratteristiche generali che deve avere l'applicativo, è emersa la necessità di mantenere un file di log dell'applicazione che registri le principali modifiche apportate alla base di dati. Attraverso questo file così, l'amministratore del database potrà analizzare le operazioni fatte nella base di dati, analizzare eventuali segnalazioni di errore oppure produrre statistiche di servizio del sistema.

Passando ora alle caratteristiche specifiche del sistema, si è scelto di descrivere i requisiti funzionali attraverso il linguaggio di modellazione UML (Unified Modeling Language) e in particolare in diagrammi Use Case in modalità grafica. Gli Use Case, come spiegato in [3], si pongono l'obiettivo di determinare i futuri utenti del sistema detti Attori e, per ogni attore, individuare gli obiettivi che intende raggiungere con l'uso del sistema. Per ogni singolo obiettivo viene dettagliata l'interazione con l'attore (cioè un caso d'uso che è proprio un singolo Use Case). Gli Use Case permettono di delineare il sistema, ossia individuarne

i confini e dettagliarne i compiti;inoltre possono essere utilizzati per progettare i test e per realizzare la manualistica utente.

2.2.1 Use case

Nella descrizione grafica di Use Case è possibile scegliere una rappresentazione semplificata e una estesa, a seconda con il livello di familiarità con gli Use Case del cliente. Nei diagrammi seguenti è stata utilizzata la rappresentazione estesa che comprende i costrutti di Include ed Extend.

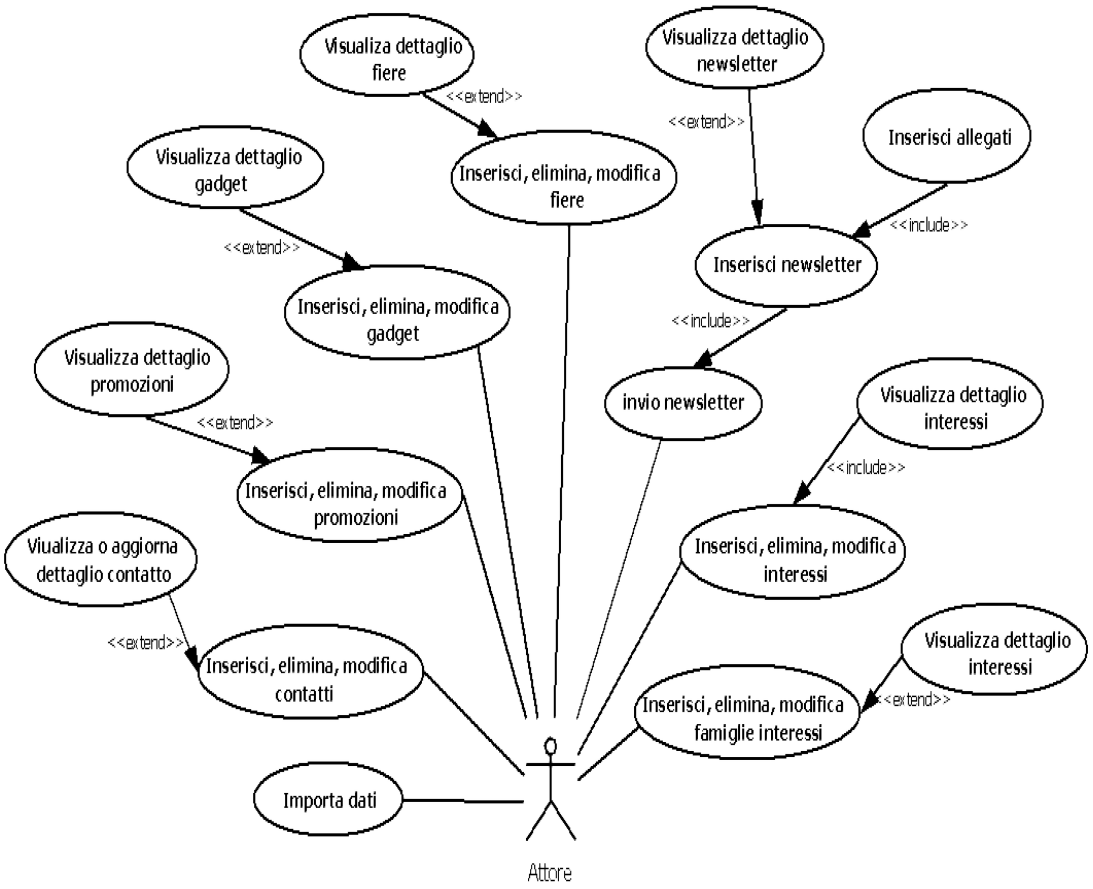

Figura 2.1: Diagramma Use Case

Di seguito sono descritte con maggiore dettaglio, attraverso le specifiche testuali, le funzionalità richieste.

INSERISCI, MODIFICA, ELIMINA CONTATTO

Descrizione:	Consente all'utente di inserire, modificare, eliminare contatti
Attori:	Utente terminalista
Flusso principale:	Il caso d'uso si ha quando l'utente seleziona l'apposito controllo per inserire, modificare o eliminare un contatto
Postcondizioni:	Applicazione permanente delle operazioni sul database

I casi d'uso relativi a inserimento, modifica ed eliminazione di promozioni, gadget, fiere, interessi e famiglie di interessi sono analoghi ma in riferimento alle entità specifiche.

VISUALIZZA O AGGIORNA DETTAGLIO CONTATTO

Descrizione:	Consente all'utente di visualizzare i dati relativi ai contatti presenti nella base di dati o di aggiornare i dettagli del contatto relativi a promozioni, fiere, gadget, interessi e famiglie di interessi associate
Attori:	Utente terminalista
Flusso principale:	Il caso d'uso si ha quando l'utente sceglie di visualizzare informazioni dettagliate sui contatti,o di aggiornare le informazioni relative nelle sezioni dedicate
Postcondizioni:	Visualizzazione delle informazioni presenti o applicazione permanente delle operazioni eseguite sul database

VISUALIZZA DETTAGLIO PROMOZIONE

Descrizione:	Consente all'utente di visualizzare i dati relativi alle promozioni presenti nella base di dati
Attori:	Utente terminalista
Flusso principale:	Il caso d'uso si ha quando l'utente, tramite il controllo associato, seleziona la visualizzazione di promozioni
Postcondizioni:	Visualizzazione nei controlli appropriati delle informazioni relative alle promozioni

I casi d'uso relativi alla visualizzazione di gadget, fiere, interessi e famiglie di interessi sono analoghi ma in riferimento alle singole entità specifiche.

IMPORTA DATI

Descrizione:	Consente all'utente di importare sul database, informazioni presenti su fogli di calcolo come Microsoft Excel
Attori:	Utente terminalista
Flusso principale:	Il caso d'uso si ha quando l'utente,tramite il controllo associato, seleziona la l'importazione dei dati
Postcondizioni:	Importazione dei dati sul database

INVIO NEWSLETTER

Descrizione:	Consente all'utente di inviare tramite posta elettronica ai mittenti selezionati, una specifica newsletter
Attori:	Utente terminalista
Precondizioni:	L'utente deve aver inserito una newsletter con i relativi allegati e ricercato tramite dei filtri i mittenti dell'e-mail
Flusso principale:	Il caso d'uso si ha quando l'utente,tramite il controllo associato, seleziona l'invio della newsletter
Postcondizioni:	Invio della newsletter

INSERISCI NEWSLETTER

Descrizione:	Consente all'utente di salvare sulla base di dati la newsletter selezionata
Attori:	Utente terminalista
Flusso principale:	Il caso d'uso si attiva quando l'utente seleziona l'invio della newsletter. L'inserimento dei dati nel database avviene in background prima dell'invio della mail
Postcondizioni:	Inserimento del dettaglio della newsletter e dei relativi allegati nella base di dati

RICERCA DESTINATARI

Descrizione:	Consente all'utente di cercare tramite un filtro i destinatari della newsletter
Attori:	Utente terminalista
Precondizioni:	L'utente deve aver composto una newsletter con i relativi allegati
Flusso principale:	L'utente,tramite il controllo associato, seleziona i filtri per la ricerca dei destinatari
Postcondizioni:	Creazione di una lista di destinatari per la newsletter

INSERISCI ALLEGATI

Descrizione:	Consente all'utente di inserire allegati in una newsletter
Attori:	Utente terminalista
Precondizioni:	L'utente deve aver composto una newsletter
Flusso principale:	L'utente,tramite il controllo associato, seleziona la gestione degli allegati
Postcondizioni:	Collegamento degli allegati alla newsletter

VISUALIZZA O AGGIORNA DETTAGLIO NEWSLETTER

Descrizione:	Consente all'utente di visualizzare o aggiornare informazioni sulle newsletter
Attori:	Utente terminalista
Flusso principale:	l'utente,tramite il controllo associato, seleziona la gestione delle newsletter
Postcondizioni:	Visualizzazione dati delle newsletter presenti. In caso di modifica avverrà il salvataggio delle variazioni e il successivo invio dell'e-mail

Capitolo 3

Progettazione della base di dati

In questo capitolo verranno illustrati gli aspetti dello sviluppo del sistema informativo che riguardano da vicino il progetto della base dati, rimandando al capitolo successivo l'attività di progettazione rivolta all'implementazione dell'architettura software; progettare una base dati significa definirne struttura, caratteristiche e contenuto. Per realizzare un prodotto di qualità si è adottata una metodologia di riferimento articolata in tre fasi: la progettazione concettuale, la progettazione logica e la progettazione fisica. Ogni fase verrà presentata in dettaglio nelle sezioni che compongono il capitolo.

3.1 Progettazione concettuale

Permette la rappresentazione dei dati della realtà di interesse in termini di un modello formale ad alto livello, indipendente dai criteri di rappresentazione utilizzati nei sistemi di gestione di basi di dati. In questa fase si è cercato di rappresentare il contenuto informativo della base di dati, senza preoccuparci dell'efficienza del programma che userà queste informazioni. Il prodotto di questa fase viene chiamato schema concettuale (schema Entità-Relazione) e fa riferimento ad uno schema concettuale dei dati.

3.1.1 Possibili strategie di progetto

Lo sviluppo di uno schema concettuale, a partire dall'analisi dei requisiti effettuata precedentemente, è stato affrontato seguendo un vero e proprio processo

di ingegnerizzazione composto da una serie di strategie di progetto, utilizzate successivamente anche in fase di progettazione software.

Esaminiamo brevemente le strategie di progetto candidate alla modellazione della base dati, mostrando successivamente la soluzione adottata. Una descrizione più dettagliata si può trovare in [2].

Strategia Top-Down: In questa strategia, lo schema concettuale viene prodotto mediante una serie di raffinamenti successivi a partire da uno schema iniziale che descrive tutte le specifiche con pochi concetti molto astratti. Lo schema viene poi raffinato in maniera iterativa fino a raggiungere il giusto livello di dettaglio dei vari concetti. Il vantaggio della strategia top-down è che il progettista descrivendo da subito tutte le specifiche dei dati, possiede sin dall'inizio di una visione globale dello schema finale. L'applicazione di questa strategia è possibile solo avendo la conoscenza di tutte le componenti del sistema, ciò è estremamente difficile in situazioni reali di una certa complessità.

Strategia Bottom Up: In questa strategia, le specifiche iniziali vengono suddivise fino a quando descrivono un frammento elementare della realtà di interesse. Le componenti ottenute vengono rappresentate da semplici schemi concettuali che, successivamente vengono fusi integrando le varie componenti fino al raggiungimento della schema finale. Rispetto alla strategia top-down, i vari concetti presenti nello schema finale vengono introdotti nelle varie fasi. Il vantaggio della strategia bottom-up sta nel decomporre il problema iniziale in più sottoproblemi, facilmente risolvibili. Presenta come svantaggio, la difficoltà che il progettista incontra nell'operazione di integrazione dei vari schemi concettuali.

Strategia Inside-Out: Questa strategia rappresenta un caso particolare della strategia bottom-up. Consiste nell'individuare inizialmente solo alcuni concetti importanti, procedendo a partire da questi a macchia d'olio esplorando i concetti più lontani da quelli di partenza. Questa strategia ha il vantaggio di non richiedere passi di integrazione. D'altro canto è necessario, di volta in volta, esaminare tutte le specifiche per individuare i concetti non ancora rappresentati, descrivendoli in dettaglio.

Strategia mista: La strategia mista cerca di combinare i vantaggi della strategia top-down con quelli della strategia bottom-up. Il progettista suddivide i requisiti in componenti separate, come nella strategia bottom-up, ma allo stesso tempo definisce uno schema scheletro contenente, a livello astratto, i concetti principali dell'applicazione. Tale schema fornisce una visione unitaria, sebbene astratta, dell'intero progetto e favorisce le fasi di integrazione degli schemi sviluppati separatamente.

3.1.2 Strategia adottata

La metodologia di progettazione adottata è di tipo misto. Si è riusciti infatti a individuare i componenti elementari e contemporaneamente a creare uno schema scheletro contenente concetti di base da espandere con raffinamenti successivi.

