Università degli studi di Padova

Dipartimento di Ingegneria dell'Informazione

Corso di Laurea Triennale in Ingegneria Informatica

Relazione di Tirocinio

Progetto SugarCRM Mobile

Laureando:
Alessandro Guaraldo

Relatore:
Prof. Michele Moro

A.A. 2011-2012

Indice

Elenco dei codici

Elenco delle figure

Elenco delle tabelle

Sommario

L'obiettivo di questo elaborato consiste nel documentare l'esperienza di tirocinio vissuta all'interno di una moderna azienda informatica, imparando a convivere e a relazionarmi con i colleghi e a lavorare su più progetti nello stesso tempo.

Il tirocinio è stato diviso in due principali momenti: all'inizio si è incentrato sullo studio dei linguaggi multi piattaforma e di un tool di sviluppo chiamato LiveCode e in un secondo tempo ho avuto modo di collaborare allo sviluppo di un progetto multi piattaforma.

Le difficoltà di operare su diversi ambienti e di sviluppare per più sistemi operativi hanno dimostrato come la crescita del web abbia notevolmente influenzato la programmazione portando alla creazione di web-app performanti come applicazioni che risiedono sulla stessa macchina dove operano; tutto ciò grazie alle reti di comunicazione di oggi che, un po' alla volta, spostano i carichi di lavoro computazionale e soprattutto i dati su server esterni in modo da archiviare i contenuti e renderli disponibili da qualsiasi punto del pianeta.

Capitolo 1

Introduzione

1.1 Obiettivi del Tirocinio

Questa tesi documenta il tirocinio di sei mesi svolto presso l'azienda Network Solution for Business; l' obiettivo principale di questo lavoro è stato lo studio dello sviluppo software multi piattaforma e l'esperienza ha avuto un buon esito per entrambe le parti. Infatti l'azienda ha sviluppato un progetto interno importante per i fini commerciali e strategici e personalmente ho appreso una grande quantità di informazioni sulle metodologie di lavoro e sulle possibilità di sviluppo software relazionandomi e organizzando le varie attività con altre persone.

La prima parte del periodo di tirocinio è stata incentrata sullo studio teorico delle tecnologie di sviluppo software e dei linguaggi di programmazione affiancando un lavoro di debug e di test su progetti in corso d'opera che utilizzavano le tecnologie studiate.

Nella seconda parte del tirocinio ho potuto utilizzare le conoscenze ottenute nella prima fase partecipando a un progetto il cui scopo era la creazione di un programma che si basava sulla possibilità di essere eseguito sulle principali piattaforme utilizzate: ho così avuto modo di conoscere tutte le alternative di progettazione multi piattaforma e di valutarne pro e contro.

1.2 Presentazione dell'azienda

La Network Solution for Business è un'azienda che lavora su un ampio spettro di mercati: dal campo medico all'odontoiatria, dalla telefonia voip all'installazione di reti cablate in fibra ottica e collegamenti wireless, fino ad arrivare al settore delle macchine utensili industriali, incentrando la varietà dei progetti sul settore dell'informatica. L'azienda si propone di dare assistenza ad

Figura 1.1: Logo aziendale NSB

aziende e professionisti per cui vengono sviluppati e installati macro progetti. Un particole che colpisce è l'età media di coloro che lavorano per l'azienda: sono presenti molti giovani con contratti part-time o a tempo pieno; inoltre si punta molto sull'innovazione e sullo spirito di squadra e spesso è presente qualche collaborazione con un polo Universitario dando la possibilità di svolgere tirocini o attività di sviluppo.

1.2.1 Prodotti

I prodotti offerti comprendono quattro categorie principali: medicale, automazione, sicurezza e hosting. Nell'ambito medicale si trovano:

- applicativi per studi odontoiatrici per diverse piattaforme;
- applicativi per banche di raccolta organizzata di materiale biologico e i dati a loro associati

Questi software sono risorse per tutto ciò che riguarda la diagnosi, la ricerca e la sperimentazione. Nell'ambito dell'automazione industriale i prodotto comprendono sia il software come applicativi per il calcolo dei dati da applicare a curvatrici automatiche, sia l'hardware con il calibro laser rotativo. La sicurezza e l'hosting offerti comprendono impianti di rete cablate e/o wireless, certificazioni e protezioni con firewall e hosting direttamente sul server aziendale

1.2.2 Servizi

Le attività che la N.S.B. svolge attualmente sono orientate alla creazione di soluzioni avanzate per il settore industriale, medicale e di assistenza sistemistica su qualsiasi piattaforma e sono in grado di coprire ogni aspetto di una

Figura 1.2: Logo LiveCode

soluzione: dall'infrastruttura di networking, alla configurazione dei server, alla realizzazione degli applicativi software.

1.3 Studio dello sviluppo multi piattaforma

La prima parte del tirocinio, come già detto, è stata dedicata allo studio della programmazione multipiattaforma e dei vari metodi di sviluppo attraverso i diversi linguaggi e in particolare ho avuto modo di studiare un tool di sviluppo, LiveCode, utilizzabile per la creazione di applicazioni per piattaforme Desktop e Mobile confrontandolo con altri linguaggi come Java e C#. Sono state poi approfondite le conoscenze di Java e della possibilità di eseguere applicativi scritti in questo linguaggio come servizi Windows o daemon Unix avendo il controllo sulla Java Virtual Machine attraverso l'uso del Java Service Wrapper creato da Tanuki Software.

1.3.1 LiveCode

Livecode usa un linguaggio interpretato molto simile alla lingua parlata e non di facile comprensione per utenti abituati a linguaggi di basso livello; per le piattaforme Desktop non è necessario aver nessun kit per la compilazione mentre per quelle Mobile ha l'esigenza di appoggiarsi alle SDK relative: questo comporta l'esclusione di sviluppi per iOS da Windows in quanto non è possibile installare l'SDK di iOS (Xcode) su Windows.

Le prestazioni di questo tool non sono comparabili a nessun linguaggio di programmazione valido e per questo sono stati effettuati alcuni test di comparazione con Java e C# con i seguenti algoritmi:

- Ricorsione;
- Pattern Matching.

Sono stati fatti tentativi di implementare anche algoritmi di inserimento e di ricerca in un albero binario ma LiveCode risultava troppo per un confronto.

Java [ms]	**C#** [ms]	**LiveCode** [ms]
21,9	1000	0
4,38	0	0
13,14	0	0
30,66	0	0
70,09	0	0
183,98	0	0
468,7	0	1000
1397,36	0	3000
3556,91	0	7000
8997,41	1000	16000
23317,03	1000	44000
3307,22	3000	118000
8581,27	6000	277000
23133,06	17000	743000
59705,29	44000	2042000
154581,06	115000	5349000
405553,51	300000	13376000
1068882,29	786000	33861000
2785818,79	2061000	88803000
7277439,02	5381000	249504000
18963681,84	14078000	666281000
49603811,85	37308000	1598868000
80350999,61	59607000	2552127000

Tabella 1.1: Tempi di esecuzione per l'algoritmo di Ricorsione

Nelle figure 1.3 e 1.4 sono riportati i grafici relativi ai tempi di esecuzione degli algoritmi inseriti nelle tabelle 1.1 e 1.2.

Come evidenziato dai dati rilevati si nota subito come le prestazione di LiveCode siano molto scadenti rispetto Java o C#: il tool infatti è stato creato con lo scopo di avere come maggior pregio la programmazione multi piattaforma. Anche Java è molto versatile e può essere utilizzato per tutte le piattaforme che supportano la JVM ma LiveCode, non usando alcuna macchina virtuale, traduce i costrutti basilari della programmazione durante la compilazione nelle relative applicazioni sfruttando le SDK delle piattaforme scelte.

LiveCode predilige gli ambienti Mobile per le poche prestazioni del tool ma soprattutto per dar modo all'utente di creare applicazioni in un unico sviluppo per Android e per iOS e di poterle pubblicare nel modo più veloce

Java [ms]	**C#** [ms]	**LiveCode** [ms]
2104,36	1000	35001000
31731,8	500	68282000
43401,3	500	98764000
54991,9	1000	131385000
28061	1000	160923000
33632,9	500	197113000
39082,2	1000	230387000
43589,7	1000	262134000
48837,4	1000	294097000
55210,9	1500	329734000

Tabella 1.2: Tempi di esecuzione per l'algoritmo di Pattern Matching

possibile: per questo motivo sono supportati molti controlli grafici e le gesture per i dispositivi multitouch.

1.4 Progetto SugarCRM Mobile

La seconda parte del tirocinio ha avuto come scopo la realizzazione di una applicazione utilizzando le conoscenze acquisite nei mesi precedenti. Il progetto assegnato a me e a un mio responsabile consisteva nella realizzazione di un client multipiattaforma alternativo a quello disponibile per l'accesso al CRM Aziendale, SugarCRM. Il client utilizzato fin'ora dagli utenti consentiva l'accesso ai dati via web ma la lentezza e il notevole carico di informazioni poco utili trasmesse non consentiva di svolgere il lavoro in modo veloce. Spesso le necessità erano di inserire dati al volo ma per varie cause, tra le quali la notevole lentezza, gli utenti trovavano più comodo inserire le informazioni in un secondo momento dimenticandosele in parte o talmente nel tempo e causandone quindi la perdita. Per questo motivo il responsabile delle vendite commerciali premeva per la realizzazione di un nuovo client più veloce e che richiedesse agli utenti solamente le informazioni necessarie per l'inserimento di nuovi dati. Una seconda richiesta era la personalizzazione del client e di SugarCRM secondo le necessità degli utenti commerciali; queste modifiche consistevano nella disattivazione dei moduli non utilizzati, nell'inserimento di nuovi moduli dedicati alle attività che SugarCRM non prevede e nella modifica dei moduli già utilizzati. L'applicativo doveva interfacciarsi quindi con i servizi offerti da SugarCRM e questo è reso possibile da una comunicazione di tipo SOAP o REST con il web service. La progettazione è iniziata con una attenta analisi di tutti i requisiti e le necessità degli utenti tenendo conto

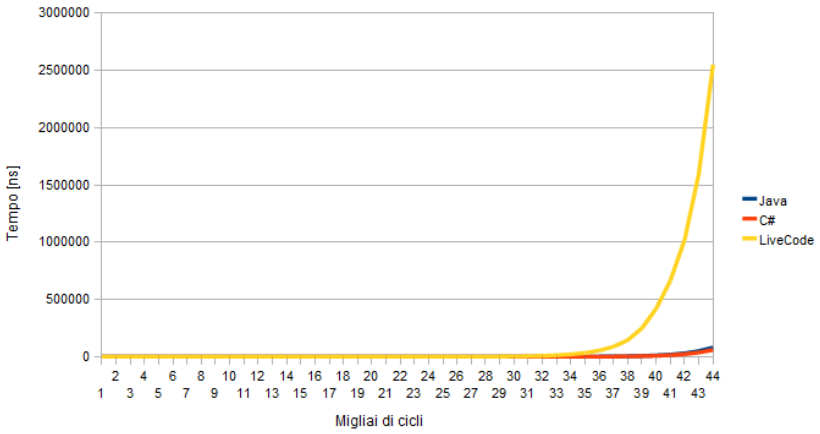

Figura 1.3: Grafico test Ricorsione

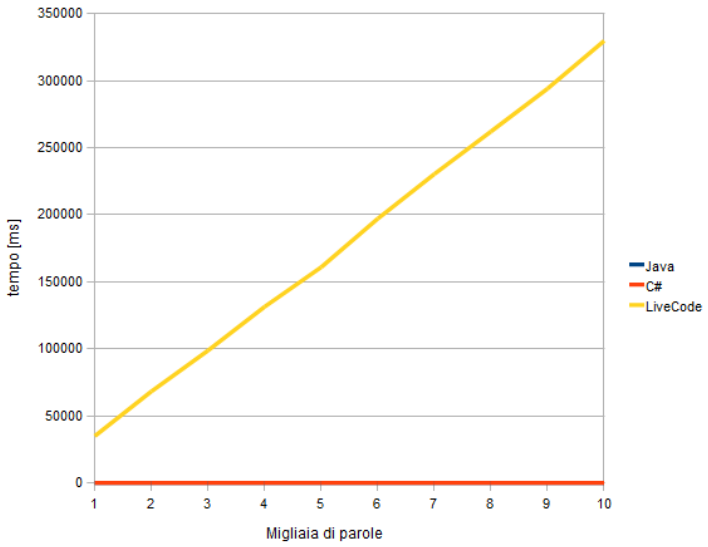

Figura 1.4: Grafico test Pattern Matching

degli ambienti di sviluppo per poi procedere con la realizzazione. Il tempo totale di sviluppo del progetto è stato di sette mesi, quattro dei quali oltre il periodo svolto in tirocinio.

