Dipartimento di Ingegneria dell'Informazione

Corso di laurea in Ingegneria Informatica

Relazione di tirocinio breve

Progettazione e realizzazione di un applicativo per la gestione cespiti per ADP Dealer Services Italia

Laureando: Petru SARBAN

Relatore: Ch.mo Prof. Girolamo GRADENIGO

Data Laurea : 30 settembre 2010

Anno accademico : 2009/2010

28 settembre 2010

Indice

Capitolo 1

Introduzione

1.1 L'Azienda

Il presente lavoro è frutto di un tirocinio svoltosi nel periodo settembre 2008 - febbraio 2009 presso l'azienda Automatic Data Processing - Dealer Services Italia (ADP DSI).

L'Automatic Data Processing è stata fondata nel 1949 da Henry Taub, vicino a New Jersey, USA. I primi servizi offerti dall'azienda consistevano nell'elaborazione dei libri paga, che fra l'altro venivano eseguiti manualmente. Divenne una società per azioni nel 1961,e nel 1974 fu trasformata in una multinazionale con sedi in Olanda e Inghilterra.

Oggi il gruppo ADP fattura oltre 8 miliardi di dollari annui, ha 585000 clienti in tutto il mondo con 40000 dipendenti. È presente nella classifica "Fortune 300" con 27 miliardi di dollari di capitalizzazione (NASDAQ - dati 2008) con un "AAA Credit Rating".

ADP è composta da due business units:

- Employer Services
- Dealer Services

Il ramo Dealer Services nasce nel 1972, con all'attivo circa 300 dipendenti. Dopo una crescita esponenziale, nel 2008, arriva a contare 7300 dipendenti con 1,36 miliardi di dollari di fatturato. Recentemente, all'inizio del 2009, acquista Automaster Oy con sede in Helsinki, Finlandia. Automaster è una società di capitali privata, fondata nel 1983, che supporta i rivenditori di auto e gli importatori con la soluzione software "Automaster". Gode di ottima reputazione sul mercato come fornitore di soluzioni DMS (Database Management Systems), con una forte presenza nei Paesi Scandinavi, in Europa Centrale ed Orientale ed in Russia. Complessivamente, Automaster gestisce

più di 500 clienti in 39 paesi con oltre 20.000 utenti su 30 marchi. In seguito a questa fusione, ADP DS è presente in 94 paesi con oltre 27000 clienti.

La ADP Dealer Services Italia, nata nel 1979, ad oggi conta 250 associati. Attualmente supporta 1600 clienti con più di 22000 utenti unici e gestisce il 51% del mercato italiano gestionali auto. Offre inoltre servizi di hosting dati, noleggio stampanti, noleggio e gestione pc, gestione delle reti informatiche e della sicurezza nonché telefonia avanzata (over ip) e traffico dati.

La sede presso la quale si è svolto il tirocinio, l'unica in Italia, è situata in via Julia 39, Busa di Vigonza, provincia di Padova.

1.2 Il progetto formativo

Il progetto formativo prevede la realizzazione di un sistema per la gestione delle risorse aziendali, hardware e software, con particolare attenzione alla gestione del licensing software. Per licensing si intende la validazione dell'installazione di un particolare software sulle workstation aziendali.

Il progetto formativo è costruito sui seguenti punti cardine:

1. gestione degli utenti (definiti a livello aziendale come associati):
   - gestione dell'inserimento,
   - gestione della cancellazione (disattivazione in quanto si vuole mantenere traccia dei vecchi utenti non più attivi),
   - gestione della modifica
2. gestione delle risorse hardware e software a disposizione:
   - gestione dell'inserimento,
   - gestione della cancellazione,
   - gestione della modifica.
3. gestione del downgrade software: cioè la gestione del licensing improprio di un software (associazione del software con una licenza non propria di tale software)
4. gestione del flusso dell'assegnazione e della dismissione temporanea dei beni aziendali
5. gestione del flusso dell'assegnazione e della dismissione dei beni aziendali

6. gestione della convalida dell'assegnazione e della restituzione dei beni aziendali
7. gestione del kit software assegnato all'associato
8. gestione dei contratti di manutenzione.
9. gestione della ricerca di un item hardware con chiave completa o parziale sul serial number
10. *data migration* informazioni da precedente data base a nuovo ambiente
11. aggiornamento automatico, da file .csv dei canoni mensili di manutenzione hardware
12. gestione reportistica a video, attraverso generazione documenti stampabili ed esportazione su file excel:
    - quadratura licenze installate ed acquistate con evidenza delta e segnale anomalia con attenzione alle casistiche di downgrade software
    - distribuzione di specifico software per utilizzatore
    - distribuzione di specifico hardware per utilizzatore
    - lista anomalie (video/stampa/esportazione su file excel)
      - hardware non assegnato
      - software non assegnato
      - elenco licenze disponibili.

Per quadratura software si intende la corrispondenza fra licenze acquistate e licenze richieste dalle installazioni dei software sulle workstation aziendali. Da parte dell'azienda è stata fatta la richiesta di avere un sistema agevolmente modificabile nel tempo, da poter integrare con sistemi aziendali preesistenti. Una delle modifiche ipotizzate è stata l'integrazione del sistema sviluppato con un sistema di gestione della documentazione che metta in relazione un campo url, riferibile al sistema di gestione della documentazione, ad un evento modellato nell'applicazione.

La soluzione sviluppata è stata integrata nel sistema informativo aziendale; in particolare, il software sviluppato permette l'autenticazione degli utenti avallata dal sistema informativo aziendale attraverso il protocollo LDAP.

L'attenzione particolare che viene richiesta nella gestione del licensing è dovuta alla particolare tipologia di azienda in questione. A parte il profilo

legale che tale gestione impone, si ha un ulteriore interesse nel conoscere approfonditamente, e con un certo livello di digitalizzazione, le proprie risorse software. Questo è dovuto ai vari rapporti che l'azienda ha a livello di partnership con fornitori di soluzioni IT come Microsoft e IBM. In particolare ADP è un Microsoft Gold Certified Partner ed un IBM Business Partner. Questi accordi permettono di godere di forti vantaggi a livello economico nell'acquisto degli strumenti che queste aziende a loro turno forniscono o utilizzano nella normale routine aziendale.

1.3 La realtà aziendale.

Il tirocinio in questione è stato sviluppato nell'ambito del gruppo Internal Information Services (IIS), entità che si occupa della gestione delle risorse informatiche all'interno dell'azienda, dalla configurazione dei personal computer alla gestione ed ampliamento della rete informatica interna. Nonostante questo, il lavoro è stato sviluppato in autonomia dal tirocinante. Le interazioni sono avvenute nell'analisi della realtà aziendale da modellare, nella sintesi di alcuni requisiti e nella selezione delle scelte di progettazione.

La struttura aziendale predilige gruppi di persone ai quali vengono assegnati differenti mansioni. Alcuni sono adibiti alla produzione vera e propria del business dell'azienda mentre altri al mantenimento della stessa struttura. Ogni gruppo ha uno o più amministratori atti a dirigere il lavoro dello stesso. Ad ogni associato all'interno dell'azienda, per il proprio lavoro, viene assegnato un insieme di risorse, che quasi sempre comprendono un personal computer (fisso e/o laptop), con il necessario corredo software per il funzionamento. Ulteriori risorse che possono essere assegnate, in funzione del profilo dell'utente, possono essere telefoni cellulari o fissi, data card, stampanti portatili, proiettori o strumenti di rete come router o switch. Gli ultimi, anche se usati nell'infrastruttura aziendale, e non solamente dall'utente assegnatario, vengono associati ai responsabili di tali apparecchiature.

All'atto di commissione del progetto esisteva un software che gestiva una semplice anagrafica degli utenti, un database con i software ed i hardware disponibili in azienda, le informazioni accessorie a tali elementi nonché una limitata gestione delle associazioni fra di essi. In pratica si parla di un'applicazione web-based sviluppata utilizzando Microsoft ASP che si basa su un database realizzato in Microsoft Access.

La realizzazione dello strumento preesistente è stata fatta ad un livello che potrebbe essere considerato ludico per quanto riguarda la parte del data base in quanto la soluzione utilizzata non sfrutta i concetti che ha reso l'adozione dei Data Base Management Systems (DBMS) un obbligo per un

efficiente funzionamento. Infatti la vecchia base di dati usa delle relazioni non correlate da vincoli di chiave esterna, nonostante si riferiscano l'un l'altra e ancor meno usa vincoli di chiave primaria sui dati. Di conseguenza non si poteva avere una garanzia sulla coerenza dei dati presenti. La verifica veniva eseguita manualmente dall'operatore. A complicare la situazione c'era anche la mancanza di documentazione a supporto dello schema sia della base di dati che dell'applicativo.

Logicamente il tool gestionale sviluppato e codificato fortemente sulla struttura della base di dati dando poche possibilità di modifica agevole.

In seguito a queste osservazioni si è chiaramente vista la necessità della completa ingegnerizzazione dello strumento, tenendo in considerazione schemi di funzionamento dei prodotti già esistenti; questo in quanto già utilizzati sul campo. La struttura della vecchia base di dati è stata utile per la determinazione delle informazioni da gestire nel mini-mondo. Allo stesso tempo è stata anche un vincolo, in quanto le informazioni del vecchio database, rappresentano la situazione, per quanto parziale, del mini-mondo, e dovevano essere portate nel nuovo modello. Questo ha appesantito e condizionato la struttura finale dell'applicazione in quanto a livello aziendale non c'era la disponibilità nell'integrare le informazioni mancanti tanto meno per una rigorosa verifica della coerenza degli stessi.

Capitolo 2

Progettazione base di dati

2.1 Progettazione concettuale

2.1.1 Analisi dei requisiti

Il principale patrimonio aziendale è l'insieme degli associati (utenti) e dalle risorse a loro disposizione.

Gli utenti hanno un account all'interno del sistema informatico aziendale con delle credenziali d'accesso. Si vuole utilizzare la procedura d'accesso di tale sistema in quanto, con ottima probabilità, più robusto dal punto di vista della sicurezza e meno oneroso in termini di lavoro. Gli utenti dal punto di vista di gestione dell'applicativo si dividono in utenti semplici, che possono visualizzare solamente dati in funzione del ruolo, e utenti amministratori che possono anche modificare la rappresentazione del mini-mondo. Gli utenti visualizzano dati in funzione del ruolo che rivestono nei gruppi. In particolare un utente amministratore visualizza dati inerenti alla propria situazione ed alla situazione degli utenti facenti parte dei gruppi nei quali ha ruolo di amministratore. La partecipazione degli utenti ai gruppi non è esclusiva: un utente può partecipare a diversi gruppi con ruoli differenti. Un utente può anche non partecipare a nessun gruppo.

Degli utenti vogliamo memorizzare le seguenti informazioni: il nome utente di login al sistema aziendale, la data d'inserimento del nominativo ed il nome ed i cognome dello stesso per una lettura più agevole. Un utente una volta inserito nel database non deve essere eliminato, anche in caso di scomparsa dal mini-mondo, ma disabilitato in quanto si vuole mantenere una cronologia delle informazioni inserite nel tempo. Un utente disabilitato può essere riabilitato nel caso si abbia un reintegro dello stesso.

Gli utenti vengono riuniti in gruppi che possono essere strutture realmente esistenti (esempio: ufficio amministrativo, ufficio vendite ecc.) al interno

dell'azienda oppure virtuali come raggruppamenti territoriali. La partecipazione al gruppo avviene con ruoli diversi: partecipazione semplice oppure partecipazione con ruolo di responsabilità per i manager del gruppo. Per ogni gruppo di utenti si vuole anche memorizzare il nome del gruppo e la data di creazione.

Le risorse di cui si vuole tenere traccia sono di due tipi: l'hardware consegnato fisicamente all'utente e le licenze associate ai software installati/associati. Si nota che non tutti gli hardware hanno necessariamente dei software associati.