E' stato quindi definito un primo schema di base contenente il concetto principale, l'entità Contatti, e le relazioni che lo legano agli altri componenti elementari individuati.

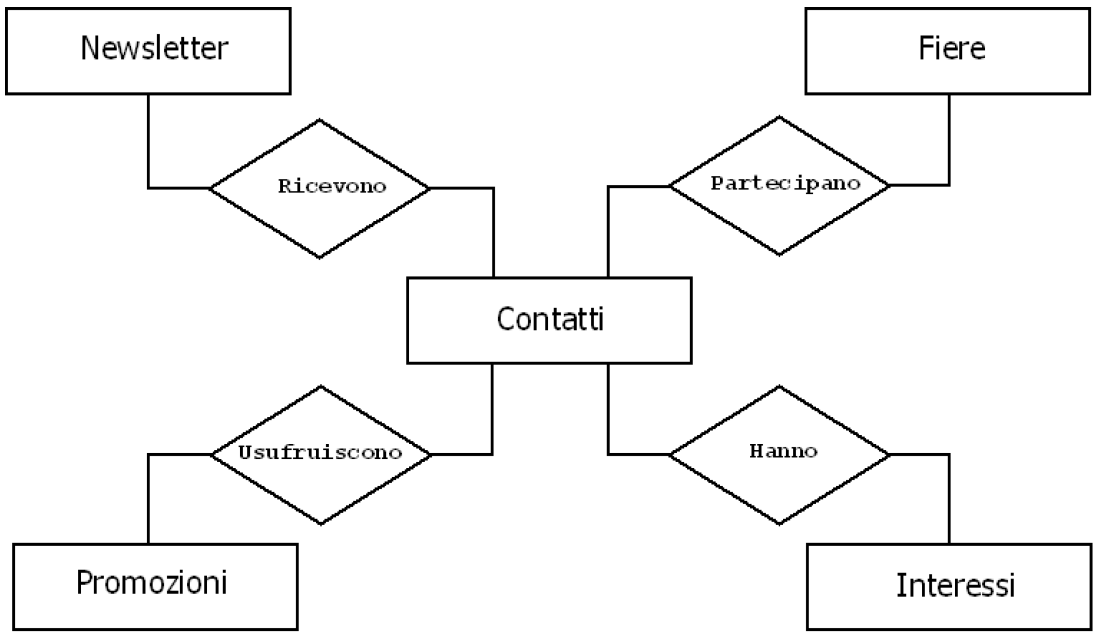

Figura 3.1: Schema scheletro

Un primo raffinamento o espansione dello schema si può effettuare osservando che dalle specifiche emerge un nuovo concetto che è in stretta relazione con l'en-

tità Promozioni, il Gadget. A prima vista, questo poteva essere pensato con un attributo proprio dell'entità Promozioni ma, analizzando il problema si è compreso che al concetto di gadget sono associabili diverse proprietà di interesse e che la sua esistenza è indipendente dal concetto di promozione
La parte di schema interessata dall'espansione viene mostrata nella figura sottostante.

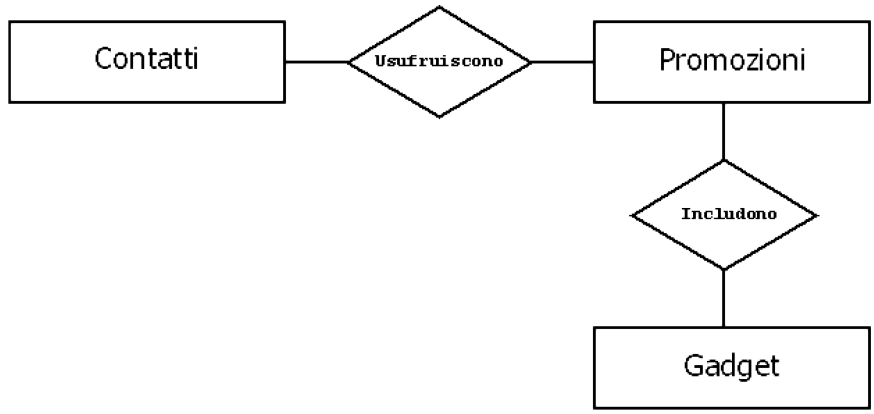

Figura 3.2: Introduzione nuova entità gadget

Un'ulteriore raffinamento si effettua notando che accanto al concetto di newsletter si può affiancare la nozione di allegati, entità distinte ma in relazione tra di loro in quanto ogni newsletter comprende un allegato specifico. Nella figura 3.3 viene mostrata la parte di schema interessata dall'espansione.

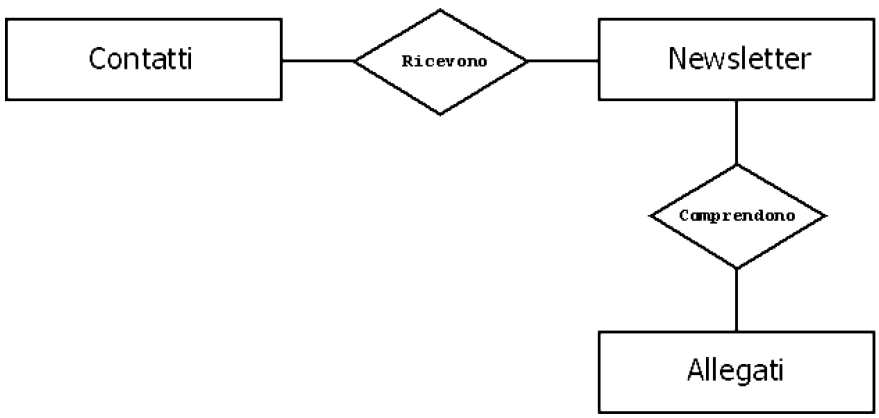

Figura 3.3: Inserimento nuova entità allegati

Osservando ulteriormente le specifiche è stato possibile raffinare ulteriormente lo schema analizzando l'entità Famiglie interessi. All'interno di queste è stata individuata una possibile distinzione in interessi generici. Questa particolarizzazione si rende utile perché permette di conoscere in modo più particolareggiato le esigenze specifiche di ogni contatto.

E' stata introdotta quindi nella porzione di schema interessato la generalizzazione parziale ed esclusiva mostrata in figura.

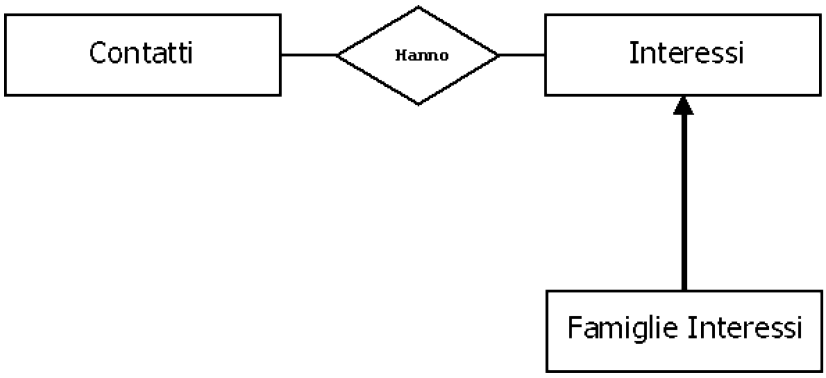

Figura 3.4: Raffinamento entità interessi

Analizzando ancora le specifiche si può sviluppare ulteriormente lo schema considerando le relazioni che intercorrono tra i contatti, le promozioni ed i gadget. Dai requisiti infatti è emerso che non solo si vuole sapere le promozioni alle quali un contatto ha aderito, con rispettivo gadget associato, ma anche, indipendentemente dalle promozioni, i gadget ricevuti. Ecco quindi che è stato opportuno inserire nella parte di schema interessata, una relazione che lega il contatto con il gadget.

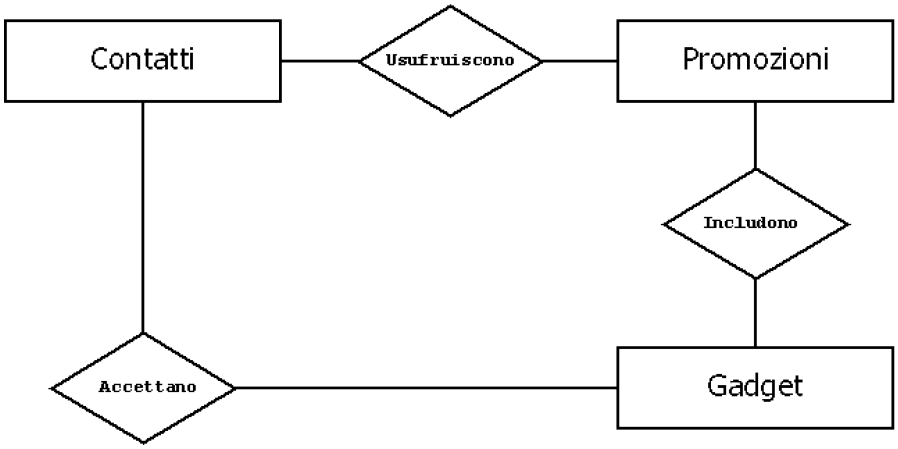

Figura 3.5: Inserimento nuova associazione

L'ultimo passo da effettuare per completare lo schema è quello di introduzione delle proprietà specifiche e identificatori di entità sotto forma di attributi. Per motivi di impaginazione nello schema finale questi non verranno riportati, ma verranno elencati nelle successive tabelle di descrizione degli elementi componenti lo schema.

Dopo aver introdotto tutte le entità con le loro relative associazioni, in questa fase verranno discusse le cardinalità delle relazioni sulla base delle informazioni ottenute dall'analisi dei requisiti.

Successivamente verranno descritte in maggiore dettaglio le entità comprensive dei loro attributi e identificatori, eventuali vincoli di integrità presenti e alla fine della sarà presentato lo schema finale dell'applicazione.

Precedendo con l'analisi delle cardinalità delle relazioni, verrà ora presa in esame la relazione che lega Contatti con Fiere in riferimento alla figura 3.1.

Si può facilmente intuire che la cardinalità della relazione è (0,n) da entrambi i lati, infatti un contatto può partecipare a nessuna oppure a più fiere e contemporaneamente ad una fiera può non partecipare nessuno oppure più contatti.

Lo stesso ragionamento si può applicare alla relazione che lega contatti con Promozioni (figura 3.1). Anche qui la cardinalità sarà (0,n) da entrambi i lati perché una promozione può essere usufruita da nessuno oppure da più contatti e viceversa un contatto può usufruire di nessuna o più promozioni.

Osservando la figura 3.5 vengono ora analizzate le cardinalità delle relazioni tra Promozioni e Gadget e tra Contatti e Gadget.

Si può dire che per ogni promozione deve essere associato uno e un solo gadget, mentre invece un gadget può o meno essere associato ad una promozione. Quindi le cardinalità saranno (1,1) dal lato Promozioni e (0,1) dal lato Gadget.
Soffermandosi invece sulla relazione che lega Contatti con Gadget si può osservare che un contatto può aver ricevuto uno o più gadget oppure nessuno e contemporaneamente, un gadget può essere stato dato a più contatti oppure non essere stato assegnato. Le cardinalità saranno quindi (0,n) da entrambi i lati.

Proseguendo l'analisi viene presa ora in considerazione la relazione che intercorre tra Contatti e Famiglie Interessi, mostrata in figura 3.1. Dal lato dei contatti la cardinalità dovrà essere necessariamente (1,n), perché un contatto deve specificare almeno un interesse per poter essere informato sempre su promozioni e novità di mercato, mentre invece dal lato interessi la cardinalità sarà (0,n) in quanto un interesse può essere specificato da nessuno o più contatti.

L'ultima relazione da esaminare è quella che sussiste tra Newsletter e Allegati. Essendo l 'esistenza dell'allegato legata all'esistenza della newsletter, questo dovrà comparire nell'associazione con partecipazione totale, quindi (1,1). Anche dal lato Newsletter la cardinalità sarà (1,1) in quanto ogni newsletter comprenderà un solo allegato.
Nella figura sottostante si riporta lo schema concettuale finale ottenuto.