Capitolo 2

SugarCRM

2.1 Customer relationship management

Il costo sostenuto per acquisire un nuovo cliente è di gran lunga superiore a quello necessario per realizzare nuove opportunità sui clienti già attivi; il Customer Relationship Management nasce con l'obiettivo di aiutare le aziende nella conoscenza dei nuovi acquirenti, nel prevederne i loro bisogni e nel rispondere alle segnalazioni col fine di realizzare nuove opportunità di marketing e di lavoro. Un cliente avrà forti motivazioni per restare fedele se ravvisa nel fornitore una significativa attenzione alla sua identità. Il CRM è lo strumento che consente la gestione delle relazioni con i clienti, col fine di averne sempre presente la situazione, prevederne le necessità ed in definitiva mantenere viva in essi l'attenzione per l'azienda. Il CRM si spinge sostanzialmente secondo quattro direzioni differenti e separate:

1. L'acquisizione di nuovi clienti (o "clienti potenziali")

2. L'aumento delle relazioni con i clienti più importanti (o "clienti coltivabili")

3. La fidelizzazione più longeva possibile dei clienti che hanno maggiori rapporti con l'impresa (definiti "clienti primo piano")

Figura 2.1: Logo SugarCRM

4. La trasformazioni degli attuali clienti in procuratori, ossia consumatori che lodano l'azienda incoraggiando altre persone a rivolgersi alla stessa per i loro acquisti

Alcune aziende cercano di non tenere conto di clienti che hanno poca importanza (definiti in gergo clienti sotto-zero) e attuano delle implicite tecniche definite, sempre gergalmente, come Demarketing.

Esistono tre tipi di CRM:

1. CRM operativo: soluzioni metodologiche e tecnologiche per automatizzare i processi di business che prevedono il contatto diretto con il cliente

2. CRM analitico: procedure e strumenti per migliorare la conoscenza del cliente attraverso l'estrazione di dati dal CRM operativo, la loro analisi e lo studio revisionale sui comportamenti dei clienti stessi

3. CRM collaborativo: metodologie e tecnologie integrate con gli strumenti di comunicazione (telefono, fax, e-mail, ecc.) per gestire il contatto con il cliente

L'errore più comune in cui ci si imbatte quando si parla di Customer Relationship Management è quello di equiparare tale concetto a quello di un software: il CRM non è una semplice questione di marketing o di sistemi informatici, bensì si avvale in maniera sempre più massiccia di strumenti informatici o comunque automatizzati per implementare il management. Il CRM è un concetto strettamente legato alla strategia, alla comunicazione, all'integrazione tra i processi aziendali, alle persone e alla cultura, che pone il cliente al centro dell'attenzione sia nel caso del business-to-business sia in quello del business-to-consumer. Le applicazioni CRM servono a tenersi in contatto con la clientela, a inserire le loro informazioni nel database e a fornire loro modalità per interagire in modo che possano essere registrate e analizzate tutte le comunicazioni e attività. Prima di seguire la strada del CRM ogni azienda deve essere consapevole che bisogna investire prima in strategia, organizzazione, comunicazione e solo dopo nella tecnologia; la scelta del software infatti non ha alcun effetto sulla probabilità di successo. Ciò non implica che i software siano tutti uguali, ma significa solo che nessun software porterà al successo un progetto sbagliato. Il CRM è adatto sia a quelle aziende che cercano un Return on investment (ROI) veloce sia a quelle che curano il processo di fidelizzazione e l'aumento del Lifetime value (LTV) dei clienti che richiede del tempo.

2.2 Caratteristiche

SugarCRM è un'applicazione open source leggera, funzionale e completa nelle funzionalità implementate quali la gestione dei clienti, dei relativi contatti, delle opportunità di business e delle richieste di supporto, la pianificazione delle attività e degli appuntamenti. Il prodotto è basato su un Web Service composto da PHP e MySQL, è implementato in modalità multilingua, offre la possibilità di scelta del template grafico, comprende funzioni di utilità per importazione ed esportazione dati. E' inoltre interamente gestibile via browser, web oriented, e si compone delle seguenti applicazioni:

- L'archiviazione e la gestione delle anagrafiche dei clienti e dei potenziali clienti tenendo traccia di come siamo venuti in contatto con l'azienda
- Razionalizzazione dei processi di vendita
- Monitoraggio preciso sullo stato delle trattative
- Automazione delle campagne di marketing
- Maggior controllo sui risultati delle azioni di marketing
- Maggior chiarezza nelle politiche commerciali
- Miglioramento dei rapporti con i clienti
- Gestione centralizzata e condivisa di tutte le informazioni che riguardano il cliente
- Comprensione della frequenza degli incidenti per migliorare la qualità del prodotto
- Condivisione delle informazioni attraverso i singoli utenti e gruppi
- Monitoraggio dello stato dell'intervento
- Misurazione dell'efficacia e dell'efficienza del servizio clienti
- Conservazione dello storico delle attività svolte per un cliente
- Pianificazione e gestione delle attività in corso legate al cliente
- Gestione delle Opportunità di vendita e controllo del loro andamento
- Organizzazione delle attività della forza vendita

SugarCRM, attraverso la sua ben organizzata interfaccia è di facile utilizzo, permette di gestire anche il supporto alla clientela mediante l'apertura di appositi "ticket" assegnabili ai vari dipendenti. SugarCRM dispone anche di un archivio centralizzato che permette di gestire tutta la documentazione aziendale; per assicurarsi di utilizzare sempre documenti validi e aggiornati gli utenti hanno la possibilità di impostare revisioni e scadenze degli stessi. Oltre a queste funzionalità, SugarCRM gestisce anche i seguenti processi:

- Campagne di Marketing e di comunicazione
- Condivisione di Informazioni
- Gestione delle Attività
- Gestione dei Progetti
- Gestione dei Documenti
- Gestione dei reclami e dei difetti di prodotto

Ogni utente di SugarCRM (l'amministratore deve creare un account per ciascuno di loro), può organizzare le proprie attività, le chiamate da effettuare, gli appuntamenti, i compiti da svolgere, seguire le trattative con i clienti e non perdere nessuna opportunità di business.

2.3 Comunicazione API SOAP e REST

Secondo la definizione data dal World Wide Web Consortium (W3C) un Web Service (servizio web come quello offerto da SugarCRM) è un sistema software progettato per supportare l'interoperabilità tra diversi elaboratori su di una medesima rete. La caratteristica fondamentale di un Web Service è quella di offrire un'interfaccia software (descritta in un formato automaticamente elaborabile quale, ad esempio, il Web Services Description Language) utilizzando la quale altri sistemi possono interagire con il Web Service stesso attivando le operazioni descritte nell'interfaccia tramite appositi "messaggi" inclusi in una "busta" (la più famosa è SOAP): tali messaggi sono, solitamente, trasportati tramite il protocollo HTTP e formattati secondo lo standard XML. Grazie all'utilizzo di standard basati su XML attraverso un'architettura fondata sui Web Service (chiamata, con terminologia inglese, Service oriented Architecture - SOA), le applicazioni software scritte in diversi linguaggi di programmazione e implementate su diverse piattaforme hardware possono essere utilizzate tramite interfacce che "espongono" pubblicamente

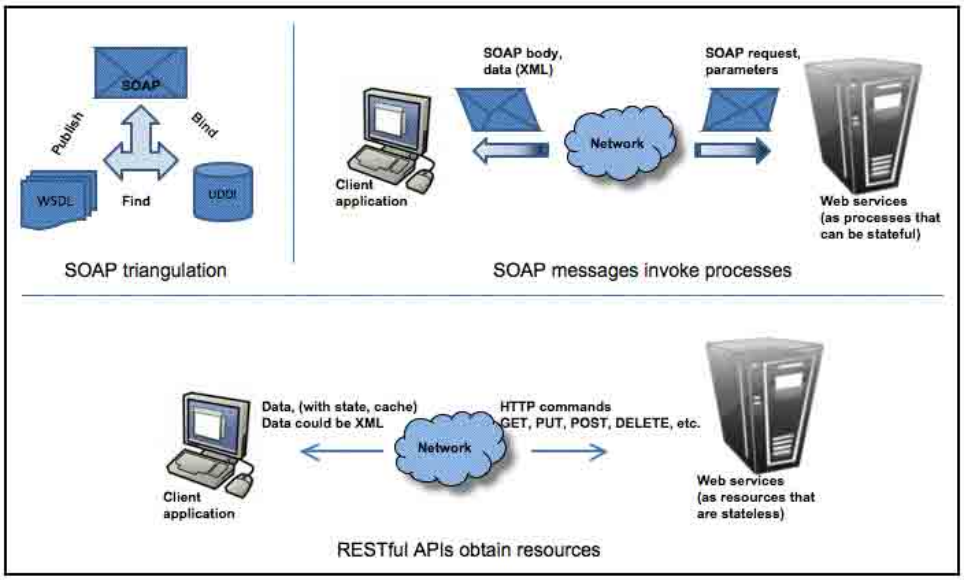

Figura 2.2: Comunicazione SOAP e REST

i servizi e mediante l'utilizzo delle funzioni che sono in grado di effettuare . I servizi composti da scambi di informazioni e da possibili operazioni complesse vengono offerti sia su reti aziendali che su Internet: la possibilità dell'interoperabilità fra diversi linguaggi di programmazione e diversi sistemi operativi è resa possibile dall'uso di standard "aperti". La comunicazione con il servizio web può avvenire attraverso il protocollo SOAP o quello REST. SOAP (inizialmente acronimo di Simple Object Access Protocol) è un protocollo leggero per lo scambio di messaggi tra componenti software (la parola object manifesta che l'uso del protocollo dovrebbe effettuarsi secondo il paradigma della programmazione orientata agli oggetti) e la sua struttura operativa (framework) è estensibile e decentralizzata potendo comunque operare su differenti protocolli di rete anche se HTTP è quello più comunemente utilizzato e l'unico ad essere stato standardizzato dal W3C; il protocollo si basa sul metalinguaggio XML e la sua struttura segue la configurazione Head-Body: il segmento opzionale Header contiene meta-informazioni come quelle che riguardano il routing, la sicurezza, le transazioni e i parametri per l'Orchestration mentre il segmento obbligatorio Body trasporta il contenuto informativo (talora viene detto carico utile) e segue uno schema definito dal linguaggio XML Schema. Soap può essere utilizzato in due modi diversi per una chiamata:

1. Richiesta via SOAP di parametri: il client controlla nel Service Registry l'oggetto d'interesse e sviluppa il messaggio secondo i parametri contenuti nel Service Registry;

2. General Purpose Messaging: un programmatore può sviluppare un suo protocollo privato, il client conosce a priori i parametri e non necessita di consultare il Service Registry.