Le risorse hardware sono di diverso tipo. Possono essere cellulari, pc laptop, pc fissi, switch, stampanti, router, scanner, supporti di memorizzazione ecc. Per identificare questi elementi vengono utilizzati diversi parametri in funzione del periodo temporale nel quale sono stati introdotti in azienda. Alcuni sono identificati attraverso una targhetta alfanumerica ([reparto].[tipo hardware].[indice numerico sequenziale]), altri in funzione del type model e serial number definiti dal produttore, presenti su tutti i prodotti attualmente in commercio. Si vogliono memorizzare ulteriori informazioni come una descrizione commerciale del modello, il numero di mesi per il quale viene garantito il prodotto dal venditore, la data di acquisto, la data dell'inserimento dell'item nel archivio, la tipologia di garanzia che il venditore offre (on site, carry in) e il marchio (che viene volutamente confuso con il produttore). Le risorse hardware hanno una vita limitata. Si vuole memorizzare anche l'eventuale data di dismissione e lo stato finale della risorsa (rottamata, rivenduta ecc). Una risorsa hardware può essere sottratta alla vita aziendale per un intervento di manutenzione temporaneo oppure sostituita in garanzia o attraverso interventi di manutenzione. Si evince dalla realtà aziendale che per alcuni item si registra anche la destinazione d'uso.

Le risorse software sono la collezione di licenze che l'azienda ha nel suo portafoglio. Il software senza una licenza che ne giustifichi l'installazione è irrilevante e illegale. Si vuole avere un'anagrafica dei software utilizzati in azienda. Per ogni software si vuole memorizzare il nome, la data di inserimento nell'anagrafica, il produttore, una stringa con l'URL (Uniform Resource Locator) del produttore o eventualmente del software. Si vuole anche avere l'informazione sulla possibilità o meno di acquistare licenze per tale software. Alcuni software presentano condizioni di licenza diversa in funzione del pacchetto linguistico associato, indi sarà necessario tenere conto del pacchetto linguistico caricato. Si vuole avere anche un flag che indichi se l'item è attivo ed utilizzabile oppure disabilitato.

Per validare le installazioni dei software bisogna associare all'installazione una licenza. Difficilmente vengono acquisite licenze singole in quanto la maggior parte dei software sono standard e vengono utilizzati da più utenti,

indi la consuetudine aziendale di chiamare le collezioni di licenze pool (consuetudine che verrà mantenuta per comodità).
Ogni pool di licenze acquistato abilita un ben definito numero di installazioni ed ha un costo. Un pool di licenze abilita lo stesso software. Il costo viene associato alla licenza singola.
Le licenze possono venire assimilate attraverso canali differenti: possono provenire dalla casa madre, che acquisisce a livello mondiale i pacchetti software comuni; a livello europeo (dalla sede con base nel Regno Unito), o ancora a livello locale, direttamente attraverso l'ufficio acquisti. La provenienza delle licenze è importante nella quadratura fra la licenze richieste e le possedute. Le licenze hanno tipologie diverse: possono essere OEM (Original Equipment Manufacturer) per indicare il legame insolubile con l'hardware associato, personali cioè fornite dagli utenti stessi ecc. Si dovrà tenere conto di un campo chiamato "tipo licenza". Alcune licenze hanno una scadenza temporale indi si vuole tenere conto della data di acquisto, di scadenza e di inserimento nel database. Una licenza può abilitare l'installazione anche di versioni diverse da quelle per la quale è stata rilasciata (fenomeno che viene chiamato downgrade a livello aziendale). Solo alcuni software possono prestarsi a donare licenze ad altri software. Il downgrade si può applicare anche con software che differiscono non solo nella versione ma anche nel nome .

Alcuni software per verificare che ci sia una licenza associata richiedono una stringa di conferma chiamata product key (pk). La pk viene verificata dal software all'atto dell'installazione normalmente. Alcuni produttori utilizzano un'unica strigna per più licenze, mentre altri individuano la singola licenza con una pk. Si vuole tenere traccia di questo comportamento in quanto lo smarrimento del product key può invalidare la licenza. Per identificare pool di licenze diversi viene anche usata una stringa alfanumerica univoca ottenuta attraverso funzioni di hash, campo chiamato "objectGuid".

Si vuole poi essere in grado anche di tenere traccia del numero di licenze necessarie in azienda per la validazione del parco installazioni. Si vuole associare ad ogni installazione di software il numero di licenze che tale installazione impiega. Verrà periodicamente fatta una verifica, chiamata quadratura, che vada a verificare che il numero di licenze richieste dalle installazioni registrate trovi riscontro nel pool licenze in possesso. La verifica viene fatta manualmente.

Ad alcune risorse hardware, prevalentemente personal computer, vengono associate delle risorse software attraverso il processo d'installazione. Normalmente l'installazione dei software avviene in gruppo, dato che è un processo prevalentemente eseguito prima dell'assegnazione all'utente. Si vuole modellare anche la situazione nella quale si associano a certe risorse hardware delle macchine virtuali che altro non sono che un raggruppamento di software. Le

macchine virtuali hanno delle denominazioni e possono essere installate su diversi sistemi, come anche i gruppi di software che vengono installati. Ad ogni installazione si vuole associare un numero di licenze che abilita la stessa per ogni software. Il numero di licenze richiesti da una installazione può essere diverso da 1 in quei sistemi adibiti a server dove i programmi vanno usati da più utenti contemporaneamente.

Alcune risorse hardware sono coperte da contratti di manutenzione. I contratti di manutenzione sono delle polizze d'assicurazione, di durata tipicamente annuale, che specificano un costo unitario per ogni risorsa hardware coperta. Questi contratti vengono stipulati solo per una parte delle risorse aziendali. I contratti di manutenzione possono avere diverse caratteristiche molto variabili nel tempo. Offrono diversi servizi in funzione della risorsa hardware. Alcune risorse hanno delle garanzie (Service Level Agreement) di servizio. Le garanzie solitamente consistono in un tempo massimo entro il quale avviene l'intervento di manutenzione. Per ogni contratto di manutenzione si vuole memorizzare la data di stipula, la data di scadenza della polizza e l'azienda che garantisce la polizza.

Ogni risorsa nella realtà aziendale ha un fornitore che può essere un venditore, un fornitore di assistenza oppure un produttore di hardware. Per ogni fornitore ci interessa conoscere una denominazione ed i suoi referenti. Ogni referente ha un nome, una sede di cui si vuole memorizzare l'indirizzo ed un campo in cui si vogliono inserire annotazioni.

Ad ogni utente vengono assegnate una o più risorse. Le risorse assegnate sono sempre risorse hardware. Delle assegnazioni si vuole tenere una traccia temporale anche quando vengono dismesse; di conseguenza avranno un identificatore univoco nel tempo. L'assegnazione di una risorsa nasce automaticamente, a livello aziendale, per esempio appena dopo l'assunzione, oppure a richiesta dell'associato interessato ad ampliare le proprie risorse lavorative. L'assegnazione avviene in due fasi: al primo livello c'è l'ufficio IIS che valuta la necessità della richiesta (potendo nel caso anche rifiutarla), in quanto gestore delle risorse aziendali. Successivamente l'ufficio amministrativo registra in formato cartaceo l'evento.

Il flusso di assegnazione inizia con la richiesta dell'associato, che successivamente deve essere approvata dal responsabile del gruppo di appartenenza. Tale richiesta poi viene presentata all'ufficio IIS, che in funzione della richiesta, valuta le risorse più adatte da assegnare e propone una soluzione tecnica approfondita. Una volta accettata la soluzione dalla parte richiedente l'assegnazione viene convalidata dal IIS. Perché il flusso d'assegnazione si consideri concluso la parte richiedente deve espletare le formalità amministrative di registrazione: in particolare deve compilare una richiesta cartacea che verrà firmata e conservata nell'archivio dell'azienda. Dopo l'approvazione

amministrativa la risorsa richiesta viene consegnata fisicamente presso l'ufficio IIS. Il processo di dismissione avviene inizialmente presso l'ufficio IIS, che una volta verificata l'integrità della risorsa hardware avvia il processo di dismissione che verrà concluso presso l'ufficio amministrativo. Come da richiesta questo flusso deve essere in parte modellato a livello applicativo in modo da facilitare il procedimento. Infatti, per quanto la procedura sia semplice, una gestione non controllata porta a notevole dispendio di tempo sottratto ad altre attività, nonché ad un elevato rischio di errori di registrazione. Sovente ci si imbatte infatti in casi di risorse alienate senza registrazione o di realtà non correttamente modellate (situazioni che in termini di tempo assorbono anche giornate intere di lavoro). Un'assegnazione può assumere diversi significati: un utente può avere assegnata una risorsa per l'utilizzo nelle sue mansioni, come responsabile di tale risorsa (esempio: un server di prova assegnato al responsabile del gruppo ingegneria atto ad essere usato come demo, viene usato da tutti gli utenti ma l'assegnazione può essere fatta al responsabile del gruppo) oppure come sostituto nel caso il responsabile non sia presente fisicamente in azienda. Un'assegnazione si sviluppa in un intervallo temporale e nel momento di avvio può avere già previsto un momento di dismissione. Per alcune assegnazioni la supposta fine può essere trascurata mentre per altre si vuole rispettare l'ipotesi di fine. Si vuole mantenere una traccia temporale del processo di assegnazione, quindi delle date di accettazione della richiesta di assegnazione, della registrazione amministrativa, dell'effettiva consegna della risorsa, della consegna della risorsa per il processo di dismissione e dell'effettiva dismissione della risorsa.

2.1.2 Specifica dei requisiti - Diagramma E-R

L'associazione principale che emerge dall'analisi si può osservare nella figura 2.1.

Si può raffinare il modello dell'entità risorsa attraverso il processo di specializzazione disgiunta, valutato sulla tipologia della risorsa, in altre tre entità: RISORSA_HW, RISORSA_SW, POOL_LICENZE, ognuna con i propri attributi. Il raffinamento è visibile nella figura 2.2.

Si nota, nel diagramma, che tutte le entità risultanti dalla specializzazione presentano un campo riguardante l'entità fornitore; in due casi il legame indica un venditore mentre nell'altro un produttore. Tali attributi convergeranno nel diagramma finale in relazioni con l'entità FORNITORE.

Una complessità maggiore si è riscontrata nel modellare il processo di installazione. Infatti, in un'installazione si va ad associare ad un particolare hardware un certo raggruppamento di software oppure un particolare software. Si decide di considerare l'installazione come avvenimento della sola pri-

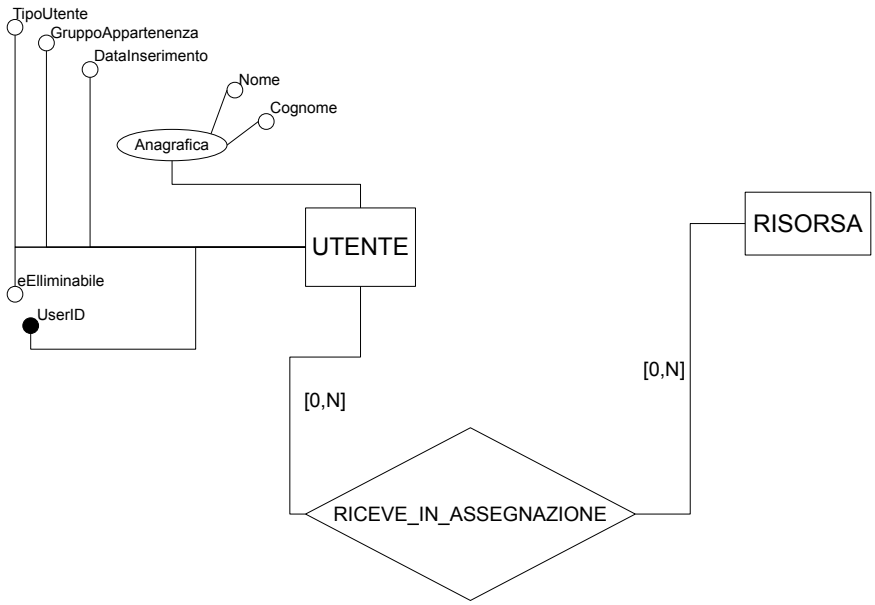


Figura 2.1: Relazione principale modello.