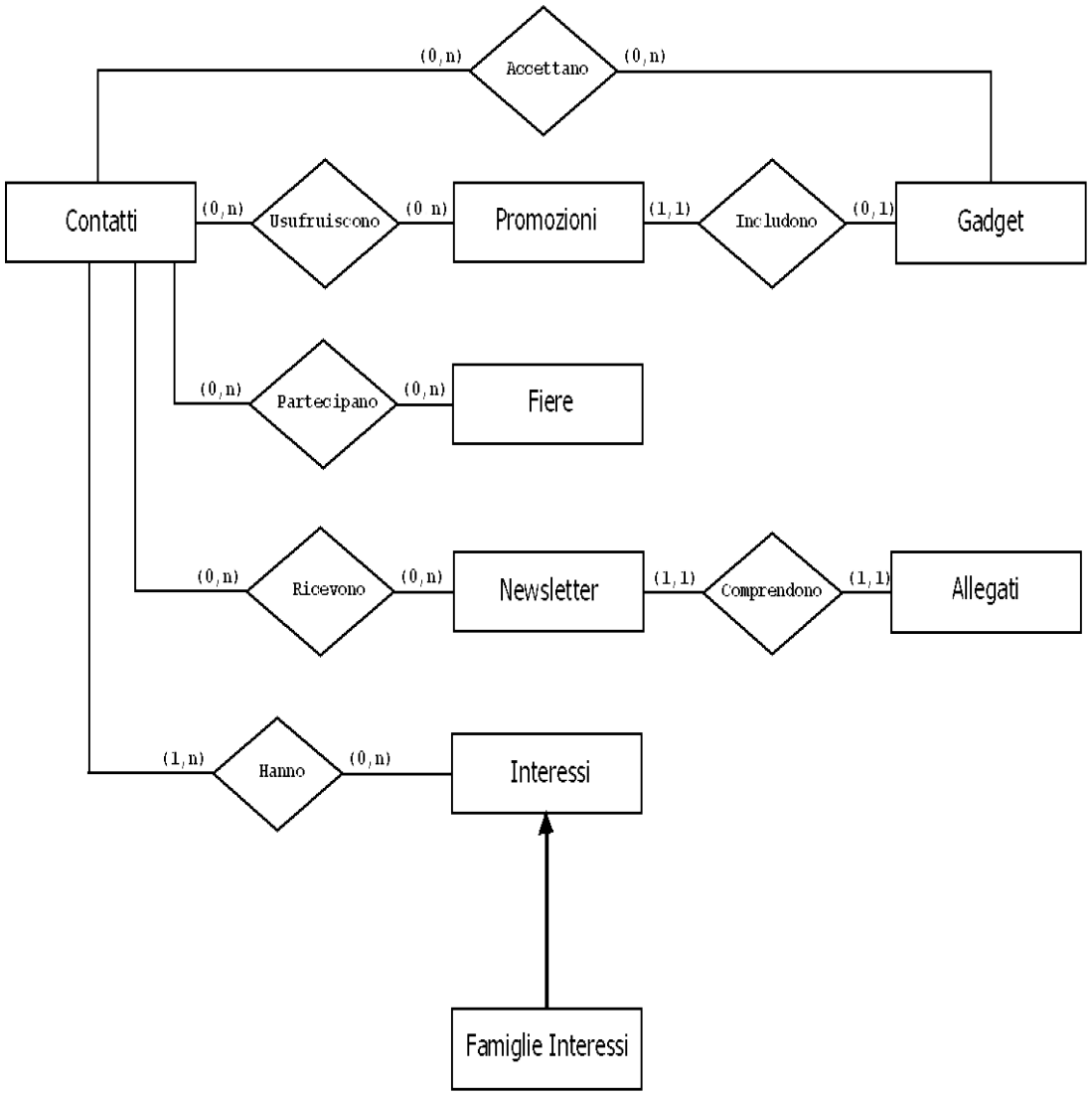

Figura 3.6: Schema E-R

DESCRIZIONE DI ENTITA' E ASSOCIAZIONI

CONTATTI

Contiene le informazioni relative ai contatti dell'azienda :

IdContatto:	Identificatore del singolo contatto
Cognome:	Cognome del contatto
Nome:	Nome del contatto
Indirizzo:	Indirizzo di residenza
Cap:	CAP di residenza
Città:	Città di residenza
Provincia:	Provincia di residenza
Nazione:	Nazione di residenza
Tel:	Numero di telefono fisso del contatto
Fax:	Eventuale numero di fax del contatto
Cell:	Numero di cellulare del contatto
Email:	Indirizzo E-mail principale
Email2:	Indirizzo E-mail alternativo
SitoWeb:	Eventuale sito internet di riferimento
RagSoc:	Ragione sociale della eventuale ditta di appartenenza
PartIva:	Partita iva della società o propria del contatto se lavoratore non dipendente
CodFisc:	Codice fiscale del contatto
Qualifica:	Valore professionale dell mansioni svolte dal contatto
Settore:	Settore di interesse dell'attività del contatto
Attività:	Principale occupazione lavorativa del contatto
FlNews:	Variabile che può assumere due valori, s o n, a seconda che il contatto abbia dato o meno l'abilitazione alla ricezione di newsletter
FlPromoz:	Variabile che può assumere due valori, s o n, a seconda che il contatto sia abilitato o meno a ricevere promozioni
Nota:	Eventuale nota di carattere generale relativa al contatto
Nota2:	Ulteriore spazio riservato alle note

PROMOZIONI

Contiene le informazioni sulle promozioni :

IdPromozione:	Identificatore della promozione
DsPromozione:	Descrizione dell'oggetto della promozione
DtInizio:	Data di inizio della promozione
DtFine:	Data di fine della promozione
LnkRegolamento:	Link al file contenente il regolamento della promozione
LnkVolantino:	Link al file contenente il volantino della promozione
Nota:	Eventuali note sulla promozione

FIERE

Contiene informazioni sulle fiere di settore :

IdFiera:	Identificatore della fiera
DsFiera:	Descrizione generale della fiera
DtInizio:	Data di inizio della fiera
DtFine:	Data di fine della fiera
Città:	Luogo di svolgimento della fiera
Nazione:	Nazione ospitante la fiera
Nota:	Eventuali note sulla fiera

GADGET

Contiene informazioni sugli oggetti omaggio :

IdGadget:	Identificatore del singolo gadget
DsGadget:	Descrizione generale delle caratteristiche dell'oggetto
Causale:	Indica il motivo per cui il gadget può essere donato, ad esempio associato ad una promozione o dato come regalo in occasioni particolari
Nota:	Eventuale nota generale sul gadget

NEWSLETTER

Contiene informazioni sulle newsletter :

IdNewsletter:	Identificatore della newsletter
DsNewsletter:	Descrizione generale dell'oggetto della newsletter
DtNewsletter:	Data di compilazione della newsletter
Testo:	Testo dell'e-mail inviata
Nota:	Eventuale nota generale sulla newsletter

ALLEGATI

Contiene le informazioni sugli allegati delle newsletter spedite :

LnkAllegato:	Link del file inviato come allegato alla newsletter, è anche l'identificatore dell'entità

FAMIGLIE INTERESSI

Entità usata per mantenere le informazioni sulle famiglie di interessi dell'azienda :

IdFamiglia:	Identificatore della famiglia specifica
DsFamiglia:	Descrizione della famiglia considerata

Si ricorda che con famiglie di interessi si intende descrivere l'interesse che un contatto può avere verso uno degli ambiti produttivi dell'azienda.

INTERESSI

Entità usata per mantenere le informazioni sugli interessi specifici presenti in ogni famiglia :

IdInteresse:	Identificatore dell'interesse specifico
DsInteresse:	Descrizione dell'interesse specifico

Per quanto riguarda le associazioni verrà elencata un breve descrizione, le entità coinvolte e le loro cardinalità.

USUFRUISCONO

Descrizione:	Associazione che permette di conoscere quali promozioni sono state usufruite da quali contatti
Entità coinvolte:	Contatti, Promozioni
Cardinalità:	(0,n) lato Contatti, (0,n) lato Promozioni

INCLUDONO

Descrizione:	Associazione che consente di legare un gadget alla relativa promozione
Entità coinvolte:	Promozioni, Gadget
Cardinalità:	(1,1) lato Promozioni, (0,1) lato Gadget

PARTECIPANO

Descrizione:	Associazione che permette di conoscere quali contatti hanno partecipato a determinate fiere di settore
Entità coinvolte:	Contatti, Fiere
Cardinalità:	(0,n) lato Contatti, (0,n) lato Fiere

HANNO

Descrizione:	Associazione che permette di conoscere gli interessi specifiche per ogni contatto
Entità coinvolte:	Contatti, Interessi
Cardinalità:	(1,n) lato Contatti, (0,n) lato Interessi

ACCETTANO

Descrizione:	Associazione che permette di sapere i gadget in possesso dai contatti indipendentemente dalle promozioni usufruite
Entità coinvolte:	Contatti, Gadget
Cardinalità:	(0,n) lato Contatti, (0,n) lato Gadget

RICEVONO

Descrizione:	Associazione che permette di conoscere lo storico di newsletter ricevute dai contatti
Entità coinvolte:	Contatti, Newsletter
Cardinalità:	(0,n) lato Contatti, (0,n) lato Newsletter

COMPRENDONO

Descrizione:	Associazione identificante che permette di associare ogni allegato con la newsletter specifica
Entità coinvolte:	Newsletter, Allegati
Cardinalità:	(1,1) lato Newsletter, (1,1) lato Allegati

Si vuole chiudere questa sezione con una nota sulla scelta dei nominativi per entità e associazioni.

La scelta dei nomi per i tipi di entità, gli attributi e le associazioni, non è sempre semplice. Si potrebbero scegliere nomi che trasmettano il più possibile i significati propri dei differenti costrutti dello schema, e la norma generale, come si evince da [2], consiglierebbe di utilizzare nomi al singolare per identificare le entità appunto perché il nome di un tipo di entità riguarda ogni singola entità che

appartiene ad esso. Contrariamente qui si è scelto di usare anche nomi al plurale per sottolineare la pluralità del tipo di entità. I nomi delle associazioni sono invece stati scelti in modo da rendere il diagramma E-R dello schema leggibile da sinistra verso destra e dall'alto verso il basso.

3.1.3 Requisiti di qualità

L'analisi di qualità, come si può vedere in [2], costituisce un importante momento di verifica dello stato corrente del progetto nel quale è spesso necessario dover effettuare delle ristrutturazioni. L'applicazione continuativa della metodologia di progettazione adottata ha permesso di arrivare ad un modello Entità-Relazione che rispetti requisiti di qualità quali :

Correttezza: Lo schema è risultato essere sintatticamente e semanticamente corretto.

Completezza: Lo schema concettuale rappresenta tutti i dati di interesse e tutte le operazioni possono essere eseguite a partire dai concetti descritti.

Leggibilità: Lo schema concettuale è leggibile, auto esplicativo e rappresenta i requisiti in maniera naturale e facilmente comprensibile.

Minimalità: Tutte le specifiche sui dati sono rappresentate una sola volta nello schema, con una sola eccezione nel ciclo che lega Contatti, Promozioni e Gadget. L'associazione tra CONTATTI e GADGET non sarebbe necessaria se non si volessero considerare anche gadget non assegnati attraverso promozioni, ma dai requisiti si deve considerare anche questa possibilità, quindi l'associazione è stata mantenuta.

3.2 Progettazione logica

L'obiettivo della progettazione logica, da [2], è quello di costruire uno schema logico in grado di descrivere, in maniera corretta ed efficiente, tutte le informazioni contenute nello schema Entità-Relazione prodotto nella fase di progettazione concettuale. La progettazione logica, costituendo la base per l'effettiva realizzazione dell'applicazione, necessita di due fasi ben distinte: una prima attività di ristrutturazione dello schema E-R a cui segue un'attività di traduzione dal modello concettuale a quello logico.

Facendo riferimento alle considerazioni fatte riguardo il carico applicati dell'applicazione, si procederà ora alla fase di ristrutturazione dello schema E-R ottenuto.

3.2.1 Ristrutturazione schema E-R

Il primo passo da seguire per la ristrutturazione è quello di analizzare le eventuali ridondanze presenti nello schema. Una ridondanza consiste, in uno schema concettuale, nella presenza di un dato derivato (cioè ottenuto da una serie di operazioni) da parte di altri.

Analizzando lo schema si osserva la presenza di un ciclo nelle associazioni che riguardano le entità Contatti, Promozioni e Gadget (figura 3.5), conseguentemente le informazioni relative ai gadget assegnati ai contatti si potrebbero derivare dai dati sulle promozioni usufruite dagli stessi. Analizzando le specifiche però viene richiesto la necessità di conoscere i gadget consegnati anche indipendentemente dalle promozioni, quindi la ridondanza viene ritenuta necessaria e non verrà rimossa.
Il passo successivo consiste nell'eliminazione delle generalizzazioni presenti.

L'unica gerarchia presente è quella che riguarda le entità FAMIGLIE INTERESSI e INTERESSI (figura 3.4). Avendo la generalizzazione una sola entità figlia, in questo caso si parla di sottoinsieme e quindi si ritiene opportuno distinguere definitivamente le due entità in quanto gli accessi all'entità figlia saranno diversi dagli accessi all'entità padre. In questo caso si può vedere come nella fase di progettazione concettuale si fosse adottata una trasformazione non idonea alla rappresentazione della realtà di interesse.

Lo schema di figura 3.4 con questa distinzione si trasforma come riportato sotto.