All'interno del body del messaggio il client sarà tenuto ad inserire i dati scritti nel formato concordato con lo sviluppatore.

REST diversamente da SOAP si riferisce ad un insieme di principi di architetture di rete, i quali delineano come le risorse siano definite e indirizzate. Il termine è spesso usato nel senso di descrivere ogni semplice interfaccia che trasmette dati su HTTP senza un livello opzionale come SOAP o la gestione della sessione tramite i cookie. È possibile progettare ogni sistema software complesso in accordo con l'architettura REST senza usare HTTP e senza interagire con il World Wide Web; inoltre è anche possibile progettare una semplice interfaccia XML+HTTP che non sia conforme ai principi REST ma che invece segua un modello di Remote Procedure Call.

Nel confronto dei due metodi di comunicazione vediamo quindi che REST è un'architettura per la creazione di applicazioni client / server mentre SOAP

è una specifica di protocollo per lo scambio di dati tra due endpoint. Anche se l'utilità dei due metodi è ugualmente valida la scelta viene effettuata in base alle esigenze specifiche; nel nostro caso vedremo che mentre per un client costruito in Java la comunicazione SOAP è pià semplice, per un client HTML-Javascript la strada migliore implica delle chiamate REST.

Capitolo 3

Analisi del Progetto

3.1 Analisi dei Requisiti

3.1.1 Piattaforme richieste

L'applicazione prevede uno sviluppo multipiattaforma, nello specifico gli ambienti di esecuzione Desktop richiesti sono Windows, Mac os x e Linux (anche se quest'ultimo è poco rilevante), mentre per quanto concerne gli ambienti Mobile si considerano i sistemi Android e iOS. L'inclusione di una così vasta scelta di piattaforme è data dalla disomogeneità di utilizzo del gestionale aziendale, il quale prevede come requisito necessario solamente la dotazione di un browser internet e permette l'accesso a tutte le funzionalità basilari tramite la navigazione web. Se in un primo momento la scelta prevedeva lo sviluppo su due strade parallele attraverso altrettante applicazioni (una per i sistemi Desktop e una per quelli Mobile), si è poi deciso di sviluppare una sola applicazione che considerasse la totalità delle piattaforme in un unico progetto cercando soluzioni innovative e probabilmente non ancora confermate dal mercato in modo da arrivare a far rientrare nel progetto tutte le specifiche richieste. La scelta delle piattaforme Mobile (Android e iOS) è stata effettuata tenendo conto sia dei dispositivi disponibili degli utenti finali sia dei probabili acquisti futuri escludendo sistemi come ad esempio Symbian o WebOS; vengono invece presi in considerazione tutti gli ambienti Desktop maggiormente utilizzati.

3.1.2 Modalità offline

Parte dello studio di sviluppo è stata dedicata alla valutazione delle operazioni in modalità offline senza quindi il supporto di internet per interfacciarsi al core dei servizi offerti dal CRM. L'utilità è data dal fatto che spesso gli utenti

finali inseriscono o cercano dati in mobilità appoggiandosi su reti di tipo 3G o (meno spesso) WiFi; questo implica una connessione instabile sia dal punto di vista della disponibilità vera e propria di connessione che da quello della velocità (ci sono operatori che abilitano la connessione solamente in presenza della tecnologia 3G o 3.5G e la disabilitano quando la rete è solo 2G, 2.5G o 2.75G mentre la velocità minima di connessione 3G è di 7,2 Mbps fino ad arrivare ai 42 Mbps ma spesso la connessione arriva a solamente 1 Mbps). La modalità offline viene valutata in base alla metodologia di sviluppo in quanto si tratta di conservare parte dati presenti all'interno del CRM; la scelta di quali dati immagazzinare deve essere studiata in previsione all'utilizzo finale e allo spazio di memoria disponibile. Ulteriori problemi più complessi sono la conservazione dei dati scritti in modalità offline e la sincronizzazione che deve essere fatta successivamente: la scelta più probabile che viene intrapresa è il divieto di scrittura di nuovi dati quando non è presente la connessione o al massimo una parziale possibilità di modificare i dati scritti in precedenza dall'utente stesso.

3.1.3 Gestione dell'upload

L'utente finale ha la possibilità di caricare file fino ad una dimensione massima di 256 MB e i file maggiormente utilizzati sono testi quali .pdf .doc e .odt o tabelle .xls. La possibilità di mantenere questa funzione nel client in studio dipende esclusivamente da due fattori quali la connessione utilizzata e la tecnologia di sviluppo. La connessione implica instabilità nell'utilizzo Mobile, il quale fa cadere il trasferimento dei dati. Si può valutare di ovviare a questo problema utilizzando protocolli studiati ad hoc. La tecnologia di sviluppo implica invece la possibilità o meno di inserire tale funzione: questo comporterà di scegliere se continuare a tener conto dell'upload di file o di escluderlo a priori per garantire maggior prestazioni alle altre funzioni basilari.

3.2 Studio delle Tecnologie

La scelta della modalità di sviluppo viene effettuata confrontando per ogni tecnologia i tre requisiti già citati:

- Piattaforme richieste;
- Modalità offline;
- Gestione dell'upload.

Inizialmente vengono prese in considerazione quattro opzioni:

- Java;
- Objective C;
- LiveCode;
- Interfaccia Web (HTML con il supporto di Javascript e PHP).

3.2.1 Java

La tecnologia Java permette la comunicazione attraverso messaggi SOAP utilizzando il framework Apache Axis il quale implementa un tool, WSDL2Java, in grado di creare automaticamente tutte le classi e gli elementi necessari (sia sorgenti che binari) per colloquiare con SugarCRM partendo dall'interfaccia WSDL che propone quali servizi il web service è in grado di erogare. Per la creazione del client e la comunicazione con il servizio è quindi sufficiente utilizzare la libreria creata e la libreria di Axis e utilizzare le classi incorporate. La difficoltà di utilizzo di queste librerie risiede nei controlli frequenti che si dovranno eseguire: il codice, anche se compilato, provoca spesso errori in esecuzione sui dati che vengono passati da una classe ottenuta dal WSDL ed una della libreria Axis per cui ogni dato immesso al client ha bisogno di un scrupolo controllo appesantendo il codice e la sua esecuzione. Lo sviluppo con Java prevede un supporto per l'intera gamma di piattaforme Desktop; per il Mobile viene escluso l'ambiente iOS (iPhone e iPad) e si dovrà verificare una compatibilità con Android delle librerie.

Per ciò che riguarda la modalità offline con l'utilizzo di Java è possibile la creazione/modifica di un database sul dispositivo in modo da copiare l'intero o parte del database CRM e operare su questo in mancanza di connessione; l'unico limite è costituito dalla disponibilità di memoria che, se da un lato non crea grossi problemi negli ambienti Desktop, dall'altro lato è da verificare nel caso di una possibile compatibilità con Android. L'inoltro di file al CRM è possibile e viene effettuato con il trasferimento diretto del file al server e l'invio di un messaggio SOAP dedicato per aggiornare il database e segnalare la presenza del file.

3.2.2 Objective C

Questo linguaggio di programmazione object-oriented è simile per molti versi a Java ed è inoltre disponile il componente WSDL2objc per creare la libreria di collegamento ai servizi SOAP di SugarCRM partendo sempre dall'interfaccia WSDL. Data la somiglianza a Java si confermano gli ambienti Desktop

mentre si escludono completamente quelli Mobile e si prevedono soluzioni simili sia per la gestione dell'upload che per la gestione della modalità offline.

Una nota particolare riguarda la difficoltà di gestione dei dati che, anche in questo caso, necessitano di numerosi controlli appesantendo il codice e la sua esecuzione ma senza arrivare a essere un problema rilevante data la mancata esecuzione su piattaforme poco potenti quali i dispositivi Mobile.

3.2.3 LiveCode

LiveCode permette una basilare comunicazione SOAP ma, oltre ad essere implementata solo in parte per l'utilizzo da Desktop, dopo alcune prove è stata verificata una non compatibilità con SugrarCRM. Questo linguaggio interpretato permette una programmazione object-oriented e per una comunicazione corretta si può cercare di implementare le classi ottenute in Java dal WSDL; questa soluzione oltre a essere lunga e complessa (alcune classi prevedono oltre 2600 righe di codice) è probabilmente impossibile date le limitazioni di LiveCode sui tipi di dati e sulla programmazione ad oggetti. Vi è anche la possibilità di accedere direttamente al database per l'esecuzione delle operazioni ma bisogna riprogettare in modo completo tutte le query e gestire interamente tutti gli errori oltre a generare correttamente i codici univoci identificatori di ogni entry; deve quindi essere studiata la possibilità di riprogettare l'intero core di SugarCrm.

Lo sviluppo, se possibile, prevede comunque un supporto per le piattaforme Desktop mentre deve essere verificato il supporto per le piattaforme Mobile. La gestione dell'upload viene risolta anche in questo caso inoltrando il file al server aggiornando il database o direttamente o attraverso un messaggio SOAP. LiveCode non permette di creare e gestire dati in un database interno ma si può aggirare in parte il problema gestendo un database in un file di testo, includendo le opportune restrizioni. Vi sono ulteriori limitazioni date dalla dimensione del file che può raggiungere al massimo circa 65000 caratteri e dalla ricerca e modifica che nel file di testo devono essere gestite in modo complesso arrivando al punto di esser troppo pesanti per il sistema di sviluppo. Utilizzando quindi un file di testo si può "gestire" il database in modalità offline ma con notevoli restrizioni per le piattaforme Desktop ed ulteriori (se non impossibili) per le piattaforme Mobile.

3.2.4 Interfaccia Web: HTML5

L'HTML non è un linguaggio di programmazione (in quanto non prevede alcuna definizione di variabili, strutture dati, funzioni, strutture di controllo)

ma solamente un linguaggio di markup che descrive le modalità di impaginazione, formattazione o visualizzazione grafica (layout) del contenuto, testuale e non, di una pagina web attraverso tag di formattazione. Tuttavia l'HTML supporta l'inserimento di script e oggetti esterni quali immagini o filmati e, se utilizzato insieme a un linguaggio di programmazione come Javascript o PHP, può essere adottato per la creazione di un client web utilizzabile dalla totalità dei browser (salvo eccezioni di compatibilità). Un documento HTML comincia con l'indicazione della definizione del tipo di documento (Document Type Definition o DTD) la quale segnala al browser l'URL delle specifiche HTML utilizzate per il documento, indicando quali elementi, attributi ed entità si possono utilizzare e a quale versione di HTML si fa riferimento. Questa informazione serve al browser per identificare le regole di interpretazione e visualizzazione appropriate per lo specifico documento precedendo tutti i tag relativi al documento stesso. Dopo il DTD il documento HTML presenta una struttura ad albero annidato composta da sezioni delimitate da tag opportuni che al loro interno contengono a loro volta sottosezioni più piccole, sempre delimitate da tag. La struttura più esterna è quella che delimita l'intero documento, eccetto la DTD, ed è compresa tra i tag <html> e </html>. All'interno dei tag lo standard prevede sempre la definizione di due sezioni ben distinte e disposte in sequenza ordinata:

- la sezione di intestazione o header, delimitata tra i tag <head> e </head>, che contiene informazioni di controllo normalmente non visualizzate dal browser, con l'eccezione di alcuni elementi

- la sezione del corpo o body, delimitata tra i tag <body> e </body>, che contiene la parte informativa vera e propria, ossia il testo, le immagini e i collegamenti che costituiscono la parte visualizzata dal browser.

