

UNIVERSITÀ DEGLI STUDI DI BOLOGNA
FACOLTÀ DI SS.MM.FF.NN.
Corso di Laurea in Informatica

**Progettazione e realizzazione
di un database finalizzato alla
digitalizzazione di cataloghi cartacei**

Tesi di Laurea di:
Sara Albieri

Relatore:
Prof. Danilo Montesi

Anno Accademico 2009 - 2010

INDICE

1. Introduzione	7
1.2 Specifiche del progetto	9
1.3 Un caso reale: Caprari.....	9
2. Requisiti	11
2.1 Raccolta dei requisiti.....	11
2.1.1 Gli stakeholder	11
2.1.2 I cataloghi esistenti e i dati del cliente.....	11
2.1.3 Caratteristiche di un catalogo	12
2.1.4 Il software di impaginazione	13
2.2 Analisi dei requisiti	14
2.2.1 Il processo attuale	14
2.2.2 Obiettivo finale	15
2.2.3 Come arriviamo all'obiettivo finale	16
2.2.4 Requisiti aziendali.....	19
3. L'analisi.....	21
3.1 I cataloghi campione	21
3.2 Le basi di dati dei clienti.....	25
3.3 Glossario dei termini.....	27
3.4 Le entità e le relazioni.....	29
3.4.1 I prodotti e le famiglie	29
3.4.2 Varianti, proprietà tecniche e descrizioni	33
3.4.3 Lingue e internazionalizzazione	36
3.4.4 La gestione delle risorse	37
3.4.5 Prezzi e scontistica.....	38
3.4.6 L'ordinamento	39

4. Progettazione.....	41
4.1 Gli identificatori.....	41
4.2 Il problema dell'ordinamento	42
4.3 I prodotti e le famiglie.....	42
4.4 Varianti, proprietà tecniche e descrizioni	45
4.5 Lingue e internazionalizzazione	49
4.6 La gestione delle risorse.....	50
4.7 Prezzi e scontistica	51
4.8 Integrità referenziale	53
4.9 Traduzione verso il modello relazionale.....	54
4.10 La normalizzazione.....	55
4.10.1 Dipendenze funzionali	56
4.10.2 Forme normali.....	56
5. Implementazione	63
5.1 I prodotti e le famiglie.....	63
5.2 Varianti, proprietà tecniche e descrizioni	67
5.3 Lingue e internazionalizzazione	70
5.4 La gestione delle risorse.....	73
5.5 Prezzi e scontistica	74
5.6 Architettura del sistema	74
5.6.1 Scelta della piattaforma di sviluppo	75
5.6.1.1 Server.....	76
5.6.1.2 DBMS - Microsoft Sql Server	76
5.6.1.3 Linguaggio di programmazione	77
5.6.1.4 Ambiente di sviluppo	77
6. Procedura automatica di esportazione	79
6.1 Software per l'impaginazione.....	79
6.2 Procedura di esportazione	82
6.2.1 Il menu	82

6.2.2 La logica applicativa.....	83
6.2.2.1 La sezione esporta dati del cliente.....	83
6.2.2.2 La sezione esporta generale.....	85
7. Un caso reale: procedura di importazione	87
7.1 La struttura	87
7.1.1 L'interfaccia grafica.....	87
7.1.1.1 Il menu	88
7.1.2 La logica applicativa.....	93
7.1.2.1 La sezione importa.....	93
7.1.2.2 La sezione cliente	97
8. Conclusioni	103
8.1. Sviluppi futuri	103
Appendice A: Codice SQL	105
Creazione del database.....	105
Creazione delle tabelle	105
Elenco delle principali insert utilizzate.....	110
Elenco delle principali select utilizzate	111
Appendice B: Estratto di codice PHP	115
Appendice C: Informazioni riguardanti lo sviluppo	123
Indice delle figure	125
Bibliografia	127
Ringraziamenti	129

1. Introduzione

La ditta Deca s.r.l., presso la quale svolgerò la mia tesi, ha sede a Lugo (RA), è stata fondata nel 1976 e ad oggi ha circa 70 dipendenti dislocati su 2 sedi (Lugo e Modena). Dal 2002 Deca s.r.l. è partner di SPX Corporation¹, una multinazionale americana, e appartiene al settore SPX Information Services.

I vari campi di lavoro di Deca sono :

- **Manualistica Tecnica:** offre a diversi livelli, soluzioni e servizi per la documentazione tecnica tra cui:
 - Manuali di Officina e di Riparazione;
 - Manuali d'Uso e Manutenzione;
 - Schemi Elettrici Funzionali;
 - Cataloghi Parti di Ricambio;
 - Cataloghi Commerciali;
 - Temperi di Garanzia;
 - Traduzioni;
 - Stampa;
- **Servizio di Outsourcing:** offre il servizio di outsourcing completo, dalla redazione alla stampa, fornendo una soluzione altamente personalizzata;
- **Soluzioni Software dedicate:** ha sviluppato soluzioni software in grado di aumentare l'efficienza di produzione e la flessibilità nel proporre e "vestire" la documentazione tecnica sempre più come servizio a valore aggiunto. Tra le loro soluzioni:
 - Schemi Elettrici interattivi;
 - Soluzioni per la redazione e pubblicazione on-line di Cataloghi Ricambi;
 - Soluzioni di e-commerce;
 - Configuratore della documentazione tecnica;
 - Soluzioni di impaginazione e pubblicazione automatica;

¹ [SPX]

- Soluzioni di redazione XML (XMLCompass);
 - Sistema per la documentazione integrata;
 - Interfacciamento con ERP;
 - Content Management System.
- Training: supporta i propri Clienti nell'intero processo della formazione tecnica.

I contatti di Deca sono:

DECA Srl SPX Information Services

Via Vincenzo Giardini, 11 - 48022 Lugo (Ra)

› Tel: +39 0545-216611

› Fax: +49 0545-216610

› E-Mail: deca@spx.com

› Internet: <http://www.spxservicesolutions.it>

L'azienda mi ha messo a disposizione due tutor, uno operante nell'ufficio di Information Technology e uno operante nell'ambito dei processi aziendali.

Il mio tutor aziendale del settore informatica si chiama Demetrio Megali, è un consulente esterno che supporta Deca nell'ufficio Information Technology. Demetrio è dipendente di Synchronika S.r.l.² di Ferrara.

Il mio tutor aziendale dell'area organizzativa si chiama Graziella De Donatis ed è anche lei un consulente esterno che opera nel campo della consulenza industriale, e supporta Deca in ambito manageriale. E' suo il compito di introdurmi nell'ambiente lavorativo.

² [Synchronika]

1.2 Specifiche del progetto

Il progetto che ho concordato con Deca per la mia tesi consiste nel creare una base di dati che permetta di rendere molto più facile agli impaginatori la realizzazione di cataloghi cartacei.

Abbiamo pensato di realizzare un database standard che possa essere, all'occorrenza, espanso o ridotto per immagazzinare i dati che si ricevono dal cliente. Da tale database, grazie ad una procedura automatica sarà possibile creare un documento intermedio per la creazione del catalogo che sarà successivamente rivisto dall'impaginatore.

1.3 Un caso reale: Caprari

Attualmente tra i clienti di Deca vi è la ditta Caprari³ S.p.A che si affida al gruppo per la realizzazione di più cataloghi contenenti tutti i loro prodotti.

Il gruppo Caprari ha sede a Modena ed è tra le principali realtà internazionali nella produzione di pompe ed elettropompe centrifughe e nella creazione di soluzioni avanzate per la gestione del ciclo integrato dell'acqua.

Abbiamo pensato di utilizzare i dati di questo cliente come test per la base di dati che andrò a realizzare essendo gli stessi complessi ed eterogenei da elaborare manualmente.

³ [Caprari]

2. Requisiti

Per raccolta dei requisiti si intende la completa individuazione dei problemi che l'applicazione da realizzare deve risolvere e le caratteristiche (come i dati e le operazioni su di essi) che tale applicazione dovrà avere.

2.1 Raccolta dei requisiti

I requisiti dell'applicazione provengono da diverse fonti, tra le quali:

- gli stakeholder che sono tutti coloro che hanno un interesse diretto o indiretto sui risultati del sistema software
- la documentazione esistente attinente al problema allo studio,
- eventuali realizzazioni preesistenti dell'applicazione.

2.1.1 Gli stakeholder

Il termine stakeholder viene usato per identificare tutti coloro che hanno un interesse diretto o indiretto sui risultati del sistema software che si andrà a sviluppare, quindi sono stati individuati i seguenti gruppi di individui:

- gli impaginatori;
- gli sviluppatori software.

In questo caso il coinvolgimento del cliente non è fondamentale, perché il sistema software non è destinato direttamente a lui; infatti, il cliente ha un interesse indiretto circa il funzionamento della base di dati perché più questa agevola il lavoro degli impaginatori e prima lui riceverà il catalogo finito.

2.1.2 I cataloghi esistenti e i dati del cliente

Per la realizzazione della base di dati sono stati consultati vari

cataloghi cartacei precedentemente impaginati a mano e alcune basi di dati di clienti a disposizione dell'azienda.

Da un esame preliminare emerge che è possibile estrarre delle caratteristiche comuni che verranno formalizzate nel loro complesso in fase di analisi: questo incoraggia lo svolgimento del progetto perché lascia presupporre l'esistenza di un meta-modello comune ai dati dei clienti.

2.1.3 Caratteristiche di un catalogo

Un catalogo è destinato ad un'utenza umana, quindi necessita di una adeguata presentazione, di cui si occupano gli impaginatori, e di contenuti consoni con gli obiettivi aziendali del cliente.

E' necessario, quindi, che nella sua redazione si tenga conto di fattori diversi, quali:

- aspetto produttivo;
- aspetto tecnico;
- aspetto marketing;
- aspetto commerciale.

L'aspetto produttivo riguarda le linee di produzione e i prodotti stessi. Ad esempio le aziende di piastrelle organizzano i loro prodotti in linee di produzione in base agli ambienti a cui pavimenti e rivestimenti sono destinati.

L'aspetto tecnico riguarda le caratteristiche dei prodotti commercializzati dal cliente. Include tutte le specifiche tecniche di importanza rilevante del prodotto che viene pubblicato nel catalogo.

Spesso un aspetto particolarmente rilevante è il materiale di cui è costituito un prodotto. Ad esempio un pompa idraulica in acciaio ha delle caratteristiche tecniche diverse da una in ferro, e tali caratteristiche influenzano la scelta del prodotto da parte dell'utente

finale che sfoglia il catalogo.

L'aspetto marketing comprende tutte le azioni aziendali riferibili al mercato destinate al piazzamento di prodotti, quali il copywriting che è l'arte di persuadere l'utente a compiere delle azioni (nel caso di un catalogo l'acquisto del prodotto) usando soltanto la scrittura.

Un esempio possibile è il mercato della telefonia mobile nel quale la scelta di un prodotto non è data dalle caratteristiche tecniche del prodotto stesso ma piuttosto dal massiccio utilizzo di artifici pubblicitari. Per questo è importante che il cliente aziendale in accordo con il proprio ufficio marketing fornisca i testi descrittivi da inserire nei cataloghi e le opportune traduzioni.

L'aspetto commerciale riguarda le politiche di prezzo adottate dall'azienda che possono essere influenzate dalla nazione in cui il prodotto viene venduto e quindi alla nazione a cui è destinato il catalogo.

Un esempio classico può essere il mercato automobilistico che impiega listini differenti a seconda della nazione in cui commercializza i prodotti.

Un altro aspetto commerciale da tenere in considerazione è la scontistica applicata ai prezzi che spesso viene pubblicizzata all'interno del catalogo. Un esempio tipico si ha nell'ambiente edile dove l'acquisto in grande quantità di materiale comporta una diminuzione del prezzo rispetto a quello di listino.

2.1.4 Il software di impaginazione

Il software utilizzato da Deca per l'impaginazione si chiama Frame Editor ed è prodotto dalla ditta Tesla Sistemi Informatici S.r.l. di Monza che utilizzato in accoppiata con Frame Maker di Adobe permette la realizzazione dei cataloghi.

Per completare il catalogo gli impaginatori devono fornire a Frame Editor un file di testo opportunamente formattato che contenga al suo

interno tutti i dati che vogliamo vengano stampati nel catalogo, concatenati in base alle specifiche imposte da Frame Editor.

In Frame Editor l'utilizzo di uno standard per l'inserimento dei dati agevola la creazione di un processo automatico per la migrazione dei dati dalla base di dati in oggetto al catalogo finale.

2.2 Analisi dei requisiti

L'analisi dei requisiti consiste nel chiarimento e nell'organizzazione delle specifiche dei requisiti. L'attività di analisi inizia con i primi requisiti ottenuti per poi procedere di pari passo con l'attività di raccolta.

2.2.1 Il processo attuale

Adesso quando il cliente ha la necessità di un nuovo catalogo, si rivolge a Deca fornendo i dati necessari tramite precedenti versioni dei cataloghi annotate manualmente, fogli di calcolo contenenti le informazioni rilevanti e ulteriori documenti in formato human readable.

Il lavoro attuale degli impaginatori consiste nella realizzazione dei cataloghi tramite il programma di impaginazione Frame Editor; ora infatti non vi è nessuna infrastruttura che aiuti gli impaginatori nel loro lavoro: i cataloghi vengono completati con i dati "a mano" estraendo le informazioni dagli uffici tecnici/marketing dei clienti attraverso i suddetti file Excel o appunti su vecchi cataloghi.

Si tratta di un lavoro lungo e laborioso che implica che tutta la logica applicativa sia svolta dall'impaginatore.

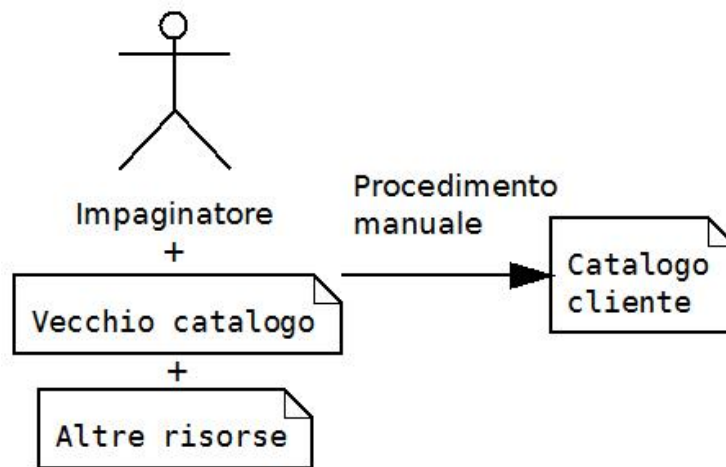


Figura 1- Come lavorano oggi

2.2.2 Obiettivo finale

L'obiettivo finale cercato da Deca consiste in un'infrastruttura che semplifichi e acceleri la creazione di cataloghi.

Si evidenzia che il collo di bottiglia del processo attuale è nell'immissione manuale da parte dell'impaginatore: tale processo potrebbe essere automatizzato da un livello software che funga da middleware tra le basi di dati aziendali e il programma di impaginazione.

Gli impaginatori, trattandosi di personale più orientato alla grafica che ai sistemi informativi, si occuperanno solamente di generare il file di testo da fornire a Frame Editor e di tutti i passi successivi relativi alla creazione del catalogo. Questa procedura deve essere il più user friendly possibile.

Gli impaginatori non si preoccuperanno dell'importazione dei dati nel nuovo catalogo ma sarà compito dell'ufficio informatico.

Gli sviluppatori dovranno adattare il database generico alle informazioni contenute nella base di dati del cliente e successivamente

revisare l'applicazione per poter importare i dati precedentemente caricati.

Il loro compito, nella parte dell'importazione, consiste prevalentemente nell'identificazione delle tabelle dalle quali estrapolare i dati e l'inserimento dei dati stessi nel database interno. Dovrà essere adattata anche l'applicazione software per la creazione del file di testo.

Un corretto riuso del codice consentirà agli sviluppatori di apportare solo minimi cambiamenti alla procedura di importazione, che risulta di volta in volta specifica per ogni cliente.

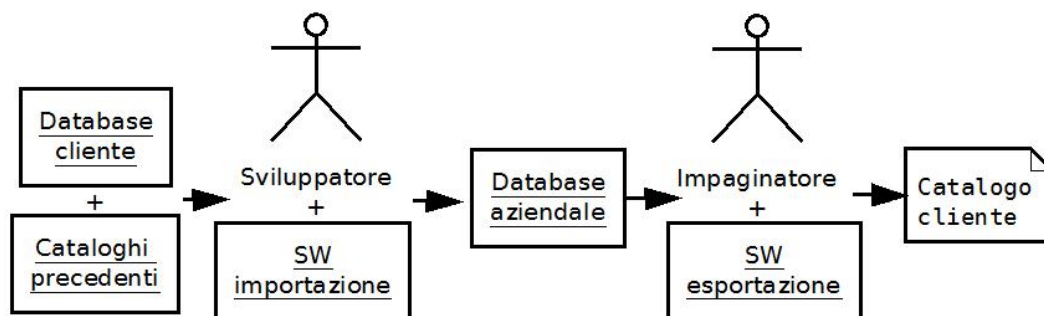


Figura 2 - Come sarà

2.2.3 Come arriviamo all'obiettivo finale

Il sistema che verrà realizzato prenderà in input il database completo del cliente importando all'interno della sua struttura solo i dati di puro interesse per il catalogo e per la creazione di immagini. Restituirà un file creato ad hoc utilizzato per l'impaginazione.

Il sistema può essere inteso come una black-box che fornisca agli attori in gioco funzionalità di importazione dei dati del cliente ed esportazione del file di testo per l'impaginazione.

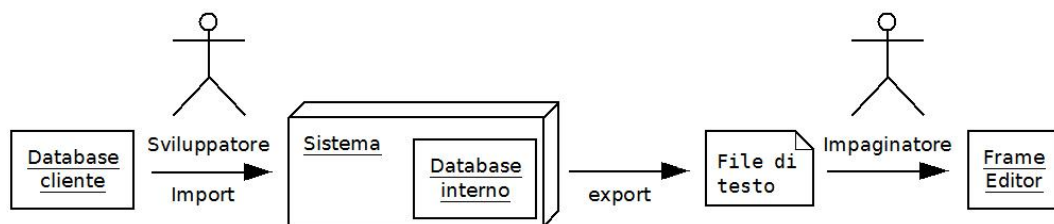


Figura 3 - Funzionamento di base

L'intervento dello sviluppatore è necessario per la realizzazione della procedura di importazione della base dati di uno specifico cliente: tuttavia il progettista è supportato dal fatto che il database interno propone una struttura standard dei dati che può essere impiegata come protocollo di scambio dati.

L'intervento dell'impaginatore si limita alla richiesta di un'operazione di esportazione al sistema complessivo. L'output ottenuto dal sistema potrà essere direttamente impiegato nel programma di impaginazione.

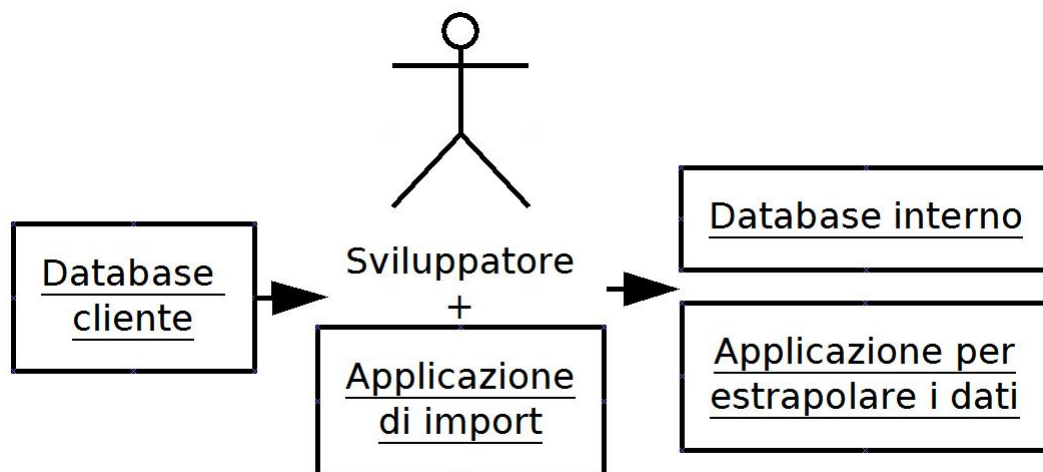


Figura 4 – Cosa farà lo sviluppatore

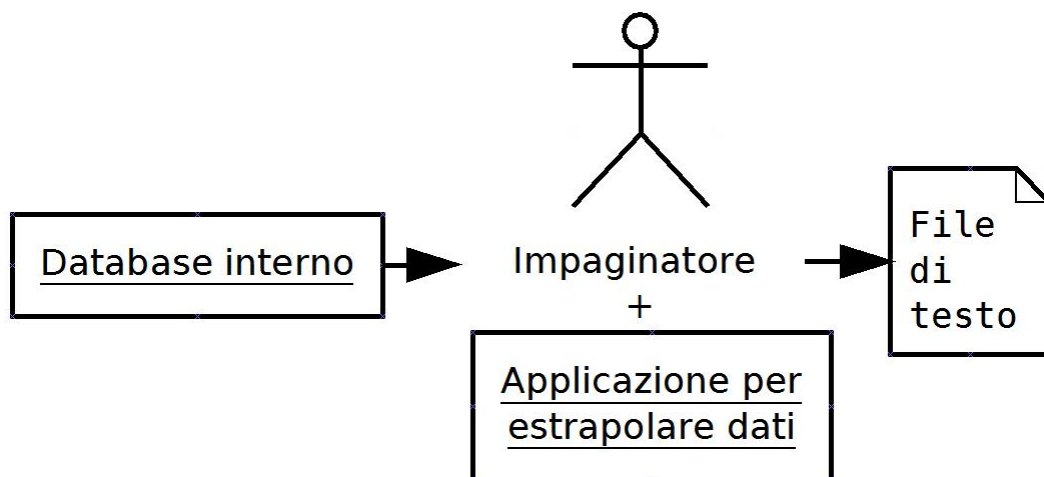


Figura 5 – Cosa farà l'Impaginatore

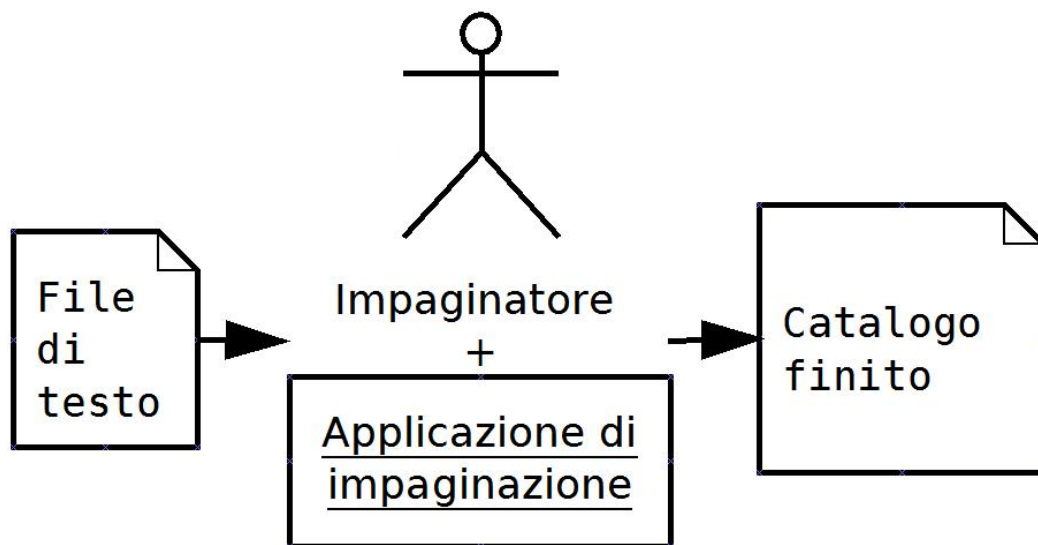


Figura 6 – Cosa farà l'impaginatore

2.2.4 Requisiti aziendali

L'azienda ha messo a mia disposizione tutti i software in suo possesso e mi impone di scegliere come soluzioni tecnologiche quelle già parte del loro patrimonio corrente onde evitare ulteriori investimenti.

Per ulteriori dettagli sulle tecnologie disponibili e impiegate si rimanda al capitolo sull'implementazione.

3. L'analisi

Per la realizzazione dell'analisi ho utilizzato una strategia mista⁴ perché combina i vantaggi della strategia top-down (si può descrivere inizialmente tutte le specifiche dei dati trascurandone i dettagli) e quelli della strategia bottom-up (decompone il problema in componenti più semplici).