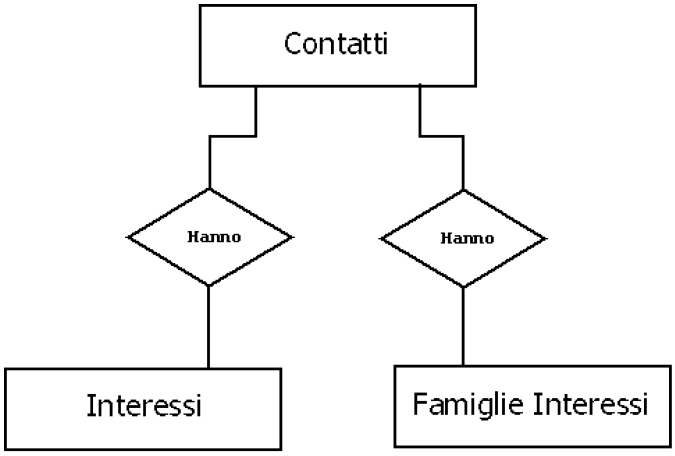

Figura 3.7: Ristrutturazione gerarchia

Quella mostrata è la prima e ultima ristrutturazione necessaria in quanto nello schema non sono presenti né attributi multivalore né attributi composti ed entità deboli. Lo schema ristrutturato viene riportato nella figura seguente.

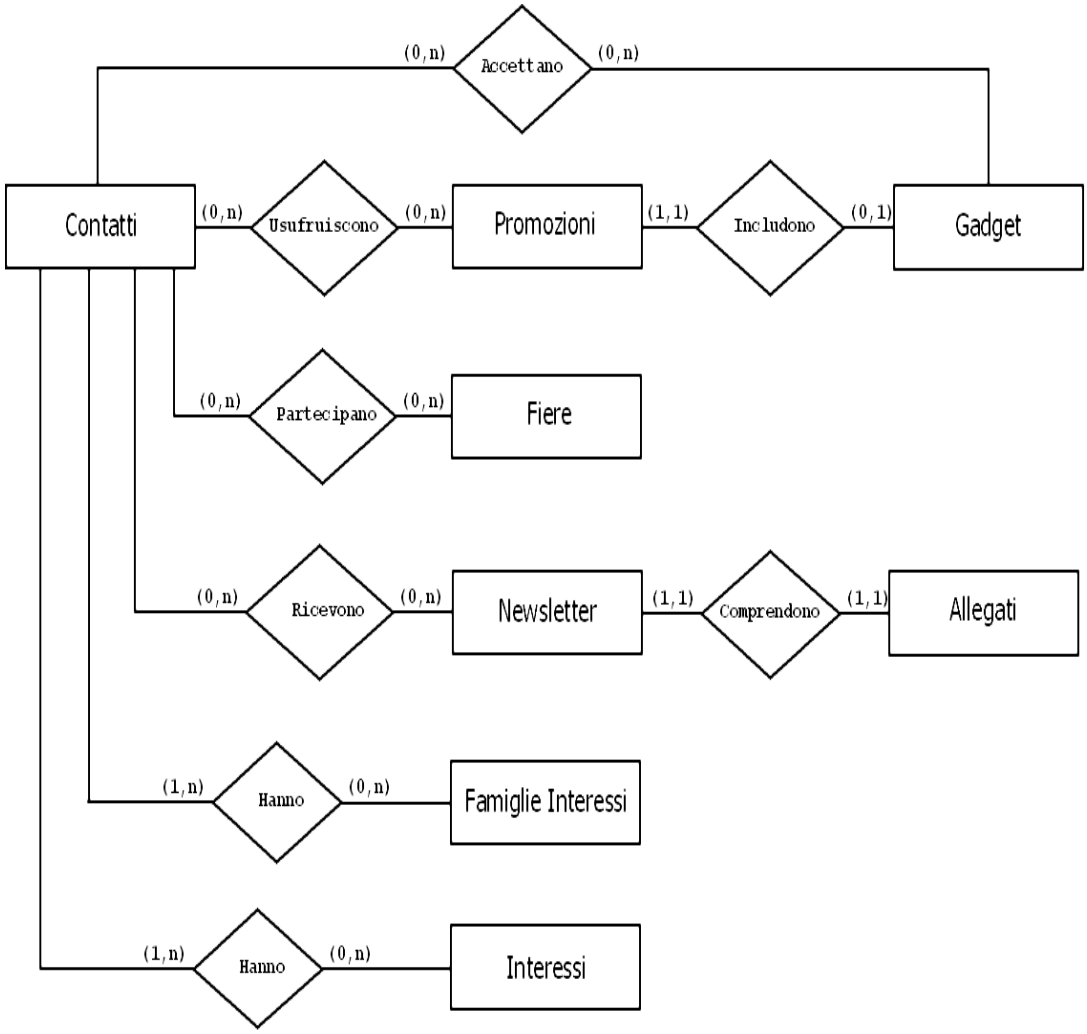

Figura 3.8: Schema E-R ristrutturato

L'ultimo passo da affrontare prima di iniziare la traduzione verso il modello logico è quello della scelta degli identificatori propri di ogni entità.
Nel nostro caso le entità presenti nello schema E-R sono dotate di un solo identificatore, scelto già in fase di progettazione concettuale in modo da soddisfare il vincolo di univocità e l'assenza di valori nulli. Pertanto, sotto questo profilo, lo schema non necessita di alcuna modifica.

3.2.2 Traduzione verso il modello relazionale

Completata la ristrutturazione dello schema concettuale, è ora possibile effettuarne la traduzione verso lo schema logico relazionale equivalente.
La traduzione avviene in modo quasi automatico semplicemente seguendo queste due semplici regole :

- Le entità diventano relazioni sui loro stessi attributi
- Le associazioni diventano relazioni sugli identificatori delle entità coinvolte, più eventuali attributi propri

Per quanto riguarda la traduzione delle associazioni è necessario inoltre prendere in considerazione :

- Il grado dell'associazione (binaria o ennaria)
- La cardinalità della relazione (1:1, 1:N, N:M)
- La partecipazione delle entità (totale, parziale)

Tenendo sempre in considerazione queste regole, successivamente verranno analizzate e tradotte una ad una le associazioni presenti nello schema.
Nei frammenti di schema concettuale presentati verranno indicati, per motivi di impaginazione, solamente gli identificatori delle entità e non l'insieme completo degli attributi.

Figura 3.9: Associazione Usufruiscono

La relazione è di tipo N:M, cioè molti-a-molti e non dipende dal tipo di partecipazione totale o parziale e si risolve con la creazione di un nuovo schema relazionale avente come chiave primaria la combinazione degli identificatori delle due entità.
Schema logico ottenuto :

CONTATTI(IdContatto, Cognome, Nome, Indirizzo, Cap, Città, Prov, Nazione, Tel, Fax, Cell, Email, Email2, SitoWeb, RagSoc, CodFisc, Qualif, Sett, Attiv, FlNews, FlPromoz, Nota, Nota2)

PROMOZIONI(IdPromoz, DsPromoz, DtInizio, DtFine, LnkRegolam, LnkVolantino, Nota)

CONT_PROMOZ(IdContatto, IdPromoz)

In questi schemi sono stati variati i nomi degli attributi Provincia, Qualifica, Settore, Attività in riferimento allo schema CONTATTI, mentre per lo schema PROMOZIONI sono stati modificati gli attributi IdPromozione, DsPromozione e LnkRegolamento.Questo per evitare di avere nomi di campi troppo lunghi nella definizione delle tabelle del database. Il nome della relazione aggiuntiva introdotta, invece, è stato modificato in CONT_PROMOZ per specificare meglio lo stretto legame che intercorre tra i due concetti.

Figura 3.10: Associazione Includono

La relazione è di tipo 1:1 con partecipazione parziale dell'entità GADGET. La traduzione dello schema in un'unica relazione in quanto si introdurrebbero valori nulli, quindi si ricorre all'uso di chiavi esterne. In particolare verrà introdotta la chiave esterna CdGadget nell'entità con partecipazione totale PROMOZIONI.

Schema logico ottenuto :

PROMOZIONI(IdPromoz, DsPromoz, DtInizio, DtFine, CdGadget LnkRegolam, LnkVolantino, Nota)

GADGET(IdGadget, DsGadget, Causale, DtGadget, Nota)

In questo caso viene introdotto nello schema PROMOZIONI la chiave primaria di GADGET (modificata in CdGadget) che funge da chiave esterna referente. Esiste quindi un vincolo di integrità referenziale tra l'attributo CdGadget della relazione PROMOZIONI e la chiave IdGadget della relazione GADGET.

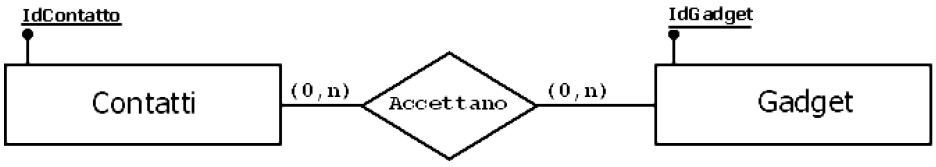

Figura 3.11: Associazione Accettano

L'associazione è di tipo N:M con partecipazione parziale di entrambe le entità. Anche in questo caso quindi si adotta la strategia usata per la trasformazione della relazione tra CONTATTI e PROMOZIONI.

Schema logico ottenuto:

CONTATTI(IdContatto, Cognome, Nome, Indirizzo, Cap, Città, Prov, Nazione, Tel, Fax, Cell, Email, Email2, SitoWeb, RagSoc, CodFisc, Qualif, Sett, Attiv, FlNews, FlPromoz, Nota, Nota2)

GADGET(IdGadget, DsGadget, Causale, DtGadget, Nota)

CONT_GADGET(IdContatto, IdGadget)

Anche qui, il nome della relazione aggiuntiva introdotta, è stato modificato in CONT_GADGET per specificare meglio lo stretto legame che intercorre tra i due concetti.

Figura 3.12: Associazione Partecipano

Anche per questa associazione si possono applicare le considerazioni fatte per la precedente.

Schema logico ottenuto:

CONTATTI(IdContatto, Cognome, Nome, Indirizzo, Cap, Città, Prov, Nazione, Tel, Fax, Cell, Email, Email2, SitoWeb, RagSoc, CodFisc, Qualif, Sett, Attiv, FlNews, FlPromoz, Nota, Nota2)

FIERE(IdFiera, DsFiera, DtInizio, DtFine, Città, Nazione, Nota)

CONT_FIERE(IdContatto, IdFiera)

Il nome della relazione aggiuntiva introdotta è stato modificato in CONT_FIERE, per specificare meglio lo stretto legame che intercorre tra i due concetti.

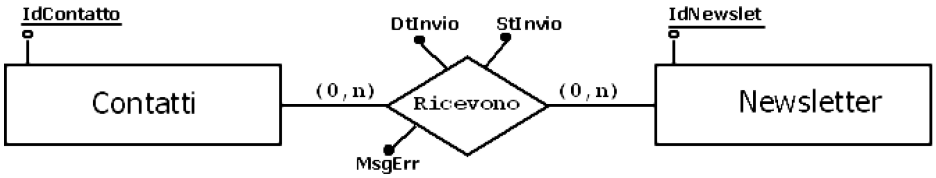

Figura 3.13: Associazione Ricevono

Analogamente per questa associazione si ottengono gli schemi :

CONTATTI(IdContatto, Cognome, Nome, Indirizzo, Cap, Città, Prov, Nazione, Tel, Fax, Cell, Email, Email2, SitoWeb, RagSoc, CodFisc, Qualif, Sett, Attiv, FlNews, FlPromoz, Nota, Nota2)

NEWSLETTER(IdNewslet, DsNewslet, DtNewslet, Testo, Nota)

CONT_NEWS(IdContatto, IdNewslet, DtInvio, StInvio, MsgErr)

Gli attributi dell'associazione sono stati inseriti nella nuova relazione creata. Sono stati modificati i nomi degli attributi IdNewsletter, DsNewsletter e DtNewsletter rispetto a quelli introdotti nello schema concettuale, ed il nome della nuova relazione introdotta è stato cambiato in CONT_NEWS.

Figura 3.14: Associazione Comprendono

L'associazione è di tipo 1:1 con partecipazione totale di entrambe le entità. In questo caso si potrebbe decidere di fondere le due entità in un'unica relazione, ma la soluzione pur tecnicamente corretta sarebbe metodologicamente errata. Infatti

già a livello concettuale i due concetti sono stati separati e tale separazione deve persistere anche a livello logico.

Si ricorre quindi all'uso di chiavi esterne, in particolare verrà introdotto nella relazione ALLEGATI l'attributo IdNewsletter, con funzione di chiave esterna referente.

Schema logico ottenuto :

NEWSLETTER(IdNewslet, DsNewslet, DtNewslet, Testo, Nota)

NEWS_ALLEGATI(LnkAllegato, IdNewsletter)

E' presente un vincolo di integrità referenziale tra l'attributo IdNewsletter della relazione NEWS_ALLEGATI e la chiave IdNewsletter della relazione NEWSLETTER.