Al di sotto di questa suddivisione generale, lo standard non prevede particolari obblighi per quanto riguarda l'ordine e il posizionamento delle ulteriori sottosezioni all'interno dell'header o del body, a parte l'indicazione del rispetto dei corretti annidamenti (le sottosezioni non si devono sovrapporre, ossia ogni sottosezione deve essere chiusa prima di iniziare la sottosezione successiva), lasciando così completa libertà allo sviluppatore o al progettista per quanto riguarda la strutturazione e l'organizzazione successiva. Per ciò che riguarda Javascript e PHP la differenza sostanziale tra i due linguaggi è la modalità di esecuzione del programma client: PHP viene definito server-side scripting o script a esecuzione lato serve mentre Javascript viene definito client-side scripting. Questi due linguaggi infatti si contrappongono nell'esecuzione perché PHP viene eseguito dal server il quale costruisce la pagina web (contenente linguaggio HTML e, se voluto, anche script Javascrip) la

quale, inviata al browser client, viene presentata all'utente dal dispositivo in uso; Javascript invece agisce all'interno del client quindi ad invio dei dati già avvenuto da parte del server. Nell'esecuzione di richieste al server PHP è quindi più favorita perché può fare delle richieste SOAP o REST che non fanno altro che eseguire altro codice PHP (nel nostro caso vi sono classi PHP che rispondono alle richieste SOAP e REST interrogando il database) o in alternativa può interrogare direttamente il database (anche se non è una scelta consigliata data l'alta possibilità di errori nelle interrogazioni dirette); Javascript invece trovandosi in esecuzione sul server deve fare delle richieste REST ben precise che verranno inviate al server e solo in un secondo momento, dopo aver atteso la risposta del server, può creare la pagina web da mostrare all'utente. Nel secondo caso la qualità dipende quindi maggiormente dalla connessione in quanto vi è un numero maggiore di richieste al server. La gestione dell'upload è possibile solamente utilizzando script PHP a causa della perdita della sessione di Javascript nel momento in cui vi è il trasferimento di dati diversi da chiamate SOAP o REST nella pagina web utilizzata come client. E' possibile tuttavia risolvere la questione utilizzando PHP e Javascript contemporaneamente (anche per il fatto che comunque il server utilizza PHP per la funzione di core di SugarCRM).

3.3 Riunioni e Sviluppi proposti

Nella prima riunione è stato richiesta agli utilizzatori finali una documentazione su tutte le esigenze principali che il client deve avere dal punto di vista di funzionalità di cui le seguenti sono quelle rivelate più utili:

Aziende e Contatti
- funzione di ricerca;
- tasto per l'inserimento di un nuovo contatto o di una nuova azienda;
- vista del contatto o dell'azienda con i campi nome e cognome, numeri vari di telefono, lista di indirizzi e-mail, lista di vari indirizzi, descrizione, impiegato a cui è assegnato il contatto o l'azienda, stato del contatto o dell'azienda;

Dati Studio
- piattaforme utilizzate nello studio;
- software posseduto;
- eventuali postazioni radiografiche;
- postazioni riunite e totali del cliente;

Cronologia dei seguenti punti ordinata per data e di un determinato periodo di tempo

- E-mail spedite
- E-mail inviate
- Attività
- Chiamate
- Note

Opportunità
- possibilità di selezionare una serie di opportunità preconfigurate con una modalità di multiscelta;
- senza scadenza;
- con data di inserimento;
- con una descrizione ben visibile;
- con un semaforo che segnali lo stato dell'opportunità ;

Evento chiamata L'inserimento di una nuova chiamata deve avere

- semafaro di riferimento rosso o verde secondo lo stato della chiamata
- pulsante per ripianificare la chiamata (permette di chiudere una telefonata e pianificare una nuova chiamata)
- descrizione chiamata
- data e ora
- durata in minuti selezionabile
- utente a cui è assegnata la chiamata
- alert flag per mandare mail con dati della chiamata all'utente a cui è assegnata
- alert flag per mandare mail con dati della chiamata al responsabile dell'utente a cui è assegnata

Evento attività L'inserimento di una nuova attività deve avere

- semafaro di riferimento rosso o verde secondo lo stato della attività
- pulsante per ripianificare l'attività (permette di chiudere un'attività e pianificarne una nuova)
- descrizione attività

- data e ora
- durata in minuti selezionabile
- utente a cui è assegnata l'attività
- tipologia dell'attività selezionabile
- alert flag per mandare mail con dati dell'attività all'utente a cui è assegnata
- alert flag per mandare mail con dati dell'attività al responsabile dell'utente a cui è assegnata

Evento nota L'inserimento di una nuova nota deve avere

- descrizione nota
- data e ora
- utente a cui è assegnata la nota
- alert flag per mandare mail con dati dell'attività all'utente a cui è assegnata
- alert flag per mandare mail con dati dell'attività al responsabile dell'utente a cui è assegnata

Dashboard contenente tutti i dati e le funzioni di telemarketing, in particolare

- Chiamata con possibilità di chiudere, modificare, vedere in dettaglio il soggetto, l'utente a cui è assegnata e la data di inserimento della telefonata;
- E-mail non risposte contenente il soggetto, l'oggetto e la data di arrivo;
- Attività con possibilità di chiudere, modificare, vedere in dettaglio il soggetto, l'utente a cui è assegnata e la data di inserimento dell'attività;

Dopo aver capito con gli utenti finali quali sono le reali esigenze è apparso chiaro che la possibilità di avere anche pochi dati da inserire e da visualizzare ma in modo veloce e intuitivo era preferibile ad avere più dati ma lenti o inseribili in modo disorganizzato. Un esempio significativo è la stessa interfaccia di SugarCRM: per inserire una nuova chiamata è possibile inserire una moltitutide di dati:

- Oggetto
- Stato

- Data e ora di inizio
- Modulo a cui viene allegata la chiamata (Azienda, Contatto..)
- Entità del modulo selezionato a cui viene allegata la chiamata (se viene selezionato il modulo azienda l'entità è l'azienda a cui viene allegata)
- Alert per avere un popup sul browser che ricorda la chiamata da effettuare
- Descrizione
- Utente a cui viene assegnata la chiamata

Oltre ad avere tutte queste informazioni da introdurre l'interfaccia non è intuitiva e tutte le scelte devono essere fatte da liste che negli ambienti mobile comportano un lungo caricamento dell'interfaccia e un'alta difficoltà di navigazione.

Per questi motivi viene stabilito che deve essere posta un'attenta analisi su quali dati devono essere richiesti all'utente per la creazione e la modifica di:

- Aziende
- Contatti
- Chiamate
- Attività
- Note

seguendo le richieste fatte nella documentazione proposta precedentemente e distinguendo tra creazione e modifica.

Per le scelte di visualizzazione dei dettagli delle singole entità deve essere invece analizzato quali dettagli mostrare a seconda di dove vengono visualizzati: ad esempio una chiamata sarà visualizzata diversamente nel caso in cui sia mostrata su una cronologia (meno informazioni) o su una lista delle chiamata da effettuare in giornata (più informazioni). Una ulteriore richiesta degli utenti è la creazione di un nuovo modulo Opportunità contenente i prodotti di vendita dell'azienda e i preventivi con la possibilità di creare nuovi preventivi aggiungendo prodotti preconfigurati nella descrizione e nelle caratteristiche e di assegnarli ad utenti.

3.4 Studio di Fattibilità

Dopo aver completato le interviste agli utenti principali e studiato i requisiti raccolti viene scelto di creare un nuovo client sviluppando una struttura e una logica tale da seguire il seguente schema:

- l'utente deve partire da una schermata che definiremo Home in cui ha la possibilità di vedere le chiamate e le attività a lui assegnate per il giorno di utilizzo;
- dalla Home l'utente ha la possibilità di creare:
  - nuove Aziende
  - nuovi Contatti
  - nuove Chiamate
  - nuove Attività
  - nuove Note
  - nuove Opportunità di lavoro

  e di assegnarle ad un utente tra quelli presenti;
- dalla Home l'utente ha la possibilità di accedere a quattro sezioni principali: tre saranno liste di Aziende, di Contatti e di Opportunità mentre l'ultima sarà la Dashboard relativa all'utente;
- la lista Azienda e la lista Contatti indicheranno le Aziende e i Contatti presenti all'interno del CRM e dai quì l'utente potrà
  - modificare i dettagli relativi ad ogni Azienda/Contatto
  - visualizzare i dettagli relativi ad ogni Azienda/Contatto
  - assegnare ad ogni Azienda/Contatto nuove chiamate, note, attività o opportunità
- dalla lista delle Opportunità l'utente potrà visualizzare i dettagli relativi a ciuscuna di quelle presenti, di cancellarle e di crearne di nuove;
- la Dashboard dà la possibilità ad ogni utente di visualizzare le attività e le chiamate principali organizzandole con una modalità simile ad un diario, in particolare sono presenti tre sottosezioni:
  - Chiamate: qui l'utente può visualizzare le chiamate e i dettagli relativi a partire da un mese prima al giorno di utilizzo fino ad un mese dopo evidenziando soprattutto quelle del giorno di utilizzo;

- Attività: qui l'utente può visualizzare le attività e i dettagli relativi a partire da un mese prima al giorno di utilizzo fino ad un mese dopo evidenziando soprattutto quelle del giorno di utilizzo;
- E-mail: quì l'utente può visualizzare le sue email inviate e ricevute organizzate in modo cronologico.

Per ogni chiamata, ogni attività e ogni oppurtunità l'utente può inoltre svolgere delle modifiche instantanee per alterarne lo stato: ad esempio deve essere possibile segnalare la chiusura di una chiamata senza andare a modificarne tutti i dettagli ma solamente lo stato.

3.5 Modalità di Sviluppo

Le tecnologie candidate per la creazione del client sono quattro:

- Java;
- Objective C;
- LiveCode;
- Interfaccia Web.

Per ognuna di queste tecnologie abbiamo studiato i pregi e i difetti soprattutto per i punti chiavi del progetto, ossia:

- Livello di supporto multi piattaforma;
- Modalità offline;
- Gestione dell'upload.

Mentre per gli ultimi due punti non sono state dichiarate delle esigenze particolari, le piattaforme supportate devono essere il più alto numero possibile. Si sono riscontrati infatti utilizzi del CRM sia da postazioni Desktop che Mobile e gli ambienti corrispondono a tutti i sistemi operativi di maggior utilizzo; dobbiamo quindi dare priorità ad una scelta che comporta l'utilizzo del client nel maggior numero di ambienti.

Lo schema nella tabella 3.1 riassume lo studio delle tecnologie prese in esame e vediamo che, dando priorità allo sviluppo multi piattaforma, LiveCode e l'Interfaccia Web coprono tutti gli ambienti interessati mentre Java esclude solamente iOS.

Tecnologia	Piattaforme supportate	Modalità offline	Gestione upload
Java	Tutte tranne iOS	Sì	Sì
Objective C	Solamente ambienti Desktop	Sì	Sì
LiveCode	Tutti gli ambienti	Sì	Sì
Interfaccia Web	Tutti gli ambienti	No	Da valutare

Tabella 3.1: Comparazione tecnologie di sviluppo del client

A questo punto si è voluto capire se iOS sia veramente importante per gli utenti perché l'implementazione in Java, oltre ad esser più stabile, dà maggiori garanzie anche per l'upload e per la modalità offline. Il responsabile aziendale ha sottolineato che altri progetti interni prevedono la dotazione di tutti i commerciali di un iPad per le operazioni via internet e per presentazioni a clienti per cui probabilmente comporterà un utilizzo del CRM da ambienti iOS. Queste informazioni orientano quindi una scelta che va ad escludere uno sviluppo con Objective C e Java.