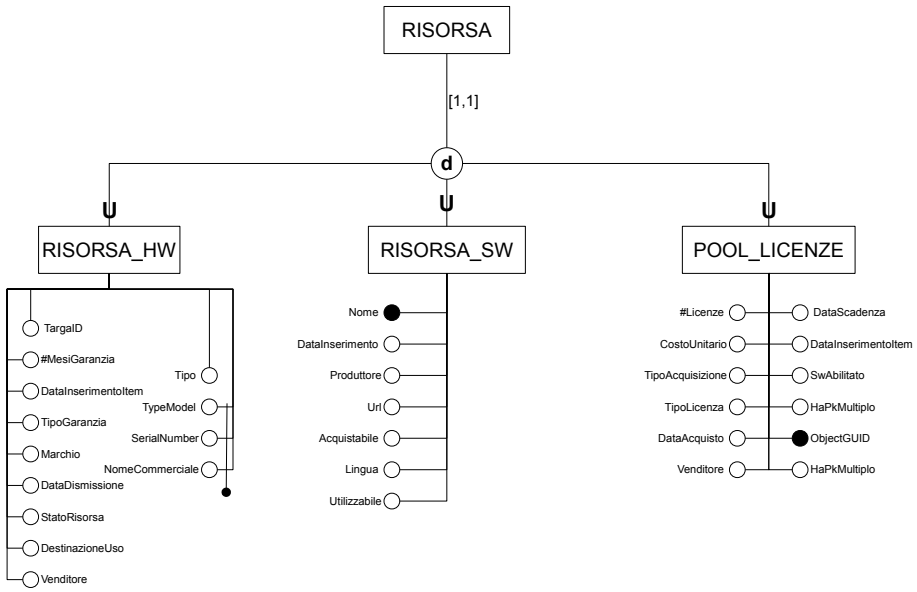


Figura 2.2: Specializzazione risorse.

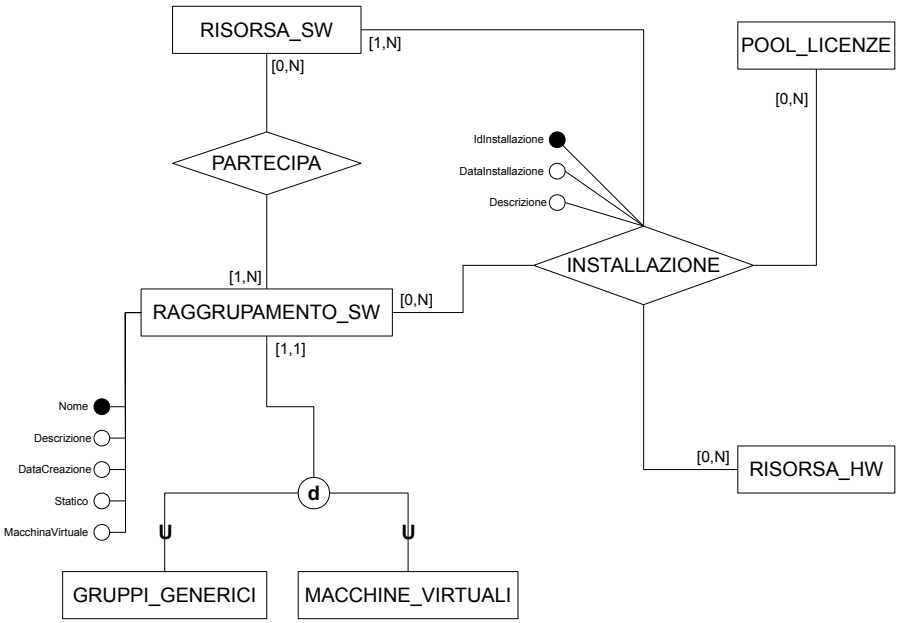


Figura 2.3: Raffinamento installazione

ma casistica. Per l'installazione di un singolo software verrà usato un gruppo contenitore formato dal solo software in questione. Questa ipotesi ci consente di semplificare la struttura logica del processo, utile poi nello sviluppo dell'applicativo, e permette anche di risparmiare spazio in memoria. Modellare l'installazione come associazione fra la risorsa hardware e il particolare software aumenterebbe notevolmente il numero di entità installazione da memorizzare. Secondariamente questa scelta ci permette di gestire agevolmente il caso di installazione di macchine virtuali in quanto per identificare una tale entità basta avere un attributo che indichi l'evenienza. Di conseguenza viene modellata un'ulteriore entità di nome RAGGRUPAMENTO_SW, implicata direttamente nell'associazione INSTALLAZIONE. Il raggruppamento si definirà statico quando la sua composizione, indi la collezione di software, non può essere modificata. Definizione utile sia per modellare i contenitori (che non sono altro che dei gruppi statici che contengono un solo elemento e hanno lo stesso nome dell'elemento contenuto) che per indicare eventuali controindicazioni alla modifica della struttura del gruppo. Omettendo gli attributi delle entità già note otteniamo il diagramma parziale visibile in figura 2.3.

L'associazione INSTALLAZIONE è del quarto grado. Questo è dovuto al

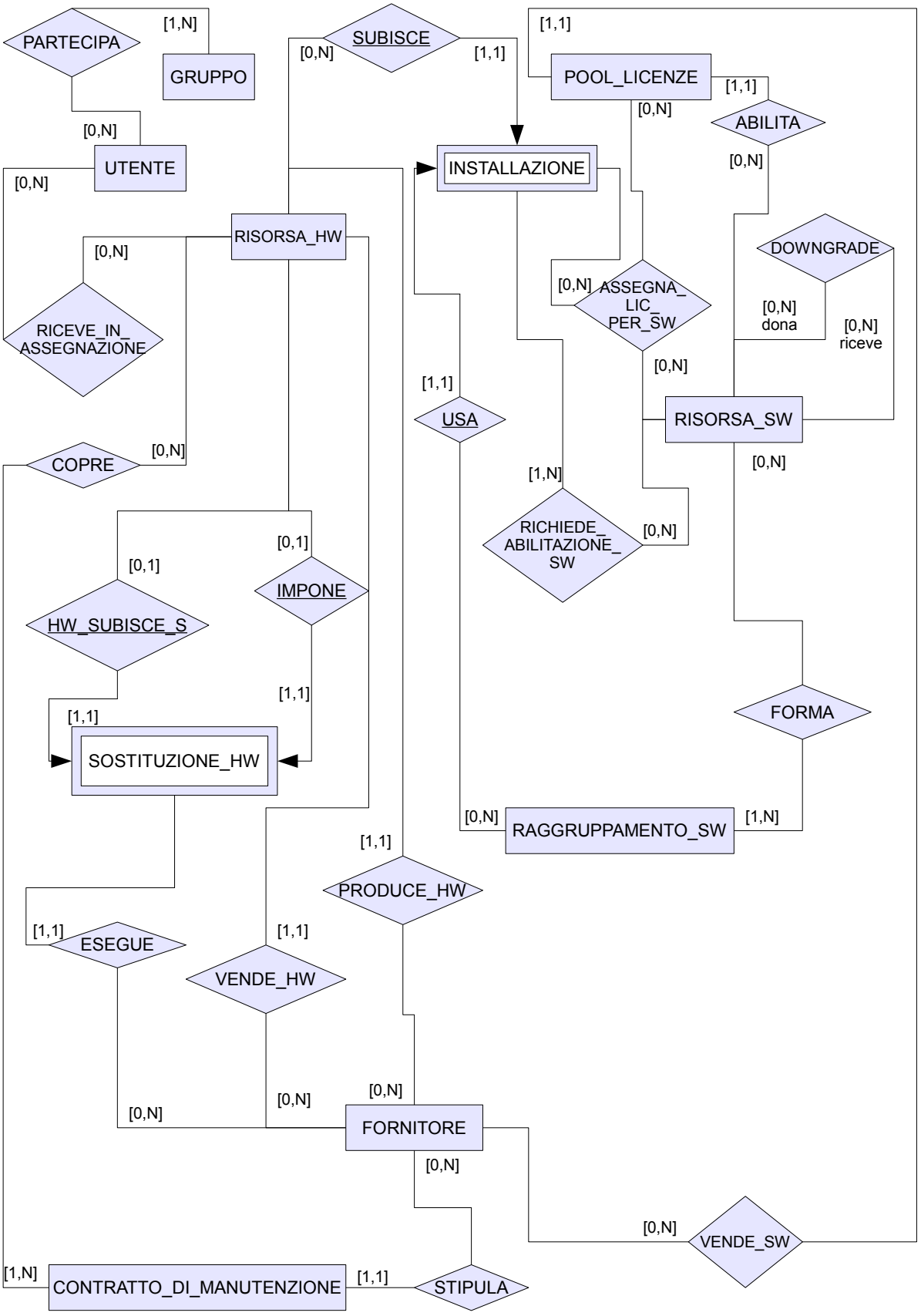


Figura 2.4: Modello E-R.

fatto che l'installazione lega il pacchetto software alla particolare macchina, giustificando l'installazione con una licenza che può essere impropria, per abilitare un determinato software, che non necessariamente è quello che per definizione la licenza abilita da definizione di licenziazione impropria. Per semplificare il modello possiamo pensare INSTALLAZIONE come un'entità debole identificata dalle relazioni USA (associazione fra RAGGRUPAMENTO_SW e INSTALLAZIONE) e SUBISCE (associazione fra RISORSA_HW e INSTALLAZIONE). La nuova entità INSTALLAZIONE sarà comunque collegata a POOL_LICENZE e a RISORSA_SW da un'associazione del terzo grado denominata ASSEGNA_LIC_PER_SW, che lega logicamente le tre entità riuscendo a modellare correttamente anche la casistica di downgrade nella licenziazione.

Si può a questo punto esprimere uno schema riassuntivo del modello complessivo, visibile nella figura 2.4. Lo schema è stato ristrutturato ed integrato con le relazioni mancanti. Vengono omessi gli attributi per salvaguardare la facilità di lettura.

Le entità deboli vengono rappresentate con il formalismo di [1][cap. 3], mentre le associazioni identificanti vengono rappresentate con il proprio nome sottolineato e collegate all'entità debole con una freccia. Per la completa specificazione dello schema si vedano le tabelle 2.1 e 2.2 riassuntive rispettivamente degli attributi di ogni entità e di ogni associazione con la denominazione finale.