Il nome della relazione ALLEGATI è stato modificato in NEWS_ALLEGATI per visualizzare, in maniera più adeguata, la relazione che lega le due entità.

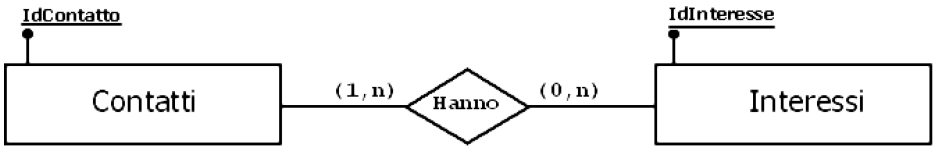

Figura 3.15: Associazione Hanno

L'associazione è di tipo N:M con partecipazione totale dell'entità Contatti e parziale dell'entità Interessi. La traduzione avviene introducendo una nuova relazione avente come chiave primaria la combinazione degli identificatori delle due entità.

Schema logico ottenuto:

CONTATTI(IdContatto, Cognome, Nome, Indirizzo, Cap, Città, Prov, Nazione, Tel, Fax, Cell, Email, Email2, SitoWeb, RagSoc, CodFisc, Qualif, Sett, Attiv, FlNews, FlPromoz, Nota, Nota2)

INTERESSI(IdInteresse, DsInteresse)

CONT_INTERESSI(IdContatto, IdInteresse)

Sul nome della nuova relazione valgono le stesse considerazioni fatte per gli schemi precedenti.
Analoga trattazione ha l'associazione che lega CONTATTI con FAMIGLIE INTERESSI.

Schema logico ottenuto:

CONTATTI(IdContatto, Cognome, Nome, Indirizzo, Cap, Città, Prov, Nazione, Tel, Fax, Cell, Email, Email2, SitoWeb, RagSoc, CodFisc, Qualif, Sett, Attiv, FlNews, FlPromoz, Nota, Nota2)

FAMIGLIE INTERESSI(IdFamiglia, DsFamiglia)

CONT_FAMIGLIE(IdContatto, IdFamiglia)

Il risultato della traduzione dallo schema E-R al modello relazionale è rappresentato mediante il seguente formalismo grafico che premette di rappresentare i legami esistenti tra le varie relazioni, ovvero i vincoli di integrità referenziale.

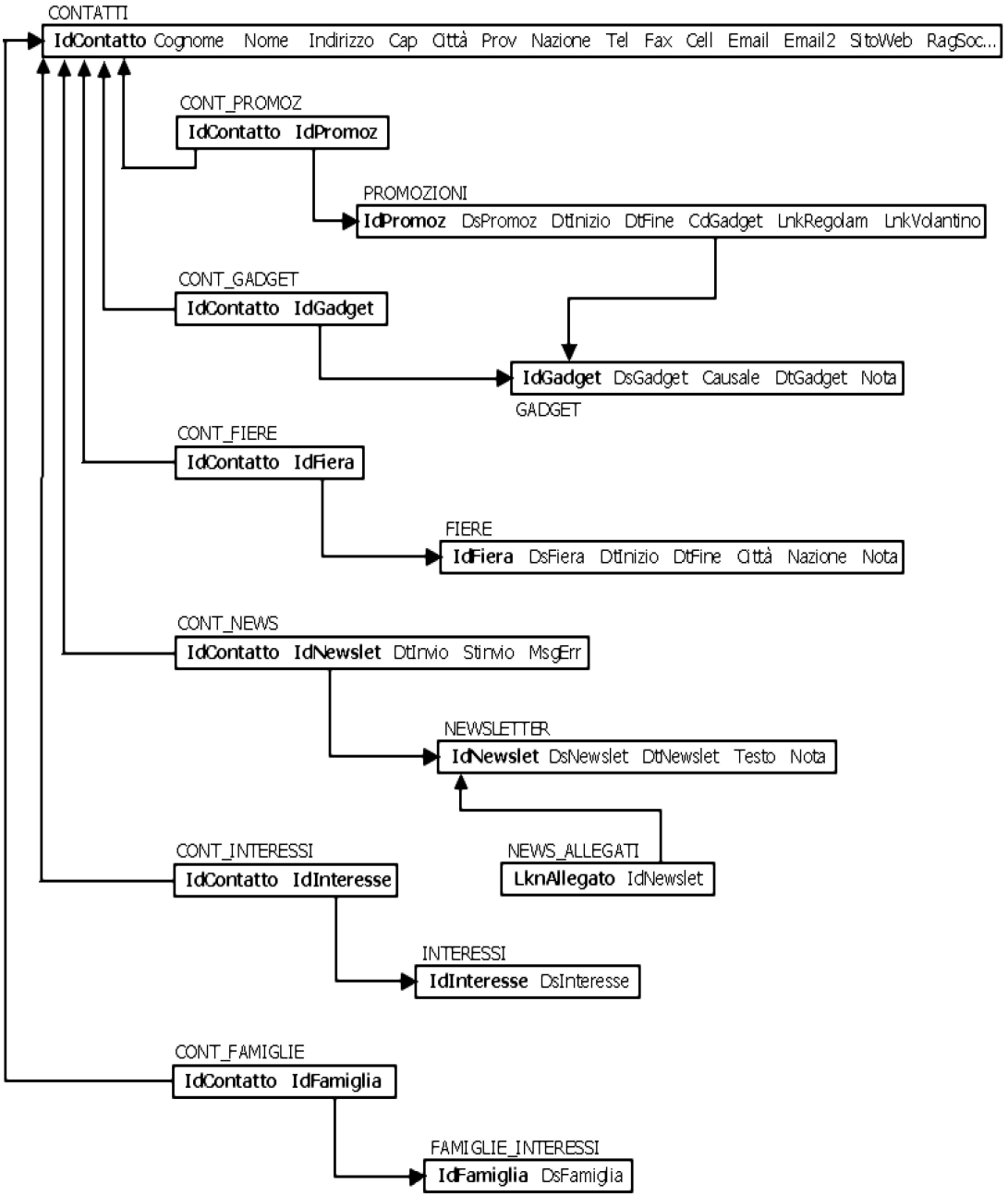

Figura 3.16: Schema logico

3.2.3 Verifica di normalizzazione

La teoria della normalizzazione,come si evince da [2] si occupa dei seguenti problemi su schemi relazionali :

- Definire quando due schemi sono equivalenti
- Definire criteri di bontà per gli schemi (forme normali)
- Determinare metodi algoritmici per ottenere uno schema migliore ed equivalente a partire da uno schema non in forma normale (normalizzazione)

La normalizzazione va usata come tecnica di verifica dei risultati della progettazione di una base di dati, non costituisce quindi una metodologia di progettazione.
La verifica viene effettuata valutando alcuni criteri, chiamati forme normali, che garantiscono la qualità dello schema, cioè l'assenza di determinate anomalie di aggiornamento sulla base dati. Queste anomalie possono verificarsi in presenza di duplicazione non necessaria di dati; sono dovute a schemi di relazione mal progettati, non ottimali, non normalizzati.

Di seguito verrà quindi effettuata una verifica di normalizzazione sugli schemi relazionali ottenuti durante la fase di progettazione logica, valutando per ogni relazione il verificarsi o meno dei requisiti previsti dalle forme normali (prima, seconda e terza forma normale e forma normale di Boyce-Codd).
Gli schemi presentati risultano soddisfare tutti al requisito di prima forma normale(1NF) in quanto tutti gli attributi delle relazioni sono semplici e non multivalore, cioè atomici.
Prima di passare ad analizzare i requisiti per le altre forme normali, vengono date alcune definizioni preliminari come sussidio ad una più chiara comprensione dei criteri adottati (si veda [2] per un'analisi più approfondita).

Dipendenza funzionale

Una dipendenza funzionale, indicata con $X \rightarrow Y$, tra due insiemi di attributi X e Y di una relazione R specifica un vincolo sulle tuple che possono formare uno stato di relazione r valido. Il vincolo è che, per ogni coppia di tuple t_1 e t_2 in r per le quali è $t_1[X] = t_2[X]$, si deve avere anche $t_1[Y] = t_2[Y]$.

Ciò significa che i valori della componente Y di una tupla in r dipendono da, o sono determinati da, i valori della componente X o, in alternativa, che i valori della componente X di una tupla determinano univocamente (o funzionalmente) i valori della componente Y.

Dipendenza funzionale completa
Una dipendenza funzionale $X \rightarrow Y$ si dice completa se la rimozione di qualsiasi attributo A da X comporta che la dipendenza non sussista più, cioè per ogni attributo $A \in X$, $(X - \{A\})$ non determina funzionalmente Y, altrimenti si dice parziale.

Dipendenza funzionale transitiva
Una dipendenza funzionale $X \rightarrow Y$ in uno schema di relazione R si dice transitiva se esiste un insieme di attributi Z, che non è né una chiave candidata né un sottoinsieme di una chiave di R, per cui valgono contemporaneamente $X \rightarrow Z$ e $Z \rightarrow Y$.

Sulla base di queste definizioni è possibile quindi analizzare i requisiti per le altre forme normali.
Il requisito di seconda forma normale (2NF) stabilisce che ogni attributo che non sia chiave primaria per uno schema di relazione R non sia parzialmente dipendente da nessuna chiave dello schema stesso.

Andando ad analizzare gli schemi ottenuti, si può affermare che l'unica relazione che non la soddisfa è la relazione CONT_NEWS. Infatti in questa relazione vi è la presenza delle dipendenze parziali $IdNewslet \rightarrow DtInvio$, $IdNewslet \rightarrow StInvio$, $IdNewslet \rightarrow MsgErr$.
Per gli schemi rimanenti la forma normale risulta verificata in presenza di due casi come segue. Nel primo, come nel caso della relazione CONTATTI, essendo la chiave primaria formata da un solo attributo, non vi possono essere dipendenze parziali da essa. Nel secondo, ad esempio nella relazione CONT_PROOMOZ, in quanto gli unici due attributi presenti sono anche chiave primaria dello schema.
Nel caso della relazione CONT_NEWS non viene considerata una possibile scomposizione sulle dipendenze funzionali individuate in quanto questa non produrrebbe alcun vantaggio ai fini dell'applicazione.

La terza forma normale (3NF) invece stabilisce che ogni attributo non chiave della relazione R debba essere funzionalmente dipendente in modo completo da ogni chiave di R e che inoltre non sia dipendente in modo transitivo da alcuna chiave di R. In termini più pratici per ogni dipendenza funzionale $X \rightarrow Y$ (non banale) risulta che : o X è una superchiave di R o Y è un attributo primo di R.
Analizzando gli schemi si può vedere che ad esempio lo schema CONTATTI non è in 3NF a causa della presenza delle dipendenze funzionali $Citta' \rightarrow Prov$, $Citta' \rightarrow Nazione$ e $Cap \rightarrow Citta'$.
Anche negli schemi di PROMOZIONI e FIERE sono presenti dipendenze funzionali che violano la 3NF, $DtInizio \rightarrow Dtfine$ e $Citta' \rightarrow Nazione$ per la relazione FIERE e $DtInizio \rightarrow DtFine$ per la relazione PROMOZIONI. Pertanto questi schemi non sono in 3NF. Lo schema CONT_NEWS non è in 3NF non essendo soddisfatti i requisiti per la 2NF.
Per questi schemi non si è valutata una loro scomposizione sulle dipendenze funzionali in quanto non produrrebbe ulteriori vantaggi ai fini dell'applicazione ma anzi in taluni casi la decomposizione aumenterebbe il numero di accessi alla base dati per reperire le informazioni necessarie.
Le rimanenti relazioni soddisfano tutte la 3NF.

L'ultima forma normale da verificare, e la più restrittiva, è la forma normale di Boyce-Codd (BCNF). Questa afferma che ogni volta che sussista in uno schema di relazione R una dipendenza funzionale non banale $X \rightarrow Y$, X sia una superchiave di R.
Si può notare che sicuramente le relazioni CONTATTI, FIERE, PROMOZIONI e CONT_NEWS non sono in BCNF non essendo in 3NF. Le rimanenti soddisfano tutte alla BCNF in quanto per ogni dipendenza funzionale $X \rightarrow Y$, X è superchiave della relazione.

Con la verifica della normalizzazione si chiude la fase di progettazione logica della base dati. Il passo successivo è quello di realizzare fisicamente il database attraverso il DBMS scelto traducendo lo schema logico ottenuto in una sequenza di comandi SQL. Si è deciso di non includere nell'indice di questo lavoro di tesi la fase relativa alla progettazione fisica in quanto non aggiungerebbe informazione aggiuntiva utile ai fini del documento stesso.

Capitolo 4

Progettazione e implementazione dell'applicativo

Nel seguente capitolo verrà presentata la fase di progettazione e sviluppo dell'applicativo che dovrà gestire la base dati creata precedentemente. Verrà dapprima definita l'architettura generale del sistema e successivamente la struttura e i componenti del software realizzato.