Il confronto tra LiveCode e l'Interfaccia Web viene valutato non secondo i punti chiave del progetto ma in base alla stabilità che il progetto potrà poi avere: abbiamo già visto che LiveCode non ha prestazioni valide come un normale linguaggio di programmazione e che la comunicazione con il Web Service di SugarCRM non funziona correttamente; l'interfaccia Web invece permette una comunicazione sia SOAP che REST, è dotata di linguaggi di programmazione validi quali Javascript e PHP dove si può trasferire il carico di lavoro ad un server centrale rendendo il client ulteriormente leggero e l'interfaccia può essere creata in modo versatile e flessibile.

Le considerazioni fatte portano quindi ad una scelta finale che prevede lo sviluppo di una Interfaccia Web, dotata di codice Javascript per tutto ciò che riguarda l'elaborazione interna del client e, se necessario, programmi PHP per rendere il programma più leggero.

L'interfaccia grafica sarà assegnata al framework jQuery Mobile che permette di utilizzare in modo flessibile il client su ambienti Desktop e Mobile, organizzando i dati e le schermate in modo ottimale basandosi sulle dimensioni del display che l'utente andrà ad utilizzare.

Capitolo 4

JQuery Mobile

4.1 Introduzione

JQuery Mobile è un framework per tutte le piattaforme mobile più utilizzate. Costruito sulla solida base di jQuery e jQuery UI, permette di creare applicazioni e pagine web in modo semplice e veloce. L'utilizzo necessita delle librerie e dei CSS (fogli di stile) del framework che agiscono come core per formattare la pagina nel browser dei dispositivi mobile, adattandone i contenuti in modo flessibile e dotando l'interfaccia di particolari strumenti in modo da rendere l'interazione con l'utente semplice. Tutti i contenuti principali quali bottoni, sezioni di input, switch, menù e checkbox sono stati sostituiti nella grafica da contenuti personalizzati che meglio si adattano a dispositivi mobile. Per l'utilizzo del framework è sufficiente utilizzare codice html ma, dato che per applicazioni web è necessario l'utilizzo del linguaggio javascript, vengono in aiuto funzioni javascript delle librerie jQuery Mobile che aiutano nei controlli e nella esecuzioni della pagina web. Vi è poi la possibilità di personalizzazione delle pagine con cinque temi di default o in alternativa altri temi creati dall'utente con l'utilizzo dello strumento proposto chiamato ThemeRoller.

Il framework è inoltre supportato da molte case note che sponsorizzano il progetto quali Nokia, BlackBerry, Mozilla corporation, Filament group,

Figura 4.1: Logo jQueryMobile

Palm, DeviceAtlas, Adobe, Jive, Rhomobile e DotMobi.

4.2 Caratteristiche

La caratteristica principale del framework è una struttura base che si appoggia al più noto jQuery (necessario nelle pagine html) rendendo disponibile una vasta e consolidata libreria javascript e html con l'ulteriore capacità di avere la flessibilità e la comodità di integrarsi perfettamente ai browser di dispositivi mobile senza pregiudicarne l'utilizzo su ambienti Desktop. Un'altra possibilità offerta è la gratuità del progetto sostenuto solamente con gli introiti offerti dagli sponsor e dalla comunità. Il framework ha anche la capacità di riconoscere gli oggetti di tipo event e provvede a modificare le loro proprietà rendendoli uniformi, semplificando la loro gestione, la loro propagazione, e fornendo un'utile modalità per impedire al browser di continuare l'esecuzione (ad esempio sulla onclick di un link). L'assegnazione di eventi quali click, load, mouseover è gestita in maniera efficace e non invadente e vengono riconosciuti eventi di tipo touch quali tap, taphold, swipe e scroll. Sono disponibili infine molte possibilità di effetti come ad esempio si possono impostare effetti di transizione delle pagine web come slide, pop e flip.

4.3 Piattaforme supportate

JQuery Mobile distingue tre gradi di supporto del framework:

1. grado A: supporto completo comprensivo delle animazioni e degli effetti basati su Ajax

2. grado B: supporto completo senza le animazioni e gli effetti basati su Ajax

3. grado C: supporto di base verificato ma non garantito.

In particolare nella prima versione stabile del framework i dispositivi supportati divisi per grado sono i seguenti:

1. Grado A

    - Apple iOS 3,2-5,0 - Testato su iPad (4.3 / 5.0), iPad 2 (4.3), iPhone (3.1), iPhone 3 (3.2), 3GS (4,3), e 4 (4.3 / 5.0)
    - Android 2,1-2,3 - Testato sul HTC Incredible (2,2), Droid (2,2), Nook (2,2), HTC Aria (2,1), Google Nexus S (2,3)

- Android Honeycomb - Testato sul Samsung Galaxy Tab 10.1 e Motorola XOOM
- Windows Phone 7-7,5 - Testato su HTC Surround (7,0) HTC Trophy (7,5), LG-E900 (7,5) 6,0 Blackberry Torch 9800 e Style 9670
- Blackberry 7 - Testato su BlackBerry Torch 9810
- Blackberry Playbook - Testato su PlayBook versione 1.0.1 / 1.0.5
- Palm WebOS (1,4-2,0) - Testato sui Palm Pixi (1,4), Pre (1,4), Pre 2 (2.0)
- Palm WebOS 3.0 - Testato su HP TouchPad
- Firebox Mobile (Beta) - Testato su Android 2.2
- Opera Mobile 11.0: Testato su iPhone 3GS e 4 (5.0/6.0), Android 2.2 (5.0/6.0)
- Meego 1.2 - Testato su 950 e Nokia N9
- Chrome Desktop 11-15 - Testato su OS X 10.6.7 e Windows 7
- Firefox 4-8 Desktop - Testato su OS X 10.6.7 e Windows 7
- Internet Explorer 7-9 - Testato su Windows XP, Vista e 7 (minori problemi di CSS)
- Opera Desktop 10-11 - Testato su OS X 10.6.7 e Windows 7

2. Grado B

    - Blackberry 5.0: Testato su Storm 2 9550, Bold 9770
    - Opera Mini (5,0-6,0) - Testato su iOS 3.2/4.3
    - Nokia Symbian v 3 - Testato su Nokia N8 (Symbian v 3), C7 (Symbian v 3), N97 (Symbian v 1)

3. Grado C

    - Blackberry 4.x - Testato su Curve 8330
    - Windows Mobile - Testato su HTC Leo (WinMo 5.2)

Capitolo 5

Sviluppo

5.1 Suddivisione dei moduli di lavoro

Con l'avvio del progetto è stato necessario individuare dei macro moduli di lavoro indipendenti tra di loro su cui concentrarsi in modo da avere delle scadenze indicative per ognuno di essi. I principali moduli individuati sono:

- Azienda
- Contatto
- Chiamata
- Nota
- Attività
- Opportunità
- Dashboard
- Invio notifiche
- Gestione dell'upload

I moduli Azienda e Contatto sono molto simili e per questo motivo sono stati studiati e realizzati in modo diverso: prima è stato realizzato il modulo Azienda in tutte le sue funzionalità e con tutte le accuratezze possibili, dopodiché è stato replicato nel modulo Contatto apportando le modiche e personalizzazioni richieste. I moduli Chiamata Attività e Nota sono stati sviluppati inizialmente solamente per il modulo Azienda in modo da assegnare nuove chiamate, nuove attività e nuove note a ogni azienda nuova o

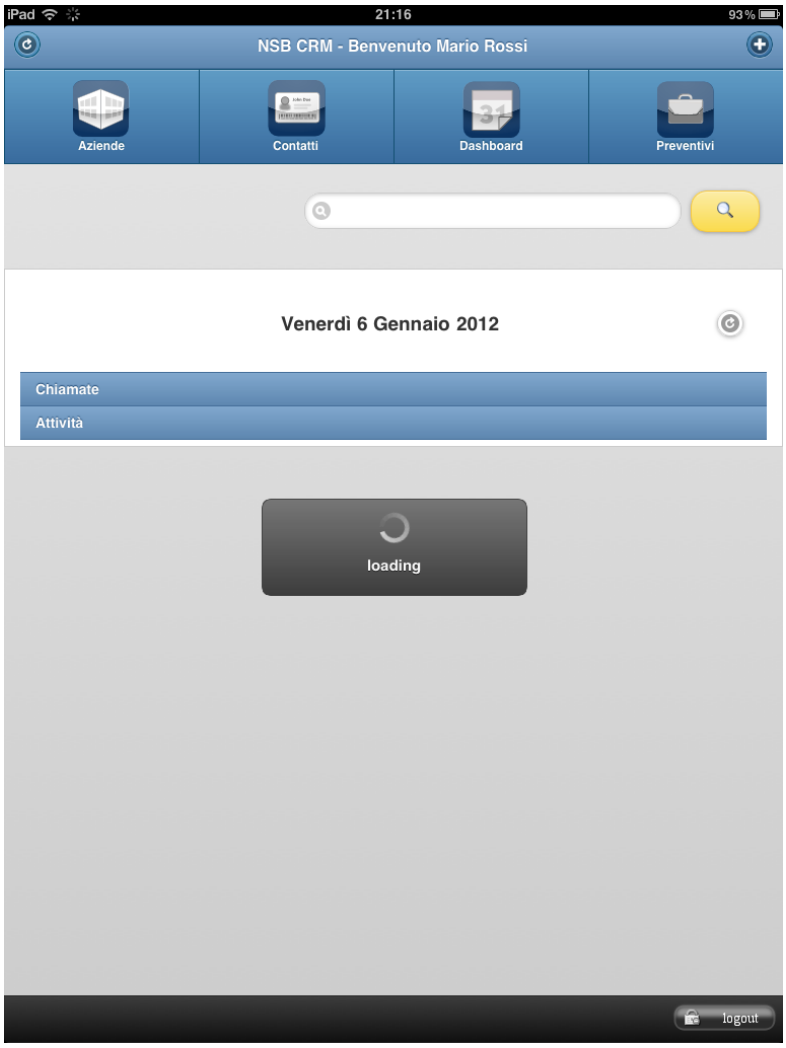

Figura 5.1: Schermata Home

già presente nel database; nel momento in cui si è arrivati a una situazione stabile e dopo aver replicato la struttura dell'Azienda per il Contatto anche queste parti sono state copiate per il modulo Contatto. La Dashboard è stata realizzata in una fase intermedia, ossia prima di replicare il modulo azienda nel modulo contatto, in modo da apportare le modifiche richieste dagli utenti nelle interazioni tra i moduli azienda e dashboard.

In questo modo si poteva disporre di un client già in una situazione stabile ma soprattutto disponibile per alcune prove riguardo prestazioni e debug: mentre alcuni utenti scelti eseguivano i test il lavoro è continuato per implementare le notifiche e la gestione dell'upload. Quando i test sono conclusi in un primo momento si sono discusse le modifiche da eseguire per poi apportare solamente quelle decise e fondamentali e in seguito è stata rilasciata una prima Beta del progetto.

5.1.1 Aziende e Contatti

Questi due moduli principali anche se molto simili tra loro distinguono due concetti distinti:

- per Azienda si intende una singola attività commerciale e non fisica, dotata di una sede propria e con persone fisiche che lavorano per essa;

- per Contatto si intende una singola persona fisica che può lavorare per una azienda o per se stessa.

Si sottolinea quindi il fatto che un contatto può appartenere a una azienda e che una azienda può avere uno o più contatti relativi. Per entrambi i moduli è necessario creare delle sezioni per la creazione, la modifica, la vista dei dettagli di ogni entità e la lista delle entità appartenenti al modulo; in particolare per la creazione, la modifica e la vista dei dettagli devono esser presenti gli attributi della tabella 5.1.