3.1 I cataloghi campione

Per la realizzazione della base di dati sono stati consultati vari cataloghi cartacei precedentemente impaginati a mano. Sono stati individuati i tratti comuni principali di ogni catalogo e sono stati suddivisi in 4 macro categorie:

- Una parte generale iniziale contenente le caratteristiche del catalogo e le informazioni sull'azienda.
- Una parte descrittiva dei prodotti (i codici di ogni prodotto, le caratteristiche tecniche, la suddivisione dei prodotti in famiglie, ecc.);
- Le lingue disponibili (sempre più cataloghi dispongono di versione multilingua);
- Le rappresentazioni grafiche;
- Listino prezzi, nel caso si tratti di listini

⁴ [ATZ09]

prodotti

TARIFFE BUS		53 posti
Transfer:		
FS Rimini/Hotel Rimini		280.000
FS Bologna/Hotel Rimini		630.000
ESCURSIONI CON PARTENZA DA RIMINI		
	HD	FD
San Marino	380.000	660.000
Ravenna	505.000	675.000
Venezia (Via Romea)		960.000
Firenze (Via Muraglione)		1.000.000
Assisi/Perugia		960.000
Bologna	675.000	785.000
Ferrara	730.000	855.000
San Leo	430.000	
Gradara	405.000	
Verucchio e Sant'Arcangelo	405.000	
Gabicco/Castel di Mezzo/Gradara	405.000	
Italia in Miniatura o Fiabilandia o Rimini o Aquafan	380.000	
Frasassi	650.000	840.000
Loreto	650.000	840.000
Torriana/Montebello	425.000	
Mirabilandia	515.000	675.000
San Leo/San Marino		695.000

EXECUTIVE LA FIORITA



Situato fronte mare a **Miramare di Rimini**, completamente rimodernato dispone di tutte camere con servizi privati - telefono e Tv color - bar - ascensore - sala soggiorno e televisione - sala ristorante - piano bar - cucina genuina - parcheggio. Riscaldamento.

tante proprietà

Studenti	08/01-20/03	21/03-26/05 17/09-29/12
B.B.		
H.B.	33.500	34.500
F.B.	35.500	36.500
S.S.	15.000	15.000
S.D.		
Adulti	08/01-26/05 17/09-29/12	
B.B.		
H.B.	45.000	
F.B.	49.000	
S.S.	18.000	

prodotto

GRAN SAN BERNARDO



Situato sul lungomare di **Riccione**. Dispone di camere con servizi privati, telefono, balcone e cassaforte. Sala ristorante, piccola sala soggiorno con angolo Tv, bar e ascensore. Ambiente familiare.

proprietà

Studenti	09/04-19/05
B.B.	
H.B.	26.500
F.B.	29.000
S.S.	15.000
S.D.	
Adulti	09/04-30/05 17/09-30/09
B.B.	
H.B.	33.000
F.B.	36.500
S.S.	16.000

prodotto

BELVEDERE



Situato a **Miramare di Rimini** fronte mare, dispone di tutte camere con servizi privati e telefono - ascensore - bar - sala soggiorno - ristorante - ampio giardino. Riscaldamento.

Studenti	08/01-20/03	21/03-26/05 17/09-29/12
B.B.		
H.B.	33.500	34.500
F.B.	35.500	36.500
S.S.	15.000	15.000
S.D.		
Adulti	08/01-26/05 17/09-29/12	
B.B.		
H.B.	45.000	
F.B.	49.000	
S.S.	18.000	

Figura 7 - Esempio di catalogo di vacanze

ZAINI E BORSE

ZAINI E BORSE

38

AN74062 Zainetto 600D

Pratico zainetto multuso in poliestere 600D 30x40x17
Cartone: 50 Imballo: 1
col: Colori: ROSSO, ARANCIONE, BEIGE, AZZURRO
Misure di stampa: 12x8

Pezzi:	50	100	250	500	1000	2000
Prezzo unit.	€ 3,86	€ 3,68	€ 3,43	€ 3,34	€ 3,28	€ 3,09
St. 1 col.	€ 0,80	€ 0,75	€ 0,70	€ 0,65	€ 0,65	€ 0,65
St. 2 col.	€ 1,75	€ 1,65	€ 1,55	€ 1,25	€ 1,25	€ 1,25

Tipologia: Serigrafia Impianto Stampa € 45,00 per colore



SZA73 ZAINO CON TASCA

Cartone: 24 Imballo: 1
Colori: Blu Navy, Azzurro Sky, Nero Beige, Bordeaux Beige

Pezzi:	50	100	250	500	1000	2000
Prezzo:	€ 6,33	€ 6,13	€ 5,88	€ 5,46	€ 5,26	€ 5,06
St. 1 col.	€ 0,90	€ 0,75	€ 0,70	€ 0,65	€ 0,65	€ 0,65
St. 2 col.	€ 1,75	€ 1,65	€ 1,55	€ 1,25	€ 1,25	€ 1,25

Tipologia: Serigrafia Impianto Stampa € 45,00 per colore



SZA29 ZAINO VIAGGIO

Cartone: 10 Imballo: 1
Colori: Marrone Beige, Nero Blu Royal

Pezzi:	50	100	250	500	1000	2000
Prezzo:	€ 9,83	€ 9,06	€ 8,68	€ 7,88	€ 7,23	€ 7,23
St. 1 col.	€ 0,80	€ 0,75	€ 0,70	€ 0,65	€ 0,65	€ 0,65
St. 2 col.	€ 1,75	€ 1,65	€ 1,25	€ 1,25	€ 1,25	€ 1,25

Tipologia: Serigrafia Impianto Stampa € 45,00 per colore



SZA66 ZAINO TEMPO LIBERO

Cartone: 30 Imballo: 1
Colori: Arancio Blu Navy, Beige Nero, Turchese Beige

Pezzi:	50	100	250	500	1000	2000
Prezzo:	€ 4,89	€ 4,64	€ 4,44	€ 4,28	€ 3,96	€ 3,83
St. 1 col.	€ 0,80	€ 0,75	€ 0,70	€ 0,65	€ 0,65	€ 0,65
St. 2 col.	€ 1,75	€ 1,65	€ 1,25	€ 1,25	€ 1,25	€ 1,25

Tipologia: Serigrafia Impianto Stampa € 45,00 per colore



Ricamo a Preventivo minimo pz. 50

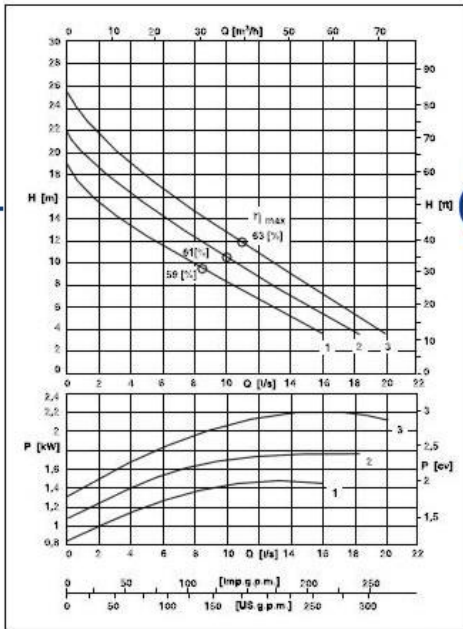
Figura 8 - Esempio di catalogo di prodotti

caprari

ELETTROPOMPE SOMMERGIBILI PER LIQUIDI CARICHI
ELECTRIC SUBMERSIBLE SEWAGE PUMPS
ELECTROPOMPES SUBMERSIBLES POUR LIQUIDES CHARGÉES

famiglia **KC+**
famiglia **DN 65**

rappresentazione



2/50 Hz **KCM065F**
girante monocale
single-channel impeller
roue monocanal
componente base
rappresentazione

CARATTERISTICHE TECNICHE - TECHNICAL FEATURES
CARACTERISTIQUES TECHNIQUES

Elettropompa tipo Electric pump type Electropompe type	Passaggio libero Free passage Passage libre	Sonda termica Thermoprobe Sonde thermique	Sonda di conduttività Conductivity probe Sonde de conductivité
KCM065F... +21N1	Ø 40	Su richiesta On Request Sur demande	Su richiesta On Request Sur demande
KCM065F... +21X1		Sì Yes Oui	Sì Yes Oui

proprietà tecniche

CARATTERISTICHE DI FUNZIONAMENTO - OPERATING DATA - CARACTERISTIQUES DE FONCTIONNEMENT

Elettropompa tipo Electric pump type Electropompe type (1)	Curva Curve Courbe	Potenza motore Motor rating Puissance moteur	Manicha Capacity Ref. admettent	PORTATA - CAPACITY - DEBIT... [l/s] [m³/h]																
				N°	P2 [kW]	DN [mm]	0	4	6	8	9	10	11	12	13	14	15	16	18	20
KCM065FG+001521N1	1	1.5	ø 65	19	13.5	11.5	10	9.2	8.4	7.6	6.8	6	5.2	4.4						
KCM065FD+001821N1	2	1.8		22	16.5	14	12.5	11.5	10.5	9.7	8.8	8	7	6.2	5.4					
KCM065FA+002221N1	3	2.2		25.5	19.5	17	15	14	13	12.5	11	10.5	9.4	8.6	7.6	5.7	3.6			

proprietà tecniche

NOTE - NOTES - NOTES

P2 = Potenza resa dal motore - Power rated by the motor - Puissance restituée par le moteur.
Tolleranze sulle prestazioni secondo norme UNI/ISO 9906 Livello 2 - Performance tolerance as per UNI/ISO 9906 Gra de 2 - Tolérances sur les performances selon normes UNI/ISO 9906 Niveau 2.
(1) Per i modelli in versione antideflagrante ATEX II 2G Exd IIB T4, la parte finale della sigla dell'elettropompa diviene +41X1
For models in the ATEX II 2G Exd IIB T4 explosion proof version, the final part of the electric pump code becomes +41X1
Pour les modèles version antideflagrante ATEX II 2G Exd IIB T4, le suffixe de l'électropompe devient +41X1
Per caratteristiche motori vedere a pagina 17 - For motor performances specification see page 17 - Pour caractéristiques techniques moteurs voir page 17.
Per accessori vedere a pagina 15/16 - For the accessories see at page 15/16 - Pour les accessoires voir page 15/16.

varianti

Figura 9 – Esempio di catalogo di prodotti

3.2 Le basi di dati dei clienti

Ho anche visionato le basi di dati di alcuni clienti e ho riscontrato molte similitudini tra loro.

Ogni base di dati contiene al suo interno tutti i prodotti creati e commercializzati dal cliente, catalogati attraverso una matricola o un codice prodotto. Da ogni prodotto è possibile risalire alla famiglia di appartenenza, agli accessori disponibili, e ai pezzi base che li compongono (che sono a loro volta commercializzati in quanto pezzi di ricambio).

Nei database dei clienti si trovano inoltre tutte le caratteristiche tecniche di ogni prodotto e di ogni materiale che lo compone.

Il cliente fornisce anche un insieme di rappresentazioni grafiche (sotto forma di immagini vettoriali o CAD) dei suoi prodotti e dei diagrammi grafici delle proprietà tecniche dei prodotti.

Per le possibili lingue del catalogo, generalmente, arriva dai traduttori un documento con tutte le traduzioni dei testi nelle lingue richieste dal cliente.

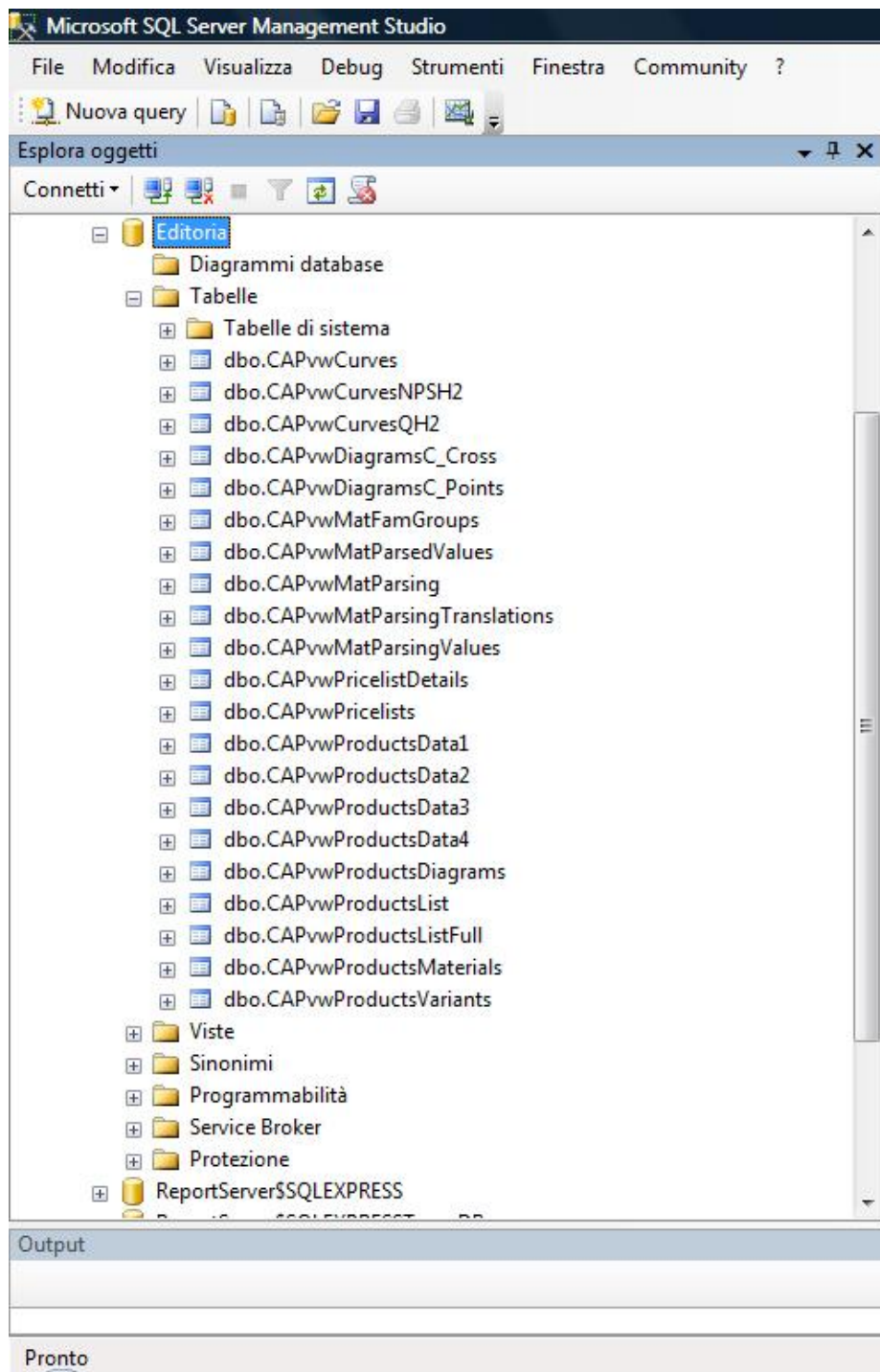


Figura 10 - Esempio di base di dati del cliente

3.3 Glossario dei termini

Durante l'analisi dei cataloghi cartacei e dei database sono emerse analogie nei contenuti che possono così essere riassunte.

Ogni catalogo deve spiegare:

- Prodotti;
- Accessori;
- Famiglie dei prodotti e degli accessori;
- Proprietà tecniche dei prodotti;
- Descrizione in linguaggio naturale di ogni prodotto in più lingue;
- Rappresentazioni grafiche dei prodotti;
- Eventuale prezzi.

Termine	Descrizione	Sinonimi	Collegamenti
Prodotto	E' un oggetto venduto	Oggetto	Componente base, Accessorio, Prodotto finito, Prezzo, Descrizione, Proprietà tecniche
Componente Base	Pezzo che compone un prodotto	Pezzo di ricambio	Accessorio, Prodotto finito, Prezzo, Descrizione, Proprietà tecniche

Accessorio	Pezzo che è in aggiunta o che accompagna il prodotto	Complementare, aggiunta	Prodotto, Componente Base, Prezzo, Descrizione, Proprietà tecniche
Prezzo	E' il valore economico dell'oggetto	Costo, importo	Prodotto, Accessorio, Componente Base, Valuta
Descrizione	E' la descrizione in linguaggio naturale del prodotto	Definizione	Prodotto, Accessorio, Componente Base
Proprietà tecniche	Descrizione delle caratteristiche del prodotto	Caratteristiche tecniche	Prodotto, Accessorio, Componente Base, Materiale
Immagine	E' la rappresentazione grafica del prodotto	Disegno, Foto	Prodotto, Accessorio, Componente Base
Lingua	Lingua del catalogo	Idioma	Prodotto, Accessorio, Componente Base
Valuta	Unità di scambio che ha lo scopo di facilitare il trasferimento di beni	Moneta	Prezzo, Lingua

Materiale	I materiali sono in generale sostanze fisiche utilizzate nella produzione di oggetti.	Materia, Sostanza	Prodotto, Accessorio, Componente Base, Proprietà tecniche
-----------	---	-------------------	---

Figura 11 – Glossario dei termini

Naturalmente vanno raccolte anche le specifiche sulle operazioni da fare su questi dati e informarci anche sulla frequenza con la quale le varie operazioni vanno eseguite.

Nel nostro caso dobbiamo distinguere in due tipi di operazioni:

- Importazione dei dati dal database cliente;
- Selezione dei dati per la creazione del file di testo.

In entrambi i casi le operazioni verranno svolte una sola volta per ogni catalogo prodotto.

Come possiamo immaginare la fase più impegnativa sarà l'importazione dei dati che implicherà, oltre allo studio e alla preparazione dei prodotti anche l'elaborazione delle caratteristiche tecniche in modo da facilitare la successiva selezione dei dati.

L'accento dunque si pone maggiormente sulla capacità del sistema di fronteggiare le eterogeneità dei database dei clienti piuttosto che sulle performance del sistema.

3.4 Le entità e le relazioni

3.4.1 I prodotti e le famiglie

Nella raccolta dei requisiti mi sono accorta che ogni prodotto appartiene ad una serie che a sua volta può appartenere ad una

famiglia di prodotti e ho pensato di rappresentare la struttura dei prodotti di ogni cliente come un albero; in questo modo per ogni prodotto possiamo costruire un percorso univoco dalla foglia alla radice dell'albero.

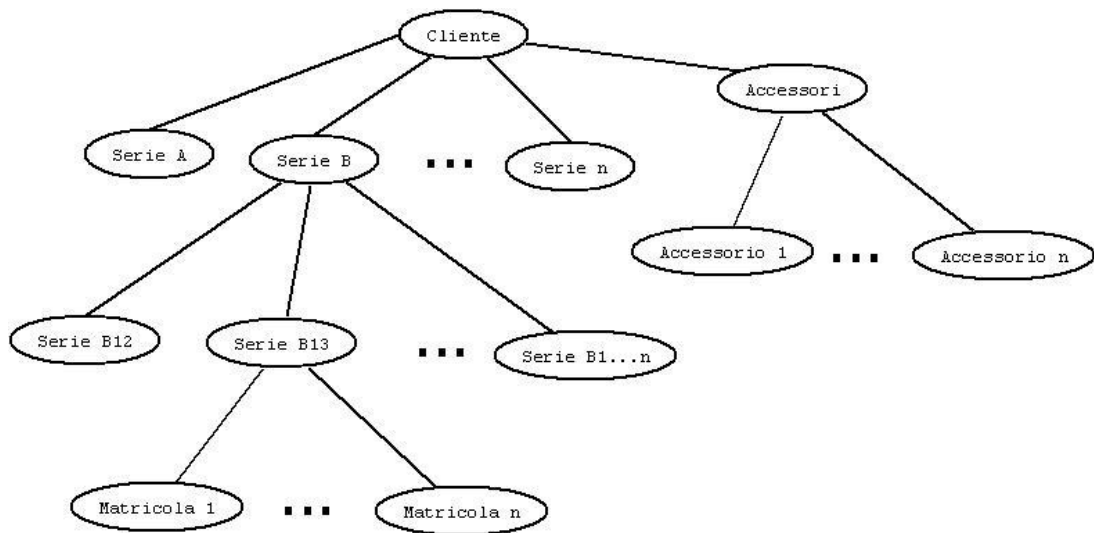


Figura 12 - Albero delle famiglie

Per riuscire a tradurre l'albero delle famiglie in uno schema UML⁵ troviamo utile una relazione ricorsiva, ovvero una relazione tra una entità e se stessa. In questo caso occorre anche definire i ruoli di partecipazione, ovvero padre e figlio.

⁵ [BRJ98]

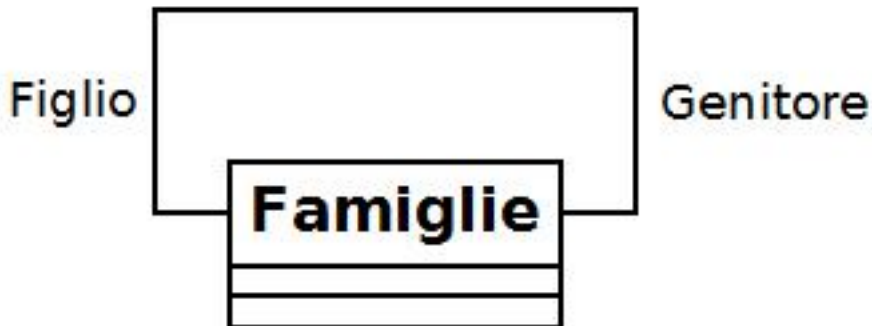


Figura 13 - UML delle famiglie

Le foglie di questo albero possono essere suddivise in 3 macro-entità:

- Prodotti finiti (sono i prodotti principali che il cliente produce e vende);
- Accessori (sono componenti secondari, che possono essere un'aggiunta al prodotto finito);
- Componenti base (sono i pezzi base che compongono un prodotto finito).

Ho pensato che *Componente Base*, *Prodotto Finito* e *Accessorio* potessero essere generalizzati in *Oggetto* dato che ognuno di essi può essere considerato un pezzo vendibile dall'azienda.

Le *generalizzazioni* rappresentano legami logici tra una classe genitore e una o più classi figlie. La classe genitore è più generale e comprende le figlie come caso particolare. Le generalizzazioni possono essere totali o parziali, esclusive o sovrapposte. Nel nostro caso ho utilizzato una generalizzazione totale ed esclusiva, perché ogni occorrenza dell'entità genitore *Oggetto* è una occorrenza di almeno una delle entità figlie e un'entità figlia non può essere contemporaneamente Prodotto Finito, Componente base o accessorio.

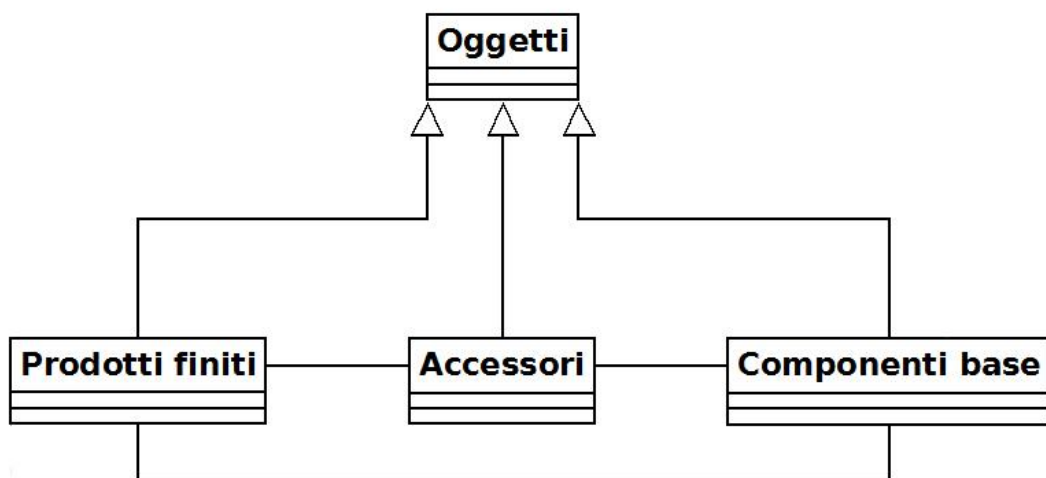


Figura 14 - Generalizzazione di Prodotti finiti, Accessori e Componenti Base in Oggetti

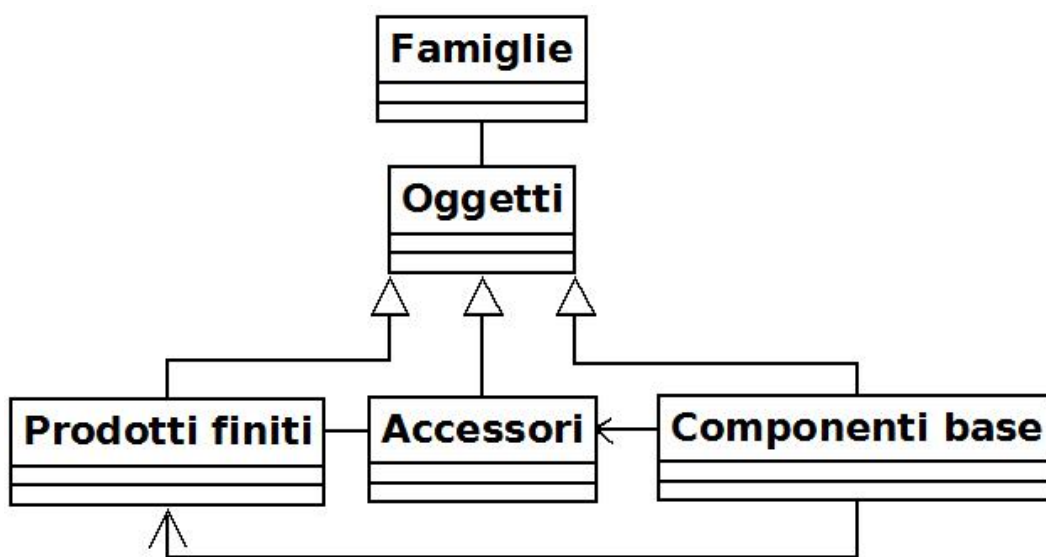


Figura 15 - Famiglie di Prodotti

3.4.2 Varianti, proprietà tecniche e descrizioni

Una volta definito il prodotto bisogna affrontare il concetto di variante; infatti dal punto di vista della produzione un prodotto può essere fisicamente realizzato con caratteristiche diverse rimanendo concettualmente identico. Ad esempio, un maglione può essere commercializzato in taglie e colori diversi che ne rappresentano le varianti.