Prima di procedere con lo sviluppo e implementazione è risultato necessario appartare alcune modifiche ai campi delle tabelle del database per soddisfare ai requisiti richiesti. In particolare i ogni tabella sono stati aggiunti dei campi di controllo, nascosti all'utente finale, utili soprattutto per una politica di sicurezza interna aziendale. Questi campi sono :

- USRCREAZ : utente del sistema che ha creato il record
- DTCREAZ : data di creazione del record
- USRMOD : utente del sistema che ha modificato il record
- DTMOD : data di modifica del record
- INDMOD : indice di modica, campo usato nelle stored procedure per controllare modifiche in accessi concorrenti

4.1 Definizione architettura

Secondo la definizione ANSI-IEEE Std1471-2000 l'architettura è l'organizzazione basilare di un sistema rappresentato dalle sue componenti dalle relazioni che esisyono tra di loro e con l'ambiente circostante, e dai principi che governano la sua progettazione e d evoluzione. Si deve quindi rielaborare il frutto dell'analisi tenendo in considerazione le possibilità offerte dall'orientamento all'oggetto.

Dalla rielaborazione dei requisiti raccolti è emersa la necessità di progettare una ben determinata tipologia di software: una applicazione database inserita in un'architettura client/server; si tratta di applicazioni che si occupano di persistere e recuperare informazioni da basi di dati, gestendo l'interazione concorrente con esse da parte di un elevato numero di utenti e che solitamente si trovano ad interagire con altre applicazioni.

Lo scenario di applicazione è appunto quello di un sistema distribuito formato da un database server e da diversi client che attraverso transazioni richiedono l'esecuzione delle stored procedure mantenute nel data dictionary del DBMS. Elaborato il risultato, il server risponderà al client, il quale attraverso l'interfaccia grafica dell' applicativo renderà disponibili i dati all'utente finale.
In questo scenario il database server svolge le funzioni di data management (la persistenza dei dati), mentre invece al client sono affidate le funzioni del livello di applicazione e presentazione, cioè gestire la logica dell'applicazione e la visualizzazione dei dati all'utente.

4.2 Progettazione e implementazione

Dopo aver analizzato brevemente l'architettura generale del sistema ora si andrà a definire più in dettaglio la struttura ed i componenti del software realizzato.
Dall' analisi dei requisiti e delle funzionalità svolta nella prima fase del progetto si è appreso che l'applicazione avrebbe dovuto consentire all'utente di interagire con il sistema manipolando graficamente degli oggetti,attraverso un'interfaccia grafica, svincolandolo dall'obbligo di imparare una serie di comandi da impartire con la tastiera come invece avviene con le più tradizionali interfacce testuali CLI (command line interface).

Uno degli aspetti principali per la progettazione e implementazione di applicazioni a interfaccia grafica è il paradigma a eventi o paradigma event driven.

4.2.1 Paradigma Event-Driven

La programmazione ad eventi è un paradigma di programmazione dell' informatica. Mentre in un programma tradizionale l'esecuzione delle istruzioni segue percorsi fissi, che si ramificano soltanto in ben determinati punti predefiniti dal programmatore, nei programmi scritti utilizzando la tecnica ad eventi il flusso del programma è largamente determinato dal verificarsi di eventi esterni.

Un evento può essere definito come "un significativo cambiamento nello stato" di un componente dell'applicazione. In questo paradigma ogni componente notifica un cambiamento nel proprio stato o in quello dell'ambiente inviando un messaggio di tipo broadcast o multicast agli altri componenti. Il componente che notifica il cambiamento di stato agisce da "generatore dell'evento" mentre i componenti che ricevono il messaggio fungono da "ascoltatori dell'evento". La ricezione del messaggio causa la reazione immediata del componente ascoltatore che provvederà a processare l'evento attraverso il proprio gestore di eventi o Event Handler.

La caratteristica generale delle applicazioni progettate con questo paradigma è quella di essere puramente "reattive", non si riesce cioè ad identificarne staticamente un flusso di controllo unitario. Il programma principale si limita a inizializzare l'applicazione, istanziando gli osservatori e associandovi gli opportuni handler.

Il paradigma a eventi è uno dei modelli sul quale si basa tuttora la progettazione e implementazione di interfacce utente grafiche. Per chiarezza viene riportato un esempio di gestione di eventi usato nell'applicazione realizzata.
Sottoscrizione dell'evento legato al click sul controllo `btnContatti` al suo gestore:

```
btnContatti.Click += new System.EventHandler(btnContatti_Click)
```

gestore evento associato:

```
private void btnContatti_Click(object sender, System.EventArgs e){
   ContattiVisMod2 frm = new ContattiVisMod2(this);
   frm.ShowDialog();
   this.DesktopLocation = frm.DesktopLocation;
   this.Visible = true; }
```

In questo caso, l'azione che genera l'evento è il click da parte dell'utente sul controllo `btnContatti` e l'evento generato è appunto l'evento `click`. A questo punto il gestore dell'evento `click` associato al controllo viene avvisato e provvederà a "consumare" l'evento intraprendendo delle azioni specifiche. Nel caso in esame viene aperta la finestra di visualizzazione contatti.
Osservando il codice del gestore si può notare che il .net framework utilizzi la convenzione di fornire un primo parametro `sender`, in modo che il gestore possa ricavare anche l'oggetto generatore dell'evento. Il secondo parametro invece conterrà l'evento con gli eventuali dati aggiuntivi che lo caratterizzano.

4.2.2 Accesso ai dati

Praticamente tutte le applicazioni, come anche quella che si andrà a realizzare, hanno la necessità di interrogare o aggiornare dati persistenti memorizzati in file, database relazionali o altri tipi di supporto di memorizzazione. Per ovviare a tale necessità, il NET framework include ADO.NET, un sottosistema per l'accesso ai dati ottimizzato per applicazioni basta su .NET appunto. Grazie alle API (Application Programming Interfaces) messe a disposizione si è riusciti ad implementare un modello di accesso ai dati semplice e allo stesso tempo efficiente e rappresentativo della base dati a cui ci si dovrà interfacciare. Questo avrà il compito di eseguire un mapping fra la rappresentazione tabellare tipica di una base dati e gli oggetti messi a diposizione dal modello stesso.

Per garantire estensibilità, modificabilità e riutilizzo del software il modulo di accesso ai dati è stato implementato in una classe specifica separata dal resto delle altre classi che compongono l'intero pacchetto software.

Vedremo ora come è stato possibile realizzare un accesso ai dati di tipo disconnesso utilizzando gli strumenti messi a disposizione dal framework, come visto in [6].

Il primo passo da eseguire è quello di stabilire una connessione con la sorgente fisica di dati. Questo viene fatto istanziando un nuovo oggetto `OdbcConnection` che rappresenta appunto una connessione aperta a un'origine di dati, e aprendo la connessione verso lo stesso. La stringa di connessione specifica viene ricavata dal file di configurazione dell'applicazione e passata come parametro al metodo.

Una volta connessi alla sorgente di dati è necessario disporre di un contenitore in cui sia possibile archiviare i dati per renderli disponibili all'applicazione.

Questo si ottiene creando un nuovo oggetto `DataSet` che come è stato visto nel capitolo uno, comprende un modello a oggetti gerarchico costituito da tabelle, righe, colonne e relazioni.
All'interno del dataset successivamente viene ricreata la struttura interna del database, instanziando oggetti `DataTable` e `Data Relation`.
Le tabelle vengono popolate ricorrendo all'uso di un `DataAdapter`, il quale funge da ponte tra il dataset e l'origine di dati attraverso il suo metodo `Fill()`.
La seguente porzione di codice esemplifica una tipica fase di popolamento delle tabelle del dataset.

```
Datatable dt = new DataTable(TB_CONTATTI);
DataAdapter daContatti=
      new OdbcDataAdapter ("SELECT * FROM "+TB_CONTATTI, conn);
daContatti.FillSchema(dt , SchemaType.Source);
daContatti.Fill(dt);
DataColumn col = dt.Columns[CL_ID_CONTATTO];
col.AutoIncrement = true;
col.AutoIncrementSeed = -1;
col.AutoIncrementStep = -1;
ds.Tables.Add(dt);
```

Una volta create e popolate le tabelle si è dovuto ricreare le relazioni esistenti tra esse anche nella loro copia locale, contenuta nel dataset. Questo si ottiene mediante l'aggiunta nella proprietà `Relations` del dataset, della relazione esistente tra le tabelle specificate.

```
ds.Relations.Add(RL_CONT_PROMOZ,
ds.Tables[DB.TB_CONTATTI].Columns[DB.CL_ID_CONTATTO],
ds.Tables[DB.TB_CONTATTI_PROMOZIONI].Columns[DB.CL_ID_CONT],true)
```

Per effettuare operazioni permanenti sulla base dati il framework mette a disposizione gli `OdbcCommand` che appunto rappresentano una stored procedure o un comando SQL da eseguire in relazione a un'origine di dati. Di seguito viene mostrato un esempio di un comando per chiamare una stored procedure per l'e-

liminazione di record dalla tabella Contatti.

```
sqlText = "CALL " + SP_CONTATTI_DELETE + " (?, ?)";
cmdDelete = new OdbcCommand(sqlText, cn);
cmdDelete.Parameters.Add(PR_ID_CONTATTO, OdbcType.Int, 0, "");
cmdDelete.Parameters.Add(PR_RETVAL, OdbcType.Int, 0, "");
cmdDelete.Parameters[PR_RETVAL].Direction=ParameterDirection.Output;
daContatti.DeleteCommand = cmdDelete;
```

I vantaggi principali dell'uso di questo modello di dati disconnesso si hanno in termini di prestazioni dell'intero sistema, in quanto si minimizza il tempo di connessione con il database. L'applicazione infatti rimane connessa alla base dati solo il tempo sufficiente a reperire o aggiornare i dati, diminuendo così il carico applicativo del database.

4.2.3 Interfaccia grafica

Come accennato nell'analisi dei requisiti, nello sviluppo di un prodotto sono importanti anche i bisogni degli utenti nell'ambito dell'usabilità e dell'interazione con il sistema e non solo lo stato delle tecnologie.
Per questo nella fase di progettazione dell'applicativo si è cercato di non tralasciare l'aspetto estetico e funzionale che l'applicazione avrebbe dovuto avere.

Nella progettazione dell'interfaccia si è cercato di adottare diverse linee guida che avrebbero permesso di ottenere un dialogo utente sistema abbastanza efficace. Prima di tutto si è cercato di utilizzare un interfaccia uniforme per tutti i possibili task che l'utente ha a disposizione. In secondo luogo l'informazione deve essere presentata all'utente in modo che la sua comprensione dello stato del sistema sia la più adeguata possibile e la modalità di interazione dell'utente con lo stesso deve essere definita in modo da non costringerlo ad azioni inutili o indesiderate. Come ultimo poi ogni operazione possibile deve essere visibile e ben chiara: le proprietà percepite e reali di un oggetto devono aiutare l'utente a determinare come usarlo o interagire con lui.
Tutte queste considerazioni hanno portato alla realizzazione di un'interfaccia grafica come quella mostrata in figura.

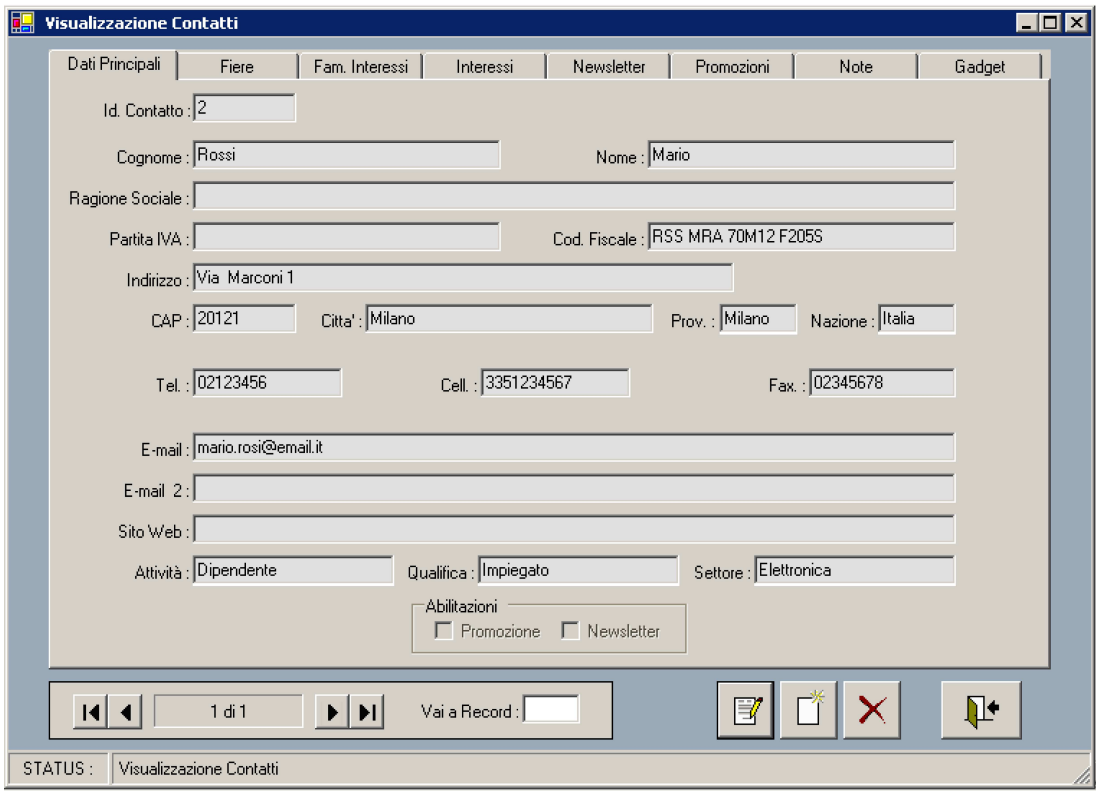

Figura 4.1: Visualizzazione contatti

Come si può vedere ogni operazione possibile all'utente è resa ben visibile e riconoscibile attraverso controlli appropriati che attraverso icone rappresentano chiaramente la funzione svolta dal controllo stesso.
La struttura a pannelli inoltre consente di aumentare la navigabilità all'interno dei dati relativi ad ogni entità. Tutti i dati relativi ad un contatto ed alle relazioni che ha con le altre entità ad esempio, sono disponibili in un'unica interfaccia senza la necessità da parte dell'utente di ricorrere a ulteriori finestre, il che concorre ad aumentare così la soddisfazione dello stesso nell'uso dell'interfaccia.