Entità	Attributo	Descrizione
UTENTE chiave: **userID**	userID	Descrive il nome utente nel sistema informatico aziendale.
	Anagrafica (Nome,Cognome)	Attributo composto che descrive il nome ed il cognome anagrafico dell'associato.
	eUtenteLogico	Indica se l'utente è logico oppure reale
	eUtenteAttivo	Indica se l'utente è attivo oppure disattivato
	eElliminato	Indica se l'utente fa parte della schiera degli utenti attivabili oppure è eliminato.
	tipoUtenteAmministrativo (nome, descrizione)	Attributo composto che che indica il tipo di utente dal punto di vista di controllo sul applicativo
	dataInserimento	Indica la data in cui l'utente è stato inserito per la prima volta nel database.
GRUPPO chiave: **nomeGruppo**	nomeGruppo	Indica il nome del gruppo
	tipoGruppo (nome, descrizione)	Attributo composto che indica la tipologia di divisione che il raggruppamento rappresenta
	dataCreazione	Indica la data in cui il gruppo è stato creato
RISORSA_HW chiave: **(typeModel , serialNumber, nomeCommerciale)**	targaID	Indica il nome che viene usato sulle targhetta identificativa di alcune risorse hardware
	destinazioneDUso (nome, descrizione)	Attributo composto che indica la destinazione d'uso per la quale è dedicato l'hardware
	tipoHw (nome, descrizione)	Attributo composto che indica la tipologia della risorsa hardware
	modelloCommerciale	Indica il modello commerciale della risorsa hardware
	typeModel	Indica il type model associato dal produttore al particolare modello della risorsa hardware

	serialNumber	Indica il serial number univoco associato dal produttore alla particolare risorsa hardware. Identifica un oggetto a livello mondiale insieme al type model ed al produttore.
	garanziaOriginaleMesi	Indica il numero di mesi per i quali il produttore garantisce il prodotto, determinabile insieme alla data di acquisto.
	tipoGaranzia (nome , descrizione)	Indica la tipologia di garanzia che il venditore offre.
	dataInserimento	Indica la data dell'inserimento del item
	dataCessazioneEsistenza	Indica la data in cui la risorsa ha cessato la propria esistenza
	dataAcquisto	Indica la data di acquisto della risorsa.
	statoHw (nome, descrizione)	Attributo composto che indica lo stato hardware in cui la risorsa si trova.
RISORSA_SW chiave: **nomeSW**	nomeSW	Indica il nome commerciale del software
	dataInserimento	Indica la data d'inserimento nella base di dati.
	url	Indirizzo internet del produttore o del software.
	acquistabile	Flag che indica se il software è acquistabile o meno.
	lingua (nome, descrizione)	Attributo composto che indica il pacchetto linguistico associato al software.
RAGGRUPPAMENTO_SW chiave: **nome**	nome	Nome del raggruppamento software.
	descrizione	Descrizione del gruppo.
	dataCreazione	Indica la data in cui è stato creato il raggruppamento.
	gruppoStatico	Flag che indica una composizione del gruppo non modificabile.
	macchinaVirtuale	Flag che indica se un gruppo rappresenta una macchina virtuale.
FORNITORE chiave: **denominazione**	denominazione	Nome del fornitore.
	tipoFornitura (nome, descrizione)	Attributo composto che indica le tipologie di servizi vengono forniti dal fornitore.
	{ referenti (nome , indirizzo, note) }	Attributo multiplo e composto che indica i referenti del fornitore.
POOL_LICENZE chiave: **objectGUID**	numeroLicenze	Indica il numero di licenze che il pool abilita.
	costoUnitarioLicenza	Indica il costo unitario di ogni licenza.
	tipoAcquisizione (nome, descrizione)	Attributo composto che indica il canale di acquisizione del pool.
	tipoLicenza (nome, descrizione)	Attributo composto che indica il tipo della licenza.
	dataAcquisizione	Indica la data di acquisizione.
	scadenza	Indica la scadenza delle licenze.
	dataInserimento	Indica la data di inserimento nella base di dati.
	productKeyMultiplo	Flag che indica se le licenze hanno un product key ognuna oppure se è presente uno generico.
	objectGUID	Codice alfanumerico per la identificazione delle licenze.
CONTRATTO_MANUTENZIONE chiave: **idContratto**	idContratto	Identificatore del contratto. Viene usato un campo numerico.
	dataStipula	Indica la data di stipula del contratto.
	dataScadenza	Indica la data di scadenza del contratto.
	dataDisdetta	Indica la data di disdetta del contratto.
INSTALLAZIONE chiave parziale: **idInstallazione**	dataInstallazione	Indica la data nella quale è stata fatta l'installazione.
	idInstallazione	Rappresenta una chiave parziale dell'installazione.
SOSTITUZIONE chiave: *identificata dall'entità forte RISORSA_HW*	data	Data nelle quale avviene la sostituzione
	motivazione	Contiene una breve motivazione per la sostituzione.
	descrizione	Contiene una descrizione maggiormente dettagliata per l'evento

Tabella 2.1: Modello ER – Entità – Tabella riassuntiva degli attributi.

Relazione	Entità	Attributo	Descrizione
PARTECIPA	UTENTE [0,N] GRUPPO [1,N]	ruolo (nome, descrizione)	Attributo composto che indica con che ruolo l'utente partecipa al gruppo
		data	Indica la data dalla nella quale si è attivata la partecipazione
RICEVE_IN_ASSEGNAZIONE	UTENTE [0,N] RISORSA_HW [0,N]	dataInserimento	Indica la data di inserimento nel sistema
		dataPreassegnazione	Indica la data di preassegnazione della risorsa
		dataPreassegnazioneA	Indica la data di preassegnazione amministrativa della risorsa.
		dataAssegnazione	Indica la data di assegnazione fisica della risorsa.
		dataScadenza	Indica la data ipotetica di dismissione dell'assegnazione.
		dataPreDismissione	Indica la data di avviamento del processo di dismissione. Coincide con il momento in cui la risorsa viene riconsegnata in ufficio IIS per la verifica
		dataDismissione	Indica la data in cui la dismissione diviene operativa.
		tipoAssegnazione (nome, descrizione)	Attributo composto che indica il significato dell'assegnazione (utilizzo, responsabilità, acting(sostituto al ruolo principale) responsabilità)
		scadenzaImportante	Flag che indica se la scadenza è particolarmente importante.
COPRE	CONTRATTO_MANUTENZIONE [1,N] RISORSA_HW [0,N]	dataInizioCopertura	Indica la data nella quale parte la copertura della particolare risorsa
		costoManutenzioneMese	Indica il costo di manutenzione della risorsa riferito ad un singolo mese.
		dataFineCopertura	Indica la data nella quale la copertura scade
		tipoManutenzione (nome, descrizione, SLA)	Attributo composto che indica la tipologia di manutenzione applicata alla particolare risorsa
HW_SUBISCE_S	RISORSA_HW [0,1] SOSTITUZIONE_HW [1,1]		
IMPONE	RISORSA_HW [0,1] SOSTITUZIONE_HW [1,1]		
ESEGUE	FORNITORE [0,N] SOSTITUZIONE_HW [1,1]		
SUBISCE	RISORSA_HW [0,N] INSTALLAZIONE [1,1]		
VENDE_HW	FORNITORE [0,N]] RISORSA_HW [1,1]		

PRODUCE_HW	FORNITORE [0,N]] RISORSA_HW [1,1]		
STIPULA	FORNITORE [0,N]] CONTRATTO_MANUTENZIONE [1,1]		
USA	INSTALLAZIONE [1,1] RAGRUPPAMENTO_SW [0,N]		
VENDE_SW	FORNITORE [0,N] RISORSA_SW [1,1]		
FORMA	RISORSA_SW [0,N] RAGRUPPAMENTO_SW [1,N]		
ABILITA	POOL_LICENZE [1,1] RISORSA_SW [0,N]		
DOWNGRADE	RISORSA_SW [0,N] - *dona* RISORSA_SW [0,N]- *riceve*	transitività	Flag che indica se il potere di downgrade si applica anche ai figli del nodo.
RICHIEDE_ABILITAZIONE_SW	INSTALLAZIONE [1,N] RISORSA_SW [0,N]	numeroLicenzePrenotate	Indica il numero di licenze che la particolare installazione necessita per il software.
ASSEGNA_LIC_PER_SW	POOL_LICENZE [1,1] RISORSA_SW [0,N] POOL_LICENZE [1,1]	numeroLicenzeInstallate	Indica il numero di licenze che la particolare installazione ha associato per il particolare software.

Tabella 2.2: Modello ER – Associazioni – Tabella riassuntiva degli attributi.

2.1.3 Specifica dei requisiti funzionali

Le principali funzioni che l'applicativo deve espletare, escluse quelle ovvie di inserimento, modifica, eliminazione e interrogazione di ogni entità, sono descritte nel progetto formativo. In particolare possiamo evidenziare le seguenti:

1. **DWD**: Definizione di downgrade software : indicazione della relazione (donatore-ricevente) di downgrade.
2. **DWU**: Utilizzazione della relazione di downgrade software in un'abilitazione di un'installazione.
3. **QD**: Quadratura delle licenze software.
4. **DSWU**: Distribuzione di specifico software per utilizzatore.
5. **DHWU**: Distribuzione di specifico hardware per utilizzatore.

6. **INST**: Registrazione di nuova installazione.
7. **GUS**: Operazioni tipiche su utente.
8. **GHW**: Operazioni tipiche su hardware.
9. **GSW**: Operazioni tipiche su software.
10. **GPOL**: Operazioni tipiche su licenze.

Normalizzando i valori, si possono definire dei gradi d'intensità (molto basso, basso, medio, alto) per quanto riguarda frequenza delle operazioni. Possiamo quindi esplicitare una mappa della frequenza delle funzioni principali, visibile nella tabella 2.3.

Operazione	Grado di Frequenza
DWD	molto basso
DWU	medio
QD	basso
DSWU	basso
DHWU	basso
INST	alto
GUS	alto
GHW	alto
GSW	alto
GPOL	basso

Tabella 2.3: Frequenza delle operazioni più rilevanti

2.1.4 Analisi carico applicativo

All'atto di realizzazione del progetto si possono osservare i volumi elencati nella tabella 2.4.

Oggetto	Tipo	Frequenza
UTENTE	E	285
GRUPPO	E	23
RISORSA_HW	E	1098
RISORSA_SW	E	604
RAGGRUPPAMENTO_SW	E	690
FORNITORE	E	38
POOL_LICENZE	E	381

CONTRATTO_MANUTENZIONE	E	5
INSTALLAZIONE	E	3202
SOSTITUZIONE	E	7
PARTECIPA	A	492
RICEVE_IN_ASSEGNAZIONE	A	1098
COPRE	A	50
HW_SUBISCE_S	A	7
IMPONE	A	7
ESEGUE	A	7
SUBISCE	A	3202
VENDE_HW	A	1098
PRODUCE_HW	A	1098
STIPULA	A	5
USA	A	3202
VENDE_SW	A	604
FORMA	A	690
ABILITA	A	381
DOWNGRADE	A	10
RICHIEDE_ABILITAZIONE_SW	A	3204
ASSEGNA_LIC_PER_SW	A	0

Tabella 2.4: Tabella riassuntiva dei volumi.

Alcuni oggetti hanno un volume diverso da quello rilevabile a regime, in quanto di alcune informazioni non viene tenuta una traccia temporale, al contrario di quello che si vuole ottenere.

2.2 Progettazione logica - Modello Relazionale

2.2.1 Traduzione nel modello relazionale

Per la traduzione nel modello relazionale utilizziamo l'algoritmo descritto in [1][sez. 7.1] applicato al modello E-R riassunto nella figura 2.4 e descritto dettagliatamente nella tabella 2.1.

Regole di traduzione.

Per la traduzione dello schema vengono seguite le seguenti regole generali:

1. *Per ogni entità (forte) E nello schema ER, si costruisce una relazione R che contenga tutti gli attributi semplici di E. Di un attributo composto si*

inseriscono solo gli attributi componenti semplici. Si sceglie come chiave primaria per R uno degli attributi chiave di E. Se la chiave scelta per E è composta, l'insieme di attributi semplici che la compongono formano nel complesso la chiave primaria di R.