Lo stesso ragionamento vale sia per accessori che per componenti base: risulta quindi evidente che il concetto di variante è di pertinenza dell'entità oggetto.

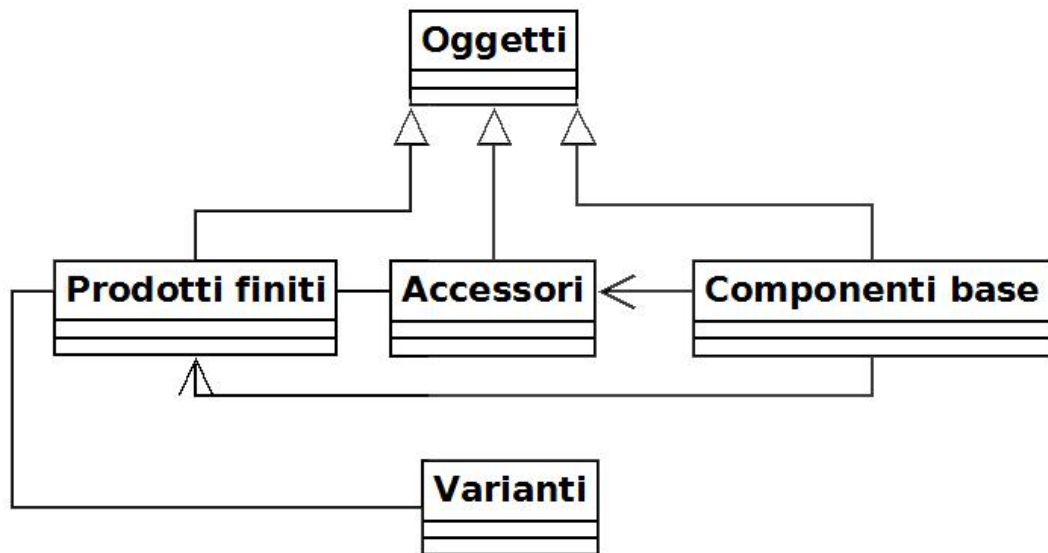


Figura 16 - UML di Varianti

In base a questa classificazione si ritiene che una parte delle caratteristiche riguardanti un prodotto, quali alcune proprietà tecniche e alcune descrizioni, debbano essere associati alla variante.

Il compito principale di un catalogo, oltre alla descrizione dei prodotti dell'azienda, è la descrizione delle caratteristiche dei prodotti.

Tutti questi dati, che possono variare dalla portata dell'acqua per determinate pompe idrauliche a caratteristiche di sicurezza delle serrature, a classificazione di trasmittanza termica per le finestre, sono stati riassunti nelle proprietà tecniche.

Avremo quindi un'associazione con ognuna delle entità individuate fino ad ora.

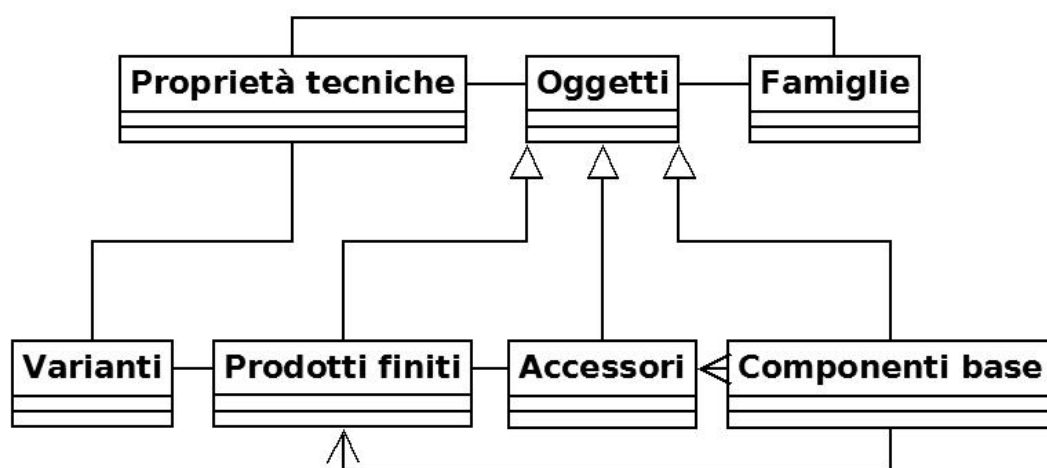


Figura 17 - UML delle Proprietà tecniche

Si ritiene opportuno mettere in relazione con le proprietà tecniche anche l'entità famiglia in quanto esistono delle caratteristiche comuni a tutti i membri della stessa famiglia, che a livello di presentazione possono essere necessarie agli impaginatori ad un livello superiore rispetto a quello di singolo prodotto.

Una proprietà tecnica più rilevante di altre, è stata individuata nei materiali; spesso infatti sono i produttori che la identificano come caratteristica particolare poiché il suo valore ha effetti collaterali anche su livello di presentazione a carico degli impaginatori.

I materiali vengono identificati come proprietà tecnica a sé stante in quanto la differenza di materiale in fase di costruzione di un prodotto può creare una variante del prodotto stesso e influenzare le politiche di prezzo dell'azienda cliente. Ho riscontrato, in altri casi, che a parità di

proprietà tecniche il materiale con il quale viene costruito l'oggetto può creare differenti famiglie di prodotti.

Anche in questo caso si ritiene opportuno consentire di associare un materiale specifico anche alla famiglia di prodotti, per garantire una corretta versatilità a livello di presentazione.

Inoltre ad un materiale possono essere associate delle proprietà tecniche specifiche che è bene rendere disponibili ai livelli superiori: questo costituisce una ulteriore motivazione per dare al concetto di materiale la dignità di entità.

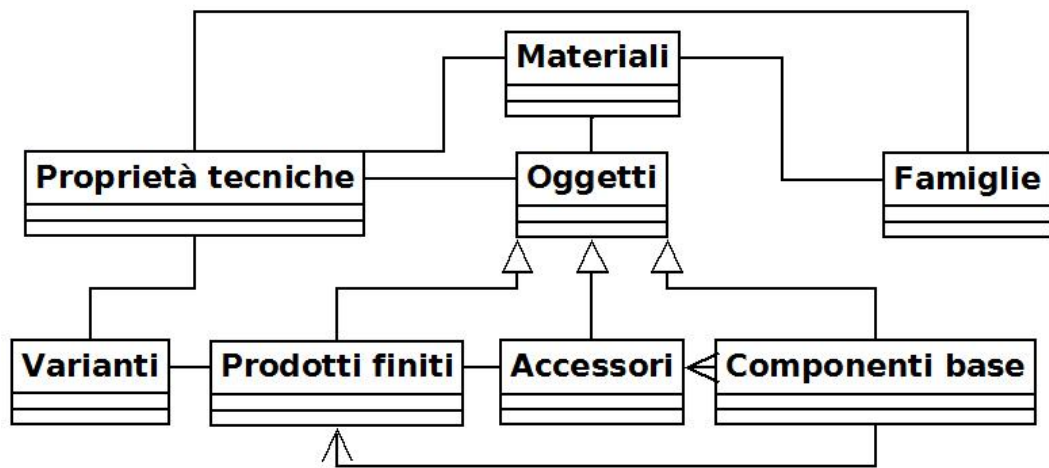


Figura 18 - UML dei materiali

Un'altra componente molto importante riscontrata in ogni catalogo e in ogni base di dati sono le descrizioni. Infatti ogni prodotto necessita di una descrizione che spieghi in linguaggio naturale l'oggetto in questione; in fondo il catalogo finale deve essere letto da esseri umani, e le descrizioni testuali consentono all'azienda cliente di attuare le opportune politiche di marketing.

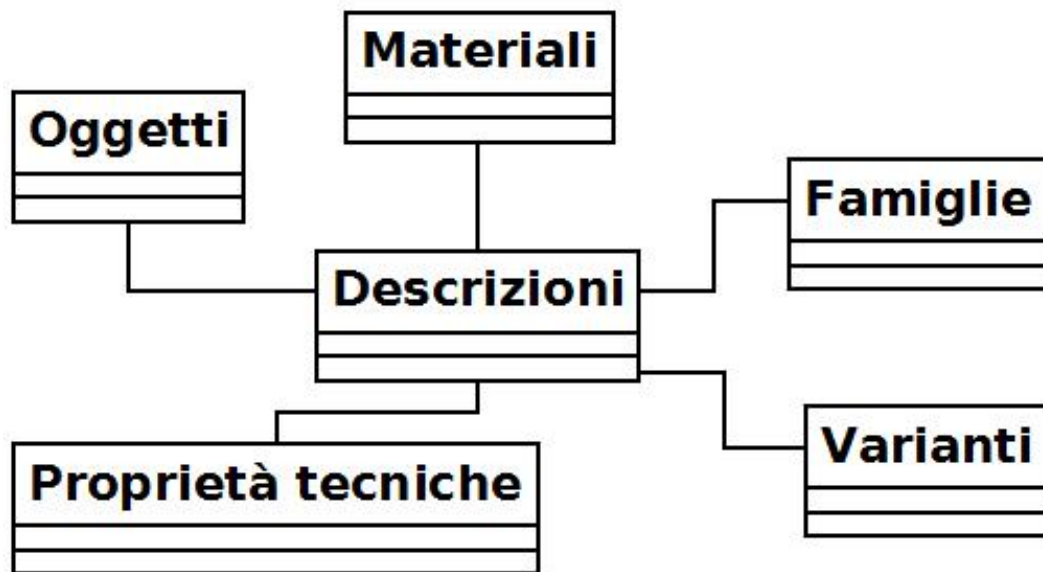


Figura 19 - UML delle descrizioni

3.4.3 Lingue e internazionalizzazione

Un'altra caratteristica comune dei cataloghi da me visionati è l'internazionalizzazione. Infatti a pensarci bene, ai giorni nostri, ogni azienda mette a disposizione dei suoi clienti stranieri dei cataloghi a loro di più facile comprensione; questo implica che avremo cataloghi multilingua (2 o 3 lingue raggruppate per catalogo), e dei cataloghi completamente tradotti in lingua straniera ma che mantengono la stessa grafica, la stessa impaginatura, e soprattutto gli stessi contenuti. Per le lingue bisogna tenere conto soprattutto dei nuovi mercati emergenti, come Est-Europa, Russia, Asia e Oriente che comportano problemi di codifica specifici utilizzando alfabeti diversi, quali il cirillico, i pittogrammi cinesi e thailandesi e la calligrafia araba.



Figura 20 - UML di lingue

3.4.4 La gestione delle risorse

Un'altra caratteristica essenziale di ogni catalogo, sono le immagini che permettono di rappresentare visivamente l'oggetto descritto.

Anche ogni famiglia può essere rappresentata da un'immagine, così come ogni variante del prodotto.

All'interno di ogni catalogo possono essere inserite immagini dei prodotti e diagrammi che semplificano la descrizione delle proprietà tecniche. Attualmente quindi, essendo i cataloghi distribuiti prevalentemente in forma cartacea, le risorse multimediali si limitano al formato grafico. Non è da escludere che in futuro si diffondano nuove tipologie di catalogo; ad esempio già al giorno d'oggi viene utilizzato il QR code che attraverso una semplice immagine bicolore consente di reperire ulteriori documenti multimediali.

Per questo motivo si ritiene opportuno non legare la risorsa ad una singola rappresentazione grafica, ma mantenere il concetto ad un livello di astrazione più alto in grado di mappare risorse multimediali eterogenee.

L'impiego di risorse multimediali potrebbe riguardare ogni aspetto del catalogo, ad esempio potrebbero contenere filmati che mostrano il funzionamento di un oggetto, presentazioni multimediali (Flash) che pubblicizzano il brand aziendale, file audio che guidano l'utilizzatore nella scelta della famiglia dei prodotti più opportuna.

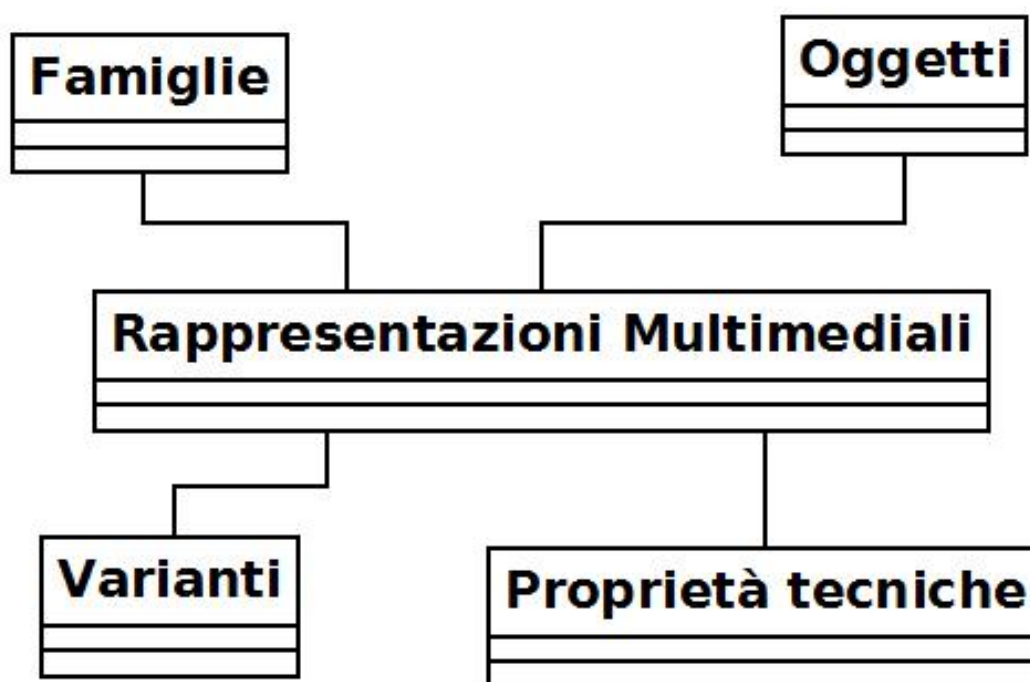


Figura 21 - UML delle rappresentazioni multimediali

3.4.5 Prezzi e scontistica

Ho ritenuto opportuno inserire nella mia analisi anche i prezzi e la relativa scontistica perché esistono tipi di cataloghi (i listini) che contengono al loro interno anche i prezzi relativi agli oggetti descritti. Sono un esempio tipico di listini quelli consegnati ai rivenditori, che contengono una parte descrittiva dei prodotti in maniera da poter mostrare il catalogo alla propria clientela e una parte riservata ai

rivenditori che contiene i prezzi, la scontistica ed eventuali altri dati che si preferisce non vengano mostrati alla clientela: si tratta della tipica situazione business to business.

Ovviamente nell'ottica della creazione di cataloghi multilingua bisognerà tenere conto anche delle valute utilizzate nelle varie nazioni; non si deve tenere conto però del cambio di valuta, problema gestito direttamente dall'ufficio commerciale del cliente, che può attuare politiche ad hoc in base alla nazione di vendita.



Figura 22 - UML di prezzo

3.4.6 Il problema dell'ordinamento

A livello di presentazione è importante che gli elementi siano ordinati secondo una politica decisa dall'ufficio marketing del cliente.

Tale ordinamento non è unico per tutto il catalogo, e non è necessariamente riconducibile a una logica nota, come l'ordinamento alfabetico.

Il problema dell'ordinamento coinvolge sia l'aspetto produttivo, dove oggetti e famiglie devono essere presentati secondo politiche precise, che l'aspetto marketing, dove i testi e le rappresentazioni multimediali devono seguire rigidamente le specifiche imposte.

Prendiamo ad esempio il caso in cui il prodotto da pubblicizzare appartiene alla telefonia mobile. In alcuni casi le immagini di dettaglio mostreranno prima l'oggetto intero, poi il monitor poi la tastiera e infine il retro; nel caso in cui tuttavia sul retro sia presente un elemento di design, come un decoro, probabilmente la foto del retro sarà la prima ad essere mostrata.



Figura 23 - Differenze di marketing

4. Progettazione

In questo capitolo inizia la progettazione vera e propria della nostra base di dati. Decideremo quali sono gli attributi di ogni entità e le cardinalità delle relazioni; ci occuperemo inoltre delle generalizzazioni, delle molteplicità e infine della normalizzazione.

Il risultato di questo capitolo sarà un modello relazionale migliorato con il quale procedere all'implementazione della nostra base di dati.

4.1 Gli identificatori

Gli identificatori⁶ principali sono essenziali nella traduzione verso il modello relazionale perché le chiavi in quest'ultimo modello sono utilizzate per stabilire legami tra dati in relazioni diverse. Inoltre le basi di dati richiedono di specificare una chiave primaria sulla quale vengono costruite automaticamente le strutture ausiliarie per il reperimento efficiente dei dati. Esistono diversi criteri per decidere quali saranno gli identificatori principali:

- Gli attributi con valore nullo non possono essere identificatori principali perché non garantiscono l'accesso a tutte le occorrenze delle entità;
- E' da preferire un identificatore composto da uno o pochi attributi perché facilita la ricerca dei dati, permette un risparmio di memoria nella realizzazione dei legami logici e facilita le operazioni di join;
- Si preferisce sia un identificatore interno piuttosto che uno esterno per gli stessi motivi sopra spiegati;
- Si preferisce un identificatore che viene utilizzato da molte operazioni, in questo modo queste operazioni possono essere eseguite efficientemente.

Se nessun identificatore soddisfa tutti i requisiti appena descritti conviene introdurre un ulteriore attributo (*codice*) all'entità generato

⁶ [ATZ09]

appositamente per identificare le occorrenze delle entità.
Questa scelta ci tornerà utile nella normalizzazione.

4.2 Il problema dell'ordinamento

L'ordinamento, di cui si è già parlato al paragrafo 3.4.6, riguarda entità omogenee; per questo può essere affrontato all'interno di ogni singola classe.

Essendo quindi una caratteristica intrinseca alle singole istanze può essere mappata come un attributo, la cui natura esatta verrà stabilita in fase di implementazione.

4.3 I prodotti e le famiglie

Per quanto riguarda le famiglie, come già anticipato nei paragrafi iniziali, esse saranno caratterizzate da un identificatore univoco ID e da un attributo di ordinamento ORDINAM.

L'ID da solo non è sufficiente per mantenere la tracciabilità tra la codifica aziendale e la codifica interna, infatti le aziende per le famiglie usano una nomenclatura basata sulle matricole; tuttavia non forniscono garanzie sull'univocità di tale codifica: per questo MATRICOLA viene trattato come un attributo a sé stante.

L'attributo NOME conterrà in linguaggio naturale il nome della famiglia: questa informazione non è necessaria al corretto funzionamento del sistema, ma aiuta gli impaginatori ad orientarsi tra i dati.

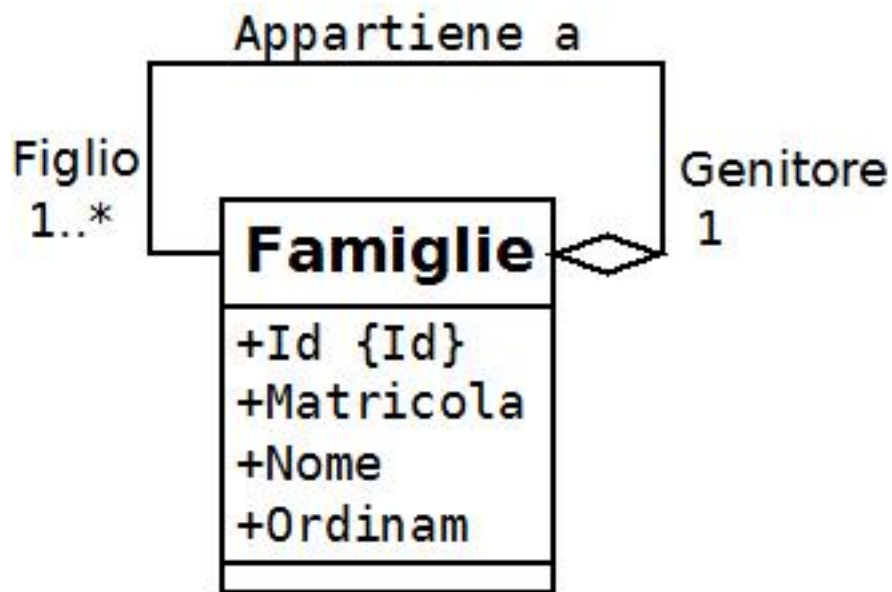


Figura 24 - Famiglie

Occupiamoci ora dei prodotti.

Dato che i modelli tradizionali, tra cui il modello relazionale, non consentono di rappresentare direttamente una generalizzazione⁷ è necessario trasformarla in entità e in associazioni. Per fare ciò esistono tre alternative:

- Accorpamento delle figlie della generalizzazione nel genitore;
- Accorpamento del genitore della generalizzazione nelle figlie;
- Sostituzione della generalizzazione con associazioni.

Per eliminare la nostra generalizzazione ho deciso di utilizzare la seconda alternativa. E' possibile utilizzare questa procedura solo se la generalizzazione è totale. In questo caso abbiamo un risparmio di memoria rispetto alla prima alternativa e una riduzione degli accessi rispetto alla terza scelta.

L'entità genitore viene eliminata e per la proprietà dell'ereditarietà i suoi attributi, il suo identificatore e le relazioni a cui tale entità

⁷ [ATZ09]

partecipava vengono aggiunti alle entità figlie.

Ho trasformato la generalizzazione nel seguente schema:

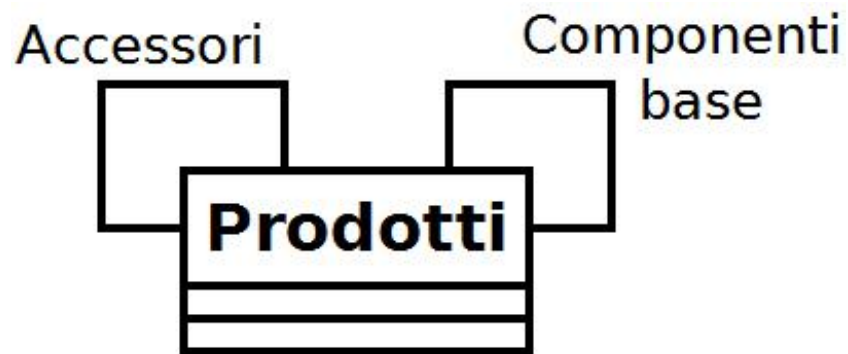


Figura 25 - Eliminazione della generalizzazione

Per quanto riguarda gli attributi ho proceduto analogamente alla famiglie, individuando un identificativo univoco ID, una MATRICOLA, un NOME e un ORDINAMENTO.

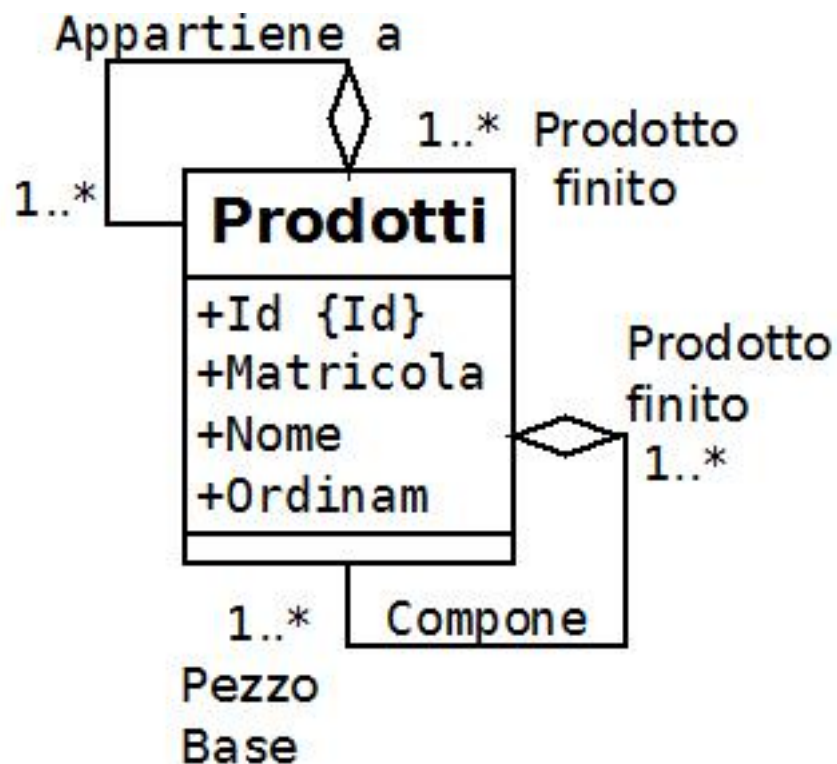


Figura 26 - Prodotti

4.4 Varianti, proprietà tecniche e descrizioni

Discorso analogo ai prodotti per le varianti, che avranno ID come identificativo univoco, avranno un NOME e un attributo ORDINAM. Solitamente le varianti non sono dotate di matricola perché è sufficiente quella del prodotto di riferimento; tuttavia alcuni clienti assegnano codici diversi anche alle varianti di prodotto: per questo si è ritenuto opportuno tenere anche a questo livello l'attributo MATRICOLA.