Il primo problema con il quale si è venuti a contatto è stato quello di come associare i dati presenti nel database alle proprietà dei controlli delle finestre.

Nelle applicazioni che si interfacciano con un database infatti, un problema fondamentale è la gestione del flusso che porta i dati dal database all'applicazione e viceversa. Data Binding, come specificato in [1,4,5] è il termine dato ad un pattern di progettazione che gestisce tutti gli aspetti di questo passaggio da una struttura dati ai controlli di un form e viceversa.

NET prevede due tipi di data binding: data binding semplice e data binding complesso. Il data binding semplice è la capacità di un controllo di interfacciarsi

con un database e permettere la visualizzazione di un singolo elemento dati. Ad esempio, un controllo `TextBox` che pemette di visualizzare il nome di un contatto presente nella tabella Contatti del database.
L'espressione data binding complesso si riferisce alla capacità di un controllo di visualizzare elementi di dati multipli nello stesso momento. I controlli `DataGridView` (per le applicazioni windows form) e `DataGrid` (per le applicazioni web) sono esempi di controlli progettati per la visualizzazione di dati in forma tabulare, usati ad esempio per visualizzare tutte le promozioni associate ad un contatto.
Ad esempio per associare alla Proprietà `Text` di un controllo `TextBox` il contenuto del campo nome nella tabella CONTATTI del dataset si usa la seguente sintassi:

```
txtNome.DataBindings.Add("Text",dvm,DB.TB_CONTATTI+"."+DB.CL_NOME);
```

In questo modo si ha la sincronizzazione automatica del contenuto dei controlli con le rispettive strutture dati.

Durante la fase di progetto dell'interfaccia si è prestata particolare attenzione alla fase di inserimento dati da parte dell'utilizzatore. Una componente fondamentale nei scenari che includono l'interazione con l'utente è la validazione dei dati inseriti, come si evince da [1,5]. Infatti, ogni volta che ci si trova ad accettare informazioni tramite form (o finestre) bisogna preoccuparsi di controllare che questi dati siano completi e conformi alla struttura interna dei dati nel database, altrimenti si incorrerebbe in spiacevoli situazioni di errore e di inconsistenza dei dati.Quindi, per prevenire l'inserimento di valori errati è necessario includere delle sezioni di codice che provvedano a validare il dato inserito nella form prima che questo sia scritto nel database.Nel caso specifico si è pensato di delegare il compito di validazione dei dati ad un particolare evento innescato all'abbandono da parte dell'utente del campo attivo, che provvede a controllare e validare il dato inserito attraverso l'uso di espressioni regolari[8].
Nell'esempio seguente viene riportata la routine evento che valida il dato relativo alla partita iva.

```
private void txtPartIva_Validating(
            object sender, System.ComponentModel.CancelEventArgs e)
{
TextBox txtData = (TextBox) sender;
Regex regPartIva = new Regex(@"\b\d+\b");
            Match m = regPartIva.Match(txtData.Text);
     if (txtData.Text.Length > 11 || !m.Success) {
       e.Cancel = true;
       txtData.Select(0,txtData.Text.Length);
          stsBarPanel2.Text = " formato Partita IVA non corretto,
                                                  modificarlo...";
} else if (stsBarPanel2.Text != "") {
stsBarPanel2.Text = "";
}
}
```

In caso di inserimento di dati errati, l'applicazione invia come risultato all'utente la medesima form con l'indicazione dell'errore. In questo modo si garantisce una delle principali caratteristiche che deve avere un 'interfaccia grafica, la ripristinabilità cioè la capacità di raggiungere l'obiettivo desiderato dopo un errore.

Tutti questi accorgimenti hanno portato allo sviluppo di un'interfaccia che sia comprensibile e di facile utilizzo e allo stesso tempo in grado di rappresentare tutti i requisiti funzionali espressi dall'utente.

4.2.4 Componenti aggiuntivi

Analizzando le funzionalità richieste dall'utente all'applicativo è emerso che questo avrebbe dovuto implementare e supportare ulteriori operazioni : l'invio di newsletter attraverso e-mail, la creazione di un file di log dell'applicazione e la possibilità di importare dati preesistenti in file Excel.

L'utilizzo del file di log è stato ritenuto utile per registrare gli eventi caratteristici dell'applicazione , come caricamento dei dati e invio di newsletter e come strumento valido per monitorare eventuali situazioni di errore.
Nel nostro caso si tratta di un file sequenziale sempre aperto in scrittura che viene aggiornato ogniqualvolta l'applicazione effettua un'operazione significante. Nelle righe seguenti viene mostrata una sezione del file creato.

```
*****************************************************************
*         03/05/2007 17.08.59  ***AVVIO APPLICAZIONE***         *
*                                                               *
            03/05/2007 17.08.59  CARICAMENTO DATI


  03/05/2007 17.09.00  12 Record caricati nella tab. CONTATTI
  03/05/2007 17.09.00  10 Record caricati nella tab. PROMOZIONI
  03/05/2007 17.09.00  5 Record caricati nella tab. CONT_PROMOZ
  03/05/2007 17.09.00  0 Record caricati nella tab. CONT_NEWS
  03/05/2007 17.09.00  0 Record caricati nella tab. NEWSLETTER
  03/05/2007 17.09.00  0 Record caricati nella tab. NEWS_ALLEGATI
  03/05/2007 17.09.00  6 Record caricati nella tab. CONT_INTERESSI
  03/05/2007 17.09.00  0 Record caricati nella tab. CONT_FAMIGLIE
  03/05/2007 17.09.00  2 Record caricati nella tab. FIERE
  03/05/2007 17.09.00  2 Record caricati nella tab. CONT_FIERE
  03/05/2007 17.09.00  0 Record caricati nella tab. GADGET
  03/05/2007 17.09.00  0 Record caricati nella tab. CONT_GADGET

           03/05/2007 17.09.00  DATI CARICATI

  03/05/2007 17.10.40  mail inviata con successo ai seguenti
                       contatti:
                              mario.rossi@email.it

*                                                               *
*         03/05/2007 17.10.44  ***FINE APPLICAZIONE***          *
*                                                               *
*****************************************************************
```

Per quanto riguarda invece l'invio delle newsletter, questa funzionalità è stata introdotta per rendere più completo e indipendente da applicazioni esterne il pacchetto software realizzato. Così facendo la soluzione vede aumentare la sua portabilità e la sua riusabilità a fronte di esigenze future.

Per completezza viene riportata in figura la finestra principale di invio newsletter.

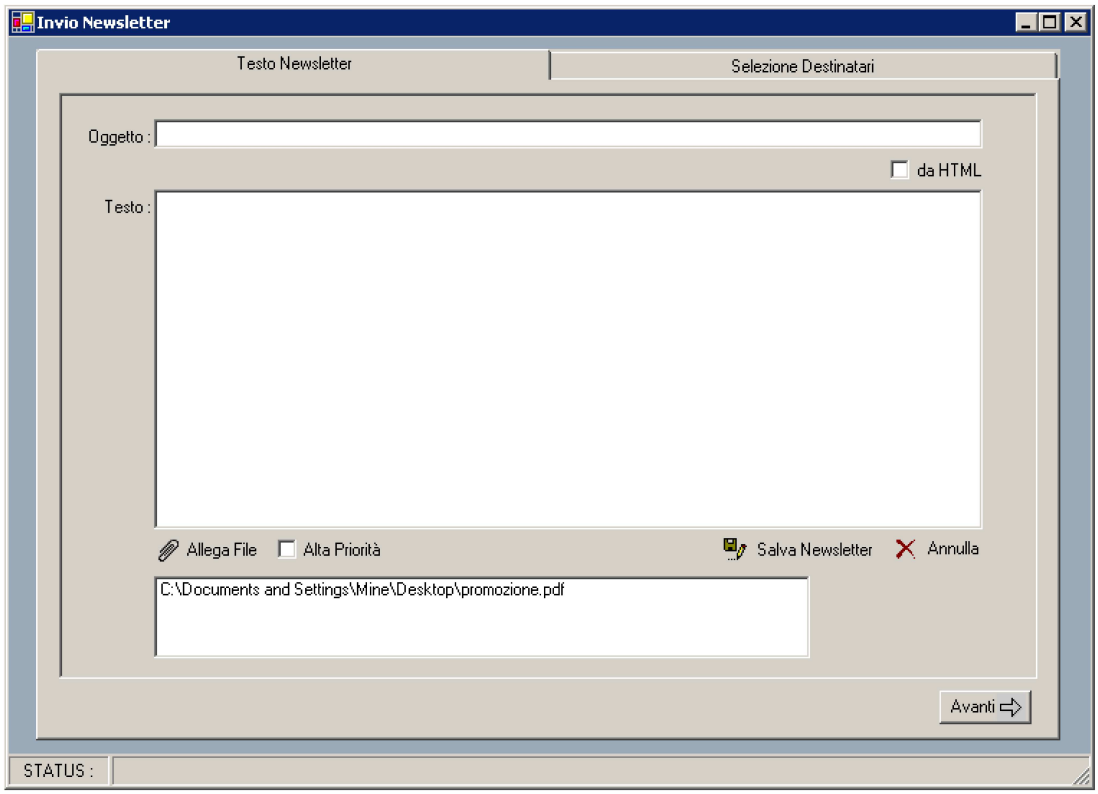

Figura 4.2: Invio e-mail

Per rendere più semplice e efficace la compilazione della form è stata prevista la possibilità di selezionare un file HTML come corpo del messaggio ed inoltre di indicarne la priorità di invio.
L'operazione di selezione dei contatti è stata implementata tramite un filtro, costruito sui parametri di selezione, applicato ad una vista generata dall'unione delle tabelle interessate dai parametri di ricerca.
Per permettere l'invio di messaggi di posta il .Net Framework mette a disposizione una serie di oggetti specifici, contenuti nella libreria `System.Web.Mail`. In particolare un nuovo messaggio di posta viene creato istanziando un nuovo oggetto `MailMessage` e tutte le caratteristiche del messaggio da inviare vengono impostate tramite le proprietà specifiche di questo oggetto. In questo caso vengono impostate le seguenti proprietà:

- `MailMessage.From` per specificare il mittente
- `MailMessage.To` per specificare i destinatari
- `Mailmessage.Subject`, che contiene l'oggetto della mail
- `MailMessage.Body`, il corpo del messaggio
- `MailMessage.Attachments` che contiene gli allegati
- `MailMessage.Priority` per settare la priorità del messaggio

Il messaggio viene poi effettivamente inviato attraverso l'uso della classe `SmtpMail`. in primo luogo è necessario specificare un server SMTP per l'invio attraverso la proprietà `SmtpMail.SmtpServer`, e successivamente il recapito della e-mail avviene invocando il metodo `Send(Messaggio)` della classe `SmtpMail`.

Per ultimo, la possibilità di importare dati da fogli di lavoro Excel è stata realizzata solo in parte, perché ritenuta non essenziale allo scopo principale del sistema. Si è preferito utilizzare il poco tempo rimanente per rendere più efficienti le funzioni già implementate. In particolare è stata realizzata la parte relativa all'interfaccia grafica e il segmento di codice che perette di leggere il foglio Excel e visualizzare ogni singola riga sull'interfaccia attraverso un controllo di tipo griglia. L'ultima sezione, cioè l'effettivo mapping dei campi componenti la riga visualizzata con i campi corrispondenti nella tabella dei contatti, richiede un'ulteriore analisi per poter ottenere una uniformità nella definizione del metodo di codifica, in quanto si è riscontrata una non uniformità nella rappresentazione dei dati nei file Excel.