I dati studio sono inseriti per tutte le aziende e i contatti in quanto i maggiori clienti trattati sono studi medici o studi ortodontici inserendo campi nuovi anche all'interno del database Sugarcrm. In particolare i dati studio contengono le informazioni:

- piattaforme utilizzate

- descrizione dell'eventuale software utilizzato al momento di conoscenza del cliente

- descrizione degli eventuali sistemi radiografici del cliente

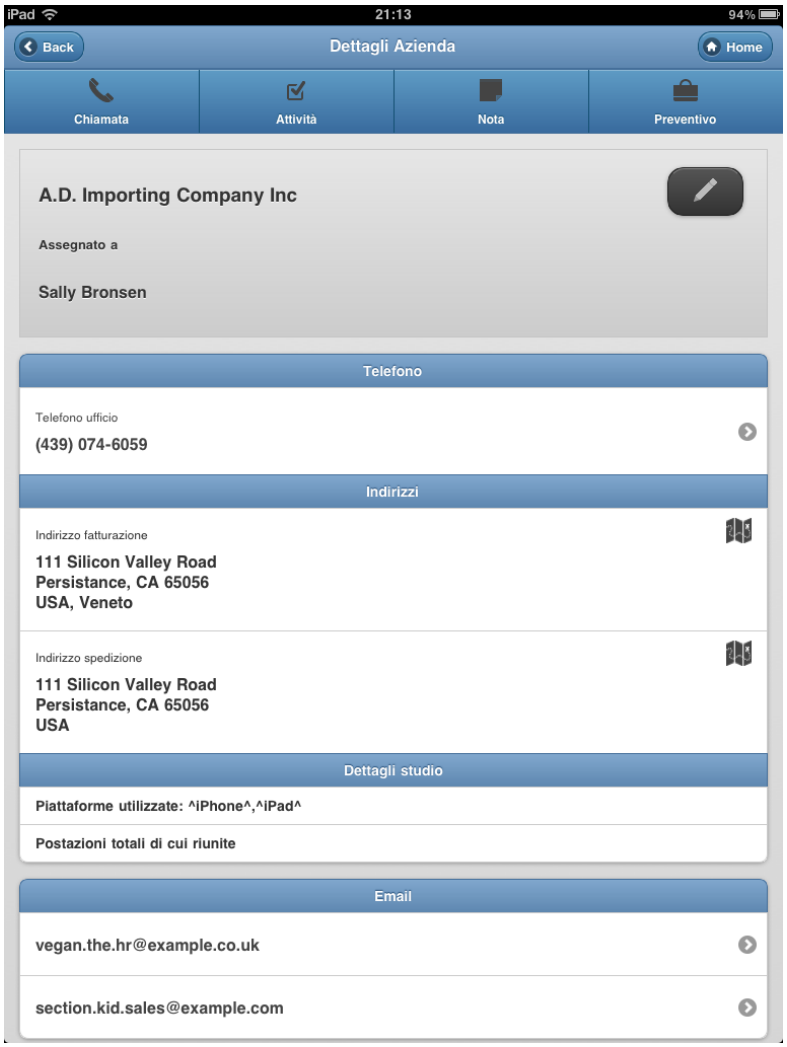

Figura 5.2: Schermata Dettagli relativi ad una azienda

Azienda	**Contatto**
Nome	Nome e Cognome
Stato	Stato
Utente a cui è assegnata	Utente a cui è assegnato
	Azienda relativa
Indirizzo Fatturazione	Indirizzo Primario
Indirizzo Spedizione	Indirizzo Secondario
Fax	Fax
Telefono ufficio	Telefono ufficio
Telefono alternativo	Telefono cellulare
	Telefono abitazione
	Altro telefono
E-mail 1	E-mail 1
E-mail 2	E-mail 2
E-mail 3	E-mail 3
Dati Studio	Dati Studio

Tabella 5.1: Attributi per i moduli Azienda e Contatto

Chiamata	Attività	Nota
Oggetto	Oggetto	Oggetto
Descrizione	Descrizione	Testo
Azienda/Contatto relativo	Azienda/Contatto relativo	Azienda/Contatto relativo
Utente a cui è assegnata	Utente a cui è assegnata	Utente a cui è assegnata
Direzione	Priorità	
Stato	Stato	
Data e ora di inizio	Data e ora di inizio	
Durata	Durata	
Notifica utente	Notifica utente	Notifica utente
Notifica responsabile	Notifica responsabile	Notifica responsabile

Tabella 5.2: Attributi per i moduli Chiamate, Attività e Note

- postazioni totali e riunite dello studio

Oltre a tutti questi dati nella sezione di dettaglio devono essere visibili chiamate, attività note e opportunità di relativi all'azienda o al contatto con la possibilità di assegnarne nuove. Il lavoro è iniziato sul modulo Azienda creando la prima pagina con la lista di tutte le aziende disponibili; da quì si è poi passati alla realizzazione della pagina di dettaglio e poi alle pagine di creazione e di modifica dell'azienda. Terminata l'Azienda in modo completo è stato replicato tutto nel modulo Contatto con le modifiche necessarie e rilevabili anche dalla tabella.

5.1.2 Chiamate Attività e Note

La fase successiva del progetto ha interessato i tre moduli già previsti e utilizzati dagli utenti in SugarCRM: Chiamate, Attività e Note. Questi tre moduli sono distinti tra di loro in quanto consistono in tre operazioni diverse ma vengono studiati e progettati insieme perché ognuno di questi moduli si collega nello stesso modo ai moduli Azienda e Contatto. Sono richieste delle sezioni di creazione e di vista dei dettagli di ogni operazione mentre la modifica è necessaria solamente per alcuni attributi di chiamate e attività. I dettagli richiesti per ogni modulo sono proposti nella tabella 5.2.

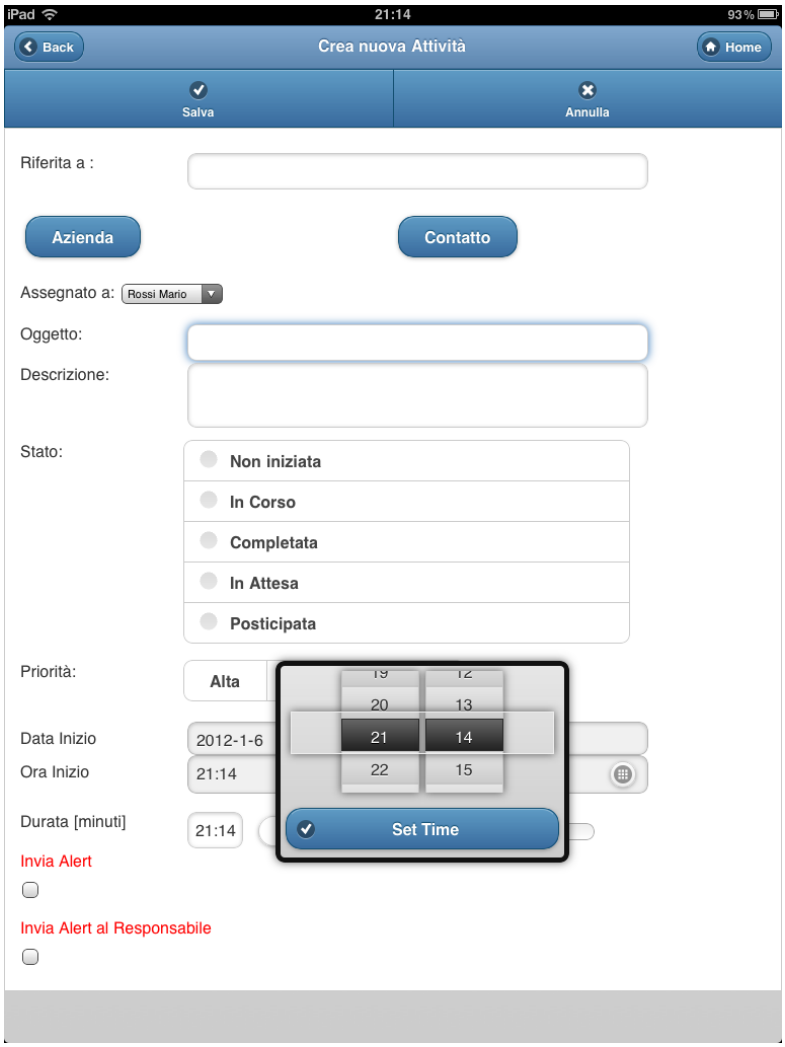

Figura 5.3: Schermata di creazione di una nuova attività

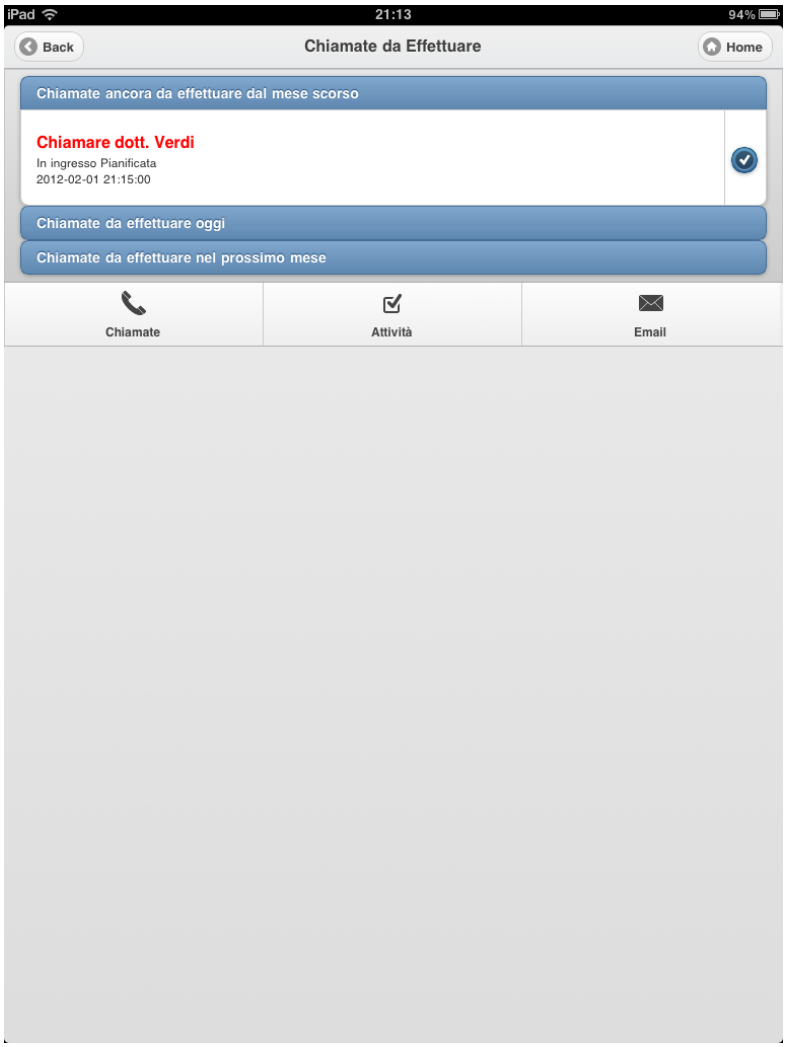

Figura 5.4: Schermata della Dashboard

5.1.3 Dashboard

La Dashboard serve all'utente per visualizzare la programmazione delle sue giornate lavorative e valuta un periodo di tempo da un mese prima a un mese dopo rispetto il giorno di utilizzo del client.

Le informazioni rappresentate sono:

- le chiamate a lui assegnate e non ancoara chiuse, con la possibilità di modificarle, di chiuderle e di crearne di nuove;
- le attività a lui assegnate e non ancoara concluse, con la possibilità di modificarle, di chiuderle e di crearne di nuove;
- le email del proprio indirizzo di posta, suddivise in inviate e ricevute.