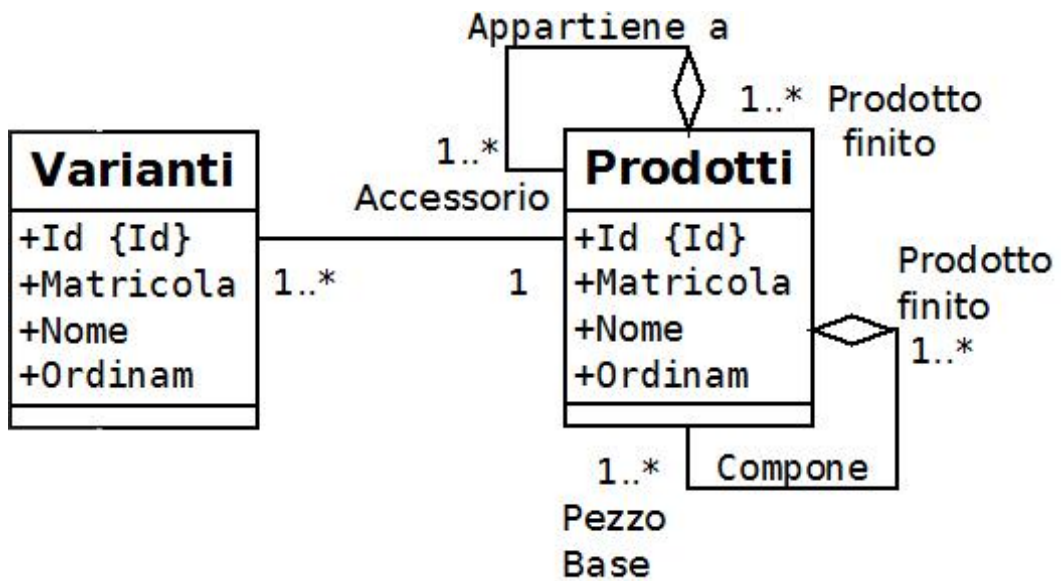


Figura 27 – Varianti

Ho identificato anche le proprietà tecniche tramite un ID univoco, e ho inserito anche l'attributo ORDINAM.

Le proprietà tecniche sono identificate da una coppia TIPO-VALORE dove TIPO rappresenta il nome della proprietà e VALORE il quantitativo numerico. Tale quantitativo può essere corredato da una specifica unità di misura denominata UDM.

Per mantenere una più completa versatilità è necessario non vincolare VALORE ad un formato numerico, ma consentire anche l'introduzione di espressioni testuali; a esempio la proprietà tecnica "consigliato ad un'utenza" spazia sul dominio dei valori ["principiante", "medio", "esperto"].

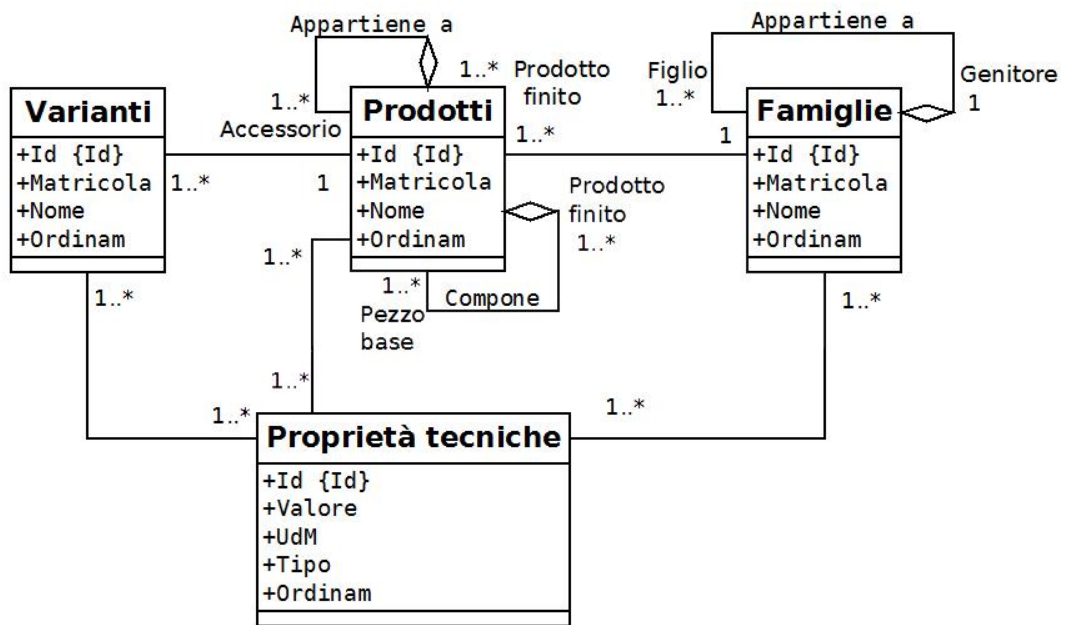


Figura 28 - Proprietà tecniche

Anche i materiali verranno identificati con un attributi ID univoco, un NOME e l'attributo ORDINAM.

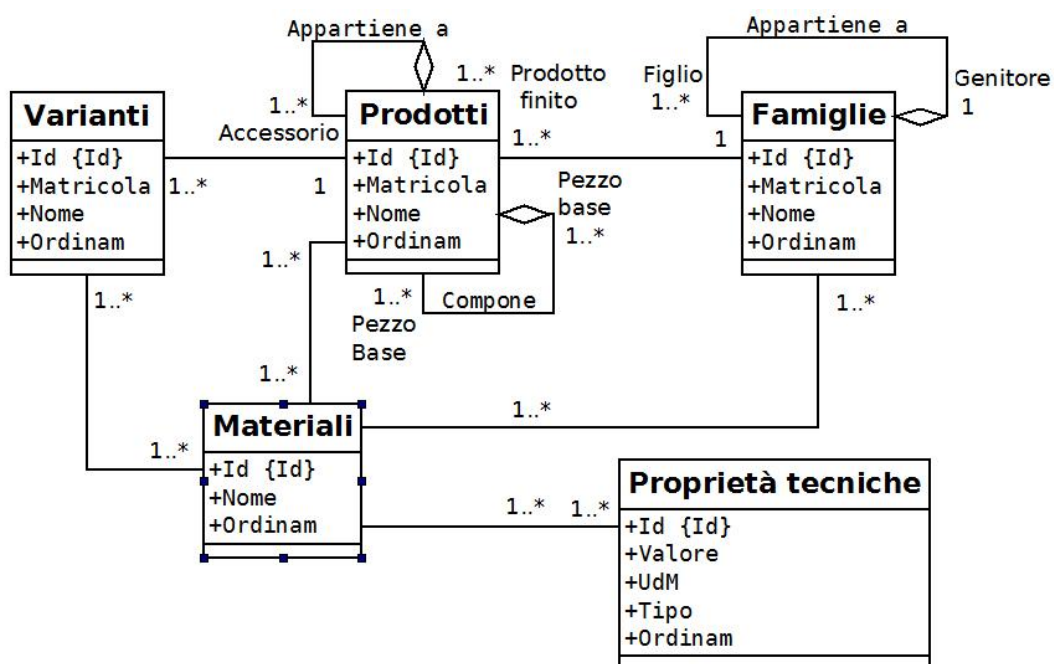


Figura 29 – Materiali

Le descrizioni saranno identificate anche loro da un ID univoco, conterranno un attributo TESTO che conterrà fisicamente il testo che vogliamo.

Ho inserito anche un attributo FUNZIONE che consentire di identificare il contesto a cui appartiene la descrizione. Ad esempio nella base di dati potremmo inserire anche le descrizioni della parte iniziale di ogni catalogo indicando nell'attributo FUNZIONE la tipologia della descrizione. Sarà così più facile indicare in quale parte del catalogo si dovrà posizionare un determinato testo.

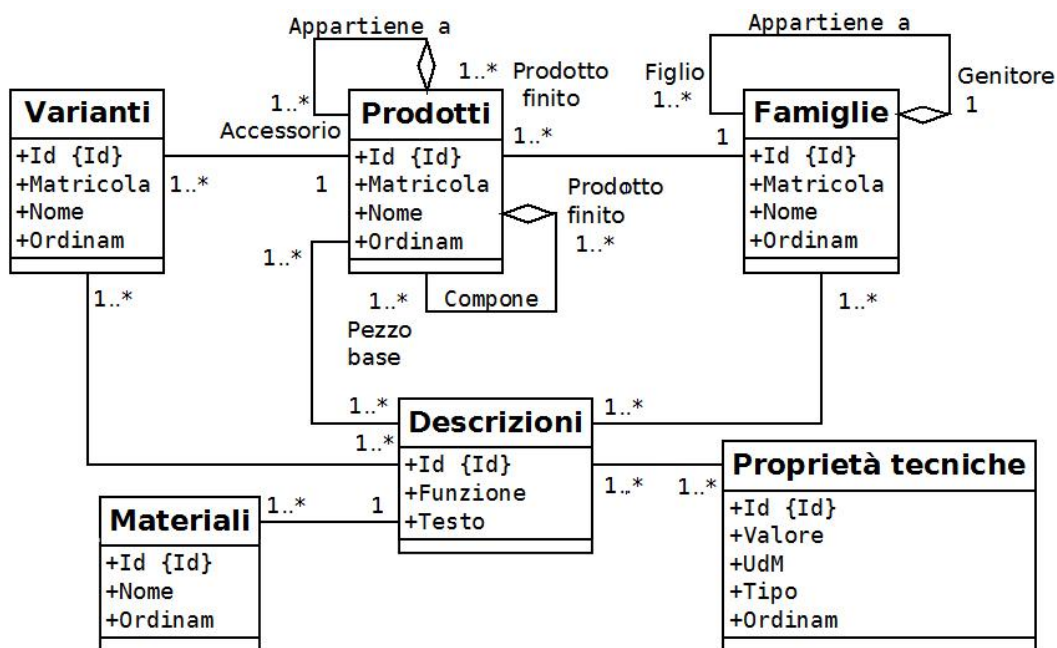


Figura 30 - Descrizioni

4.5 Lingue e internazionalizzazione

Analizziamo l'entità nazioni. Ovviamente ogni nazione dispone di un nome proprio che la identifica, purtroppo però nelle varie lingue il nome di ogni nazione risulta essere differente; ho pensato allora di identificare ogni nazione in maniera univoca tramite un ID a cui verrà associato un attributo STATO. Questa informazione non è necessaria al corretto funzionamento del sistema, ma aiuta gli sviluppatori a capire meglio a quale nazione si stanno riferendo.

Ho inserito anche un attributo VALUTA che conterrà la valuta ufficiale di ogni stato, l'attributo SIMBOLO che permetterà di individuare il simbolo utilizzato per indicare la valuta e l'attributo CODCORRVALUTA che è un modo univoco per indicare ogni valuta. Ad esempio la valuta in uso in Italia è chiamata comunemente

EURO, il suo simbolo è €, ma a livello internazionale è identificata come EUR.

Nell'entità lingue ho inserito l'attributo LINGUA che descrive la lingua parlata in una determinata nazione e l'ho identificata in maniera univoca tramite l'identificativo ID.

Ho inserito anche l'attributo ABBREVIAZIONE per rendere più omogenea la realizzazione dei cataloghi multilingua, in quanto a volte la lingua italiana viene abbreviata come IT e altre come ITA.

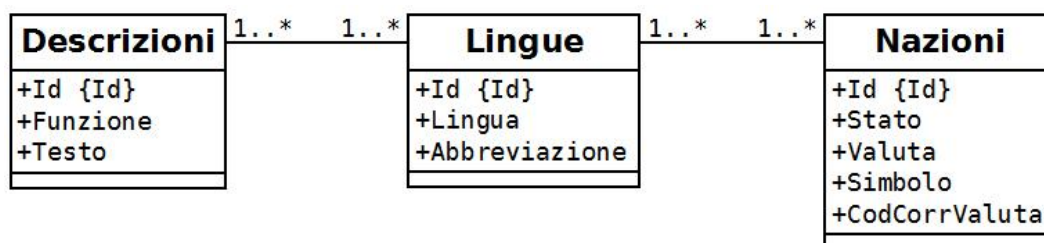


Figura 31 - Lingue e nazioni

4.6 La gestione delle risorse

Le rappresentazioni multimediali saranno identificate da un ID univoco, un attributo NOME che conterrà il nome del file di cui vogliamo tenere traccia al quale ho associato un attributo PATH che indica il percorso in cui reperire i dati e un attributo TYPE che mi descriverà il tipo di file. Ad esempio l'immagine di un prodotto potrà essere in un file .JPEG, in un file .CAD, ecc.

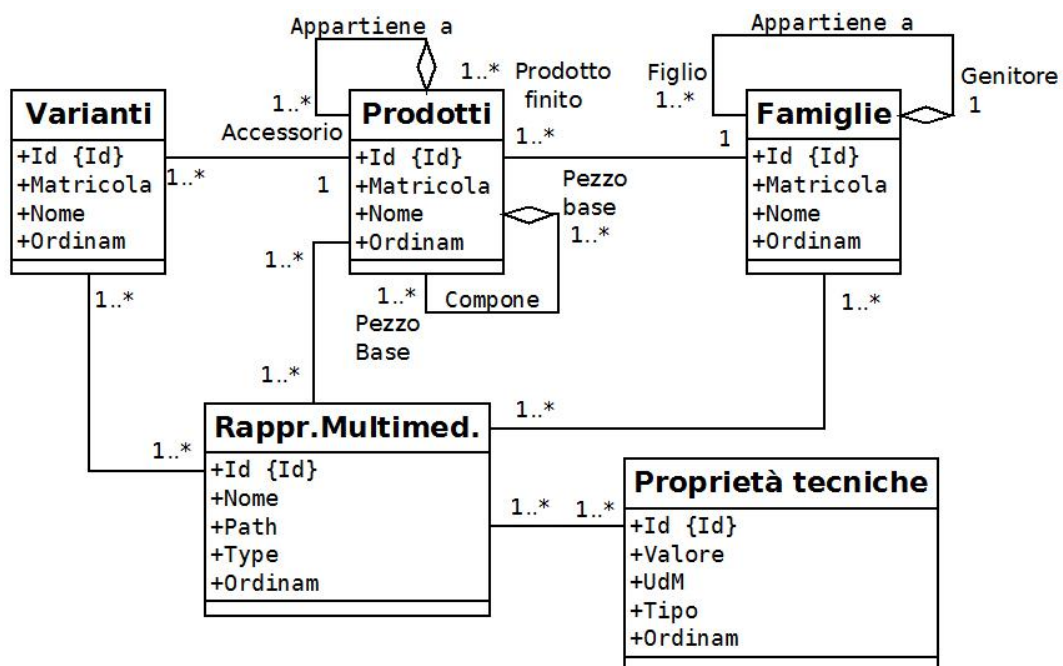


Figura 32 - Rappresentazioni multimediali

4.7 Prezzi e scontistica

All'entità prezzi assegno due attributi: l'identificativo univoco ID e l'attributo VALORE.

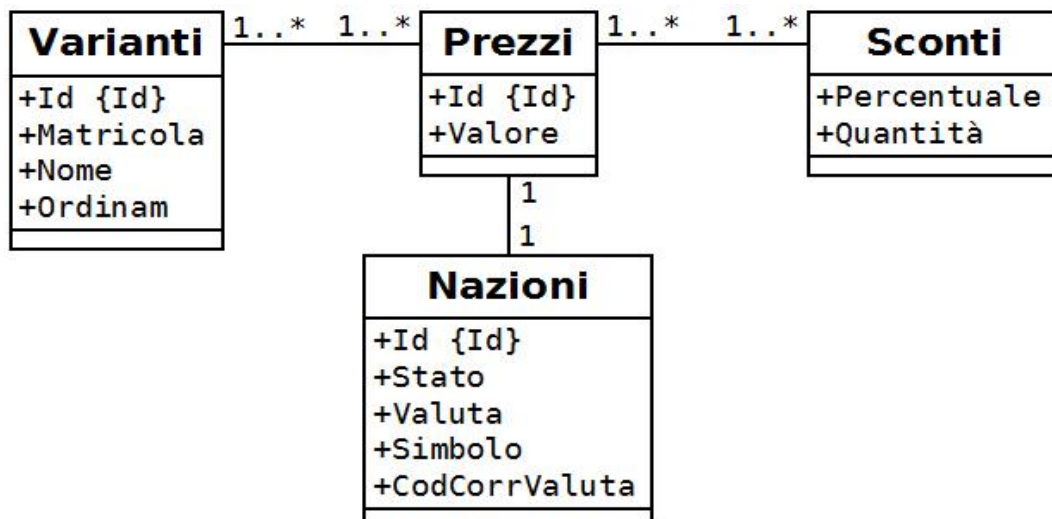


Figura 33 - Prezzi e scontistica

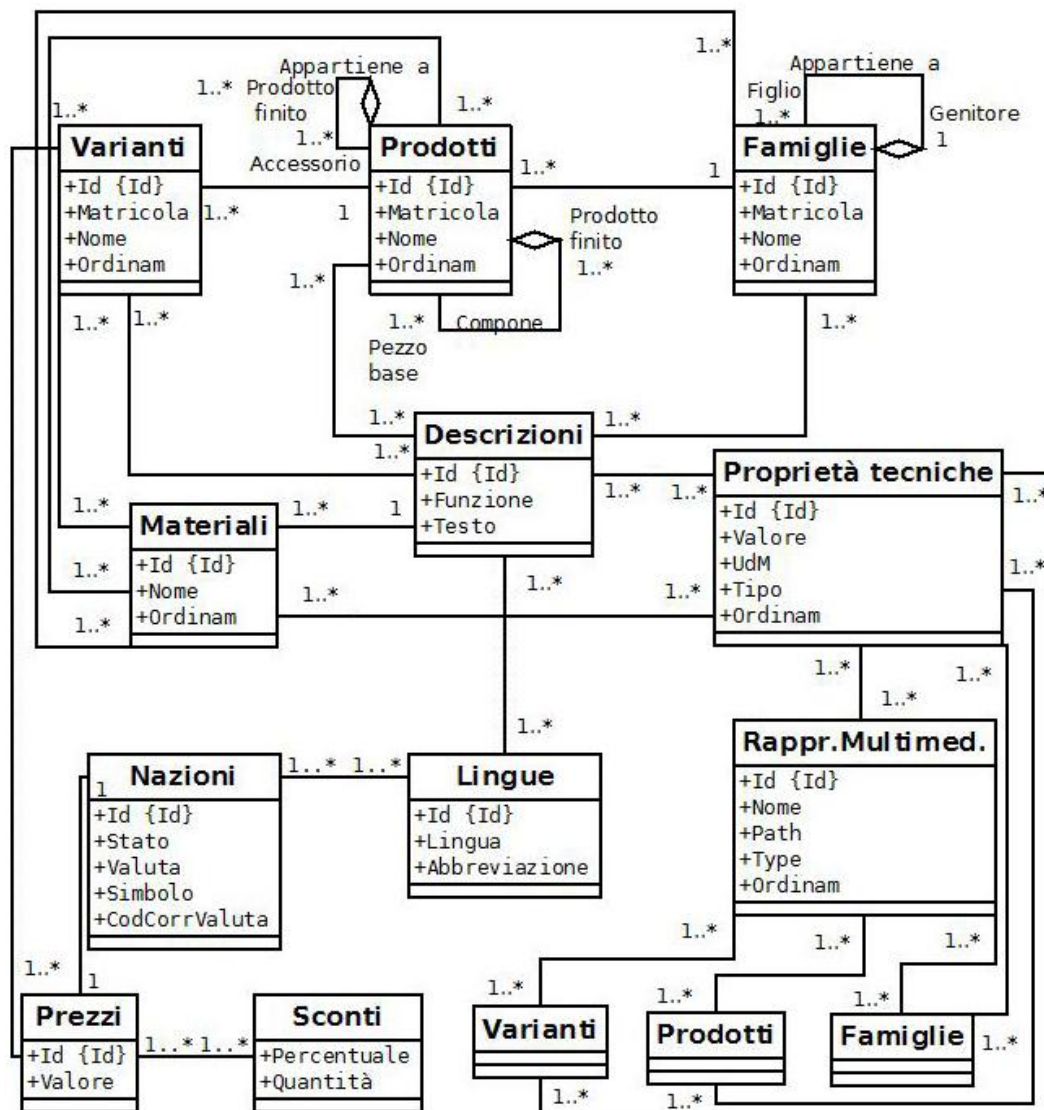


Figura 34 - Schema UML finale⁸

⁸ Data la complessità dello schema UML completo, ho preferito duplicare le entità Varianti, Prodotti e Famiglie per una migliore comprensione.

4.8 Integrità referenziale

L'integrità referenziale⁹ (foreign key) è un insieme di regole del modello relazionale che garantiscono l'integrità dei dati quando si hanno relazioni associate tra loro attraverso la chiave esterna: queste regole servono per rendere valide le associazioni tra le tabelle e per eliminare gli errori di inserimento, cancellazione o modifica di dati collegati tra loro.

L'integrità referenziale viene rispettata quando per ogni valore non nullo della chiave esterna, esiste un valore corrispondente della chiave primaria nella tabella associata.

Quando viene applicata l'integrità referenziale, è necessario osservare le seguenti regole pratiche:

- Non è possibile immettere un valore nella chiave esterna della tabella associata, se tale valore non esiste tra le chiavi della tabella primaria.
- Non è possibile eliminare una n-pla dalla tabella primaria, se esistono righe legate ad essa attraverso la chiave esterna nella tabella correlata
- Non si può modificare, come è ovvio, il valore della chiave nella tabella primaria, se ad essa corrispondono righe nella tabella correlata.

Ho individuato il problema dell'integrità referenziale in quelle entità che sono in relazione con prodotti, varianti, famiglie e accessori, ad esempio proprietà tecniche, rappresentazioni, prezzi, ecc.

Ho individuato due possibili soluzioni:

Soluzione 1: nel caso volessi mantenere l'integrità referenziale a livello della base di dati potrei assicurare l'integrità realizzando per ogni tabella che ha come riferimento Prodotti, Accessori, Famiglie,

⁹ [ATZ09]

Varianti, delle ulteriori tabelle in modo da mappare tutte le possibili combinazioni.

Prendiamo come esempio la tabella Rappresentazione.

Dovrei creare 4 distinte tabelle:

- Rappresentazione_Prodotti,
- Rappresentazione_Famiglie,
- Rappresentazione_Accessori,
- Rappresentazione_Varianti.

In questo modo, oltre a dover gestire un elevato numero di tabelle all'interno della base di dati, avrei come difetto la ridondanza dei dati, cosa che assolutamente è sconsigliata.

Soluzione 2: un altro modo per mantenere l'integrità referenziale è a livello applicativo; questo implica la creazione di una tabella indipendente, quindi senza vincoli di relazione con Prodotti, Famiglie, Accessori e Varianti, all'interno della quale verranno catalogati gli ID di ogni tabella con il relativo nome_tabella. In questa soluzione il vincolo di integrità è garantito dall'applicativo che si occuperà della gestione del database.

Intendo utilizzare quest'ultima soluzione nella realizzazione della mia base di dati.

4.9 Traduzione verso il modello relazionale

Nel trasformare i miei diagrammi UML di progettazione in un modello relazionale¹⁰ devo tradurre anche le associazioni in modo che le loro informazioni vengano inserite nelle tabelle.

In questa fase mi occupo, quindi, dei vincoli interrelazionali che definiscono legami tra due o più tabelle. Questo compito è eseguito dalla chiave esterna che identifica una colonna o un insieme di colonne di una tabella che referencia una colonna o un insieme di

¹⁰ [ATZ09]

colonne di un'altra tabella. Ciò implica che un record nella tabella referenziante non può contenere valori che non esistono nella tabella referenziata (eccetto nel caso particolare di valori NULL).

Otengo così il seguente schema logico:

- Famiglie (Id, Matricola, Id_Padre, Nome, Ordinam)
- Prodotti (Id, Matricola, Nome, Ordinam)
- Accessori (Id, Matricola, Id_Ref, Nome, Ordinam)
- Composizioni (Id, Id_Pezzo, Id_PezzoBase, Quantita)
- Varianti (Id, Matricola, Id_Ref, Nome, Ordinam)
- Var_Acc (Id, Id_Acc, Id_Var)
- Tabelle (Id, Id_Ref, Nome Tabella)
- Proprietà (Id, Id_Ref, Id_Desc, Id_Mat, Id_Rap, Valore, UdM, Ordinam)
- Materiali (Id, Id_Ref, Id_Desc, Ordinam)
- Descrizioni (Id, Id_Ref, Funzione, Testo)
- Rappresentazioni (Id, Id_Tab, Nome, Path, Type, Ordinam)
- Nazioni (Id, Stato, Valuta, Simbolo, CodCorrenteValuta)
- Lingue (Id, Lingua, ABB)
- Nazioni_Lingue (Id, Id_Lingua, Id_Nazione)
- Prezzi (Id, Id_Ref, Valore, Valuta, Percentuale, Quantita)
- Traduzioni (Id, Id_Desc, Id_Lingua, Testo)

4.10 La normalizzazione

La normalizzazione¹¹ è un procedimento volto all'eliminazione delle ridondanze e del rischio di incoerenza del database. Questo concetto può essere utilizzato per effettuare controlli di qualità di basi di dati e costituisce per questo un utile strumento di analisi nell'ambito dell'attività di progettazione.