2. *Per ogni tipo di entità debole W dello schema ER con tipo di entità proprietario E, si costruisce una relazione R e si inseriscono tutti gli attributi semplici, o componenti semplici di attributi composti, di W come attributi di R. Inoltre si inseriscono come attributi di chiave esterna di R gli attributi di chiave primaria delle relazioni corrispondenti ai tipi di entità proprietari. La chiave primaria di R è data dalla combinazione delle chiavi primarie delle relazioni proprietarie e dalla eventuale chiave parziale dell'entità debole.*

3. *Per ogni attributo multivalore A si costruisce una nuova relazione R. La relazione R comprende un attributo corrispondente ad A, più l'attributo di chiave primaria K (come chiave esterna di R) della relazione che rappresenta il tipo di entità o il tipo di associazione che ha A come attributo. La chiave primaria di R è data dalla combinazione di A e K. Nel caso di attributo multivalore composto si considerano le sue componenti semplici.*

4. *Se un'entità E partecipa ad un'associazione binaria A, con vincoli [1,1], allora alla relazione R costruita per tale entità vengono aggiunti gli attributi di A ed un riferimento alla chiave primaria (come chiave esterna) della controparte F legata ad E dall'associazione.*

5. *Per ogni tipo di associazione R binaria di tipo M:N, cioè E – [0,N] – R – [1,N] –F oppure E – [0,N] – R – [0,N] – F, si costruisce una nuova relazione S che rappresenta R. Si inseriscono come attributi di chiave esterna di S le chiavi primarie delle relazioni che rappresentano i tipo di entità partecipanti; le loro combinazioni formano la chiave primaria di S. Si inseriscono fra gli attributi di S anche tutti gli attributi semplici del tipo di associazione.*

6. *Per le associazioni R di grado maggiore di 2, nel caso in esame al massimo di grado 3, si costruisce una nuova relazione S. Si inseriscono in S come attributi di chiave esterna le chiavi primarie delle relazioni che rappresentano i tipi di entità partecipanti. Si inseriscono come attributi di S anche tutti gli attributi di R. La chiave primaria della relazione risultante S è la combinazione di tutte le chiavi esterne che riferiscono le relazioni rappresentanti i tipi di entità partecipanti ad esclusione di quelle che partecipano con vincolo di cardinalità massima unitario.*

Lo schema relazionale ottenuto dalla traduzione viene descritto nella tabella 2.5. Nel caso in osservazione si nota che per l'entità UTENTE l'attributo *Anagrafica(Nome,Cognome)* anche se tradotto in un unico attributo *nomeCognomeAnagrafico* non presenta una perdita rilevante d'informazione in quanto l'anagrafica verrà usata solo per un'agevole lettura dell'operatore.

Relazione	Attributo	Riferisce	Note
Utenti	**userID**		chiave primaria
	nomeCognomeAnagrafico		traduce emphAnagrafica
	eUtenteLogico		
	eUtenteAttivo		
	eElliminato		
	tipoUtenteAmministrativo	_tipoUtente.nome	chiave esterna
	dataInserimento		
Gruppi	**nomeGruppo**		
	tipoGruppo	_tipoGruppo.nome	chiave esterna
	dataCreazione		
RisorsaHW	idTID		traduce *targaID*
	destinazioneDUso	_destinazioneUso.nome	chiave esterna
	tipoHw	_tipoHW.nome	chiave esterna
	modelloCommerciale		
	typeModel		
	serialNumber		
	garanziaOriginaleMesi		
	tipoGaranzia	_tipoGaranzia.nome	chiave esterna
	dataInserimento		
	dataCessazioneEsistenza		
	dataAcquisto		
	statoHw	_tipoStatoHW.nome	chiave esterna
	idFornitore	Fornitore.ragioneSociale	chiave esterna ereditata attraverso l'associazione VENDE_HW
	marca	Fornitore.ragioneSociale	chiave esterna ereditata attraverso l'associazione PRODUCE_HW
Risorse_SW	**nomeSW**		
	dataInserimento		
	url		
	acquistabile		
	lingua	_tipoLingua.nome	chiave esterna
	venditore	Fornitore.ragioneSociale	chiave esterna ereditata attraverso l'associazione VENDE_SW
gruppiSW	**nome**		
	descrizione		
	dataCreazione		
	gruppoStatico		
	macchinaVirtuale		
Fornitore	**ragioneSociale**		traduce *denominazione*
	eProduttoreSW		traducono *tipoFornitura*, richiesta dal committente
	eProduttoreHW		

	faAssistenza		
	eDistributoreHW		
	eDistributoreSW		
PoolLicenze	**objectGUID**		
	numeroLicenze		
	costoUnitarioLicenza		
	tipoAcquisizione	_tipoAcquisizione.nome	chiave esterna
	tipoLicenza	_tipoLicenza.nome	chiave esterna
	dataAcquisizione		
	dataScadenza		traduce *scadenza*
	dataInserimento		
	productKeyMultiplo		
	swAbilitato	RisorseSW.nomeSW	chiave esterna ereditata attraverso l'associazione ABILITA
ContrattiManutenzione	**idContratto**		
	dataStipula		
	dataScadenza		
	dataDisdetta		
	fornitore	Fornitore.ragioneSociale	chiave esterna ereditata attraverso l'associazione STIPULA
InstallazioniGruppiSW	**idInstallazione**		chiave parziale dell'installzione
	modelloCommercialeHW	RisorsaHW.modelloCommerciale	chiave primaria ereditata da RisorsaHW attraverso l'associazione SUBISCE
	typeModelHW	RisorsaHW.typeModel	chiave primaria ereditata da RisorsaHW attraverso l'associazione SUBISCE
	serialNumberHW	RisorsaHW.serialNumber	chiave primaria ereditata da RisorsaHW attraverso l'associazione SUBISCE
	nomeGSW	gruppiSW.nome	chiave primaria ereditata da gruppiSW attraverso l'associazione USA
	dataInstallazione		
SostituzioniHW	**modComHWSostituito**	RisorsaHW.modelloCommerciale	chiave primaria ereditata da RisorsaHW attraverso l'associazione HW_SUBISCE_S
	tpModHWSostituito	RisorsaHW.typeModel	chiave primaria ereditata da RisorsaHW attraverso l'associazione HW_SUBISCE_S
	SlNumHWSostituito	RisorsaHW.serialNumber	chiave primaria ereditata da RisorsaHW attraverso l'associazione HW_SUBISCE_S
	modComHWSostituente	RisorsaHW.modelloCommerciale	chiave primaria ereditata da RisorsaHW attraverso l'associazione IMPONE

	tpModHWSostituente	RisorsaHW.typeModel	chiave primaria ereditata da RisorsaHW attraverso l'associazione IMPONE
	SNumHWSostituente	RisorsaHW.serialNumber	chiave primaria ereditata da RisorsaHW attraverso l'associazione IMPONE
	fornitore	Fornitore.ragioneSociale	chiave primaria ereditata da Fornitore attraverso l'associazione ESEGUE
	dataSostituzione		
	motivo		
	descrizione		
uPartecipaG$_{Rel}$ =PARTECIPA$_{ER}$	**idUtente**	Utenti.userID	chiave primaria ereditata da Utenti
	idGruppo		chiave primaria ereditata da Gruppi
	ruolo	_tipoPartecipazione.nome	attributo dell'associazione
	dataAssegnazione		attributo dell'associazione
Assegnazione$_{Rel}$ = RICEVE_IN_ASSEGNAZIONE$_{ER}$	**idUtente**	Utenti.userID	chiave primaria ereditata da Utenti
	modelloCommercialeHW	RisorsaHW.modelloCommerciale	chiave primaria ereditata da RisorsaHW
	typeModelHW	RisorsaHW.typeModel	chiave primaria ereditata da RisorsaHW
	serialNumberHW	RisorsaHW.serialNumber	chiave primaria ereditata da RisorsaHW
	dataPreassegnazione		
	dataPreassegnazioneA		
	dataAssegnazione		
	dataScadenza		
	dataPreDismissione		
	dataDismissione		
	tipoAssegnazione	_tipoAssegnazione.nome	
	scadenzaImportante		
hwSotto-Manutenzione$_{Rel}$ = COPRE$_{ER}$	**modelloCommercialeHW**	RisorsaHW.modelloCommerciale	chiave primaria ereditata da RisorsaHW
	typeModelHW	RisorsaHW.typeModel	chiave primaria ereditata da RisorsaHW
	serialNumberHW	RisorsaHW.serialNumber	chiave primaria ereditata da RisorsaHW
	idContratto		
	dataInizioCopertura		attributo dell'associazione
	costoManutenzioneMese		attributo dell'associazione
	dataFineCopertura		attributo dell'associazione
	tipoManutenzione	_tipoManutenzione.nome	attributo dell'associazione
swPartecipa-Gruppo$_{Rel}$ = FORMA$_{ER}$	**idSW**	RisorseSW.nomeSW	chiave primaria esterna ereditata da RisorseSW attraverso l'associazione FORMA

	idGruppo	gruppiSW.nome	chiave primaria esterna ereditata da gruppiSW attraverso l'associazione FORMA
DownGradeSW$_{Rel}$ = DOWNGRADE$_{ER}$	**swPadre**	RisorseSW.nomeSW	chiave primaria esterna ereditata da RisorseSW attraverso l'associazione DOWNGRADE, ramo *dona*
	swFiglio	RisorseSW.nomeSW	chiave primaria esterna ereditata da RisorseSW attraverso l'associazione DOWNGRADE, ramo *riceve*
	transitivo		attributo dell'associazione
cardinalitaSW-Installati$_{Rel}$ = RICHIEDE_ABILITAZIONE_SW$_{ER}$	**idInstallazione**		chiave parziale
	modelloCommercialeHW	RisorsaHW.modelloCommerciale	chiave primaria ereditata da InstallazioniGruppiSW attraverso l'associazione RICHIEDE_ABILITAZIONE_SW
	typeModelHW	RisorsaHW.typeModel	chiave primaria ereditata da InstallazioniGruppiSW attraverso l'associazione RICHIEDE_ABILITAZIONE_SW
	serialNumberHW	RisorsaHW.serialNumber	chiave primaria ereditata da InstallazioniGruppiSW attraverso l'associazione RICHIEDE_ABILITAZIONE_SW
	nomeGSW	gruppiSW.nome	chiave primaria ereditata da InstallazioniGruppiSW attraverso l'associazione RICHIEDE_ABILITAZIONE_SW
	idSW	RisorseSW.nomeSW	chiave primaria ereditata da RisorseSW attraverso l'associazione RICHIEDE_ABILITAZIONE_SW
	cardPrenotazione		attributo dell'associazione
licenzeDiGruppo-Installate$_{Rel}$ = ASSEGNA_LIC_PER_SW$_{ER}$	**idInstallazione**		chiave parziale
	modelloCommercialeHW	RisorsaHW.modelloCommerciale	chiave primaria ereditata da InstallazioniGruppiSW attraverso l'associazione ASSEGNA_LIC_PER_SW

	typeModelHW	RisorsaHW.typeModel	chiave primaria ereditata da InstallazioniGruppiSW attraverso l'associazione ASSEGNA_LIC_PER_SW
	serialNumberHW	RisorsaHW.serialNumber	chiave primaria ereditata da InstallazioniGruppiSW attraverso l'associazione ASSEGNA_LIC_PER_SW
	nomeGSW	gruppiSW.nome	chiave primaria ereditata da InstallazioniGruppiSW attraverso l'associazione ASSEGNA_LIC_PER_SW
	idLicenza	PoolLicenze.objectGUID	chiave primaria ereditata da PoolLicenze attraverso l'associazione ASSEGNA_LIC_PER_SW
	idSW	RisorseSW.nomeSW	chiave primaria ereditata da RisorseSW attraverso l'associazione ASSEGNA_LIC_PER_SW
	numeroLicenzeInstallate		attributo dell'associazione
_tipoUtente	**nomeTipo**		attributo semplice
	descrizione		attributo semplice
_tipoGruppo	**nome**		attributo semplice
	descrizione		attributo semplice
_destinazioneDUSO	**nome**		attributo semplice
	descrizione		attributo semplice
_tipoHW	**nome**		attributo semplice
	descrizione		attributo semplice
_tipoGaranzia	**nome**		attributo semplice
	descrizione		attributo semplice
_tipoStatoHW	**nome**		attributo semplice
	descrizione		attributo semplice
_tipoLingua	**nome**		attributo semplice
	descrizione		attributo semplice
_tipoLicenza	**nome**		attributo semplice
	descrizione		attributo semplice
_tipoAcquisizione	**nome**		attributo semplice
	descrizione		attributo semplice
_tipoPartecipazione	**nome**		attributo semplice
	descrizione		attributo semplice
_tipoManutenzione	**nome**		attributo semplice
	descrizione		attributo semplice
	eSLA		attributo semplice
_tipoAssegnazione	**nome**		attributo semplice
	descrizione		attributo semplice

Tabella 2.5: Modello Relazionale.