Tutti le tre sezioni della Dashboard sono implementabili tramite chiamate REST; anche la sezione che comprende le email infatti viene gestita da SugarCRM con un modulo indipendente. Il CRM dà la possibilità di inviare e di ricevere posta elettronica e ad ogni sincronizzazione con il server di posta salva automticamente tutte le email nel proprio database. In questo modo è possibile ottenere una lista di tutte le email ricevute ed inviate ordinandole cronologicamente.

5.2 Comunicazione REST via JavaScript

Per il trasferimento dei dati dal web service al client viene creata una comunicazione di tipo REST attraverso una funzione Javascript in cui, in base ai parametri passati, viene restituito un array con tutte le informazioni richieste.

La struttura seguita è sempre la stessa sia nel caso in cui si voglia interrogare il database che nel caso sia necessario eseguire inserimenti o modifiche dei dati; viene rappresentata con il codice 5.1.

Codice 5.1: Richiesta REST

```
$.get("../service/v2/rest.php", {
        method: "get_entry_list", //selezione della funzione REST
        input_type: "JSON",
        response_type: "JSON",
        rest_data: {
                "session":"'+SugarSessionId+'", //Id di Sessione
                "module_name":"Accounts",   //nome modulo
                "query":"",      //query SQL integrabile
                "order_by":"name",    //tipo di ordinamento
                "offset":0, //offset dei dati
```

```
11                  "select_fields"[     //selezione attributi
                            "name",
13                          "billing_address_city",
                            "billing_address_state"
15                          ],
                    "link_name_to_fields_array":"", //moduli collegati
17                  "max_results":1000,    //numero massimo
                    "deleted":0       //numero di dati cancellati
19          }
```

Dall'esempio possiamo vedere che inizialmente viene inserito il metodo della chiamata REST ossia la funzione che viene applicata; i metodi più frequesti sono:

- login(), restituisce l'id necessario per tutte le altre operazioni
- logout(), chiude la sessione
- get_entry_list(), restituisce una lista di dati
- get_entries(), restituisce le informazioni riguardo una singola entità
- set_entry(), inserisce o modifica dati
- set_relationship(), crea un collegamento tra due entità

L'input_type e il response_type indicano il tipo di input inserito e il tipo di output restituito nella chiamata; in questo caso viene selezionato in entrambi il tipo JSON che crea degli array associativi ossia sequenze chiave-valore che per ogni attributo indica il valore relativo. Nell'esempio vediamo che i dati REST vengono creati indicando per ogni chiave (session, module_name, query, order_by, offset, select_fields, link_name_to_fields_array, max_results e deleted) i dati relativi necessari o non.

I tipi di dati disponibili sono tutti quelli previsti da Javascript ma bisogna porre attenzione in quanto se per ogni chiave non viene immesso il tipo di dato relativo la chiamata non andrà a buon fine. Per controllare i tipi di dato richiesti per ogni attributo è disponibile un'ulteriore funzione REST, utilizzabile solo nel periodo di progettazione, get_module_fields che indica tutti gli attributi del modulo selezionato e il tipo di dato di ciascuno.

Infine all'interno dei dati REST (rest_data) deve essere indicato il modulo su cui si opera attraverso la chiave module_name.

Tutta la documentazione offerta da SugarCRM che tratta di queste chiamate è disponibile sul web nella pagina ufficiale ma è risultata scarsa ai fini del progetto; ci sono molte mancanze su tutti i casi particolari delle funzioni

e solamente con numerose prove e richieste nei forum si è potuto capire la struttura da eseguire per ogni richiesta REST.

5.3 Utilizzo del framework jQuery Mobile

L'utilizzo del framework jQuery Mobile serve a rendere l'interfaccia compatibile con tutti gli ambienti da supportare e allo stesso tempo ad organizzando i contenuti in maniera ottimale.

La creazione di pagine web con la grafica gestita dal framework avviene inserendo nel codice HTML i seguenti punti:

1 bisogna inizialmente inserire nella sezione header i collegamenti a jQueryMobile o ai file corrispondenti come nel codice 5.2

Codice 5.2: Link a jQueryMobile

```
1
<link rel="stylesheet" href="http://.../jquery.mobile-1.0.1.min.css" />
3 <script src="http://.../jquery-1.6.4.min.js"></script>
<script src="http://.../jquery.mobile-1.0.1.min.js"></script>
```

in cui jquery.mobile-1.0.1.min.css è un foglio di stile per gli elementi di grafica e jquery.mobile-1.0.1.min.js è un file Javascript per l'organizzazione all'interno della pagina; viene anche inserito il Javascript relativo a jQuery jquery-1.6.4.min.js in quanto jQueryMobile ha la necessità di appoggiarsi;

2 si utilizzano poi le componenti HTML relative a jQueryMobile all'interno della sezione body.

Vi sono molti contenuti che sostituiscono i tag standard HTML con quelli proposti da JqueryMobile e per un uso corretto si fa riferimento alla documentazione fornita in modo esaustivo.

La semplicità dell'HTML viene infatti in aiuto in quanto per comporre i dati organizzandoli in determinati modi si opera sempre nello stesso modo: ad esempio se è necessario organizzare i contenuti dividendo in tre parti la pagina è semplice utilizzare il codice 5.3:

Codice 5.3: jQueryMobile: Organizzazione della pagina

```
1
<div class="ui-grid-b">
3       <div class="ui-block-a">Primo Blocco</div>
```

```
        <div class="ui-block-b">Secondo Blocco</div>
        <div class="ui-block-c">Terzo Blocco</div>
</div><!-- end /grid-b -->
```

in cui ui-grid-b indica il numero di sezioni in base alla lettera (b in questo caso divide in tre sezioni, c in quattro e così via).

Un altro contenuto molto utilizzato è la costruzione di liste dati come nell'esempio del codice 5.4

Codice 5.4: jQueryMobile: Creazione liste

```

<h2>Divided, filterable list</h2>
        <ul data-role="listview" data-filter="true" data-inset="true">
                <li data-role="list-divider">A</li>
                <li><a href="index.html">Adam Kinkaid</a></li>
                <li><a href="index.html">Alex Wickerham</a></li>
                <li><a href="index.html">Avery Johnson</a></li>
                <li data-role="list-divider">B</li>
                <li><a href="index.html">Bob Cabot</a></li>
                <li data-role="list-divider">C</li>
                <li><a href="index.html">Caleb Booth</a></li>
                <li><a href="index.html">Christopher Adams</a></li>
        </ul>
```

dove data_filter crea un campo di ricerca diretto degli elementi della lista, list-divider indica che l'elemento divide gli elementi della lista e i tag <li> contrassengano ogni singolo elemento.

5.4 Personalizzazioni e Moduli Custom

5.4.1 Attributi personalizzati

Le richieste degli utenti comprendono personalizzazioni dei moduli Azienda e Contatto in modo da avere tutte le informazioni necessarie per ogni cliente ma allo stesso tempo solamente le più importanti. Per far ciò si può operare in due passi:

- Filtrare le informazioni richieste rispetto a tutte quelle disponibili
- Aggiungere nuovi attributi ai due moduli compatibilmente alle richieste

Il primo punto si può risolvere in modo immediato selezionando gli attributi richiesti al momento di costruzione delle richieste REST, questo viene fatto inserendo il codice 5.5;

Codice 5.5: Personalizzazione attributi nella richiesta

```
$.get("../service/v2/rest.php", {
        method: "get_entry_list",
        input_type: "JSON",
        response_type: "JSON",
        rest_data: {
                "session":"'+SugarSessionId+'",
                "module_name":"Accounts",
                "query":"",
                "order_by":"name",
                "offset":0,
                "select_fields"[
                        "name",
                        "billing_address_city",
                        "billing_address_state"
                        ],
                "link_name_to_fields_array":"",
                "max_results":1000,"deleted":0
        }
```

in questo caso l'istruzione richiede via REST gli attributi

- name
- billing_address_city (città relativa all'indirizzo di spedizione)
- billing_address_state (stato relativo all'indirizzo di spedizione)

di ogni Azienda disponibile nel CRM.

Il secondo punto comporta invece una modifica alla struttura del database utilizzato da Sugarcrm; l'aggiunta di nuovi attributi è prevista dal CRM e si può effettuare grazie al modulo Studio accessibile via web dall'amministratore di sistema e situato nella pagina di configurazione. Studio ha la particolarità di mostrare ogni modulo comparandolo ad una cartella dei normali sistemi desktop; ogni modulo è strutturato così:

- Nome Modulo
  - Etichette
  - Campi
  - Relazioni
  - Maschere
    - di modifica

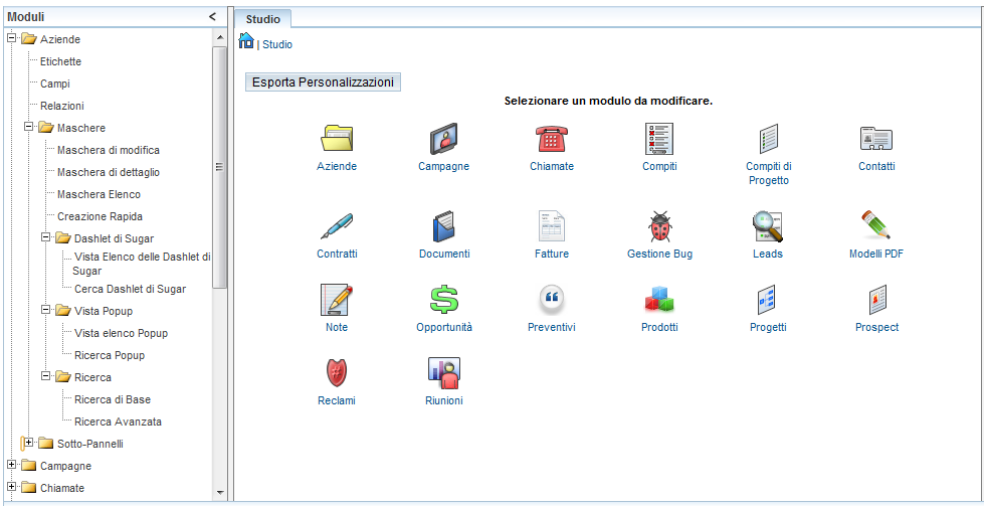

Figura 5.5: Schema Moduli Studio

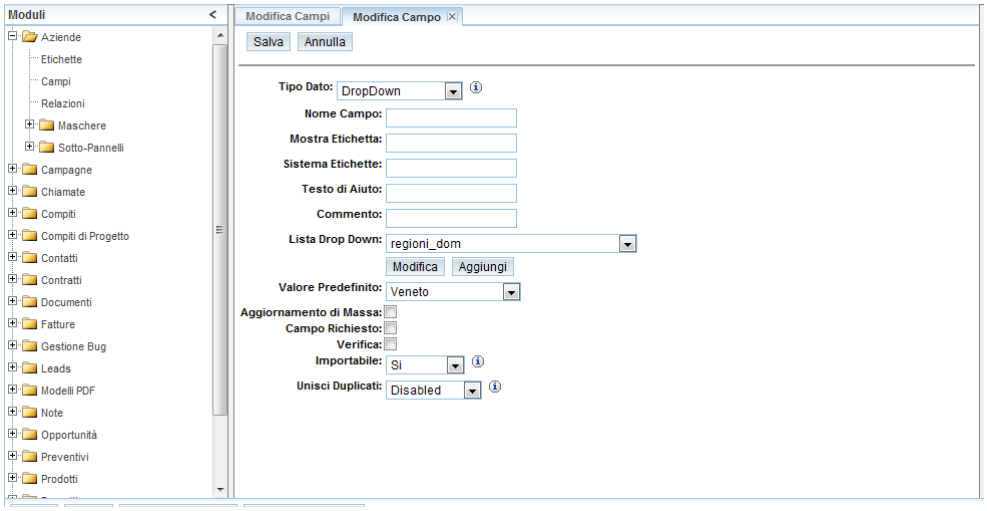

Figura 5.6: Creazione nuovo attributo con DropDown

        - di dettaglio
        - elenco
        - di creazione rapida
        - dashlet
            - vista elenco
            - cerca
        - vista Popup
            - vista elenco
            - cerca
        - Ricerca
            - di base
            - avanzata
    - Sottopannelli
        - altri moduli

In particolare per il nostro scopo prenderemo in considerazione solamente i Campi; da quì è possibile creare nuovi attributi per ogni modulo. La creazione consiste nel specificare inizialmente il tipo di attributo desiderato tra

Valuta

Indirizzo

Checkbox

Data

Decimale

Intero

DropDown

Selezione multipla

Telefono

Radio

Area di testo

e inserendo in seguito altre informazioni relative ad ogni tipo di dato (ad esempio la lunghezza massima del testo nell'area di testo). Si può inoltre inserire una selezione da una lista predefinita (DropDown): in questo caso bisogna prima creare una eventuale nuova lista desiderata e poi selezionarla nella creazione dell'attributo.