Capitolo 5

Conclusioni e possibili sviluppi

Lo scopo di questo documento è stato quello di descrivere il lavoro effettuato durante il periodo di stage/tirocinio svolto presso la ditta Vemer S.p.A. con l'obiettivo di implementare un'applicazione per la gestione della rubrica di contatti dell'azienda.

Il lavoro si è svolto in più fasi successive, ognuna finalizzata ad uno scopo specifico. In primo luogo, nella prima parte del periodo di tirocinio, molte ore delle prime settimane sono state dedicate all'acquisizione delle competenze necessarie allo sviluppo dell'intero sistema.

La fase successiva invece è stata dedicata alla conoscenza approfondita del problema. Ripetuti colloqui con i responsabili dei vari settori dell'azienda hanno permesso inoltre di conoscere le dinamiche e le peculiarità dell'organizzazione interna di un'industria moderna.

La fase finale è stata quella in cui si è svolto effettivamente il lavoro di progettazione. Ciò ha permesso di trovarsi di fronte allo sviluppo di un'applicazione reale e di arricchire le conoscenze tecnologiche sulla base delle scelte effettuate.

Il lavoro, nelle sue componenti essenziali, è stato portato a termine nei tempi previsti e con scrupolo e diligenza, ottenendo un positivo riscontro sia dal responsabile di settore che dagli utenti finali.

Ci si rammarica che, per mancanza di tempo, non sia stato possibile realizzare interamente alcune componenti accessorie del sistema. Questo e la realizzazione dell'invio differito delle newsletter saranno uno dei punti di partenza di possibili sviluppi futuri del software. Inoltre potrà costituire la base per un sistema più complesso di gestione ordini e assistenza. In tal senso l'intero pacchetto andrà

integrato con le informazioni relative agli articoli prodotti. Per soddisfare ulteriormente le esigenze della clientela, si potrà dotare il sito internet dell'azienda di una pagina web che permetterà di visualizzare e/o scaricare le informazioni sui prodotti presenti nell'archivio.

Appendice A

Stored procedure

Uno degli strumenti fondamentali che ha pieno supporto e che è pienamente implementato nel DBMS scelto sono le Stored Procedure.
Durante lo sviluppo di una applicazione client le richieste per l'interrogazione di dati possono essere inviate direttamente al database, dal programma, tramite istruzioni di INSERT, SELECT, UPDATE o altro ancora, oppure è possibile creare piccoli programmi o funzioni all'interno del database e poi farle eseguire dal programma esterno. Queste procedure si chiamano Stored Procedure.

Una stored procedure,come specificato in [7], è un programma scritto in SQL od in altri linguaggi, mantenuto nel database stesso, archiviato nel cosiddetto database data dictionary. Spesso è scritta in versioni proprietarie di sql, che sono dei veri e propri linguaggi strutturati (Max DB può operare in INTERNAL, ANSI, DB2 e ORACLE MODE come versioni di SQL).
Si tratta di moduli o routine che incapsulano codice in modo da poterlo riutilizzare. Una stored procedure può accettare parametri di input, restituire al client risultati tabulari o scalari, richiamare istruzioni DDL (Data Definition Language) e DML (Data Manipulation Language) e restituire parametri di output.
Generalmente il DBMS compila le stored procedure, il che significa che le ottimizza e stabilisce il piano di esecuzione delle query.

Vediamo ora quali sono i vantaggi che l 'uso di questo strumento può portare nello sviluppo dell'intero pacchetto software. Indipendentemente da come sono definite, la principale peculiarità delle stored procedure è quella di poter essere invocate tramite comandi SQL oppure direttamente dalle applicazioni che interagiscono con il database (ad es.,tramite ODBC).

In questo senso esse aiutano a separare l'applicazione client dalla sottostante architettura dati e a semplificare la scrittura del codice, aumentando la stabilità e la scalabilità dell'applicazione.

Le stored procedure aumentano enormemente le performance dei programmi perché la compilazione avviene una volta sola, al loro inserimento. Ogni volta che la procedura viene richiamata viene semplicemente eseguita e questo aumenta significativamente le prestazioni.

I casi in cui l'utilizzo delle procedure è fortemente consigliato sono principalmente quando si abbia a che fare con molti client, scritti magari in linguaggi differenti o basati su piattaforme diverse, i quali però necessitino tutti di effettuare le stesse operazioni sui database oppure quando la sicurezza delle operazioni sia una delle caratteristiche principali del progetto: utilizzando stored procedure si introduce un ulteriore livello di sicurezza dato che le applicazioni e gli utenti non accederanno direttamente le tabelle ma potranno solamente eseguire particolari funzioni all'interno delle quali sarà possibile inserire dei meccanismi di autenticazione dedicati.
Di seguito viene riportato un esempio di stored procedure implementata nel sistema.

```
CREATE DBPROC SP_TAB_NEWSLETTER_UPDATE(IN PARIDNEWS INT,
                                       IN PARDSNEWS VARCHAR(50),
                                       IN PARDTNEWS TIMESTAMP,
                                       IN PARTESTONEWS VARCHAR(4000),
                                       IN PARNOTANEWS VARCHAR(1000),
                                       IN PARUSRMOD CHAR (8),
                                       OUT PARDTAMOD TIMESTAMP,
                                       INOUT PARINDMOD FIXED(16),
                                       OUT RETVAL INT)


AS

VAR IND_MOD_RECORD FIXED(16);
SET RETVAL = 0;
```

```
SUBTRANS BEGIN;

SELECT INDMOD INTO :IND_MOD_RECORD
FROM PGM.NEWSLETTER
WHERE IDNEWSLET = :PARIDNEWS;

IF $RC <> 0 THEN BEGIN
    SUBTRANS ROLLBACK;
    IF $RC = 100 THEN BEGIN
        SET RETVAL = 1;
        STOP (1, 'Record non piu'' presente.
               Eliminato da un altro utente.');
    END;
    SET RETVAL = 99;
    STOP ($RC, $ERRMSG);
END;

IF (IND_MOD_RECORD <> PARINDMOD) THEN BEGIN
    SUBTRANS ROLLBACK;
    SET RETVAL = 2;
    STOP (2,'Record gia'' modificato da un altro utente.');
END;

UPDATE PGM.NEWSLETTER
SET  DSNEWSLET = :PARDSNEWS,
         DTNEWSLET = :PARDTNEWS,
         TESTO = :PARTESTONEWS,
         NOTA = :PARNOTANEWS,
         USRMOD = :PARUSRMOD,
         DTMOD = TIMESTAMP,
         INDMOD = SEQ_IND_MOD.NEXTVAL
WHERE IDNEWSLET = :PARIDNEWS;
```

```
IF $RC <> 0 THEN BEGIN
    SUBTRANS ROLLBACK;
    SET RETVAL = 99;
    STOP ($RC, $ERRMSG);
END;

SELECT INDMOD, DTMOD INTO :PARINDMOD, :PARDTAMOD
FROM PGM.NEWSLETTER
WHERE IDNEWSLET = :PARIDNEWS;

IF $RC <> 0 THEN BEGIN
    SUBTRANS ROLLBACK;
    IF $RC = 100 THEN BEGIN
        SET RETVAL = 2;
        STOP (2,'Record non presente dopo inserimento.');
    END;
    SET RETVAL = 99;
    STOP ($RC, $ERRMSG);
END;

SUBTRANS END;
```

Si può notare come viene usato il campo INDMOD confrontandolo con il parametro corrispondente passato alla stored procedure per il controllo degli accessi concorrenti.

Bibliografia

[1] Erik Brown, *Windows Forms programming with C#*, Manning, 2002

[2] Ramez A. Elmasri, Shamkant B. Navathe, *Sistemi di basi di dati, fondamenti*, Addison Wesley, 2001

[3] Martin Fowler, *UML distilled: Applying the standard Object Modelling Language*, Addison Wesley, 2003

[4] Jesse Liberty, *Programming C#, 4th edition*, O'Reilly, 2005

[5] Gregory S. Macbeth, *C# programmers handbook*, Apress, 2004

[6] Rebecca M. Riordan, *ADO.NET 2.0 passo per passo*, Mondadori informatica, 2006

[7] MaxDB Library, `http://wiki.sdn.sap.com/wiki/`

[8] MDSN Library, `http://msdn.microsoft.com/`

[9] Regular Expression, `http://www.regular-expressions.info/`

Ringraziamenti

Ringrazio prima di tutto il Ch.mo Prof. Gradenigo per essere stato il mio tutor universitario durante il periodo di tirocinio e per essere sempre stato molto disponibile durante il periodo di stesura di questo documento; l'ing. Dal Gallo, tutor aziendale, per aver dimostrato la sua disponibilità a chiarimenti e spiegazioni durante lo stage in azienda; Vania Lira dell'Associazione Industriali della provincia di Belluno per avermi dato la possibilità di partecipare a questo stage.

Ringrazio i miei genitori per il sostegno psicologico ed economico datomi in questo percorso di studi. Ringrazio soprattutto mia madre che ha saputo sostenermi e guidarmi nei momenti difficili credendo nella mie capacità. Ringrazio anche mio padre per i continui incoraggiamenti a caratteri intimidatorio che mi ha fatto in tutti questi anni e per avermi sempre trovato qualche compito da eseguire nel periodo in cui ero a casa e avrei dovuto scrivere questo documento. Ringrazio in fine mio fratello per tutti quei consigli e aiuti che mi ha saputo dare durante il periodo di studi ed anche perchè sa che può contare su di me nel momento del bisogno.

Ringrazio tutti i miei parenti, da mia nonna ai miei zii e cugini per l'affetto dimostrato soprattutto nei momenti di necessità; Fabio e Paolo per i momenti all'insegna dell'allegria e divertimento trascorsi (vedi feston, sagra di Fonzaso, FonzieVolley) ; Barbara e Chiara per la fiducia che mi dimostrano come loro tecnico informatico.

Prima di ringraziare tutti i miei amici volevo ringraziare la GruppoForm S.p.A solo per il contributo economico dato, che mi ha permesso di sopravvivere negli ultimi due anni fino ad oggi.

Tra tutti i miei amici di Padova voglio ringraziare in particolare Matteo e Richard, storici amici e condomini di via 15/bis; Matteo per aver condiviso la sua connessione internet e per i consigli e aiuti, sia universitari e non, che mi ha saputo dare nel momento del bisogno e Richard che da buon psicologo mi ha sempre sostenuto e incoraggiato nei momenti difficili. Li ringrazio anche per i

bei momenti trascorsi assieme, dalle pizze di condominio ai pomeriggi e serate a chiacchierare e a spettegolare su qualsiasi cosa. Ringrazio anche le due trentine per le cene offerte e per le lunghe chiacchierate alla finestra. Come non ringraziare Leo Diego per gli innumerevoli mercoledì trascorsi in piazza per il canonico rito dello spritz e per tutte le ragazze che mi ha presentato. Ringrazio tantissimo anche il nonno e tutta la compagnia (martin, il cuffi, beppone, il bistecca, il mele, gioffry e tutti quelli che ora non ricordo) per le partite a calcetto e le serate alternative in prato e in piazza. Ringrazio soprattutto il nonno, per essere stato un fedele compagno di studi e anche per la bella avventura di Londra.

Ringrazio tutti gli amici di Fonzaso e dintorni per tutte le avventure passate assieme. Ringrazio soprattutto gli amici dalla Dynamo(meglio conosciuta come team stramal) e i tifosi della curva americhe e del nucleo romanella per le cene da franco zurigo e gli infiniti sabato sera trascorsi tra impero mezzaterra e cicchetteria. Ringrazio inoltre tutti gli amici conosciuti nelle varie squadre di calcio in cui ho militato, soprattutto gli amici di Foen per gli storici dopo-allenamento del venerdì sera.

Ringrazio anche Matteo ed Andrea per avermi sopportato tutti quei sabato che andavo a trovarli in negozio ed anche per l'occhio di riguardo che hanno nei miei confronti come loro cliente fidato.

Ringrazio Mark Zuckerberg che nell'ultimo periodo mi ha fatto perdere pomeriggi interi con i suoi giochetti tipo brain challenge e geo challenge.

Ringrazio inoltre gli InFlames, Cradle of Filth, Within Temptation, Manowar, Iron Maiden, Dark Tranquillity, Arch Enemy che mi hanno allietato con la loro musica durante tutti i viaggi in macchina a Padova e nei momenti di tempo libero; La Bethesda Softwork per Oblivion e la Rockstar Games per GTA IV. Ringrazio inoltre J.R.R. Tolkien e P. Jackson per la trilogia del Signore degli anelli.
Alla fine non mi resta che dire...Grazie a tutti!

Roberto