2.2.2 Schema modificato dal committente

Lo schema cosi ottenuto è stato sottoposto ai referenti aziendali. Dopo una discussione sulle varie scelte adottate si è deciso di modificare leggermente lo schema finale della base di dati.

Le ragioni principali delle modifiche sono state mantenere una struttura semplice dell'applicativo e cercare di mantenete il più possibile la compatibilità con la vecchia base di dati. Molte delle informazioni modellate nell'attuale progetto non trovano riscontro rendendo molto difficile poi la sintesi dell'applicativo di transizione dei dati. Una delle principali richieste è stata quella di allentare il vincolo di chiave sulla relazione RisorsaHW. Come evidenziato nell'analisi, in vari periodi temporali sono state usate diverse tecniche di identificazione delle risorse. Questo implica che il parco delle risorse registrate usa diversi parametri per l'identificazione a seconda del periodo. Lo stesso problema si riscontra per la relazione PoolLicenze. In generale si è deciso anche di sostituire le chiavi primarie multiple di entità a se stanti, ove possibile, con chiavi singole e quelle alfanumeriche con chiavi numeriche da usare come indice.

Soggetto di ulteriore modifica è stata la relazione "Assegnazione" trasformata in un'entità a se stante. Questo è preferibile nel caso si voglia mantenere la cronologia delle installazioni nel tempo. Il vincolo di univocità in funzione del tempo verrà implementato a livello applicativo.

E stato richiesto inoltre la presenza su ogni relazione rilevante di due campi di testo, *descrizione* e *note*, nel caso non fossero presenti per permettere di memorizzare informazioni accessorie non strutturate. Il campo *descrizione* viene ipotizzato per offrire le proprie informazioni a tutti gli utenti ce hanno accesso alle informazioni generali sulla risorsa mentre il campo note si vuole riservare per l'utilizzo dei soli "addetti ai lavori", amministratori o personale tecnico.

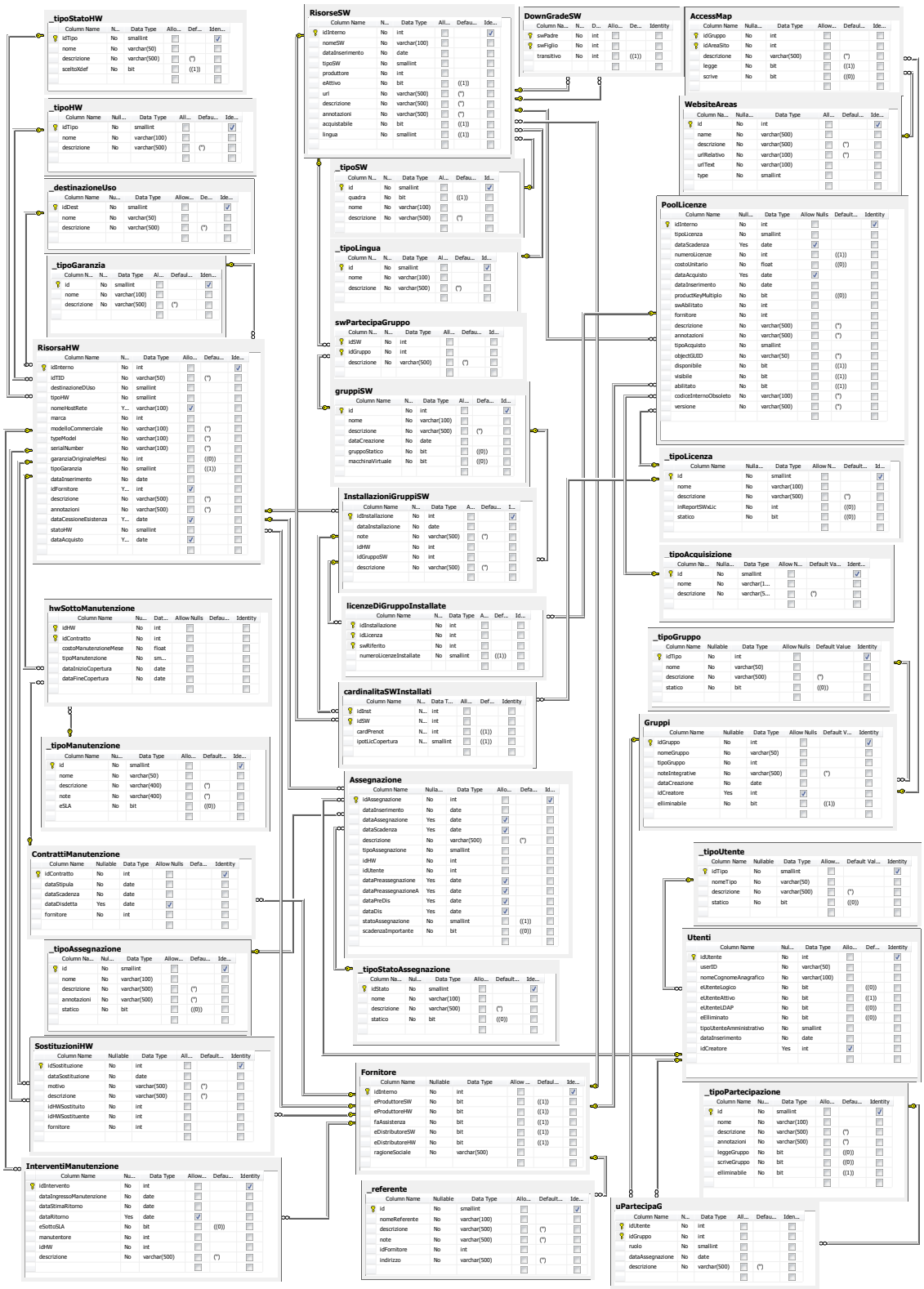
La versione dello schema della base di dati, effettivamente implementato, viene presentata nella figura 2.5.

2.3 Progettazione fisica - Vincoli del committente

La progettazione fisica della base di dati è stata immediata grazie all'analisi effettuata in principio. Ad ogni attributo rappresentante una denominazione, una descrizione oppure un'annotazione è stato associato il tipo

dato varchar. La lunghezza massima delle stringhe è stata definita con i committenti, generalmente alle denominazioni sono stati associati 100 caratteri mentre ai campi rappresentanti descrizioni oppure annotazioni 500. I flag sono stati mappati con il tipo boolean mentre le date con il tipo date. Le chiavi primarie virtuali inserite sono state mappate con il tipo dati int/smallint in funzione della previsione di valore massimo che si può ipotizzare. Per alcuni attributi sono stati individuati dei parametri più probabili di default.

Uno dei vincoli progettuali imposti dal committente è stato quello di implementare la base di dati su un sistema Microsoft SQL Server, versione 2000. Questo in quanto l'intranet aziendale possedeva già dei servizi di questo tipo attivi e l'aggiunta di una base di dati sarebbe stata molto agevole. Tale vincolo non ha condizionato minimamente il processo di sviluppo in quanto il DBMS in questione implementa tutti i vincoli d'integrità espressi dal modello relazionale. Al contrario è stata molto apprezzata l'interfaccia grafica che tale sistema mette a disposizione per la definizione dello schema. Tale interfaccia permette di rappresentare la base di dati su un'area piana sulla quale si possono definire nuove relazioni con l'aiuto del solo puntatore del mouse. A tali relazioni si possono aggiungere attributi definendo semplicemente il nome nel campo adibito. Una volta scelto il tipo di dato si possono definire i vincoli referenziali e di chiave primaria. I vincoli di chiave primaria vengono associati ad un simbolo rappresentante una chiave. I vincoli referenziali vengono rappresentati da dei connettori orientati che collegano le due relazioni implicate. Il connettore presenta un simbolo rappresentante una chiave dalla parte della chiave primaria referenziata ed il simbolo rappresentante l'infinito dalla parte della relazione contenente la chiave esterna. Si può visionare una rappresentazione grafica della base di dati implementata fisicamente nella figura 2.5.



1-1

Figura 2.5: Schema Relazionale implementato

Capitolo 3

Progettazione Applicativo

3.1 Introduzione

L'analisi dei requisiti per la progettazione dell'applicativo è strettamente legata a quella per la progettazione della base di dati in quanto si devono legare le due strutture per modellare la realtà aziendale. L'applicativo deve essere quindi in grado di implementare le operazioni descritte nel progetto formativo ed implicitamente nell'analisi funzionale. Per *operazioni* ci si riferisce prevalentemente alle azioni base di inserimento, modifica, eliminazione ed interrogazione delle relazioni ottenute attraverso l'analisi sviluppata precedentemente. Si dovranno ulteriormente implementare tutti quei vincoli che sono stati tralasciati a livello di strutturazione della base di dati, come il rilassamento dei vincoli di chiave sulla relazione *RisorsaHW* o *Associazione*. Tali vincoli verranno verificati precedentemente ad ogni azione di modifica dello stato della base di dati. In pratica prima di eseguire qualunque operazione di inserimento o modifica si dovrà verificare lo stato nella quale si trova la base di dati e lo stato raggiunto attraverso la modifica che non è altro che il processo che implementa il DBMS per il rispetto dei vincoli relazionali.

3.1.1 Scelta dell'architettura

Le architetture possibili per la realizzazione dell'applicativo sono molteplici. Un approccio possibile, obsoleto ed utilizzato prevalentemente nel periodo precedente al consolidamento di Internet e delle tecnologie associate, è quello di codificare un eseguibile che s'interfacci con il server DBMS e fornisca all'utente tutte le funzioni richieste. L'utilizzo multiutente su diverse workstation si può ottenere fornendo ad ogni utilizzatore una copia da installare sulla propria workstation opportunamente configurata. Per quanto funzionante, una soluzione simile è altamente inefficiente. Si consideri come

giustificazione di tale affermazione l'esempio di update delle funzioni del software oppure di correzione di bug, evenienze molto frequenti nel mondo del software. In un'architettura simile si dovrebbe ripetere l'installazione su tutte le macchine interessate dall'applicativo. Secondariamente lo sviluppo ed il mantenimento dell'applicativo diventa un lavoro rilevante.

L'alternativa più adatta per quelle che sono le necessità di una applicazione come quella in discussione è l'architettura basata sulle applicazioni web (web-application). In un'applicazione web il codice viene eseguito presso un server (che normalmente coincide con quello web attraverso moduli o plugin aggiunti), mentre l'I/O viene trasmesso all'utente attraverso la rete.

Dal punto di osservazione dell'utente, l'unico strumento necessario per usare una tale applicazione è un browser web attraverso il quale collegarsi al server ed interrogare lo stesso. La modifica del codice di una web-application avviene senza alcuna modifica da parte dell'utilizzatore e sovente senza che quest'ultimo abbia percezione di tali modifiche, a meno che non siano evidenti (esempio: modifica all'interfaccia utente).

3.1.2 Tecnologie utilizzate

Considerando i vincoli del committente sulle piattaforme da utilizzare, in particolare "SQL Server 2000" per quanto riguarda il DBMS e "Internet Information Services" versione 5.1 o superiore per quanto riguarda il server web, si è trovata come soluzione ottimale l'abbinamento a tali tecnologie di "ASP.NET 3.5", per la codifica dell'applicativo.