¹¹ [ATZ09]

Il processo si fonda su un semplice criterio: se una relazione presenta più concetti tra loro indipendenti, la si decompone in relazioni più piccole, una per ogni concetto. Questo tipo di processo non è sempre applicabile in tutte le tabelle, dato che in alcuni casi potrebbe comportare una perdita d'informazioni.

Il mio obiettivo è di raggiungere, attraverso raffinamenti del modello relazionale, una forma normale che sottostando ad una serie di vincoli, riduca o, meglio, elimini la possibilità di anomalie.

4.10.1 Dipendenze funzionali

Le dipendenze funzionali sono un particolare vincolo di integrità per il modello relazionale che descrive legami di tipo funzionale tra gli attributi e una relazione.

4.10.2 Forme normali

Le forme normali “certificano” la qualità dello schema di una base di dati relazionale. Le forme normali¹² sono le seguenti:

Prima Forma Normale: una base dati è in prima forma normale se vale il seguente vincolo per ogni relazione contenuta nella base dati. Una relazione è in 1NF se e solo se:

- non presenta gruppi di attributi che si ripetono (ossia ciascun attributo è definito su un dominio con valori atomici).
- esiste una chiave primaria (ossia esiste un insieme di attributi, che identifica in modo univoco ogni tupla della relazione)

Seconda Forma Normale: una base dati è invece in seconda forma normale quando è in 1NF e per ogni tabella tutti i campi non chiave dipendono funzionalmente dall'intera chiave composta e non da una parte di essa.

¹² [ATZ09]

Terza Forma Normale: una base dati è in terza forma normale se è in 2NF e per ogni dipendenza funzionale è vera una delle seguenti condizioni:

- X è una superchiave della relazione
- Y è membro di una chiave della relazione

Quindi, una relazione si dice in terza forma normale quando è innanzitutto in seconda forma normale e tutti gli attributi non-chiave dipendono soltanto dalla chiave, ossia non esistono attributi che dipendono da altri attributi non-chiave.

Forma Normale di Boyce e Codd: una relazione R è in forma normale di Boyce e Codd se e solo se è in terza forma normale e, per ogni dipendenza funzionale non banale, X è una superchiave per R. Quindi, dato un insieme di relazioni, non è possibile garantire sempre il raggiungimento di questa forma normale; in particolare il mancato raggiungimento di questo obiettivo è indice che la base dati è affetta da un'anomalia di cancellazione (ossia è possibile perdere dati a seguito di un'operazione di cancellazione).

Analizziamo gli schemi uno ad uno e vediamo se sono normalizzati.

Tutti gli schemi di relazione sono in prima forma normale in quanto tutti gli attributi sono atomici.

Famiglie (Id, Matricola, Id_Padre, Nome, Ordinam):

- 2FN: sì, perché la chiave dello schema è formata da un solo attributo, quindi non possono esserci dipendenze parziali dalla chiave.
- 3FN: sì, perché non ci sono dipendenze transitive dalla chiave.
- BCFN: sì, perché per ogni dipendenza funzionale $X \rightarrow Y$, X è superchiave dello schema.

Prodotti (Id, Matricola, Nome, Ordinam):

- 2FN: si, perché la chiave dello schema è formata da un solo attributo, quindi non possono esserci dipendenze parziali dalla chiave.
- 3FN: si, perché non ci sono dipendenze transitive dalla chiave.
- BCFN: si, perché per ogni dipendenza funzionale $X \rightarrow Y$, X è superchiave dello schema.

Varianti (Id, Matricola, Id_Ref, Nome, Ordinam):

- 2FN: si, perché la chiave dello schema è formata da un solo attributo, quindi non possono esserci dipendenze parziali dalla chiave.
- 3FN: si, perché non ci sono dipendenze transitive dalla chiave.
- BCFN: si, perché per ogni dipendenza funzionale $X \rightarrow Y$, X è superchiave dello schema.

Accessori (Id, Matricola, Id_Ref, Nome, Ordinam):

- 2FN: si, perché la chiave dello schema è formata da un solo attributo, quindi non possono esserci dipendenze parziali dalla chiave.
- 3FN: si, perché non ci sono dipendenze transitive dalla chiave.
- BCFN: si, perché per ogni dipendenza funzionale $X \rightarrow Y$, X è superchiave dello schema.

Var_Acc (Id, Id_Acc, Id_Var):

- 2FN: si, perché la chiave dello schema è formata da un solo attributo, quindi non possono esserci dipendenze parziali dalla chiave.
- 3FN: si, perché non ci sono dipendenze transitive dalla chiave.
- BCFN: si, perché per ogni dipendenza funzionale $X \rightarrow Y$, X è superchiave dello schema.

Tabelle (Id, Id_Ref, Nome Tabella):

- 2FN: si, perché la chiave dello schema è formata da un solo attributo, quindi non possono esserci dipendenze parziali dalla chiave.
- 3FN: si, perché non ci sono dipendenze transitive dalla chiave.
- BCFN: si, perché per ogni dipendenza funzionale $X \rightarrow Y$, X è superchiave dello schema.

Composizioni (Id, Id_Pezzo, Id_PezzoBase):

- 2FN: si, perché la chiave dello schema è formata da un solo attributo, quindi non possono esserci dipendenze parziali dalla chiave.
- 3FN: si, perché non ci sono dipendenze transitive dalla chiave.
- BCFN: si, perché per ogni dipendenza funzionale $X \rightarrow Y$, X è superchiave dello schema.

Lingue (Id, Lingua , ABB):

- 2FN: si, perché la chiave dello schema è formata da un solo attributo, quindi non possono esserci dipendenze parziali dalla chiave.

- 3FN: si, perché non ci sono dipendenze transitive dalla chiave.
- BCFN: si, perché per ogni dipendenza funzionale $X \rightarrow Y$, X è superchiave dello schema.

Nazioni (Id, Stato, Valuta, Simbolo, CodCorrenteValuta):

- 2FN: si, perché la chiave dello schema è formata da un solo attributo, quindi non possono esserci dipendenze parziali dalla chiave.
- 3FN: si, perché non ci sono dipendenze transitive dalla chiave.
- BCFN: si, perché per ogni dipendenza funzionale $X \rightarrow Y$, X è superchiave dello schema.

Nazioni_Lingue (Id, Id_Lingua, Id_Nazione):

- 2FN: si, perché la chiave dello schema è formata da un solo attributo, quindi non possono esserci dipendenze parziali dalla chiave.
- 3FN: si, perché non ci sono dipendenze transitive dalla chiave.
- BCFN: si, perché per ogni dipendenza funzionale $X \rightarrow Y$, X è superchiave dello schema.

Traduzioni (Id, Id_Descr, Id_Lingua, Testo):

- 2FN: si, perché la chiave dello schema è formata da un solo attributo, quindi non possono esserci dipendenze parziali dalla chiave.
- 3FN: si, perché non ci sono dipendenze transitive dalla chiave.
- BCFN: si, perché per ogni dipendenza funzionale $X \rightarrow Y$, X è superchiave dello schema.

Descrizioni (Id, Id_Ref, Funzione, Testo)

- 2FN: sì, perché la chiave dello schema è formata da un solo attributo, quindi non possono esserci dipendenze parziali dalla chiave.
- 3FN: sì, perché non ci sono dipendenze transitive dalla chiave.
- BCFN: sì, perché per ogni dipendenza funzionale $X \rightarrow Y$, X è superchiave dello schema.

Rappresentazioni (Id, Id_Ref, Nome, Path, Type, Ordinam):

- 2FN: sì, perché la chiave dello schema è formata da un solo attributo, quindi non possono esserci dipendenze parziali dalla chiave.
- 3FN: sì, perché non ci sono dipendenze transitive dalla chiave.
- BCFN: sì, perché per ogni dipendenza funzionale $X \rightarrow Y$, X è superchiave dello schema.

Prezzi (Id, Id_Ref, Valore, Valuta, Percentuale, Quantita)

- 2FN: sì, perché la chiave dello schema è formata da un solo attributo, quindi non possono esserci dipendenze parziali dalla chiave.
- 3FN: sì, perché non ci sono dipendenze transitive dalla chiave.
- BCFN: sì, perché per ogni dipendenza funzionale $X \rightarrow Y$, X è superchiave dello schema.

Proprietà (Id, Id_Ref, Id_Desc, Id_Mat, Id_Rap, Valore, UdM,

Ordinam):

- 2FN: si, perché la chiave dello schema è formata da un solo attributo, quindi non possono esserci dipendenze parziali dalla chiave.
- 3FN: si, perché non ci sono dipendenze transitive dalla chiave.
- BCFN: si, perché per ogni dipendenza funzionale $X \rightarrow Y$, X è superchiave dello schema.

Materiali (Id, Id_Ref, Id_Desc, Ordinam):

- 2FN: si, perché la chiave dello schema è formata da un solo attributo, quindi non possono esserci dipendenze parziali dalla chiave.
- 3FN: si, perché non ci sono dipendenze transitive dalla chiave.
- BCFN: si, perché per ogni dipendenza funzionale $X \rightarrow Y$, X è superchiave dello schema.

Il database è in forma normale Boyce e Codd.

5. Implementazione

Per il linguaggio con il quale realizzare la base di dati ho deciso di utilizzare SQL¹³ standard, in maniera da poter sviluppare successivamente il database con qualsiasi DBMS; questa scelta è motivata anche dal fatto che lo standard consente di modellare completamente la base di dati senza che costrutti non standard possano portare benefici in termini computazionali o di efficienza.

5.1 I prodotti e le famiglie

Come abbiamo visto nel paragrafo della progettazione gli attributi dell'entità famiglia sono:

- ID: come soluzione ottimale per l'identificativo univoco ho pensato a un tipo di dato intero auto incrementante che sarà così definito: "INT PRIMARY KEY IDENTITY";
- MATRICOLA: ho pensato che potesse essere rappresentato come un set di caratteri alfanumerici con una lunghezza abbastanza lunga a soddisfare i nomi di matricola di ogni possibile database cliente. Ho ritenuto possibile che una famiglia non disponga di una matrice quindi questo attributo può risultare anche NULL; matricola sarà così definito: "VARHCHAR(50)";
- NOME: questo attributo conterrà il nome in linguaggio umano della famiglia, quindi il tipo di dato ideale è VARCHAR con la possibilità di essere NULL; sarà così definito: "VARCHAR (MAX)"
- ORDINAM: ho pensato che la soluzione migliore per l'ordinamento fosse quello numerico con la possibilità di non essere presente, sarà definito come "INT".

¹³ [W3School]

La chiave esterna è:

- **ID_PADRE**: ha una relazione di riferimento con l'identificativo ID della tabella Prodotti, quindi il tipo di dato sarà definito allo stesso modo;

Nell'entità Prodotti gli attributi sono:

- **ID**: come soluzione ottimale per l'identificativo univoco ho pensato a un tipo di dato intero auto incrementante che sarà così definito: "INT PRIMARY KEY IDENTITY";
- **MATRICOLA**: ho pensato che potesse essere rappresentato come un set di caratteri alfanumerici con una lunghezza abbastanza lunga a soddisfare i nomi di matricola di ogni possibile database cliente. Ho ritenuto possibile che una famiglia non disponga di una matrice quindi questo attributo può risultare anche NULL; matricola sarà così definito: "VARCHAR(50)";
- **NOME**: questo attributo conterrà il nome in linguaggio umano della famiglia, quindi il tipo di dato ideale è VARCHAR con la possibilità di essere NULL; sarà così definito: "VARCHAR (MAX)";
- **ORDINAM**: ho pensato che la soluzione migliore per l'ordinamento fosse quello numerico con la possibilità di non essere presente, sarà definito come "INT".

L'entità Accessori ha i seguenti attributi:

- **ID**: come soluzione ottimale per l'identificativo univoco ho pensato a un tipo di dato intero auto incrementante che sarà così definito: "INT PRIMARY KEY IDENTITY";
- **MATRICOLA**: ho pensato che potesse essere rappresentato come un set di caratteri alfanumerici con una lunghezza abbastanza lunga a soddisfare i nomi di matricola di ogni

possibile database cliente. Ho ritenuto possibile che una famiglia non disponga di una matrice quindi questo attributo può risultare anche NULL; matricola sarà così definito: “VARCHAR(50)”;

- NOME: questo attributo conterrà il nome in linguaggio umano della famiglia, quindi il tipo di dato ideale è VARCHAR con la possibilità di essere NULL; sarà così definito: “VARCHAR (MAX)”;
- CATEGORIA: questo attributo spiega a quale tipologia di accessori appartiene il prodotto in questione, il tipo di dato ideale è VARCHAR con la possibilità di essere NULL; sarà così definito: “VARCHAR (MAX)”;
- ORDINAM: ho pensato che la soluzione migliore per l’ordinamento fosse quello numerico con la possibilità di non essere presente, sarà definito come “INT”.

La chiave esterna è:

- ID_REF: ha una relazione di riferimento con l’identificativo ID della tabella Varianti, quindi il tipo di dato, come abbiamo deciso sopra, è un intero, e sarà definito come “INT”.

L’entità Var_Acc ha i seguenti attributi:

- ID: come soluzione ottimale per l’identificativo univoco ho pensato a un tipo di dato intero auto incrementante che sarà così definito: “INT PRIMARY KEY IDENTITY”;

Le chiavi esterne sono:

- ID_ACC: ha una relazione di riferimento con l’identificativo ID della tabella Accessori, quindi il tipo di dato, come abbiamo deciso sopra, è un intero, e sarà definito come “INT”.

- ID_VAR: ha una relazione di riferimento con l'identificativo ID della tabella Varianti, quindi il tipo di dato, come abbiamo deciso sopra, è un intero, e sarà definito come "INT".

L'entità Composizioni ha i seguenti attributi:

- ID: come soluzione ottimale per l'identificativo univoco ho pensato a un tipo di dato intero auto incrementante che sarà così definito: "INT PRIMARY KEY IDENTITY";
- QUANTITA: questo attributo ci dice in che quantità un pezzo ne compone un altro, quindi la soluzione ideale per lui è un intero, sarà definito come "INT";

Le chiavi esterne sono:

- ID_Pezzo: ha una relazione di riferimento con l'identificativo ID della tabella Tabelle, quindi il tipo di dato, come abbiamo deciso sopra, è un intero, e sarà definito come "INT".
- ID_PezzoBase: ha una relazione di riferimento con l'identificativo ID della tabella Prodotti, quindi il tipo di dato, come abbiamo deciso sopra, è un intero, e sarà definito come "INT".

L'entità Tabella ha i seguenti attributi:

- ID: come soluzione ottimale per l'identificativo univoco ho pensato a un tipo di dato intero auto incrementante che sarà così definito: "INT PRIMARY KEY IDENTITY";
- NOME_TABELLA: questo attributo ci dice il nome della tabella a cui appartiene ID_REF e ho deciso di utilizzare un set di caratteri sarà così definito: "VARCHAR (MAX)";
- ID_REF: rappresenta l'identificativo degli ID delle tabelle Prodotti, Famiglie, Varianti, Accessori, quindi il tipo di dato,

come abbiamo deciso sopra, è un intero, e sarà definito come “INT”.

In questa tabella non ho potuto inserire delle chiavi esterne, l’integrità referenziale sarà mantenuta tramite software.

5.2 Varianti, proprietà tecniche e descrizioni

Nella tabella Varianti gli attributi sono:

- ID: come soluzione ottimale per l’identificativo univoco ho pensato a un tipo di dato intero auto incrementante che sarà così definito: “INT PRIMARY KEY IDENTITY”;
- MATRICOLA: ho pensato che potesse essere rappresentato come un set di caratteri alfanumerici con una lunghezza abbastanza lunga a soddisfare i nomi di matricola di ogni possibile database cliente. Ho ritenuto possibile che una famiglia non disponga di una matrice quindi questo attributo può risultare anche NULL; matricola sarà così definito: “VARCHAR(50)”;
- NOME: questo attributo conterrà il nome in linguaggio umano della famiglia, quindi il tipo di dato ideale è VARCHAR con la possibilità di essere NULL; sarà così definito: “VARCHAR (MAX)”
- ORDINAM: ho pensato che la soluzione migliore per l’ordinamento fosse quello numerico con la possibilità di non essere presente, sarà definito come “INT”.

In questa tabella abbiamo una chiave esterna:

- ID_REF: ha una relazione di riferimento con l’identificativo ID della tabella Prodotti, quindi il tipo di dato, come abbiamo deciso sopra, è un intero, e sarà definito come “INT”.

Nella tabella Proprietà tecniche gli attributi sono:

- ID: come soluzione ottimale per l'identificativo univoco ho pensato a un tipo di dato intero auto incrementante che sarà così definito: "INT PRIMARY KEY IDENTITY";
- VALORE: in questo attributo, oltre ad annotare il valore numerico di una determinata proprietà, riporterò anche le espressioni testuali che mi permettono di valutare una proprietà, quindi utilizzerò un set di caratteri, definito come "VARCHAR(MAX)";
- UDM: questo attributo invece rappresenta l'unità di misura della proprietà e ho pensato che potesse essere rappresentato come un set di caratteri alfanumerici, infatti può contenere sia caratteri alfabetici che simboli. E' definito come "VARCHAR(10)";
- TIPO: rappresenta il nome della proprietà tecnica, sarà definito come "VARCHAR(MAX)";
- ORDINAM: ho pensato che la soluzione migliore per l'ordinamento fosse quello numerico con la possibilità di non essere presente, sarà definito come "INT".

Le chiavi esterne sono:

- ID_REF: ha una relazione di riferimento con l'identificativo ID della tabella Tabelle, quindi il tipo di dato, come abbiamo deciso sopra, è un intero, e sarà definito come "INT".
- ID_DESC: ha una relazione di riferimento con l'identificativo ID della tabella Descrizioni, quindi il tipo di dato sarà definito allo stesso modo;
- ID_MAT: ha una relazione di riferimento con l'identificativo ID della tabella Materiali, quindi il tipo di dato sarà definito allo stesso modo;

- ID_RAP: ha una relazione di riferimento con l'identificativo ID della tabella Rappresentazioni, quindi il tipo di dato sarà definito allo stesso modo.

Nella tabella dei Materiali gli attributi sono:

- ID: come soluzione ottimale per l'identificativo univoco ho pensato a un tipo di dato intero auto incrementante che sarà così definito: "INT PRIMARY KEY IDENTITY";
- NOME: questo attributo conterrà il nome in linguaggio umano della famiglia, quindi il tipo di dato ideale è VARCHAR con la possibilità di essere NULL; sarà così definito: "VARCHAR(MAX)"
- ORDINAM: ho pensato che la soluzione migliore per l'ordinamento fosse quello numerico con la possibilità di non essere presente, sarà definito come "INT".

Le chiavi esterne sono:

- ID_REF: ha una relazione di riferimento con l'identificativo ID della tabella Tabelle, quindi il tipo di dato, come ho deciso sopra, è un intero, e sarà definito come "INT".
- ID_DESC: ha una relazione di riferimento con l'identificativo ID della tabella Descrizioni, quindi il tipo di dato sarà definito allo stesso modo;

Nella tabella delle Descrizioni ci sono:

- ID: come soluzione ottimale per l'identificativo univoco ho pensato a un tipo di dato intero auto incrementante che sarà così definito: "INT PRIMARY KEY IDENTITY";
- FUNZIONI: consente di identificare le tipologie dei testi, quindi ho pensato di utilizzare un set di caratteri: VARCHAR(MAX);

- TESTO: conterrà fisicamente il testo della descrizione, quindi posso utilizzare un set di caratteri alfanumerici, come “VARCHAR(MAX)”;

La chiave esterna è:

- ID_REF: ha una relazione di riferimento con l’identificativo ID della tabella Tabelle, quindi il tipo di dato, come abbiamo deciso sopra, è un intero, e sarà definito come “INT”.

5.3 Lingue e internazionalizzazione

Nella tabella Nazioni sono presenti i seguenti attributi:

- ID: corrisponde all’identificativo univoco di uno stato, quindi ho pensato di utilizzare uno standard internazionale per mappare gli stati.

Ho pensato di utilizzare ISO 3166¹⁴ che è una codifica geografica standardizzata divisa in 3 parti per codificare i nomi degli stati, dei territori dipendenti e delle principali suddivisioni amministrative dei paesi.

L’ISO 3166 comprende codici alfabetici e numerici per ciascun paese e territorio, secondo questa suddivisione:

- ISO 3166-1 alpha-2: codici paese composti da due lettere (ad esempio utilizzato per i domini di primo livello in Internet)
- ISO 3166-1 alpha-3: codici paese composti da tre lettere;
- ISO 3166-1 numerico: codice paese composti da tre cifre.

Ho ritenuto più comodo lo standard ISO 3166-1 alpha2.

Quindi ID sarà un “VARCHAR(2) PRIMARY KEY”;

¹⁴ [ISO]

- STATO: indica in linguaggio umano il nome dello stato quindi utilizzerò un set di caratteri per descriverlo, sarà un “VARCHAR(MAX);
- VALUTA: anche in questo caso utilizzerò un set di caratteri per identificare il nome della valuta utilizzata, useremo un “VARCHAR(15)”;
- SIMBOLO: i simboli di valuta sono segni grafici usati spesso come abbreviazione per i nomi delle valute. Ho quindi utilizzato un set di caratteri che rappresentano dati UNICODE: e ho definito questo attributo come un “NVARCHAR(3)”;
- CODCORRENTEVALUTA: l'ISO 4217¹⁵ è uno standard internazionale che descrive codici di tre lettere per definire i nomi delle valute. Le prime due lettere del codice corrispondono al codice nazionale definito dall'ISO 3166-1 alpha-2, la terza lettera di solito corrisponde all'iniziale del nome della valuta, l'attributo è definito come un set di caratteri “VARCHAR(3)”.

Nella tabella Lingue gli attributi sono:

- ID: come soluzione ottimale per l'identificativo univoco ho pensato a un tipo di dato intero auto incrementante che sarà così definito: “int primary key identity”;
- LINGUA: utilizzo un set di caratteri per descrivere la lingua parlata, l'attributo sarà “VARCHAR(15)”;
- ABB: utilizzo un set di caratteri per rappresentare l'abbreviazione: “VARCHAR(3)”;

La chiave esterna è:

¹⁵ [ISO]

- **ID_STATO:** ha una relazione di riferimento con l'identificativo ID della tabella Nazioni, quindi il tipo di dato sarà definito allo stesso modo come un **VARCHAR(2)**.

Nella tabella Nazioni_Lingue gli attributi sono:

- **ID:** come soluzione ottimale per l'identificativo univoco ho pensato a un tipo di dato intero auto incrementante che sarà così definito: **"INT PRIMARY KEY IDENTITY"**;

Le chiavi esterne sono:

- **ID_LINGUA:** ha una relazione di riferimento con l'identificativo ID della tabella Lingue, quindi il tipo di dato sarà definito allo stesso modo come un **"INT"**;
- **ID_NAZIONE:** ha una relazione di riferimento con l'identificativo ID della tabella Nazioni, quindi il tipo di dato sarà definito allo stesso modo come un **VARCHAR(2)**.

Nella tabella Traduzioni gli attributi sono:

- **ID:** come soluzione ottimale per l'identificativo univoco ho pensato a un tipo di dato intero auto incrementante che sarà così definito: **"INT PRIMARY KEY IDENTITY"**;
- **TESTO:** utilizzo un set di caratteri che rappresentano dati **UNICODE** per i testi delle traduzioni; infatti dovranno contenere anche simboli non tipici nel nostro vocabolario **"NVARCHAR(MAX)"**.

Le chiavi esterne sono:

- **ID_DESC:** ha una relazione di riferimento con l'identificativo ID della tabella Descrizioni, quindi il tipo di dato sarà definito allo stesso modo come un **"INT"**;

- ID_LINGUA: ha una relazione di riferimento con l'identificativo ID della tabella Lingue, quindi il tipo di dato sarà definito allo stesso modo come un "INT".

5.4 La gestione delle risorse

Nella tabella Rappresentazioni gli attributi sono:

- ID: come soluzione ottimale per l'identificativo univoco ho pensato a un tipo di dato intero auto incrementante che sarà così definito: "INT PRIMARY KEY IDENTITY";
- NOME: questo attributo conterrà il nome in linguaggio umano della famiglia, quindi il tipo di dato ideale è VARCHAR con la possibilità di essere NULL; sarà così definito: "VARCHAR (MAX)";
- TYPE: con questo attributo identifico il tipo di rappresentazione, che sia un file .jpeg, un file .fla, un file .cad, ecc. Essendo un campo in forte crescita, ho pensato di mantenere abbastanza libero l'inserimento dei tipi. Ho quindi pensato di utilizzare un set di caratteri, "VARCHAR(5)";
- PATH: è il percorso all'interno del quale è possibile trovare il file in questione; sarà quindi "VARCHAR(MAX)";
- ORDINAM: ho pensato che la soluzione migliore per l'ordinamento fosse quello numerico con la possibilità di non essere presente, sarà definito come "INT".