Ad esempio, nella creazione dell'attributo Regione per il modulo Azienda si procederà in questo modo:

1 si crea la lista region_list all'interno delle liste DropDown attraverso il modulo Studio di SugarCRM;

2 si crea l'attributo Regione_azienda sempre dal modulo Studio con le caratteristiche:

- **Tipo Dato:** DropDown
- **Nome Campo:** Regione_azienda
- **Mostra Etichetta:** Regione
- **Sistema Etichette:** Regione_azienda
- **Testo di Aiuto:** Regione dell'azienda
- **Commento:**
- **Lista Drop Down:** region_list
- **Valore Predefinito:**
- **Aggiornamento di Massa:** sì
- **Campo Richiesto:** no
- **Verifica:** no
- **Importabile:** sì
- **Unisci Duplicati:** disabilitato

3 infine si abilita il nuovo attributo anche nelle Maschere di interesse sempre attraverso il modulo Studio.

Il nuovo attributo sarà quindi creato secondo le specifiche di interesse e verrà inserito e gestito nel database completamente e in modo autonomo da SugarCRM.

5.4.2 Modulo delle Opportunità di lavoro

SugarCRM permette come abbiamo visto una personalizzazione dei moduli che lo compongono; è possibile però creare dei moduli nuovi, di esportarli e di importarli.

Per questo motivo spesso la community che utilizza il web service pubblica nuovi moduli già testati con cui gli utenti possono ampliare le offerte di SugarCRM e gestire più informazioni contemporaneamente.

Il progetto di nostro interesse prevede la creazione di un nuovo modulo per gestire e documentare le Opportunità di lavoro in modo da inserire nel CRM documenti riguardo ai preventivi che vengono fatti ai clienti. Sul web viene offerta l'installazione di un modulo chiamato Advanced Open Sales e che permette di gestire tutto ciò previsto dalle esigenze degli utenti e molto altro. In particolare gestisce:

- Preventivi (o Opportunità di vendita) assegnabili a ogni utente con diverse priorità e stati;

- Prodotti di vendita, ognuno associabile ad ogni preventivo e con caratteristiche varie come prezzo, costo, iva e altro;

- Fatture personalizzabili per ogni preventivo.

Per il progetto è sufficiente l'utilizzo dei preventivi e dei prodotti mentre per le fatture non è stata fatta alcuna gestione ma solamente una dimostrazione completa del modulo agli utenti nel caso volessero più avanti utilizzarlo in modo completo.

L'installazione del modulo è molto semplice: il file zip in cui viene compresso dai progettisti viene caricato su una sezione di SugarCRM chiamata Carica Moduli e installato mediante una procedura guidata. A fine installazione basterà fare il login e SugarCRM presenterà da subito tutte le nuove funzionalità senza nessuna configurazione ulteriore.

Infine per ciò che riguarda la comunicazione con il nuovo modulo da parte del client non è necessario alcun accorgimento in quanto le richieste SOAP e REST funzionano correttamente e nello stesso modo in cui funzionano per il resto.

5.5 Invio delle email di alert

La gestione delle notifiche è una richiesta di secondo piano da parte degli utenti e per questo è stata studiata nel periodo finale del progetto. Queste e-mail devono essere inviate nei casi di creazione e/o modifica di:

- chiamate
- note
- attività

e nello specifico per ognuna di queste possono essere scelti come destinatari delle notifiche:

- l'utente a cui è assegnata la chiamata, l'attività o la nota
- il responsabile dell'utente a cui è assegnata la chiamata, l'attività o la nota

La scelta di un destinatario non esclude comunque la scelta dell'altro e di default la scelta comprende entrambi nel caso di creazione e nessuno nel caso di modifica.

La difficoltà principale di invio di notifiche viene dal fatto che l'invio di e-mail avviene dal lato server mentre il progetto lavora nella maggior parte dal lato client. Dato che l'unica parte di esecuzione lato server è la risposta alle richieste REST la soluzione proposta consiste nell'integrare i servizi del web service inserendo una nuova chiamata che attraverso opportuni parametri crea e invia mail di notifica. Ogni chiamata che viene fatta al server passa dal file SugarWebServiceImpl.php che filtra il tipo di richiesta e chiamando le opportune classi esegue accessi al database restituendo le informazioni utili; in questo file viene inserita una nuova funzione Send_Mail() con i seguenti parametri:

- mittname, nome del mittente
- mittmail, indirizzo e-mail del mittente
- dest, indirizzo e-mail del destinatario
- subj, oggetto della e-mail
- body, corpo della e-mail (anche HTML)

Quando il client web farà una chiamata REST con i parametri opportuni la funzione creata costruirà la mail con una struttura rigida e provvederà a inviare la mail appoggiandosi alla classe PHP PHPMailer, uno script già pronto all'utilizzo e opensource.

La richiesta nel nostro caso restituirà o un messaggio di invio o un messaggio con l'errore riscontrato dal server; in base a questi valori compariranno degli alert all'utente per indicargli lo stato di invio delle notifiche.

La nuova funzione avrà il codice 5.6,

Codice 5.6: Funzione PHP Send_Mail()

```
function send_mail($mittname, $mittmail, $dest, $subj, $body){

        $messaggio = new PHPmailer();

        $messaggio->Host='server.host.it';

        $messaggio->IsHTML(true);

        $messaggio->From=$mittmail;
        $messaggio->FromName=$mittname;
        $messaggio->AddAddress($dest);
        $messaggio->Subject=$subj;
        $messaggio->Body=$body;


        if(!$messaggio->Send()){
                return $messaggio->ErrorInfo;
        }else{
          return 'Email inviata correttamente';
        }

        unset($messaggio);

}
```

dove oltre ai dati già citati vi sono altri paramtri come Host che rappresenta l'indirizzo del server host che invierà la mail e IsHTML indica con il valore True che il corpo dell'email (body) viene scritto in linguaggio HTML.

5.6 Gestione dell'Upload

L'upload deve essere gestito quando viene creata una nuova nota a cui si vuole allegare un documento; in questo caso infatti bisogna trasferire il file al server e trattarlo attraverso una nuova funzione REST di SugarCRM: la funzione set_note_attachment().

Come già detto la documentazione di queste funzioni è estremamente scarsa e dopo numerosi tentativi è risultato che tale funzione non esegue il trasferimento del file in modo corretto.

Un'altra soluzione consisteva nel trasferire in altri modi il file ma comportava una gestione diversa dell'Interfaccia Web causando perdite di sessione continue.

Dato che la gestione dell'upload è stata la parte meno importante del progetto e per questo è stata rimandata alla fine dello sviluppo si è deciso di non implementarla in questa versione del client.

Capitolo 6

Conclusioni

6.1 Punto di arrivo

La versione dell'applicazione web creata rispecchia quasi tutte le esigenze fornite dagli utenti finali con le limitazioni causate dalla metodologia di sviluppo. In particolare non è stato possibile integrare la gestione dell'upload inizialmente ipotizzata. Dopo aver fatto più prove di accesso e utilizzo attraverso un iPad 2 con connessione 3G abbiamo scoperto che il client è notevolmente più veloce di quello fornito da SugarCRM; il merito si può attribuire ad una gestione dei dati che rende le connessioni più efficienti e al framework jQuery Mobile che rende l'interfaccia semplice intuitiva e ben organizzata per l'utente finale. Sono state fatte ottimizzazioni anche a livello hardware spostanto il web service su un nuovo server più potente di quello dove risiedeva.

Anche dopo queste migliorie sono stati notati comunque dei rallentamenti che avvengono anche in rete locale e localizzati dopo molte ricerche sulla comunicazione REST: il servizio fornito via SOAP e REST da SugarCRM è infatti molto scarso e solo lunghe analisi l'hanno potuto evidenziare. Queste lentezza fortunatamente non causano blocchi eccessivi nel nuovo client che consente di operare comunque in velocità.

6.2 Sviluppi futuri

Gli sviluppi che possono essere proposti partono dal principale difetto che può esser rilevato nel client sviluppato: la comunicazione REST utilizzata crea blocchi in momenti casuali dello svolgimento delle operazioni. Si può quindi pensare di eliminare questo problema operando nel modo analago di SugarCRM ossia una gestione principale via PHP di tutte le richieste lato

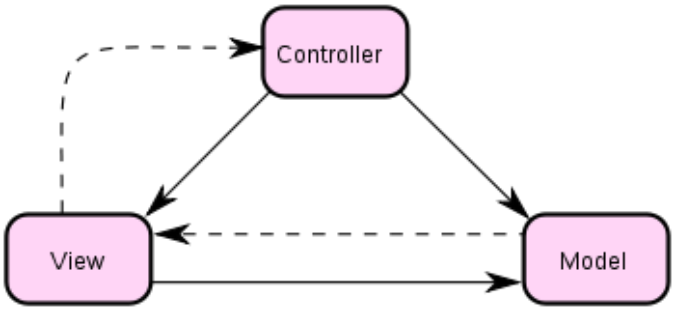

Figura 6.1: Schema MVC

server e una comunicazione univoca di tipo HTML; in questo caso le richieste a SugarCRM saranno fatte però direttamente al database con query SQL. Questo sviluppo non è particolarmente oneroso perché consente di riutilizzare tutto il lavoro fatto fin'ora attraverso una architettura MVC (Model-View-Controller) basata sulla separazione dei compiti fra i componenti software che interpretano tre ruoli principali:

- il model fornisce i metodi per accedere ai dati utili all'applicazione;
- il view visualizza i dati contenuti nel model e si occupa dell'interazione con utenti e agenti;
- il controller riceve i comandi dell'utente (in genere attraverso il view) e li attua modificando lo stato degli altri due componenti.

Questo schema, fra l'altro, implica anche la tradizionale separazione fra la logica applicativa a carico del controller e del model, e l'interfaccia utente a carico del view potendo inserire tutta la parte relativa all'interfaccia grafica scritta nell'index.html in questo modulo senza grosse modifiche.

Un altro punto a favore di questo tipo di sviluppo è infine la possibilità già accertata di gestire l'upload in maniera semplice ed efficiente.

Ringraziamenti

Desidero ringraziare l'Ing. Michele Moro per la disponibilità offerta durante il periodo di tirocinio e nella stesura della tesi e Marco Zuffolato, mio tutore aziendale, per avermi dato la possibilità dell'esperienza vissuta all'interno dell'azienda NSB.

Ringrazio la mia famiglia per avermi motivato e sostenuto in tutto in questi anni di studio.

Grazie anche a Giulia per avermi aiutato ad andar avanti e supportato nello studio nei momenti più difficili.

Un pensiero infine per tutte le persone che mi sono state accanto negli ambienti universitari con le quali ho potuto instaurare buoni rapporti di amicizia.