Come ambiente di sviluppo è stato usato "Microsoft Visual Studio Express Edition", uno strumento fortemente indirizzato verso la realizzazione di applicazioni web. L'ambiente è composto dai seguenti tool:

- *Visual Basic 2008 Express Edition*
- *Visual C# 2008 Express Edition*
- *Visual C++ 2008 Express Edition*
- *Visual Web Developer 2008 Express Edition*

Per lo sviluppo dell'applicativo è stato impiegato "Visual Web Developer 2008 Express Edition" come IDE e Visual C# come linguaggio di programmazione. Come testing DBMS sono stati usati "SQL Server 2008 Express Edition" ed "SQL Server 2000".

3.1.3 ASP.NET

ASP.NET è sostanzialmente un insieme di classi, basate sul framework ".NET", che possono essere istanziate usando un qualsiasi linguaggio compatibile con Common Language Runtime. Quindi teoricamente una web-application può essere sviluppata con un linguaggio diverso per ogni sezione. Uno schema sintetico del framework .NET si può visualizzare nella figura 3.1.

Il funzionamento delle web-application si basa sulla tecnologia degli HTML Forms, introdotta con HTML 2.0. HTML Forms è un modo per listati HTML di inviare informazioni dall'utente verso il server attraverso campi testo o pulsanti.

Una web-application sviluppata in ASP.NET generalmente possiede uno o più template (denominati *pagine master*, salvate con estensione ".master"). Un template non è altro che una veste grafica, in HTML, che descrive la parte comune delle pagine web. Il layout usato per l'applicativo sviluppato si può osservare nella figura 3.2.

Al layout in HTML vengono aggiunti dei blocchi di codice (controlli *ContentPlaceHolder*) che si comportano come dei contenitori. Una web-application sviluppata in ASP.NET si articola in uno o più file ".aspx" contenenti il codice (HTML e ASP.NET) che andrà a posizionarsi nei vari contenitori del template specificati nel relativo listato. Un listato può specificare codice per tutti, parte, o nessun contenitore del template.

Ad ogni file ".aspx" viene associato un file di codice puro con estensione ".cs" (nel caso il codice venga sviluppata in c#) che ha lo stesso nome del file ".aspx". L'applicazione si costruisce semplicemente costruendo dentro il file ".aspx" la veste grafica alla quale si aggiungono dei controlli creati ad hoc, ognuno con le proprie funzionalità. Ad un blocco di controllo si possono associare delle azioni (metodi scritti in c#) in funzione di alcuni eventi modellati diversi da controllo a controllo. Quando si verifica l'evento il controllo del flusso viene passato al blocco di codice associato. Il codice viene inserito nel file ".cs" e viene organizzato in metodi.

Ogni web-application ha un file di configurazione, "web.config", sviluppato in XML che definisce le impostazioni di sicurezza, di stato dell'applicazione oppure delle variabili globali (anche se per tale finalità esiste una struttura più adatta) ecc. Per reazioni a eventi e variabili globali dell'applicazione viene usato il file "Global.asax".

Per inserire un controllo in una pagina HTML abbiamo due possibilità utilizzando "Visual Web Developer'. La più immediata è quella di utilizzare la modalità "Design". I controlli si inseriscono semplicemente trascinandoli dentro una rappresentazione del layout. L'ambiente si occuperà di inserire il codice relativo nonché una rappresentazione grafica nel layout. Per definire

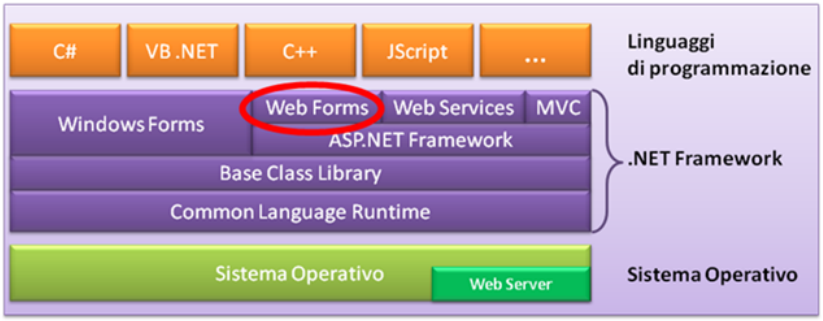


Figura 3.1: Framework .NET.

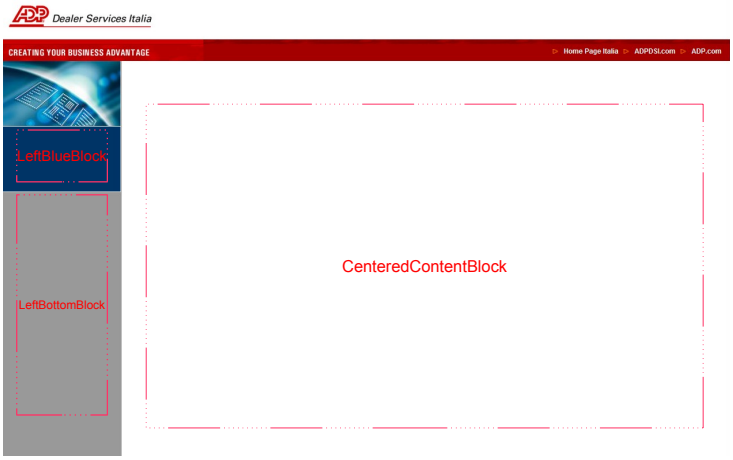


Figura 3.2: Template applicativo.

i parametri del controllo si usa un menu interattivo raggiungibile attraverso il click destro del mouse. Alternativa alla modalità ”Design” è la modalità ”Source” nella quale il codice viene inserito testualmente. Nello sviluppo dell'applicativo è stata utilizzata quest'ultima modalità in quanto rende molto più semplice definire la posizione del controllo nel flusso dell'applicativo essendo anche molto più snella e veloce.

Per inserire un controllo testualmente in un listato ”.aspx” lo si racchiude in un markup HTML preceduto da dai caratteri ”asp:”. La forma finale sarà ”<asp:[nomeControllo] [attributi] runat=”server” >”. L'attributo ”runat=server” è obbligatorio se si vuole che il comando venga eseguito a livello server. La mancanza di tale attributo rende il markup uguale ad un normale comando HTML.

3.2 Strumenti e metodi d'implementazione

3.2.1 Controlli ASP

Le applicazioni realizzate in ASP.NET utilizzano dei costrutti a livello server che possono facilitare molto il processo di codifica. ASP.NET fornisce alcuni controlli che sono predefiniti, ma che possono anche essere creati dall'utente in funzione delle proprie necessità. I controlli si sostituiscono all'utente nella realizzazione di molteplici operazioni basilari e ripetitive dando la possibilità di concentrarsi principalmente sull'organizzazione logica dell'applicazione.

Normalmente, o meglio detto nelle versioni precedenti di ASP, nel realizzare un applicativo, si doveva gestire nei minimi dettagli sia la parte grafica che la parte logica sottostante. Il controllo evita di dover per forza gestire la parte grafica, logica o entrambe mettendo a disposizione delle traduzioni standardizzate delle varie operazioni più frequenti. Un esempio è la realizzazione di una tabella che visualizzi il risultato di una interrogazione sulla base di dati. Senza i controlli si sarebbe dovuta realizzare prima una veste grafica minima e generare un ciclo che per ogni tupla del risultato generi l'adeguata codifica HTML per inserire i dati nella tabella. ASP.NET fornisce il controllo "GridView" al quale si fornisce una fonte dei dati ed alcuni parametri per definirne il comportamento ed il controllo realizza automaticamente la veste grafica senza coinvolgere minimamente l'utente. Ovviamente la veste grafica può essere più o meno complessa in funzione della configurazione del controllo. Questa strutturazione è molto comoda in quanto la velocità con la quale si può ipoteticamente generare codice funzionante è molto elevata, sicuramente molto più elevata rispetto ai paradigmi, che potremo chiamare "imperativi", che non presentano questa strutturazione. Ovviamente questo approccio presenta delle problematiche, non indifferenti, nel caso si necessiti di funzioni non standard in quanto definire i propri controlli non è un'operazione propriamente agevole.

Di seguito verranno elencati i principali controlli utilizzati per lo sviluppo dell'applicativo:

ContentPlaceHolder – Controllo che permette di definire un'area riservata in un template dove andrà ad inserirsi il codice definito nel controllo *Content*.

Content – Controllo che contiene il listato che viene inserito nel template indicato nell'attributo *ContentPlaceHolderID*.

SqlDataSource – Controllo che definisce all'interno della struttura nella quale viene dichiarato (solitamente all'interno di un controllo *Content*) una connessione ad un database. La stringa di connessione, che indirizza il

controllo verso il database d'interesse, viene definita nel attributo *ConnectionString*. Il controllo può essere utilizzato sia per interrogare la base di dati che per modificarla. Per ogni operazione d'interrogazione, modifica, o eliminazione vengono definite le relative istruzioni SQL rispettivamente negli attributi *SelectCommand*, *UpdateCommand*,*DeleteCommand*. Le operazioni possono essere anche parametrizzate attraverso l'utilizzo di opportuni parametri collegati ad altri controlli nell'applicazione. Gli attributi vengono definiti nella relativa stringa SQL attraverso il costrutto "@[nome attributo]. Secondariamente, all'interno del blocco di codice del controllo viene definita una sezione, per ogni tipologia di operazione, nella quale vengono specificati i legami fra la stringa "nome attributo" ed un relativo controllo della pagina.

LinkButton – Semplice controllo che definisce un link. La differenza con un normale link HTML è che viene riferito livello server e gli si possono associare azioni in funzione degli eventi che lo investono.

Image – Controllo che implementa il funzionamento del tag <img>. Riferibile a livello server (come *LinkButton*).

Calendar – Controllo che implementa un oggetto calendario. Utile nella GUI.

GridView – Controllo che permette di rappresentare una relazione di un data in una pagina web. Permette sia l'interrogazione sulla base di dati che operazioni come modifica e inserimento.

Button – Semplice controllo che implementa un pulsante.

ListBox – Controllo che implementa il costrutto HTML <select></select>, cioè un elenco di opzioni che possono essere selezionate. La selezione può essere singola oppure multipla. Può essere collegato ad un controllo di tipo *SQLDataSource*. Può solo implementare operazioni di interrogazioni della base di dati.

DropDownList – Controllo molto simile al controllo *ListBox* che però visualizza e seleziona una sola opzione. Può essere collegato ad un oggetto del tipo *SQLDataSource* potendo solamente visualizzare dati.

DetailsView – Controllo che visualizza una sola tupla di una selezione sulla base di dati.

Repeater – Controllo che costruisce un elenco con associazione a dati che consente di ottenere un layout personalizzato ripetendo un modello specificato per ogni elemento visualizzato.

MultiView – Controllo utilizzato come contenitore per un gruppo di controlli *View*. Un controllo *View* non è altro che una vista. Attraverso questo controllo possiamo creare applicazioni multi-view dentro una pagina web.

Tutti i controlli hanno un attributo *id* che identificano gli oggetti istanziata durante l'esecuzione della particolare pagina ASP.NET . L'id del oggetto è molto importante in quanto permette di riferirlo modificandone anche le

proprietà durante la normale esecuzione. Ogni controllo che prevede la costruzione di codice HTML in fase di istanziazione, possiede anche attributi tipici dei costrutti HTML come la definizione dei fogli di stile del controllo (*CssClass*), definizione dei colori di background ecc.