La chiave esterna è:

- ID_REF: ha una relazione di riferimento con l'identificativo ID della tabella Tabelle, quindi il tipo di dato, come abbiamo deciso sopra, è un intero, e sarà definito come "INT".

5.5 Prezzi e scontistica

La tabella Prezzi ha come attributi i seguenti campi:

- ID: come soluzione ottimale per l'identificativo univoco ho pensato a un tipo di dato intero auto incrementante che sarà così definito: "INT PRIMARY KEY IDENTITY";
- VALORE: ho rappresentato il valore del prezzo come un intero: "INT";
- PERCENTUALE: sarà rappresentata da un intero "INT";
- QUANTITA: sarà rappresentata da un intero "INT".

Le chiavi esterne sono:

- ID_REF: ha una relazione di riferimento con l'identificativo ID della tabella Tabelle, quindi il tipo di dato, come abbiamo deciso sopra, è un intero, e sarà definito come "INT".
- VALUTA: ha una relazione di riferimento con l'identificativo ID della tabella Nazioni, quindi il tipo di dato, come abbiamo deciso sopra, è un intero, e sarà definito come "INT".

5.6 Architettura del sistema

Per politiche aziendali del gruppo a cui appartiene DECA s.r.l. ho scoperto che non è possibile utilizzare sistemi operativi diversi da quelli marchiati Microsoft Corporation.

Il web server principale utilizzato in Deca è IIS di Microsoft, ma per applicazioni minori viene utilizzato anche Apache.

Il DBMS primariamente utilizzato è Oracle¹⁶, anche se nelle idee aziendali c'è il passaggio in toto a MSSQL come database principale.

¹⁶ [Oracle]

Gli ambienti di sviluppo utilizzati sono quello testuale o Netbeans; mentre i linguaggi più utilizzati sono PHP, HTML, XML.

5.6.1 Scelta della piattaforma di sviluppo

La piattaforma scelta per lo sviluppo delle applicazioni è WAMP¹⁷, che è un acronimo con cui si indica una piattaforma di sviluppo web/database che prende il nome dalle iniziali dei componenti software con cui è realizzata.

Prerequisito:

- Windows: il sistema operativo deve essere già installato sul PC;

Componenti installati:

- Apache: il Web server;
- MySQL: il database management system (o database server) con SQLite e relativi tool grafici;
- PHP, Perl e/o Python: i linguaggi di scripting.

WAMP è gratuito e libero ed è rilasciato sotto la GNU General Public License.

Di fatto, WAMP è la versione adattata per Windows della piattaforma AMP, così come il LAMP è quella adattata per GNU/Linux.

Analizziamo ora ogni componente singolarmente.

5.6.1.1 Server

Apache HTTP Server¹⁸, o più comunemente Apache, è il nome dato alla piattaforma server Web modulare più diffusa (ma anche al gruppo

¹⁷[WAMP]

¹⁸ [Apache]

di lavoro open source che ha creato, sviluppato e aggiornato il software server), in grado di operare da sistemi operativi UNIX-Linux e Microsoft.

Apache è un software che realizza le funzioni di trasporto delle informazioni, di internetwork e di collegamento.

5.6.1.2 DBMS - Microsoft Sql Server

Un Database Management System (abbreviato in DBMS¹⁹) è un sistema software progettato per consentire la creazione e la manipolazione efficiente di database (ovvero di collezioni di dati strutturati) solitamente da parte di più utenti. I DBMS svolgono un ruolo fondamentale in numerose applicazioni informatiche, dalla contabilità, alla gestione delle risorse umane e alla finanza fino a contesti tecnici come la gestione di rete o la telefonia.

Se in passato i DBMS erano diffusi principalmente presso le grandi aziende e istituzioni (che potevano permettersi l'impegno economico derivante dall'acquisto delle grandi infrastrutture hardware necessarie per realizzare un sistema di database efficiente), oggi il loro utilizzo è diffuso praticamente in ogni contesto.

Su richiesta di Deca stessa il dbms utilizzato sarà MSSQL server²⁰.

Microsoft SQL Server è un DBMS relazionale, meglio noto come Relational Database Management System (RDBMS), prodotto da Microsoft.

Microsoft SQL Server usa una variante del linguaggio SQL standard (lo standard ISO certificato nel 1992) chiamata T-SQL²¹ o Transact-SQL.

SQL Server Management Studio²² è un ambiente integrato per l'accesso, la configurazione, la gestione, l'amministrazione e lo

¹⁹ [ATZ09]

²⁰ [MSSQL]

²¹ [T-SQL]

²² [STA09]

sviluppo di tutti i componenti di SQL Server. SQL Server Management Studio abbina un gruppo esteso di strumenti grafici con numerosi editor di script per consentire a sviluppatori e amministratori con qualsiasi livello di esperienza di accedere a SQL Server.

SQL Server Management Studio unisce in un unico ambiente le caratteristiche di Enterprise Manager, Query Analyzer e Analysis Manager inclusi nelle versioni precedenti di SQL Server.

5.6.1.3 Linguaggio di programmazione

PHP²³ (acronimo ricorsivo di “PHP: Hypertext Preprocessor”, preprocessore di ipertesti) è un linguaggio di scripting interpretato, con licenza open source e libera (ma incompatibile con la GPL), originariamente concepito per la programmazione Web ovvero la realizzazione di pagine web dinamiche.

Attualmente è utilizzato principalmente per sviluppare applicazioni²⁴ web lato server ma può essere usato anche per scrivere script a riga di comando o applicazioni stand-alone con interfaccia grafica.

5.6.1.4 Ambiente di sviluppo

NetBeans²⁵ è un ambiente di sviluppo multi-linguaggio – uno strumento destinato ai programmatori per scrivere, compilare ed eseguire il debug ed il deploy di programmi. E’ nato nel 2000 ed è scritto in Java ma può supportare qualsiasi linguaggio di programmazione. È l’ambiente scelto dalla Sun Microsystems come IDE ufficiale, da contrapporre al più diffuso Eclipse.

NetBeans è un progetto open-source mirato a fornire un solido prodotto per sviluppare software (come NetBeans IDE e NetBeans Platform) che guida le necessità di sviluppatori, utenti e commerciali che si affidano a NetBeans per i loro prodotti.

²³ [PHP]

²⁴ [LNM09]

²⁵ [NetBeans]

6. Procedura automatica di esportazione

6.1 Software per l'impaginazione

Come funziona Frame Editor²⁶?

La prima fase è la definizione della grafica d'impaginazione della pubblicazione. La costruzione del layout (o dei layout) della pubblicazione avviene su Adobe FrameMaker²⁷ e importata successivamente a Frame Editor.

Per automatizzare una pubblicazione già esistente, quindi, è sufficiente utilizzarne i layout precedenti ed adattarli alle esigenze attuali.

La fase successiva è quella di definizione dei dati, qui si decide quali sono quelli necessari alla pubblicazione. Questi dati possono essere letti dai database aziendali (es.: AS/400, SQLServer, Oracle, ...) oppure dai consueti programmi di gestione delle informazioni (Access, Excel, ecc ...); in questa operazione si realizza un file di testo tab delimited, in gergo chiamato "tracciato".

Le informazioni sul progetto grafico e sulla definizione dei dati vengono salvate in quello che viene definito "automazione del progetto editoriale".

Dopo il processo di impaginazione otteniamo uno o più file di tipo FM, MIF (Adobe FrameMaker) oppure direttamente PDF, pronti per essere consegnati al processo di stampa, sia questo direttamente in azienda (stampa interna a colori o bianco/nero), sia per l'offset che per il digitale.

²⁶ [FE]

²⁷ [FM]

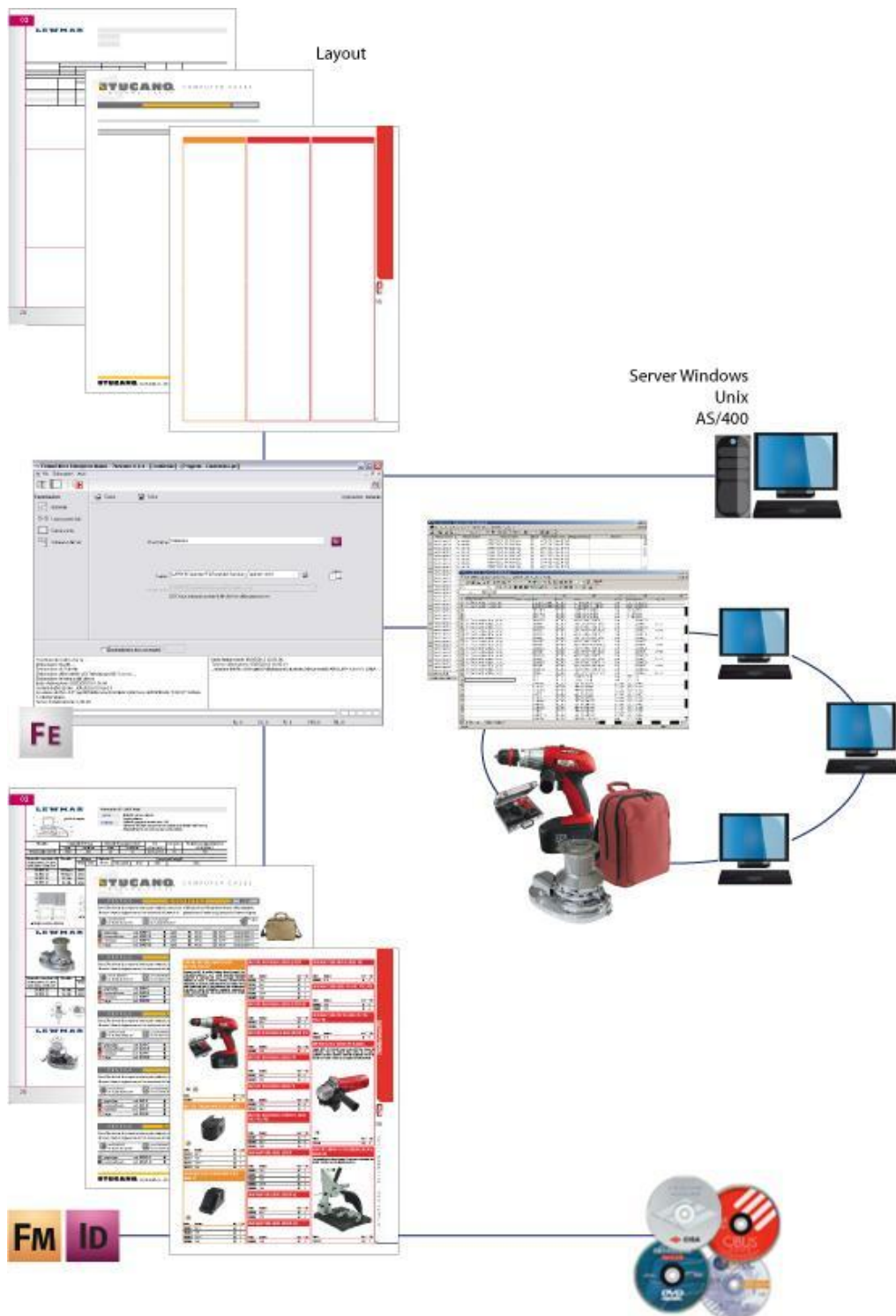


Figura 35 - Schema funzionamento FrameEditor

Microsoft Excel - Elaborato.txt																		
Type a question for help																		
Sezione																		
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
13	1 DN	65	4 KCW065F	ELETTROI	996485I01Diagr_KCW065F_4.eg										Poli			Gir
14	1 DN	65	4 KCW065F	ELETTROI	996485I01Diagr_KCW065F_4.eg										Poli			Gir
15	1 DN	65	4 KCW065F	ELETTROI	996485I01Diagr_KCW065F_4.eg										Poli			Gir
16	1 DN	65	4 KCW065F	ELETTROI	996485I01Diagr_KCW065F_4.eg										Poli			Gir
17	1 DN	65	4 KCW065F	ELETTROI	996485I01Diagr_KCW065F_4.eg										Poli			Gir
18	1 DN	65	4 KCW065F	ELETTROI	996485I01Diagr_KCW065F_4.eg										Poli			Gir
19	1 DN	65	4 KCW065F	ELETTROI	996485I01Diagr_KCW065F_4.eg										Poli			Gir
20	1 DN	65	4 KCW065F	ELETTROI	996485I01Diagr_KCW065F_4.eg										Poli			Gir
21	1 DN	65	4 KCW065F	ELETTROI	996485I01Diagr_KCW065F_4.eg										Poli			Gir
22	1 DN	65	4 KCW065F	ELETTROI	996485I01Diagr_KCW065F_4.eg										Poli			Gir
23	1 DN	65	4 KCW065F	ELETTROI	996485I01Diagr_KCW065F_4.eg										Poli			Gir
24	1 DN	65	4 KCW065F	ELETTROI	996485I01Diagr_KCW065F_4.eg										Poli			Gir
25	1 DN	65	4 KCW065F	ELETTROI	996485I01Diagr_KCW065F_4.eg										Poli			Gir
26	1 DN	65	4 KCW065F	ELETTROI	996485I01Diagr_KCW065F_4.eg										Poli			Gir
27	1 DN	65	4 KCW065F	ELETTROI	996485I01Diagr_KCW065F_4.eg										Poli			Gir
28	1 DN	65	4 KCW065F	ELETTROI	996485I01Diagr_KCW065F_4.eg										Poli			Gir
29	1 DN	65	4 KCW065F	ELETTROI	996485I01Diagr_KCW065F_4.eg										Poli			Gir
30	1 DN	65	4 KCW065F	ELETTROI	996485I01Diagr_KCW065F_4.eg										Poli			Gir
31	2 DN	65	4 KCW065F	Elettroporr	DIMENSIC966440F03-10										Poli Ita			Gir
32	2 DN	65	4 KCW065F	Elettroporr	DIMENSIC966440F03-10										Poli Ita			Gir
33	2 DN	65	4 KCW065F	Elettroporr	DIMENSIC966440F03-10										Poli Ita			Gir
34	2 DN	65	4 KCW065F	Elettroporr	DIMENSIC966440F03-10										Poli Ita			Gir
35	2 DN	65	4 KCW065F	Elettroporr	DIMENSIC966440F03-10										Poli Ita			Gir
36	2 DN	65	4 KCW065F	Elettroporr	DIMENSIC966440F03-10										Poli Ita			Gir
37	2 DN	65	4 KCW065F	Elettroporr	DIMENSIC966440F03-10										Poli Ita			Gir
38	2 DN	65	4 KCW065F	Elettroporr	DIMENSIC966440F03-10										Poli Ita			Gir
39	2 DN	65	4 KCW065F	Elettroporr	DIMENSIC966440F03-10										Poli Ita			Gir
40	2 DN	65	4 KCW065F	Elettroporr	DIMENSIC966440F03-10										Poli Ita			Gir
41	2 DN	65	4 KCW065F	Elettroporr	DIMENSIC966440F03-10										Poli Ita			Gir
42	2 DN	65	4 KCW065F	Elettroporr	DIMENSIC966440F03-10										Poli Ita			Gir
43																		
44	2 DN	65	4 KCW065F	Elettroporr	DIMENSIC966440F03-10										Poli Ita			Gir
45																		
46	2 DN	65	4 KCW065F	Elettroporr	DIMENSIC966440F03-10										Poli Ita			Gir
47																		
48	2 DN	65	4 KCW065F	Elettroporr	DIMENSIC966440F03-10										Poli Ita			Gir
49	2 DN	65	4 KCW065F	Elettroporr	DIMENSIC966440F03-10										Poli Ita			Gir
50	2 DN	65	4 KCW065F	Elettroporr	DIMENSIC966440F03-10										Poli Ita			Gir
51	2 DN	65	4 KCW065F	Elettroporr	DIMENSIC966440F03-10										Poli Ita			Gir
52	2 DN	65	4 KCW065F	Elettroporr	DIMENSIC966440F03-10										Poli Ita			Gir
53	2 DN	65	4 KCW065F	Elettroporr	DIMENSIC966440F03-10										Poli Ita			Gir
54	2 DN	65	4 KCW065F	Elettroporr	DIMENSIC966440F03-10										Poli Ita			Gir
55	2 DN	65	4 KCW065F	Elettroporr	DIMENSIC966440F03-10										Poli Ita			Gir
56	1 DN	65	2 KCM065F	ELETTROI	996485I01Diagr_KCM065F_2.eg										Poli			Gir
57	1 DN	65	2 KCM065F	ELETTROI	996485I01Diagr_KCM065F_2.eg										Poli			Gir
58	1 DN	65	2 KCM065F	ELETTROI	996485I01Diagr_KCM065F_2.eg										Poli			Gir
59	1 DN	65	2 KCM065F	ELETTROI	996485I01Diagr_KCM065F_2.eg										Poli			Gir
60	1 DN	65	2 KCM065F	ELETTROI	996485I01Diagr_KCM065F_2.eg										Poli			Gir
61	1 DN	65	2 KCM065F	ELETTROI	996485I01Diagr_KCM065F_2.eg										Poli			Gir
62	1 DN	65	2 KCM065F	ELETTROI	996485I01Diagr_KCM065F_2.eg										Poli			Gir
63	1 DN	65	2 KCM065F	ELETTROI	996485I01Diagr_KCM065F_2.eg										Poli			Gir

Figura 36 - Tracciato

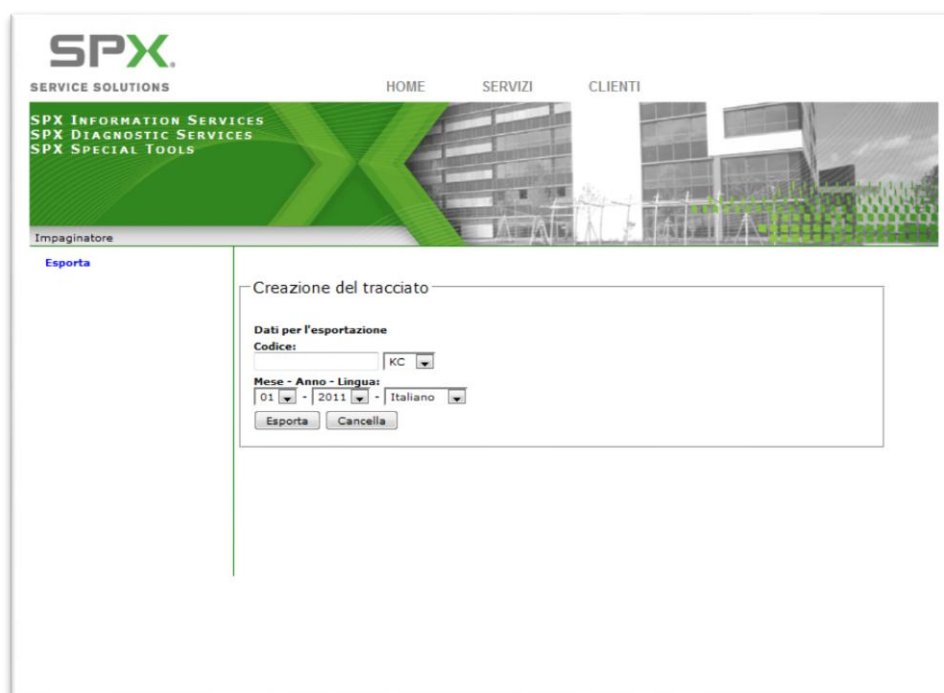
6.2 Procedura di esportazione

La procedura di esportazione supporta l'impaginatore nella creazione del catalogo. Essendo l'impaginatore una figura non necessariamente esperta nel settore informatico, ho ritenuto opportuno sviluppare un'interfaccia che lo supporti nella configurazione dei dati da esportare. Ho scelto di impiegare la piattaforma web perché, vista la sua ampia diffusione, sicuramente non rappresenta un ostacolo anche per persone non specializzate nel settore.

6.2.1 Il menu

Le operazioni possibili elencate nel menu sono le seguenti:

- Esporta;



The screenshot displays the SPX web application interface. At the top, the SPX logo is visible, along with navigation links for 'HOME', 'SERVIZI', and 'CLIENTI'. Below the navigation, there is a banner with the text 'SPX INFORMATION SERVICES', 'SPX DIAGNOSTIC SERVICES', and 'SPX SPECIAL TOOLS'. The main content area is titled 'Impaginatore' and features a blue 'Esporta' button. A form titled 'Creazione del tracciato' is present, containing the following fields:

- Dati per l'esportazione**
- Codice:** A text input field followed by a dropdown menu showing 'KC'.
- Mese - Anno - Lingua:** Three dropdown menus showing '01', '2011', and 'Italiano'.
- Buttons for 'Esporta' and 'Cancella'.

Figura 37 - Esporta

6.2.2 La logica applicativa

La logica applicativa è stata separata in due sezioni: `esporta` (`esporta.php`) che riguarda gli aspetti generali della procedura di esportazione e `esporta_cliente` (`esporta_cliente.php`) che si occupa delle particolarità che caratterizzano i dati.

Tale divisione è stata effettuata al fine di agevolare il più possibile il riuso del codice, isolando gli aspetti del caso specifico da quelli generali.

6.2.2.1 La sezione `esporta` dati del cliente

Nel file `esporta_cliente.php` possiamo trovare le seguenti funzioni:

- `function formEsporta ()`: questa funzione si occupa di generare del codice html che consente all'impaginatore di decidere il nome, la lingua e la data del tracciato. Viene chiamata dal file `impaginatore.php`.
- `function esportaTracciato($POST)`: questa funzione si occupa della creazione del tracciato, le viene passato la variabile `$POST` che contiene le informazioni di base, come ad esempio il nome che daremo al tracciato, la lingua nella quale verrà generato e la data di creazione.

La funzione viene chiamata dal file `impaginatore.php`.

- `function cercaAlim_descr()`: questa funzione si occupa di cercare la descrizione di un particolare dato del tracciato. Restituisce l'id della descrizione.
- `function cercaAlim($id_descr_alim, $id_descr_ausi, $id_ref)`: questa funzione cerca due determinate proprietà di un prodotto. Accetta le seguenti variabili:
 - `$id_descr_alim` indica l'id della descrizione della prima proprietà che cerchiamo;

- \$sid_descr_ausi indica l'id della descrizione della seconda proprietà che cerchiamo;
- \$sid_ref indica l'id del prodotto.

Restituisce il valore delle due proprietà.

- function righeVuote(\$matrow, \$arrayinsertblankrow): questa funzione scrive un determinato numero di righe di tracciato.
 - \$matrow: indica a che numero di righe siamo arrivati;
 - \$arrayinsertblankrow è dove inseriamo le righe vuote.

Restituisce \$arrayinsertblankrow con il totale delle righe vuote scritte.

- function cercaEsportaLS (\$sid_ref): questa funzione cerca una determinata proprietà di un prodotto indicato da \$sid_ref e ne restituisce il valore.
- function cercaEsportaHS (\$sid_ref): questa funzione cerca una determinata proprietà di un prodotto indicato da \$sid_ref e ne restituisce il valore.
- function cercaEsportaNPSH (\$sid_ref): questa funzione cerca una determinata proprietà di un prodotto indicato da \$sid_ref e ne restituisce il valore.
- function cercaEsportaDimensioni (\$sid_ref): questa funzione cerca una determinata proprietà di un prodotto indicato da \$sid_ref e ne restituisce il valore.
- function Control (\$mat_control, \$poli, \$famiglia): questa funzione effettua dei controlli per stabilire quali sono i prodotti da visualizzare del tracciato. Prende in input:
 - \$mat_control: rappresenta la matricola del prodotto che vogliamo controllare;
 - \$poli: rappresenta una proprietà tecnica del prodotto che incide nella scelta che effettua la funzione.

- \$famiglia: rappresenta la famiglia di appartenenza del prodotto.

6.2.2.2 La sezione esporta generale

Nel file esporta.php possiamo trovare le seguenti funzioni:

- function cercaEsportaFamiglia(): questa funzione cerca le famiglia e e le relative sottofamiglie che ci interessa inserire nel tracciato.
- function cercaEsportaProdotti (\$famiglia): cerca tutti i prodotti che appartengono a una determinata famiglia, in questo caso indicata con \$famiglia.
- function cercaEsportaVarianti (\$prodotto, \$poli): cerca tutte le varianti relative ad un prodotto (\$prodotto) caratterizzato da una determinata proprietà tecnica (\$poli).
- function _get_header(\$filename): questa funzione preleva da un determinato file (\$filename) i campi del tracciato che funzionano da guida per la creazione del nostro output.
- function PrintOutputFile (\$outmatrix, \$file_name, \$headerarray): questa funzione ci permette di scrivere il contenuto di un array all'interno di un determinate file. I dati in input sono:
 - \$outmatrix: indica l'array che vogliamo copiare nel file;
 - \$file_name: indica il nome del file;
 - \$headerarray: indica i campi del tracciato che vogliamo vengano copiato all'interno del file.

7. Un caso reale: procedura di importazione

7.1 La struttura

La struttura che ho pensato di implementare si divide in due moduli: il primo che si occupa dell'interfaccia grafica e il secondo che si occupa della logica applicativa.

L'interfaccia grafica agevola lo sviluppatore nell'eseguire la procedura di importazione, che varia il suo comportamento in base ai parametri inseriti.

7.1.1 L'interfaccia grafica

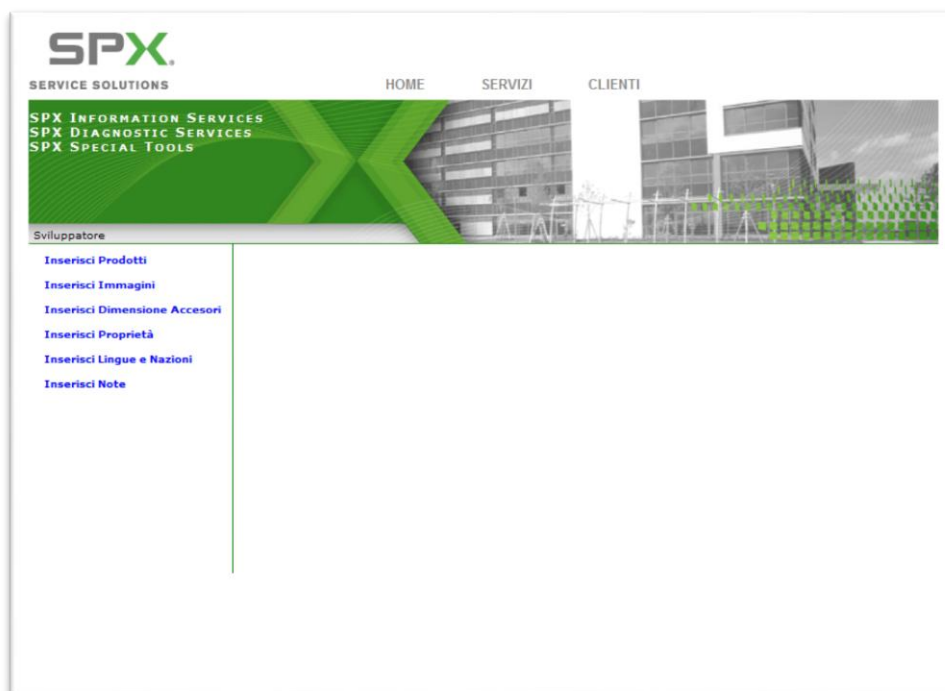


Figura 38 – Pagina iniziale

L'area di sinistra contiene l'elenco delle operazioni disponibili allo sviluppatore: rappresenta quindi il menu di partenza con cui l'utilizzatore deve interagire.

L'area centrale consente l'interazione vera e propria con lo sviluppatore; permette sia di configurare l'operazione da svolgere che di ottenere informazioni sulle condizioni di terminazione del processo.

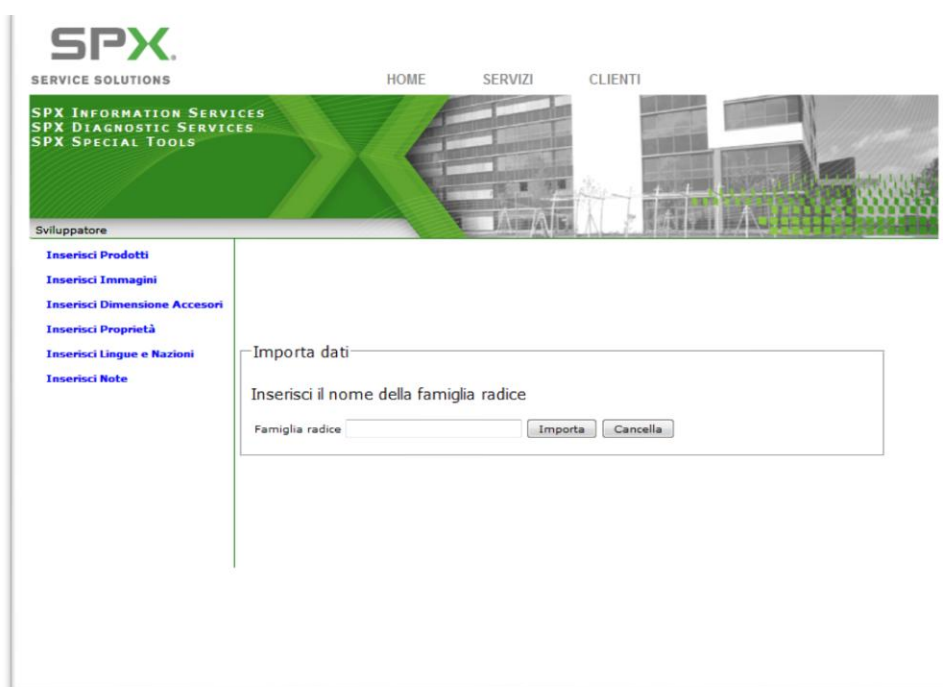
7.1.1.1 Il menu

La schermata iniziale che si presenterà allo sviluppatore sarà la seguente.

Le operazioni possibili elencate nel menu sono le seguenti:

- Inserisci prodotti:

Nella parte centrale si ha la possibilità di inserire il nome della famiglia principale.



The screenshot shows the SPX developer interface. At the top, there is a navigation bar with the SPX logo and the text 'SERVICE SOLUTIONS'. Below this, there are three main menu items: 'HOME', 'SERVIZI', and 'CLIENTI'. A large green banner with a white 'X' shape is visible, containing the text 'SPX INFORMATION SERVICES', 'SPX DIAGNOSTIC SERVICES', and 'SPX SPECIAL TOOLS'. Below the banner, there is a section titled 'Sviluppatore' (Developer) with a list of menu items: 'Inserisci Prodotti', 'Inserisci Immagini', 'Inserisci Dimensione Accessori', 'Inserisci Proprietà', 'Inserisci Lingue e Nazioni', and 'Inserisci Note'. The 'Inserisci Prodotti' item is selected. The main content area contains a form with the following elements: a text input field labeled 'Importa dati', a text input field labeled 'Inserisci il nome della famiglia radice', and a text input field labeled 'Famiglia radice' with two buttons, 'Importa' and 'Cancella', to its right.

Figura 39 - Inserisci prodotti

Il risultato ottenuto sarà la conferma del buon successo dell'operazione.

- **Inserisci immagini:**

Con questa operazione vengono inserite tutte le immagini relative ad ogni prodotto.

Il risultato sarà una schermata con la conferma dell'avvenuta operazione.

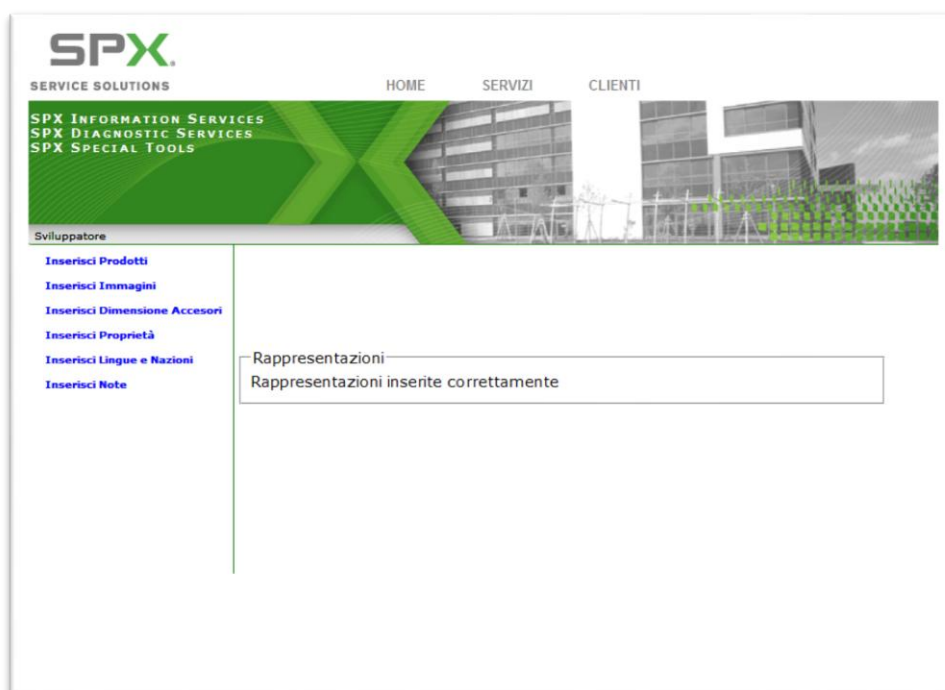


Figura 40 - Inserisci immagini

- **Inserisci dimensioni accessori:**

Con questa operazione inseriamo delle proprietà tecniche (le dimensioni di ingombro degli accessori).

Il risultato sarà una schermata con la conferma dell'avvenuta operazione.

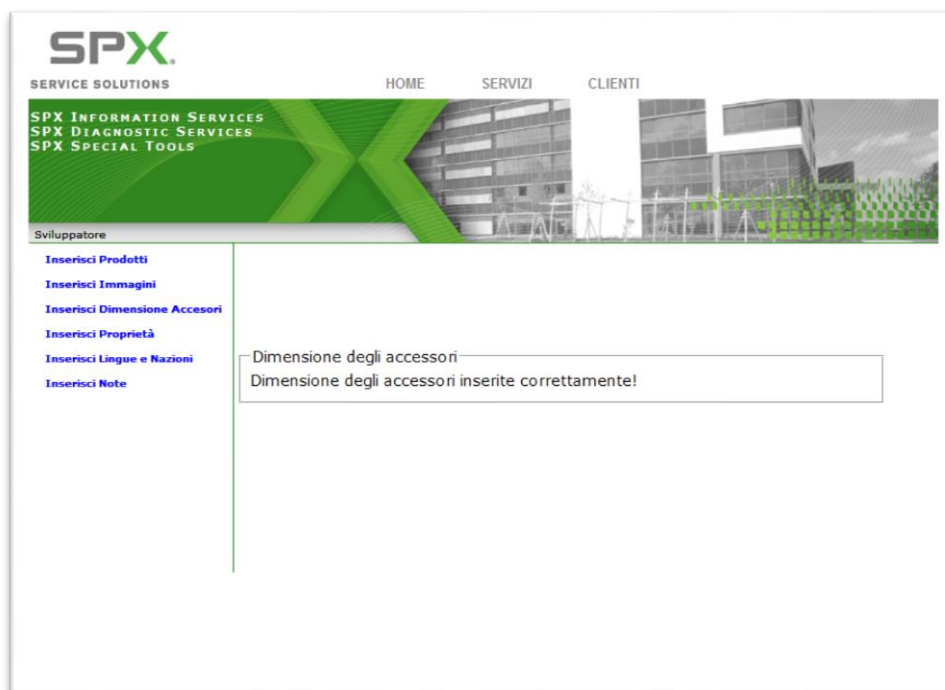


Figura 41 - Inserisci dimensioni accesorio

- Inserisci proprietà tecniche:

Dovendo inserire un elevato numero di proprietà tecniche di tipologia diversa ho pensato di raggrupparle in più punti.

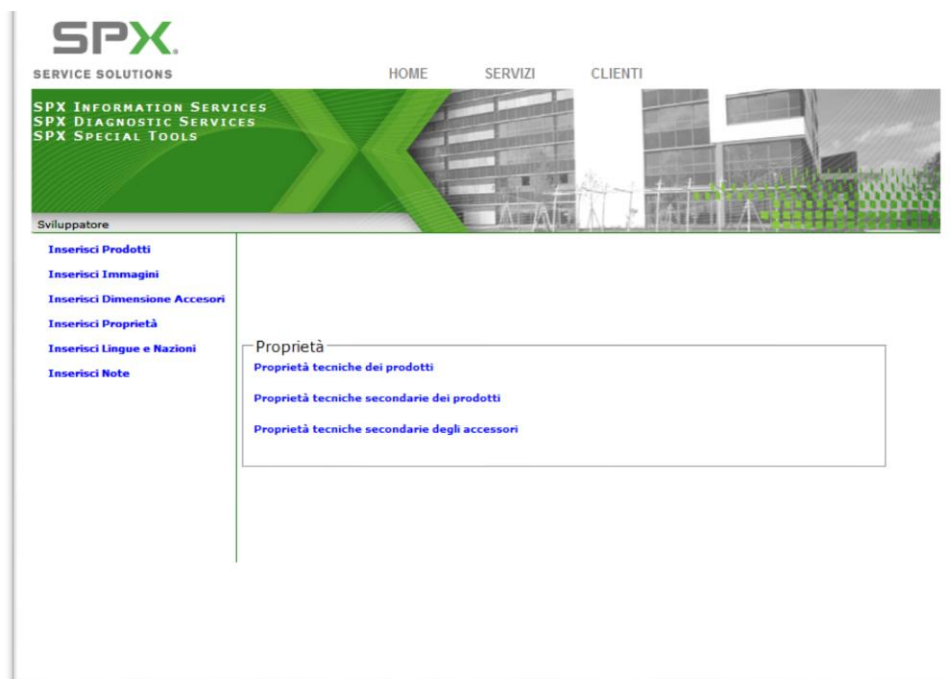


Figura 42 - Inserisci proprietà tecniche

Per ogni operazione effettuata il risultato sarà una schermata con la conferma dell'avvenuta operazione.

- **Inserisci lingue e nazioni:**

Con questo comando lo sviluppatore inserirà tutte le lingue e le relative nazioni.

Il risultato sarà una schermata con la conferma dell'avvenuta operazione.

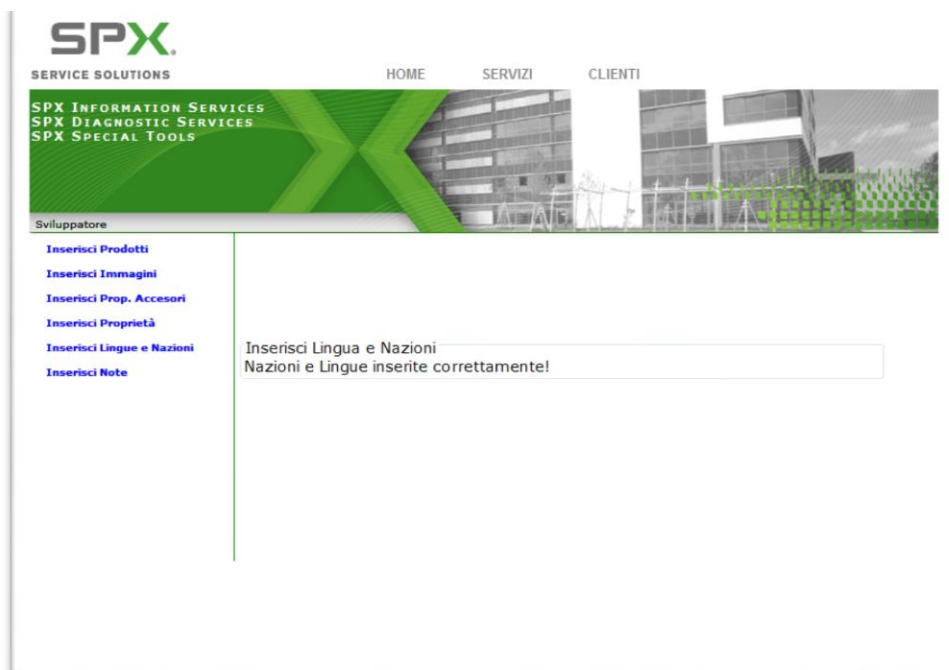


Figura 43 - Inserisci Lingua e Nazioni

- **Inserisci note:**

Con questa operazione lo sviluppatore inserirà tutte le note con le relativi traduzioni.

Il risultato sarà una schermata con la conferma dell'avvenuta operazione.

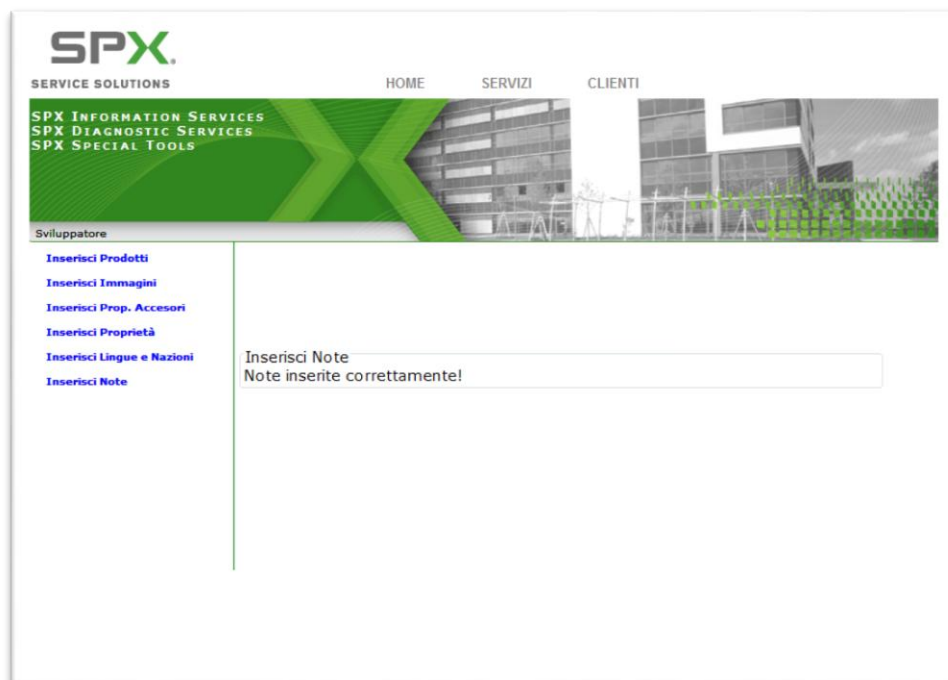


Figura 44 - Inserisci note

7.1.2 La logica applicativa

La logica applicativa è stata separata in due sezioni: importa (importa.php) che riguarda gli aspetti generali della procedura di importazione e cliente (cliente.php) che incapsula tutti gli aspetti specifici del database selezionato.

Tale divisione è stata effettuata al fine di agevolare il più possibile il riuso del codice, isolando gli aspetti del caso specifico da quelli generali.

7.1.2.1 La sezione importa

Nel file importa.php possiamo trovare le seguenti funzioni:

- function inserisciFamiglia (\$value, \$id_root):

Inserisce un prodotto nel database catalogo. Le variabili che le passo sono:

- \$value: rappresenta la matricola della famiglia
- \$id_root: rappresenta l'id del padre della famiglia che stiamo inserendo

- function inserisciProdotto (\$valore, \$nodo, \$ordinam):

Inserisce un prodotto nel database catalogo. Le variabili che le passo sono:

- \$valore è la matricola che identifica il prodotto
- \$nodo è l'id della famiglia a cui appartiene il prodotto
- \$ordinam è un numero con il quale vogliamo successivamente ordinare i prodotti

- function inserisciVarianti (\$variante, \$id_prod, \$nome, \$ordinam):

Inserisce una variabile di un prodotto all'interno del database. Le variabili che le passo sono:

- \$variante: è la matricola che identifica la variante
- \$id_prod: è l'id del prodotto a cui appartiene la variante
- \$nome: è il nome della variante
- \$ordinam: è un numero con il quale vogliamo successivamente ordinare le varianti.

- function inserisciAccessori (\$matricola, \$id_var, \$nome, \$categoria):

Inserisce nel database gli accessori relativi a un determinato prodotto o variante di prodotto. Le variabili che le devo passare sono:

- \$matricola: è la matricola che identifica l'accessorio
- \$id_var: è l'id della variante o del prodotto a cui appartiene l'accessorio
- \$nome: è il nome dell'accessorio
- \$categoria: rappresenta la categoria a cui appartiene l'accessorio

- function inserisciTabelle (\$nometabella)

Inserisce nel database la coppia di dati (nometabella e id). Le passo la variabile \$nomeTabella che identifica i nomi delle tabelle di cui tener traccia.

- function inserisciRappresentazione (\$immagine, \$path, \$tipo, \$id_tab)

Inserisce nella base di dati le rappresentazioni grafiche. Le variabili che le devo passare sono:

- \$immagine: è il nome della rappresentazione
- \$path: rappresenta il percorso dove si trova la rappresentazione
- \$tipo: è il tipo di file (estensione) di cui disponiamo
- \$id_tab: è l'id dell'oggetto che viene rappresentato.

- function inserisciDescrizioni (\$id_ref, \$funzione, \$testo):

Inserisce le descrizioni nel database. Le passo le seguenti variabili:

- \$id_ref: rappresenta l'id dell'oggetto che viene descritto dalla descrizione
 - \$funzione: indica la funzione della nostra descrizione
 - \$testo: contiene il testo della descrizione
- function inserisciPropTecnica (\$id_tab, \$id_descr, \$id_mat, \$id_rap, \$valore, \$udm):

Questa funzione si occupa dell'inserimento delle proprietà tecniche. Le passo le seguenti variabili:

- \$id_tab: è l'id dell'oggetto a cui appartiene la proprietà tecnica
- \$id_descr: è l'id della descrizione della proprietà tecnica
- \$id_mat è l'id del materiale
- \$id_rap: è l'id della rappresentazione
- \$valore: indica il valore della proprietà che stiamo inserendo
- \$udm: è l'unità di misura del valore.

- function inserisciNazioni (\$path_nazioni):

Questa funzione si occupa dell'inserimento delle nazioni che si trovano all'interno del file identificato da \$path_nazioni.

- function inserisciLingue (\$path_lingue):

Questa funzione si occupa dell'inserimento delle lingue che si trovano all'interno del file identificato da \$path_lingue.

- function inserisciNazioni_Lingue (\$path_nazioni_lingue):

Questa funzione si occupa dell'inserimento della coppia (nazioni, lingue) che si trovano all'interno del file identificato da \$path_nazioni_lingue

- function inserisciNote (\$path_note):

Questa funzione si occupa dell'inserimento delle note che si trovano all'interno del file identificato da \$path_note.

- function inserisciTraduzioni (\$id_descr, \$id_lingua, \$testo):

Questa funzione si occupa di inserire le traduzioni delle note. Le passo le seguenti varianti:

- \$id_descr è l'id della descrizione di cui inserisco la traduzione
- \$id_lingua è l'id della lingua della traduzione
- \$testo è il testo della traduzione

7.1.2.2 La sezione cliente

Nel file cliente.php possiamo trovare le seguenti funzioni:

- function smistaProdotti (\$id_famiglie, \$root, \$id_root):

Questa funzione mi permette di smistare i prodotti del cliente tra le varie famiglie a disposizione. Le passo le seguenti variabili:

- \$id_famiglie: l'id della famiglia di appartenenza
- \$root: è il nome del capostipite di tutte le famiglie
- \$id_root: è l'id del capostipite di tutte le famiglie

- function cercaVarianti (\$root, \$valore, \$id_prod):

Questa funzione mi permette di cercare all'interno del database del cliente le varianti di un determinato prodotto. Le passo le seguenti variabili:

- \$root: è il nome del capostipite di tutte le famiglie
- \$valore : è il nome del prodotto a cui appartengono le varianti da cercare
- \$id_prod: è l'id del prodotto

- function cercaRappresentazioni ():

Questa funzione cerca all'interno del database del cliente le rappresentazioni.

- function cercaPropTecnica_ProductsData3 ():

Questa funzione cerca all'interno del database del cliente un determinato tipo di proprietà tecniche.

- function cercaDescrizioni ():

Questa funzione cerca all'interno del database del cliente le descrizioni.

- function gestionePropTecniche ():

Questa funzione si occupa di gestire tutte le proprietà tecniche

- function calcolals_m3h (\$codice, \$id_tab, \$id_tab_descr_ls, \$id_tab_descr_m3h):

Questa funzione si occupa prevalentemente delle proprietà tecniche ls e m3h. Le passo le seguenti variabili:

- \$codice: è il codice che identifica un prodotto all'interno del database del cliente
 - \$id_tab: è l'id del prodotto nella base di dati aziendale
 - \$id_tab_descr_ls: è l'id della descrizione ls
 - \$id_tab_descr_m3h: è l'id della descrizione m3h
- function calcolanpsh (\$codice, \$id_tab, \$id_tab_descr_npsh, \$id_tab_descr_hs, \$ls, \$idcurve, \$iddiagram):

Questa funzione si occupa prevalentemente della proprietà tecnica npsh. Le passo le seguenti variabili:

- \$codice: è il codice che identifica un prodotto all'interno del database del cliente
 - \$id_tab: è l'id del prodotto nella base di dati aziendale
 - \$id_tab_descr_npsh: è l'id della descrizione npsh
 - \$id_tab_descr_hs: è l'id della descrizione hs
 - \$ls: è un array contenente i dati ls
 - \$idcurve: è un dato ricavato dalla funzione calcolals_m3h che serve per calcolare npsh
 - \$iddiagram: è un dato ricavato dalla funzione calcolals_m3h che serve per calcolare npsh
- function cercaPoli(\$id_tab, \$id_descr_poli):

Questa funzione si occupa di cercare la proprietà tecnica chiamata poli. Le passo le seguenti variabili:

- \$id_tab: è l'id del prodotto
 - \$id_descr_poli: è l'id della descrizione relativa ai poli
- function cercaProprieta():

Questa funzione si occupa della gestione di alcune particolari proprietà.