3.2.2 Struttura dell'applicativo

Per la realizzazione dell'applicativo si è pensato di dividere il problema in tanti segmenti specializzati ognuno in certe operazioni. Si sono così evidenziate le seguenti aree operazionali:

1. *Gestione associati*
2. *Gestione hardware*
3. *Gestione gruppi*
4. *Gestione gruppi software*
5. *Gestione risorse software*
6. *Gestione pool licenze*
7. *Gestione assegnazioni*
8. *Gestione installazione software*
9. *Gestione licenziazione installazioni*
10. *Gestione assegnazione IIS*
11. *Gestione assegnazione amministrativa*
12. *Gestione reportistica*
13. *Gestione applicativo*

Per ogni area sono stati sviluppati dei moduli per la maggior parte composti da una pagina ".aspx" e la relativa pagina di codice c#. Ogni modulo si comporta come una mini applicazione a se stante, isolata ed auto sufficiente che posiziona il proprio codice nei relativi campi del layout descritto nella figura 3.2 e utilizza le informazioni comuni accedendo al file web.config nel quale è stato deciso di inserire i pochi dati comuni. Questi dati consistono prevalentemente nelle stringhe di connessione al database e nelle espressioni regolari utilizzate nella formattazione dei campi d'inserimento. All'interno di

ogni mini applicazione vengono usati gli ambienti multi-view per gestire le varie sezioni nelle quali alcuni moduli vengono suddivisi.

La gestione dell'accesso alle varie aree dell'applicazione viene gestita dinamicamente. Nel database è stato aggiunta una relazione

WebsiteAreas (id , name, desrizione, urlRelativo, urlText, type).

Per ogni area del sito viene aggiunta una tupla a questa relazione che verrà secondariamente utilizzata per la costruzione del link all'area del sito. *WebsiteAreas* è collegata ai gruppi di utenti attraverso la relazione

AccessMap (idGruppo, idAreaSito)

che mette in relazione ogni gruppo di utenti alle aree del applicazione alle quali gli utenti facenti parte dei relativi gruppo hanno accesso.

Ogni modulo dell'applicativo esegue un controllo sull'utente che vi accede verificandone l'effettiva abilitazione. Se l'utente non è abilitato gli si segnala l'errore e lo si indirizza verso la pagina principale dell'applicazione.

La sezione *Gestione applicativo* è formata da diversi moduli. In questa sezione si definiscono la relazione fra i gruppi e le aree del sito alle quali quest'ultimi hanno accesso, le operazioni di sincronizzazione dell'insieme degli utenti con il sistema informativo aziendale nonché alcune verifiche sulla coerenza dei dati nella base di dati.

3.2.3 Sessioni

HTTP è un protocollo senza stato, cioè non mantiene informazioni temporali sulle pagine che visitiamo. Se vogliamo che una esecuzione di una pagina al tempo t_b ricordi informazioni della sua stessa esecuzione al tempo t_a dobbiamo usare un metodo per memorizzare tale informazioni. Una scelta possibile sono le variabili di sessione le quali sono coppie di chiavi memorizzate temporaneamente nella memoria del server. Una delle due chiavi memorizza il nome della variabile mentre l'altra il valore.

Le sessioni non sono l'unico modo di mantenere memoria sullo stato in una applicazione web e nemmeno il più adatto per passaggio di dati fra istanze temporali diverse. Infatti per la memorizzazione viene usata direttamente la memoria centrale. Per grossi quantitativi di dati si rischia di mandare offline il servizio di hosting web o addirittura generare perdita di dati se il server dati è presente sulla stessa macchina fisica. Nel nostro caso data la quantità minima di informazioni non si corre il rischio di creare alcun rallentamento in quanto sono state usate per la memorizzazione di indici numerici, o di singole stringhe di testo limitate.

3.2.4 Espressioni regolari

In alcune operazioni di input dati da parte dell'utente si ha la necessità di controllare, o per meglio dire limitare, l'insieme di stringhe che si possono inserire. In ASP.NET questo viene ottenuto attraverso l'utilizzo del controlli *RegularExpressionValidator*. Il modulo viene inserito nella pagina HTML nel punto nel quale si vogliono visualizzare eventualmente i messaggi di errore definiti nell'attributo *ErrorMessage*. Per il funzionamento, il controllo ha bisogno di una stringa di definizione dell'insieme delle stringhe che possono essere accettate da un certo campo di input definito nell'attributo *ControlToValidate*. Tale stringa di definizione viene denominata *Regular Expression (Espressione Regolare)*.

Le espressioni regolari non sono altro che delle regole logiche che indicano quali, come ed in che numero alcuni caratteri possono comparire in una stringa. Per esempio se abbiamo un capo rappresentante una mail ci possiamo aspettare ragionevolmente che il valore di un tale campo sia formato da una stringa alfanumerica iniziale, il simbolo @ ed un'ulteriore stringa alfanumerica indicante il nome del server presso il quale il servizio viene domiciliato.

Nella teoria delle espressioni regolari i caratteri vengono divisi in due tipologie: caratteri propri, denominati *literals*, e caratteri generici, *metacharacters*. A questa distinzione si aggiungono degli operatori che denotano insiemi e operazioni fra insiemi di queste due tipologie di caratteri.

3.2.5 Stampa / download di informazioni su file

Uno dei requisiti contenuti nel progetto formativo era quello di dare la possibilità di stampare o scaricare informazioni dall'applicativo.

Per la stampa la soluzione scelta è stata quella di mettere in risalto a video solamente il contenitore centrale del layout abbinato al logo superiore dell'azienda. Si ottiene cosi una pagina web contenente solamente le informazioni d'interesse che può essere stampata agevolmente usando la funzione messa a disposizione dal browser.

Per il download delle informazioni si è scelto di usare formati semplici di strutturazione dei file. Per dati organizzati in tabelle si è scelto il formato ".csv" (Comma Separated Values) che permette di salvare dati sotto forma testuale. Le colonne si separeranno utilizzando il simbolo ";" mentre le righe della tabella vengono inserite in linee separate del file di testo. I valori di testo vengono inseriti fra doppi apici in modo da avere già una formattazione corretta nei software che verranno utilizzati per gestire questi dati (prevalentemente Microsoft Excel 2000).

Per ogni oggetto, solitamente di tipo *GridView*, è stata creata una funzione , collegata ad un oggetto di tipo *Button* o *Link*, attraverso la quale si sono compattate tulle le informazioni presenti in tale oggetto. La stringa cosi ottenuta viene poi inviata al browser del utente sotto forma di file di testo, con la relativa estensione, ".csv" oppure ".doc".

Nel processo di assegnazione di una risorsa hardware si è voluto implementale anche la stampa automatica del modulo di assegnazione, che si può visualizzare in figura 3.3, precompilato con i dati dell'utente e della risorsa hardware da assegnare. Per generare questo file si è utilizzato il formato ".doc". L'applicativo dispone di una copia digitale di tale modulo nella quale sono stati inserite delle apposite stringhe che al momento della generazione del file vengono ricercate e sostituite con i dati relativi all'assegnazione operata. Il modulo cosi ottenuto può essere stampato e controfirmato.

3.3 Integrazione con la struttura informatica aziendale

Il maggiore lavoro d'integrazione sviluppato per questo l'applicativo è stata la procedura di login. Su richiesta del committente, si sono volute utilizzare le stesse credenziali del sistema aziendale. Per implementare il processo si richiedono all'utente username e password attraverso un collegamento sicuro (HTTPS). Le due stringhe vengono usate per avviare un collegamento con i servizi *Active Directory* aziendali. Se il collegamento viene accettato e l'identità dell'utente verificata si procede alla verifica sull'abilitazione all'accesso delle aree del sito. Viene cosi fatta una selezione delle pagine alle quali l'utente ha accesso e creato il menu laterale con i link di accesso.

Dealer Services Italia

ASSEGNAZIONE TEMPORANEA BENI AZIENDALI

RICHIESTA

Il sottoscritto _____ richiede la dotazione temporanea dei seguenti beni aziendali:
_____
_____
_____
_____
_____
_____

Data: _____ Firma: _____

AUTORIZZAZIONE

Superiore diretto: _____

Direttore di struttura: _____

Direzione Risorse Umane: _____

ACCETTAZIONE

Il sottoscritto _____ riceve in dotazione temporanea i seguenti beni aziendali:
_____
_____
_____
_____
_____
_____

e dichiara di essere a conoscenza che l'azienda si riserva, in qualunque momento, la facoltà di chiedere la restituzione del materiale citato a fronte della cessazione del rapporto di lavoro, di passaggio ad altre mansioni, di cambiamenti di carattere organizzativo.

Data: _____ Firma: _____

RESTITUZIONE

Data:_____ Firma:_____

Per ricevuta
(ADP Dealer Services Italia - Direzione Risorse Umane): _____

Versione 12/01

Figura 3.3: Modulo assegnazione.

Capitolo 4

Conclusioni e considerazioni

Il presente lavoro di tirocinio è stato molto istruttivo per diverse ragioni dal punto di vista professionale. Attraverso il processo di sviluppo supportato si è compreso a fondo l'importanza delle varie fasi in cui viene divisa teoricamente la progettazione di un applicativo. In particolar modo si è evidenziata l'importanza che riveste l'analisi dei requisiti del committente. Un'analisi non completa o imprecisa comporta quasi sempre la modifica di buona parte del progetto con relativa perdita di tempo annessa. In aggiunta non si ha la possibilità di sintetizzare una progettazione di qualità in quanto ci sono sempre zone d'ombra nelle quali, rinviando il problema di analisi, si rischia sempre di inciampare in problemi di diversa natura, dalla coerenza logica del modello alla complessità risultante dell'applicativo.

Si è altrettanto sentita la necessità durante lo sviluppo della sintesi di un insieme di strumenti di lavoro, che dovrebbe essere elaborata in contemporanea al lavoro di analisi. Questo perché si è notato che non sempre le tecnologie utilizzate offrono tutti gli strumenti che possono servire. Avere un'analisi che dia la possibilità di modellarsi ed adattarsi è sicuramente da preferirsi come d'altronde studiare preliminarmente tutti gli strumenti necessari ad implementare il modello sintetizzato. Esempio sono i controlli offerti da ASP.NET che possono rivelarsi utilissimi fintanto che le operazioni da svolgere rientrano in certi canoni. Un'operazione non prevista da tali controlli può risultare molto più difficile da implementare che in un ambiente meno strutturato.

La difficoltà di sviluppare continuando un lavoro preesistente è stato un altro fattore di complessità. Questo sia perché il progetto preesistente può avere lacune di documentazione, come capitato in questa situazione, sia perché generalmente se un progetto presenta dei limiti in partenza, difficilmente si riesce a correggerli con un lavoro che non sia pari o addirittura superiore ad una reingegnerizzazione completa. Nel caso in esame ci si è scontrati con entrambe le situazioni.

Un'ulteriore difficoltà incontrata è stata quella di ricavare le informazioni per l'analisi dei requisiti. Infatti il lavoro è stato sviluppato all'interno della normale vita aziendale. Il lavoro di analisi, per quando sintetico possa essere, richiede comunque dei tempi che non sono nulli e non tutti gli attori del mini-mondo sono disposti a fornire. É stato infatti difficoltoso, a volte, avere le informazioni di cui si necessitava.

Alla fine del tirocinio, l'applicativo frutto del presente elaborato è stato completato e parzialmente testato. Non è stato eseguito un lavoro rigoroso di testing per mancanza di tempo (lavoro che il committente si è impegnato a sviluppare per conto proprio). L'applicativo è stato comunque consegnato funzionante e completo, avendo superato anche una verifica finale delle funzionalità con approvazione da parte del committente.

Bibliografia

[1] R.A. Elmasri, S.B. Navathe: *Sistemi di basi di dati - Fondamenti*, Addison Wesley, 4th edition, 2005

[2] R.A. Elmasri, S.B. Navathe: *Sistemi di basi di dati - Complementi*, Addison Wesley, 4th edition, 2005

[3] Matthew MacDonald: *Beginning ASP.NET 3.5 in C# 2008*, Apress,2nd edition, 2008

[4] Matthew MacDonald, Mario Szpuszta: *Pro ASP.NET 3.5 in C# 2008*, Apress,2nd edition, 2008

[5] Bill Evjen, Scott Hanselman, Devin Rader: *Professional ASP.NET 3.5 In C# and VB*, Wiley Publishing, 2008

Elenco delle figure

Elenco delle tabelle