- function cercaDNMandata(\$sid_tab, \$codice, \$sid_descr_mand):

Questa funzione si occupa di cercare la proprietà tecnica chiamata DN Mandata. Le passo le seguenti variabili:

- \$sid_tab: è l'id del prodotto
- \$codice: è il codice che identifica un prodotto all'interno del database del cliente
- \$sid_descr_mand: è l'id della descrizione relativa a Dn mandata

- function cercaPassaggioLibero(\$sid_tab, \$codice, \$sid_descr_pl):

Questa funzione si occupa di cercare la proprietà tecnica chiamata passaggio libero. Le passo le seguenti variabili:

- \$sid_tab: è l'id del prodotto
- \$codice: è il codice che identifica un prodotto all'interno del database del cliente
- \$sid_descr_pl: è l'id della descrizione relativa al passaggio libero

- function cercaSonde(\$sid_tab, \$codice, \$sid_descr_sonde):

Questa funzione si occupa di cercare la proprietà tecnica chiamata poli. Le passo le seguenti variabili:

- \$sid_tab: è l'id del prodotto
- \$codice: è il codice che identifica un prodotto all'interno del database del cliente
- \$sid_descr_sonde: è l'id della descrizione relativa alle sonde

- function cercaCavi(\$id_tab, \$codice, \$id_descr_alim, \$id_descr_ausi):

Questa funzione si occupa di cercare la proprietà tecnica chiamata cavi. Le passo le seguenti variabili:

- \$id_tab: è l'id del prodotto
- \$codice: è il codice che identifica un prodotto all'interno del database del cliente
- \$id_descr_alim: è l'id della descrizione relativa ai cavi di alimentazione
- \$id_descr_ausi: è l'id della descrizione relativa ai cavi ausiliari

- function Conv_1(\$s):

Questa è una funzione che ci serve per il calcolo di alcune proprietà tecniche.

- function _ValApprox(\$x, \$min, \$max):

Questa è una funzione che ci aiuta nell'approssimazione dei dati di alcune proprietà tecniche.

8. Conclusioni

Il problema iniziale per il quale ho intrapreso questa tesi era la difficoltà nella realizzazione di cataloghi cartacei. L'obiettivo, oltre a facilitare la creazione di un impaginato, era nell'ideare un processo guida per trovare e identificare i dati necessari alla compilazione.

Ho pensato allora alla realizzazione di un database intermedio standard nel quale si potessero inserire i dati di interesse e dal quale si potessero estrapolare con un meccanismo standard a tutti i cataloghi.

Ho analizzato la struttura dei cataloghi cartacei preesistenti, ho effettuato uno studio dei requisiti, ho svolto uno stadio di analisi e ho realizzato la progettazione della base di dati, ottimizzandola secondo i metodi canonici²⁸.

Ho implementato la base di dati e sviluppato un'applicazione per l'importazione e l'esportazione testando entrambi con un caso reale.

Non sono stati effettuati test e benchmark comparativi sui tempi di esecuzione poiché il sistema va a sostituire un processo manuale e l'ottimizzazione della durata della procedura non è un requisito significativo dell'azienda.

8.1. Sviluppi futuri

Tra i possibili sviluppi futuri che ho individuato per l'applicazione vi sono:

- La crittografia delle password di accesso dello sviluppatore e dell'impaginato per una eventuale utilizzo del software in remoto da una sede decentrata;
- Analisi e sviluppo di un'infrastruttura di sicurezza per l'applicazione di importazione ed esportazione dei dati;

²⁸ [ATZ09]

- La realizzazione di un sistema per il semplice aggiornamento dei dati nel caso di nuovi prodotti inseriti dal cliente;

Un ulteriore punto di interesse riguarda la complessità della base di dati progettata: essa consente di modellare tutti i possibili scenari di database cliente, ma proprio per questo motivo nella maggior parte dei casi buona parte delle strutture dati in essa contenute non vengono impiegate, causando un cattivo uso delle risorse a disposizione.

Si ritiene dunque opportuno valutare come sviluppo futuro l'aggiunta all'applicazione di codice per la realizzazione di interfaccia web per la scelta dinamica delle tabelle nella creazione della base di dati.

Appendice A: Codice SQL

Creazione del database

```
Create database 'TESI';
```

Creazione delle tabelle

```
create table Famiglie
(
  Id int primary key identity,
  Matricola varchar(50) not null,
  Id_Padre int,
  Nome varchar (MAX),
  Ordinam int
  foreign Key(Id_Padre)
    references Famiglie(Id)
)
create table Prodotti
(
  Id int primary key identity,
  Matricola varchar(50) not null,
  Id_Fam int,
  Nome varchar (MAX),
  Ordinam int,
  foreign Key(Id_Fam)
    references Famiglie(Id)
)

create table Varianti
(
  Id int primary key identity,
```

```

Matricola varchar(50) not null,
Id_Ref int,
Nome varchar (MAX),
Ordinam int,
foreign Key(Id_Ref)
    references Prodotti(Id)
)

```

```

create table Accessori
(
Id int primary key identity,
Matricola varchar(50) not null,
Nome varchar (MAX),
Categoria varchar(MAX),
Ordinam int,
)

```

```

create table Var_Acc
(
Id int primary key identity,
Id_Acc int,
id_var int,
foreign Key(Id_Acc)
    references Accessori(Id),
foreign Key(Id_Var)
    references Varianti(Id)
)

```

```

create table Tabelle
(
Id int primary key identity,
Id_Ref int,
NomeTabella varchar(MAX)
)

```

```

create table Composizioni
(

```

```

Id int primary key identity,
Id_Pezzo int NOT NULL,
Id_PezzoBase int,
Quantità int,
foreign Key(Id_Pezzo)
    references Tabelle(Id),
foreign Key(Id_PezzoBase)
    references Tabelle(Id)
)

```

```

create table Rappresentazioni
(
Id int primary key identity,
Id_Ref int,
Nome varchar(15),
Path varchar(50),
Type varchar(5),
Ordinam int,
foreign Key(Id_Ref)
    references Tabelle(Id),
)

```

```

create table Nazioni
(
Id varchar(2)primary key,
Stato varchar(MAX),
Valuta varchar(MAX),
Simbolo nvarchar(4),
CodCorrenteValuta varchar(3)
)

```

```

create table Lingue
(
Id int primary key identity,
Lingua varchar(MAX),
ABB varchar(3)
)

```

```

create table Nazioni_Lingue
(
  Id int primary key identity,
  Id_Lingua int,
  Id_Nazione varchar(2),
  foreign Key(Id_Lingua)
    references Lingue(Id),
  foreign Key(Id_Nazione)
    references Nazioni(Id)
)

```

```

create table Prezzi
(
  Id int Primary key identity,
  Id_Ref int,
  Valore int,
  Valuta varchar(2),
  Percentuale int,
  Quantità int,
  foreign Key(Valuta)
    references Nazioni(Id),
  foreign Key(Id_Ref)
    references Tabelle(Id)
)

```

```

create table Descrizioni
(
  Id int primary key identity,
  Id_Ref int,
  Funzione varchar(MAX),
  Testo varchar(MAX),
  foreign Key(Id_Ref)
    references Tabelle(Id)
)

```

```

create table Traduzioni

```

```
(
Id int primary key identity,
Id_Desc int,
Id_Lingua int,
Testo nvarchar(MAX),
foreign Key(Id_Desc)
    references Descrizioni(Id),
foreign Key(Id_Lingua)
    references Lingue(Id)
)
```

```
create table Materiali
(
Id int Primary Key identity,
Id_Ref int,
Id_Desc int,
Nome varchar(MAX),
Ordinam int,
foreign Key(Id_Ref)
    references Tabelle(Id),
foreign Key(Id_Desc)
    references Descrizioni(Id)
)
```

```
create table Proprieta
(
Id int Primary Key identity,
Id_Ref int,
Id_Desc int,
Id_Mat int,
Id_Rap int,
Valore varchar(MAX),
UdM varchar(10),
Ordinam int,
foreign Key(Id_Ref)
    references Tabelle(Id),
foreign Key(Id_Desc)
```

```

        references Descrizioni (Id),
foreign Key (Id_Mat)
        references Materiali (Id),
foreign Key (Id_Rap)
        references Rappresentazioni (Id)
)

```

Elenco delle principali insert utilizzate

Per inserire i dati di un prodotto:

```

INSERT INTO [Tesi_Caprari].[dbo].[Prodotti]
VALUES (<Matricola>, <Id_Fam>, <Nome>,
        <Ordinam>)

```

Per inserire i dati di una famiglia:

```

INSERT INTO [Tesi_Caprari].[dbo].[Famiglie]
VALUES (<Matricola>, <Id_Padre>,
        <Nome>, <Ordinam>)

```

Per inserire i dati di una variante:

```

INSERT INTO [Tesi_Caprari].[dbo].[Varianti]
VALUES (<Matricola>, <Id_Ref>, <Nome>,
        <Ordinam>)

```

Per inserire i dati di un accessorio:

```

INSERT INTO [Tesi_Caprari].[dbo].[Accessori]
VALUES (<Matricola>, <Nome>, <Categoria>,
        <Ordinam>)

```

Per inserire una rappresentazione:

```

INSERT INTO
    [Tesi_Caprari].[dbo].[Rappresentazioni]
VALUES (<Id_Ref>, <Nome>, <Path>, <Type>,
        <Ordinam>)

```

Per inserire le descrizioni:

```
INSERT INTO [Tesi_Caprari].[dbo].[Descrizioni]
VALUES (<Id_Ref>, <Funzione>, <Testo>)
```

Per inserire un materiale:

```
INSERT INTO [Tesi_Caprari].[dbo].[Materiali]
VALUES (<Id_Ref>, <Id_Desc>, <Nome>,
<Ordinam>)
```

Per inserire i dati delle proprietà tecniche:

```
INSERT INTO [Tesi_Caprari].[dbo].[Proprieta]
VALUES (<Id_Ref>, <Id_Desc>, <Id_Mat>,
<Id_Rap>, <Valore>, <UdM>,
<Ordinam>)
```

Per inserire le lingue:

```
INSERT INTO [Tesi_Caprari].[dbo].[Lingue]
VALUES (<Lingua>, <ABB>)
```

Per inserire le nazioni:

```
INSERT INTO [Tesi_Caprari].[dbo].[Nazioni]
VALUES (<Id>, <Stato>, <Valuta>, <Simbolo>,
<CodCorrenteValuta>)
```

Per inserire i prezzi

```
INSERT INTO [Tesi_Caprari].[dbo].[Prezzi]
VALUES (<Id_Ref>, <Valore>, <Valuta>,
<Percentuale>, <Quantità>)
```

Elenco delle principali select utilizzate

Per selezionare l'ultimo ID appena inserito:

```
select distinct (@@IDENTITY) as 'identity';
```

Per selezionare le varianti e i prodotti appartenenti ad una famiglia <famiglia>:

```
SELECT Prodotti.Matricola as Prodotto,  
       substring(Varianti.Matricola,14,1) as Poli  
FROM Famiglie, Prodotti, Varianti  
WHERE Prodotti.Id_fam = Famiglie.Id  
       AND Famiglie.Matricola = '<famiglia>'  
       AND Varianti.Id_Ref = Prodotti.Id  
GROUP BY Prodotti.Matricola, Prodotti.Ordinam,  
       substring(Varianti.Matricola,14,1)  
ORDER BY substring(Prodotti.Matricola,4,4),  
       Poli desc;
```

Per selezionare le varianti appartenenti ad un prodotto <prodotto> che non contengano il codice 'X':

```
SELECT Varianti.Matricola, Varianti.Nome,  
       Varianti.Ordinam  
FROM Prodotti, Varianti  
WHERE Varianti.Id_Ref = Prodotti.Id  
       AND Prodotti.Matricola = '<prodotto>'  
       AND SUBSTRING (Varianti.Matricola, 16,1) not  
       like 'X'  
ORDER BY Varianti.Ordinam desc;
```

Per selezionare tutte le descrizioni delle note in una determinata lingua <lang> e appartenenti ad una determinata funzione <funz>:

```
SELECT Descrizioni.Id, Descrizioni.Funzione,  
       Descrizioni.Testo, Traduzioni.Testo as  
       LANG  
FROM Descrizioni, Traduzioni  
WHERE Funzione like '<funz>'
```



```
AND Traduzioni.Id_Lingua = '<lang>'
AND Descrizioni.Id = Traduzioni.Id_Desc;
```

Per selezionare i dati delle proprietà <prop1> e <prop2> di un determinato prodotto <prodotto>:

```
SELECT *
FROM Proprieta
WHERE EXISTS
    (SELECT Id
     FROM Descrizioni
     WHERE (Testo = '<prop1>'
           OR Testo = '<prop2>')
          AND Descrizioni.Id =Proprieta.Id_Desc
          AND Proprieta .Id_Ref = '<prodotto>');
```

Per selezionare il valore di una proprietà in base alla sua descrizione e al relativo prodotto <prodotto>:

```
SELECT Valore
FROM Proprieta
WHERE (Id_Desc = '<descr1>'
      OR Id_Desc = '<descr2>')
      AND Id_Ref = '<prodotto>;
```

Per selezionare l'id di una lingua <lingua>

```
select Id from Lingue where Lingua='<lingua>;
```


Appendice B: Estratto di codice PHP

File global.php contenente le variabili globali:

```
$server_name =  
    'http://localhost:81/Tesi_Codice';  
$server_index = 'index.php';  
$base_path = 'Tesi_Codice';  
$document_root = 'http://localhost:81/';  
$absolute_base_path =  
    $document_root . $base_path;
```

Variabili per la connessione al db

```
$dbType = "mssql";  
$dbHost = "localhost";  
$dbUser = "Sara";  
$dbPass = "";  
$dbDatabaseBase = "Tesi";  
  
$dbCliente = "Editoria";  
$dbAzienda = "Tesi_Caprari";  
$user = ".[dbo].";
```

```
$tabProdotti =  
    "[".$dbAzienda.""].$user."[Prodotti];  
$tabFamiglie =  
    "[".$dbAzienda.""].$user."[Famiglie];  
$tabVarianti =  
    "[".$dbAzienda.""].$user."[Varianti];  
$tabAccessori =  
    "[".$dbAzienda.""].$user."[Accessori];  
$tabTabelle =  
    "[".$dbAzienda.""].$user."[Tabelle];
```

```
$tabRappresentazioni =  
    "[".$dbAzienda.""].$user."[Rappresenta  
    zioni]";
```

Codice per effettuare connessioni al database con select:

```
global $dbUser,$dbPass,$dbCliente,  
    $ErrorMessage;  
$ErroreLocale='';  
$db = new database($dbUser, $dbPass,  
    $dbCliente);  
$db->connetti();  
$query="<query_select>";  
$db->esegui($query, "select");  
$result = $db->result();  
$db->disconnetti();
```

Codice per effettuare connessioni al database con insert:

```
global $dbUser,$dbPass,$dbCliente,$dbAzienda;  
$db = new database($dbUser, $dbPass,  
    $dbAzienda);  
$db->connetti();  
$query="<query_insert>";  
$db->esegui($query, "insert");  
$db->disconnetti();
```

```
$db = new database($dbUser, $dbPass,  
    $dbAzienda);  
$db->connetti();  
$query="select distinct(@@IDENTITY) as  
    'identity'";  
$db->esegui($query, "select");  
$result = $db->result();  
$ID = $result['0']['identity'];
```

```

$db->disconnetti();
if ($ID == 0)
{
    $ErroreLocale .= $ErrorMessage;
    $ErrorMessage = '';
    continue;
}

```

La seguente funzione si occupa di cercare i prodotti all'interno del database del cliente, li smista utilizzando opportuni parametri e li inserisce all'interno della base dati aziendale.

```

function smistaProdotti ($ID_FAMIGLIE, $root,
$ID_ROOT) {
global
    $dbUser, $dbPass, $dbCliente, $ErrorMessage;
$ErroreLocale='';
$db=new database($dbUser, $dbPass, $dbCliente);
$db->connetti();
$query="select distinct (SUBSTRING(MAT,1,7)) as
MAT
From      [Editoria].[CAPvwProductsListFull],
[Editoria].[CAPvwMatParsedValues],
[Editoria].[CAPvwMatParsing],
[Editoria].[CAPvwMatFamGroups]
where strFamGroup = 'KKK'
and strTokenName = 'serie'

```

```

and
[Editoria].[dbo].[CAPvwMatFamGroups].guiRowId =
[Editoria].[dbo].[CAPvwMatParsing].guiFamGroup

and
[Editoria].[dbo].[CAPvwMatParsedValues].guiMatP
arsing
=
[Editoria].[dbo].[CAPvwMatParsing].guiRowId

and [Editoria].[dbo].[CAPvwMatParsedValues].cod
=
[Editoria].[dbo].[CAPvwProductsListFull].cod;";

$db->esegui($query, "select");

$result = $db->result();

$db->disconnetti();

if ($result){
    foreach ($result as $value) {
        if((substr($value['MAT'],0,7)== 'KCT040F')
        || (substr($value['MAT'],0,7) == 'KCT040H')
        || (substr($value['MAT'],0,7) == 'KST040F')
        ||(substr($value['MAT'],0,7) == 'KST040H'))
        { $nodo = 'DN040';
            $MAT = $value['MAT'];
            $ORDINAM = 0;
        }
    }
else if((substr($value['MAT'],0,7)=='KCW065F')

```

```

        || (substr($value['MAT'],0,7) == 'KCM065F')
        || (substr($value['MAT'],0,7) == 'KSW065F'))
    {$nodo = 'DN065';
    $MAT = $value['MAT'];
    if (substr($MAT,2,1) == 'W')
        $ORDINAM = 1;
    if (substr($MAT,2,1) == 'M')
        $ORDINAM = 2;
    }
    $ID_PROD = inserisciProdotto($MAT,
        $ID_FAMIGLIE[$nodo], $ORDINAM);
    if ($ID_PROD==0)
    {$ErroreLocale .= $ErrorMessage;
    $ErrorMessage = '';
    continue;}
    cercaVarianti($root, $MAT, $ID_PROD);
    }
}
}

```

La seguente funzione viene chiamata da `smistaProdotti` e si occupa della ricerca delle varianti nella base dati del cliente e del relativo inserimento nel database aziendale.

```
function cercaVarianti($root, $valore,
    $ID_PROD) {
    global $dbUser,$dbPass,$dbCliente;
    $db=new database($dbUser, $dbPass, $dbCliente);
    $db->connetti();
    $query="select *
    from [Editoria].[CAPvwProductsListFull]
    where exists(
        select cod
        from [Editoria].[CAPvwMatParsedValues]
        where exists(
            select guiRowId
            from [Editoria].[CAPvwMatParsing]
            where exists(
                select guiRowId
                from
                    [Editoria].[CAPvwMatFamGroups]
                where strFamGroup = '$root'
                and strTokenName = 'serie'
```



```

        and

        [Editoria].[CAPvwMatFamGroups].guiRowId =

        [Editoria].[CAPvwMatParsing].guiFamGroup

        and

[Editoria].[CAPvwMatParsedValues].guiMatParsing
= [Editoria].[CAPvwMatParsing].guiRowId

        and

[Editoria].[CAPvwMatParsedValues].Cod           =
[Editoria].[CAPvwProductsListFull].cod

        and

substring([Editoria].[CAPvwProductsListFull].MAT,1,7) =

        '$valore')));";

$db->esegui($query, "select");

$result_var = $db->result();

$db->disconnetti();

If ($result_var){

    foreach ($result_var as $value)

        {$variante = $value['MAT'];

            $nome = $value['DESCLNG1'];

            $codice = $value['COD'];

            if (substr($variante, 13, 1) == '2')

                $ORDINAM = 2;

```

```
if (substr($variante, 13, 1) == '4')
    $ORDINAM = 4;
if (substr($variante, 13 ,1) == '6')
    $ORDINAM = 6;
if (substr($variante, 13 ,1) == '8')
    $ORDINAM = 8;
$ID_VAR = inserisciVarianti($variante,
    $ID_PROD, $nome, $ORDINAM);
}
}
```

Appendice C: Informazioni riguardanti lo sviluppo

Ambiente di sviluppo²⁹

Configurazione di sistema:

- *Sistema operativo*: Microsoft Windows Vista
- *Calcolatore*: E5400
- *Processore*: Pentium Dual-Core @ 2.70 GHz
- *Memoria*: 3 GB di RAM

Strumenti utilizzati:

- *Web-Server*: Apache³⁰, SQLServer 2008³¹
- *Database*: SQLServer³²
- *Linguaggio di programmazione*: PHP³³
- *Ambiente di sviluppo*: NetBeans 6.9.1³⁴

Software ausiliari:

- *Schema E-R*: Dia³⁵
- *Impaginazione*: Frame Editor³⁶ – Tesla Sistemi Informatici S.r.l.
- *Impaginazione*: FrameMaker 9.0³⁷ – Adobe System Inc.
- *Visualizzazione*: Acrobat Reader – Adobe System Inc.
- *Foglio di scrittura*: Word 2010 – Microsoft Corporation
- *Foglio di calcolo*: Excel 2010 – Microsoft Corporation

²⁹ Tutti i marchi anche se non espressamente menzionati sono delle relative case costruttrici.

³⁰ [Apache]

³¹ [STA09]

³² [MSSQL]

³³ [PHP]

³⁴ [Netbeans]

³⁵ [DIA]

³⁶ [FE]

³⁷ [FM]

Indice delle figure

Figura 1 - Come lavorano oggi	15
Figura 2 - Come sarà.....	16
Figura 3 - Funzionamento di base.....	17
Figura 4 - Cosa farà lo sviluppatore.....	17
Figura 5 - Cosa farà l'Impaginatore.....	18
Figura 6 - Cosa farà l'impaginatore	18
Figura 7 - Esempio di catalogo di vacanze	22
Figura 8 - Esempio di catalogo di prodotti	23
Figura 9 - Esempio di catalogo di prodotti	24
Figura 10 - Esempio di base di dati del cliente.....	26
Figura 11 - Glossario dei termini	29
Figura 12 - Albero delle famiglie.....	30
Figura 13 - UML delle famiglie.....	31
Figura 14 - Generalizzazione di Prodotti finiti, Accessori e.....	32
Figura 15 - Famiglie di Prodotti.....	32
Figura 16 - UML di Varianti.....	33
Figura 17 - UML delle Proprietà tecniche	34
Figura 18 - UML dei materiali.....	35
Figura 19 - UML delle descrizioni.....	36
Figura 20 - UML di lingue.....	37
Figura 21 - UML delle rappresentazioni multimediali	38
Figura 22 - UML di prezzo	39
Figura 23 - Differenze di marketing	40
Figura 24 - Famiglie.....	43
Figura 25 - Eliminazione della generalizzazione.....	44
Figura 26 - Prodotti	45
Figura 27 - Varianti	46
Figura 28 - Proprietà tecniche.....	47
Figura 29 - Materiali.....	48

Figura 30 - Descrizioni	49
Figura 31 - Lingue e nazioni	50
Figura 32 - Rappresentazioni multimediali.....	51
Figura 33 - Prezzi e scontistica	51
Figura 34 - Schema UML finale	52
Figura 35 - Schema funzionamento FrameEditor.....	80
Figura 36 - Tracciato.....	81
Figura 37 - Esporta.....	82
Figura 38 - Pagina iniziale	87
Figura 39 - Inserisci prodotti.....	88
Figura 40 - Inserisci immagini	89
Figura 41 - Inserisci dimensioni accesori	90
Figura 42 - Inserisci proprietà tecniche	91
Figura 43 - Inserisci Lingua e Nazioni	92
Figura 44 - Inserisci note	93

Bibliografia

Principali libri di interesse:

- [ATZ09] - P. Atzeni, S. Ceri, S. Paraboschi, R. Torlone: “Basi di dati - Modelli linguaggi di interrogazione”, McGrawHill 2009
- [ATZ07] - P. Atzeni, S. Ceri, P. Fraternali, S. Paraboschi, R. Torlone: “Basi di dati - Architetture e linee di evoluzione”, McGrawHill 2007
- [AN07] - Arlow e Neustadt, “UML e Unified Process”, McGrawHill, 2007
- [BRJ98] - Booch , Rumbaugh, Jacobson, “The UML User Guide”, AW, 1998
- [STA09] - William R. Stanek, “Microsoft SQL Server 2008. Guida all'uso ”, Mondadori Informatica, 2009
- [LNM09] - T. Lecky, S. Nowicki, T. Myer, “PHP 6 - Guida per lo sviluppatore”, Hoepli, 2009

Principali siti di interesse:

[Wikipedia] - www.wikipedia.com

[W3School] - <http://www.w3schools.com>

[ISO] - <http://www.iso.org>

[PHP] - www.php.net

[T-SQL] -

[http://msdn.microsoft.com/it-it/library/ms189826\(v=sql.90\).aspx](http://msdn.microsoft.com/it-it/library/ms189826(v=sql.90).aspx)

[MSSQL] - <http://www.microsoft.com/italy/server/sql/default.msp>

[WAMP] - www.wampserver.com/en

[FE] - <http://www.frameeditor.it/>

[FM] - <http://www.adobe.com/it/products/framemaker.html>

[Apache] - <http://www.apache.org/>

[Netbeans] - <http://netbeans.org/>

[DIA] - <http://live.gnome.org/Dia>

[SPX] - <http://www.spxservicesolutions.it>

[Caprari] - <http://www.caprari.it/>

[Sincronika] - <http://www.sincronika.it/>

Ringraziamenti

